Elette Boyle
Mohammad Mahmoody (Eds.)

LNCS 15366

# Theory of Cryptography

22nd International Conference, TCC 2024
Milan, Italy, December 2–6, 2024
Proceedings, Part III

3 Part III

INTERNATIONAL ASSOCIATION FOR CRYPTOLOGIC RESEARCH

Springer

Elette Boyle · Mohammad Mahmoody
Editors

# Theory of Cryptography

22nd International Conference, TCC 2024
Milan, Italy, December 2–6, 2024
Proceedings, Part III

Springer

*Editors*
Elette Boyle
NTT Research
Sunnyvale, CA, USA

Reichman University
Herzliya, Israel

Mohammad Mahmoody
University of Virginia
Virginia, VA, USA

ISSN 0302-9743 ISSN 1611-3349 (electronic)
Lecture Notes in Computer Science
ISBN 978-3-031-78019-6 ISBN 978-3-031-78020-2 (eBook)
https://doi.org/10.1007/978-3-031-78020-2

# Preface

The 22nd Theory of Cryptography Conference (TCC 2024) was held during December 2–6, 2024, at Bocconi University in Milano, Italy. It was sponsored by the International Association for Cryptologic Research (IACR). The general chair of the conference was Emmanuela Orsini.

The conference received 172 submissions, of which the Program Committee (PC) selected 68 for presentation, giving an acceptance rate of 39.5%. Each submission was reviewed by at least three PC members in a single-blind process. The 50 PC members (including PC chairs), all top researchers in our field, were helped by 185 external reviewers, who were consulted when appropriate. These proceedings consist of the revised versions of the 68 accepted papers. The revisions were not reviewed, and the authors bear full responsibility for the content of their papers.

We are extremely grateful to Kevin McCurley for providing fast and reliable technical support for the HotCRP review software. We also thank Kay McKelly for her help with the conference website.

This was the tenth year that TCC presented the Test of Time Award to an outstanding paper that was published at TCC at least eight years ago, making a significant contribution to the theory of cryptography, preferably with influence also in other areas of cryptography, theory, and beyond. This year, the Test of Time Award Committee selected the following paper, published at TCC 2004: "Notions of Reducibility between Cryptographic Primitives," by Omer Reingold, Luca Trevisan, and Salil P. Vadhan. The award committee recognized this paper "for providing a rigorous and systematic taxonomy of reductions in cryptography, and in particular coining fully black-box reductions and motivating their use in barrier results."

We are greatly indebted to the many people who were involved in making TCC 2024 a success. Thank you to all the authors who submitted papers to the conference and to the PC members for their hard work, dedication, and diligence in reviewing and selecting the papers. We are also thankful to the external reviewers for their volunteered hard work and investment in reviewing papers and answering questions. Finally, thank you to the general chair Emmanuela Orsini and her team at Bocconi University, as well as to the TCC Steering Committee.

October 2024

Elette Boyle
Mohammad Mahmoody

# Organization

## General Chair

| | |
|---|---|
| Emmanuela Orsini | Bocconi University, Italy |

## Program Committee Chairs

| | |
|---|---|
| Elette Boyle | Reichman University, Israel & NTT Research, USA |
| Mohammad Mahmoody | University of Virginia, USA |

## Steering Committee

| | |
|---|---|
| Yuval Ishai | Technion, Israel |
| Huijia (Rachel) Lin | University of Washington, USA |
| Tal Malkin | Columbia University, USA |
| Jesper Buus Nielsen | Aarhus University, Denmark |
| Krzysztof Pietrzak | Institute of Science and Technology Austria, Austria |
| Manoj M. Prabhakaran | IIT Bombay, India |
| Salil Vadhan | Harvard University, USA |

## Program Committee

| | |
|---|---|
| Prabhanjan Ananth | UC Santa Barbara, USA |
| Benny Applebaum | Tel Aviv University, Israel |
| Amos Beimel | Ben-Gurion University of the Negev, Israel |
| Chris Brzuska | Aalto University, Finland |
| Yilei Chen | Tsinghua University, China |
| Ran Cohen | Reichman University, Israel |
| Geoffroy Couteau | CNRS, IRIF, Université Paris Cité, France |
| Itai Dinur | Ben-Gurion University of the Negev, Israel |
| Yevgeniy Dodis | New York University, USA |
| Stefan Dziembowski | University of Warsaw & IDEAS NCBR, Poland |
| Nils Fleischhacker | Ruhr University Bochum, Germany |

| | |
|---|---|
| Chaya Ganesh | Indian Institute of Science, Bangalore, India |
| Aarushi Goel | NTT Research, USA |
| Siyao Guo | NYU Shanghai, China |
| Mohammad Hajiabadi | University of Waterloo, Canada |
| Carmit Hazay | Bar-Ilan University, Israel |
| Justin Holmgren | NTT Research, USA |
| Aayush Jain | Carnegie Mellon University, USA |
| Zhengzhong Jin | Northeastern University, USA |
| Dakshita Khurana | University of Illinois at Urbana-Champaign, USA |
| Susumu Kiyoshima | NTT Social Informatics Laboratories, Japan |
| Lisa Kohl | CWI Amsterdam, Netherlands |
| Ilan Komargodski | Hebrew University of Jerusalem, Israel & NTT Research, USA |
| Eyal Kushilevitz | Technion, Israel |
| Huijia (Rachel) Lin | University of Washington, USA |
| Alex Lombardi | Princeton University, USA |
| Fermi Ma | Simons Institute and UC Berkeley, USA |
| Hemanta K. Maji | Purdue University, USA |
| Giulio Malavolta | Bocconi University, Italy & Max Planck Institute, Germany |
| Noam Mazor | Tel Aviv University, Israel |
| Pierre Meyer | Aarhus University, Denmark |
| Ryo Nishimaki | NTT Social Informatics Laboratories, Japan |
| Omer Paneth | Tel Aviv University, Israel |
| Krzysztof Pietrzak | Institute of Science and Technology Austria, Austria |
| Manoj Prabhakaran | IIT Bombay, India |
| Willy Quach | Weizmann Institute of Science, Israel |
| Divya Ravi | University of Amsterdam, Netherlands |
| Alon Rosen | Bocconi University, Italy and Reichman University, Israel |
| Lior Rotem | Stanford University, USA |
| Peter Scholl | Aarhus University, Denmark |
| Sruthi Sekar | IIT Bombay, India |
| Luisa Siniscalchi | Technical University of Denmark, Denmark |
| Eliad Tsfadia | Georgetown University, USA |
| Prashant Nalini Vasudevan | National University of Singapore, Singapore |
| Muthu Venkitasubramaniam | Georgetown University, USA |
| Mingyuan Wang | UC Berkeley, USA |
| Daniel Wichs | Northeastern University & NTT Research, USA |
| Takashi Yamakawa | NTT Social Informatics Laboratories, Japan |

## Additional Reviewers

Behzad Abdolmaleki
Anasuya Acharya
Amit Agarwal
Divesh Aggarwal
Andris Ambainis
Gilad Asharov
Thomas Attema
David Balbás
Laasya Bangalore
James Bartusek
Tyler Besselman
Rishabh Bhadauria
Kaartik Bhushan
Alexander Bienstock
Aniruddha Biswas
Alexander Block
Jeremiah Blocki
Katharina Boudgoust
Nicholas Brandt
Rares Buhai
Alper Cakan
Matteo Campanelli
Ran Canetti
Rutchathon Chairattana-Apirom
Benjamin Chan
Anirudh Chandramouli
Rohit Chatterjee
Megan Chen
Jessica Chen
Binyi Chen
Arka Rai Choudhuri
Sandro Coretti-Drayton
Quand Dao
Pratish Datta
Giovanni Deligios
Marian Dietz
Fangqi Dong
Nico Döttling
Ehsan Ebrahimi
Christoph Egger
Saroja Erabelli
Grzegorz Fabiański
Pooya Farshim
Giacomo Fenzi
Ben Fisch
Pouyan Forghani
Cody Freitag
Phillip Gajland
Karthik Gajulapalli
Rachit Garg
Sanjam Garg
Riddhi Ghosal
Satrajit Ghosh
Suparno Ghoshal
Niv Gilboa
Eli Goldin
Tian Gong
Junqing Gong
Jiaxin Guan
Aditya Gulati
Taiga Hiroka
Iftach Haitner
David Heath
Aditya Hegde
Hans Heum
Minki Hhan
Yao-ching Hsieh
Zihan Hu
Jihun Hwang
Yuval Ishai
Abhishek Jain
Daniel Jost
Eliran Kachlon
Fatih Kaleoglu
Chethan Kamath
Simon Kamp
Julia Kastner
Shuichi Katsumata
Hannah Keller
Hamidreza Amini Khorasgani
Taechan Kim
Elena Kirshanova
Ohad Klein
Karen Klein
Dimitris Kolonelos
Chelsea Komlo

Manu Kondapaneni
Venkata Koppula
Alexis Korb
Nishat Koti
Roman Langrehr
Seunghoon Lee
Keewoo Lee
Zeyong Li
Yunqi Li
Hanjun Li
Xiao Liang
Fuchun Lin
Chuanwei Lin
Haoxing Lin
Yao-Ting Lin
Tianren Liu
Jiahui Liu
Chen-Da Liu-Zhang
Zhenjian Lu
Donghang Lu
Vadim Lyubashevsky
Ulysse Léchine
Nir Magrafta
Bernardo Magri
Nathan Manohar
Xinyu Mao
Marcin Mielniczuk
Ethan Mook
Tomoyuki Morimae
Changrui Mu
Saachi Mutreja
Anne Müller
Varun Narayanan
Barak Nehoran
Ky Nguyen
Hai Hoang Nguyen
Guilhem Niot
Oded Nir
Aysan Nishaburi
Mahak Pancholi
Aditi Partap
Anat Paskin-Cherniavsky
Rutvik Patel
Shravani Patil
Sikhar Patranabis
Alice Pellet-Mary
Paola de Perthuis
Naty Peter
Spencer Peters
Bertram Poettering
Guru Vamsi Policharla
Alexander Poremba
Luowen Qian
Rajeev Raghunath
Debasish Ray Chawdhuri
Hanlin Ren
Doreen Riepel
Ron D. Rothblum
Adeline Roux-Langlois
Lawrence Roy
Elahe Sadeghi
Pratik Sarkar
Rahul Satish
Benjamin Schlosser
Akash Shah
Jad Silbak
Mark Simkin
Fabrizio Sisinni
Tomer Solomon
Fang Song
Katerina Sotiraki
Noah Stephens-Davidowitz
Gilad Stern
Björn Tackmann
Kel Zin Tan
Er-cheng Tang
Athina Terzoglou
Jean-Pierre Tillich
Pratyush Ranjan Tiwari
Daniel Tschudi
Prashant Vasudevan
Ivan Visconti
Benedikt Wagner
William Wang
Benjamin Wesolowski
Jiawei Wu
David Wu
Yu Xia
Zhiye Xie
Jeff Xu

# Contents – Part III

## Encryption

Multi-authority Functional Encryption with Bounded Collusions from Standard Assumptions . . . . 3
*Rishab Goyal and Saikumar Yadugiri*

Multi-client Attribute-Based and Predicate Encryption from Standard Assumptions . . . . 31
*David Pointcheval and Robert Schädlich*

Adaptively Secure Attribute-Based Encryption from Witness Encryption . . . . 65
*Brent Waters and Daniel Wichs*

Limits on Adaptive Security for Attribute-Based Encryption . . . . 91
*Zvika Brakerski and Stav Medina*

Tighter Adaptive IBEs and VRFs: Revisiting Waters' Artificial Abort . . . . 124
*Goichiro Hanaoka, Shuichi Katsumata, Kei Kimura, Kaoru Takemure, and Shota Yamada*

Distributed Broadcast Encryption from Lattices . . . . 156
*Jeffrey Champion and David J. Wu*

## Quantum II and Separations

Unclonable Commitments and Proofs . . . . 193
*Vipul Goyal, Giulio Malavolta, and Justin Raizes*

Unclonable Cryptography with Unbounded Collusions and Impossibility of Hyperefficient Shadow Tomography . . . . 225
*Alper Çakan and Vipul Goyal*

Quantum Key-Revocable Dual-Regev Encryption, Revisited . . . . 257
*Prabhanjan Ananth, Zihan Hu, and Zikuan Huang*

On Black-Box Separations of Quantum Digital Signatures from Pseudorandom States . . . . 289
*Andrea Coladangelo and Saachi Mutreja*

On the Black-Box Complexity of Private-Key Inner-Product Functional Encryption . . . . 318
*Mohammad Hajiabadi, Roman Langrehr, Adam O'Neill, and Mingyuan Wang*

**Authentication and Sequentiality**

Hide-and-Seek and the Non-resignability of the BUFF Transform . . . . 347
*Jelle Don, Serge Fehr, Yu-Hsuan Huang, Jyun-Jie Liao, and Patrick Struck*

From One-Time to Two-Round Reusable Multi-signatures Without Nested Forking . . . . 371
*Lior Rotem, Gil Segev, and Eylon Yogev*

Composability in Watermarking Schemes . . . . 400
*Jiahui Liu and Mark Zhandry*

Space-Lock Puzzles and Verifiable Space-Hard Functions from Root-Finding in Sparse Polynomials . . . . 431
*Nico Döttling, Jesko Dujmovic, and Antoine Joux*

Black-Box Timed Commitments from Time-Lock Puzzles . . . . 460
*Hamza Abusalah and Gennaro Avitabile*

**Author Index** . . . . 495

# Encryption

# Multi-authority Functional Encryption with Bounded Collusions from Standard Assumptions

Rishab Goyal and Saikumar Yadugiri(✉)

University of Wisconsin-Madison, Madison, WI 53706, USA
{rishab,saikumar}@cs.wisc.edu

**Abstract.** Multi-Authority Functional Encryption (MA-FE) [*Chase, TCC'07; Lewko-Waters, Eurocrypt'11; Brakerski et al., ITCS'17*] is a popular generalization of functional encryption (FE) with the central goal of decentralizing the trust assumption from a single central trusted key authority to a group of multiple, *independent and non-interacting*, key authorities. Over the last several decades, we have seen tremendous advances in new designs and constructions for FE supporting different function classes, from a variety of assumptions and with varying levels of security. Unfortunately, the same has not been replicated in the multi-authority setting. The current scope of MA-FE designs is rather limited, with positive results only known for certain attribute-based functionalities or from general-purpose code obfuscation. This state-of-the-art in MA-FE could be explained in part by the implication provided by Brakerski et al. (ITCS'17). It was shown that a general-purpose obfuscation scheme can be designed from any MA-FE scheme for circuits, even if the MA-FE scheme is secure only in a bounded-collusion model, where at most *two* keys per authority get corrupted.

In this work, we revisit the problem of MA-FE, and show that existing implication from MA-FE to obfuscation is not tight. We provide new methods to design MA-FE for circuits from simple and minimal cryptographic assumptions. Our main contributions are summarized below–

1. We design a $\mathsf{poly}(\lambda)$-authority MA-FE for circuits in the bounded-collusion model. Under the existence of public-key encryption, we prove it to be statically simulation-secure. Further, if we assume sub-exponential security of public-key encryption, then we prove it to be adaptively simulation-secure in the Random Oracle Model.
2. We design a $O(1)$-authority MA-FE for circuits in the bounded-collusion model. Under the existence of 2-party or 3-party non-interactive key exchange and public-key encryption, we prove it to be adaptively simulation-secure.
3. We provide a new generic bootstrapping compiler for MA-FE for general circuits to design a simulation-secure $(n_1 + n_2)$-authority MA-FE from any two $n_1$-authority and $n_2$-authority MA-FE.

R. Goyal—Support for this research was provided by OVCRGE at UW-Madison with funding from the Wisconsin Alumni Research Foundation.

## 1 Introduction

Functional encryption ($\mathsf{FE}$) [20,56] has revolutionized the study of public-key encryption [31]. It challenged the widespread belief that encryption is an *all-or-nothing* primitive, where you either learn the full plaintext or nothing. $\mathsf{FE}$ provides fine-grained access over encrypted data enabling recovery of only partial information about plaintext. Over the last several years, viewing through the $\mathsf{FE}$ lens has led to dramatic re-envisioning of encryption with varying levels of expressiveness such as identity-based encryption (IBE) [18,27,59], attribute-based encryption (ABE) [43,56], inner-product functional encryption (IPFE) [22, 49], 1-sided/2-sided predicate encryption (PE) [22,41,42,49,62] and more.

In the standard $\mathsf{FE}$ formulation, there is a central trusted authority that generates the global parameters $(\mathsf{MPK}, \mathsf{MSK})$, a master public-secret key pair. Using the master secret key $\mathsf{MSK}$, it can generate a decryption key $\mathsf{SK}_x$ for any attribute $x$, where $\mathsf{SK}_x$ enables decryption to $f(x)$ from any ciphertext $\mathsf{CT}$ encrypting a function $f$. Unfortunately, this trust model is too strong for many applications, as it implicitly embeds a key-escrow problem. To address this deficiency, functional encryption has been generalized and studied in far more weaker trust models. (See [2] for a detailed discussion on many such generalizations.)

*Decentralizing FE.* One of the most well adopted such generalizations of $\mathsf{FE}$ is multi-authority functional encryption ($\mathsf{MA\text{-}FE}$) [24–26,51]. $\mathsf{MA\text{-}FE}$ generalizes $\mathsf{FE}$ by decentralizing the trust to a group of multiple, *independent and non-interacting*, key authorities. Each key authority generates its own master public-secret key $(\mathsf{mpk}_i, \mathsf{msk}_i)$ *asynchronously* and completely oblivious of other authorities. From an end-user's perspective, $\mathsf{MA\text{-}FE}$ just looks like a regular (single-authority) $\mathsf{FE}$ system with master public key $\mathsf{MPK} = (\mathsf{mpk}_i)_i$, except it receives its decryption key $\mathsf{SK}_x$ in the form of $n$ disjoint partial keys $(\mathsf{sk}_{\mathsf{GID},i,x[i]})_i$ from $n$ different key authorities (where each authority just uses its own $\mathsf{msk}_i$ and $\mathsf{GID}$ is the user's global identifier to tie together partial keys coming from different authorities). In other words, each key authority controls the master key material for only a portion of the full encryption system. This decentralizes trust from one central authority to a group of $n$ individually operating authorities, while $\mathsf{GID}$ ensures partial keys issued to the same user by different authorities are "linked".

To model real-world threats while formalizing $\mathsf{MA\text{-}FE}$ security, the standard approach is to consider two types of corruptions – (1) *partial decryption key corruptions:* here an adversary gets a partial key $\mathsf{sk}_{\mathsf{GID},i,x[i]}$ for some authority $i$ and partial attribute $x[i]$ and identifier $\mathsf{GID}$, and (2) *key authority corruptions:* here an adversary gets a master key $\mathsf{msk}_i$ for authority $i$. Informally, $\mathsf{MA\text{-}FE}$ security states that an attacker cannot learn anything from a ciphertext $\mathsf{ct}$, encrypting function $f$, except from what it can learn by legitimately decrypting using valid combinations of the partial decryption keys and authority master keys it has.

*What Do We Know?* Although MA-FE was formally introduced almost a decade ago[1] [24], we have not seen significant progress in terms of new designs for MA-FE. And, this is not due to the lack of community-wide efforts. Since the original proposal of multi-authority attribute-based encryption (MA-ABE) by Chase [25] in 2007, numerous works have studied MA-FE for different classes, but the progress from simple assumptions has either been stuck at MA-ABE for monotone span programs [51], or its generalization to inner-product functionality [2]. Moreover, to the best of our knowledge, the only MA-FE construction that we currently have, that goes beyond such attribute-based functionalities, is the original obfuscation based construction by Brakerski et al. [24], who formally defined and designed MA-FE for arbitrary polynomial-time computations based on *sub-exponentially secure* indistinguishability obfuscation (iO) [37,57] and injective one-way functions.

The current state-of-the-art for MA-FE is embarrassingly dissatisfying! This is unlike (single-authority) FE, where we have seen tremendous progress over the years. Just in the bounded collusion model, we have had multiple FE constructions for general circuits for over a decade [3,4,9,35,36,39,55]. And, even in the fully collusion resistant setting, we have numerous advanced systems such as for quadratic functions [11,52], "degree 2.5" or partially hiding quadratic functions [6], attribute-based predicates and their variants [19,40,43], recently culminating in FE for general circuits from a combination of well-founded assumptions [46,47].

*Why is MA-FE this Hard?* Although the lack of progress towards new designs of MA-FE for general circuits suggests an inherent difficulty, there is a deeper underlying barrier that we discuss next. In addition to the MA-FE construction from (sub-exponential) iO, Brakerski et al. [24] provided a complementary result proving that any secure MA-FE scheme for general circuits implies an obfuscation scheme for general circuits. They proved that the implication to obfuscation follows as long as the MA-FE system is secure when either (i) one key authority gets corrupted, or (ii) two partial decryption keys can be corrupted for every key authority. This two-sided implication between MA-FE and obfuscation highlights the technical barrier in designing new MA-FE constructions.

At this point, it seems quite convincing that breaking new ground in the context of MA-FE suffers from the same barriers that we have for code obfuscation [12]. Moreover, the implication by Brakerski et al. [24] follows even when an attacker just learns 2 partial decryption keys per authority (i.e., total of $2n$ partial keys with $n$ authorities). Thus, it appears that even designing MA-FE in the bounded collusion model [33,38,55] faces the same strong barriers as we have for code obfuscation. For example, this points to the impossibility of simulation-secure MA-FE for general circuits in the standard model.

*Is this Implication Tight? Does Bounded-Collusion MA-FE Really Give Obfuscation?* Recall the MA-FE-based obfuscation construction by Brakerski et al. [24].

[1] Its predecessor, multi-authority attribute-based encryption (MA-ABE) [25,26,51], was proposed nearly two decades ago.

To obfuscate an $n$-bit circuit $C$, an obfuscator samples $n$ $\mathsf{MA\text{-}FE}$ master key pairs $(\mathsf{mpk}_i, \mathsf{msk}_i)$, and it encrypts the circuit $C$ under the joint master public key $\mathsf{MPK} = (\mathsf{mpk}_i)_i$ to create an encrypted circuit $\mathsf{Enc}(\mathsf{MPK}, C)$. Now to enable circuit evaluation, the obfuscator generates $2n$ partial decryption keys $(\mathsf{sk}_{\mathsf{GID},i,b})_{i,b}$. That is, for authority $i$, it generates two partial keys for both attribute bits, 0 and 1. The obfuscated circuit contains the encrypted circuit $\mathsf{Enc}(\mathsf{MPK}, C)$ and $2n$ partial secret keys $(\mathsf{sk}_{\mathsf{GID},i,b})_{i,b}$. An evaluator picks half of the keys $(\mathsf{sk}_{\mathsf{GID},i,x[i]})_i$, depending upon its $n$-bit input $x$, and uses them to decrypt the $\mathsf{MA\text{-}FE}$ ciphertext. Correctness of obfuscation follows from $\mathsf{MA\text{-}FE}$ correctness, and as long as $\mathsf{MA\text{-}FE}$ is secure even when two partial keys per key authority can be corrupted, then security of obfuscation follows. Brakerski et al. [24] used the above argument (and its extensions[2]) to argue necessity of obfuscation.

While this might seem tight, a closer inspection reveals a fundamental issue. The claim that $\mathsf{MA\text{-}FE}$ is as hard as code obfuscation, even when a bounded number of secret keys get corrupted, is not true! This is because in the above obfuscation construction, the obfuscator generates **two** partial secret keys for the same $\mathsf{GID}$ for every authority. In other words, it generates two partial keys for same $\mathsf{GID}$ with *distinct* attribute bits. At first this seems a benign thing to do, but this conflicts with the classical motivation and application scenario of $\mathsf{MA\text{-}FE}$. As explained by Chase, Chow, Lewko, and Waters [25,26,51], the notion of a per-user global identifier was introduced to avoid "mix-and-match" attacks. To expand on this, since the key authorities are completely decentralized and working asynchronously, and each authority only gives a partial key for its portion of the attribute, then what prevents two users from combining their partial secret keys!? To avoid this in the multi-authority regime, each user is associated with a public global identifier $\mathsf{GID}$ (which may or may not contain information[3] about its full attribute $x$). This ensures that only partial keys generated for the same $\mathsf{GID}$ can be used together, preventing mix-and-match attacks.

The summary of above discussion is that, in any typical application scenario, an authority does not need to generate *more than one* partial key for a single $\mathsf{GID}$. Thus, it seems most reasonable to consider attackers that receive only **one** partial key per authority for every unique $\mathsf{GID}$. This is the standard approach in the vast MA-ABE literature too [2,5,25,26,28–30,35,50,51,53,54,60]. However, to design obfuscation from $\mathsf{MA\text{-}FE}$, we need security when at least **two** partial keys per authority (for some $\mathsf{GID}$) have been corrupted, or an attacker can corrupt key authorities themselves. Due to this mismatch, it is unclear whether the

[2] The extension was to consider authority corruptions instead, and that could reduce $n$, the number of key authorities, to just 1 while maintaining the implication to obfuscation.

[3] The motivation behind not including attribute $x$ in the clear in identifier $\mathsf{GID}$ is to get some hiding property about a user's attribute. Since each key authority only learns a portion of the user's attribute along with the public user identifier, thus it provides attribute privacy from a malicious key authority.

implication from MA-FE to obfuscation still holds when an attacker can only corrupt a fixed number of identifiers $(\mathsf{GID}_q)_q$.

*Our Results.* In this work, we provide new methods to design MA-FE for general circuits from simple and minimal assumptions. All our results are in the bounded-collusion model [39,55], naturally generalized to the multi-authority setting [35, 60]. We summarize our main results below.

1. For all polynomials $n = n(\lambda), Q = Q(\lambda)$, we design a *statically*[4]-secure $n$-authority MA-FE for general circuits secure, as long as at most $Q$ user get corrupted, under the minimal assumption of PKE.
2. Next, we show two rather interesting and incomparable approaches to boost security of our MA-FE to full adaptive security.
   (a) First, we show that under the additional assumption of an $n$-party non-interactive key exchange (niKE) [21,23,32,48], we can improve our design to an *adaptively* secure $n$-authority MA-FE for general circuits with $Q$-corruptions.
   (b) Second, we show that by assuming sub-exponential hardness of PKE, we can improve our design to an *adaptively* secure $n$-authority MA-FE for general circuits with $Q$-corruptions in the Random Oracle Model (ROM) [14].
3. Lastly, we also provide a new generic bootstrapping compiler for MA-FE for general circuits. We show that any adaptively secure $n$-authority MA-FE for general circuits with $Q$-corruptions can be generically upgraded to an adaptively secure $2n$-authority MA-FE for general circuits with $Q$-corruptions.

All our MA-FE schemes are proven to be simulation-secure. And, by plugging in 2/3-party key exchange protocols based on DDH hard groups/pairing-friendly groups, we obtain an adaptively secure 2/3-authority MA-FE for general circuits based on DDH or standard bilinear pairing assumptions. Further, by applying our generic bootstrapping compiler, we can design an *adaptively* secure $O(1)$-authority MA-FE for general circuits in the bounded-collusion model, under DDH or standard pairing assumptions. Alternatively, we can also design an *adaptively* secure $\mathsf{poly}(\lambda)$-authority MA-FE for general circuits in the bounded-collusion model from regular public-key encryption, in the Random Oracle Model.

*Related Work.* Fully collusion resistant FE with indistinguishability-based security is known to be (nearly) equivalent to iO [7,8,16,37], while simulation-based security is known to make the object impossible [1]. However, simulation-secure FE with bounded collusion resistance [33,38,55] is achievable [3,4,9,35,36,39,55] from minimal assumptions. In the multi-authority setting, numerous constructions for fully collusion resistant MA-ABE have been designed in the past decade; see [2,5,25,26,28–30,50,51,53,54] and the references there in. While in the bounded collusion setting, [35,60] have designed MA-ABE for monotone boolean formulae and circuits (respectively) from minimal assumptions.

[4] As we discuss later, by static security we mean that an attacker declares all secret key queries at the beginning of the security game. Actually this construction is secure in a slightly stronger model (as we elaborate in the technical overview), but for simplicity we just state it to be statically secure for the purposes of this introduction.

## 2 Technical Overview

In this section, we provide a high level overview of our techniques, and summarize the key ideas.

**Reviewing $\mathsf{MA\text{-}FE}$.** A multi-authority functional encryption scheme contains four[5] algorithms – $\mathsf{AuthSetup}$, $\mathsf{KeyGen}$, $\mathsf{Enc}$, $\mathsf{Dec}$. The authority setup algorithm, $\mathsf{AuthSetup}$, is used by an authority to create its public-secret key pair $(\mathsf{mpk}_{\mathsf{id}}, \mathsf{msk}_{\mathsf{id}})$, where $\mathsf{id}$ denotes its identity/index. Consider there are $n$ total authorities with public keys $\mathsf{mpk}_1, \ldots, \mathsf{mpk}_n$. All $n$ keys jointly are regarded as the master public key $\mathsf{MPK}$ for the full system. The special feature of $\mathsf{MA\text{-}FE}$ is that any authority can use its secret key $\mathsf{msk}_{\mathsf{id}}$ to create a partial secret key $\mathsf{sk}_{\mathsf{GID},\mathsf{id},x_{\mathsf{id}}}$ for some identifier $\mathsf{GID}$ and attribute bit $x_{\mathsf{id}}$[6]. An encryptor takes the full public key $\mathsf{MPK}$ and encrypts a circuit $C$, such that any user with identifier $\mathsf{GID}$ and partial keys $\mathsf{sk}_{\mathsf{GID},1,x_1}, \ldots, \mathsf{sk}_{\mathsf{GID},n,x_n}$, can decrypt to learn $C(x_1, \ldots, x_n)$. $\mathsf{MA\text{-}FE}$ schemes are very useful because they do not require interaction between authorities and users to generate keys or compute ciphertexts.

To formally capture adversarial corruptions, one can consider attackers that can corrupt key authorities as well as partial secret keys. However, Brakerski et al. [24] showed that $\mathsf{MA\text{-}FE}$, secure in presence of a single corrupt authority, is as powerful as obfuscation. Thus, we only study key corruptions and not study authority corruptions in the sequel. A bit more formally, we consider attackers that receive master public keys $(\mathsf{mpk}_{\mathsf{id}})_{\mathsf{id}}$ for all authorities, and can make key generation queries to any authority, where it must not submit more than one key query for a given $\mathsf{GID}$ to any key authority. That is, the attacker can submit as many key queries as it wants, as long as, for any $\mathsf{GID}$, it does not obtain more than one key per authority[7].

In this work, we consider simulation-based security for $\mathsf{MA\text{-}FE}$ with bounded collusions. In this setting, the adversary must a-priori commit to a collusion bound $Q$, where $Q$ denotes the maximum number of unique $\mathsf{GID}$[8] it queries to any single authority in the security game. Throughout the paper, we refer to this as $Q$-$\mathsf{GID}$ corruption model. Simulation security states that no PPT distinguisher can distinguish between a simulated transcript and an honestly generated

[5] Technically, $\mathsf{MA\text{-}FE}$ schemes also have a global setup algorithm that generates global public parameters. Typically, this is just some common random string that can be computed easily in the ROM, or by any party without compromising system security. Thus, we ignore this detail in this overview.

[6] For simplicity, we consider each authority controls a single attribute bit. This can be generically extended to longer attributes.

[7] As discussed earlier, this closely models the real world scenarios, where any user with $\mathsf{GID}$ can only receive a key for a single attribute bit from any authority. This is fairly common approach in many prior works [2,25,26,51].

[8] In our constructions, we denote $Q$ to be the total number of queries the adversary makes across $n$ authorities. However, for the ease of exposition, we denote $Q$ to be the number of unique $\mathsf{GID}$s the adversary queries throughout the overview. This would only alter the query bound by a factor of $n$ in the actual security game.

transcript[9]. And, the simulated transcript must only reveal $\{(\mathsf{GID}, x_{\mathsf{GID}}, C(x_{\mathsf{GID}})) : \text{for every unique } \mathsf{GID}\}$, where $x_{\mathsf{GID}}$ denotes the input string obtained by appropriately concatenating attribute bits queried to each authority for identifier $\mathsf{GID}$.

## 2.1 Step 1: Adaptively Secure MA-FE with 1-GID Corruption

We start by designing MA-FE that is secure as long as only 1 key per authority gets corrupted, i.e. in the 1-GID corruption model. Later, we show a bootstrapping approach to design MA-FE secure in the $Q$-GID corruption model, for any polynomial $Q$. To begin, let us keep our focus on proving non-adaptive security, where the attacker has to make all key queries before receiving challenge ciphertext.

**Non-adaptive 1-GID MA-FE.** Our starting point is the Sahai-Seyalioglu (single-authority) FE construction [55]. A brief overview of this construction is as follows.

**Setup:** Sample $2n$ public and secret key pairs $(\mathsf{PK}_{i,b}, \mathsf{SK}_{i,b})_{i \in [n], b \in \{0,1\}}$.
**Key Generation:** Given $x \in \{0,1\}^n$, Output $n$ secret keys $(\mathsf{SK}_{i,x_i})_{i \in [n]}$.
**Encryption:** Given a circuit $C$, garble the circuit and encrypt the $2n$ wire labels accordingly with $\mathsf{PK}_{i,b}$.

Surprisingly, this construction already satisfies the multi-authority criterion. There is an inherent divisibility property where each secret key for the bit $x_i$ and each encryption for the $i$-th wire labels are independent of other bits and wire labels. We can easily transform this construction into a multi-authority version where each authority samples 2 public and secret key pair corresponding to the attribute 0 and 1. When the authority is queried for an attribute $b$, we simply output $\mathsf{SK}_{i,b}$. During decryption, secret key from each authority reveals one wire label each.

For security, we crucially rely on the garbled circuit's security where security is guaranteed if at most half the wire labels are revealed. If an adversary corrupts more than one secret key from an authority or corrupt an authority, this scheme will no longer be secure. But the security for this scheme fits perfectly into our 1-GID version. If the adversary only queries with one unique GID and only queries each authority once, the security of the scheme follows from the security of SS10 construction.

However, even in this 1-GID corruption setting, this is not adaptively secure. Unless the challenger knows the value of $C(x) = C(x_1, \ldots, x_n)$, we can't leverage the security of the garbling scheme. If an adversary queries keys for attributes $x_i$ in an adaptive fashion, we cannot rely on the garbling scheme's security during encryption. Hence, this scheme is only non-adaptively secure. In order to boost the security of this non-adaptively secure 1-GID MA-FE scheme, we look at the

[9] By transcript, we mean the full interaction transcript between the challenger and attacker.

transformation of non-adaptively secure one-key $\mathsf{FE}$ to adaptively secure one-key $\mathsf{FE}$ from GVW12. The key idea used in this transformation is a non-committing encryption scheme ($\mathsf{NCE}$). A $\mathsf{NCE}$ scheme is a public-key encryption scheme with the additional property that we can "fake" a ciphertext and later "reveal" it to be encryption of the message $m$ by tailoring the secret key for $m$.

*How to Use* $\mathsf{NCE}$*?* As a first attempt, let us naively plug-in $\mathsf{NCE}$ in place of $\mathsf{PKE}$ in the construction of the non-adaptive version. Although this lets us fake the wire labels for adaptive queries, we still can't get the wire labels as we don't see $x$ as a whole. Diving a bit further, for any non-adaptive key corruption query, we can encrypt and decrypt the wire labels similar to the $\mathsf{PKE}$ version. And for any adaptive key corruption queries, we can fake the encryption of wire labels and reveal them later when we receive the query. However, in order to generate these wire labels by relying on garbling scheme's security, we require the input $x$ as a whole by the time of encryption. So, we still have the same issue we did when we were using $\mathsf{PKE}$.

It looks like using $\mathsf{NCE}$ is still not enough for boosting the non-adaptive 1-$\mathsf{GID}$ scheme's security. The issue is that in order to leverage non-adaptive 1-$\mathsf{GID}$ scheme's security, we need to make sure that only after all the authorities are queried, we can compute the ciphertext. Furthermore, this issue is exacerbated because of authorities who can reveal one wire label each independently of other authorities. If each authority has a say in decryption of all the wire labels, then wire labels can be revealed only after all the authorities are queried. Then, we can rely on the non-adaptive security of SS10 construction. This is the high level approach we follow.

*Using* $\mathsf{NCE}$ *Smartly.* In order to bypass the above issue, we employ a novel way of combining $\mathsf{NCE}$, additive secret sharing, and hybrid encryption. Let $\mathsf{ct}_{\mathsf{id}}$ denote the encryption of $\mathsf{id}$-th wire labels of the garbled circuit for $C$, i.e., $\mathsf{ct}_{\mathsf{id}} = ($ garbled circuit, $\{\mathsf{Enc}(\mathsf{PK}_{\mathsf{id},b}, w_{\mathsf{id},b})\}_b)$. We sample random strings $R_1, \ldots, R_n$ of the same length as $\mathsf{ct}_{\mathsf{id}}$, and encrypt the wire labels for the $\mathsf{id}$-th input as $(\mathsf{NCE}_1.\mathsf{Enc}(R_1), \ldots, \mathsf{NCE}_n.\mathsf{Enc}(R_n), \bigoplus_{\mathsf{id}\in[n]} R_{\mathsf{id}} \oplus \mathsf{ct}_{\mathsf{id}})$. This can be simply viewed as additively secret sharing the one-time key used in hybrid encryption. In the key generation algorithm, along with $\mathsf{SK}_{\mathsf{id},x_{\mathsf{id}}}$, each authority provides a secret key for $\mathsf{NCE}_{\mathsf{id}}$. Using these secret keys, each authority has equal contribution in the decryption $\mathsf{ct}_{\mathsf{id}}$. For instance, using keys from authority $\mathsf{id} = 1$, we can only decrypt $R_1$. As $R_2, \ldots, R_n$ are encrypted, $\mathsf{ct}_{\mathsf{id}}$ remains perfectly hidden. Only after all the authorities provide their secret keys, $\mathsf{ct}_{\mathsf{id}}$ will be revealed in the clear.

Now, we can argue adaptive security for this scheme using non-adaptive security of the SS10-based construction. Assume that an adversary only queries the $n$-th authority in the adaptive manner. We can provide the encryption for $\mathsf{ct}_{\mathsf{id}}$ as $(\mathsf{NCE}_1.\mathsf{Enc}(R_1), \ldots, \mathsf{NCE}_{n-1}.\mathsf{Enc}(R_{n-1}), \mathsf{NCE}_n.\mathsf{Fake}, \widetilde{R})$. Even with $R_1, \ldots R_{n-1}$ revealed from non-adaptive queries, no information about $\mathsf{ct}_{\mathsf{id}}$ is revealed. When the $n$-th authority is queried, we see the whole $x$ and calculate $(\mathsf{ct}_{\mathsf{id}})_{\mathsf{id}\in[n]}$. We provide the secret key for the $n$-th authority as $(\mathsf{SK}_{n,x_n}, \mathsf{NCE}.\mathsf{Reveal}(\widehat{R}))$ where

$\widehat{R} = \widetilde{R} \oplus \bigoplus_{\mathsf{id} \in [n-1]} R_{\mathsf{id}} \oplus \mathsf{ct}_{\mathsf{id}}$. We need to do this for all $\mathsf{ct}_{\mathsf{id}}$ in the full construction. In order to understand our full construction, it helps to think about it in the following stages.

– During encryption, we calculate an $n \times n$ matrix which consists of NCE ciphertexts.
– We add the sum of the id-th *row*'s plaintexts to $\mathsf{ct}_{\mathsf{id}}$.
– Using the secret keys from id-th authority, the id-th *column* of this matrix is decrypted.

Only after all the columns are decrypted, the row values for each id-th row will be revealed and thus, $\mathsf{ct}_{\mathsf{id}}$ is revealed. A brief overview of the construction is as follows.

**Authority Setup:** For the id-th authority, Generate $(\mathsf{PK}_{\mathsf{id},b}, \mathsf{SK}_{\mathsf{id},b})$ for $b \in \{0,1\}$ and for each $i \in [n]$, generate the pair $(\mathsf{NCE.pk}_{\mathsf{id},i}, \mathsf{NCE.sk}_{\mathsf{id},i})$.

**Encryption:** Generate the garbled circuit and encrypt the wire labels similar to SS10 construction. Sample the $n \times n$ matrix $\mathbf{M} = [\mathsf{NCE.Enc}(R_{\mathsf{id},i})]_{\mathsf{id} \in [n], i \in [n]}$ and set $R_{\mathsf{id}} = \bigoplus_{i \in [n]} R_{\mathsf{id},i}$. Output (id-th row of $\mathbf{M}$, $R_{\mathsf{id}} \oplus \mathsf{ct}_{\mathsf{id}}$) for each id.

**Key Generation:** Given $x_{\mathsf{id}} \in \{0,1\}$, output $\mathsf{SK}_{\mathsf{id},x_{\mathsf{id}}}$ and reveal the secret keys for the id-th *column*.

The above idea gives way to our basic result about MA-FE.

**Theorem 1 (Informal).** *Assuming the existence of* NCE *and garbled circuits (both implied by public-key encryption), there exists a* 1-GID MA-FE *scheme for* P/Poly *circuits.*

For our construction of 1-GID MA-FE schemes, we only need a basic notion of NCE that is readily provided by public-key encryption [39,44,45]. For more details about this result and security analysis, please check the full version of our paper.

## 2.2 Step 2: Upgrading 1-GID MA-FE to $Q$-GID MA-FE

In the previous section, we gave an overview of the construction of an adaptively secure 1-GID MA-FE scheme with an overarching goal of boosting this scheme to a $Q$-GID MA-FE scheme using the techniques present in GVW12, AV19. It would only make sense to take a closer look at the techniques used in these papers and figure out any bottlenecks we might face in generalizing them to the multi-authority setting. We provide an overview of the techniques used in AV19 to construct a bounded functional encryption scheme (BFE) for P/Poly circuits. Specifically, we provide the overview of correlated garbling, client-server framework, and how the security of these schemes pave the way for a BFE scheme. We present the major hurdles present in generalizing this approach as opposed to the natural generalization of SS10 as seen in the previous section. We provide potential ways to overcome these hurdles and the approaches we took in this paper.

**Overview of $\mathsf{BFE}$ Constructions from Minimal Assumptions.** One of the core insights in GVW12,AV19 can be summarized as follows: an FE scheme, $\mathsf{BFE}$, secure under $Q$ collusions can be viewed as a very particular combination of $\mathsf{poly}(Q)$ FE schemes, $\mathsf{1FE}$, secure under a 'single' collusion. The core idea was that by using randomized encodings from [10] and specific multi-party computation protocols inspired by [13,15], one can tie together such $\mathsf{poly}(Q)$ instantiations of $\mathsf{1FE}$ such that, by employing some statistical combinatorial arguments, we can argue that attacking the outer $\mathsf{BFE}$ scheme is as hard as attacking at least one $\mathsf{1FE}$ scheme. In other words, the multi-party computation protocol is built using special randomized encodings which ensure that there is enough redundancy in the system such that even if a small subset of the users become corrupted (explained later), the protocol remains computationally secure. And, if we can generate such a protocol with two rounds, we can play this protocol in-the-head and use each round's messages in the encryption and key generation algorithm to construct $\mathsf{BFE}$.

Diving a bit further, AV19 constructs a multi-party computation protocol, that they called client-server framework ($\mathsf{csf}$). This protocol is constructed using specialized randomized encoding schemes and leverages certain statistical arguments. In short, it is a neatly tied package of all the complications required to construct $\mathsf{BFE}$. This $\mathsf{csf}$ protocol along with $\mathsf{1FE}$ paved the way to construct $\mathsf{BFE}$. Hence, we provide a brief overview as follows.

*The Client-Server Framework.* In this framework[10], there is a *server* with a circuits $C_1, \ldots, C_Q$ and a *client* with an input $x$. Both these parties want to compute the value of $C_1(x), \ldots, C_Q(x)$ together. The security guarantee requires that no information about $x$ should be leaked to any party in the protocol apart from what is leaked from $\{(C_q, C_q(x)) : q \in [Q]\}$. Note that this already resembles the simulator definition for a bounded functional encryption scheme. In order to achieve this task, the client and server perform $Q$ rounds of computation where they delegate the computation to $N$ *users*. The protocol proceeds in three phases described as follows.

**Offline Phase:** The client equipped with the information about the number of rounds $Q$ and the maximum size of the circuits $s$, encodes their input into $(\widehat{x}^1, \ldots, \widehat{x}^N)$ using $\mathsf{ClientEnc}(1^\lambda, 1^Q, 1^s, x)$ procedure and sends the $u$-th encoding $\widehat{x}^u$ to the $u$-th user.

**Online Phase:** This phase is executed for $Q$ rounds. In each round, the server encodes the $q$-th circuit $C_q$ into $(\widehat{C}_q^1, \ldots, \widehat{C}_q^N)$ using $\mathsf{ServEnc}(1^\lambda, 1^Q, 1^s, C_q)$ procedure with the $u$-th encoding $\widehat{C}_q^u$ intended for the $u$-th user. However, in each round, not all the users are utilized. Only a subset $\mathbf{S}_q \subset [N]$ for

[10] We made few modifications to the actual protocol present in AV19 to fit our narrative.

users are invoked in the $q$-th round and each user $u \in \mathbf{S}_q$, performs $\widehat{y}_q^u \leftarrow \mathsf{UserComp}(\widehat{C}_q^u, \widehat{x}^u)$ to obtain the $u$-th encoding of the output, $y_q$.

**Decoding Phase:** This phase is executed for $Q$ rounds. The output encodings and the subset of users $\mathbf{S}_q$ are published to the client who can recover the output $y_q$ using $\mathsf{Decode}(\mathbf{S}_q, \{\widehat{y}_q^u\}_{u \in \mathbf{S}_q})$.

To utilize this construction for a bounded functional encryption, it is crucial that the offline and online phases do not share any state. Abstractly, the bounded functional encryption (BFE) scheme that uses csf proceeds as follows.

**Encryption:** Compute the offline phase messages $(\widehat{x}^1, \ldots, \widehat{x}^N)$ from csf. Encrypt each encoding using an adaptively secure one-key functional encryption scheme (1FE). Output the $N$ ciphertexts.

**Key Generation:** Compute the online phase messages $(\widehat{C}^1, \ldots, \widehat{C}^N)$ from csf. Sample random $\mathbf{S} \subset [N]$ and for each $u \in \mathbf{S}$, generate a key for $\mathsf{UserComp}(\widehat{C}^u, \cdot)$ using 1FE. Output $\mathbf{S}$ and these secret keys.

The decryption algorithm performs 1FE decryption to reveal the output encodings as in the decoding phase of csf and thus retrieve $y$. It is not hard to see that if 1FE is adaptively secure and by the csf security, we get a bounded functional encryption. We denote this construction that uses 1FE and csf as the BFE blueprint.

*Security of* csf*.* The security definition of csf is tailored for BFE. An adversary against the csf scheme sends the number of rounds $Q$ at the beginning of the game and proceeds to ask for at most $Q$ server encodings and output encodings from users. Adversary sends at most one input $x$ and does not receive the full client encoding. The adversary also gets to chooses the subset of users $\mathbf{S}_q$ for each of the $Q$ rounds adaptively. But this might result in a user being invoked twice in two different rounds and thus the corresponding 1FE instantiation will be rendered insecure. Hence, a user that is invoked twice is deemed to be corrupted and for such a corrupted user, the client encodings $\widehat{x}^u$ for the corrupted users, $u \in \mathbf{S}_{\mathsf{corr}}$ are revealed to the adversary. The adversary also sends the set of non-corrupted users $\widetilde{\mathbf{S}} = [N] \setminus \mathbf{S}_{\mathsf{corr}}$ at the beginning of the security game. Hence, the transcript of the communication to the adversary which includes input encodings of the corrupted users, the server encodings and the output encodings for each $q \in [Q]$ shouldn't reveal any more information about $x$ than what is revealed by $\{(C_q, C_q(x)) : q \in [Q]\}$.

csf *for* $\mathsf{NC}^1$ *Circuits.* GVW12 constructed a csf scheme for all $\mathsf{NC}^1$ policies using the celebrated Shamir secret sharing scheme [58]. For the online phase, using a $(N, t)$-Shamir secret sharing scheme, sample shares for all $N$ users for the secret input $x$. In the offline phase, the server uses $C_q$ as the server encoding for the circuit $C_q$. As public-key BFE does not guarantee function-hiding, it is fine to reveal the circuit $C_q$. The user computation is evaluation of $C_q$ for each round on $\widehat{x}^u$. That is $\widehat{y}^u = C_q(\widehat{x}^u)$. Note that correctness of the scheme relies on $C_q$ being used as a polynomial and thus will only work for $\mathsf{NC}^1$ circuits. As long as the subset of users invoked in each round, $\mathbf{S}_q$ are more than $t$, we can guarantee the correctness of the scheme.

Security of the scheme relies on Shamir secret sharing. However, in order to instantiate Shamir secret sharing, we need information about the number of users that can potentially corrupted, i.e., $t$. Recall that a user $u \in [N]$ is corrupted if there are $q, q' \in [Q]$ such that $u \in \mathbf{S}_q \cap \mathbf{S}_{q'}$. If we can bound the size of the set $\cup_{q \neq q'} \mathbf{S}_q \cap \mathbf{S}_{q'}$ by $t$, Shamir secret sharing will inherently provide us the security guarantee we are looking for. That is we instantiate secret sharing by $(N, t)$, at most $t$ users are corrupted, and thus at most $t$ client encodings $\widehat{x}^u$ are leaked. In order to achieve this, GVW12 rely on a special statistical lemma when number of rounds $Q$ is known in advance, called *small pairwise intersection lemma* which can be informally stated as follows.

**Lemma 1 ([39], informal).** *For specific $N = N(Q, \lambda), t = t(Q, \lambda)$, and $|\mathbf{S}_q| = D = D(Q, \lambda), D \geq t, \mathbf{S}_q \subseteq [N]$, the size of the set $\cup_{q \neq q'} (\mathbf{S}_q \cap \mathbf{S}_{q'})$ is at most $t$ for randomly sampled $\mathbf{S}_1, \ldots, \mathbf{S}_Q$.*

Although the $\mathsf{csf}$ as described above looks secure, there is a subtle issue that renders the scheme insecure. As in each round $|\mathbf{S}_q| > t$, the polynomial used in Shamir secret sharing is completely revealed by $\{\widehat{y}_q^u\}_{u \in \mathbf{S}_q}$. That is, anyone can evaluate this polynomial at any point as we have more than $t$ values of the polynomial and in particular, we can find $\widehat{y}_q^u$ for $u \notin \mathbf{S}_q$. In order to fix this, GVW12 adds randomness to the output of each user computation. Once again, if the randomness is fixed, the scheme would suffer from similar issues. It is required that for each round, we need "unique" randomness to mask $\widehat{y}_q^u$. GVW12 uses another statistical lemma called *cover-freeness lemma* which can be stated as follows.

**Lemma 2 ([39], informal).** *For specific $T = T(Q, \lambda)$ and $|\Delta_q| = v = v(Q, \lambda), \Delta_q \subseteq [T]$, there exists at least one unique $j_q^* \in \Delta_q$ for each randomly sampled $\Delta_1, \ldots, \Delta_Q$.*

By utilizing this lemma, as part of the offline encodings, along with $x$, the client shares the $T$ shares of 0, $\{\zeta_j^u\}_{u \in [N], j \in [T]}$ using $(N, D-1)$-Shamir secret sharing. In each round, the samples a random cover-free set $\Delta_q \subseteq [T]$, $|\Delta_q| = v$ and sends the encoding as $(C_q, \Delta_q)$. The users in each round set $\widehat{y}_q^u = C_q(\widehat{x}^u) + \sum_{j \in \Delta_q} \zeta_j^u$. Due to the unique index $j_q^*$, $\widehat{y}_q^u$ is random in each round and the protocol satisfies the required security definition. This way GVW12 realized a $\mathsf{csf}$ protocol for $\mathsf{NC}^1$ circuits and thus $\mathsf{BFE}$ for $\mathsf{NC}^1$ circuits using the $\mathsf{BFE}$ blueprint from minimal assumptions.

*Upgrading* $\mathsf{csf}$ *to* $\mathsf{P/Poly}$ *Circuits.* In order to upgrade $\mathsf{csf}$ from $\mathsf{NC}^1$ to any general circuit, it would be sufficient to represent a $\mathsf{P/Poly}$ circuit as a polynomial. However, this might not be possible for any $\mathsf{P/Poly}$ circuit. GVW12 utilized randomized encodings of Applebaum-Ishai-Kushilevitz, AIK06 [10] that uses $\mathsf{PRG}$s in $\mathsf{NC}^1$ to encode any $\mathsf{P/Poly}$ circuit as an $\mathsf{NC}^0$ circuit with locality 4. This way any $\mathsf{P/Poly}$ circuit can be represented as a polynomial. $\mathsf{PRG}$s in $\mathsf{NC}^1$ are known from various standard assumptions like $\mathsf{LWE}$, factoring, etc. However, it is not guaranteed that the minimal assumption of one-way functions will yield $\mathsf{PRG}$s in $\mathsf{NC}^1$. In order to resolve this, AV19 proposed a specialized reusable garbling scheme for $\mathsf{P/Poly}$ circuits in which the garbled circuit is an $\mathsf{NC}^0$ circuit with locality 4. They called it *correlated garbling* ($\mathsf{CorrGarb}$). It is easy to see that once we have such a garbling scheme, we can randomly encode any $\mathsf{P/Poly}$ circuit into a polynomial and use in $\mathsf{csf}$ for $\mathsf{NC}^1$ circuits readily to realize a $\mathsf{BFE}$ scheme for $\mathsf{P/Poly}$ circuits. Hence, the last missing piece of the $\mathsf{BFE}$-puzzle is $\mathsf{CorrGarb}$ and we provide its overview as follows.

*Overview of* $\mathsf{CorrGarb}$. At a high-level, $\mathsf{CorrGarb}$ crucially relies on the fact that $\mathsf{csf}$ is required only for an a-priori bounded $Q$ number of rounds and generalizes the randomized encodings provided by AIK06. To understand $\mathsf{CorrGarb}$ better, it would be helpful to look at Yao's garbling scheme [63] for $\mathsf{P/Poly}$ circuits. In this garbling scheme, we encode the wires using random strings known as "wire labels". We sample two wire labels for each bit value the wire can have. We encode each gate of a circuit[11] using double secret-key encryption scheme as follows:

$$\mathsf{SK\text{-}Enc}\,(\mathsf{SK} = \text{wire 2's label}, \mathsf{outer\text{-}msg})$$
$$\text{where } \mathsf{outer\text{-}msg} = \mathsf{SK\text{-}Enc}\left(\mathsf{SK} = \text{wire 1's label}, \mathsf{inner\text{-}msg} = \begin{array}{c}\text{labels for}\\ \underline{\text{two}}\text{ output wires}\end{array}\right)$$

If you have the right wire labels for the input wires of a gate, you can retrieve the output wire labels from $\mathsf{inner\text{-}msg}$ and the process proceeds for each gate in the circuit. If we replace the secret-key encryption using one-time pad, we get a perfect randomized encoding. However, as wire labels for each layer shrinks by half, we can use this procedure to encode $\mathsf{NC}^1$ circuits.

In $\mathsf{CorrGarb}$, the main idea (building on AIK06) is to set the wire labels to random length-expanding $\mathsf{PRG}$ seeds and use a "computational" one-time pad, i.e., $\mathsf{PRG}$(seed)$\oplus$ ($\underline{\text{two}}$ output seeds). AIK06 used to similar procedure but required $\mathsf{PRG}$s in $\mathsf{NC}^1$ to keep the encoded output in $\mathsf{NC}^0$. AV19 used *any* length-expanding $\mathsf{PRG}$ and gave the randomized encoding algorithm the outputs of the $\mathsf{PRG}$ evaluation so they can readily be used. This way, we can skip the computational one-time pad from AIK06 and instead simply use these $\mathsf{PRG}$ values readily. The decoding algorithm receives just the $\mathsf{PRG}$ seeds for the input wires and performs the computational one-time pad process to reveal the output wire labels and the process continues for each gate.

However, these encodings are only useful once. As the $\mathsf{PRG}$ seeds are leaked in evaluation, we cannot rely on the same seeds and $\mathsf{PRG}$ security to compute the encodings for $C_1, \ldots, C_Q$. In order to overcome this, AV19 relied on Lemma 2 and used $T$ distinct $\mathsf{PRG}$ seeds as each wire's label. The computational one-time pad is performed as follows.

[11] w.l.o.g, assume that each gate is a two fan-in, two fan-out NAND gate.

$$\bigoplus_{j \in \Delta_q} \mathsf{PRG}(j\text{-th input wire label}) \oplus \begin{pmatrix} \text{labels for } \underline{\text{two}} \text{ output wires} \\ \text{for each } j \in \Delta_q \end{pmatrix}$$

Note that the encoding process is deterministic given $\Delta_q$. The security flows naturally because of the $j_q^*$-th PRG seed in each round. This way, AV19 constructed a deterministic garbling scheme which encodes any P/Poly circuit into an $\mathsf{NC}^0$ circuit with locality 4 (this is by extension of AIK06 result) in which each encoding is correlated with each other. In conclusion, using CorrGarb, Lemmas 1 and 2, and csf, GVW12 and AV19 constructed BFE from minimal assumptions.

**MA-FE** Using **BFE Blueprint?** Ideally, we would like to follow the BFE blueprint to develop a $Q$-GID MA-FE scheme. However, several technical issues arise while using this blueprint. Mainly these issues arise because of the independence between $n$ authorities which is crucial in any multi-authority setting. The csf protocol, as described, is constructed for a single authority. We need a "distributional" version of csf where $n$ independent authorities can come up with a valid encoding for a circuit $C$. Even if we overcome this hurdle, we have another subtle issue to deal with. In the key generation of BFE, the subset of users $\mathbf{S}$ is sampled randomly. If we are in a system which uses $n$ independent authorities, with only negligible probability, all of these authorities agree upon the same $\mathbf{S}$. Because of these two issues, we can't readily use BFE blueprint to build a $Q$-GID MA-FE scheme.

*A New Client-Server Framework.* In order to overcome the first issue, we propose our distributed client-server framework (dCSF) that, simply put, guarantees that $n$ independent servers can come up with a valid server encoding for a circuit $C$. Our construction uses two key insights from csf. The server encoding algorithm of csf randomly samples a cover-free set $\Delta$ and outputs $(C, \Delta)$ as the $u$-the server encoding of $C$ for each $u \in [N]$. *What if we receive the cover-free set $\Delta$ from an external source with a guarantee that they are randomly generated?* In such a scenario, we can look at the server encoding algorithm as a deterministic encoding that is a conglomeration of $|C|$-many independent encodings. That is, we can look at this deterministic variant of server encoding as an aggregation of encodings from $|C|$ servers each providing $(C_i, \Delta)$ for $i \in [|C|]$ (moreover, each of these servers provide deterministic encodings too).

Using this idea, we can split the single server encoding to a distributional variant in which there are $|C|$ servers which can work independently on their own bit. However, this way of splitting a server is not quite natural. Recall that GVW12 relied on splitting the circuit description to construct a key-policy 1FE scheme using the ciphertext-policy variant of SS10. In our work, we consider ciphertext-policy MA-FE and use the input $x$ in the KeyGen algorithm. The switch between ciphertext-policy and key-policy FE can be made using the universal circuit $\mathcal{U}$. Coincidentally, the garbling algorithm of correlated garbling, is performed for the circuit $\mathcal{U}$. Hence, we can generically use this algorithm for our ciphertext-policy MA-FE scheme. Using these insights, we construct a dCSF protocol in which the servers possess the string $x$ and the client possesses the circuit $C$.

The major difference between our $\mathsf{dCSF}$ protocol and $\mathsf{csf}$ is in the online phase. In this phase, for each round, the $\mathsf{id}$-th server is invoked for some $\mathsf{id} \in [n]$. The $\mathsf{id}$-th server, encodes the input $(\mathsf{GID}, x_{\mathsf{id}}, \varDelta)$ into $\{\widehat{x}^u_{\mathsf{GID},\mathsf{id}}\}_{u\in[N]}$ using the $\mathsf{ServEnc}$ procedure. We then provide these server encodings to a subset of users $\mathbf{S} \subset [N]$. We require that for a specific $\mathsf{GID}$, $(\mathbf{S}, \varDelta)$ have to be uniform across all the server queries. The $u$-th output encoding $\widehat{y}^u_{\mathsf{GID}}$ is computed *if and only if* all the sever encodings are received from the $n$ independent servers. In other words, only if each $u \in \mathbf{S}$ receives $\{\widehat{x}^u_{\mathsf{GID},\mathsf{id}}\}_{\mathsf{id}\in[n]}$, does the $u$-th user compute $\widehat{y}^u_{\mathsf{GID}} \leftarrow \mathsf{UserComp}(\{\widehat{x}^u_{\mathsf{GID},\mathsf{id}}\}_{\mathsf{id}\in[n]}, \widehat{C}^u)$. The offline and decoding phases are similar to $\mathsf{csf}$.

*Security and Construction of* $\mathsf{dCSF}$*.* Our security definition is non-trivial and deviates from $\mathsf{csf}$'s version. A succinct and sanitized definition is as follows—similar to $\mathsf{csf}$ security, we allow the adversary to choose the subset users $\mathbf{S}$ each round. In addition, the adversary will also send the cover-free set $\varDelta$. $(\mathbf{S}, \varDelta)$ are such that they are same across all queries that use the same $\mathsf{GID}$. Looking ahead, in our constructions for $\mathsf{MA\text{-}FE}$ schemes, we obtain these sets using the $\mathsf{GID}$ information provided by the adversary. The adversary is not allowed to corrupt the client and sends the set of non-corrupted users $\widetilde{\mathbf{S}}$ at the beginning of the security game. We construct our $\mathsf{dCSF}$ scheme using a variation of $\mathsf{CorrGarb}$ in the full version of our paper. This variation is a weakening of $\mathsf{CorrGarb}$ present in AV19 and can be derived naturally from their version. Using such a $\mathsf{dCSF}$ scheme, coupled with a one-key multi-authority scheme, we can follow the $\mathsf{BFE}$ blueprint and create a $Q$-$\mathsf{GID}$ $\mathsf{MA\text{-}FE}$ scheme.

While the above ideas make some progress towards generalizing the single-authority bounded-collusion FE toolkit, we still have *not* navigated around the issue that $n$ independent authorities must agree upon the subset of users $\mathbf{S}$ and cover-free sets $\varDelta$. This is a significant technical hurdle, and it is a fundamental problem that does not exist in the single-authority setting! Thus, we need new technical ideas to resolve this. In the rest of the paper, we provide different technical approaches to resolve this. Each technical choice leads to a different level of security and has its own strengths and limitations as we discuss next. Below we explain our first approach to bypass this. It can be viewed as a stepping stone for the rest of our constructions.

*The* $\mathsf{MA\text{-}FE}$ *Sampling Bottleneck.* Let us start by sketching our candidate design for $Q$-$\mathsf{GID}$ $\mathsf{MA\text{-}FE}$ based on adaptive 1-$\mathsf{GID}$ $\mathsf{MA\text{-}FE}$ scheme ($\mathsf{1MAFE}$) and an adaptive $\mathsf{dCSF}$ protocol. Following AV19, the idea is to proceed as follows.

**Authority Setup:** Sample $N$ $\mathsf{1MAFE}$ instantiations.

**Encryption:** Generate client encodings of $C$ and encrypt the $u$-th user computation circuit with the $u$-th client encoding hardwired using the $u$-th $\mathsf{1MAFE}$ instantiation. Output all $N$ ciphertexts.

**Key Generation:** For the $\mathsf{id}$-th authority with input $\mathsf{GID}, x_{\mathsf{id}}$, sample random $(\mathbf{S}, \varDelta)$ and generate the $\mathsf{id}$-th server encodings and keys for $u$-th server encoding with the $u$-th $\mathsf{1MAFE}$ instantiation. Output $\mathbf{S}$, secret keys for $u \in \mathbf{S}$.

As mentioned earlier, the main issue is that $\mathsf{BFE}$ blueprint only works if all $n$ independent authorities come up with the same subset of users $\mathbf{S}$ and the cover-free set $\varDelta$. By sampling randomly, $(\mathbf{S}_{\mathsf{id}}, \varDelta_{\mathsf{id}})$ for $\mathsf{id} \in [n]$ will be same with only negligible probability. If they are not the same, it is not clear whether the small subset of users $\mathbf{S}_{\mathsf{id}} \subset [N]$ will even have an overlap. Thus, we cannot use one of the core building blocks of the blueprint, small pairwise intersection lemma. What we ideally require is a modified blueprint for $\mathsf{MA\text{-}FE}$ schemes in which during key generation, $(\mathbf{S}, \varDelta)$ are sampled using $\mathsf{GID}$ *such that every authority selects the same value, and yet they look random!* We refer to this as the $\mathsf{MA\text{-}FE}$ sampling bottleneck.

*A* $\mathsf{PRF}$ *to the Rescue?* So, the main question is: how do we sample $(\mathbf{S}, \varDelta)$ using $\mathsf{GID}$? One simple thought is to use a (deterministic) pseudorandom function ($\mathsf{PRF}$). That is, sample a key $K$ for a $\mathsf{PRF}$ that on input $\mathsf{GID}$, deterministically samples $(\mathbf{S}, \varDelta)$, and pass it on to each of the authorities. But how do we accomplish this? We need to pass a piece of information between $n$ independent authorities. To this end, we work in the common random string ($\mathsf{crs}$) model, and treat this key as the $\mathsf{crs}$. We remark that the $\mathsf{crs}$ should just be viewed as the global public parameters. In prior works, information about bilinear pairing groups, LWE modulus, etc., was treated as the $\mathsf{crs}$. In our work, we proceed similarly where we view the $\mathsf{PRF}$ key as the $\mathsf{crs}$.

While a $\mathsf{PRF}$ might seem to get us around the $\mathsf{MA\text{-}FE}$ sampling bottleneck, this generates some additional issues. As the $\mathsf{PRF}$ key $K$ is part of the $\mathsf{crs}$, in the security game, the adversary obtains this key $K$. In such a scenario, we can't rely on $\mathsf{PRF}$ security anymore. In order to terraform the realm of $Q$-$\mathsf{GID}$ $\mathsf{MA\text{-}FE}$, let us construct a scheme which satisfies a much weaker notion of security, we define as static-$Q$-$\mathsf{GID}$ security. In this notion, an adversary needs to query all the pre-challenge and post-challenge queries for $Q$ unique $\mathsf{GID}$s *before receiving the* $\mathsf{crs}$ *from challenger.* This way, we can sample $(\mathbf{S}_q, \varDelta_q)_{q \in [Q]}$ unique $\mathsf{GID}$ queries before the key $K$ is sent to the adversary.

This still leaves an issue with the statistical lemmas from GVW12 which should be argued about pseudorandom strings. The small pairwise intersection lemma and cover-freeness lemma are defined for uniformly random strings and they might not hold for pseudorandom strings. We remark that this issue does not stop us from realizing static-$Q$-$\mathsf{GID}$ security. The idea is that if the statistical lemmas do not hold for pseudorandom strings, then we can construct a polynomial-time distinguisher for $\mathsf{PRF}$ scheme. The distinguisher simply queries the oracle for $(\mathbf{S}, \varDelta)$ with $Q$ unique $\mathsf{GID}$s and checks whether these lemmas hold for the received sets. If it doesn't, distinguisher outputs $\mathsf{PRF}$, otherwise random oracle. If the statistical lemmas do not hold for pseudorandom strings, the advantage of this distinguisher is non-negligible, which breaks the $\mathsf{PRF}$ security.

Hence, we can rely on the appropriately strengthened versions of the statistical lemmas which are used to construct $\mathsf{BFE}$ for pseudorandom strings and rely on PRF's security against computational adversaries to build a static-$Q$-GID variation of MA-FE. This gives us the following result.

**Theorem 2 (Informal).** *Assuming the existence of a secure* PRF, 1-GID MA-FE *for* P/Poly *circuits, there exists a static-$Q$-*GID MA-FE *for* P/Poly *circuits.*

The scheme as mentioned uses the BFE blueprint and in the key generation phase, samples $(\mathbf{S}, \Delta)$ using a PRF whose key is provided as part of crs. The security of this scheme is easy to visualize as all the secret key generations for all the authorities occur at once. To develop some intuition, one can view all the $n$ authorities as a single aggregated authority which reuses $(\mathbf{S}, \Delta)$ for a specific GID. The remaining hybrid arguments can then follow the structure of AV19 closely. Although, this is not exactly how the security proof works, because we can't just assume all key generators to be a single party. But, we state it this way to give some high level intuition about the security proof of the remaining construction without opening up the box which would include a lot of unnecessary technical details. At a high level, our assertion is that once we clear out the issue of consistent randomness $(\mathbf{S}, \Delta)$, we can naturally generalize the core proof insights from the single-authority case (BFE), and make it work for the multi-authority setting. For more details into the construction and security proof, please refer to the full version of our paper.

## 2.3 Step 3: Towards Adaptive $Q$-GID MA-FE from Static Security

In the previous subsection, we saw how to construct $Q$-GID MA-FE in a weaker security setting which we defined as static-$Q$-GID security. However, it would be ideal to construct adaptively secure $Q$-GID MA-FE scheme using standard assumptions. This is not a overzealous goal as adaptive constructions for BFE are possible from minimal assumptions. In order to realize adaptive security, we need to overcome the MA-FE sampling bottleneck. In this section, we provide an overview of the two approaches which we used in this work to construct adaptive $Q$-GID MA-FE schemes. In approach 1, we explain the non-interactive key exchange scheme, the motivation and its placement in the context of MA-FE. In approach 2, we look at careful complexity leveraging arguments in ROM. In particular, we explore the nuances in guessing the minimal information required for overcoming the MA-FE sampling bottleneck and how sub-exponential security helps in this context.

**Approach 1: Non-interactive Key Exchange.** MA-FE *with Trusted Oracle.* In the static-$Q$-GID secure construction, we needed the secret key queries to be provided in a static manner (before crs is generated) because we can't rely on the security of PRF if the key is revealed to the adversary as part of crs. As a hypothetical argument, consider a trusted oracle $(\mathcal{O})$ which is only accessible by the authorities. If such $\mathcal{O}$ exists, we can instantiate it using $\mathsf{PRF}(K, \cdot)$ and each authority can query this oracle for the sets $(\mathbf{S}, \Delta)$.

As a result, all the authorities sample consistent $(\mathbf{S}, \varDelta)$ and we can rely on PRF security. It looks like having such an oracle solves all of our issues in constructing a $Q$-GID MA-FE scheme using the BFE blueprint. Unfortunately, such an oracle is not an ideal assumption to rely on. Recall that we are trying to move away from a central authority in the multi-authority setting. Moreover, assuming that the adversary cannot access $\mathcal{O}$ is also an unrealistic assumption.

However, observe that we do not need such a strong oracle. Let us weaken our assumption a bit and say that the key $K$ is only available to authorities and not the adversary. This is also a good enough assumption for realizing adaptive security using BFE blueprint. As $K$ is not revealed to the adversary, we can rely on PRF security and use BFE blueprint readily to construct an adaptively secure MA-FE scheme. So, *is this assumption realistic?* In other words, is there a cryptographic object that allows $n$ independent parties to compute a key $K$ such that the adversary remains none the wiser? The answer surprisingly is YES! The object we need to instantiate this assumption is non-interactive key exchange[12] (niKE) for $n$ parties [21,32,48].

*Reviewing* niKE *for $n$ Parties.* In a niKE scheme for $n$ parties (in the crs model), each party computes a public and secret value pair and posts their public values on a bulletin board. For any $\mathsf{id} \in [n]$, the $\mathsf{id}$-th party uses its secret value and public values from all the parties to generate a key $K$. The correctness of niKE requires that each party should be able to derive the same key $K$ using its secret value.

For instance, consider the simplest case of $n = 2$ and the classic Diffie-Hellman key exchange protocol [32]. Party $A$ randomly generates the secret value $a$ and the public value $g^a$, where $g$ is a generator of group $\mathbb{G}$. Similarly, party $B$ randomly generates a secret value $b$ and the public value $g^b$. Both $A$ and $B$ post their public values $g^a$, $g^b$ respectively to a bulletin board. Party $A$ computes using $a, g^a, g^b$, the key $g^{ab}$. Party $B$ computes using $b, g^a, g^b$, the key $g^{ab}$. As both the keys are same, this is a correct niKE scheme for 2 parties. A niKE scheme for 3 parties is present in [48] which is constructed using bilinear pairings.

The security of a niKE scheme guarantees that any adversary that does not possess secret value of any party, cannot distinguish between a randomly generated key and the key that honest parties can compute. Case in point, assuming the Decisional Diffie-Hellman assumption (DDH), the niKE scheme for 2 parties as defined above is secure as for any PPT adversary, $\{\mathbb{G}, g, g^a, g^b, g^{ab}\} \approx_c \{\mathbb{G}, g, g^a, g^b, g^c\}$, for random $a, b, c$. Similarly assuming the Bilinear Decisional Diffie-Hellman assumption (BDDH), the niKE scheme for 3 parties from [48] is also secure.

*Replacing Trusted Oracle $\mathcal{O}$ with* niKE. A brief overview of construction of MA-FE from niKE is as follows.

[12] As we consider no authority corruptions, we consider a statically secure scheme with no corruptions.

**CRS Generation:** Generate the $n$ party niKE's crs. Embed this information in crs.

**Authority Setup:** Generate $N$ 1MAFE master public and secret key pairs. Generate public and secret values for id-th party in the niKE scheme.

**Key Generation:** For the id-th authority with input $\mathsf{GID}, x_{\mathsf{id}}$, sample $(\mathbf{S}, \varDelta)$ using PRF scheme with $K$ generated from niKE. Generate the id-th server encodings and keys for $u$-th server encoding with the $u$-th 1MAFE instantiation. Output $\mathbf{S}$, secret keys for $u \in \mathbf{S}$.

As $K$ is indistinguishable from a randomly generated key $K$, it is only with negligible probability that an adversary could guess the key $K$ correctly. Because of this, we can rely on PRF security in the security game and handle all the queries from adversary, adaptively. We have to be careful in reproving our statistical lemmas, and we provide detailed information about the construction and the security proof in the full version of our paper. As mentioned, we rely on the niKE security and PRF security to randomly sample the sets $(\mathbf{S}, \varDelta)$ statically in the security game. After that, the security is argued by appropriately combining our proof strategy for the static-$Q$-GID security model. This gives us the following result.

**Theorem 3 (Informal).** *Assuming the existence of $n$ party* niKE*, a secure* PRF*, adaptively secure* 1-GID MA-FE*, there exists an adaptively secure $Q$-*GID MA-FE *scheme for* P/Poly *circuits.*

We have the following corollary for the theorem,

**Corollary 1.** *Assuming polynomial hardness of* DDH *(or* BDDH*), there exists a 2-authority (or 3-authority) $Q$-*GID MA-FE *for* P/Poly *circuits.*

The usage of niKE for $n$ parties is an interesting approach on its own and we think that this is an interesting avenue of research which is even more intriguing in the corrupted authorities model. To the best of our knowledge, this is the first time an $n$ party niKE scheme is utilized in the context of bounded collusion functional encryption for general circuits.

**Approach 2: Complexity Leveraging in Random Oracle Model.** In this section, we will look at an alternate approach to solve the MA-FE sampling bottleneck by using random oracles. We provide an overview of a specific notion of stronger security that our static-$Q$-GID construction generically satisfies in ROM. We call this *partial* adaptive security. We also provide a brief insight into the complexity leveraging argument that can used to boost any partial adaptively secure $Q$-GID MA-FE scheme to adaptively secure $Q$-GID scheme.

*Why ROM?* In the static-$Q$-GID construction and throughout approach 1, we relied on a deterministic PRF scheme to solve the MA-FE sampling bottleneck. At its core, this bottleneck is the issue of sampling shared randomness among independent parties. However, there is another way to generate shared randomness for the authorities. We can use a hash function $\mathcal{H}$ modelled as a random oracle for the system and work in the Random Oracle Model [14]! The motivation behind this approach is twofold:

1. One issue with deterministic $\mathsf{PRF}$s is that if the key is lost, we cannot rely on its security to build our schemes. This is precisely why we ended up using $\mathsf{niKE}$ for adaptive security.
2. Currently, $\mathsf{niKE}$ for $n > 3$ is only known to be facilitated by strong assumptions such as Multilinear Maps or iO [21,23]. Ideally, we would like to build an adaptively secure scheme which can support any $n = \mathsf{poly}(\lambda)$ authorities for $\mathsf{P}/\mathsf{Poly}$ circuits from minimal/standard assumptions.

*Issues with* $Q$-$\mathsf{GID}$ $\mathsf{MA\text{-}FE}$ *in ROM.* Note that the construction in ROM is similar to that of static-$Q$-$\mathsf{GID}$ $\mathsf{MA\text{-}FE}$. Except now, we use a hash function which takes $\mathsf{GID}$ as an input and deterministically samples $(\mathbf{S}, \Delta)$ and provide it as to all the users of the system. In the security proof however, there is an issue with the number of strings that adversary receives.

In the random oracle model, the adversary can make arbitrary number of queries (say $P$) to $\mathcal{H}$ and receive an arbitrary number of strings. $P$ can be much larger than $Q$. In that case, the Lemmas 1 and 2 with the parameter regime as presented in GVW12 do not hold. This is because they do not guarantee that the lemmas will hold for any subset of size $Q$ of a larger set of size $P$. We demonstrate a slight technical fix for these issues. Crudely speaking, we multiply each parameter of small pairwise intersection and cover-freeness lemmas by $Q$. The idea is that, if in the Chernoff's bound argument of these lemmas, we increase the exponent by a factor of $Q$, then by a union bound on the number of subsets of $P$ of size $Q$, the lemma still holds[13]. We refer the reader to the full version of our paper for a detailed overview and proof of these augmented lemmas.

*Partial Adaptive* $\mathsf{MA\text{-}FE}$ *in ROM.* Now that as we do not have any $\mathsf{PRF}$ key $K$ and $\mathcal{H}$ is modelled as a random oracle, there is no need for the adversary to make the queries in a static manner. In other words, the construction in ROM already satisfies a stronger security than static-$Q$-$\mathsf{GID}$ security. This construction *does not* satisfy adaptive security yet. In order to understand why, we need to understand a subtle technical issue in the security proofs of GVW12, AV19. In both of these constructions, as $(\mathbf{S}, \Delta)$ as randomly sampled *and as adversary only receives these after the challenger samples it*, they are able to sample these sets statically in the security game. An added benefit of this approach is that the challenger can determine which users are going to be corrupted[14]. However, working in the random oracle model, this is not feasible as the adversary can

[13] As the number of such subsets are bounded by $P^Q$.

[14] Recall that a user $u$ is corrupted if it is invoked in two different rounds of $\mathsf{csf}$. As we can sample sets $\mathbf{S}$ early in [9,39], we can determine the corrupted users well in advance.

always query the result of a $\mathsf{GID}$ adaptively to $\mathcal{H}$ first and then to the challenger. *What if the adversary submits all the* $\mathsf{GID}$*s for which the adversary will query secret keys for statically, i.e., before receiving the* $\mathsf{crs}$ *(and hence,* $\mathcal{H}$*) from the challenger?*

In such a scenario, using a programmable $\mathcal{H}$, we can sample the responses to these $\mathsf{GID}$ queries before hand and determine the set of non-corrupted users which is required to argue the security of $\mathsf{dCSF}$. So, it looks like our construction for static-$Q$-$\mathsf{GID}$ when we replace the $\mathsf{PRF}$ function with $\mathcal{H}$ satisfies a stronger security where we only require the unique $\mathsf{GID}$s that the adversary is going to query for secret keys from authorities statically. Moreover, we can handle $n = \mathsf{poly}(\lambda)$ authorities in this construction and depend upon minimal assumptions. We denote this security definition as *partial* adaptive security. The security of the scheme follows from the $\mathsf{BFE}$ blueprint. As we can compute the set of non-corrupted users, we can rely on the adaptive security of $\mathsf{dCSF}$ and the adaptive security of $\mathsf{1MAFE}$ to realize partial adaptive security of this scheme. We refer the reader to the full version of our paper for more details on the construction and security analysis.

*Final Step: Complexity Leveraging for Adaptive Security.* Partial adaptive security is still *not* adaptive security. The only thing standing in between this construction and an adaptively secure version is that we crucially require to get all the $Q$ unique $\mathsf{GID}$s that the adversary might query. *Can we guess the* $\mathsf{GID}$ *queries and incur a small loss in the reduction advantage?* As these $|\mathsf{GID}| = \mathsf{poly}(\lambda)$, when we guess all the $Q$ $\mathsf{GID}$ strings, we will incur an exponential loss. Nevertheless, we can rely on sub-exponential security of the underlying primitives and still incur a "small" loss by setting the security parameter appropriately [17]. In the full version of our paper, we show that by setting the parameters appropriately, we can construct a adaptively secure $Q$-$\mathsf{GID}$ $\mathsf{MA}$-$\mathsf{FE}$ scheme assuming the sub-exponential security of the underlying assumptions in ROM. This gives us the following result.

**Theorem 4 (Informal).** *Assuming the existence of sub-exponentially secure* $\mathsf{PKE}$ *and* $\mathsf{PRG}$ *schemes, there exists a* $(t, \epsilon)$*-adaptively secure bounded* $\mathsf{MA}$-$\mathsf{FE}$ *scheme for* $\mathsf{P}/\mathsf{Poly}$ *circuits in the random oracle model.*

For the exact values of $t, \epsilon$, and a detailed security analysis, we refer the reader to the full version of our paper. We remark that the security analysis of this scheme is very non-trivial, and this is because as a reduction algorithm, we cannot just use a fully adaptive adversary to break the underlying partial adaptive security. This will be an incorrect argument, as the adversary's behavior is correlated with the random selections made by a reduction algorithm at the beginning. Thus, when the attacker's choice of $\mathsf{GID}$ queries match the reduction algorithm's guess, then there could be negative correlation between the reduction's success and the adversary's success. Similar issues commonly arose in the context of proving adaptive security of Identity-Based Encryption [61]. To this end, Waters [61] proposed an "artificial abort" technique to not use an adversary when there is negative correlation, and this opens the door to a successful reduction. In this work, we borrow ideas from the *artificial abort* technique, and use it to prove adaptive security of our construction. Our exact proof template is inspired by the *advantage counting variation of the artificial abort technique* developed in [34]. These techniques have been used at many places in the IBE and ABE literature, and to the best of our knowledge, our utilization of this technique in the context of multi-authority functional encryption for general circuits is a first. We believe future works in multi-authority encryption might benefit from our proof strategies.

## 2.4 Bootstrapping Compiler for $Q$-GID MA-FE

In this section, we provide the motivation and techniques behind our bootstrapping compiler which combines two $Q$-GID MA-FE schemes to generically produce schemes supporting a larger number of authorities. Our techniques rely only on the additional assumption of a secure PRF scheme. We remark that our compiler can only be used to construct a $O(1)$-authority $Q$-GID MA-FE scheme and provide a brief insight into this issue. However due to Theorem 3 and Theorem 5 (stated below), we can create an adaptively-secure $O(1)$-authority $Q$-GID MA-FE scheme for P/Poly circuits from DDH or BDDH.

*Motivation for Studying Bootstrapping.* While studying new bootstrapping compilers is already a very interesting research topic in functional encryption and its generalizations, we were able to find an additional motivation for developing a bootstrapping compiler for MA-FE. In a few words, our additional motivation is stated as Corollary 1. It is a very interesting result which states that there exists a $2/3$-authority adaptively secure $Q$-GID MA-FE scheme from polynomially-hard standard assumptions. Compared to this, other results for adaptively secure $Q$-GID MA-FE schemes in our work require either strong assumptions ($n$-party niKE) or sub-exponentially secure PKE in the ROM. Ideally, we want to design adaptively-secure construction for any $n = \mathsf{poly}(\lambda)$ under just polynomially-hard standard assumptions. Since $n$-party niKE is a very strong assumption, not currently known from standard assumptions for $n > 3$. Thus, it appears that we have only one approach to design bounded MA-FE schemes for $\mathsf{poly}(\lambda)$-authorities in the standard model. And, that approach goes via $n$-party niKE which is only known from indistinguishability obfuscation.

Observe that any multi-authority scheme itself is inherently a way of combining $n$ independent authorities for arguing the security $n$-ary circuits. In addition, we can also compose circuits where one circuit outputs the description of other circuits. These observations lead us to a natural question.

*Can we combine two $n$-authority* MA-FE *schemes to produce a $2n$-authority* MA-FE *scheme?*

Motivated by this question, we started to look into an MA-FE compiler for bootstrapping the maximum number of authorities that can be supported. This way, we can realize an $n$-authority MA-FE scheme for P/Poly starting from a

2-authority or 3-authority scheme in $O(\log n)$ combinations. We provide such a bootstrapping compiler that combines two $n$-authority $\mathsf{MA\text{-}FE}$ schemes to a $2n$-authority $\mathsf{MA\text{-}FE}$ scheme.

*Circuit Composition for Bootstrapping* $\mathsf{MA\text{-}FE}$. The main idea of the compiler can be understood as follows. Given a $2n$-ary circuit $C$, one can construct an $n$-ary circuit $F_C$ such that $F_C(x_{n+1}, \ldots, x_{2n}) = C(\cdot, \ldots, \cdot, x_{n+1}, \ldots, x_{2n})$. That is, $F_C$ is an $n$-ary $\mathsf{P/Poly}$ circuit that takes the last $n$ inputs meant for $C$ and outputs the description of an $n$-ary circuit which is nothing but $C$ with the last $n$ inputs hardwired inside it. So, given a $2n$-ary circuit, we can split it "recursively" into two $n$-ary circuits. We can simply encrypt $F_C$ with the second instantiation of $n$-authority $\mathsf{MA\text{-}FE}$. Technically, we need to modify $F_C$ so that it outputs the encryption of the $n$-ary version of $C$ under the first $n$-authority $\mathsf{MA\text{-}FE}$ instantiation. We refer the reader to the full version of our paper for more details. In summary, we transform $C$ into $F_C$ and encrypt $F_C$ under the second $n$-authority instantiation where $F_C$ recursively calls the first $n$-authority instantiation's encryption algorithm.

The high level technical idea seems to bear resemblance to the bootstrapping techniques used to build indistinguishability obfuscation from succinct one-key functional encryption [7,16]. However, the resemblance seems to fade away when we carefully compare our compiler with theirs due to significantly diverging goals. Regardless, the main point we want to get across is that by using this tree-esuqe bootstrapping compiler, it looks like we can build an $n$-authority $\mathsf{MA\text{-}FE}$ scheme for any $n = \mathsf{poly}(\lambda)$. Because we can go from $2n$-authority instantiations to $4n$ and so on. And, proceeding this way, ideally, within $O(\log n)$, we can build an $\mathsf{poly}(\lambda)$-authority bounded $\mathsf{MA\text{-}FE}$ scheme. Unfortunately, this is not the case!

The issue is that with $d$ layers of recursive construction, size of $F_C$ grows with $\mathsf{poly}(\lambda)^d$. This is because size of $F_C$ is at least the size of the encryption circuit of first $n$-authority $\mathsf{MA\text{-}FE}$ scheme. We are encrypting $F_C$ again. Hence, any non-linear circuit size will result in exponential blow-up in encryption circuit's size making it too inefficient to use it for more than constant many layers of combination. A basic overview of the compiler is as follows.

**CRS Generation:** Output $\mathsf{crs}$ for both the instantiations.

**Authority Setup:** For $\mathsf{id} \in [n]$, output key pair from first instantiation and for the rest from the second.

**Key Generation:** If $\mathsf{id} \in [n]$, generate secret using the first instantiation. Otherwise, the second.

**Encryption:** Given $C$, generate $F_C$ which outputs the encryption of first instantiation. Encrypt $F_C$ using second instantiation and output the ciphertext.

The adaptive simulation security of our resulting scheme can be appropriately reduced to the adaptive simulation security of the underlying two instantiations. This is because once we rely on the security of the second instantiation, we can internally rely on the security of the first instantiation. Thus, using the above compiler, we get the following result.

**Theorem 5 (Informal).** *Assuming the existence of two $n$-authority* MA-FE *schemes, there exists a $2n$-authority* MA-FE *scheme where $n > 1, n = O(1)$ in the* crs *model.*

Similarly, we have the following corollary.

**Corollary 2.** *Assuming polynomial hardness of* DDH *or* BDDH*, there exists a $O(1)$-authority $Q$-*GID MA-FE *scheme for* P/Poly *circuits.*

The above discussion is merely a high-level overview of our compiler. For an in-depth overview, construction, and security analysis of this compiler, please check the full version of our paper. We conclude by stating that, to the best of our knowledge, this is the first bootstrapping compiler for any type of multi-authority functional encryption system. We think our bootstrapping compiler might be of independent interest, and could be useful for future research on MA-FE. Lastly, we leave an open question, which is to create a compiler that can efficiently support a super-constant number of authorities. This will readily lead to poly($\lambda$)-authority $Q$-GID MA-FE scheme for P/Poly circuits under polynomially hard standard assumptions.

## References

1. Agrawal, S., Gorbunov, S., Vaikuntanathan, V., Wee, H.: Functional encryption: new perspectives and lower bounds. In: Canetti, R., Garay, J.A. (eds.) CRYPTO 2013. LNCS, vol. 8043, pp. 500–518. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-40084-1_28
2. Agrawal, S., Goyal, R., Tomida, J.: Multi-party functional encryption. In: Nissim, K., Waters, B. (eds.) TCC 2021. LNCS, vol. 13043, pp. 224–255. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-90453-1_8
3. Agrawal, S., Maitra, M., Vempati, N.S., Yamada, S.: Functional encryption for Turing machines with dynamic bounded collusion from LWE. In: Malkin, T., Peikert, C. (eds.) CRYPTO 2021. LNCS, vol. 12828, pp. 239–269. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-84259-8_9
4. Agrawal, S., Rosen, A.: Functional encryption for bounded collusions, revisited. In: Kalai, Y., Reyzin, L. (eds.) TCC 2017. LNCS, vol. 10677, pp. 173–205. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-70500-2_7
5. Ambrona, M., Gay, R.: Multi-authority ABE, revisited. Cryptology ePrint Archive (2021)
6. Ananth, P., Jain, A., Lin, H., Matt, C., Sahai, A.: Indistinguishability obfuscation without multilinear maps: new paradigms via low degree weak pseudorandomness and security amplification. In: Boldyreva, A., Micciancio, D. (eds.) CRYPTO 2019. LNCS, vol. 11694, pp. 284–332. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-26954-8_10
7. Ananth, P., Jain, A.: Indistinguishability obfuscation from compact functional encryption. In: Gennaro, R., Robshaw, M. (eds.) CRYPTO 2015. LNCS, vol. 9215, pp. 308–326. Springer, Heidelberg (2015). https://doi.org/10.1007/978-3-662-47989-6_15
8. Ananth, P., Jain, A., Sahai, A.: Indistinguishability obfuscation from functional encryption for simple functions. Cryptology ePrint Archive (2015)

9. Ananth, P., Vaikuntanathan, V.: Optimal bounded-collusion secure functional encryption. In: Hofheinz, D., Rosen, A. (eds.) TCC 2019. LNCS, vol. 11891, pp. 174–198. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-36030-6_8
10. Applebaum, B., Ishai, Y., Kushilevitz, E.: Computationally private randomizing polynomials and their applications. Comput. Complex. **15**(2), 115–162 (2006)
11. Baltico, C.E.Z., Catalano, D., Fiore, D., Gay, R.: Practical functional encryption for quadratic functions with applications to predicate encryption. In: Katz, J., Shacham, H. (eds.) CRYPTO 2017. LNCS, vol. 10401, pp. 67–98. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-63688-7_3
12. Barak, B., et al.: On the (im)possibility of obfuscating programs. J. ACM **59**(2), 6 (2012)
13. Beaver, D., Micali, S., Rogaway, P.: The round complexity of secure protocols. In: Proceedings of the Twenty-Second Annual ACM Symposium on Theory of Computing, pp. 503–513 (1990)
14. Bellare, M., Rogaway, P.: Random oracles are practical: a paradigm for designing efficient protocols. In: Proceedings of the 1st ACM Conference on Computer and Communications Security, pp. 62–73 (1993)
15. BenOr, M., Goldwasser, S., Wigderson, A.: Completeness theorems for noncryptographic fault-tolerant distributed computations. In: Proceedings of the 20th Annual Symposium on the Theory of Computing (STOC'88), pp. 1–10 (1988)
16. Bitansky, N., Vaikuntanathan, V.: Indistinguishability obfuscation from functional encryption. In: FOCS (2015)
17. Boneh, D., Boyen, X.: Efficient selective-ID secure identity-based encryption without random oracles. In: Cachin, C., Camenisch, J.L. (eds.) EUROCRYPT 2004. LNCS, vol. 3027, pp. 223–238. Springer, Heidelberg (2004). https://doi.org/10.1007/978-3-540-24676-3_14
18. Boneh, D., Franklin, M.: Identity-based encryption from the Weil pairing. In: Kilian, J. (ed.) CRYPTO 2001. LNCS, vol. 2139, pp. 213–229. Springer, Heidelberg (2001). https://doi.org/10.1007/3-540-44647-8_13
19. Boneh, D., et al.: Fully key-homomorphic encryption, arithmetic circuit ABE and compact garbled circuits. In: Nguyen, P.Q., Oswald, E. (eds.) EUROCRYPT 2014. LNCS, vol. 8441, pp. 533–556. Springer, Heidelberg (2014). https://doi.org/10.1007/978-3-642-55220-5_30
20. Boneh, D., Sahai, A., Waters, B.: Functional encryption: definitions and challenges. In: Ishai, Y. (ed.) TCC 2011. LNCS, vol. 6597, pp. 253–273. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-19571-6_16
21. Boneh, D., Silverberg, A.: Applications of multilinear forms to cryptography. Contemp. Math. **324**(1), 71–90 (2003)
22. Boneh, D., Waters, B.: Conjunctive, subset, and range queries on encrypted data. In: Vadhan, S.P. (ed.) TCC 2007. LNCS, vol. 4392, pp. 535–554. Springer, Heidelberg (2007). https://doi.org/10.1007/978-3-540-70936-7_29
23. Boneh, D., Zhandry, M.: Multiparty key exchange, efficient traitor tracing, and more from indistinguishability obfuscation. In: Garay, J.A., Gennaro, R. (eds.) CRYPTO 2014. LNCS, vol. 8616, pp. 480–499. Springer, Heidelberg (2014). https://doi.org/10.1007/978-3-662-44371-2_27
24. Brakerski, Z., Chandran, N., Goyal, V., Jain, A., Sahai, A., Segev, G.: Hierarchical functional encryption. In: 8th Innovations in Theoretical Computer Science Conference (ITCS 2017). Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik (2017)
25. Chase, M.: Multi-authority attribute based encryption. In: Vadhan, S.P. (ed.) TCC 2007. LNCS, vol. 4392, pp. 515–534. Springer, Heidelberg (2007). https://doi.org/10.1007/978-3-540-70936-7_28

26. Chase, M., Chow, S.S.M.: Improving privacy and security in multi-authority attribute-based encryption. In: ACM Conference on Computer and Communications Security, pp. 121–130 (2009)
27. Cocks, C.: An identity based encryption scheme based on quadratic residues. In: Honary, B. (ed.) Cryptography and Coding 2001. LNCS, vol. 2260, pp. 360–363. Springer, Heidelberg (2001). https://doi.org/10.1007/3-540-45325-3_32
28. Datta, P., Komargodski, I., Waters, B.: Decentralized multi-authority ABE for DNFs from LWE. In: Canteaut, A., Standaert, F.-X. (eds.) EUROCRYPT 2021. LNCS, vol. 12696, pp. 177–209. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-77870-5_7
29. Datta, P., Komargodski, I., Waters, B.: Decentralized multi-authority ABE for NC 1 from BDH. J. Cryptol. **36**(2), 6 (2023)
30. Datta, P., Pal, T.: Decentralized multi-authority attribute-based inner-product FE: large universe and unbounded. In: Boldyreva, A., Kolesnikov, V. (eds.) PKC 2023. LNCS, vol. 13940, pp. 587–621. Springer, Cham (2023). https://doi.org/10.1007/978-3-031-31368-4_21
31. Diffie, W., Hellman, M.: New directions in cryptography. IEEE Trans. Inf. Theory **22**(6), 644–654 (1976)
32. Diffie, W., Hellman, M.E.: Multiuser cryptographic techniques. In: AFIPS National Computer Conference, pp. 109–112 (1976)
33. Dodis, Y., Katz, J., Xu, S., Yung, M.: Key-insulated public key cryptosystems. In: International Conference on the Theory and Applications of Cryptographic Techniques (2002)
34. Freitag, C., et al.: Signature schemes with randomized verification. In: Gollmann, D., Miyaji, A., Kikuchi, H. (eds.) ACNS 2017. LNCS, vol. 10355, pp. 373–389. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-61204-1_19
35. Garg, R., Goyal, R., Lu, G.: Dynamic collusion functional encryption and multi-authority attribute-based encryption. In: Tang, Q., Teague, V. (eds.) PKC 2024. LNCS, vol. 14604, pp. 69–104. Springer, Cham (2024). https://doi.org/10.1007/978-3-031-57728-4_3
36. Garg, R., Goyal, R., Lu, G., Waters, B.: Dynamic collusion bounded functional encryption from identity-based encryption. In: Dunkelman, O., Dziembowski, S. (eds.) EUROCRYPT 2022. LNCS, vol. 13276, pp. 736–763. Springer, Cham (2022). https://doi.org/10.1007/978-3-031-07085-3_25
37. Garg, S., Gentry, C., Halevi, S., Raykova, M., Sahai, A., Waters, B.: Candidate indistinguishability obfuscation and functional encryption for all circuits. SIAM J. Comput. **45**(3), 882–929 (2016)
38. Goldwasser, S., Lewko, A., Wilson, D.A.: Bounded-collusion IBE from key homomorphism. In: Cramer, R. (ed.) TCC 2012. LNCS, vol. 7194, pp. 564–581. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-28914-9_32
39. Gorbunov, S., Vaikuntanathan, V., Wee, H.: Functional encryption with bounded collusions via multi-party computation. In: Safavi-Naini, R., Canetti, R. (eds.) CRYPTO 2012. LNCS, vol. 7417, pp. 162–179. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-32009-5_11
40. Gorbunov, S., Vaikuntanathan, V., Wee, H.: Attribute-based encryption for circuits. In: STOC (2013)
41. Gorbunov, S., Vaikuntanathan, V., Wee, H.: Predicate encryption for circuits from LWE. In: Gennaro, R., Robshaw, M. (eds.) CRYPTO 2015, Part II. LNCS, vol. 9216, pp. 503–523. Springer, Heidelberg (2015). https://doi.org/10.1007/978-3-662-48000-7_25

42. Goyal, R., Koppula, V., Waters, B.: Lockable obfuscation. In: 58th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2017, pp. 612–621 (2017)
43. Goyal, V., Pandey, O., Sahai, A., Waters, B.: Attribute-based encryption for fine-grained access control of encrypted data. In: Proceedings of the 13th ACM Conference on Computer and Communications Security, pp. 89–98 (2006)
44. Hiroka, T., Kitagawa, F., Morimae, T., Nishimaki, R., Pal, T., Yamakawa, T.: Certified everlasting secure collusion-resistant functional encryption. Technical report, and more. Cryptology ePrint Archive, Report 2023/236 (2023)
45. Hiroka, T., Morimae, T., Nishimaki, R., Yamakawa, T.: Certified everlasting functional encryption. arXiv preprint arXiv:2207.13878 (2022)
46. Jain, A., Lin, H., Sahai, A.: Indistinguishability obfuscation from well-founded assumptions. In: Proceedings of the 53rd Annual ACM SIGACT Symposium on Theory of Computing, pp. 60–73 (2021)
47. Jain, A., Lin, H., Sahai, A.: Indistinguishability obfuscation from LPN over $\mathbb{F}_p$, DLIN, and PRGs in $NC^0$. In: Dunkelman, O., Dziembowski, S. (eds.) EUROCRYPT 2022. LNCS, vol. 13275, pp. 670–699. Springer, Cham (2022). https://doi.org/10.1007/978-3-031-06944-4_23
48. Joux, A.: A one round protocol for tripartite Diffie-Hellman. J. Cryptol. **17**(4), 263–276 (2004)
49. Katz, J., Sahai, A., Waters, B.: Predicate encryption supporting disjunctions, polynomial equations, and inner products. In: Smart, N. (ed.) EUROCRYPT 2008. LNCS, vol. 4965, pp. 146–162. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-78967-3_9
50. Kim, S.: Multi-authority attribute-based encryption from LWE in the OT model. Cryptology ePrint Archive (2019)
51. Lewko, A., Waters, B.: Decentralizing attribute-based encryption. In: Paterson, K.G. (ed.) EUROCRYPT 2011. LNCS, vol. 6632, pp. 568–588. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-20465-4_31
52. Lin, H.: Indistinguishability obfuscation from SXDH on 5-linear maps and locality-5 PRGs. In: Katz, J., Shacham, H. (eds.) CRYPTO 2017. LNCS, vol. 10401, pp. 599–629. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-63688-7_20
53. Michalevsky, Y., Joye, M.: Decentralized policy-hiding ABE with receiver privacy. In: Lopez, J., Zhou, J., Soriano, M. (eds.) ESORICS 2018. LNCS, vol. 11099, pp. 548–567. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-98989-1_27
54. Okamoto, T., Takashima, K.: Decentralized attribute-based encryption and signatures. IEICE Trans. Fundam. Electron. Commun. Comput. Sci. **103**(1), 41–73 (2020)
55. Sahai, A., Seyalioglu, H.: Worry-free encryption: functional encryption with public keys. In: Proceedings of the 17th ACM Conference on Computer and Communications Security. pp. 463–472 (2010)
56. Sahai, A., Waters, B.: Fuzzy identity-based encryption. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 457–473. Springer, Heidelberg (2005). https://doi.org/10.1007/11426639_27
57. Sahai, A., Waters, B.: How to use indistinguishability obfuscation: deniable encryption, and more. In: Proceedings of the Forty-Sixth Annual ACM Symposium on Theory of Computing, pp. 475–484 (2014)
58. Shamir, A.: How to share a secret. Commun. ACM **22**(11), 612–613 (1979)
59. Shamir, A.: Identity-based cryptosystems and signature schemes. In: Blakley, G.R., Chaum, D. (eds.) CRYPTO 1984. LNCS, vol. 196, pp. 47–53. Springer, Heidelberg (1985). https://doi.org/10.1007/3-540-39568-7_5

60. Wang, Z., Fan, X., Liu, F.-H.: FE for inner products and its application to decentralized ABE. In: Lin, D., Sako, K. (eds.) PKC 2019. LNCS, vol. 11443, pp. 97–127. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-17259-6_4
61. Waters, B.: Efficient identity-based encryption without random oracles. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 114–127. Springer, Heidelberg (2005). https://doi.org/10.1007/11426639_7
62. Wichs, D., Zirdelis, G.: Obfuscating compute-and-compare programs under LWE. In: 58th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2017, pp. 600–611 (2017)
63. Yao, A.C.C.: How to generate and exchange secrets. In: 27th Annual Symposium on Foundations of Computer Science (SFCS 1986), pp. 162–167. IEEE (1986)

# Multi-client Attribute-Based and Predicate Encryption from Standard Assumptions

David Pointcheval and Robert Schädlich(✉)

DIENS, École normale supérieure, CNRS, Inria, PSL University, Paris, France
robert.schaedlich@ens.fr

**Abstract.** Multi-input Attribute-Based Encryption (ABE) is a generalization of key-policy ABE where attributes can be independently encrypted across several ciphertexts, and a joint decryption of these ciphertexts is possible if and only if the combination of attributes satisfies the policy of the decryption key. We extend this model by introducing a new primitive that we call Multi-Client ABE (MC-ABE), which provides the usual enhancements of multi-client functional encryption over multi-input functional encryption. Specifically, we separate the secret keys that are used by the different encryptors and consider the case that some of them may be corrupted by the adversary. Furthermore, we tie each ciphertext to a label and enable a joint decryption of ciphertexts only if all ciphertexts share the same label. We provide constructions of MC-ABE for various policy classes based on SXDH. Notably, we can deal with policies that are not a conjunction of local policies, which has been a limitation of previous constructions from standard assumptions.

Subsequently, we introduce the notion of Multi-Client Predicate Encryption (MC-PE) which, in contrast to MC-ABE, does not only guarantee message-hiding but also attribute-hiding. We present a new compiler that turns any constant-arity MC-ABE into an MC-PE for the same arity and policy class. Security is proven under the LWE assumption.

## 1 Introduction

**Attribute-Based Encryption.** Attribute-based encryption (ABE) [31,42] is a powerful generalization of classical public-key encryption that enables fine-grained access control on encrypted data. In (key-policy) ABE, a ciphertext $\mathsf{CT}_x$ encrypting a message $\mu$ is generated with respect to a public attribute $x$ while a secret decryption key $\mathsf{DK}_f$ is generated with respect to a policy $f$. The decryption key $\mathsf{DK}_f$ is authorized to decrypt the ciphertext $\mathsf{CT}_x$ if and only if the attribute $x$ satisfies the policy $f$, *i.e.* $f(x) = 1$. Security requires indistinguishability in the presence of collusion attacks. That is, for any attribute $x$ and any pair of messages $(\mu^0, \mu^1)$, ciphertexts corresponding to $(x, \mu^0)$ and to $(x, \mu^1)$ are indistinguishable, even for adversaries possessing a set of decryption keys $\{\mathsf{DK}_{f_i}\}_i$ unless one of the keys $\mathsf{DK}_{f_i}$ is individually authorized to decrypt. A strengthening of this

© International Association for Cryptologic Research 2025
E. Boyle and M. Mahmoody (Eds.): TCC 2024, LNCS 15366, pp. 31–64, 2025.
https://doi.org/10.1007/978-3-031-78020-2_2

notion, traditionally referred to as Predicate Encryption (PE) [19,43], requires ciphertexts to not only hide messages but also their associated attributes.

**Decentralized Encryption.** Until recently, ABE and PE were solely studied in the centralized setting, *i.e.* the considered policies $f$ have arity one. The notions of Multi-Input ABE (MI-ABE) [11,20] and Multi-Input PE (MI-PE) [26] overcome this limitation by considering $n$ encryptors who each encrypt their inputs $(x_1, \mu_1), \ldots, (x_n, \mu_n)$ independently using uncorrelated random coins. The key generator provides decryption keys for arity $n$ functions $f$, and a joint decryption recovers $(\mu_1, \ldots, \mu_n)$ if and only if $f(x_1, \ldots, x_n) = 1$. As in the single-input case, the security model of MI-ABE only guarantees to hide the message encoded in a ciphertext while MI-PE hides in addition the associated attribute.

In practice, the multi-input versions of ABE and PE often seem more realistic as they allow data to be encrypted in different locations or at different points in time. As an example, consider a company that stores its client data in encrypted form on a server. Each employee has their own decryption key which they can use to decrypt parts of the data depending on the employee's role. At one point, the company decides to expand and opens several new branches across the country. Clients should be able to be served from each branch. This requires that data can be independently encrypted and uploaded to the central server while still being subject to the global access control for employees. To implement these requirements, we could use an MI-ABE or MI-PE.

Let us extend the above scenario as follows. First, suppose that one of the company's branches falls victim to a hacker attack. Since MI-ABE and MI-PE use the same master secret key across all encryption slots, an attack on a single branch threatens to compromise the security of the entire system. Instead, it would be better if a restricted form of security could be preserved even if a few branches are corrupted. Second, the company's data might be time-sensitive in a sense that, say, data from different years should not be authorized for a joint decryption. To prevent unintended decryptions and the resulting data leakage, we may wish to equip ciphertexts with timestamps and allow a joint decryption if and only if all involved ciphertexts share the same timestamp.

To deal with this extended scenario, we introduce two natural generalization of MI-ABE and MI-PE which we dub *Multi-Client ABE* (MC-ABE) and *Multi-Client PE* (MC-PE). In contrast to MI-ABE and MI-PE, our new notions implement two additional features. First, they separate the secret keys of the slots and guarantee security even if some of them are known to the adversary. Second, encryption proceeds with respect to a label that can be used to realize a timestamp.

## 1.1 Related Work

The notion of MI-ABE had been studied first by Brakerski et al. [20] as a new pathway for achieving witness encryption. However, they did not consider strong security notions nor did they provide any constructions. In [11], Agrawal et al. provided the first constructions for 2-input ABE for $\mathsf{NC}^1$ from $\mathsf{LWE}$ and a nonstandard assumption on pairings. They also gave heuristic constructions for 2-input ABE for $\mathsf{P}$ and 3-input ABE for $\mathsf{NC}^1$. Additionally, they gave a compiler that lifts a constant-input ABE scheme to a PE scheme for the same arity and policy class, using a sophisticated nesting technique of lockable obfuscation which can be based on $\mathsf{LWE}$. In an independent work, Francati et al. [26] built MI-PE for *conjunctions* of (bounded) polynomial-depth circuits from $\mathsf{LWE}$[1]. Notably, they can support a polynomial number of inputs. Furthermore, when restricting to constant arity, they provide a construction that remains secure under user corruptions. On the negative side, neither of their MI-PE constructions can be proven secure under collusions. Very recently, Agrawal et al. [8] presented a constant-arity MI-ABE for $\mathsf{NC}^1$ whose security is based on *evasive* $\mathsf{LWE}$ which is a strong knowledge type assumption. When additionally assuming tensor $\mathsf{LWE}$, they can upgrade their scheme to support arbitrary policies in $\mathsf{P}$.

MI-ABE and MI-PE can both be viewed as a special case of the more general primitive Multi-Input Functional Encryption (MIFE) [28]. The notion of MIFE has been the subject of extensive studies resulting in large body of works with various trade-offs between expressiveness, security, underlying assumptions and efficiency, *e.g.* [1–6,14,22,24,28,36,44]. As MIFE for $\mathsf{NC}^1$ is known to imply indistinguishability Obfuscation (iO) [15,27] it remains an important area of research to build MIFE schemes for simpler function classes from assumptions not known to imply iO. Some of these function classes are still powerful enough to imply MI-ABE. Specifically, Nguyen et al. [41] built the first attribute-based Multi-Client Functional Encryption (MCFE) for inner products, where policies are conjunctions of Linear Secret Sharings (LSS). As they consider MCFE instead of MIFE, their construction supports corruption of users and encryption with respect to labels. They make use of pairings and proof security under the $\mathsf{SXDH}$ assumption. However, their security model does not allow repetitions[2]. In [9], Agrawal et al. presented the first attribute-based MIFE for attribute-weighted sums. The supported policies are conjunctions of $\mathsf{NC}^1$ for a polynomial number of slots. Their construction uses pairings and is proven secure under the matrix $\mathsf{DDH}$ assumption. By plugging the scheme into the compiler from [11], it can be lifted to MI-PE for constant arity and without corruptions.

## 1.2 Our Results

In this work, we introduce the notions of MC-ABE and MC-PE as a generalization of their multi-input siblings. We discuss our constructions for the two primitives below. For a comparison with known results, please see Table 1.

**MC-ABE for Non-conjunctions.** Prior to our work, all known constructions of MI-ABE and MI-PE fall into one of two categories: they are either based

[1] A policy $f$ is said to be a *conjunction* of a policy class $\mathcal{F}$ if there exist policies $f_1, \ldots, f_n \in \mathcal{F}$ such that $f(x_1, \ldots, x_n) = f_1(x_1) \wedge \cdots \wedge f_n(x_n)$.

[2] Unless stated otherwise, we use the term MCFE as a generalization of MIFE, so it allows multiple uses of labels. In contrast, a weaker notion of MCFE has been considered in the literature [22] where each label can be used only once, thus it does not imply MIFE. We refer to this weaker version as MCFE *without repetitions*.

on standard assumptions but their supported policies are only conjunctions of local policies [9,26,41], or they can handle more complex policy classes such as $\mathsf{NC}^1$ or $\mathsf{P}$ but their security proof relies on nonstandard assumptions [8,11]. We therefore raise the following question:

Is it possible to build MI-ABE or MC-ABE from standard assumptions for policies that are not a conjunction of local policies?

It is well known that MI-ABE for LSS can be generically upgraded to MI-ABE for $\mathsf{NC}^1$ via a doubling of the attribute space, and that (polynomial-arity) MI-ABE for $\mathsf{NC}^1$ implies Witness Encryption (WE) for languages that can be verified in $\mathsf{NC}^1$ [20]. Considering the fact that the construction of WE with $\mathsf{NC}^1$ verification from standard assumptions is still an open problem, it is not surprising that MI-ABE for policies that are not a conjunction has also remained elusive so far, even for relatively simple policy classes such as LSS. As previous works [9,26], we are not able to build MI-ABE or MC-ABE for a policy class that is powerful enough to trigger the entire chain of implications up to WE. Nonetheless, we identify various special cases that circumvent the known implications to WE, thereby giving an affirmative answer to the above question. Specifically, we present constructions of MC-ABE for the following situations:

1. *Small Parameters.* If the arity $n$ and the attribute space $\{0,1\}^k$ are small (more precisely, they satisfy $kn = O(\log \lambda)$), then we can build an MC-ABE for all $\mathsf{NC}^1$ policies.
2. *Simple Policies.* If $k = n = \text{poly}(\lambda)$, then we can build MC-ABEs for $\mathsf{NC}^0$ policies and threshold policies with constant threshold, *i.e.* policies that accept any combination of at least $\tau$ out of the total of $kn$ attributes, where $\tau = O(1)$ or $\tau = kn - O(1)$. Note that $\mathsf{NC}^0$ policies can only depend on a constant number of inputs whereas threshold policies depend on *all* inputs; so they cannot be implemented in $\mathsf{NC}^0$.
3. *Weaker Security.* In the weaker MCFE model without repetitions, we can choose $k = n = \text{poly}(\lambda)$ and build MC-ABE for the policy class $\mathsf{NC}^1$ and no user corruptions, or for the policy class LSS with user corruptions.

We discuss the relation between our constructions and the (non-)implications to WE in more detail below.

**From MC-ABE to MC-PE.** In [11], the authors present two generic compilers that lift an MI-ABE scheme to MI-PE for the same policy class. The first one can deal with any constant arity but works only in a weak security model where the adversary must not obtain valid decryption keys for *any* ciphertext, even if the ciphertext corresponds to a "non-challenge" encryption query $(x, \mu^0, \mu^1)$ where $\mu^0 = \mu^1$. Note that the ability to decrypt such non-challenge ciphertexts does not render the security game trivial and admitting this kind of queries yields a stronger security model. Indeed, their second compiler allows the decryption of non-challenge ciphertexts, but works only for arity 2.

In this work, we present a new generic compiler that works for a constant number of inputs and achieves the stronger security model. Moreover, it can deal

**Table 1.** Comparison with existing works in MI-ABE and MI-PE

| Work | Arity | Attribute[1] | Collusion | Corruption | Labels | Policy Class | Assumptions |
|---|---|---|---|---|---|---|---|
| [11] | 2 | private | ✓ | ✗ | ✗ | $\mathsf{NC}^1$ | KOALA[2], LWE |
| [8] | const | private | ✓ | ✗ | ✗ | $\mathsf{NC}^1$ | Evasive LWE[3] |
| | | | | | | $\mathsf{P}$ | Evasive and Tensor LWE |
| [41] | poly | public | ✓ | ✓ | OT[4] | Conjunctions of LSS | SXDH |
| [26] | poly | private | ✗ | ✗ | ✗ | Conjunctions of $\mathsf{P}$ | LWE |
| | const | | | ✓ | | | |
| [9] | poly | public | ✓ | ✓ | ✗ | Conjunctions of $\mathsf{NC}^1$ | Matrix DDH |
| [9] + [11] | const | private | ✓ | ✗ | ✗ | Conjunctions of $\mathsf{NC}^1$ | Matrix DDH |
| full version | poly | public | ✓ | ✓ | OT | LSS | SXDH |
| | | | | ✗ | | $\mathsf{NC}^1$ | |
| Sect. 5 | log[5] | public | ✓ | ✓ | ✓ | $\mathsf{NC}^1$ | SXDH |
| | poly | | | | | $\mathsf{NC}^0$ or const threshold | |
| Sec. 5 + Sect. 6 | const[6] | private | ✓ | ✓ | ✓ | $\mathsf{NC}^1$ | SXDH, LWE |
| [9] + Sect. 6 | const | private | ✓ | ✓ | ✗ | Conjunctions of $\mathsf{NC}^1$ | Matrix DDH, LWE |

[1] Public attributes correspond to ABE and private attributes to PE.
[2] KOALA is a nonstandard knowledge type assumption on pairings.
[3] Evasive LWE is a nonstandard knowledge type assumption on lattices.
[4] OT refers to *one-time* labels, *i.e.* the weaker MCFE model without repetitions.
[5] More precisely, the scheme's arity $n$ and attribute space $\{0,1\}^k$ are subject to the
condition $kn = O(\log \lambda)$.
[6] The limitation from 5 is still in place and translates into $k = O(\log \lambda)$. Therefore, it does not include the next row which allows only conjunctions but $k = \text{poly}(\lambda)$.

with labels and corruption of users, thus turning MC-ABE into MC-PE. Similar to [11], our construction relies on lockable obfuscation whose security can be based on $\mathsf{LWE}$.

## 1.3 Relation to Witness Encryption

A *witness encryption* (WE) scheme for an $\mathsf{NP}$ relation $\mathcal{R}$ defined over a language $\mathcal{L}$ allows a sender to efficiently encrypt a message $\mu$ with respect to a problem instance $x$. A receiver holding a witness $w$ can recover the message $\mu$ if $(x, w) \in \mathcal{R}$. Security requires that ciphertexts for messages $\mu^0$ and $\mu^1$ are computationally indistinguishable if $x \notin \mathcal{L}$. The authors of [20] define a relaxation of classical WE that they call *non-trivially eXponentially efficient WE* (XWE), where the runtime of the encryption algorithm for witnesses of length $n$ is $\widetilde{O}(2^{\gamma n})$ for some constant $\gamma < 1$ called the *compression factor*.

In [20], the authors show that $n$-input ABE for a policy class $\mathcal{F}$ implies WE for relations with length $n$ witnesses whose verification algorithm is in $\mathcal{F}$. If $n = \text{poly}(\lambda)$ and $\mathcal{F} = \mathsf{P}$, we obtain WE for all NP relations. But even for smaller arity or simpler policy classes there are nontrivial implications. Since there are $\mathsf{NP}$ relations that can be verified in $\mathsf{NC}^1$ (*e.g.* 3-SAT), MI-ABE for $\mathsf{NC}^1$ policies already implies WE for certain $\mathsf{NP}$ relations. Furthermore, it is shown that $n$-input ABE for $n < \text{poly}(\lambda)$ implies XWE with a compression factor of $\gamma = 1/(n+1)$. Plugging the two-input ABE from [11] or the $O(1)$-input ABE from [8] into the conversion to XWE, one obtains compression factors of $\gamma = 1/3$ and $\gamma = 1/O(1)$, respectively; although the latter result may be less interesting as their hardness assumption, evasive LWE, is already known to imply WE [45,46]. When relying on standard assumptions, the best known compression factor is still $\gamma = 1/2$ which corresponds to a classical single-input ABE scheme, and any improvement would be highly interesting. Unfortunately, all our constructions fail to improve the compression factor due to the following reasons. (1) We either need $n = \text{poly}(\lambda)$ and $k = 1$ in which case we immediately get (polynomially efficient) WE, or $n < \text{poly}(\lambda)$ and $k = \text{poly}(\lambda)$ in which case we obtain XWE with compression factor $1/(n+1)$. If both $k, n < \text{poly}(\lambda)$, it is unclear how the compression factor could be improved. (2) $\mathsf{NC}^0$ or constant-threshold policies are not powerful enough to verify an $\mathsf{NP}$ language. (3) The weaker MC-ABE model without repetitions does not imply MI-ABE, thus fails to imply (X)WE.

The work [26] presents an interesting alternative pathway towards WE. If the MI-ABE is secure under corruptions, then a two-input scheme for conjunctions of some policy class $\mathcal{F}$ implies WE for any relation whose verification algorithm lies in $\mathcal{F}$. Importantly, for the conversion of [26] to work, the first slot must have a wildcard while the second slot must not. This property is achieved by all our constructions. However, even in this case our construction for $\mathsf{NC}^1$ fails to imply WE because for $n = 2$, the constraint $kn = O(\log \lambda)$ translates into $k = O(\log \lambda)$ which is not enough as witnesses must be of polynomial length.

## 2 Technical Overview

We first introduce our new primitives MC-ABE and MC-PE. Our syntax closely follows [8]. Specifically, the 0-th client (the "encryptor") runs an algorithm $\mathsf{Enc}$ which takes as input a label $\mathsf{lab}_0$, an attribute $x_0$ and a message $\mu$ to create a ciphertext $\mathsf{CT}_{\mathsf{lab}_0,x_0}$. The other clients $1,\ldots,n-1$ (the "attribute key generators") run an algorithm $\mathsf{AKeyGen}$ which takes only a label $\mathsf{lab}_i$ and an attribute $x_i$ to generate a decryption key $\mathsf{DK}_{\mathsf{lab}_i,x_i}$. Policy decryption keys $\mathsf{DK}_f$ for a policy $f$ are generated by a central authority which runs an algorithm $\mathsf{PKeyGen}$. $\mathsf{CT}_{\mathsf{lab}_0,x_0}$ can be decrypted using $\{\mathsf{DK}_{\mathsf{lab}_i,x_i}\}_i$ and $\mathsf{DK}_f$ if $\mathsf{lab}_0 = \cdots = \mathsf{lab}_{n-1}$ and $f(x_0,\ldots,x_{n-1}) = 1$. For MC-ABE security, we require the usual ciphertext indistinguishability against collusion attacks under corruptions. MC-PE security additionally considers left-or-right queries for attributes in both slot 0 ciphertexts and slot $i$ attribute decryption keys for all $i \in [n-1]$. This leads to a subtle yet important difference in the security models. In MC-ABE, the encryption oracle of client 0 is the only left-or-right ("challenge") oracle. In this case, public-key security is stronger than secret-key security which is why we provide client 0 with a master public key $\mathsf{MPK}$ and all other clients $i \in [n-1]$ with a secret key $\mathsf{SK}_i$. In MC-PE on the other hand, the oracles of all clients take left-or-right queries. Now considering a public encryption algorithm would actually make the primitive weaker due to inevitable leakage. This is a well-known phenomenon in the context of MIFE and MCFE in general. For this reason, we consider MC-PE in the secret-key setting where the encryptor takes a secret key $\mathsf{SK}_0$ instead of a public key $\mathsf{MPK}$. We summarize the syntax as follows:

$$\begin{aligned} \text{Client } 0: &\quad \mathsf{Enc}(\mathsf{MPK}/\mathsf{SK}_0, \mathsf{lab}, x_0, \mu) \to \mathsf{CT}_{x_0} \\ \text{Client } i \in [n-1]: &\quad \mathsf{AKeyGen}(\mathsf{SK}_i, \mathsf{lab}, x_i) \to \mathsf{DK}_{x_i} \\ \text{Authority}: &\quad \mathsf{PKeyGen}(\mathsf{MSK}, f) \to \mathsf{DK}_f \end{aligned}$$

For completeness, we mention that MC-ABE and MC-PE can also be considered in an $n$-message setting where not only the 0-th but all slots encrypt a message. In [11], it was shown that a single-message MI-ABE scheme can be generically lifted to an $n$-message scheme. The conversion is extremely simple and basically runs $n$ single-message schemes in parallel with rotated slots. The same technique generalizes to MC-ABE and MC-PE.

### 2.1 Construction of MC-ABE

**Ingredients to Our Constructions.** Let $\mathbb{G} = (\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_\mathsf{t}, g_1, g_2, >, e, p)$ be a pairing group. For $i \in \{1, 2, \mathsf{t}\}$ and $a \in \mathbb{Z}_p$, we write $[\![a]\!]_i = g_i^a$ and use additive notations for the group operations.

An *inner-product functional encryption* (IPFE) scheme based on $\mathbb{G}$ enables the generation of ciphertexts $\mathsf{iCT}([\![\mathbf{x}]\!]_1)$ associated with vectors $\mathbf{x} \in \mathbb{Z}_p^N$ encoded in $\mathbb{G}_1$ and decryption keys $\mathsf{iDK}([\![\mathbf{y}]\!]_2)$ for vectors $\mathbf{y} \in \mathbb{Z}_p^N$ encoded in $\mathbb{G}_2$ such that the decryption of $\mathsf{iCT}([\![\mathbf{x}]\!]_1)$ with $\mathsf{iDK}([\![\mathbf{y}]\!]_2)$ reveals only the inner product $[\![\mathbf{x}^\top \mathbf{y}]\!]_\mathsf{t}$

of $\mathbf{x}$ and $\mathbf{y}$ encoded in $\mathbb{G}_t$ and hides all other information about $\mathbf{x}$ whereas $[\![\mathbf{y}]\!]_2$ is usually public. When we use several IPFE schemes in parallel, we add an index to indicate the respective instance, *e.g.* for the $i$-th IPFE instance, we write $\mathsf{iCT}_i([\![\mathbf{x}]\!]_1)$ and $\mathsf{iDK}_i([\![\mathbf{y}]\!]_2)$.

In the same vein, *identity-based encryption* (IBE) allows creating ciphertexts $\mathsf{idCT}(i, \mu)$ associated with an identity $i$ for a message $\mu$, and decryption keys $\mathsf{idDK}(i')$ associated with an identity $i'$. Decryption is possible if $i = i'$.

Given a vector $\mathbf{x} \in \{0,1\}^k$, we write $\Pi_\mathbf{x} = \{j \in [k] : \mathbf{x}[j] = 1\}$. A *linear secret sharing* (LSS) scheme allows to decompose a secret scalar $s \in \mathbb{Z}_p$ into a vector of shares $\mathsf{Share}(s, f) \to \mathbf{s} \in \mathbb{Z}_p^k$ with respect to some policy $f\colon \{0,1\}^k \to \{0,1\}$ such that $s$ can be reconstructed from a subset of the shares $\{\mathbf{s}[j]\}_{j\in\Pi_\mathbf{x}}$ for $\mathbf{x} \in \{0,1\}^k$ if and only if $f(\mathbf{x}) = 1$. For this reconstruction, there exists an efficient algorithm $\mathsf{FindCoef}(\Pi_\mathbf{x}, f)$ that outputs coefficients $\omega_1, \ldots, \omega_k$ such that $\omega_j = 0$ for all $j \notin \Pi_\mathbf{x}$ and $\sum_{j\in[k]} \omega_j \mathbf{s}[j] = s$.

**Key-Policy ABE for LSS Policies.** Our starting point is a technique that combines IPFE with secret sharing schemes. The same approach has recently been used to build ciphertext-policy ABEs with interesting new features [10,12, 13,35]. Very roughly, these works view a secret sharing as a weak form of one-time, non-collusion resistant ABE, which is then lifted to full ABE using IPFE. To encrypt a message $\mu$ from a polynomial-size space[3] under a policy $f$, they generate secret shares $\mathsf{Share}(\mu, f) \to \mathbf{s} \in \mathbb{Z}_p^k$ and encode them in the ciphertexts of $k$ independent IPFE instances. To generate a key for an attribute vector $\mathbf{x} \in \{0,1\}^k$, one picks a uniformly random scalar $r$ and generates IPFE secret keys of $r$ for those IPFE instances that correspond to indices in $\Pi_\mathbf{x}$:

$$\left.\begin{array}{ll}\mathsf{CP\text{-}ABE.CT}_f: & \{\mathsf{iCT}_j([\![\mathbf{s}[j]]\!]_2)\}_{j\in[k]} \\ \mathsf{CP\text{-}ABE.DK}_\mathbf{x}: & [\![r]\!]_t, \left\{\mathsf{iDK}_j([\![r]\!]_1)\right\}_{j\in\Pi_\mathbf{x}}\end{array}\right\} \quad [\![r]\!]_t, \left\{[\![r\cdot\mathbf{s}[j]]\!]_t\right\}_{j\in\Pi_\mathbf{x}}$$

IPFE decryption yields target group encodings of $r\mathbf{s}[j]$ for all $j \in \Pi_\mathbf{x}$. If $f(\mathbf{x}) = 1$, one can run $\mathsf{FindCoef}(\Pi_\mathbf{x}, f) \to \{\omega_j\}_j$ and recover the product $r\cdot\mu$ encoded in $\mathbb{G}_t$:

$$\textstyle\sum_{j\in\Pi_\mathbf{x}} \omega_j [\![r\cdot\mathbf{s}[j]]\!]_t = [\![*]\!] r \cdot \sum_{j\in\Pi_\mathbf{x}} \omega_j \cdot \mathbf{s}[j]_t = [\![r\cdot\mu]\!]_t$$

Then one can find $\mu$ by solving the discrete logarithm of $[\![r\cdot\mu]\!]_t$ in basis $[\![r]\!]_t$. Under an appropriate hardness assumption, the presence of $r$ prevents adversaries from meaningfully "combining" information obtained from decryptions with different ABE decryption keys.

To turn this into a key-policy scheme, the obvious idea is to flip ciphertexts and decryption keys. However, there is one subtlety: when generating a secret sharing for a policy $f$ during the key generation, the message $\mu$ is not known. So one cannot generate secret shares of $\mu$. Therefore, we generate the secret sharing for a random scalar $s$ which is used to mask $\mu$. Then one uses another IPFE

[3] The restriction to a polynomial-size message space is only for notational convenience throughout the technical overview. For superpolynomial size, one can simply view the construction as a KEM with messages in $\mathbb{G}_t$ that can be used as a one-time pad.

instance to enable decryption:

$$\left.\begin{array}{ll}\mathsf{KP\text{-}ABE.CT}_{\mathbf{x}}: & \mathsf{iCT}_0([\![r,\mu]\!]_1), \{\mathsf{iCT}_j([\![r]\!]_1)\}_{j\in\Pi_{\mathbf{x}}} \\ \mathsf{KP\text{-}ABE.DK}_f: & \mathsf{iDK}_0([\![s,1]\!]_2), \{\mathsf{iDK}_j([\![\mathbf{s}[j]]\!]_2)\}_{j\in[k]}\end{array}\right\} \begin{array}{l}[\![r\cdot s+\mu]\!]_{\mathsf{t}}, \\ \{[\![r\cdot \mathbf{s}[j]]\!]_{\mathsf{t}}\}_{j\in\Pi_{\mathbf{x}}}\end{array}$$

Similar to above, if $f(\mathbf{x})=1$, one can run $\mathsf{FindCoef}(\Pi_{\mathbf{x}}, f)\to\{\omega_j\}_j$ and recover the message $\mu$ encoded in $\mathbb{G}_{\mathsf{t}}$:

$$[\![r\cdot s+\mu]\!]_{\mathsf{t}} - \textstyle\sum_{j\in\Pi_{\mathbf{x}}}\omega_j[\![r\cdot\mathbf{s}[j]]\!]_{\mathsf{t}} = [\![r\cdot s+\mu]\!]_{\mathsf{t}} - [\![*]\!]r\cdot\sum_{j\in\Pi_{\mathbf{x}}}\omega_j\mathbf{s}[j]_{\mathsf{t}} = [\![\mu]\!]_{\mathsf{t}}$$

**MC-ABE for LSS Without Repetitions.** We next discuss how the generation of the ciphertext $\mathsf{CT}_{\mathbf{x}}$ can be distributed so as to turn the above key-policy ABE into an MC-ABE. A natural approach is to follow [22,41] who construct (Decentralized) MCFE for inner products. Even though not explicitly stated as such, they essentially use an independent IPFE instance for each client, and the common randomness $r$ in the ciphertexts facing a secret sharing $(\mathbf{s}[j])_j$ in the decryption keys is provided by a random oracle. Translating this idea into our context, each client $i\in[0;n-1]$ holds the master secret keys of $k$ independent IPFE instances, where $k$ is the dimension of the attribute vectors[4]. The 0-th client additionally holds $\mathsf{iMSK}_0$ as it takes the message input. To obtain the common random scalar $[\![r]\!]_1$ encoded in $\mathbb{G}_1$, we use a hash function $\mathsf{H}\colon\{0,1\}^*\to\mathbb{G}_1$. To generate a decryption key for an attribute vector $\mathbf{x}_i\in\{0,1\}^k$ with respect to a label $\mathsf{lab}$, the corresponding client $i\in[n-1]$ computes $[\![r]\!]_1\leftarrow\mathsf{H}(\mathsf{lab})$ and issues $\{\mathsf{iCT}_{i,j}([\![r]\!]_1)\}_{j\in\Pi_{\mathbf{x}_i}}$. Similarly, to encrypt a message $\mu$ with respect to $\mathbf{x}_0\in\{0,1\}^k$, the 0-th client computes $\{\mathsf{iCT}_{0,j}([\![r]\!]_1)\}_{j\in\Pi_{\mathbf{x}_0}}$ and additionally provides $\mathsf{iCT}_0([\![r,\mu]\!]_1)$. Decryption keys $\mathsf{DK}_f$ for policies $f$ are still generated by a central authority, so the policy key generation algorithm does not need to be modified. This leads us to the following MC-ABE for LSS:

$$\left.\begin{array}{ll}\mathsf{MC\text{-}ABE.CT}_{\mathbf{x}_0}: & \mathsf{iCT}_0([\![r,\mu]\!]_1), \big\{\mathsf{iCT}_{0,j}([\![r]\!]_1)\big\}_{j\in\Pi_{\mathbf{x}_0}} \\ \mathsf{MC\text{-}ABE.DK}_{\mathbf{x}_i}: & \big\{\mathsf{iCT}_{i,j}([\![r]\!]_1)\big\}_{j\in\Pi_{\mathbf{x}_i}} \\ \mathsf{MC\text{-}ABE.DK}_f: & \mathsf{iDK}_0([\![s,1]\!]_2), \big\{\mathsf{iDK}_{i,j}([\![\mathbf{s}[i,j]]\!]_2)\big\}_{i\in[0;n-1]}^{j\in[k]}\end{array}\right\} \begin{array}{l}[\![rs+\mu]\!]_{\mathsf{t}}, \\ \big\{[\![r\mathbf{s}[i,j]]\!]_{\mathsf{t}}\big\}_{i\in[0;n-1]}^{j\in\Pi_{\mathbf{x}_i}}\end{array}$$

where $[\![r]\!]_1\leftarrow\mathsf{H}(\mathsf{lab})$ and $\mathbf{s}[i,j]$ denotes the entry of the share vector corresponding to the $j$-th coordinate of $\mathbf{x}_i$ for $i\in[0;n-1]$ and $j\in[k]$. The security notion that we can achieve for this scheme suffers from the same limitations as [22,41]; most importantly, we cannot prove security under repetitions. Moreover, not being able to prove security under repetitions implies that the encryption algorithm must take a secret key, as otherwise the adversary could create multiple ciphertexts under the same label by herself. The reason for these restrictions is the fact that our only source of randomness is the random oracle whose only input is the label. Hence, to achieve security in a stronger model, our first step is to remove the random oracle from the construction.

[4] Ciphertexts and decryption keys corresponding to the $j$-th IPFE scheme of client $i$ are denoted by $\mathsf{iCT}_{i,j}$ and $\mathsf{iDK}_{i,j}$.

**Removing the Random Oracle and Enabling Public Encryption.** In our first attempt above, the random oracle provides common randomness across independently generated ciphertexts and keys. Clearly, this is not possible anymore without a random oracle. Therefore, it seems inevitable to have one client (say, the 0-th) generate all the IPFE ciphertexts $\{\mathsf{iCT}_{i,j}(\llbracket r \rrbracket_1)\}_{i,j}$. However, when generating $\mathsf{CT}_{\mathbf{x}_0}$, the vectors $\mathbf{x}_1, \ldots, \mathbf{x}_{n-1}$ are unknown, so it is unclear which $\mathsf{iCT}_{i,j}(\llbracket r \rrbracket_1)$ for $i > 0$ should be included in $\mathsf{CT}_{\mathbf{x}_0}$.

As a solution, we let client 0 generate *all* ciphertexts $\{\mathsf{iCT}_{i,j}(\llbracket r \rrbracket_1)\}_{i \in [n-1]}^{j \in [k]}$, but instead of providing them "in the clear", we hide them with an additional layer of identity-based encryption. Specifically, the 0-th client encrypts each $\mathsf{iCT}_{i,j}(\llbracket r \rrbracket_1)$ with respect to the identity $(\mathsf{lab}, j)$ using the public key of client $i$. Correspondingly, client $i \in [n-1]$ provides the identity-based decryption keys $\mathsf{idDK}_i(\mathsf{lab}, j)$ for each $j \in \mathit{\Pi}_{\mathbf{x}_i}$ needed to recover the IPFE ciphertexts generated by client 0. This idea yields the following MC-ABE for LSS in the standard model:

$$\left.\begin{array}{ll} \mathsf{CT}_{\mathbf{x}_0}: & \begin{bmatrix} \mathsf{iCT}_0(\llbracket r, \mu \rrbracket_1), \left\{\mathsf{iCT}_{0,j}(\llbracket r \rrbracket_1)\right\}_{j \in \mathit{\Pi}_{\mathbf{x}_0}} \\ \left\{\mathsf{idCT}_i\big((\mathsf{lab}, j), \mathsf{iCT}_{i,j}(\llbracket r \rrbracket_1)\big)\right\}_{i \in [n-1]}^{j \in [k]} \end{bmatrix} \\ \mathsf{DK}_{\mathbf{x}_i}: & \left\{\mathsf{idDK}_i(\mathsf{lab}, j)\right\}_{j \in \mathit{\Pi}_{\mathbf{x}_i}} \\ \mathsf{DK}_f: & \mathsf{iDK}_0(\llbracket s, 1 \rrbracket_2), \left\{\mathsf{iDK}_{i,j}(\llbracket \mathbf{s}[i,j] \rrbracket_2)\right\}_{i \in [0;[n-1]}^{j \in [k]} \end{array}\right\} \begin{array}{l} \llbracket r \cdot s + \mu \rrbracket_{\mathsf{t}}, \\ \left\{\llbracket r \cdot \mathbf{s}[i,j] \rrbracket_{\mathsf{t}}\right\}_{i \in [0;n]}^{j \in \mathit{\Pi}_{\mathbf{x}_i}} \end{array} \tag{1}$$

Here, $r \xleftarrow{\$} \mathbb{Z}_p$ is a fresh random scalar for each ciphertext. Due to this fact, the scheme remains secure even under several encryption queries for the same label and, in particular, enables a public encryption algorithm. Indeed, if each encryption samples a fresh $r \xleftarrow{\$} \mathbb{Z}_p$, then each message $\llbracket \mu \rrbracket_{\mathsf{t}}$ is hidden by a fresh looking mask $\llbracket r \cdot s \rrbracket_{\mathsf{t}}$. So the ability to create ciphertexts by herself does not help the adversary to recover information from a challenge ciphertext anymore.

On the negative side, the scheme in (1) is still not secure under repetitions for slots $i \in [n-1]$. For example, consider an adversary that submits queries for decryption keys $\mathsf{DK}_{\mathbf{x}_i}$ and $\mathsf{DK}_{\mathbf{x}'_i}$ for two attribute vectors $\mathbf{x}_i, \mathbf{x}'_i \in \{0,1\}^k$. Then $\mathsf{DK}_{\mathbf{y}_i} := \mathsf{DK}_{\mathbf{x}_i} \cup \mathsf{DK}_{\mathbf{x}'_i}$ is a decryption key for the vector $\mathbf{y}_i \in \{0,1\}^k$ having a 1 in all coordinates $j \in [k]$ where $1 \in \{\mathbf{x}[j], \mathbf{x}'[j]\}$. Thus, $\mathsf{DK}_{\mathbf{y}_i}$ may be used to decrypt ciphertexts that cannot be decrypted by neither $\mathsf{DK}_{\mathbf{x}_i}$ nor $\mathsf{DK}_{\mathbf{x}'_i}$.

**MC-ABE for LSS With Repetitions.** To achieve security under repetitions for slots $i \in [n-1]$, we must make sure that multiple decryption keys for the same label-slot pair $(\mathsf{lab}, i)$ cannot be combined in a meaningful way as it is possible for the scheme in (1). In other words, all components of a decryption key $\mathsf{DK}_{\mathbf{x}_i}$ should "depend on" the entire vector $\mathbf{x}_i$ instead of only a single coordinate $\mathbf{x}_i[j]$. To this end, we now let $\mathsf{DK}_{\mathbf{x}_i} = \mathsf{idDK}_i(\mathsf{lab}, \mathbf{x}_i)$ as opposed to $\{\mathsf{idDK}_i(\mathsf{lab}, j)\}_{j \in \mathit{\Pi}_{\mathbf{x}_i}}$. Then security of the MC-ABE under repetitions directly corresponds to the collusion resistance of the employed IBE.

On the other hand, correctness is no longer straightforward. This is because a successful decryption using the new keys requires the 0-th client to provide encryptions of the IPFE ciphertexts $\{\mathsf{iCT}_{i,j}(\llbracket r \rrbracket_1)\}_{i,j}$ with respect to identities that depend on attribute vectors $\mathbf{x}_1, \ldots, \mathbf{x}_{n-1}$. These vectors are not given as input and, thus, are unknown at encryption time. Moreover, decryption with a key $\mathsf{DK}_f$ is supposed to work with *any* combination of $\mathbf{x}_1, \ldots, \mathbf{x}_{n-1}$ satisfying $f(\mathbf{x}_0, \ldots, \mathbf{x}_{n-1}) = 1$. Therefore, the problem is not only that these attribute vectors are unknown, but in general there can be many possible choices that should allow decrypting. In particular, when $f$ is the constant function that always outputs 1, then decryption must succeed for any choice of $\mathbf{x}_1, \ldots, \mathbf{x}_{n-1}$. This observation ultimately forces the encryptor to provide encryptions of the IPFE ciphertexts $\{\mathsf{iCT}_{i,j}([\![r]\!]_1)\}_j$ with respect to every identity $(\mathsf{lab}, \mathbf{x}_i)$ such that $\mathbf{x}_i \in \{0,1\}^k$, for $i \in [n-1]$. More precisely, a ciphertext $\mathsf{CT}_{\mathbf{x}_0}$ for a message $\mu$ consists of the following components:

$$\mathsf{CT}_{\mathbf{x}_0} : \left\{ \begin{matrix} \mathsf{iCT}_0([\![r_{\mathbf{x}}, \mu]\!]_1), \left\{\mathsf{iCT}_{0,j}([\![r_{\mathbf{x}}]\!]_1)\right\}_{j \in \Pi_{\mathbf{x}_0}}, \\ \left\{\mathsf{idCT}_i\big((\mathsf{lab}, \mathbf{x}_i), \mathsf{iCT}_{i,j}([\![r_{\mathbf{x}}]\!]_1)\big)\right\}_{i \in [n-1]}^{j \in \Pi_{\mathbf{x}_i}} \end{matrix} \right\}_{\mathbf{x} = (\mathbf{x}_1, \ldots, \mathbf{x}_{n-1}) \in \{0,1\}^{(n-1)k}} \tag{2}$$

where $r_{\mathbf{x}} \xleftarrow{\$} \mathbb{Z}_p$ for each $\mathbf{x} \in \{0,1\}^{(n-1)k}$. It is clear that these ciphertexts have exponential size if $k$ or $n$ are chosen too large. However, it remains polynomial if one chooses *e.g.* $k \cdot n = O(\log \lambda)$ which gives $|\{0,1\}^{(n-1)k}| = \mathrm{poly}(\lambda)$.

**Upgrading the Policy Class to $\mathsf{NC}^1$.** Let $f$ be a policy specified by an $\mathsf{NC}^1$ circuit over the variables $(\mathbf{x}_0, \ldots, \mathbf{x}_{n-1}) \in \{0,1\}^{[0;n-1] \times [k]}$. We can view $f$ as a Boolean formula consisting of (fan-in 1) $\neg$ gates and (fan-in 2) $\wedge$ and $\vee$ gates. Using De Morgan laws, we can push the $\neg$ gates to the leaves such that all internal nodes consist only of $\wedge$ and $\vee$ gates, while leaves are labeled by either attributes or their negations. In this way, we obtain a monotone formula $\overline{f} \colon \{0,1\}^{[0;n-1] \times [k] \times \{0,1\}} \to \{0,1\}$ that is "equivalent" to $f$ in the following sense. For each $\mathbf{x} \in \{0,1\}^{[k]}$, we define the *extended* vector $\overline{\mathbf{x}} \in \{0,1\}^{[k] \times \{0,1\}}$ component-wise via $\overline{\mathbf{x}}[(j,1)] = \mathbf{x}[j]$ and $\overline{\mathbf{x}}[(j,0)] = 1 - \mathbf{x}[j]$ for each $j \in [k]$[5]. Then we have $f(\mathbf{x}_0, \ldots, \mathbf{x}_{n-1}) = \overline{f}(\overline{\mathbf{x}}_0, \ldots, \overline{\mathbf{x}}_{n-1})$ for each $(\mathbf{x}_0, \ldots, \mathbf{x}_{n-1})$. Lewko and Waters [34] presented an LSS for all monotone access structures which implies that $\overline{f}$ can be captured by an LSS.

Given an MC-ABE $\overline{\mathsf{aFE}}$ for LSS policies $\overline{f} \colon \{0,1\}^{[0;n-1] \times [k] \times \{0,1\}} \to \{0,1\}$, we can build an MC-ABE $\mathsf{aFE}$ for $\mathsf{NC}^1$ policies $f \colon \{0,1\}^{[0;n-1] \times [k]} \to \{0,1\}$ by simply replacing the inputs $\mathbf{x}_0, \ldots, \mathbf{x}_{n-1}$ and $f$ with $\overline{\mathbf{x}}_0, \ldots, \overline{\mathbf{x}}_{n-1}$ and $\overline{f}$. In general, $\mathsf{aFE}$ is only secure if the adversary is not allowed to corrupt users. To see this, we first note that there exist vectors $\overline{\mathbf{x}} \in \{0,1\}^{[k] \times \{0,1\}}$ that are not an extension of a vector $\mathbf{x} \in \{0,1\}^{[k]}$. More precisely, $\overline{\mathbf{x}}$ is an extension of some $\mathbf{x}$ if and only if $\overline{\mathbf{x}}[j,0] = 1 - \overline{\mathbf{x}}[j,1]$ for all $j \in [k]$. Let us call such vectors $\overline{\mathbf{x}}$ *valid*. By construction, an $\mathsf{aFE}$ decryption key for a vector $\mathbf{x}_i$ is an $\overline{\mathsf{aFE}}$ decryption key for the extended vector $\overline{\mathbf{x}}_i$. Therefore, to reduce the security of $\mathsf{aFE}$ to the security of $\overline{\mathsf{aFE}}$, we must argue that the adversary cannot obtain $\overline{\mathsf{aFE}}$ decryption keys for vectors that are not valid. Without corruptions, this is easy to see. However, if

[5] We can think of $\overline{\mathbf{x}}$ as a vector of length $2k$ whose coordinates are indexed by the set $[k] \times \{0,1\}$ for convenience.

the adversary can obtain a client's secret key, then this is no longer the case, as the adversary could generate decryption keys for invalid vectors by herself. Thus, using this conversion *generically*, we can convert the schemes in (1) and (2) into MC-ABEs for $\mathsf{NC}^1$ without corruptions.

Moreover, we can even achieve security with corruptions when performing a *concrete* security analysis for the scheme in (2). For this, we recall that the secret key $\mathsf{SK}_i$ of some client $i \in [n-1]$ consists of an IBE master secret key that is used to generate decryption keys for identities $(\mathsf{lab}, \overline{\mathbf{x}}_i)$. Even though this key could be maliciously used to generate decryption keys for invalid vectors $\overline{\mathbf{x}}_i$, this does not help to win the security game as these identities do not occur in the challenge ciphertext.

**Other Policy Classes.** We recall from (2) that our scheme becomes inefficient when we choose $k, n = \text{poly}(\lambda)$ since then there is an exponential number of $\mathbf{x} = (\mathbf{x}_1, \dots, \mathbf{x}_{n-1}) \in \{0,1\}^{[n-1]\times[k]}$. Nevertheless, there exist nontrivial subclasses of $\mathsf{NC}^1$ which do not require to consider all $\mathbf{x}$ during the encryption procedure.

- $\mathsf{NC}^0$ *Policies.* The output of an $\mathsf{NC}^0$ policy $f$ depends only on a set $L$ of size $\tau = O(1)$ out of the total of $kn$ inputs. As the remaining $kn - \tau$ inputs can be chosen arbitrarily without changing the output, it suffices to consider vectors $\mathbf{x}$ that are 0 outside $L$. There are $\binom{kn}{\tau} \leq (kn)^\tau = \text{poly}(\lambda)$ possible sets $L$ and for each choice we only need to consider $2^\tau = O(1)$ vectors $\mathbf{x}$.
- *Threshold Policies with Constant Threshold.* Threshold policies with a threshold $\tau \leq O(1)$ are not in $\mathsf{NC}^0$ as they depend on *all* $kn$ inputs. However, they have the property that each authorized set also has an authorized subset of size $\tau$. This allows an argument similar to above where we only deal with subsets $L$ of size $\tau$. Symmetrically, we can also handle policies with a threshold $kn - \tau$ where we consider $\binom{kn}{kn-\tau} \leq (kn)^\tau = \text{poly}(\lambda)$ sets $L$ of size $kn - \tau$.

While the idea is simple, the concrete implementation requires some care because it must be guaranteed that the choices of $L$ in the 0-th client remain compatible with the IBE keys provided by the other clients. For details, please see Sect. 5.

**Security.** To get a grasp of the security proof, it is instructive to first consider the case where the IPFE is simulation secure. This means the only values that the adversary learns are

- encodings $[\![s]\!]_2$ of random scalars $s \overset{\$}{\leftarrow} \mathbb{Z}_p$ sampled during the key generation and their corresponding share vectors $[\![\mathbf{s}]\!]_2$, and
- target group encodings of the form $[\![r \cdot s + \mu]\!]_\mathrm{t}$ (IPFE instance 0) and $[\![r \cdot \mathbf{s}[i,j]]\!]_\mathrm{t}$ (IPFE instance $(i,j)$) for random scalars $r \overset{\$}{\leftarrow} \mathbb{Z}_p$ sampled during the encryption of the challenge message.

Importantly, nothing about $r$ is leaked in $\mathbb{G}_1$. So we can rely on the $\mathsf{DDH}$ to obtain a fresh looking mask $[\![r \cdot s]\!]_\mathrm{t}$ with a fresh share vector $[\![r \cdot \mathbf{s}]\!]_2$ for each combination of $r$ and $s$. Then we can exploit the one-time security provided by the LSS scheme to replace the individual secret sharings with random values: by the admissibility of the adversary, there does not exist any pair $(r, s)$ such that the adversary has sufficient information about a subset of shares that allows her to recover the mask $[\![r \cdot s]\!]_{\mathrm{t}}$. Instead, they look uniformly random and, thus, perfectly hide the challenge message.

Unfortunately, simulation security for many ciphertexts is known to be impossible in the standard model [18]. Therefore, we can only rely on an IPFE with indistinguishability-based security. This makes the proof slightly more complex since we cannot directly conclude anymore that the adversary learns the scalars $r$ only as part of the inner products encoded in $\mathbb{G}_{\mathrm{t}}$. To circumvent this problem, we use a primitive called *slotted IPFE* [38] which is a mix between public-key and private-key IPFE that provides standard security on the public part and additionally hides the function vectors in the private part. Using this primitive, we can move the scalar $r$ from the message vectors encoded in $\mathbb{G}_1$ into a hidden coordinate of the function vectors in $\mathbb{G}_2$. Subsequently, we can rely on the $\mathsf{DDH}$ in $\mathbb{G}_2$ and proceed with the proof as in the case of simulation security.

Finally, we want to mention an important detail that occurs during the security proof. The adversary's admissibility condition only covers the case when she obtains at least one key for each slot (via either corruption or attribute key generation queries). Therefore, we must protect against so-called incomplete queries, where the adversary does not submit a query for every slot, but still has sufficient information to decrypt. In the context of IPFE (without access control) this can be done using a primitive called all-or-nothing encoding [23]. In the context of attribute-based MIFE for attribute-weighted sums, [9] uses a ciphertext-policy ABE for arithmetic branching programs which was recently proposed by Lin and Luo [39]. In our case, we can avoid the usage of a complex primitive like ABE because we can model the completeness condition as part of our policies. This is feasible since our construction can check a global condition before releasing any information. Previous works considered only conjunctions of local checks in each slot which is not powerful enough to verify completeness.

## 2.2 MC-PE from MC-ABE and Lockable Obfuscation

**Lockable Obfuscation.** We make use of a primitive called *lockable obfuscation* (LO) [30,49]. Roughly speaking, LO allows to obfuscate a circuit $C$ with respect to a message $\mu$ and a lock value $\sigma$. Correctness asks that an evaluation of the obfuscated circuit on some input $x$ yields $\mu$ if $C(x) = \sigma$ and $\perp$ otherwise. Simulation security requires that if $\sigma$ looks random to the adversary, then the obfuscated circuit is computationally indistinguishable from a garbage program that does not carry any information about $\mu$ or $C$.

**The Compilers of** [11]**.** The authors of [11] present two compilers from MI-ABE to MI-PE which nest several obfuscated circuits in a sophisticated manner. Very roughly, the obfuscated circuit $\widetilde{C}_0$ for the zeroth slot takes as input another obfuscated circuit $\widetilde{C}_1$ for the first slot, which in turn takes an obfuscated circuit $\widetilde{C}_2$ for the second slot and so on until one arrives at the last slot $n-1$. $\widetilde{C}_0$ is generated with respect to an attribute $x_0$ and a message $\mu$ whereas the other $\widetilde{C}_i$'s only depend on an attribute $x_i$. The crucial part of the construction is to establish "communication" between consecutive circuits without violating attribute privacy. The idea is to build a recursive evaluation chain where the innermost circuit checks the condition $f(x_0, \ldots, x_{n-1}) = 1$ using the MI-ABE; and a successful evaluation of an obfuscated circuit $\widetilde{C}_i$, for $i \in [n-1]$, unlocks the lock and reveals a secret which is needed for a successful evaluation of $\widetilde{C}_{i-1}$.

In their first compiler, these secret values are *global* secrets. This leads to a straightforward construction as all clients know these common secrets when they obfuscate their circuits. However, the supported security model is weak. This is because once the adversary submits any combination of oracle queries that enables a valid decryption process, these global secrets are revealed and security collapses even if all involved oracle queries have the same left and right input. To achieve security in a stronger model, their second compiler avoids these global secrets. However, this makes the construction more complex, and they are able to deal with only two slots. Our new construction can be viewed as a generalization of this arity-2 compiler to any constant arity. We therefore recall the arity-2 construction as a warm-up.

**Construction in the Two-Input Setting.** We start from an MI-ABE $(\mathsf{aSetup}, \mathsf{aEnc}, \mathsf{aAKeyGen}, \mathsf{aPKeyGen}, \mathsf{aDec})$. For notational convenience, we use the shorthand notations $\mathsf{aCT}_\ell(x_0, \mu)$, $\mathsf{aDK}_\ell(i, x_i)$ and $\mathsf{aDK}_\ell(f)$ to denote executions of $\mathsf{aEnc}_\ell(\mathsf{aMPK}_\ell, x_0, \mu)$, $\mathsf{aAKeyGen}(\mathsf{aSK}_{\ell,i}, x_i)$ and $\mathsf{aPKeyGen}(\mathsf{aMSK}_\ell, f)$, for two independently generated $\mathsf{aFE}$ instances $(\mathsf{aMPK}_\ell, \mathsf{aMSK}_\ell, \{\mathsf{aSK}_{\ell,i}\}_i) \leftarrow \mathsf{aSetup}(1^\lambda)$ indexed by $\ell \in \{0, 1\}$. The MI-PE encryptor (client 0) possesses $(\mathsf{aMPK}_0, \mathsf{aSK}_1)$ and the attribute key generator (client 1) possesses $(\mathsf{aMPK}_1, \mathsf{aSK}_0)$. The master secret key contains the ABE master secret keys $(\mathsf{aMSK}_0, \mathsf{aMSK}_1)$. To encrypt a message $\mu$ with respect to an attribute $x_0$, client 0 samples a random lock value $\sigma_0$ and computes $\mathsf{aCT}_0(x_0, \sigma_0)$ and $\mathsf{aDK}_1(1, x_0)$. The final ciphertext is an obfuscation $\widetilde{C}_0$ of a circuit $C_0[\mathsf{aCT}_0(x_0, \sigma_0), \mathsf{aDK}_1(1, x_0)]$ with respect to the message $\mu$ and lock value $\sigma_0$. The notation $C[\alpha]$ indicates that the value $\alpha$ is hardwired in the description of the circuit $C$. Similarly, to produce a decryption key with respect to an attribute $x_1$, client 1 samples a lock value $\sigma_1$, generates $\mathsf{aCT}_1(x_1, \sigma_1)$ and $\mathsf{aDK}_0(1, x_1)$ and outputs an obfuscation $\widetilde{C}_1$ of a circuit $C_1[\mathsf{aCT}_1(x_1, \sigma_1)]$ with respect to the message $\mathsf{aDK}_0(1, x_1)$ and lock value $\sigma_1$. An MI-PE decryption key consists of a set of MI-ABE decryption keys $\{\mathsf{aDK}_0(f), \mathsf{aDK}_1(f)\}$.

The pivotal point that makes the whole scheme work is the definition of the circuits. Specifically, decryption evaluates the obfuscated outer circuit $\widetilde{C}_0$ on input the obfuscated inner circuit $\widetilde{C}_1$ and the MI-ABE keys $\{\mathsf{aDK}_0(f), \mathsf{aDK}_1(f)\}$. Suppose that $f(x_0, x_1) = 1$. For a successful decryption, we must unlock $\widetilde{C}_0$. The lock value $\sigma_0$ is already hardwired in the circuit $C_0[\mathsf{aCT}_0(x_0, \sigma_0), \mathsf{aDK}_1(1, x_0)]$, however it is hidden in the ciphertext $\mathsf{aCT}_0(x_0, \sigma_0)$. To decrypt this ciphertext, we need the decryption key $\mathsf{aDK}_0(1, x_1)$ embedded in $\widetilde{C}_1$. For this reason $C_0[\mathsf{aCT}_0(x_0, \sigma_0), \mathsf{aDK}_1(1, x_0)]$ starts by evaluating $\widetilde{C}_1$ on input $(\mathsf{aDK}_1(1, x_0), \mathsf{aDK}_1(f))$. From its inputs, the inner circuit $C_1[\mathsf{aCT}_1(x_1, \sigma_1)]$ obtains everything it needs to decrypt its hardwired ciphertext $\mathsf{aCT}_1(x_a, \sigma_1)$ and to recover the correct lock value $\sigma_1$ which unlocks $\widetilde{C}_1$ and reveals $\mathsf{aDK}_0(1, x_1)$. At this point, $C_0[\mathsf{aCT}_0(x_0, \sigma_0), \mathsf{aDK}_1(1, x_0)]$ can perform a similar computation by decrypting the ciphertext $\mathsf{aCT}_0(x_0, \sigma_0)$ and recovering $\sigma_0$. This eventually unlocks $\widetilde{C}_0$ and outputs $\mu$. Importantly, this construction does not use global secrets, hence its security is not compromised after one successful decryption.

**Generalization to Constant-Arity MC-ABE.** Our new compiler generalizes this framework to more than two slots and the more general MC-PE model. We will use independent MC-ABE instances for each slot to check if decryption is permitted, and each MC-PE client holds the key of one slot from each MC-ABE instance. Specifically, we let client $i \in [0; n-1]$ control

- the $i$-th slot of the MC-ABE instances $\ell \in [0; i-1]$,
- the 0-th slot of the MC-ABE instance $\ell = i$, and
- the $(i+1)$-th slot of the MC-ABE instances $\ell \in [i+1; n-1]$.

In particular, we note that each MC-PE client is the encryptor in exactly one of the MC-ABE schemes.

To encrypt a message $\mu$ with respect to a label $\mathsf{lab}$ and an attribute $x_0$, client 0 samples a random lock value $\sigma_0$ and creates $\mathsf{aCT}_0(\mathsf{lab}, x_0, \sigma_0)$ and $\mathsf{aDK}_\ell(\mathsf{lab}, 1, x_0)$ for all $\ell \in [n-1]$. Then, it issues an obfuscation of a circuit $C_0[\mathsf{aCT}_0(\mathsf{lab}, x_0, \sigma_0), \{\mathsf{aDK}_\ell(\mathsf{lab}, 1, x_0)\}_{\ell \in [n-1]}]$ generated with respect to the message $\mu$ and the lock value $\sigma_0$. Similarly, to generate a key for a label $\mathsf{lab}$ and an attribute $x_i$, client $i \in [n-1]$ samples a lock value $\sigma_i$ and creates $\mathsf{aCT}_i(\mathsf{lab}, x_i, \sigma_i)$, $\mathsf{aDK}_\ell(\mathsf{lab}, i, x_i)$ for $\ell \in [0; i-1]$, and $\mathsf{aDK}_\ell(\mathsf{lab}, i+1, x_i)$ for $i \in [i+1; n-1]$. Then it outputs an obfuscation of a circuit $C_i[\mathsf{aCT}_i, \{\mathsf{aDK}_\ell(\mathsf{lab}, i+1, x_i)\}_{\ell \in [i+1;n-1]}]$ generated with respect to the message $\{\mathsf{aDK}_\ell(\mathsf{lab}, i, x_i)\}_{\ell \in [0;i-1]}$ and lock value $\sigma_i$. Decryption keys for a policy $f$ are a set of MC-ABE decryption keys $\{\mathsf{aDK}_\ell(f)\}_{\ell \in [0;n-1]}$.

As in the two-input case, the crucial point is to establish communication between the obfuscated circuits in a secure way. However, the nested evaluations become more complex now. We first observe the following properties satisfied by all obfuscated circuits $\widetilde{C}_i$ for $i \in [0; n-1]$:

1. Decryption keys $\mathsf{aDK}_\ell(\mathsf{lab}, i+1, x_i)$ for $\ell > i$ are hardwired in the description of the circuit. This means they can be accessed during the evaluation of $\widetilde{C}_i$ and passed as input to the evaluation of $\widetilde{C}_j$ for $j > i$.
2. Decryption keys $\mathsf{aDK}_\ell(\mathsf{lab}, i, x_i)$ for $\ell < i$ are stored as the message of $\widetilde{C}_i$ which is revealed in case of a successful evaluation. This means they can be recovered and used during the evaluation of $\widetilde{C}_j$ for $j < i$.

Suppose that $f(x_0, \ldots, x_{n-1}) = 1$. Decryption evaluates $\widetilde{C}_0$ on input the obfuscated circuits $\{\widetilde{C}_i\}_{i \in [n-1]}$ and the MC-ABE keys $\{\mathsf{aDK}_\ell(f)\}_{\ell \in [0;n-1]}$. The lock value of $\widetilde{C}_0$ is hidden in its hardwired ciphertext $\mathsf{aCT}_0(\mathsf{lab}, x_0, \sigma_0)$. To decrypt this ciphertext, we need the keys $\{\mathsf{aDK}_0(\mathsf{lab}, i, x_i)\}_{i \in [n-1]}$ stored in the messages of $\{\widetilde{C}_i\}_{i \in [n-1]}$, so we need to evaluate them first. Specifically, via a chain of recursive calls where each $\widetilde{C}_i$ invokes the evaluation of $\widetilde{C}_{i+1}$, we arrive at the evaluation of $\widetilde{C}_{n-1}$. From property 1 , it follows that $\widetilde{C}_{n-1}$

receives as input all the keys $\{\mathsf{aDK}_{n-1}(\mathsf{lab}, i, x_i)\}_{i\in[n-1]}$ to decrypt its hardwired ciphertext $\mathsf{aCT}_{n-1}(\mathsf{lab}, x_{n-1}, \sigma_{n-1})$ and to recover the lock value $\sigma_{n-1}$. In this way, $\widetilde{C}_{n-1}$ can be unlocked and its message revealed. In the next step, it follows from property 2 that now the evaluation of $\widetilde{C}_{n-2}$ has everything it needs to perform a similar computation to recover $\sigma_{n-2}$ and unlock $\widetilde{C}_{n-2}$, and so on.

While at first glance it may seem that this decryption procedure is efficient for any (polynomial) number of slots, there is a subtle problem: each obfuscation increases the size of the circuit by a polynomial factor. As we nest the evaluation of the circuits, this leads to an exponential blow-up in the number of slots. Therefore, the decryption algorithm is only efficient for $n = O(1)$, *i.e.* constant arity.

**Security.** The security proof is a simple sequence of hybrids over all slots from $n - 1$ to 0. In each hybrid, if $f(x_0, \ldots, x_{n-1}) = 0$, then we can rely on the security of the $i$-th MC-ABE instance to replace the ciphertext $\mathsf{aCT}_i(x_i, \sigma_i)$ hardwired in $\widetilde{C}_i$ with a ciphertext of the zero string $\mathsf{aCT}_i(x_i, 0)$. Then the lock value $\sigma_i$ appears random to the adversary and the obfuscated circuit $\widetilde{C}_i$ can be replaced with a simulated obfuscation that carries no information about $x_i$. In the last step, we replace $\widetilde{C}_0$ with a simulation that erases all information about $x_0$ and $\mu$.

## 3 Preliminaries

### 3.1 Notational Conventions

Let $\lambda \in \mathbb{N}$ be the security parameter. Except in the definitions, we will suppress $\lambda$ in subscripts for brevity. A nonnegative function $\varepsilon \colon \mathbb{N} \to \mathbb{R}$ is negligible if $\varepsilon(\lambda) = O(\lambda^{-n})$ for all $n \in \mathbb{N}$. An algorithm is said to be *efficient* if it runs in probabilistic polynomial time (PPT) in the security parameter.

To avoid confusion, we always write vectors $\mathbf{v}$ and matrices $\mathbf{A}$ in boldface and use uppercase letters for the latter. Scalars $s$ are written in italics. Unless otherwise stated, all vectors $\mathbf{v}$ are viewed as column vectors. The corresponding row vector is denoted by $\mathbf{v}^\top$.

**Sets and Indexing.** We denote by $\mathbb{Z}$ and $\mathbb{N}$ the sets of integers and natural numbers (positive integers). For integers $m$ and $n$, we write $[m; n]$ to denote the set $\{z \in \mathbb{Z} : m \leq z \leq n\}$ and let $[n] := [1; n]$. For a prime number $p$, $\mathbb{Z}_p$ denotes the finite field of integers modulo $p$. For a finite set $S$, we let $2^S$ denote the power set of $S$.

To index a vector or the columns of a matrix, we write $\mathbf{v}[i]$ and $\mathbf{A}[j]$. In contrast, objects of some collection that is not regarded as a vector or matrix are indexed using subscripts (or superscripts in some cases). For instance, $\mathbf{v}_i$ represents a vector, not a component of some vector. If $i$ runs through some index set $[n]$, it means that there are $n$ vectors $\mathbf{v}_1, \ldots, \mathbf{v}_n$. If the $n$ objects are scalars (or not explicitly vectors), we will write $v_1, \ldots, v_n$ instead.

For convenience, objects might be indexed by arbitrary sets, not just integers. For finite sets $\mathfrak{s}, A$, we write $A^{\mathfrak{s}} := \{(\mathbf{v}[i])_{i\in\mathfrak{s}} : \mathbf{v}[i] \in A\}$ for the set of vectors whose entries are in $A$ and indexed by $\mathfrak{s}$, *e.g.* $\mathbb{Z}_p^{[n]}$ is just $\mathbb{Z}_p^n$. Suppose $\mathfrak{s}_1, \mathfrak{s}_2$ are two index sets with $\mathfrak{s}_1 \subseteq \mathfrak{s}_2$. For a vector $\mathbf{v} \in \mathbb{Z}_p^{\mathfrak{s}_2}$, we denote by $\mathbf{u} = \mathbf{v}|_{s_1}$ its canonical projection onto $\mathbb{Z}_p^{\mathfrak{s}_1}$, *i.e.* $\mathbf{u} \in \mathbb{Z}_p^{\mathfrak{s}_1}$ and $\mathbf{u}[i] = \mathbf{v}[i]$ for all $i \in \mathfrak{s}_1$. Conversely, for any vector $\mathbf{u} \in \mathbb{Z}_p^{\mathfrak{s}_1}$, we write $\mathbf{v} = \mathbf{u}|^{\mathfrak{s}_2}$ for its zero-extension into $\mathbb{Z}_p^{\mathfrak{s}_2}$, *i.e.* $\mathbf{v} \in \mathbb{Z}_p^{\mathfrak{s}_2}$ and $\mathbf{v}[i] = \mathbf{u}[i]$ if $i \in \mathfrak{s}_1$, and $\mathbf{v}[i] = 0$ if $i \in \mathfrak{s}_2 \setminus \mathfrak{s}_1$.

## 3.2 Pairing Groups and Hardness Assumptions

**Pairing Groups.** Our constructions use a sequence of pairing groups

$$\mathbb{G} = \{\mathbb{G}_\lambda = (\mathbb{G}_{\lambda,1}, \mathbb{G}_{\lambda,2}, \mathbb{G}_{\lambda,\mathrm{t}}, g_{\lambda,1}, g_{\lambda,2}, g_{\lambda,\mathrm{t}}, e_\lambda, p_\lambda)\}_{\lambda\in\mathbb{N}} \ ,$$

where $\mathbb{G}_{\lambda,1}$ (resp. $\mathbb{G}_{\lambda,2}$, $\mathbb{G}_{\lambda,\mathrm{t}}$) is a cyclic group of order $p_\lambda$ generated by $g_{\lambda,1}$ (resp. $g_{\lambda,2}$, $g_{\lambda,\mathrm{t}}$), and $e_\lambda \colon \mathbb{G}_{\lambda,1} \times \mathbb{G}_{\lambda,2} \to \mathbb{G}_{\lambda,\mathrm{t}}$ is the pairing operation satisfying $e_\lambda(g_{\lambda,1}^a, g_{\lambda,2}^b) = g_{\lambda,\mathrm{t}}^{ab}$ for all integers $a, b$. The group operations and the pairing map are required to be efficiently computable.

Following the implicit notation in [25], we write $[\![a]\!]_i$ to denote $g_{\lambda,i}^a$ for $i \in \{1, 2, \mathsf{t}\}$. This notation extends component-wise to matrices and vectors having entries in $\mathbb{Z}_p$. Equipped with these notations, group operations are written additively and the pairing operation multiplicatively, *e.g.* $[\![\mathbf{A}]\!]_1 - \mathbf{B}[\![\mathbf{C}]\!]_1\mathbf{D} = [\![\mathbf{A} - \mathbf{BCD}]\!]_1$ and $[\![\mathbf{A}]\!]_1[\![\mathbf{B}]\!]_2 = [\![\mathbf{AB}]\!]_\mathsf{t}$.

**Computational Assumptions.** We state the assumptions needed for our constructions. Let $\{\mathbb{G}_\lambda = (\mathbb{G}_{\lambda,1}, \mathbb{G}_{\lambda,2}, \mathbb{G}_{\lambda,\mathrm{t}}, g_{\lambda,1}, g_{\lambda,2}, g_{\lambda,\mathrm{t}}, e_\lambda, p_\lambda)\}_{\lambda\in\mathbb{N}}$ be a sequence of pairing groups.

**Definition 1 (Decisional Diffie-Hellman Assumption (DDH)).** *Let $i \in \{1, 2, t\}$. The* DDH *assumption holds in $\{\mathbb{G}_{\lambda,i}\}_{\lambda\in\mathbb{N}}$ if $\{[\![a, b, ab]\!]_i\}_{\lambda\in\mathbb{N}} \approx_c \{[\![a, b, ab + c]\!]_i\}_{\lambda\in\mathbb{N}}$ for $a, b, c \xleftarrow{\$} \mathbb{Z}_{p_\lambda}$.*

**Definition 2 (Symmetric eXternal Diffie-Hellman Assumption (SXDH)).** *The* SXDH *assumption holds in $\{\mathbb{G}_\lambda\}_{\lambda\in\mathbb{N}}$ if the* DDH *assumption holds in both $\{\mathbb{G}_{\lambda,1}\}_{\lambda\in\mathbb{N}}$ and $\{\mathbb{G}_{\lambda,2}\}_{\lambda\in\mathbb{N}}$.*

## 3.3 Monotone Access Structures and Linear Secret Sharing Schemes

Let $\mathcal{X} = \{0, 1\}^{\mathfrak{s}}$ be the attribute universe with index set $\mathfrak{s}$. An access structure on $\mathcal{X}$ is a collection $\mathcal{S} \subseteq 2^{\mathfrak{s}} \setminus \varnothing$ of nonempty subsets of $\mathfrak{s}$. We call the sets in $\mathcal{S}$ authorized, and those in $2^{\mathfrak{s}} \setminus \mathcal{S}$ unauthorized. Each access structure $\mathcal{S}$ corresponds to an access policy $f \colon \mathcal{X} \to \{0, 1\}$ defined via

$$f(\mathbf{x}) = \begin{cases} 1 & \text{if } \{i \in \mathfrak{s} : \mathbf{x}[i] = 1\} \in \mathcal{S} \\ 0 & \text{if } \{i \in \mathfrak{s} : \mathbf{x}[i] = 1\} \notin \mathcal{S} \ . \end{cases}$$

An access structure $\mathcal{S} \subseteq 2^{\mathfrak{s}}$ is said to be *monotone* if the following condition is satisfied for all $S_1, S_2 \subseteq \mathfrak{s}$: if $S_1 \in \mathcal{S}$ and $S_1 \subseteq S_2$, then $S_2 \in \mathcal{S}$. A policy is said to be monotone if its corresponding access structure is monotone.

We next recall the definition of a linear secret sharing scheme.

**Definition 3 (Linear Secret Sharing (LSS) Scheme** [16,17]**).** *Let $\ell, n \in \mathbb{N}$ and $p$ be a prime number. We denote $\mathbf{e}_1 = (1, 0, \ldots, 0)^\top \in \mathbb{Z}_p^n$ the first unit-vector in $\mathbb{Z}_p^n$. A* linear secret sharing (LSS) scheme *over $\mathbb{Z}_p$ for an access structure $\mathcal{S} \subseteq 2^{\mathfrak{s}}$ on an attribute universe $\mathcal{X} = \{0,1\}^{\mathfrak{s}}$ is specified by a share generating matrix $\mathbf{M} \in \mathbb{Z}_p^{n \times \ell}$ and a function $\rho\colon [\ell] \to \mathfrak{s}$ mapping the columns of $\mathbf{M}$ to indices in $\mathfrak{s}$, which satisfy the following condition:*

$$S \in \mathcal{S} \iff \mathbf{e}_1 \in \mathsf{span}\{\mathbf{M}[j] : j \in [\ell], \rho(j) \in S\} \ . \tag{3}$$

For convenience, we often do not distinguish between an access structure $\mathcal{S}$, its corresponding policy $f$ and a pair $(\mathbf{M}, \rho)$ satisfying (3). In particular, we may write $f = (\mathbf{M}, \rho)$. In order to share a value $s \in \mathbb{Z}_p$ using an LSS scheme over $\mathbb{Z}_p$, one samples $\mathbf{u} \xleftarrow{\$} \mathbb{Z}_p^{n-1}$ and computes the share vector

$$\mathbf{s} = (s, \mathbf{u}[1], \ldots, \mathbf{u}[n-1]) \cdot \mathbf{M} \in \mathbb{Z}_p^\ell \ .$$

Then a set $\{\mathbf{s}[j]\}_{j \in J}$ for some $J \subseteq [\ell]$ can be used to reconstruct $s$ if and only if $\{\rho(j)\}_{j \in J}$ is authorized with respect to the access structure corresponding to $f = (\mathbf{M}, \rho)$. Indeed, in this case there exist coefficients $\omega_1, \ldots, \omega_\ell \in \mathbb{Z}_p$ such that $\omega_j = 0$ for all $j \in [\ell] \setminus J$ and $\sum_{j \in [\ell]} \omega_j \mathbf{M}[j] = \mathbf{e}_1$. These coefficients can be used to compute

$$\sum_{j \in J} \omega_j \mathbf{s}[j] = \sum_{j \in [\ell]} \omega_j \mathbf{s}[j] = (s, \mathbf{u}[1], \ldots, \mathbf{u}[n-1]) \cdot \sum_{j \in [\ell]} \omega_j \mathbf{M}[j] = s \ .$$

Lewko and Waters [34] presented an LSS scheme for all monotone access structures.

## 3.4 Function-Hiding Slotted Inner-Product Functional Encryption

We recall the definition of slotted IPFE from [38]. Similar to [10,35,38], this primitive will allow us to employ techniques akin to dual system encryption [33,47]. To adhere to the formalism used in this work, we present the syntax in a pairing-based setting.

**Definition 4 (Slotted IPFE).** *Let $\mathbb{G} = \{\mathbb{G}_\lambda = (\mathbb{G}_{\lambda,1}, \mathbb{G}_{\lambda,2}, \mathbb{G}_{\lambda,t}, g_{\lambda,1}, g_{\lambda,2}, g_{\lambda,t}, e_\lambda, p_\lambda)\}_{\lambda \in \mathbb{N}}$ be a sequence of pairing groups. A* slotted IPFE scheme *based on $\mathbb{G}$ consists of five efficient algorithms:*

$\mathsf{Setup}(1^\lambda, \mathfrak{s}_{\mathsf{pub}}, \mathfrak{s}_{\mathsf{pri}}) \to (\mathsf{MPK}, \mathsf{MSK})$*: On input the security parameter and two disjoint index sets, the public slot $\mathfrak{s}_{\mathsf{pub}}$ and the private slot $\mathfrak{s}_{\mathsf{pri}}$, this algorithm outputs a pair of a master public and a master secret key $(\mathsf{MPK}, \mathsf{MSK})$. We denote the whole index set by $\mathfrak{s} := \mathfrak{s}_{\mathsf{pub}} \cup \mathfrak{s}_{\mathsf{pri}}$.*

$\mathsf{Enc}(\mathsf{MSK}, [\![\mathbf{x}]\!]_1) \to \mathsf{CT}$*: On input a master secret key* $\mathsf{MSK}$ *and an encoding of a vector* $\mathbf{x} \in \mathbb{Z}_{p_\lambda}^{\mathfrak{s}}$ *in* $\mathbb{G}_{\lambda,1}$*, this algorithm outputs a ciphertext* $\mathsf{CT}$ *for* $\mathbf{x}$*.*

$\mathsf{KeyGen}(\mathsf{MSK}, [\![\mathbf{y}]\!]_2) \to \mathsf{SK}$*: On input a master secret key* $\mathsf{MSK}$ *and an encoding of a vector* $\mathbf{y} \in \mathbb{Z}_{p_\lambda}^{\mathfrak{s}}$ *in* $\mathbb{G}_{\lambda,2}$*, this algorithm outputs a decryption key* $\mathsf{DK}$ *for* $\mathbf{y}$*.*

$\mathsf{Dec}(\mathsf{DK}, \mathsf{CT}) \to [\![d]\!]_t$*: On input a decryption key* $\mathsf{DK}$ *and a ciphertext* $\mathsf{CT}$*, this algorithm outputs an element* $[\![d]\!]_t \in \mathbb{G}_{\lambda,t}$*.*

$\mathsf{SlotEnc}(\mathsf{MPK}, [\![\mathbf{x}_{\mathsf{pub}}]\!]_1) \to \mathsf{CT}$*: On input a master public key* $\mathsf{MPK}$ *and an encoding of a message vector* $\mathbf{x}_{\mathsf{pub}} \in \mathbb{Z}_{p_\lambda}^{\mathfrak{s}_{\mathsf{pub}}}$ *in* $\mathbb{G}_{\lambda,1}$*, this algorithm outputs a ciphertext for the vector* $\mathbf{x} = \mathbf{x}_{\mathsf{pub}}|^{\mathfrak{s}} \in \mathbb{Z}_{p_\lambda}^{\mathfrak{s}}$*.*

**Correctness.** A slotted IPFE scheme satisfies *decryption correctness* if for all $\lambda \in \mathbb{N}$, all disjoint index sets $\mathfrak{s}_{\mathsf{pub}}, \mathfrak{s}_{\mathsf{pri}}$ and all vectors $\mathbf{x}, \mathbf{y} \in \mathbb{Z}_{p_\lambda}^{\mathfrak{s}}$, it holds that

$$\Pr\left[\mathsf{Dec}(\mathsf{DK},\mathsf{CT}) = [\![\langle \mathbf{x},\mathbf{y}\rangle]\!]_{\mathsf{t}} \;\middle|\; \begin{array}{l} (\mathsf{MPK},\mathsf{MSK}) \leftarrow \mathsf{Setup}(1^\lambda, \mathfrak{s}_{\mathsf{pub}}, \mathfrak{s}_{\mathsf{pri}}) \\ \mathsf{CT} \leftarrow \mathsf{Enc}(\mathsf{MSK}, [\![\mathbf{x}]\!]_1) \\ \mathsf{DK} \leftarrow \mathsf{KeyGen}(\mathsf{MSK}, [\![\mathbf{y}]\!]_2) \end{array}\right] = 1 \ .$$

Furthermore, we say that a slotted IPFE scheme satisfies *slot-mode correctness* if for all $\lambda \in \mathbb{N}$, all disjoint index sets $\mathfrak{s}_{\mathsf{pub}}, \mathfrak{s}_{\mathsf{pri}}$, all $i \in \mathcal{I}_\lambda$ and $\mathbf{x}_{\mathsf{pub}} \in \mathbb{Z}_p^{\mathfrak{s}_{\mathsf{pub}}}$, the following distributions $\mathcal{D}_0, \mathcal{D}_1$ are identical:

$$\mathcal{D}_0 = \left\{ (\mathsf{MPK},\mathsf{MSK},\mathsf{CT}) \;\middle|\; \begin{array}{l} (\mathsf{MPK},\mathsf{MSK}) \leftarrow \mathsf{Setup}(1^\lambda, \mathfrak{s}_{\mathsf{pub}}, \mathfrak{s}_{\mathsf{pri}}) \\ \mathsf{CT} \leftarrow \mathsf{Enc}(\mathsf{MSK}, i, [\![\mathbf{x}_{\mathsf{pub}}|^{\mathfrak{s}}]\!]_1) \end{array}\right\} ,$$

$$\mathcal{D}_1 = \left\{ (\mathsf{MPK},\mathsf{MSK},\mathsf{CT}) \;\middle|\; \begin{array}{l} (\mathsf{MPK},\mathsf{MSK}) \leftarrow \mathsf{Setup}(1^\lambda, \mathfrak{s}_{\mathsf{pub}}, \mathfrak{s}_{\mathsf{pri}}) \\ \mathsf{CT} \leftarrow \mathsf{SlotEnc}(\mathsf{MPK}, i, [\![\mathbf{x}_{\mathsf{pub}}]\!]_1) \end{array}\right\} ,$$

where the probability is taken over the random coins of the algorithms.

**Security.** We define adaptive function-hiding IND-CPA security.

**Definition 5 (Function-Hiding Security).** *For a slotted IPFE scheme* $\mathsf{iFE}$ *and a PPT adversary* $\mathcal{A}$*, we define the security experiment* $\mathbf{Exp}_{\mathsf{iFE},\mathcal{A}}^{\mathsf{sl\text{-}ipfe}\text{-}b}(1^\lambda)$ *as shown in Fig. 1. The oracles* $\mathcal{O}\mathsf{KeyGen}$ *and* $\mathcal{O}\mathsf{Enc}$ *can be called in any order and any (polynomial) number of times. The adversary* $\mathcal{A}$ *is admissible with respect to* $\mathcal{Q}_{\mathsf{enc}}$ *and* $\mathcal{Q}_{\mathsf{key}}$*, denoted by* $\mathsf{adm}(\mathcal{A}) = 1$*, if all* $([\![\mathbf{x}_0]\!]_1, [\![\mathbf{x}_1]\!]_1) \in \mathcal{Q}_{\mathsf{enc}}$ *and* $([\![\mathbf{y}_0]\!]_2, [\![\mathbf{y}_1]\!]_2) \in \mathcal{Q}_{\mathsf{key}}$ *satisfy* $\langle \mathbf{x}_0, \mathbf{y}_0\rangle = \langle \mathbf{x}_1, \mathbf{y}_1\rangle$ *and* $\mathbf{y}_0|_{\mathfrak{s}_{\mathsf{pub}}} = \mathbf{y}_1|_{\mathfrak{s}_{\mathsf{pub}}}$*. Otherwise, we say that* $\mathcal{A}$ *is not admissible and write* $\mathsf{adm}(\mathcal{A}) = 0$*. We call* $\mathsf{iFE}$ function-hiding *if* $\mathbf{Exp}_{\mathsf{iFE},\mathcal{A}}^{\mathsf{sl\text{-}ipfe\text{-}0}}(1^\lambda) \approx_c \mathbf{Exp}_{\mathsf{iFE},\mathcal{A}}^{\mathsf{sl\text{-}ipfe\text{-}1}}(1^\lambda)$*.*

There exists a slotted IPFE scheme based on $\mathbb{G}$ which can be proven (adaptively) function-hiding under the SXDH[6] assumption in $\mathbb{G}$. The construction is based on a sequence of works [7,37,40,48] and has been described explicitly in [38].

[6] More precisely, the security proof only relies on $\mathsf{MDDH}_k$, for any $k > 1$, in both $\mathbb{G}_1$ and $\mathbb{G}_2$. This assumption is implied by SXDH on $\mathbb{G}$.

```
$\underline{\mathsf{Initialize}(1^\lambda, \mathfrak{s}_{\mathsf{pub}}, \mathfrak{s}_{\mathsf{pri}})}$:
$\mathcal{Q}_{\mathsf{enc}}, \mathcal{Q}_{\mathsf{key}} \leftarrow \varnothing$
$(\mathsf{MPK}, \mathsf{MSK}) \leftarrow \mathsf{Setup}(1^\lambda, \mathfrak{s}_{\mathsf{pub}}, \mathfrak{s}_{\mathsf{pri}})$
Return $\mathsf{MPK}$

$\underline{\mathcal{O}\mathsf{Enc}([\![\mathbf{x}_0]\!]_2, [\![\mathbf{x}_1]\!]_2)}$:
$\mathcal{Q}_{\mathsf{enc}} \leftarrow \mathcal{Q}_{\mathsf{enc}} \cup \{([\![\mathbf{x}_0]\!]_1, [\![\mathbf{x}_1]\!]_1)\}$
Return $\mathsf{CT} \leftarrow \mathsf{Enc}(\mathsf{MSK}, [\![\mathbf{x}_b]\!]_1)$

$\underline{\mathcal{O}\mathsf{KeyGen}([\![\mathbf{y}_0]\!]_2, [\![\mathbf{y}_1]\!]_2)}$:
$\mathcal{Q}_{\mathsf{key}} \leftarrow \mathcal{Q}_{\mathsf{key}} \cup \{([\![\mathbf{y}_0]\!]_2, [\![\mathbf{y}_1]\!]_2)\}$
Return $\mathsf{DK} \leftarrow \mathsf{KeyGen}(\mathsf{MSK}, [\![\mathbf{y}_b]\!]_2)$
$\underline{\mathsf{Finalize}(b')}$:
If $\mathsf{adm}(\mathcal{A}) = 1$, return $\beta \leftarrow (b' \stackrel{?}{=} b)$
Else, return a random bit $\beta \stackrel{\$}{\leftarrow} \{0,1\}$
```

**Fig. 1.** Security game $\mathbf{Exp}^{\mathsf{sl\text{-}ipfe\text{-}}b}_{\mathsf{iFE},\mathcal{A}}(1^\lambda)$ for Definition 5

## 3.5 Identity-Based Encryption

We recall the definition of identity-based encryption (IBE).

**Definition 6 (Identity-Based Encryption).** *Let $\mathcal{M} = \{\mathcal{M}_\lambda\}_{\lambda\in\mathbb{N}}$ and $\mathcal{I} = \{\mathcal{I}_\lambda\}_{\lambda\in\mathbb{N}}$ be sequences of message and identity spaces, respectively. An* identity-based encryption scheme *for $\mathcal{M}$ and $\mathcal{I}$ consists of four efficient algorithms:*

$\mathsf{Setup}(1^\lambda) \to (\mathsf{MPK}, \mathsf{MSK})$**:** *On input the security parameter, this algorithm outputs a pair of a master public key* $\mathsf{MPK}$ *and a master secret key* $\mathsf{MSK}$*.*

$\mathsf{Enc}(\mathsf{MPK}, i, \mu) \to \mathsf{CT}$**:** *On input a master public key* $\mathsf{MSK}$*, an identity $i \in \mathcal{I}_\lambda$ and a message $\mu \in \mathcal{M}_\lambda$, this algorithm outputs a ciphertext* $\mathsf{CT}$ *for $\mu$ created with respect to $i$.*

$\mathsf{KeyGen}(\mathsf{MSK}, i') \to \mathsf{DK}$**:** *On input a master secret key* $\mathsf{MSK}$ *and an identity $i' \in \mathcal{I}_\lambda$, this algorithm outputs a decryption key* $\mathsf{DK}$ *for $i'$.*

$\mathsf{Dec}(\mathsf{DK}, \mathsf{CT}) \to \mu' \vee \bot$**:** *On input a decryption key* $\mathsf{DK}$ *and a ciphertext* $\mathsf{CT}$*, this algorithm outputs an element $\mu' \in \mathcal{M}_\lambda$ or $\bot$.*

**Correctness.** An IBE scheme is said to be *correct* if for all $\lambda \in \mathbb{N}$, all identities $i \in \mathcal{I}_\lambda$ and all messages $\mu \in \mathcal{M}_\lambda$, it holds that

$$\Pr\left[\mathsf{Dec}(\mathsf{DK}, \mathsf{CT}) = \mu \;\middle|\; \begin{array}{l} (\mathsf{MPK}, \mathsf{MSK}) \leftarrow \mathsf{Setup}(1^\lambda) \\ \mathsf{CT} \leftarrow \mathsf{Enc}(\mathsf{MPK}, i, \mu) \\ \mathsf{DK} \leftarrow \mathsf{KeyGen}(\mathsf{MSK}, i) \end{array}\right] = 1 \ ,$$

where the probability is taken over the random coins of the algorithms.

**Security.** We define adaptive IND-CPA security.

**Definition 7 (Security).** *For an IBE scheme* $\mathsf{IBE}$ *and a PPT adversary $\mathcal{A}$, we define the security experiment $\mathbf{Exp}^{\mathsf{ibe}\text{-}b}_{\mathsf{IBE},\mathcal{A}}(1^\lambda)$ as shown in Fig. 2. The oracle $\mathcal{O}\mathsf{KeyGen}$ can be called any (polynomial) number of times whereas the oracle $\mathcal{O}\mathsf{Enc}$ can be called only once. We call* $\mathsf{IBE}$ secure *if $\mathbf{Exp}^{\mathsf{ibe}\text{-}0}_{\mathsf{IBE},\mathcal{A}}(1^\lambda) \approx_c \mathbf{Exp}^{\mathsf{ibe}\text{-}1}_{\mathsf{IBE},\mathcal{A}}(1^\lambda)$.*

There exist various IBE schemes in the group-based setting, *e.g.* [21,32,47].

```
$\underline{\mathsf{Initialize}(1^\lambda)}$:
$\mathcal{Q} \leftarrow \varnothing$; $i_{\mathsf{enc}} \leftarrow \bot$
$(\mathsf{MPK}, \mathsf{MSK}) \leftarrow \mathsf{Setup}(1^\lambda)$
Return $\mathsf{MPK}$

$\underline{\mathcal{O}\mathsf{KeyGen}(i')}$:
$\mathcal{Q} \leftarrow \mathcal{Q} \cup \{i'\}$
Return $\mathsf{DK} \leftarrow \mathsf{KeyGen}(\mathsf{MSK}, i')$

$\underline{\mathcal{O}\mathsf{Enc}(i, \mu^0, \mu^1)}$:
$i_{\mathsf{enc}} \leftarrow i$
Return $\mathsf{CT} \leftarrow \mathsf{Enc}(\mathsf{MPK}, i, \mu^b)$

$\underline{\mathsf{Finalize}(b')}$:
If $i_{\mathsf{enc}} \notin \mathcal{Q}$, return $\beta \leftarrow (b' \overset{?}{=} b)$
Else, return a random bit $\beta \overset{\$}{\leftarrow} \{0,1\}$
```

**Fig. 2.** Security game $\mathbf{Exp}^{\mathsf{ibe}\text{-}b}_{\mathsf{IBE},\mathcal{A}}(1^\lambda)$ for Definition 7

## 3.6 Lockable Obfuscation

We recall the definition of a lockable obfuscator [30,49]. Given polynomials $n = n(\lambda)$,$m = m(\lambda)$ and $d = d(\lambda)$, we denote by $\mathcal{C}_{n,m,d}(\lambda)$ the class of depth $d(\lambda)$ circuits with $n(\lambda)$ bits input and $m(\lambda)$ bits output.

**Definition 8 (Lockable Obfuscation).** *Let $\mathcal{M} = \{\mathcal{M}_\lambda\}_{\lambda\in\mathbb{N}}$ be a sequence of message spaces and $\{\mathcal{C}_{n,m,d}(\lambda)\}_{\lambda\in\mathbb{N}}$ a sequence of circuit classes. A lockable obfuscator for $\mathcal{M}$ and $\mathcal{C}$ is a tuple of two efficient algorithms:*

$\mathsf{Obf}(1^\lambda, C, \mu, \sigma) \to (\widetilde{C})$**:** *On input $1^\lambda$, a circuit $C \in \mathcal{C}_{n,m,d}(\lambda)$, a message $\mu \in \mathcal{M}_\lambda$ and a "lock value" $\sigma \in \{0,1\}^{m(\lambda)}$, this algorithm outputs an obfuscated circuit $\widetilde{C}$.*

$\mathsf{Eval}(\widetilde{C}, x) \to \mu' \vee \bot$**:** *On input an obfuscated circuit $\widetilde{C}$ and an input $x \in \{0,1\}^{n(\lambda)}$, this algorithm outputs a value $\mu' \in \mathcal{M}_\lambda$ or $\bot$.*

**Correctness.** A lockable obfuscator satisfies *(perfect) correctness* if for all $\lambda \in \mathbb{N}$, all circuits $C \in \mathcal{C}_{n,m,d}(\lambda)$, all messages $\mu \in \mathcal{M}_\lambda$ and all inputs $x \in \{0,1\}^{n(\lambda)}$, the following two implications are satisfied:

1. if $C(x) = \sigma$, then $\mathsf{Eval}(\mathsf{Obf}(1^\lambda, C, \mu, \sigma), x) = \mu$
2. if $C(x) \neq \sigma$, then $\mathsf{Eval}(\mathsf{Obf}(1^\lambda, C, \mu, \sigma), x) = \bot$

**Security.** We define security against multiple challenges. In [11], this definition was observed to be equivalent to the original single-challenge version from [30].

**Definition 9 (Security against Multiple Queries).** *For a lockable obfuscation scheme $\mathsf{LObf} = (\mathsf{Obf}, \mathsf{Eval})$ and an efficient algorithm $\mathsf{Sim}$, we define the following oracles:*

$\mathcal{O}\mathsf{Obf}^0(C, \mu)$*: sample $\sigma \overset{\$}{\leftarrow} \{0,1\}^{m(\lambda)}$ and return $\widetilde{C} \leftarrow \mathsf{Obf}(1^\lambda, C, \mu, \sigma)$*
$\mathcal{O}\mathsf{Obf}^1(C, \mu)$*: return $\mathsf{Sim}(1^\lambda, 1^{|C|}, 1^{|\mu|})$*

*We call* $\mathsf{LObf}$ secure *if there exists a PPT simulator* $\mathsf{Sim}$ *such that for all PPT adversaries $\mathcal{A}$, there exists a negligible function* $\mathrm{negl}(\cdot)$ *such that*

$$\mathbf{Adv}^{\mathsf{lock}}_{\mathsf{LObf},\mathcal{A}}(\lambda) := \left|\Pr\left[\mathcal{A}^{\mathcal{O}\mathsf{Obf}^1} \to 1\right] - \Pr\left[\mathcal{A}^{\mathcal{O}\mathsf{Obf}^0} \to 1\right]\right| \leq \mathrm{negl}(\lambda) \ .$$

Perfectly correct lockable obfuscators for general circuits are known to exist under the LWE assumption [29,30].

## 4 Multi-client Attribute-Based and Predicate Encryption

We define multi-client attribute-based encryption (MC-ABE) and multi-client predicate encryption (MC-PE). Since the only difference between these notions lies in the security game, we unify the syntax of the algorithms.

**Definition 10 (Public-Key Syntax).** *Let $n = n(\lambda)$ be a polynomial. Furthermore, let $\mathcal{M} = \{\mathcal{M}_\lambda\}_{\lambda\in\mathbb{N}}$ be a sequence of message spaces, $\mathcal{X} = \{\mathcal{X}_\lambda\}_{\lambda\in\mathbb{N}}$ a sequence of attribute universes, $\mathcal{L} = \{\mathcal{L}_\lambda\}_{\lambda\in\mathbb{N}}$ a sequence of label spaces and $\mathcal{F} = \{\mathcal{F}_\lambda\}_{\lambda\in\mathbb{N}}$ a sequence of policy classes, where each policy $f_\lambda \in \mathcal{F}_\lambda$ maps from $\mathcal{X}_\lambda^n$ to $\{0,1\}$. An* MC-ABE *(resp.* MC-PE*) scheme for $\mathcal{M}$, $\mathcal{X}$, $\mathcal{F}$ and $\mathcal{L}$ consists of five efficient algorithms:*

$\mathsf{Setup}(1^\lambda) \to (\mathsf{MPK}, \mathsf{MSK}, \{\mathsf{SK}_i\}_{i\in[n-1]})$**:** *On input the security parameter $1^\lambda$, this algorithm outputs a pair of master public key $\mathsf{MPK}$ and master secret key $\mathsf{MSK}$ as well as a set of secret keys $\{\mathsf{SK}_i\}_{i\in[n-1]}$.*

$\mathsf{Enc}(\mathsf{MPK}, \mathsf{lab}, x_0, \mu) \to \mathsf{CT}_\mathsf{lab}$**:** *On input the master public key $\mathsf{MPK}$, a label $\mathsf{lab} \in \mathcal{L}_\lambda$, an attribute $x_0 \in \mathcal{X}_\lambda$ and a message $\mu \in \mathcal{M}_\lambda$, this algorithm outputs a ciphertext $\mathsf{CT}_\mathsf{lab}$.*
*In case of an MC-ABE scheme, we assume that $\mathsf{CT}_\mathsf{lab}$ implicitly includes $x_0$.*

$\mathsf{AKeyGen}(\mathsf{SK}_i, \mathsf{lab}, x_i) \to \mathsf{DK}_{\mathsf{lab},i}$**:** *On input a secret key $\mathsf{SK}_i$ for some $i \in [n-1]$, a label $\mathsf{lab} \in \mathcal{L}_\lambda$ and an attribute $x_i \in \mathcal{X}_\lambda$, this algorithm outputs a decryption key $\mathsf{DK}_{\mathsf{lab},i}$.*
*In case of an MC-ABE scheme, we assume that $\mathsf{DK}_{\mathsf{lab},i}$ implicitly includes $x_i$.*

$\mathsf{PKeyGen}(\mathsf{MSK}, f) \to \mathsf{DK}_f$**:** *On input the master secret key $\mathsf{MSK}$ and a policy $f \in \mathcal{F}_\lambda$, this algorithm outputs a decryption key $\mathsf{DK}_f$.*
*We assume that $\mathsf{DK}_f$ implicitly includes a description of $f$.*

$\mathsf{Dec}(\mathsf{DK}_f, \{\mathsf{DK}_{\mathsf{lab},i}\}_{i\in[n-1]}, \mathsf{CT}_\mathsf{lab}) \to \mu' \vee \bot$*: On input a decryption key $\mathsf{DK}_f$ for a policy $f \in \mathcal{F}_\lambda$, a set of attribute decryption keys $\{\mathsf{DK}_{\mathsf{lab},i}\}_{i\in[n-1]}$ generated with respect to some label $\mathsf{lab} \in \mathcal{L}_\lambda$ and a ciphertext $\mathsf{CT}_\mathsf{lab}$ created with respect to the same label $\mathsf{lab}$, this algorithm outputs an element $\mu' \in \mathcal{M}_\lambda$ or $\bot$.*

Below, we discuss security in the public-key and secret-key setting. In the secret-key setting, we slightly change the syntax, as we find it more intuitive to let the encryption algorithm take a secret key $\mathsf{SK}_0$ instead of a master public key $\mathsf{MPK}$ if this key is not given to the adversary.

**Correctness.** An MC-ABE (resp. MC-PE) is *correct* if for every $\lambda, n \in \mathbb{N}$, label $\mathsf{lab} \in \mathcal{L}_\lambda$, message $\mu \in \mathcal{M}_\lambda$, policy $f \in \mathcal{F}_\lambda$ and attributes $x_0, \ldots, x_{n-1} \in \mathcal{X}_\lambda$ such that $f(x_0, \ldots, x_{n-1}) = 1$, it holds that

$$\Pr\left[\mu' = \mu \;\middle|\; \begin{array}{l} (\mathsf{MPK}, \mathsf{MSK}, \{\mathsf{SK}_i\}_{i\in[n-1]}) \leftarrow \mathsf{Setup}(1^\lambda) \\ \mathsf{CT}_\mathsf{lab} \leftarrow \mathsf{Enc}(\mathsf{MPK}, \mathsf{lab}, x_0, \mu) \\ \forall i \in [n-1]\colon \mathsf{DK}_{\mathsf{lab},i} \leftarrow \mathsf{AKeyGen}_i(\mathsf{SK}_i, \mathsf{lab}, x_i) \\ \mathsf{DK}_f \leftarrow \mathsf{PKeyGen}(\mathsf{MSK}, f) \\ \mu' := \mathsf{Dec}(\mathsf{DK}_f, \{\mathsf{DK}_{\mathsf{lab},i}\}_{i\in[n-1]}, \mathsf{CT}_\mathsf{lab}) \end{array}\right] = 1$$

**Security.** We define security for MC-ABE in the public-key setting as well as security for MC-PE in the secret-key setting.

**Definition 11 (Public-Key Security for MC-ABE).** *Let* $\mathsf{xxx} \in \{\mathsf{sel}, \mathsf{adap}\}$ *and* $\mathsf{yyy} \in \{\mathsf{norep}, \mathsf{rep}\}$*. For an MC-ABE scheme* $\mathsf{aFE}$ *and a PPT adversary* $\mathcal{A}$*, we define the experiment* $\mathbf{Exp}^{\mathsf{mc\text{-}abe\text{-}}b}_{\mathsf{aFE},\mathcal{A}}(1^\lambda)$ *as shown in Fig. 3. The oracles* $\mathcal{O}\mathsf{Corrupt}$*,* $\mathcal{O}\mathsf{Enc}$*,* $\mathcal{O}\mathsf{AKeyGen}$ *and* $\mathcal{O}\mathsf{PKeyGen}$ *can be called in any order and any polynomial number of times, except for* $\mathcal{O}\mathsf{Enc}$ *which can be called only once. Let* $(\mathsf{lab}, x_0, \mu^0, \mu^1)$ *denote the single query to* $\mathcal{O}\mathsf{Enc}$*. The adversary* $\mathcal{A}$ *is admissible, denoted by* $\mathsf{adm}(\mathcal{A}) = 1$*, if it satisfies the following conditions:*

1. *For all* $f \in \mathcal{Q}_{\mathsf{key}}$ *and* $x_1, \ldots, x_{n-1} \in \mathcal{X}_\lambda$ *such that* $(i, \mathsf{lab}, x_i) \in \mathcal{Q}_{\mathsf{akey}}$ *for all* $i \in [n-1] \setminus \mathcal{C}$*, it holds* $f(x_0, \ldots, x_{n-1}) = 0$*.*
2. *If* $\mathsf{xxx} = \mathsf{sel}$*, then the adversary cannot call* $\mathcal{O}\mathsf{Corrupt}$*,* $\mathcal{O}\mathsf{Enc}$ *and* $\mathcal{O}\mathsf{AKeyGen}$ *anymore after submitting the first query to* $\mathcal{O}\mathsf{PKeyGen}$*.*
3. *If* $\mathsf{yyy} = \mathsf{norep}$*, then for each* $i \in [n-1]$ *and* $\mathsf{lab} \in \mathcal{L}$ *the adversary submits at most one query of the form* $\mathcal{O}\mathsf{AKeyGen}(i, \mathsf{lab}, \star)$*, i.e. we have* $|\{x_i \in \{0,1\}^k : (i, \mathsf{lab}, x_i) \in \mathcal{Q}_{\mathsf{akey}}\}| \leq 1$*.*

*Otherwise, we say that* $\mathcal{A}$ *is not admissible and write* $\mathsf{adm}(\mathcal{A}) = 0$*. We call* $\mathsf{aFE}$ xxx-yyy-secure *if* $\mathbf{Exp}^{\mathsf{mc\text{-}abe\text{-}}0}_{\mathsf{aFE},\mathcal{A}}(1^\lambda) \approx_c \mathbf{Exp}^{\mathsf{mc\text{-}abe\text{-}}1}_{\mathsf{aFE},\mathcal{A}}(1^\lambda)$*.*

```
$\underline{\mathsf{Initialize}(1^\lambda)}$:
    $\mathcal{C}, \mathcal{Q}_{\mathsf{enc}}, \mathcal{Q}_{\mathsf{akey}}, \mathcal{Q}_{\mathsf{pkey}} \leftarrow \varnothing$
    $(\mathsf{MPK}, \mathsf{MSK}, \{\mathsf{SK}_i\}_{i \in [n-1]}) \leftarrow \mathsf{Setup}(1^\lambda)$
    Return $\mathsf{MPK}$

$\underline{\mathcal{O}\mathsf{Corrupt}(i)\text{ for } i \in [n-1]}$:
    $\mathcal{C} \leftarrow \mathcal{C} \cup \{i\}$; return $\mathsf{SK}_i$

$\underline{\mathcal{O}\mathsf{Enc}(\mathsf{lab}, x_0, \mu^0, \mu^1)}$:
    $\mathcal{Q}_{\mathsf{enc}} \leftarrow \mathcal{Q}_{\mathsf{enc}} \cup \{(\mathsf{lab}, x_0, \mu^0, \mu^1)\}$
    Return $\mathsf{CT} \leftarrow \mathsf{Enc}(\mathsf{MPK}, \mathsf{lab}, x_0, \mu^b)$

$\underline{\mathcal{O}\mathsf{AKeyGen}(i, \mathsf{lab}, x_i)}$:
    $\mathcal{Q}_{\mathsf{akey}} \leftarrow \mathcal{Q}_{\mathsf{akey}} \cup \{(i, \mathsf{lab}, x_i)\}$
    Return $\mathsf{DK}_i \leftarrow \mathsf{AKeyGen}(\mathsf{SK}_i, \mathsf{lab}, x_i)$

$\underline{\mathcal{O}\mathsf{PKeyGen}(f)}$:
    $\mathcal{Q}_{\mathsf{pkey}} \leftarrow \mathcal{Q}_{\mathsf{pkey}} \cup \{f\}$
    Return $\mathsf{DK}_f \leftarrow \mathsf{PKeyGen}(\mathsf{MSK}, f)$

$\underline{\mathsf{Finalize}(b')}$:
    If $\mathsf{adm}(\mathcal{A}) = 1$, return $\beta \leftarrow (b' \overset{?}{=} b)$
    Else, return a random bit $\beta \overset{\$}{\leftarrow} \{0,1\}$
```

**Fig. 3.** Security game $\mathbf{Exp}^{\mathsf{mc\text{-}abe\text{-}}b}_{\mathsf{aFE},\mathcal{A}}(1^\lambda)$ for Definition 11

**Definition 12 (Secret-Key Security for MC-PE).** *Let* $\mathsf{xxx} \in \{\mathsf{sel}, \mathsf{adap}\}$ *and* $\mathsf{yyy} \in \{\mathsf{norep}, \mathsf{rep}\}$*. For an MC-PE scheme* $\mathsf{pFE}$ *and a PPT adversary* $\mathcal{A}$*, we define the experiment* $\mathbf{Exp}^{\mathsf{mc\text{-}pe\text{-}}b}_{\mathsf{pFE},\mathcal{A}}(1^\lambda)$ *as shown in Fig. 4. The oracles* $\mathcal{O}\mathsf{Corrupt}$*,* $\mathcal{O}\mathsf{Enc}$*,* $\mathcal{O}\mathsf{AKeyGen}$ *and* $\mathcal{O}\mathsf{PKeyGen}$ *can be called in any order and any polynomial number of times. Let* $\mathsf{lab} \in \mathcal{L}$*. We define* $\mathcal{Q}'_{0,\mathsf{lab}} = \{(x_0^0, x_0^1, \mu^0, \mu^1) : (\mathsf{lab}, x_0^0, x_0^1, \mu^0, \mu^1) \in \mathcal{Q}_{\mathsf{enc}}\}$ *and* $\mathcal{Q}'_{i,\mathsf{lab}} = \{(x_i^0, x_i^1) : (i, \mathsf{lab}, x_i^0, x_i^1) \in \mathcal{Q}_{\mathsf{akey}}\}$ *as well as*

$$\mathcal{Q}_{0,\mathsf{lab}} = \begin{cases} \mathcal{Q}'_{0,\mathsf{lab}} & \textit{if } 0 \in [0;n-1] \setminus \mathcal{C} \\ \mathcal{Q}'_{0,\mathsf{lab}} \cup \{(x_0, x_0, \mu, \mu) : x_0 \in \mathcal{X}, \mu \in \mathcal{M}\} & \textit{if } 0 \in \mathcal{C} \end{cases}$$

$$\mathcal{Q}_{i,\mathsf{lab}} = \begin{cases} \mathcal{Q}'_{i,\mathsf{lab}} & \textit{if } i \in [0;n-1] \setminus \mathcal{C} \\ \mathcal{Q}'_{i,\mathsf{lab}} \cup \{(x_i, x_i) : x_i \in \mathcal{X}\} & \textit{if } i \in \mathcal{C} \end{cases}$$

*for all* $i \in [n-1]$. *The adversary* $\mathcal{A}$ *is* admissible, *denoted by* $\mathsf{adm}(\mathcal{A}) = 1$, *if it satisfies the following conditions:*

1. *For all* $\mathsf{lab} \in \mathcal{L}$, $(x_0^0, x_0^1, \mu^0, \mu^1) \in \mathcal{Q}_{0,\mathsf{lab}}$, $(x_1^0, x_1^1) \in \mathcal{Q}_{1,\mathsf{lab}}, (x_2^0, x_2^1) \in \mathcal{Q}_{2,\mathsf{lab}}, \ldots, (x_{n-1}^0, x_{n-1}^1) \in \mathcal{Q}_{n-1,\mathsf{lab}}$ *and policies* $f \in \mathcal{Q}_{\mathsf{key}}$, *it holds* $f(x_0^0, \ldots, x_{n-1}^0) = f(x_0^1, \ldots, x_{n-1}^1) = 0$ *or* $(x_0^0, \ldots, x_{n-1}^0, \mu^0) = (x_0^1, \ldots, x_{n-1}^1, \mu^1)$.
2. *If* $\mathsf{xxx} = \mathsf{sel}$, *then the adversary cannot call* $\mathcal{O}\mathsf{Corrupt}$, $\mathcal{O}\mathsf{Enc}$ *and* $\mathcal{O}\mathsf{AKeyGen}$ *anymore after submitting the first query to* $\mathcal{O}\mathsf{PKeyGen}$.
3. *If* $\mathsf{yyy} = \mathsf{norep}$, *then for each* $i \in [n-1]$ *and* $\mathsf{lab} \in \mathcal{L}$ *the adversary submits at most one query of the form* $\mathcal{O}\mathsf{AKeyGen}(i, \mathsf{lab}, \star, \star)$, i.e. *we have* $|\{x_i \in \{0,1\}^k : (i, \mathsf{lab}, x_i) \in \mathcal{Q}_{\mathsf{akey}}\}| \leq 1$.

*Otherwise, we say that* $\mathcal{A}$ *is not admissible and write* $\mathsf{adm}(\mathcal{A}) = 0$. *We call* pFE xxx-yyy-secure *if* $\mathbf{Exp}^{\mathsf{mc\text{-}pe\text{-}0}}_{\mathsf{pFE},\mathcal{A}}(1^\lambda) \approx_c \mathbf{Exp}^{\mathsf{mc\text{-}pe\text{-}1}}_{\mathsf{pFE},\mathcal{A}}(1^\lambda)$.

```
$\underline{\mathsf{Initialize}(1^\lambda)}$:
    $\mathcal{C}, \mathcal{Q}_{\mathsf{enc}}, \mathcal{Q}_{\mathsf{akey}}, \mathcal{Q}_{\mathsf{pkey}} \leftarrow \varnothing$
    $(\{\mathsf{SK}_i\}_{i \in [0;n-1]}, \mathsf{MSK}) \leftarrow \mathsf{Setup}(1^\lambda)$

$\underline{\mathcal{O}\mathsf{Corrupt}(i) \text{ for } i \in [0;n-1]]}$:
    $\mathcal{C} \leftarrow \mathcal{C} \cup \{i\}$; return $\mathsf{SK}_i$

$\underline{\mathcal{O}\mathsf{Enc}(\mathsf{lab}, x_0^0, x_0^1, \mu^0, \mu^1)}$:
    $\mathcal{Q}_{\mathsf{enc}} \leftarrow \mathcal{Q}_{\mathsf{enc}} \cup \{(\mathsf{lab}, x_0^0, x_0^1, \mu^0, \mu^1)\}$
    Return $\mathsf{CT} \leftarrow \mathsf{Enc}(\mathsf{SK}_1, \mathsf{lab}, x_0^b, \mu^b)$

$\underline{\mathcal{O}\mathsf{AKeyGen}(i, \mathsf{lab}, x_i^0, x_i^1)}$:
    $\mathcal{Q}_{\mathsf{akey}} \leftarrow \mathcal{Q}_{\mathsf{akey}} \cup \{(i, \mathsf{lab}, x_i^0, x_i^1)\}$
    Return $\mathsf{DK}_i \leftarrow \mathsf{AKeyGen}(\mathsf{SK}_i, \mathsf{lab}, x_i^b)$

$\underline{\mathcal{O}\mathsf{PKeyGen}(f)}$:
    $\mathcal{Q}_{\mathsf{pkey}} \leftarrow \mathcal{Q}_{\mathsf{pkey}} \cup \{f\}$
    Return $\mathsf{DK}_f \leftarrow \mathsf{PKeyGen}(\mathsf{MSK}, f)$

$\underline{\mathsf{Finalize}(b')}$:
    If $\mathsf{adm}(\mathcal{A}) = 1$, return $\beta \leftarrow (b' \overset{?}{=} b)$
    Else, return a random bit $\beta \overset{\$}{\leftarrow} \{0,1\}$
```

**Fig. 4.** Security game $\mathbf{Exp}^{\mathsf{mc\text{-}pe\text{-}}b}_{\mathsf{pFE},\mathcal{A}}(1^\lambda)$ for Definition 12

## 5 Construction of MC-ABE

In this section, we present our construction for MC-ABE with repetitions. The full version contains an additional construction of MC-ABE for all LSS schemes without repetitions which can be upgraded to $\mathsf{NC}^1$ policies when not considering corruptions. Our construction needs the following two definitions.

**Complete Policies.** Let $k, n \in \mathbb{N}$. We define $c\colon \{0,1\}^{[n-1]\times\{0\}} \to \{0,1\}$ by $c((x_{i,0})_{i\in[n-1]}) = \bigwedge_{i\in[n-1]} x_{i,0}$. Given a policy $f\colon \{0,1\}^{[0;n-1]\times[k]} \to \{0,1\}$, we write $(c \wedge f)$ for the *complete* policy

$$\begin{aligned}(c\wedge f)\colon \{0,1\}^{[0;n-1]\times[0;k]} &\to \{0,1\}\\ (x_{i,j})_{i,j} &\mapsto c\big((x_{i,0})_{i\in[n-1]}\big) \wedge f\big((x_{i,j})_{(i,j)\in[0;n-1]\times[k]}\big) \ .\end{aligned}$$

**Transformation to Monotone Policies.** For $(x_{i,j})_{i,j} \in \{0,1\}^{[0;n-1]\times[0;k]}$, we define $(\overline{x}^{\beta}_{i,j})^{\beta}_{i,j} \in \{0,1\}^{[0;n-1]\times[0;k]\times\{0,1\}}$ by $\overline{x}^1_{i,j} = x_{i,j}$ and $\overline{x}^0_{i,j} = 1 - x_{i,j}$. Given a policy $f\colon \{0,1\}^{[0;n-1]\times[0;k]} \to \{0,1\}$ computable by an $\mathsf{NC}^1$ circuit, we construct the corresponding monotone policy $\overline{f}\colon \{0,1\}^{[0;n-1]\times[0;k]\times\{0,1\}} \to \{0,1\}$ as follows: first, we view $f$ as a Boolean formula consisting of (fan-in 1) $\neg$ gates and (fan-in 2) $\wedge$ and $\vee$ gates. Then, using De Morgan laws, we push the $\neg$ gates to the leaves such that all internal nodes consist only of $\wedge$ and $\vee$ gates, while leaves are labeled by either attributes or their negations. Finally, for each $(i,j) \in [0;n-1] \times [0;k]$ we identify the attribute $x_{i,j} \in \{0,1\}$ with the attribute $\overline{x}^1_{i,j}$ and the negation of $x_{i,j}$ with $\overline{x}^0_{i,j}$. The resulting formula $\overline{f}$ is monotone and equivalent to $f$ in the sense that it satisfies $\overline{f}((\overline{x}^{\beta}_{i,j})^{\beta}_{i,j}) = f((x_{i,j})_{i,j})$ for all inputs $(x_{i,j})_{i,j} \in \{0,1\}^{[0;n-1]\times[0;k]}$.

We combine this transformation with complete policies. Given an $\mathsf{NC}^1$ policy $f\colon \{0,1\}^{[0;n-1]\times[k]} \to \{0,1\}$, we denote by $\overline{(c\wedge f)}\colon \{0,1\}^{[0;n-1]\times[0;k]\times\{0,1\}} \to \{0,1\}$ the monotone policy obtained by applying the above transformation to the complete policy $(c \wedge f)\colon \{0,1\}^{[0;n-1]\times[0;k]} \to \{0,1\}$.

**Our Policy Classes.** We consider various policy classes $\mathcal{F}$ containing policies of the form $f\colon \{0,1\}^{[0;n-1]\times[k]} \to \{0,1\}$.

- *Small Parameters.* Let $k, n$ such that $kn = O(\log\lambda)$, *i.e.* the total length of the input is logarithmic in $\lambda$. We let $\mathcal{F}^{\textsf{log-att}}$ denote the class of all policies with input $\{0,1\}^{[0;n-1]\times[k]}$ computable by an $\mathsf{NC}^1$ circuit and, for $\mathbf{x}_1, \ldots, \mathbf{x}_n \in \{0,1\}^k$, we define the sets

$$\Pi^{\textsf{log-att}'}_{\mathbf{x}_0} = \left\{(\mathbf{y}'_0, \ldots, \mathbf{y}'_{n-1}) : \mathbf{y}'_0 = \mathbf{x}_0 \wedge \mathbf{y}'_1, \ldots, \mathbf{y}'_{n-1} \in \{0,1\}^k\right\} \ ,$$

  and $\Omega^{\textsf{log-att}'}_{\mathbf{x}_i} = \{\mathbf{x}_i\}$ for $i \in [n-1]$.
- $\mathsf{NC}^0$ *Policies.* Let $k, n = \mathrm{poly}(\lambda)$ and $d = O(1)$ be some fixed upper bound on the depth of the considered circuits. Then each policy depends on at most $\tau = 2^d = O(1)$ out of the $kn$ input bits. We denote by $\mathcal{F}^{\textsf{const-dep}}$ the set of all $\mathsf{NC}^0$ policies with depth $d$ and input $\{0,1\}^{[0;n-1]\times[k]}$. For $\mathbf{x}_1, \ldots, \mathbf{x}_n \in \{0,1\}^k$, we define the sets

$$\Pi^{\textsf{const-dep}'}_{\mathbf{x}_0} = \left\{(\mathbf{y}'_0, \ldots, \mathbf{y}'_{n-1}) : \begin{array}{c}\mathbf{y}'_0 = \mathbf{x}_0 \wedge \mathbf{y}'_1, \ldots, \mathbf{y}'_{n-1} \in \{0,1,\perp\}^k\\ \text{s.t. } \sum_{i\in[0;n-1]} \delta(\mathbf{y}'_i) = \tau\end{array}\right\}$$

$$\Omega^{\textsf{const-dep}'}_{\mathbf{x}_i} = \left\{\mathbf{z}'_i \in \{0,1,\perp\}^k : (\forall j \in [k].\ \mathbf{z}'_i[j] \in \{\mathbf{x}_i[j], \perp\}) \wedge (\delta(\mathbf{z}'_i) \leq \tau)\right\} \ ,$$

  where $\delta(\mathbf{y}) = |\{j \in [k] : \mathbf{y}[j] \in \{0,1\}\}|$ denotes the number of coordinates being not equal to $\perp$.

- *Threshold Policies.* Let $k, n = \text{poly}(\lambda)$. Instead of a constant input locality, we may also consider policies, where every authorized set has a constant-size subset that is also authorized. This property is satisfied by *e.g.* threshold policies with a constant threshold $\tau = O(1)$. We denote by $f^{t\text{-thr}}$ the threshold policy which allows the reconstruction of the secret from arbitrary $t$ (out of the total of $kn$) shares. Then we define the policy class $\mathcal{F}^{\leq\text{const-thr}} = \{f^{t\text{-thr}} : t \in [\tau]\}$ and, for $\mathbf{x}_1, \ldots, \mathbf{x}_n \in \{0,1\}^k$, we set

$$\Pi_{\mathbf{x}_0}^{\leq\text{const-thr}'} = \left\{ (\mathbf{y}'_0, \ldots, \mathbf{y}'_{n-1}) : \begin{array}{l} \mathbf{y}'_0 = \mathbf{x}_0 \wedge \mathbf{y}'_1, \ldots, \mathbf{y}'_{n-1} \in \{1, \bot\}^k \\ \text{s.t. } \sum_{i \in [0;n-1]} \delta(\mathbf{y}'_i) = \tau \end{array} \right\}$$
$$\Omega_{\mathbf{x}_i}^{\leq\text{const-thr}'} = \left\{ \mathbf{z}'_i \in \{1, \bot\}^k : (\forall j \in [k].\ \mathbf{z}'_i[j] \in S_{\mathbf{x}_i[j]}) \wedge (\delta(\mathbf{z}'_i) \leq \tau) \right\} ,$$

where $S_0 = \{\bot\}$ and $S_1 = \{1, \bot\}$. Note that threshold policies are in particular monotone policies, which is why we can pick $\mathbf{y}'_i, \mathbf{z}'_i \in \{1, \bot\}^k$ as opposed to $\mathbf{y}'_i, \mathbf{z}'_i \in \{0, 1, \bot\}^k$. This will improve the efficiency of the scheme as it reduces the size of the sets $\Pi_{\mathbf{x}_0}^{\leq\text{const-thr}'}$ and $\Omega_{\mathbf{x}_i}^{\leq\text{const-thr}'}$.
Conversely, we define the policy class $\mathcal{F}^{\geq\text{const-thr}} = \{f^{t\text{-thr}} : t \in [kn - \tau; kn]\}$ and the sets

$$\Pi_{\mathbf{x}_0}^{\geq\text{const-thr}'} = \left\{ (\mathbf{y}'_0, \ldots, \mathbf{y}'_{n-1}) : \begin{array}{l} \mathbf{y}'_0 = \mathbf{x}_0 \wedge \mathbf{y}'_1, \ldots, \mathbf{y}'_{n-1} \in \{1, \bot\}^k \\ \text{s.t. } \sum_{i \in [0;n-1]} \delta(\mathbf{y}'_i) \geq kn - \tau \end{array} \right\}$$

and $\Omega_{\mathbf{x}_i}^{\geq\text{const-thr}'} = \{\mathbf{z}'_i\}$ where $\mathbf{z}'_i \in \{1, \bot\}^k$ is defined coordinate-wise as $\mathbf{z}'_i[j] = s_{\mathbf{x}_i[j]}$ for all $j \in [k]$ where $s_0 = \bot$ and $s_1 = 1$.

We must protect against incomplete queries by considering the complete policy $(c \wedge f)$ instead of $f$. For this, we define

$$\Pi_{\mathbf{x}_0}^{\text{type}} = \left\{ (\mathbf{y}_0, \ldots, \mathbf{y}_{n-1}) : \begin{array}{l} \mathbf{y}_0 = (\bot, \mathbf{y}'_0) \wedge \mathbf{y}_1 = (1, \mathbf{y}'_1) \wedge \cdots \wedge \mathbf{y}_{n-1} = (1, \mathbf{y}'_{n-1}) \\ \text{s.t. } (\mathbf{y}'_0, \ldots, \mathbf{y}'_{n-1}) \in \Pi_{\mathbf{x}_0}^{\text{type}'} \end{array} \right\}$$
$$\Omega_{\mathbf{x}_i}^{\text{type}} = \left\{ \mathbf{z}_i = (1, \mathbf{z}'_i) : \mathbf{z}'_1 \in \Omega_{\mathbf{x}_i}^{\text{type}'} \right\} ,$$

where $\text{type} \in \{\text{log-att}, \text{const-dep}, \leq\text{const-thr}, \geq\text{const-thr}\}$ and $i \in [n-1]$.

**Construction 1 (MC-ABE with Repetitions).** *Let* $\text{type} \in \{\text{log-att}, \text{const-loc}, \leq\text{const-thr}, \geq\text{const-thr}\}$. *If* $\text{type} = \text{log-att}$*, pick* $k$ *and* $n$ *such that* $kn = O(\log \lambda)$*. Otherwise, let* $k = n = \text{poly}(\lambda)$*. Our construction uses the following ingredients:*

- *A slotted IPFE scheme* $\text{iFE} = (\text{iSetup}, \text{iKeyGen}, \text{iEnc}, \text{iDec})$ *based on a pairing group* $\mathbb{G} = (\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_t, g_1, g_2, >, e, p)$*.*
- *An identity-based encryption scheme* $\text{idFE} = (\text{idSetup}, \text{idKeyGen}, \text{idEnc}, \text{idDec})$ *with identity space* $\mathcal{I} = \mathcal{L} \times \{0,1\}^{[0;k]}$ *for* $\mathcal{L} = \{0,1\}^{\text{poly}(\lambda)}$ *and message space being the ciphertext space of* iFE

*The MC-ABE scheme* aFE *for* $n$ *clients and the policy class* $\mathcal{F}^{\text{type}}$ *with message space* $\mathcal{M} = \mathbb{G}_t$*, label space* $\mathcal{L}$ *and attribute universe* $\mathcal{X} = \{0,1\}^k$ *works as follows:*

$\mathsf{Setup}(1^\lambda)$ *takes as input the security parameter* $1^\lambda$ *and generates*

$$(\mathsf{iMPK}_0, \mathsf{iMSK}_0) \leftarrow \mathsf{iSetup}(1^\lambda, \{1,2\}, \{3\})$$
$$\left\{(\mathsf{iMPK}_{i,j}^\beta, \mathsf{iMSK}_{i,j}^\beta) \leftarrow \mathsf{iSetup}(1^\lambda, \{1\}, \{2\})\right\}_{(i,j)\in[0;n-1]\times[0;k]}^{\beta\in\{0,1\}}$$
$$\left\{(\mathsf{idMPK}_i, \mathsf{idMSK}_i) \leftarrow \mathsf{idSetup}(1^\lambda)\right\}_{i\in[n-1]} .$$

*Then it outputs* $(\mathsf{MPK}, \mathsf{MSK}, \{\mathsf{SK}_i\}_{i\in[n-1]})$ *as follows:*

$$\mathsf{MPK} = \left(\mathsf{iMPK}_0, \{\mathsf{iMPK}_{i,j}^\beta\}_{(i,j)\in[0;n-1]\times[0;k]}^{\beta\in\{0,1\}}, \{\mathsf{idMPK}_i\}_{i\in[n-1]}\right)$$
$$\mathsf{MSK} = \left(\mathsf{iMSK}_0, \{\mathsf{iMSK}_{i,j}^\beta\}_{(i,j)\in[0;n-1]\times[0;k]}^{\beta\in\{0,1\}}\right)$$
$$\{\mathsf{SK}_i = \mathsf{idMSK}_i\}_{i\in[n-1]} .$$

*We implicitly parse these keys in the algorithms below.*

$\mathsf{Enc}(\mathsf{MPK}, \mathsf{lab}, \mathbf{x}_0, [\![\mu]\!]_t)$ *takes* $\mathsf{MPK}$*, a label* $\mathsf{lab} \in \mathcal{L}$*, an attribute* $\mathbf{x}_0 \in \{0,1\}^k$ *and a message* $[\![\mu]\!]_t \in \mathbb{G}_t$ *as input. We define* $q_0 := |\Pi_{\mathbf{x}_0}^{\mathsf{type}}|$ *and parse* $\Pi_{\mathbf{x}_0}^{\mathsf{type}} = \{\mathbf{y}^\nu\}_{\nu\in[q_0]}$ *where* $\mathbf{y}^\nu = (\mathbf{y}_1^\nu, \ldots, \mathbf{y}_n^\nu)$*. For* $i \in [0; n-1]$*, let* $Y_i^\nu = \{i\} \times \{j \in [0;k] : \mathbf{y}_i^\nu[j] \neq \perp\}$*. For convenience, we also set* $Y^\nu = \bigcup_{i\in[0;n-1]} Y_i^\nu$ *and* $Y_{\geq 1}^\nu = Y^\nu \setminus Y_0^\nu$*. The algorithm samples random elements* $[\![r^1]\!]_1, \ldots, [\![r^{q_0}]\!]_1$*,* $[\![\sigma]\!]_1 \xleftarrow{\$} \mathbb{G}_1$*, computes* $[\![d]\!]_t = [\![\sigma + \mu]\!]_t$ *and generates*

$$\left\{\mathsf{iCT}_0^\nu \leftarrow \mathsf{iSlotEnc}(\mathsf{iMPK}_0, [\![(r^\nu, \sigma)]\!]_1)\right\}^{\nu\in[q_0]}$$
$$\left\{\mathsf{iCT}_{i,j}^\nu \leftarrow \mathsf{iSlotEnc}(\mathsf{iMPK}_{i,j}^{\mathbf{y}_i^\nu[j]}, [\![r^\nu]\!]_1)\right\}_{(i,j)\in Y^\nu}^{\nu\in[q_0]}$$
$$\left\{\mathsf{idCT}_{i,j}^\nu \leftarrow \mathsf{idEnc}(\mathsf{idMPK}_i, (\mathsf{lab}, \mathbf{y}_i^\nu), \mathsf{iCT}_{i,j}^\nu)\right\}_{(i,j)\in Y_{\geq 1}^\nu}^{\nu\in[q_0]} .$$

*Finally, it outputs the ciphertext*

$$\mathsf{CT}_{\mathsf{lab}} = \left([\![d]\!]_t, \{\mathsf{iCT}_0^\nu\}^{\nu\in[q_0]}, \{\mathsf{iCT}_{0,j}^\nu\}_{(0,j)\in Y_0^\nu}^{\nu\in[q_0]}, \{\mathsf{idCT}_{i,j}^\nu\}_{(i,j)\in Y_{\geq 1}^\nu}^{\nu\in[q_0]}\right) .$$

$\mathsf{AKeyGen}(\mathsf{SK}_i, \mathsf{lab}, \mathbf{x}_i)$ *takes as input* $\mathsf{SK}_i$ *for some* $i \in [n-1]$*, a label* $\mathsf{lab} \in \mathcal{L}$ *and an attribute* $\mathbf{x}_i \in \{0,1\}^k$*. We define* $q_i := |\Omega_{\mathbf{x}_i}^{\mathsf{type}}|$ *and parse* $\Omega_{\mathbf{x}_i}^{\mathsf{type}} = \{\mathbf{z}_i^\nu\}_{\nu\in[q_i]}$*. The algorithm outputs* $\mathsf{DK}_{\mathsf{lab},i} = \{\mathsf{idSK}_i^\nu\}_{\nu\in[q_i]}$ *computed as follows:*

$$\mathsf{idSK}_i^\nu \leftarrow \mathsf{idKeyGen}(\mathsf{idMSK}_i, (\mathsf{lab}, \mathbf{z}_i^\nu)) .$$

$\mathsf{PKeyGen}(\mathsf{MSK}, f)$ *takes as input* $\mathsf{MSK}$ *and a policy* $f \in \mathcal{F}^{\mathsf{type}}$*. Let* $\overline{(c \wedge f)} = (\mathbf{M} \in \mathbb{Z}_p^{m\times\ell}, \rho = (\rho_1, \rho_2, \rho_3)\colon [\ell] \rightarrow [0; n-1] \times [0;k] \times \{0,1\})$*. The algorithm samples* $s \xleftarrow{\$} \mathbb{Z}_p$ *and* $\mathbf{u} \xleftarrow{\$} \mathbb{Z}_p^{m-1}$*, computes* $\mathbf{s} = (s, \mathbf{u}^\top) \cdot \mathbf{M}$ *and generates*

$$\mathsf{iSK}_0 \leftarrow \mathsf{iKeyGen}(\mathsf{iMSK}_0, [\![(s,1,0)]\!]_2)$$
$$\left\{\mathsf{iSK}_\kappa \leftarrow \mathsf{iKeyGen}(\mathsf{iMSK}_{\rho_1(\kappa),\rho_2(\kappa)}^{\rho_3(\kappa)}, [\![(\mathbf{s}[\kappa], 0)]\!]_2)\right\}_{\kappa\in[\ell]} .$$

*Finally, it outputs* $\mathsf{DK}_f = \{\mathsf{iSK}_\kappa\}_{\kappa\in[0;\ell]}$*.*

$\mathsf{Dec}(\mathsf{DK}_f, \{\mathsf{DK}_{\mathsf{lab},i}\}_{i\in[n-1]}, \mathsf{CT}_{\mathsf{lab}})$ *takes as input a decryption key* $\mathsf{DK}_f$ *for a policy* $f \in \mathcal{F}^{\mathsf{type}}$, *a set of decryption keys* $\{\mathsf{DK}_{\mathsf{lab},i}\}_{i\in[n-1]}$ *created with respect to attributes* $\mathbf{x}_1, \ldots, \mathbf{x}_{n-1} \in \{0,1\}^k$ *and a label* $\mathsf{lab} \in \mathcal{L}$, *and a ciphertext* $\mathsf{CT}_{\mathsf{lab}}$ *created with respect to an attribute* $\mathbf{x}_0 \in \{0,1\}^k$ *and the same label* $\mathsf{lab}$. *Parse* $\mathsf{DK}_f$, $\{\mathsf{DK}_{\mathsf{lab},i}\}_{i\in[n-1]}$, $\mathsf{CT}_{\mathsf{lab}}$, $\Pi^{\mathsf{type}}_{\mathbf{x}_0}$, $\{\Omega^{\mathsf{type}}_{\mathbf{x}_i}\}_{i\in[n-1]}$ *and* $\{Y^\nu\}_{\nu\in[q_0]}$ *as in the algorithms above. Let* $\overline{(c \wedge f)} = (\mathbf{M} \in \mathbb{Z}_p^{m\times\ell}, \rho\colon [\ell] \to [0;n-1]\times[0;k]\times\{0,1\})$ *The algorithm picks indices* $\nu_0 \in [q_0], \nu_1 \in [q_1], \ldots, \nu_{n-1} \in [q_{n-1}]$ *such that*

1. $\mathbf{y}_i^{\nu_0} = \mathbf{z}_i^{\nu_i}$ *for all* $i \in [n-1]$, *and*
2. $X = \{(i,j,\mathbf{y}_i^{\nu_0}[j]) : (i,j) \in Y^{\nu_0}\} \cap \rho([\ell])$ *satisfies the policy* $\overline{(c \wedge f)}$.

*If no such indices exist, then the algorithm outputs* $\perp$. *Otherwise, it decrypts*

$$\left\{\mathsf{iCT}_{i,j} \leftarrow \mathsf{idDec}(\mathsf{idSK}_i^{\nu_i}, \mathsf{idCT}_{i,j}^{\nu_0})\right\}_{(i,j)\in Y_{\geq 1}^{\nu_0}} ,$$

*and finds coefficients* $\omega_1, \ldots, \omega_\ell \in \mathbb{Z}_p$ *such that* $\sum_{\kappa\in[\ell]} \omega_\kappa \mathbf{M}[\kappa] = \mathbf{e}_1$ *and* $\omega_\kappa = 0$ *for all* $\kappa \notin \rho^{-1}(X)$. *Finally, the algorithm computes*

$$\begin{aligned} &[\![d_0]\!]_t \leftarrow \mathsf{iDec}(\mathsf{iSK}_0, \mathsf{iCT}_0) \\ &\left\{[\![d_\kappa]\!]_t \leftarrow \mathsf{iDec}(\mathsf{iSK}_\kappa, \mathsf{iCT}_{\rho_1(\kappa),\rho_2(\kappa)})\right\}_{\kappa\in\rho^{-1}(X)} ,\end{aligned}$$

*and outputs* $[\![\mu']\!]_t = [\![d]\!]_t - [\![d_0]\!]_t + \sum_{\kappa\in\rho^{-1}(X)} \omega_\kappa [\![d_\kappa]\!]_t$.

A detailed correctness and efficiency analysis can be found in the full version. Moreover, the scheme satisfies selective security with repetitions.

**Proposition 1.** *Let* $\mathsf{type} \in \{\mathsf{log\text{-}att}, \mathsf{const\text{-}dep}, \leq\mathsf{const\text{-}thr}, \geq\mathsf{const\text{-}thr}\}$. *If the* DDH *assumption holds in* $\mathbb{G}_2$, iFE *is slot-mode correct and function-hiding and* idFE *is secure, then Construction 1 is* sel-rep-*secure.*

The proof can be found in the full version.

## 6 Construction of MC-PE

In this section, we present our new compiler that turns any $O(1)$-client ABE into an $O(1)$-client PE scheme for the same policy class.

For each $\ell \in [0; n-1]$, we define a permutation $\pi_\ell$ of the set $[0; n-1]$ via

$$\pi_\ell(i) = \begin{cases} i+1 & \text{if } i \in [0; \ell-1] \\ 0 & \text{if } i = \ell \\ i & \text{if } i \in [\ell+1; n-1] \end{cases} .$$

Correspondingly, for a policy $f \in \mathcal{F}$ and $\ell \in [0; n-1]$, we define $\pi_\ell(f)$ as a variant of $f$ with permuted inputs:

$$\big(\pi_\ell(f)\big)(\mathbf{x}_0, \ldots, \mathbf{x}_{n-1}) = f(\mathbf{x}_{\pi_\ell(0)}, \ldots, \mathbf{x}_{\pi_\ell(n-1)}) .$$

**Construction 2 (Multi-client Predicate Encryption).** *The construction uses the following ingredients:*

- *An MC-ABE scheme* $\mathsf{aFE} = (\mathsf{aSetup}, \mathsf{aEnc}, \mathsf{aAKeyGen}, \mathsf{aPKeyGen}, \mathsf{aDec})$ *for a message space* $\mathcal{M} = \{0,1\}^m$ *for some* $m = \mathrm{poly}(\lambda)$*, a label space* $\mathcal{L}$*, an attribute universe* $\mathcal{X}$ *and a policy class* $\mathcal{F}$*.*
- *A lockable obfuscation scheme* $\mathsf{LObf} = (\mathsf{Obf}, \mathsf{Eval})$ *with lock space* $\mathcal{M}$ *and message space* $\mathcal{M}'$*. We write* $C[x](y)$ *to indicate that a circuit* $C$ *has the value* $x$ *hardwired in its description and takes* $y$ *as input.*

*The MC-PE scheme* $\mathsf{pFE}$ *for message space* $\mathcal{M}'$*, label space* $\mathcal{L}$*, attribute universe* $\mathcal{X}$ *and policy class* $\mathcal{F}$ *works as follows:*

```
hardwired values :
  • an aFE ciphertext $\mathsf{aCT}_i$
  • a set of aFE attribute decryption keys $\{\mathsf{aDK}_{\ell,i+1}\}_{\ell\in[i+1;n-1]}$
input :
  • a set of obfuscated circuits $\{\widetilde{C}_\ell\}_{\ell\in[i+1;n-1]}$
  • a set of aFE attribute decryption keys $\{\mathsf{aDK}_{\ell,j}\}_{\ell\in[i;n-1]}^{j\in[i]}$
  • a set of aFE policy decryption keys $\{\mathsf{aDK}_{\ell,f}\}_{\ell\in[i;n-1]}$
output : an element of the lock space $\sigma_i \in \mathcal{M}$ or $\perp$
initialize $\{\mathsf{aDK}_{\ell,j} \leftarrow \perp\}_{\ell\in[0;j-1]}^{j\in[i+1;n-1]}$
for $k \leftarrow i+1$ to $n-1$ do
    $c_k \leftarrow \mathsf{Eval}(\widetilde{C}_k, (\{\widetilde{C}_\ell\}_{\ell\in[k+1;n-1]}, \{\mathsf{aDK}_{\ell,j}\}_{\ell\in[k;n-1]}^{j\in[k]}, \{\mathsf{aDK}_{\ell,f}\}_{\ell\in[k;n-1]}))$
    if $c_k = \perp$ then return $\perp$
    else parse $\{\mathsf{aDK}_{\ell,k}\}_{\ell\in[0;k-1]} \leftarrow c_k$
end
return $\sigma_i \leftarrow \mathsf{aDec}(\mathsf{aDK}_{i,f}, \{\mathsf{aDK}_{i,j}\}_{j\in[n-1]}, \mathsf{aCT}_i)$
```

**Fig. 5.** Definition of the circuit $C_i[\mathsf{aCT}_i, \{\mathsf{aDK}_{\ell,i+1}\}_{\ell\in[i+1;n-1]}]$ on input $(\{\widetilde{C}_\ell\}_{\ell\in[i+1;n-1]}, \{\mathsf{aDK}_{\ell,j}\}_{\ell\in[i;n-1]}^{j\in[i]}, \{\mathsf{aDK}_{\ell,f}\}_{\ell\in[i;n-1]})$

$\mathsf{Setup}(1^\lambda)$ *takes as input the security parameter* $1^\lambda$ *and generates* $n$ $\mathsf{aFE}$ *instances*

$$\left\{(\mathsf{aMPK}_\ell, \mathsf{aMSK}_\ell, \{\mathsf{aSK}_{\ell,i}\}_{i\in[n-1]}) \leftarrow \mathsf{aSetup}(1^\lambda)\right\}_{\ell\in[0;n-1]} .$$

*Then the algorithm outputs* $(\{\mathsf{SK}_i\}_{i\in[0;n-1]}, \mathsf{MSK})$ *as follows:*

$$\left\{\mathsf{SK}_i := (\mathsf{aMPK}_i, \{\mathsf{aSK}_{\ell,j}\}_{(\ell,j)\in J_i})\right\}_{i\in[0;n-1]} , \quad \mathsf{MSK} := \{\mathsf{aMSK}_\ell\}_{\ell\in[0;n-1]} ,$$

*where* $J_i = \{(\ell, \pi_\ell(i))\}_{\ell\in[0;n-1]\setminus\{i\}}$*. We implicitly parse these keys in the algorithms below.*

$\mathsf{Enc}(\mathsf{SK}_0, \mathsf{lab}, x_0, \mu)$ *takes as input* $\mathsf{SK}_0$*, a label* $\mathsf{lab} \in \mathcal{L}$*, an attribute* $x_0 \in \mathcal{X}$ *and a message* $\mu \in \mathcal{M}'$*. The algorithm samples* $\sigma_0 \xleftarrow{\$} \mathcal{M}$*, runs*

$$\begin{aligned} \mathsf{aCT}_0 &\leftarrow \mathsf{aEnc}(\mathsf{aMPK}_0, \mathsf{lab}, x_0, \sigma_0) \\ \{\mathsf{aDK}_{\ell,1} &\leftarrow \mathsf{aAKeyGen}(\mathsf{aSK}_{\ell,1}, \mathsf{lab}, x_0)\}_{\ell \in [n-1]} \\ \widetilde{C}_0 &\leftarrow \mathsf{Obf}(1^\lambda, C_0[\mathsf{aCT}_0, \{\mathsf{aDK}_{\ell,1}\}_{\ell \in [n-1]}], \mu, \sigma_0) \end{aligned}$$

*and outputs* $\mathsf{CT}_{\mathsf{lab}} := \widetilde{C}_0$*. The circuit* $C_0$ *is described in Fig. 5.*

$\mathsf{AKeyGen}(\mathsf{SK}_i, \mathsf{lab}, x_i)$ *takes as input* $\mathsf{SK}_i$ *for some* $i \in [n-1]$*, a label* $\mathsf{lab} \in \mathcal{L}$ *and an attribute* $x_i \in \mathcal{X}$*. The algorithm samples* $\sigma_i \xleftarrow{\$} \mathcal{M}$*, runs*

$$\begin{aligned} \mathsf{aCT}_i &\leftarrow \mathsf{aEnc}(\mathsf{aMPK}_i, \mathsf{lab}, x_i, \sigma_i) \\ \{\mathsf{aDK}_{\ell,j} &\leftarrow \mathsf{aAKeyGen}(\mathsf{aSK}_{\ell,j}, \mathsf{lab}, x_i)\}_{(\ell,j) \in J_i} \\ \widetilde{C}_i &\leftarrow \mathsf{Obf}(1^\lambda, C_i[\mathsf{aCT}_i, \{\mathsf{aDK}_{\ell,i+1}\}_{\ell \in [i+1;n-1]}], \{\mathsf{aDK}_{\ell,i}\}_{\ell \in [0;i-1]}, \sigma_i) \end{aligned}$$

*and outputs* $\mathsf{DK}_{\mathsf{lab},i} := \widetilde{C}_i$ *with* $C_i$ *being described in Fig. 5.*

$\mathsf{PKeyGen}(\mathsf{MSK}, f)$ *takes as input* $\mathsf{MSK}$ *and a policy* $f \in \mathcal{F}$*, runs*

$$\{\mathsf{aDK}_{\ell,f} \leftarrow \mathsf{aPKeyGen}(\mathsf{aMSK}_\ell, \pi_\ell(f))\}_{\ell \in [0;n-1]} ,$$

*and outputs* $\mathsf{DK}_f := \{\mathsf{aDK}_{\ell,f}\}_{\ell \in [n]}$.

$\mathsf{Dec}(\mathsf{DK}_f, \{\mathsf{DK}_{\mathsf{lab},i}\}_{i \in [n-1]}, \mathsf{CT}_{\mathsf{lab}})$ *takes as input a policy decryption key* $\mathsf{DK}_f = \{\mathsf{aDK}_{\ell,f}\}_{\ell \in [0;n-1]}$*, a set of attribute decryption keys* $\{\mathsf{DK}_{\mathsf{lab},i} = \widetilde{C}_i\}_{i \in [n-1]}$ *and a ciphertext* $\mathsf{CT}_{\mathsf{lab}} = \widetilde{C}_0$*. The algorithm outputs* $\mu'$ *computed as follows:*

$$\mu' \leftarrow \mathsf{Eval}(\widetilde{C}_0, (\{\widetilde{C}_i\}_{i \in [n-1]}, \varnothing, \{\mathsf{aDK}_{\ell,f}\}_{\ell \in [0;n-1]})) .$$

The compiler is correct and efficient for arity $n = O(1)$. Furthermore, it preserves the security level of the underlying MC-ABE.

**Proposition 2.** *Let* $\mathsf{xxx} \in \{\mathsf{sel}, \mathsf{adap}\}$ *and* $\mathsf{yyy} \in \{\mathsf{rep}, \mathsf{norep}\}$*. If* $\mathsf{aFE}$ *is* $\mathsf{xxx}$-$\mathsf{yyy}$-*secure and* $\mathsf{LObf}$ *is secure, then the MC-PE scheme* $\mathsf{pFE}$ *in Construction 2 is also* $\mathsf{xxx}$-$\mathsf{yyy}$-*secure.*

Detailed proofs of correctness, efficiency and security can be found in the full version.

**Acknowledgments.** This work was supported in part by the France 2030 ANR Project ANR-22-PECY-003 SecureCompute.

## References

1. Abdalla, M., Benhamouda, F., Gay, R.: From single-input to multi-client inner-product functional encryption. In: Galbraith, S.D., Moriai, S. (eds.) ASIACRYPT 2019. LNCS, vol. 11923, pp. 552–582. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-34618-8_19

2. Abdalla, M., Benhamouda, F., Kohlweiss, M., Waldner, H.: Decentralizing inner-product functional encryption. In: Lin, D., Sako, K. (eds.) PKC 2019. LNCS, vol. 11443, pp. 128–157. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-17259-6_5
3. Abdalla, M., Catalano, D., Fiore, D., Gay, R., Ursu, B.: Multi-input functional encryption for inner products: function-hiding realizations and constructions without pairings. In: Shacham, H., Boldyreva, A. (eds.) CRYPTO 2018. LNCS, vol. 10991, pp. 597–627. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-96884-1_20
4. Abdalla, M., Gay, R., Raykova, M., Wee, H.: Multi-input inner-product functional encryption from pairings. In: Coron, J.-S., Nielsen, J.B. (eds.) EUROCRYPT 2017. LNCS, vol. 10210, pp. 601–626. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-56620-7_21
5. Agrawal, S., Goyal, R., Tomida, J.: Multi-input quadratic functional encryption from pairings. In: Malkin, T., Peikert, C. (eds.) CRYPTO 2021. LNCS, vol. 12828, pp. 208–238. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-84259-8_8
6. Agrawal, S., Goyal, R., Tomida, J.: Multi-input quadratic functional encryption: stronger security, broader functionality. In: Kiltz, E., Vaikuntanathan, V. (eds.) TCC 2022, Part I. LNCS, vol. 13747, pp. 711–740. Springer, Cham (2022). https://doi.org/10.1007/978-3-031-22318-1_25
7. Agrawal, S., Libert, B., Stehlé, D.: Fully secure functional encryption for inner products, from standard assumptions. In: Robshaw, M., Katz, J. (eds.) CRYPTO 2016. LNCS, vol. 9816, pp. 333–362. Springer, Heidelberg (2016). https://doi.org/10.1007/978-3-662-53015-3_12
8. Agrawal, S., Rossi, M., Yadav, A., Yamada, S.: Constant input attribute based (and predicate) encryption from evasive and tensor LWE. In: Handschuh, H., Lysyanskaya, A. (eds.) CRYPTO 2023, Part IV. LNCS, vol. 14084, pp. 532–564. Springer, Cham (2023). https://doi.org/10.1007/978-3-031-38551-3_17
9. Agrawal, S., Tomida, J., Yadav, A.: Attribute-based multi-input FE (and more) for attribute-weighted sums. In: Handschuh, H., Lysyanskaya, A. (eds.) CRYPTO 2023, Part IV. LNCS, vol. 14084, pp. 464–497. Springer, Cham (2023). https://doi.org/10.1007/978-3-031-38551-3_15
10. Agrawal, S., Wichs, D., Yamada, S.: Optimal Broadcast Encryption from LWE and Pairings in the Standard Model. In: Pass, R., Pietrzak, K. (eds.) TCC 2020. LNCS, vol. 12550, pp. 149–178. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-64375-1_6
11. Agrawal, S., Yadav, A., Yamada, S.: Multi-input attribute based encryption and predicate encryption. In: Dodis, Y., Shrimpton, T. (eds.) CRYPTO 2022, Part I. LNCS, vol. 13507, pp. 590–621. Springer, Cham (2022). https://doi.org/10.1007/978-3-031-15802-5_21
12. Ambrona, M., Gay, R.: Multi-authority ABE, revisited. Cryptology ePrint Archive, Report 2021/1381 (2021). https://eprint.iacr.org/2021/1381
13. Ambrona, M., Gay, R.: Multi-authority ABE for non-monotonic access structures. In: Boldyreva, A., Kolesnikov, V. (eds.) PKC 2023, Part II. LNCS, vol. 13941, pp. 306–335. Springer, Cham (2023). https://doi.org/10.1007/978-3-031-31371-4_11
14. Ananth, P., Jain, A.: Indistinguishability obfuscation from compact functional encryption. In: Gennaro, R., Robshaw, M. (eds.) CRYPTO 2015. LNCS, vol. 9215, pp. 308–326. Springer, Heidelberg (2015). https://doi.org/10.1007/978-3-662-47989-6_15

15. Barak, B., et al.: On the (im)possibility of obfuscating programs. In: Kilian, J. (ed.) CRYPTO 2001. LNCS, vol. 2139, pp. 1–18. Springer, Heidelberg (2001). https://doi.org/10.1007/3-540-44647-8_1
16. Beimel, A.: Secure schemes for secret sharing and key distribution. Ph.D. thesis, Technion - Israel Institute of Technology, Haifa, Israel (1996). https://www.cs.bgu.ac.il/~beimel/Papers/thesis.pdf
17. Benaloh, J., Leichter, J.: Generalized secret sharing and monotone functions. In: Goldwasser, S. (ed.) CRYPTO 1988. LNCS, vol. 403, pp. 27–35. Springer, New York (1990). https://doi.org/10.1007/0-387-34799-2_3
18. Boneh, D., Sahai, A., Waters, B.: Functional encryption: definitions and challenges. In: Ishai, Y. (ed.) TCC 2011. LNCS, vol. 6597, pp. 253–273. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-19571-6_16
19. Boneh, D., Waters, B.: Conjunctive, subset, and range queries on encrypted data. In: Vadhan, S.P. (ed.) TCC 2007. LNCS, vol. 4392, pp. 535–554. Springer, Heidelberg (2007). https://doi.org/10.1007/978-3-540-70936-7_29
20. Brakerski, Z., Jain, A., Komargodski, I., Passelègue, A., Wichs, D.: Non-trivial witness encryption and null-iO from standard assumptions. In: Catalano, D., De Prisco, R. (eds.) SCN 18. LNCS, vol. 11035, pp. 425–441. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-98113-0_23
21. Chen, J., Lim, H.W., Ling, S., Wang, H., Wee, H.: Shorter IBE and signatures via asymmetric pairings. In: Abdalla, M., Lange, T. (eds.) Pairing 2012. LNCS, vol. 7708, pp. 122–140. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-36334-4_8
22. Chotard, J., Dufour Sans, E., Gay, R., Phan, D.H., Pointcheval, D.: Decentralized multi-client functional encryption for inner product. In: Peyrin, T., Galbraith, S. (eds.) ASIACRYPT 2018. LNCS, vol. 11273, pp. 703–732. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-03329-3_24
23. Chotard, J., Dufour-Sans, E., Gay, R., Phan, D.H., Pointcheval, D.: Dynamic decentralized functional encryption. In: Micciancio, D., Ristenpart, T. (eds.) CRYPTO 2020. LNCS, vol. 12170, pp. 747–775. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-56784-2_25
24. Datta, P., Okamoto, T., Tomida, J.: Full-hiding (unbounded) multi-input inner product functional encryption from the $k$-linear assumption. In: Abdalla, M., Dahab, R. (eds.) PKC 2018. LNCS, vol. 10770, pp. 245–277. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-76581-5_9
25. Escala, A., Herold, G., Kiltz, E., Ràfols, C., Villar, J.: An algebraic framework for Diffie-Hellman assumptions. In: Canetti, R., Garay, J.A. (eds.) CRYPTO 2013. LNCS, vol. 8043, pp. 129–147. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-40084-1_8
26. Francati, D., Friolo, D., Malavolta, G., Venturi, D.: Multi-key and multi-input predicate encryption from learning with errors. In: Hazay, C., Stam, M. (eds.) EUROCRYPT 2023, Part III. LNCS, vol. 14006, pp. 573–604. Springer, Cham (2023). https://doi.org/10.1007/978-3-031-30620-4_19
27. Garg, S., Gentry, C., Halevi, S., Raykova, M., Sahai, A., Waters, B.: Candidate indistinguishability obfuscation and functional encryption for all circuits. In: 54th FOCS, pp. 40–49. IEEE Computer Society Press (2013). https://doi.org/10.1109/FOCS.2013.13
28. Goldwasser, S., et al.: Multi-input functional encryption. In: Nguyen, P.Q., Oswald, E. (eds.) EUROCRYPT 2014. LNCS, vol. 8441, pp. 578–602. Springer, Heidelberg (2014). https://doi.org/10.1007/978-3-642-55220-5_32

29. Goyal, R., Koppula, V., Vusirikala, S., Waters, B.: On perfect correctness in (lockable) obfuscation. In: Pass, R., Pietrzak, K. (eds.) TCC 2020. LNCS, vol. 12550, pp. 229–259. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-64375-1_9
30. Goyal, R., Koppula, V., Waters, B.: Lockable obfuscation. In: Umans, C. (ed.) 58th FOCS, pp. 612–621. IEEE Computer Society Press (2017). https://doi.org/10.1109/FOCS.2017.62
31. Goyal, V., Pandey, O., Sahai, A., Waters, B.: Attribute-based encryption for fine-grained access control of encrypted data. In: Juels, A., Wright, R.N., De Capitani di Vimercati, S. (eds.) ACM CCS 2006, pp. 89–98. ACM Press (2006). https://doi.org/10.1145/1180405.1180418, available as Cryptology ePrint Archive Report 2006/309
32. Jutla, C.S., Roy, A.: Shorter quasi-adaptive NIZK proofs for linear subspaces. J. Cryptol. **30**(4), 1116–1156 (2016). https://doi.org/10.1007/s00145-016-9243-7
33. Lewko, A., Waters, B.: New techniques for dual system encryption and fully secure HIBE with short ciphertexts. In: Micciancio, D. (ed.) TCC 2010. LNCS, vol. 5978, pp. 455–479. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-11799-2_27
34. Lewko, A., Waters, B.: Decentralizing attribute-based encryption. In: Paterson, K.G. (ed.) EUROCRYPT 2011. LNCS, vol. 6632, pp. 568–588. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-20465-4_31
35. Li, H., Lin, H., Luo, J.: ABE for circuits with constant-size secret keys and adaptive security. In: Kiltz, E., Vaikuntanathan, V. (eds.) TCC 2022, Part I. LNCS, vol. 13747, pp. 680–710. Springer, Cham (2022). https://doi.org/10.1007/978-3-031-22318-1_24
36. Libert, B., Ţiţiu, R.: Multi-client functional encryption for linear functions in the standard model from LWE. In: Galbraith, S.D., Moriai, S. (eds.) ASIACRYPT 2019. LNCS, vol. 11923, pp. 520–551. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-34618-8_18
37. Lin, H.: Indistinguishability obfuscation from SXDH on 5-linear maps and locality-5 PRGs. In: Katz, J., Shacham, H. (eds.) CRYPTO 2017. LNCS, vol. 10401, pp. 599–629. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-63688-7_20
38. Lin, H., Luo, J.: Compact adaptively secure ABE from $k$-Lin: beyond $\mathsf{NC}^1$ and towards $\mathsf{NL}$. In: Canteaut, A., Ishai, Y. (eds.) EUROCRYPT 2020. LNCS, vol. 12107, pp. 247–277. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-45727-3_9
39. Lin, H., Luo, J.: Succinct and adaptively secure ABE for ABP from $k$-Lin. In: Moriai, S., Wang, H. (eds.) ASIACRYPT 2020, Part III. LNCS, vol. 12493, pp. 437–466. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-64840-4_15
40. Lin, H., Vaikuntanathan, V.: Indistinguishability obfuscation from DDH-like assumptions on constant-degree graded encodings. In: Dinur, I. (ed.) 57th FOCS, pp. 11–20. IEEE Computer Society Press (2016). https://doi.org/10.1109/FOCS.2016.11
41. Nguyen, K., Phan, D.H., Pointcheval, D.: Multi-client functional encryption with fine-grained access control. In: Agrawal, S., Lin, D. (eds.) ASIACRYPT 2022, Part I. LNCS, vol. 13791, pp. 95–125. Springer, Cham (2022). https://doi.org/10.1007/978-3-031-22963-3_4
42. Sahai, A., Waters, B.: Fuzzy identity-based encryption. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 457–473. Springer, Heidelberg (2005). https://doi.org/10.1007/11426639_27

43. Shi, E., Bethencourt, J., Chan, H.T.H., Song, D.X., Perrig, A.: Multi-dimensional range query over encrypted data. In: 2007 IEEE Symposium on Security and Privacy, pp. 350–364. IEEE Computer Society Press (2007). https://doi.org/10.1109/SP.2007.29
44. Tomida, J.: Tightly secure inner product functional encryption: multi-input and function-hiding constructions. In: Galbraith, S.D., Moriai, S. (eds.) ASIACRYPT 2019. LNCS, vol. 11923, pp. 459–488. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-34618-8_16
45. Tsabary, R.: Candidate witness encryption from lattice techniques. In: Dodis, Y., Shrimpton, T. (eds.) CRYPTO 2022, Part I. LNCS, vol. 13507, pp. 535–559. Springer, Cham (2022). https://doi.org/10.1007/978-3-031-15802-5_19
46. Vaikuntanathan, V., Wee, H., Wichs, D.: Witness encryption and null-IO from evasive LWE. In: Agrawal, S., Lin, D. (eds.) ASIACRYPT 2022, Part I. LNCS, vol. 13791, pp. 195–221. Springer, Cham (2022). https://doi.org/10.1007/978-3-031-22963-3_7
47. Waters, B.: Dual system encryption: realizing fully secure IBE and HIBE under simple assumptions. In: Halevi, S. (ed.) CRYPTO 2009. LNCS, vol. 5677, pp. 619–636. Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-642-03356-8_36
48. Wee, H.: Attribute-hiding predicate encryption in bilinear groups, revisited. In: Kalai, Y., Reyzin, L. (eds.) TCC 2017. LNCS, vol. 10677, pp. 206–233. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-70500-2_8
49. Wichs, D., Zirdelis, G.: Obfuscating compute-and-compare programs under LWE. In: Umans, C. (ed.) 58th FOCS, pp. 600–611. IEEE Computer Society Press (2017). https://doi.org/10.1109/FOCS.2017.61

# Adaptively Secure Attribute-Based Encryption from Witness Encryption

Brent Waters[1,2](✉) and Daniel Wichs[2,3]

[1] University of Texas at Austin, Austin, TX, USA
bwaters@cs.utexas.edu
[2] NTT Research, Sunnyvale, CA, USA
[3] Northeastern University, Boston, MA, USA

**Abstract.** *Attribute-based encryption (ABE)* enables fine-grained control over which ciphertexts various users can decrypt. A master authority can create secret keys $\mathsf{sk}_f$ with different functions (circuits) $f$ for different users. Anybody can encrypt a message under some attribute $x$ so that only recipients with a key $\mathsf{sk}_f$ for a function such that $f(x) = 1$ will be able to decrypt. There are a number of different approaches toward achieving *selectively secure* ABE, where the adversary has to decide on the challenge attribute $x$ ahead of time before seeing any keys, including constructions via bilinear maps (for NC1 circuits), learning with errors, or witness encryption. However, when it comes *adaptively secure* ABE, the problem seems to be much more challenging and we only know of two potential approaches: via the "dual systems" methodology from bilinear maps, or via indistinguishability obfuscation. In this work, we give a new approach that constructs adaptively secure ABE from witness encryption (along with statistically sound NIZKs and one-way functions). While witness encryption is a strong assumption, it appears to be fundamentally weaker than indistinguishability obfuscation. Moreover, we have candidate constructions of witness encryption from some assumptions (e.g., evasive LWE) from which we do not know how to construct indistinguishability obfuscation, giving us adaptive ABE from these assumptions as a corollary of our work.

## 1 Introduction

Attribute-Based Encryption (ABE) [SW05] is an advanced form of encryption where the user's ability to decrypt ciphertexts is governed by a policy attached to their key. In such a system a ciphertext encrypting a message $m$ is associated with a attribute string $x$. A secret key in turn will be issued by some authority which associates it with some predicate function $f$ to generate $\mathsf{sk}_f$. Decryption semantics dictate that $\mathsf{sk}_f$ will be able to decrypt a ciphertext associated with attribute $x$ if $f(x) = 1$. A system is informally said to be secure if no attacker can distinguish between an encryption of message $m_0$ from $m_1$ under attribute $x^*$ so long as it only obtains secret keys for functions $f_1, \ldots, f_q$ where $f_i(x^*) = 0$. Over the past two decades ABE has emerged as an important construct for both encrypted access control as well as at tool for building other cryptographic primitives (e.g., [PRV12, GKP+13]).

The first constructions of Attribute-Based Encryption [SW05, GPSW06] utilized groups with efficiently computable bilinear maps and supported functions that could be expressed as boolean formulas or circuits of logarithmic depth in the security parameter. Several years later construction based on lattices [GVW13, BGG+14] emerged that were provably secure from the learning with errors (LWE) [Reg05] assumption. Remarkably, these construction supported policies that could be expressed as any circuit of a priori bounded depth and thus in principle of any function of fixed runtime. Around the same time a third avenue for realizing ABE systems manifested when Garg, Gentry, Sahai and Waters [GGSW13] proposed the concept of witness encryption and showed how to build ABE from it. Witness is encryption is a powerful, yet general primitive where one encrypts a message $m$ to a statement $z$ and decryption is achievable for any decryptor which knows a witness $w$ such that $R(z, w) = 1$ for some family of relations indexed by the security parameter.

Proving security is a central and involved part of building ABE systems. All three (bilinear map, LWE and witness encryption) paths for realizing Attribute-Based Encryption first established solutions in the selective model of security where an attacker declares an attribute string $x^*$ *before* seeing either the public parameters of the system or receiving any private keys. This notion is meaningful; however, if fails to capture many "real life" attacks where an attacker might somehow influence the attribute string of a ciphertext in a way that depends on such information. While we can bridge the gap from selective to adaptive security using a complexity leveraging guessing strategy in conjunction with subexponential hardness assumptions, this is somewhat unsatisfactory both from the stronger assumption requirement and from an intellectual understanding standpoint.

Over the years achieving adaptive security has borne out to be quite challenging. Unlike Identity-Based Encryption (IBE) [Sha84] which admits a varied number of approaches [BF01, BB04, Wat05, Gen06, Wat09, DG17, Tsa19], ABE systems must maintain the "structure" and semantics of the attribute string which rule out many hashing techniques. Going further it was formally shown [LW14] that one cannot prove adaptive security using "partitioning" reductions which were integral to proving security for many IBE schemes.[1]

The first solutions [LOS+10] for adaptively secure Attribute-Based Encryption applied the dual system encryption methodology of Waters [Wat09] using bilinear groups. In a dual system encryption proof, the challenge ciphertext is first changed to a semi-functional form. Following this each secret key issued will be changed one at a time to a semi-functional form which is inherently incompatible with the challenge ciphertext, but still compatible with all other normally generated ciphertexts. Unfortunately, to this point it has proven difficult to find adaptations of these ideas to either the LWE or witness

[1] A weaker notion called semi-adaptive security [BV16, GKW16] is known to be significantly easier to achieve, but appears to still be far from fully adaptive security.

encryption avenues described above. (One exception is the work of [Tsa19] that gives an ABE system for a subset functionality which is more expressive than IBE string matching, but well short of ABE for boolean formulas or circuits.) From the learning with errors side, the algebraic analogs of bilinear map tools have not come fully to fruition. While witness encryption is a powerful primitive in some ways, it is arguably quite limited in others. In particular, it lacks the "hidden computation" aspect that is present in the more powerful concept of indistinguishability obfuscation. As such the only solutions for achieving adaptively secure ABE beyond bilinear maps have required indistinguishability obfuscation or functional encryption [Wat15, ABSV15] which precisely rely on such hidden computation properties.

*Our Results: Adaptive ABE from Witness Encryption.* In this work, we construct adaptively secure attribute-based encryption from witness encryption along with statistically sound NIZKs and one-way functions. At a high level, we do so by showing how to employ dual system encryption techniques using witness encryption.

This is both an important and technically challenging endeavor. While we already had adaptive ABE from indistinguishability obfuscation (iO) [Wat15, ABSV15] have recently seen iO proven from "well founded" assumptions [JLS21], witness encryption appears to be a fundamentally weaker primitive than iO. For example, we have black-box separations showing that witness encryption does not generically imply iO [GMM17]. Furthermore, witness encryption may admit solutions from a broader set of cryptographic assumptions. Two recent examples include the witness encryption built from variants of the evasive LWE assumption [Tsa22, VWW22] as well as a direction towards achieving witness encryption from pairing free groups [BIOW20]. Therefore, we get adaptively secure ABE from (e.g.,) evasive LWE as a corollary of our work. Overall, similarly to the recent work of [FWW23], we view the construction of advanced cryptosystems from *plain* witness encryption rather than iO as a well motivated and worthwhile endeavour.

Technically, witness encryption does not seem to support any form of hidden computation and thus appears to be incompatible with developing dual system encryption type proofs where we want to incrementally and undetectably change the form of the challenge ciphertext and private keys to make them mutually exclusive in a working decryption operation. We surmount this challenge by developing new tools and techniques for bringing in "outside" cryptography primitives to augment witness encryption to allow for such an argument.

## 1.1 Technical Overview

*Selective ABE from WE.* The prior work of [GGSW13] constructed selectively secure ABE from witness encryption. The main idea behind their solution is to set the master public/secret key to be a the verification/signing key for a special type of signature scheme. The secret keys $\mathsf{sk}_f$ are signature of the functions $f$, and an encryption under an attribute $x$ is a witness encryption that there exists some signature for some function $f$ such that $f(x) = 1$. In the proof of security, we can indistinguishably "constrain" the special signature scheme on the challenge attribute $x^*$ so that there only exist valid signatures $\pi$ for functions $f$ for which $f(x^*) = 0$. Then the security of witness encryption ensures that the message is hidden. The signature itself is implemented using statistically binding commitments and statistically sound NIZKs. Unfortunately, this proof strategy inherently only achieves selective security since we need to know the challenge attribute $x^*$ when creating the master public key of the ABE.

*Overview of Our Approach.* While our approach can also be seen relying on a special form of constrained signatures instantiated from commitments and NIZKs, the way we use these to achieve adaptive security is more sophisticated and is inspired by dual-system techniques [Wat09,LOS+10]. There are three main elements of our construction: (a) we introduce a new notion called a *functional tag system*, (b) we use a functional tag system to construct adaptive ABE from witness encryption (together with statistically binding commitments and statistically sound NIZKs), (c) we show how to construct a functional tag system from one-way functions. We now elaborate on each of these elements one by one.[2]

*Functional Tag System.* A *functional tag system* allows us to generate "input tags" $\mathsf{tag}_x$ for inputs $x$ and "function tags" $\mathsf{tag}_f$ for functions (i.e., circuits) $f$. There is a dummy (D) way to generate such tags $\mathsf{tag}_x \leftarrow \mathsf{DInputTag}(x), \mathsf{tag}_f \leftarrow \mathsf{DFunctionTag}(f)$ randomly and independently of each other. There is also a smart (S) way to generate these using some common secret key $\mathsf{tsk}$ with $\mathsf{tag}_x \leftarrow \mathsf{SInputTag}(\mathsf{tsk}, x), \mathsf{tag}_f \leftarrow \mathsf{SFunctionTag}(\mathsf{tsk}, f)$. There is an efficient predicate $\mathsf{Trigger}(\mathsf{tag}_f, \mathsf{tag}_x)$ that checks if some pair of function/input tag "trigger". Dummy pairs of tags trigger with only negligible probability. Smart pairs of tags generated using a common key $\mathsf{tsk}$ always trigger if $f(x) = 1$. A fully adaptive adversary who gets to see a single input tag $\mathsf{tag}_x$ for an input $x$ and many function tags $\mathsf{tag}_{f_i}$ for functions $f_i$ cannot tell the difference between seeing all dummy tags versus all smart tags generated using a common key $\mathsf{tsk}$ as long as $f_i(x) = 0$ for all $i$.

*ABE from WE via a Functional Tag System.* We set the master public/secret key of the ABE to be the verification/signing key for a special type of "constrained" signature scheme, described later on. Each function secret key $\mathsf{sk}_f = (f, \mathsf{tag}_f, \pi)$ consists of a randomly generated "dummy function tag" $\mathsf{tag}_f \leftarrow \mathsf{DFunctionTag}(f)$ along with a signature $\pi$ of the pair $(f, \mathsf{tag}_f)$. To encrypt a message under an attribute $x$, we generate a "dummy input tag" $\mathsf{tag}_x \leftarrow \mathsf{DInputTag}(x)$ and send it along with a witness encryption of the

[2] In the main body, we reverse the order and present (c) before (b), but for the introduction we prefer this ordering.

message under the NP statement "there exists some pair $(f, \mathsf{tag}_f)$ that has a valid signature $\pi$ such that $f(x) = 1$ and $\mathsf{Trigger}(\mathsf{tag}_f, \mathsf{tag}_x) = 0$".[3]

In the proof of security, we first switch to using "smart function tags" $\mathsf{tag}_f \leftarrow \mathsf{SFunctionTag}(\mathsf{tsk}, f)$ in the secret keys $\mathsf{sk}_f$ and a "smart input tag" $\mathsf{tag}_x \leftarrow \mathsf{SInputTag}(\mathsf{tsk}, x)$ in the challenge ciphertext, all generated using a common key $\mathsf{tsk}$. By the adaptive security of the functional tag system, this is indistinguishable. We then indistinguishably "constrain" the special signature scheme so that valid signatures $\pi$ only exist for pairs $(f, \mathsf{tag}_f)$ where $\mathsf{tag}_f \leftarrow \mathsf{SFunctionTag}(\mathsf{tsk}, f)$. Finally, we argue that the NP statement used for the witness encryption is false, and therefore witness encryption security ensures that the encrypted message is hidden. This holds because whenever $\pi$ is a valid signature of $(f, \mathsf{tag}_f)$ then it must be the case that $\mathsf{tag}_f \leftarrow \mathsf{SFunctionTag}(\mathsf{tsk}, f)$, and if $f(x) = 1$, then it must also be the case that $\mathsf{Trigger}(\mathsf{tag}_f, \mathsf{tag}_x) = 1$.

The special constrained signature scheme is constructed from statistically binding commitments and statistically sound NIZKs as follows. The verification key consist of two commitments $\mathsf{com}_0, \mathsf{com}_1$ to 0, along with the CRS of the NIZK; the signing key is a decommitment of $\mathsf{com}_0$. The signature $\pi$ for a pair $(f, \mathsf{tag}_f)$ is a NIZK proof that "either $\mathsf{com}_0$ is a commitment to 0 or $\mathsf{com}_1$ is a commitment to $\mathsf{tsk}$ and there is some randomness $r$ such that $\mathsf{tag}_f = \mathsf{SFunctionTag}(\mathsf{tsk}, f; r)$". The NIZK proof is generated using the decommitment of $\mathsf{com}_0$ as the witness. To constrain the signature, we set $\mathsf{com}_0$ to be a commitment to 1, $\mathsf{com}_1$ to be a commitment to $\mathsf{tsk}$ and we generate the NIZKs using the the decomitment to $\mathsf{com}_1$ and the randomness used to generate $\mathsf{tag}_f$ as the witness. The corresponding constrained verification key and signatures are indistinguishable.

*Functional Tag System from One-Way Functions.* Finally, we construct a functional tag system from one-way functions using "blind garbled circuits" [BLSV18]. In blind garbled circuits, for any distribution over input $x$ and circuit $C$ for which $C(x)$ is uniformly random, the corresponding garbled input/circuit pair $\widetilde{x}, \widetilde{C}$ look like uniformly random bits. We rely on a slightly more complex version of blind garbled circuits where the adversary can see many different garbled circuits $\widetilde{C}_i$ but only one garbled input $\widetilde{x}$; furthermore we allow *semi-adaptive security* where the circuits and the input can be chosen adaptively, *but* the challenge circuit must be chosen after the input. The detailed definition is somewhat cumbersome and we defer it to the main body, but we show that the basic "point and permute" construction of garbled circuits from one-way functions achieves this notion similarly to [BLSV18].

To construct a functional tag system from blind garbled circuits, we set dummy input tags $\mathsf{tag}_x$ and dummy function tags $\mathsf{tag}_f$ to be uniformly random

[3] We note that, in contrast to the selectively secure ABE schemes from LWE of [GVW13,BGG+14], our ABE is not succinct and the encryption run-time and ciphertext size scales with the circuit size of the supported functions $f$. Constructing even selectively secure succinct ABE from Witness Encryption is an intriguing open problem.

values of appropriate size. To determine if a input/function tag pair $(\mathsf{tag}_f, \mathsf{tag}_x)$ "triggers" we interpret $\mathsf{tag}_x = \widetilde{x}$ as a garbled input and $\mathsf{tag}_f = (\widetilde{C}, t)$ as a garbled circuit together with a target value $t$ of length security parameter, and output 1 if the evaluation of the garbled circuit $\widetilde{C}$ on the garbled input $\widetilde{x}$ produces the target value $t$. In the dummy case, this only happens with negligible probability, ensuring "dummy correctness". A smart input tag for $x$ consists of a correctly garbled input $\mathsf{tag}_x = \widetilde{x}$, and a smart function tag for $f$ consists of $\mathsf{tag}_f = (\widetilde{C}, t)$ where $t$ is a random target value and $\widetilde{C}$ is a garbling of the circuit $C$ that evaluates $f(x)$ and if the output is 1 it outputs the target value $t$ else it outputs a random independent value $u$. This ensures that a smart input/function tag pair $\mathsf{tag}_x, \mathsf{tag}_f$ does trigger when $f(x) = 1$.

For security, we intuitively want to rely on blind garbled circuits to ensure that we can replace dummy function tags with smart ones in the case where $f(x) = 0$, by relying on the fact that the circuit $C(x)$ outputs a random independent value $u$ in this case. However, there is an issue with adaptivity. Blind garbled circuits only provide *semi-adaptive* security, where the challenge circuit $C$ must be chosen after the input $x$, while functional tag systems require *fully adaptive* security where the challenge functions $f$ can be chosen before or after the input $x$. We resolve this issue using techniques developed in the study of adaptively secure garbled circuits [HJO+16]. In particular, we encrypt the garbled circuit with a "somewhere equivocal PRF" whose key is part of the input tag. For any circuit $C$ chosen before the input $x$, this allows us to give a fake ciphertext inside $\mathsf{tag}_f$ and only later equivocate the garbled circuit $\widetilde{C}$ inside the ciphertext after the input $x$ is chosen, in affect allowing $\widetilde{C}$ to depend on $x$ inside the security proof. Therefore, we can rely on semi-adaptive security of the blind garbled circuits to achieve fully adaptive security of the functional tag system.

## 2 Preliminaries

For any integer $n \geq 1$, define $[n] = \{1, \ldots, n\}$. A function $\nu : \mathbb{N} \to \mathbb{N}$ is said to be negligible, denoted $\nu(n) = \mathrm{negl}(n)$, if for every positive polynomial $p(\cdot)$ and all sufficiently large $n$ it holds that $\nu(n) < 1/p(n)$. We use the abbreviation PPT for probabilistic polynomial time. For a finite set $S$, we write $a \leftarrow S$ to mean $a$ is sampled uniformly randomly from $S$. For a randomized algorithm $A$, we let $a \leftarrow A(\cdot)$ denote the process of running $A(\cdot)$ and assigning the outcome to $a$; when $A$ is deterministic, we write $a := A(\cdot)$ instead. For a randomized algorithm $A$ we use the notation $a := A(\cdot; r)$ to denote the process of running the randomized algorithm $A$ with some fixed randomness $r$. We denote the security parameter by $\lambda$. For two distributions $X, Y$ parameterized by $\lambda$ we say that they are computationally indistinguishable, denoted by $X \approx_c Y$ if for every PPT distinguisher $D$ we have $|\Pr[D(X) = 1] - \Pr[D(Y) = 1]| = \mathrm{negl}(\lambda)$.

### 2.1 Attribute Based Encryption (ABE)

We define an ABE scheme with adaptive security.

**Definition 1 (Attribute-Based Encryption (ABE)).** *An ABE scheme a function class* $\mathcal{F}_\lambda \subseteq \{f : \{0,1\}^{n(\lambda)} \to \{0,1\}\}$ *consists of PPT procedures* $(\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ *with the following syntax:*

- $(\mathsf{mpk}, \mathsf{msk}) \leftarrow \mathsf{Setup}(1^\lambda)$*: Generates a master public key* $\mathsf{mpk}$ *and master secret key* $\mathsf{msk}$.
- $\mathsf{sk}_f \leftarrow \mathsf{KeyGen}(\mathsf{msk}, f)$*: Generates a function key* $\mathsf{sk}_f$ *for a function* $f \in \mathcal{F}_\lambda$.
- $\mathsf{ct} \leftarrow \mathsf{Enc}(\mathsf{mpk}, x, b)$*: Given an attribute* $x \in \{0,1\}^{n(\lambda)}$ *and a bit* $b \in \{0,1\}$ *outputs a ciphertext* $\mathsf{ct}$.
- $b := \mathsf{Dec}(\mathsf{sk}_f, \mathsf{ct})$*: Decrypts* $\mathsf{ct}$ *using* $\mathsf{sk}_f$.

*We require correctness and adaptive security defined as follows:*

- ***Correctness:*** *There is some negligible function* $\mu$ *such that for all* $\lambda \in \mathbb{N}$ *all* $f \in \mathcal{F}_\lambda$ *all* $x \in \{0,1\}^{n(\lambda)}$ *such that* $f(x) = 1$ *all* $b \in \{0,1\}$ *we have:*

$$\Pr\left[\mathsf{Dec}(\mathsf{sk}_f, \mathsf{ct}) = b \ : \ \begin{array}{l} (\mathsf{mpk}, \mathsf{msk}) \leftarrow \mathsf{Setup}(1^\lambda) \\ sk_f \leftarrow \mathsf{KeyGen}(\mathsf{msk}, f) \\ \mathsf{ct} \leftarrow \mathsf{Enc}(\mathsf{mpk}, x, b) \end{array}\right] \leq \mu(\lambda).$$

- ***Adaptive Security:*** *We define the game* $\mathsf{ABEGame}^b_{\mathcal{A}}(1^\lambda)$ *between a challenger and an stateful adversary* $\mathcal{A}(1^\lambda)$ *as follows:*
  - *The challenger chooses* $(\mathsf{mpk}, \mathsf{msk}) \leftarrow \mathsf{Setup}(1^\lambda)$ *and gives* $\mathsf{mpk}$ *to* $\mathcal{A}$.
  - Pre-challenge key queries: *The adversary can make arbitrarily many queries* $f_i \in \mathcal{F}_\lambda$ *and the challenger responds with* $\mathsf{sk}_{f_i} \leftarrow \mathsf{KeyGen}(\mathsf{msk}, f_i)$.
  - Challenge ciphertext: *The adversary chooses an attribute* $x \in \{0,1\}^{n(\lambda)}$ *such that* $f_i(x) = 0$ *for all pre-challenge key queries* $f_i$*, and the challenger responds with the challenge ciphertext* $\mathsf{ct} \leftarrow \mathsf{Enc}(\mathsf{mpk}, x, b)$.
  - Post-challenge key queries: *The adversary can make arbitrarily many additional queries* $f_i \in \mathcal{F}_\lambda$ *such that* $f_i(x) = 0$ *and the challenger responds with* $\mathsf{sk}_{f_i} \leftarrow \mathsf{KeyGen}(\mathsf{msk}, f_i)$.
  - *The adversary output a bit* $b'$ *which is the output of the game.*

  *We require that for all PPT* $\mathcal{A}$ *we have*

$$\left|\Pr[\mathsf{ABEGame}^0_{\mathcal{A}}(1^\lambda) = 1] - \Pr[\mathsf{ABEGame}^1_{\mathcal{A}}(1^\lambda) = 1\right| \leq \mathrm{negl}(\lambda).$$

*An* ABE for circuits *allows us to instantiate an ABE scheme for the function class* $\mathcal{C}^{s,n}_\lambda$ *consisting of boolean circuits of size* $s(\lambda)$ *with* $n(\lambda)$*-bit input, for any polynomials* $s(\lambda), n(\lambda)$.

## 2.2 Commitments

We define statistically binding commitments in the Common Reference String (CRS) model.

**Definition 2 (Statistically Binding Commitments).** *A commitment scheme consists of PPT algorithms* $(\mathsf{Setup}, \mathsf{Commit})$ *with the following syntax:*

- $\mathsf{crs} \leftarrow \mathsf{Setup}(1^\lambda)$: *generates a common reference string* $\mathsf{crs}$.
- $\mathsf{com} := \mathsf{Commit}_{\mathsf{crs}}(b; r)$: *generates a commitment* $\mathsf{com}$ *to a bit* $b \in \{0,1\}$ *using randomness* $r \in \{0,1\}^\lambda$.

*We require hiding and statistical binding:*

- ***Hiding:*** *We have* $(\mathsf{crs}, \mathsf{com}_0) \approx_c (\mathsf{crs}, \mathsf{com}_1)$ *where* $\mathsf{crs} \leftarrow \mathsf{Setup}(1^\lambda), \mathsf{com}_b \leftarrow \mathsf{Commit}_{\mathsf{crs}}(b)$.
- ***Statistical Binding:*** *We say that a* $\mathsf{crs}$ *is* binding *if there do not exist any* $r_0, r_1$ *such that* $\mathsf{Commit}_{\mathsf{crs}}(0; r_0) = \mathsf{Commit}_{\mathsf{crs}}(1; r_1)$. *We require that:* $\Pr[\mathsf{crs} \text{ is binding } : \mathsf{crs} \leftarrow \mathsf{Setup}(1^\lambda)] = 1 - \mathrm{negl}(\lambda)$.

*We abuse notation and write* $\mathsf{Commit}_{\mathsf{crs}}(x)$ *for a string* $x \in \{0,1\}^\ell$ *to denote the process of committing to each bit of* $x$ *separately.*

Naor's commitment scheme [Nao91] gives statistically binding commitments assuming only one-way functions. In particular, it constructs commitments from a pseudorandom generator $\mathsf{PRG} : \{0,1\}^\lambda \rightarrow \{0,1\}^{3\lambda}$, where $\mathsf{Setup}(1^\lambda)$ outputs a uniformly random $\mathsf{crs} \leftarrow \{0,1\}^{3\lambda}$ and $\mathsf{Commit}_{\mathsf{crs}}(b; r) = \mathsf{PRG}(r) \oplus (b \cdot \mathsf{crs})$. Hiding follows from PRG security and binding follows since

$$\Pr_{\mathsf{crs}}[\exists r_0, r_1 : \mathsf{PRG}(r_0) = \mathsf{PRG}(r_1) \oplus \mathsf{crs}] \leq \sum_{r_0, r_1} \Pr[\mathsf{crs} = \mathsf{PRG}(r_0) \oplus \mathsf{PRG}(r_1)] \leq 2^{2\lambda}/2^{3\lambda} \leq 2^{-\lambda}.$$

**Theorem 1 ([Nao91]).** *Assuming one-way functions, there exist statistically binding commitments.*

## 2.3 NIZKs

We define statistically sound NIZKs in the CRS model with witness indistinguishability.

**Definition 3 (Statistically Sound Non-Interactive Zero-Knowledge (NIZK)).** *A NIZK proof system for an NP relation* $R_\lambda \subseteq \{0,1\}^{n(\lambda)} \times \{0,1\}^{m(\lambda)}$ *with a corresponding NP language* $L_\lambda = \{x : \exists w \ (x, w) \in R_\lambda\}$ *consists of PPT algorithms* $(\mathsf{Setup}, \mathsf{Prove}, \mathsf{Verify})$ *with the following syntax:*

- $\mathsf{crs} \leftarrow \mathsf{Setup}(1^\lambda)$: *generates a common reference string* $\mathsf{crs}$.
- $\pi \leftarrow \mathsf{Prove}_{\mathsf{crs}}(x, w)$: *generates a proof* $\pi$ *for the statement* $x$ *using witness* $w$.
- $b = \mathsf{Verify}_{\mathsf{crs}}(x, \pi)$: *verifies the proof* $\pi$ *for a given statement* $x$ *and outputs a decision bit* $0$ *(reject) or* $1$ *(accept).*

*We require the following properties:*

- ***Completeness:*** *There exists a negligible function* $\mu$ *such that for all* $\lambda \in \mathbb{N}$, *all* $(x, w) \in R_\lambda$ *we have:*

$$\Pr\left[\mathsf{Verify}_{\mathsf{crs}}(x, \pi) = 1 : \mathsf{crs} \leftarrow \mathsf{Setup}(1^\lambda), \pi \leftarrow \mathsf{Prove}_{\mathsf{crs}}(x, w)\right] \geq 1 - \mu(\lambda).$$

- ***Statistical Soundness:*** *We say that a* $\mathsf{crs}$ *is sound if for all* $x \notin L_\lambda$ *and all* $\pi$ *we have* $\mathsf{Verify}_{\mathsf{crs}}(x, \pi) = 0$. *We require that* $\Pr[\mathsf{crs}$ *is sound* $: \mathsf{crs} \leftarrow \mathsf{Setup}(1^\lambda)] = 1 - \mathrm{negl}(\lambda)$.
- ***Witness Indistinguishability:*** *For any ensemble* $x_\lambda, w_\lambda^0, w_\lambda^1$ *such that* $(x_\lambda, w_\lambda^b) \in R_\lambda$ *for* $b \in \{0,1\}$ *we have* $(\mathsf{crs}, \pi_0) \approx_c (\mathsf{crs}, \pi_1)$ *where* $\mathsf{crs} \leftarrow \mathsf{Setup}(1^\lambda)$ *and* $\pi_b \leftarrow \mathsf{Prove}_{\mathsf{crs}}(x_\lambda, w_\lambda^b)$ *for* $b \in \{0,1\}$.

*A* NIZKs for NP *allows us to instantiate NIZK for any polynomial-time NP relation* $R_\lambda$. *We remark that the property of witness indistinguishability is weaker than an implied by the zero knowledge property typically associated with NIZKs.*

**Theorem 2 ([FLS90, CHK03, GOS06, CCH+19, PS19]).** *Statistically sound NIZKs in the CRS model exist assuming any one of: (1) hardness of factoring, or (2) the decisional linear assumption in bilinear groups, or (3) the learning with errors assumption.*

## 2.4 Witness Encryption

We define a witness encryption scheme.

**Definition 4 (Witness Encryption).** *A witness encryption scheme for an NP relation* $R_\lambda \subseteq \{0,1\}^{n(\lambda)} \times \{0,1\}^{m(\lambda)}$ *with a corresponding NP language* $L_\lambda = \{x : \exists w \ (x, w) \in R_\lambda\}$ *consists of PPT algorithms* $(\mathsf{Enc}, \mathsf{Dec})$ *with the following syntax:*

- $\mathsf{ct} \leftarrow \mathsf{Enc}(1^\lambda, x, b)$: *Encrypts a bit* $b \in \{0,1\}$ *under the NP statement* $x \in \{0,1\}^{n(\lambda)}$.
- $b = \mathsf{Dec}(\mathsf{ct}, w)$: *Decrypts the ciphertexts using a witness* $w$.

*We require the following properties:*

- ***Correctness:*** *There exists a negligible function* $\mu$ *such that for all* $\lambda \in \mathbb{N}$, *all* $(x, w) \in R_\lambda$ *we have:*

$$\Pr\left[\mathsf{Dec}(\mathsf{Enc}(1^\lambda, x, b), w) = b\right] \geq 1 - \mu(\lambda).$$

- ***Security :*** *For any ensemble* $\{x_\lambda\}_{\lambda \in \mathbb{N}}$ *such that* $x_\lambda \notin L_\lambda$ *for all* $\lambda$, *we have*

$$\mathsf{Enc}(1^\lambda, x_\lambda, 0) \approx_c \mathsf{Enc}(1^\lambda, x_\lambda, 1).$$

*A* WE for NP *allows us to instantiate WE for any polynomial-time NP relation* $R_\lambda$.

We abuse notation and write $\mathsf{Enc}(1^\lambda, x, m)$ for a long message $m \in \{0,1\}^\ell$ to denote the process of encrypting the message bit-wise $\mathsf{Enc}(1^\lambda, x, m_1), \ldots, \mathsf{Enc}(1^\lambda, x, m_\ell)$.

## 2.5 Somewhere Equivocal PRF

We define the notion of a somewhere equivocal PRF (SEPRF) from [HJO+16, Definition 7] (also refereed to as a 1-SEPRF there). Intuitively, an SEPRF consists of a pseudorandom function $y = \mathsf{PRF}(\mathsf{key}, x)$ that maps inputs $x$ to outputs $y$ using a secret $\mathsf{key}$. For any input $x^*$, there is a way to generate an equivocal key $\mathsf{eqKey}$ that leaves the output of the PRF unspecified at $x^*$, but allows us to evaluate it at all input $x \neq x^*$ by computing $\mathsf{PRF}(\mathsf{eqKey}, x)$. Later for any output $y^*$ we can fix the output of the PRF at $x^*$ to $y^*$ by generating a key $\mathsf{key}$ such that $\mathsf{PRF}(\mathsf{key}, x^*) = y^*$, while ensuring $\mathsf{PRF}(\mathsf{key}, x) = \mathsf{PRF}(\mathsf{eqKey}, x)$ for all $x \neq x^*$. Moreover, for any $x^*$, one cannot distinguish between an honestly generated $\mathsf{key}$ versus first generating $\mathsf{eqKey}$ that is equivocal at $x^*$ and later fixing the output of the PRF at $x^*$ to a uniformly random $y^*$ by generating the corresponding $\mathsf{key}$.[4]

**Definition 5 (SEPRF).** *An SEPRF with input length $n(\lambda)$ and output length $m(\lambda)$ consists of the PPT algorithms $(\mathsf{KeyGen}, \mathsf{PRF}, \mathsf{Sim}_1, \mathsf{Sim}_2)$ with the following syntax:*

- $\mathsf{key} \leftarrow \mathsf{KeyGen}(1^\lambda)$*: generates a PRF key.*
- $y = \mathsf{PRF}(\mathsf{key}, x)$*: A deterministic algorithm that takes as input $x \in \{0,1\}^{n(\lambda)}$ and outputs $y \in \{0,1\}^{m(\lambda)}$.*
- $\mathsf{eqKey} \leftarrow \mathsf{Sim}_1(1^\lambda, x^*)$*: Given $x^* \in \{0,1\}^{n(\lambda)}$ outputs a key $\mathsf{eqKey}$ that is equivocal on $x^*$.*
- $\mathsf{key} \leftarrow \mathsf{Sim}_2(\mathsf{eqKey}, y^*)$*: Given an output $y^* \in \{0,1\}^{m(\lambda)}$ creates an equivocated key $\mathsf{key}$.*

*We require two properties:*

- ***Correctness:** For all $x^* \in \{0,1\}^{n(\lambda)}$, $y^* \in \{0,1\}^{m(\lambda)}$ we have*

$$\Pr\left[\begin{array}{l} \mathsf{PRF}(\mathsf{key}, x^*) = y^* \\ \wedge \quad \forall x \neq x^* \ : \ \mathsf{PRF}(\mathsf{key}, x) = \mathsf{PRF}(\mathsf{eqKey}, x) \end{array} : \begin{array}{l} \mathsf{eqKey} \leftarrow \mathsf{Sim}_1(1^\lambda, x^*) \\ \mathsf{key} \leftarrow \mathsf{Sim}_2(\mathsf{eqKey}, y^*) \end{array}\right] = 1.$$

- ***Security:** We define the game $\mathsf{SEPRFGame}^b_{\mathcal{A}}(1^\lambda)$ between a challenger and an stateful adversary $\mathcal{A}(1^\lambda)$ as follows:*
  - *The adversary chooses $x^* \in \{0,1\}^{n(\lambda)}$.*
  - *If $b = 0$ the challenger chooses $\mathsf{key} \leftarrow \mathsf{KeyGen}(1^\lambda)$ and gives $\mathsf{key}$ to the adversary.*
    *If $b = 1$ the challenger chooses $\mathsf{eqKey} \leftarrow \mathsf{Sim}_1(1^\lambda, x^*)$, $y^* \leftarrow \{0,1\}^{m(\lambda)}$, $\mathsf{key} \leftarrow \mathsf{Sim}_2(\mathsf{eqKey}, y^*)$ and gives $\mathsf{key}$ to the adversary.*
  - *The adversary outputs a bit $b'$ which is the output of the game.*

[4] Our definition below is slightly syntactically simplified from the one in [HJO+16] since we have $\mathsf{Sim}_1$ output a single value $\mathsf{eqKey}$ while the one in [HJO+16] outputs a pair $\mathsf{key}', \mathsf{state}$ where the former is used to evaluate the PRF and the latter is used by $\mathsf{Sim}_2$. However, we can always set $\mathsf{eqKey} = (\mathsf{key}', \mathsf{state})$ and have the PRF evaluation ignore the second component to derive a scheme matching our syntax. Note that there is no requirement that $\mathsf{eqKey}$ looks like $\mathsf{key}$.

*We require that for all PPT $\mathcal{A}$ we have*

$$\left|\Pr[\mathsf{SEPRFGame}^0_{\mathcal{A}}(1^\lambda) = 1] - \Pr[\mathsf{SEPRFGame}^1_{\mathcal{A}}(1^\lambda) = 1\right| \leq \mathrm{negl}(\lambda).$$

**Theorem 3 ([HJO+16]).** *Assuming the existence of one-way functions, for any polynomials $n = n(\lambda), m = m(\lambda)$ there exists an SEPRF with input length $n$ and output length $m$.*

## 3 Functional Tag System

Our paper consists of three main components: (1) introducing a new notion of *functional tag systems* in this section, (2) constructing a functional tag system from one-way functions via garbled circuits in Sect. 4, and (3) constructing adaptively secure ABE from WE via a functional tag system in Sect. 5.

A *functional tag system* allows us to generate "input tags" $\mathsf{tag}_x$ for inputs $x$ and "function tags" $\mathsf{tag}_f$ for functions $f$. There is a dummy (D) way to generate these randomly/independently and a smart (S) way to generate these in a coordinated way using some common secret key $\mathsf{tsk}$. There is an efficient procedure that checks if some combinations of $(\mathsf{tag}_f, \mathsf{tag}_x)$ "trigger". Dummy ones trigger with negligible probability. Smart ones always trigger if $f(x) = 1$. A fully adaptive adversary who gets to see a single input tag for an input $x$ and many function tags for functions $f_i$ cannot tell the difference between dummy and smart as long as $f_i(x) = 0$ for all $i$.

**Definition 6 (Functional Tag System).** *A functional tag system for a function class $\mathcal{F}_\lambda \subseteq \{f : \{0,1\}^{n(\lambda)} \to \{0,1\}\}$ consists of PPT procedures*

$$(\mathsf{DInputTag}, \mathsf{DFunctionTag}, \mathsf{SGen}, \mathsf{SInputTag}, \mathsf{SFunctionTag}, \mathsf{Trigger})$$

*with the following syntax:*

- $\mathsf{tag}_x \leftarrow \mathsf{DInputTag}(1^\lambda, x)$ *takes as input $x \in \{0,1\}^{n(\lambda)}$ and generates a "dummy input tag".*
- $\mathsf{tag}_f \leftarrow \mathsf{DFunctionTag}(1^\lambda, f)$ *takes as input $f \in \mathcal{F}_\lambda$ and generates a "dummy function tag".*
- $\mathsf{tsk} \leftarrow \mathsf{SGen}(1^\lambda)$ *generates a tag key* $\mathsf{tsk}$.
- $\mathsf{tag}_x \leftarrow \mathsf{SInputTag}(\mathsf{tsk}, x)$ *takes as input $x \in \{0,1\}^{n(\lambda)}$ and generates a "smart input tag".*
- $\mathsf{tag}_f \leftarrow \mathsf{SFunctionTag}(\mathsf{tsk}, f)$ *takes as input $f \in \mathcal{F}_\lambda$ and generates a "smart function tag".*
- $b = \mathsf{Trigger}(\mathsf{tag}_f, \mathsf{tag}_x)$ *a deterministic procedure that outputs 0 (not triggered) or 1 (triggered).*

*The scheme has the following properties:*

1. ***Dummy Correctness:*** *There exists some negligible $\mu$ such that for all $\lambda \in \mathbb{N}$, $f \in \mathcal{F}_\lambda, x \in \{0,1\}^{n(\lambda)}$:*

$$\Pr\left[\mathsf{Trigger}(\mathsf{tag}_f, \mathsf{tag}_x) = 1 \;:\; \begin{array}{l}\mathsf{tag}_x \leftarrow \mathsf{DInputTag}(x)\\ \mathsf{tag}_f \leftarrow \mathsf{DFunctionTag}(f)\end{array}\right] \leq \mu(\lambda).$$

2. ***Smart Correctness:*** *For all* $\lambda \in \mathbb{N}$, $f \in \mathcal{F}_\lambda, x \in \{0,1\}^{n(\lambda)}$ *such that* $f(x) = 1$*:*

$$\Pr\left[\mathsf{Trigger}(\mathsf{tag}_f, \mathsf{tag}_x) = 1 \ : \ \begin{array}{l} \mathsf{tsk} \leftarrow \mathsf{SGen}(1^\lambda) \\ \mathsf{tag}_x \leftarrow \mathsf{SInputTag}(\mathsf{tsk}, x) \\ \mathsf{tag}_f \leftarrow \mathsf{SFunctionTag}(\mathsf{tsk}, f) \end{array}\right] = 1.$$

3. ***Security:*** *We define the game* $\mathsf{FunTagGame}^b_{\mathcal{A}}(1^\lambda)$ *between a challenger with a bit* $b$ *and an stateful adversary* $\mathcal{A}(1^\lambda)$ *as follows:*
   - *If* $b = 1$*, the challenger samples a random* $\mathsf{tsk} \leftarrow \mathsf{SGen}(1^\lambda)$.
   - Pre-challenge function tag queries: *The adversary can make arbitrarily many queries* $f_i \in \mathcal{F}_\lambda$*. If* $b = 0$ *the challenger responds with* $\mathsf{tag}_{f_i} \leftarrow \mathsf{DFunctionTag}(1^\lambda, f_i)$ *and if* $b = 1$ *the challenger responds with* $\mathsf{tag}_{f_i} \leftarrow \mathsf{SFunctionTag}(\mathsf{tsk}, f_i)$.
   - Challenge input tag: *The adversary chooses an input* $x \in \{0,1\}^{n(\lambda)}$ *such that* $f_i(x) = 0$ *for all prior function tag queries* $f_i$*. If* $b = 0$ *the challenger responds with* $\mathsf{tag}_x \leftarrow \mathsf{DInputTag}(1^\lambda, x)$ *and if* $b = 1$ *the challenger responds with* $\mathsf{tag}_x \leftarrow \mathsf{SInputTag}(\mathsf{tsk}, x)$.
   - Post-challenge function tag queries: *The adversary can make arbitrarily many additional queries* $f_i \in \mathcal{F}_\lambda$ *such that* $f_i(x) = 0$*. If* $b = 0$ *the challenger responds with* $\mathsf{tag}_{f_i} \leftarrow \mathsf{DFunctionTag}(1^\lambda, f_i)$ *and if* $b = 1$ *the challenger responds with* $\mathsf{tag}_{f_i} \leftarrow \mathsf{SFunctionTag}(\mathsf{tsk}, f_i)$.
   - *The adversary output a bit* $b'$ *which is the output of the game.*

   *We require that for all PPT* $\mathcal{A}$ *we have*

$$\left|\Pr[\mathsf{FunTagGame}^0_{\mathcal{A}}(1^\lambda) = 1] - \Pr[\mathsf{FunTagGame}^1_{\mathcal{A}}(1^\lambda) = 1]\right| \leq \operatorname{negl}(\lambda).$$

*A* functional tag system for circuits *allows us to instantiate a functional tag system for the class* $\mathcal{C}^{s,n}_\lambda$ *consisting of boolean circuits of size* $s(\lambda)$ *with* $n(\lambda)$*-bit input, for any polynomials* $s(\lambda), n(\lambda)$.

## 4 A Functional Tag System from One-Way Functions

We construct a functional tag system for circuits from one-way functions. Our main tool is a special form of blind garbled circuits. We first define and construct this form of blind garbled circuits and then proceed to use them to construct a functional tag system.

### 4.1 Blind Garbled Circuits

We rely on blind garbled circuits, originally defined in [BLSV18]. In a blind garbled circuits, if one gets a garbled input together with a garbled circuit that outputs a uniformly random value on that input, the pair looks like completely random bits. We rely on a variant of blind garbled circuit security that we call *semi-adaptive blind garbled circuits*, defined formally below. Informally, we consider a game where the adversary can get arbitrarily many garbled circuits and a single garbled input $\widetilde{x}$ of a value $x$, all chosen adaptively. In addition the adversary chooses a challenge circuit $C$ *after* it gets the garbled input $\widetilde{x}$ and should not be able to distinguish between a real garbling $\widetilde{C}$ of the challenge circuit $C$ versus a simulated one. Note that the input $x$ cannot depend on the garbled circuit $\widetilde{C}$, which avoids the main difficulty in adaptively secure garbled circuits. The simulator needs to simulate the garbled circuit $\widetilde{C}$ given $x, C(x)$, but without knowing the circuit $C$. For blindness, we require that, for a uniformly random output $C(x)$, the corresponding simulated garbled circuit $\widetilde{C}$ is uniformly random. While the definition is incomparable to the one in [BLSV18], we show that the same "point-and-permute" construction used in [BLSV18] satisfies our definition as well.

**Definition 7 (Semi-adaptive Blind Garbled Circuit).** *Let $\mathcal{C}_\lambda^{s,n,m}$ be a class of circuits of size $s = s(\lambda)$ with $n = n(\lambda)$-bit input and $m = m(\lambda)$-bit output. A* semi-adaptive blind garbled circuit *scheme for $\mathcal{C}_\lambda^{s,n,m}$ consist of PPT algorithms:* $(\mathsf{GarbleGen}, \mathsf{GInput}, \mathsf{GCircuit}, \mathsf{SimCircuit}, \mathsf{Eval})$ *and garbled circuit size parameter $\ell = \ell(\lambda)$ with the following syntax.*

- $\mathsf{sk} \leftarrow \mathsf{GarbleGen}(1^\lambda)$*: generates a garbling secret key* $\mathsf{sk}$.
- $\widetilde{x} \leftarrow \mathsf{GInput}(\mathsf{sk}, x)$*: garbles an input $x \in \{0,1\}^n$.*
- $\widetilde{C} \leftarrow \mathsf{GCircuit}(\mathsf{sk}, C)$*: garbles a circuit $C \in \mathcal{C}_\lambda^{s,n,m}$ with $\widetilde{C} \in \{0,1\}^\ell$.*
- $y := \mathsf{Eval}(\widetilde{C}, \widetilde{x})$*: a deterministic algorithm that evaluates the garbled circuit on the garbled input and yields output $y \in \{0,1\}^m$.*
- $\widetilde{C} \leftarrow \mathsf{SimCircuit}(\mathsf{sk}, x, y)$*: produces a simulated circuit $\widetilde{C} \in \{0,1\}^\ell$ for a given output $y = C(x)$ without knowing $C$.*

*We require the following properties:*

**Correctness:** *For all $\lambda, C \in \mathcal{C}_\lambda^{s,n,m}$ $x \in \{0,1\}^n$ we have*

$$\Pr[\mathsf{Eval}(\widetilde{C}, \widetilde{x}) = C(x) \ : \ \mathsf{sk} \leftarrow \mathsf{GarbleGen}(1^\lambda), \widetilde{x} \leftarrow \mathsf{GInput}(\mathsf{sk}, x), \widetilde{C} \leftarrow \mathsf{GCircuit}(\mathsf{sk}, C)] = 1.$$

**Semi-adaptive Simulation Security:** *We define the game $\mathsf{GCGame}_\mathcal{A}^b(1^\lambda)$ between a challenger with a bit $b$ and an stateful adversary $\mathcal{A}(1^\lambda)$ as follows:*

- *Challenger picks* $\mathsf{sk} \leftarrow \mathsf{GarbleGen}(1^\lambda)$.
- *Adversary $\mathcal{A}^{\mathsf{GCircuit}(\mathsf{sk},\cdot)}$ picks $x \in \{0,1\}^n$ and gets back* $\widetilde{x} \leftarrow \mathsf{GInput}(\mathsf{sk}, x)$.
- *Adversary $\mathcal{A}^{\mathsf{GCircuit}(\mathsf{sk},\cdot)}$ picks a boolean circuit $C \in \mathcal{C}_\lambda^{n,m}$. The adversary gets back either* $\widetilde{C} \leftarrow \mathsf{GCircuit}(\mathsf{sk}, C)$ *if $b = 0$ or* $\widetilde{C} \leftarrow \mathsf{SimCircuit}(\mathsf{sk}, x, C(x))$ *if $b = 1$.*
- *Adversary $\mathcal{A}^{\mathsf{GCircuit}(\mathsf{sk},\cdot)}$ outputs a bit $b'$ which is the output of the game.*

*We require that for all PPT $\mathcal{A}$ we have*

$$\left|\Pr[\mathsf{GCGame}_\mathcal{A}^0(1^\lambda) = 1] - \Pr[\mathsf{GCGame}_\mathcal{A}^1(1^\lambda) = 1]\right| \leq \text{negl}(\lambda).$$

**Blindness:** *For every fixed choice of* $\mathsf{sk}$ *in the support of* $\mathsf{GarbleGen}(1^\lambda)$ *every $x \in \{0,1\}^n$ we have:*

$$\mathsf{SimCircuit}(\mathsf{sk}, x, U_m) \equiv U_\ell,$$

*where $U_i$ denotes the uniform distribution over $\{0,1\}^i$ and "$\equiv$" denotes distributional equivalence.*

*Construction.* Let $s(\lambda), n(\lambda), m(\lambda)$ be arbitrary polynomials. We assume that circuits in $\mathcal{C}_\lambda^{s,n,m}$ *have some fixed topology.* In particular, each circuit $C \in \mathcal{C}_\lambda^{s,n,m}$ consists of $s$ gates and $s+n+m$ wires, with $n$ input wires denoted $\mathsf{in}_1, \ldots \mathsf{in}_n$, $m$ output wires denoted $\mathsf{out}_1, \ldots, \mathsf{out}_m$ and $s$ internal wires. Each gate $g \in [s]$ gets 2 input wires and 1 output wire; we allow arbitrary fan-out since each output wire can be an input to arbitrarily many other gates. Each gate $g$ computes some function $f_g : \{0,1\}^2 \to \{0,1\}$. The gates are connected via some fixed topology that is the same for all circuits in the class: that is, any gate $g \in [s]$ has some fixed input writes $w_{g,1}, w_{g,2}$ and output wire $w_{g,w}$ for all $C \in \mathcal{C}_\lambda^{s,n,m}$. The only distinction between different circuits $C \in \mathcal{C}_\lambda^{s,n,m}$ are the functions $f_g$ computed by each gate. Note that we can convert general circuits into ones having a fixed topology with only a polylogarithmic blowup in circuit size via *universal circuits*, and therefore the above assumption is without loss of generality. Let $\mathsf{PRF} : \{0,1\}^\lambda \times \{0,1\}^* \to \{0,1\}^{\lambda+1}$ be a pseudorandom function. The "point-and-permute" construction of blind garbled circuits for the class $\mathcal{C}_\lambda^{s,n,m}$ works as follows:

- $\mathsf{sk} \leftarrow \mathsf{GarbleGen}(1^\lambda)$: For each of the $n$ input wires $\mathsf{in}_i$, sample PRF keys $s_{\mathsf{in}_i,b} \leftarrow \{0,1\}^\lambda$ and random bits $\alpha_{\mathsf{in}_i} \leftarrow \{0,1\}$ for $i \in [n], b \in \{0,1\}$. Let $\mathsf{sk} = (s_{\mathsf{in}_i,b}, \alpha_{\mathsf{in}_i})_{i\in[n],b\in\{0,1\}}$.
- $\widetilde{x} \leftarrow \mathsf{GInput}(\mathsf{sk}, x)$: For $x = (x_1, \ldots, x_n) \in \{0,1\}^n$ output $\widetilde{x} = (s_{\mathsf{in}_i,x_i}, \alpha_{\mathsf{in}_i} \oplus x_i)_{i\in[n]}$.
- $\widetilde{C} \leftarrow \mathsf{GCircuit}(\mathsf{sk}, C)$: Sample a random circuit nonce $c \leftarrow \{0,1\}^\lambda$. For each wire $w$ that is not an input wire sample fresh PRF keys $s_{w,b} \leftarrow \{0,1\}^\lambda$ for $b \in \{0,1\}$ along with a random bit $\alpha_w \leftarrow \{0,1\}$. The corresponding values $s_{\mathsf{in}_i,b}, \alpha_{\mathsf{in}_i}$ for the input wires $\mathsf{in}_i$ are contained in $\mathsf{sk}$. For each gate $g$, let $f_g : \{0,1\}^2 \to \{0,1\}$ be the Boolean function computed by the gate. For every gate $g \in [s]$ with input wires $w_1, w_2$ and output wire $w_3$, and for all $\beta_1, \beta_2 \in \{0,1\}$, set $\beta_3 := f_g(\alpha_{w_1} \oplus \beta_1, \alpha_{w_2} \oplus \beta_2) \oplus \alpha_{w_3}$, and compute the table entry:

$$T_g^{\beta_1,\beta_2} := (s_{w_3,\alpha_{w_3}\oplus\beta_3} \parallel \beta_3) \oplus \mathsf{PRF}(s_{w_1,\alpha_{w_1}\oplus\beta_1}, c \parallel g \parallel \beta_1 \parallel \beta_2) \oplus \mathsf{PRF}(s_{w_2,\alpha_{w_2}\oplus\beta_2}, c \parallel g \parallel \beta_1 \parallel \beta_2). \quad (1)$$

  Define the table for gate $g$ as $T_g := (T_g^{\beta_1,\beta_2})_{\beta_1,\beta_2\in\{0,1\}}$. Then, output the garbled circuit consisting of:

$$\widetilde{C} = \left(c \ , \ (T_g)_{g\in[s]} \ , \ (\alpha_{\mathsf{out}_j})_{j\in[m]}\right),$$

  with $\widetilde{C} \in \{0,1\}^\ell$ for $\ell = \lambda + 4(\lambda+1)s + m$.
- $y := \mathsf{Eval}(\widetilde{C}, \widetilde{x})$: Parse $\widetilde{x} = (s_{\mathsf{in}_i}, \beta_{\mathsf{in}_i})_{i\in[n]}$. For every gate $g \in [s]$ in topological order, let $w_1, w_2$ be its input wires and let $w_3$ be its output wire. Then, given $(s_{w_1} \parallel \beta_{w_1}), (s_{w_2} \parallel \beta_{w_2})$, compute

$$(s_{w_3} \parallel \beta_{w_3}) = T_g^{\beta_{w_1},\beta_{w_2}} \oplus \mathsf{PRF}(s_{w_1}, c \parallel g \parallel \beta_{w_1} \parallel \beta_{w_2}) \oplus \mathsf{PRF}(s_{w_2}, c \parallel g \parallel \beta_{w_1} \parallel \beta_{w_2}).$$

  Finally, upon obtaining $\beta_{\mathsf{out}_j}$, set $y_j := \beta_{\mathsf{out}_j} \oplus \alpha_{\mathsf{out}_j}$ for $j \in [m]$ and output $y := (y_1, \ldots, y_m) \in \{0,1\}^m$.

- $\widetilde{C} \leftarrow \mathsf{SimCircuit}(\mathsf{sk}, x, y)$: Sample a random circuit nonce $c \leftarrow \{0,1\}^\lambda$. For each wire $w$ that is not an input wire sample a fresh PRF key $s_w \leftarrow \{0,1\}^\lambda$ along with a random bit $\beta_w \leftarrow \{0,1\}$. For each input wire $\mathsf{in}_i$, set $s_{\mathsf{in}_i} := s_{\mathsf{in}_i, x_i}$ $\beta_{\mathsf{in}_i} := x_i \oplus \alpha_{\mathsf{in}_i}$ using the values from $\mathsf{sk}$. For each gate $g$ with input wires $w_1, w_2$ and output wire $w_3$, compute

$$T_g^{\beta_{w_1}, \beta_{w_2}} := (s_{w_3} \parallel \beta_{w_3}) \oplus \mathsf{PRF}(s_{w_1}, c \parallel g \parallel \beta_{w_1} \parallel \beta_{w_2}) \oplus \mathsf{PRF}(s_{w_2}, c \parallel g \parallel \beta_{w_1} \parallel \beta_{w_2}), \tag{2}$$

and choose $T_g^{\beta_0, \beta_1} \leftarrow \{0,1\}^{\lambda+1}$ uniformly at random for all $(\beta_0, \beta_1) \neq (\beta_{w_1}, \beta_{w_2})$. Define the table for gate $g$ as $T_g := (T_g^{\beta_1, \beta_2})_{\beta_1, \beta_2 \in \{0,1\}}$. Output

$$\widetilde{C} = \big(c \ \ , \ \ (T_g)_{g \in [s]} \ \ , \ \ (\beta_{\mathsf{out}_j} \oplus y_j)_{j \in [m]}\big) .$$

**Theorem 4.** *Assuming one-way functions, there exist semi-adaptive blind garbled circuits for the class $\mathcal{C}_\lambda^{s,n,m}$ for any polynomials $s, n, m$.*

*Proof.* We show that the "point-and-permute" construction from pseudorandom functions described above is a semi-adaptive garbled circuit. The theorem then follows using the fact that pseudorandom functions can be constructed from one-way functions.

*Perfect Correctness.* For the perfect correctness of the construction, note that during evaluation it holds that each computed value $(s_w, \beta_w) = (s_{w,\mathsf{val}(w)}, \mathsf{val}(w) \oplus \alpha_w)$, where $\mathsf{val}(w)$ is the value on the wire $w$ during the computation $C(x)$, and $s_{w,b}, \alpha_w$ are the values chosen during garbling. This is true for the input wires, and is easily seen to be true for all subsequent wires by induction. Therefore it holds that for the output wires $\beta_{\mathsf{out}_j} = y_j \oplus \alpha_{\mathsf{out}_j}$ and therefore evaluation computes the correct outputs $y_j$.

*Semi-Adaptive Simulation Security.* To prove semi-adaptive simulation security, we do a sequence of hybrids where we change the challenge garbled circuit $\widetilde{C}$ from real to simulated. Firstly, we define the games $\widehat{\mathsf{GCGame}}^b$ identically to $\mathsf{GCGame}^b$, except that the game outputs 0 if the circuit nonce $c$ used in the challenge garbled circuit $\widetilde{C}$ is ever used in any other garbled circuit created by the $\mathsf{GCircuit}(\mathsf{sk}, \cdot)$ oracle. For $b \in \{0,1\}$, the games $\mathsf{GCGame}^b$ and $\widehat{\mathsf{GCGame}}^b$ are statistically indistinguishable. Therefore it suffices to show that $\widehat{\mathsf{GCGame}}^0$ and $\widehat{\mathsf{GCGame}}^1$ are computationally indistinguishable.

To do so, we iterate over all gates $g \in [s]$ in topological order starting with the input layer. For $i \in [s+1]$, define hybrids $\mathsf{Game}_i$ where the challenge garbled circuit

$$\widetilde{C} = \big(c \ \ , \ \ (T_g)_{g \in [s]} \ \ , \ \ (\alpha_{\mathsf{out}_j})_{j \in [m]}\big)$$

is sampled as follows. We sample the values $c, s_{w,b}, \alpha_w$ as specified by $\mathsf{GCircuit}$. Define $s_w := s_{w,\mathsf{val}(w)}, \beta_w := \alpha_w \oplus \mathsf{val}(w)$ where $\mathsf{val}(w)$ is the value on the wire $w$ during the computation $C(x)$, which is well defined since the input $x$ is chosen before the challenge circuit $\widetilde{C}$ is created. For the gates $g \geq i$ the tables

$T_g := (T_g^{\beta_1,\beta_2})_{\beta_1,\beta_2\in\{0,1\}}$ are created as in $\mathsf{GCircuit}$ following Eq. 1. For gates $g < i$, the tables $T_g := (T_g^{\beta_1,\beta_2})_{\beta_1,\beta_2\in\{0,1\}}$ are instead created as in $\mathsf{SimCircuit}$; namely if the gate $g$ has input wires $w_1, w_2$ and output wire $w_3$, then the table entry $T_g^{\beta_{w_1},\beta_{w_2}}$ is created as in Eq. 2 and the other entries are sampled randomly with $T_g^{\beta_0,\beta_1} \leftarrow \{0,1\}^{\lambda+1}$ for all $(\beta_0,\beta_1) \neq (\beta_{w_1},\beta_{w_2})$. It is easy to see that $\mathsf{Game}_1$ is identical to $\widehat{\mathsf{GCGame}}^0$. Furthermore, for all $g \in [s]$, $\mathsf{Game}_g$ is computationally indistinguishable from $\mathsf{Game}_{g+1}$. The only difference between the games is how the entries $T_g^{\beta_0,\beta_1}$ for $(\beta_0,\beta_1) \neq (\beta_{w_1},\beta_{w_2})$ are sampled. However, it is easy to show that the games are indistinguishable by PRF security. In particular, for these entries, at least one of the two PRF outputs in Eq. 1 involves a PRF key $s_{w,b}$ that is not used in the game in any other way beyond black-box PRF evaluation $\mathsf{PRF}(s_{w,b},\cdot)$ and the input $c \parallel g \parallel \beta_1 \parallel \beta_2$ on which the PRF is evaluated is not used anywhere else. Therefore, we can replace this PRF output by uniform. Lastly, we observe that $\mathsf{Game}_{s+1}$ is identical to $\widehat{\mathsf{GCGame}}^1$. This simply follows since, for each non-input wire, the values $s_w := s_{w,\mathsf{val}(w)}, \beta_w := \alpha_w \oplus \mathsf{val}(w)$ are uniformly random over the choice of $s_{w,0}, s_{w,1}, \alpha_w$ and for the output wires we have $\alpha_{\mathsf{out}_j} = \beta_{\mathsf{out}_j} \oplus \mathsf{val}(\mathsf{out}_j) = \beta_{\mathsf{out}_j} \oplus y_j$.

*Blindness.* Finally, to show blindness, we need to show that the distribution of the simulated garbled circuit

$$\widetilde{C} = \left(c \quad , \quad (T_g)_{g\in[s]} \quad , \quad (\beta_{\mathsf{out}_j} \oplus y_j)_{j\in[m]}\right) \leftarrow \mathsf{SimCircuit}(\mathsf{sk}, x, U_m)$$

satisfies $\widetilde{C} \equiv \{0,1\}^\ell$ for $\ell = \lambda + 4(\lambda+1)s + m$. First, $\widetilde{C} \equiv \left(c \quad , \quad (T_g)_{g\in[s]} \quad , \quad U_m\right)$ since the $y \leftarrow U_m$ is uniformly random and independent of $c, (T_g)_{g\in[s]}$ or $\{\beta_w\}$. Second, we proceed in reverse topological order starting at the output layer and show that each table $T_g$ is uniformly random over $\{0,1\}^{4(\lambda+1)}$ even given $c$ and all the tables $T_i$ for $i < g$. This follows from Eq. 2 and the fact that $(s_{w_3} \parallel \beta_{w_3})$ is uniformly random and is not used in the construction of tables $T_i$ for $i < g$. Therefore $\widetilde{C} \equiv \left(c \quad , \quad (T_g)_{g\in[s]} \quad , \quad U_m\right) \equiv (c, U_{4(\lambda+1)s}, U_m) \equiv U_\ell$.

## 4.2 Functional Tag System from Blind Garbled Circuits

We construct a functional tag system (Definition 6) from one-way functions using blind garbled circuits (Definition 7) and a somewhere equivocal PRF (Definition 5). As a starting point of our construction, we set dummy input tags to be garbled inputs $\mathsf{tag}_x = \widetilde{x}$ using a fresh garbling key $\mathsf{sk}$, and dummy function tags $\mathsf{tag}_f = (\widetilde{C}, t)$ consist of a uniformly random garbled circuit $\widetilde{C} \leftarrow \{0,1\}^\ell$ along with some target value $t$. An input/function tag "triggers" if evaluating the garbled circuit $\widetilde{C}$ on the garbled input $\widetilde{x}$ produces the target value $t$. In the dummy case, this only happens with negligible probability, ensuring "dummy correctness". A smart input tag is a garbled inputs $\mathsf{tag}_x = \widetilde{x}$ using a garbling key $\mathsf{sk}$ contained in the tag key $\mathsf{tsk} = \mathsf{sk}$, and a smart function tag $\mathsf{tag}_f = (\widetilde{C}, t)$ consists of a random target value $t$ along with a correctly garbled circuit $\widetilde{C} \leftarrow \mathsf{GCircuit}(\mathsf{tsk}, C)$ of the circuit $C$ that evaluates $f(x)$ and if the output is 1 it outputs the target value $t$ else it outputs a random independent value $u$. This ensures that a smart input/function tag pair $\mathsf{tag}_x, \mathsf{tag}_f$ does trigger when $f(x) = 1$. For security, we intuitively want to rely on blind garbled circuits to ensure that we can replace dummy function tags with smart ones in the case where $f(x) = 0$, by relying on the fact that the circuit $C(x)$ outputs a random independent value $u$ in this case. However, there is an issue with adaptivity. Blind garbled circuits only provide *semi-adaptive* security, where the challenge circuit $C$ must be chosen after the input $x$, while functional tag systems require *fully adaptive* security where the challenge functions $f$ can be chosen before or after the input $x$. We resolve this issue by encrypting the garbled circuit with a somewhere equivocal PRF whose key is part of the input tag. For any circuit $C$ chosen before the input $x$, this allows us to give a fake ciphertext inside $\mathsf{tag}_f$ and only later equivocate the garbled circuit $\widetilde{C}$ inside the ciphertext after the input $x$ is chosen, in affect allowing $\widetilde{C}$ to depend on $x$ inside the security proof. Therefore, we can rely on semi-adaptive security of the blind garbled circuits to achieve fully adaptive security of the functional tag system.

*Construction.* Let $n = n(\lambda), s = s(\lambda)$ be any polynomials. We construct a functional tag system for the class $\mathcal{F}_\lambda = \mathcal{C}_\lambda^{s,n}$ consisting of circuits of size $s$ with $n$-bit input and 1-bit output. Let $(\mathsf{GarbleGen}, \mathsf{GInput}, \mathsf{GCircuit}, \mathsf{Eval})$ be a semi-adaptive blind garbled circuit for the class $\mathcal{C}_\lambda^{s',n,m=\mathrm{sec}}$, where $s' = s + O(\lambda)$ will be defined later, and let $\ell = \ell(\lambda)$ be the corresponding garbled circuit size. Let $(\mathsf{KeyGen}, \mathsf{PRF}, \mathsf{Sim}_1, \mathsf{Sim}_2)$ be a somewhere equivocal PRF with input length $\lambda$ and output length $\ell$. We construct a functional tag system $(\mathsf{DInputTag}, \mathsf{DFunctionTag}, \mathsf{SGen}, \mathsf{SInputTag}, \mathsf{SFunctionTag}, \mathsf{Trigger})$ defined as follows:

– $\mathsf{tag}_x \leftarrow \mathsf{DInputTag}(x)$: Choose $\mathsf{sk} \leftarrow \mathsf{GarbleGen}(1^\lambda)$, $\mathsf{key} \leftarrow \mathsf{KeyGen}(1^\lambda)$, $\widetilde{x} \leftarrow \mathsf{GInput}(\mathsf{sk}, x)$. Output $\mathsf{tag}_x = (\mathsf{key}, \widetilde{x})$.
– $\mathsf{tag}_f \leftarrow \mathsf{DFunctionTag}(f)$: Output $\mathsf{tag}_f = (t_0, t_1, t_2) \leftarrow \{0,1\}^\lambda \times \{0,1\}^\ell \times \{0,1\}^\lambda$.
– $\mathsf{tsk} \leftarrow \mathsf{SGen}(1^\lambda)$: Choose $\mathsf{sk} \leftarrow \mathsf{GarbleGen}(1^\lambda)$, $\mathsf{key} \leftarrow \mathsf{KeyGen}(1^\lambda)$ and set $\mathsf{tsk} = (\mathsf{sk}, \mathsf{key})$.
– $\mathsf{tag}_x \leftarrow \mathsf{SInputTag}(\mathsf{tsk}, x)$: Choose $\widetilde{x} \leftarrow \mathsf{GInput}(\mathsf{sk}, x)$. Output $\mathsf{tag}_x = (\mathsf{key}, \widetilde{x})$.
– $\mathsf{tag}_f \leftarrow \mathsf{SFunctionTag}(\mathsf{tsk}, f)$: Choose $t_0, t_2, u \leftarrow \{0,1\}^\lambda$ and let $C^*_{f,u,t_2}$ be the circuit that on input $x$ outputs $u$ if $f(x) = 0$ and outputs $t_2$ if $f(x) = 1$. We define the parameter $s' = s + O(\lambda)$ be the size of $C^*_{f,u,t_2}$ for $f \in \mathcal{C}^{s,n}$. Let $\widetilde{C} \leftarrow \mathsf{GCircuit}(\mathsf{sk}, C^*_{f,u,t_2})$ and set $t_1 = \mathsf{PRF}(\mathsf{key}, t_0) \oplus \widetilde{C}$. The output is $\mathsf{tag}_f = (t_0, t_1, t_2)$.
– $\mathsf{Trigger}(\mathsf{tag}_f, \mathsf{tag}_x)$: Parse $\mathsf{tag}_x = (\mathsf{key}, \widetilde{x}), \mathsf{tag}_f = (t_0, t_1, t_2)$. Let $\widetilde{C} := \mathsf{PRF}(\mathsf{key}, t_0) \oplus t_1$. Output 1 iff $\mathsf{Eval}(\widetilde{C}, \widetilde{x}) = t_2$.

**Theorem 5.** *Assuming one-way functions, for any polynomials $n = n(\lambda), s = s(\lambda)$, there exists a functional tag system for the class $\mathcal{F}_\lambda = \mathcal{C}_\lambda^{s,n}$.*

*Proof.* We start by proving **dummy correctness**. Let $f, x$ be arbitrary and let $\mathsf{tag}_x \leftarrow \mathsf{DInputTag}(x), \mathsf{tag}_f \leftarrow \mathsf{DFunctionTag}(f)$ with $\mathsf{tag}_x = (\mathsf{key}, \widetilde{x}), \mathsf{tag}_f = (t_0, t_1, t_2)$ and let $\widetilde{C} = \mathsf{PRF}(\mathsf{key}, t_0) \oplus t_1$. Since $t_2$ is uniformly random and independent of $\widetilde{C}, \widetilde{x}$, we have:

$$\Pr[\mathsf{Trigger}(\mathsf{tag}_f, \mathsf{tag}_x) = 1] = \Pr[\mathsf{Eval}(\widetilde{C}, \widetilde{x}) = t_2] = 2^{-\lambda}.$$

Next we prove **smart correctness**. Let $f, x$ be arbitrary such that $f(x) = 1$ and let $\mathsf{tsk} \leftarrow \mathsf{SGen}(1^\lambda)$, $\mathsf{tag}_x \leftarrow \mathsf{SInputTag}(\mathsf{tsk}, x), \mathsf{tag}_f \leftarrow \mathsf{SFunctionTag}(\mathsf{tsk}, f)$ with with $\mathsf{tag}_x = (\mathsf{key}, \widetilde{x}), \mathsf{tag}_f = (t_0, t_1 = \mathsf{PRF}(\mathsf{key}, t_0) \oplus \widetilde{C}, t_2)$. Then

$$\Pr[\mathsf{Trigger}(\mathsf{tag}_f, \mathsf{tag}_x) = 1] = \Pr[\mathsf{Eval}(\widetilde{C}, \widetilde{x}) = t_2] = 1$$

by the perfect correctness of garbled circuits.

Lastly, we prove the **security** of the functional tag system via a sequence of hybrid games where we change how the challenger generates input and function tags.

- $\mathsf{Game}_0$: This is the game $\mathsf{FunTagGame}^0$ which outputs $\mathcal{A}^{\mathsf{DInputTag}(1^\lambda,\cdot),\mathsf{DFunctionTag}(1^\lambda,\cdot)}(1^\lambda)$, where the adversary $\mathcal{A}$ has the restrictions that: (1) it makes a single challenge input tag query $x$ to the oracle $\mathsf{DInputTag}(1^\lambda, \cdot)$ and (2) all the queries $f_i$ made to the $\mathsf{DFunctionTag}(1^\lambda, \cdot)$ oracle (both pre-challenge and post-challenge) satisfy $f_i(x) = 0$.
- $\mathsf{Game}_1$: In this game, if the oracle $\mathsf{DFunctionTag}(1^\lambda, \cdot)$ ever samples a value $t_0$ that was already used in the response to a previous query, we define the output of the game to be 0.

  *It is easy to see that* $\mathsf{Game}_0$ *and* $\mathsf{Game}_1$ *are statistically indistinguishable since* $t_0 \leftarrow \{0,1\}^\lambda$ *is chosen randomly each time.*
- $\mathsf{Game}_2$: In this game, we choose $\mathsf{tsk} \leftarrow \mathsf{SGen}(1^\lambda)$ at the very beginning of the game and change the first oracle from $\mathsf{DInputTag}(1^\lambda, \cdot)$ to $\mathsf{SInputTag}(\mathsf{tsk}, \cdot)$.

  $\mathsf{Game}_1$ *and* $\mathsf{Game}_2$ *are identically distributed by the definition of* $\mathsf{DInputTag}, \mathsf{SInputTag}$ *and the fact that* $\mathsf{tsk}$ *is never used anywhere else.*
- $\mathsf{Game}_3$: For all *pre-challenge function-tag queries*, switch to answering them using $\mathsf{SFunctionTag}(\mathsf{tsk}, \cdot)$ instead of $\mathsf{DFunctionTag}(\cdot)$. Indistinguishability follows via a sequence of internal hybrids $\mathsf{Game}^i_{2\to3}$ where the first $i$ pre-challenge function-tag queries are answered using $\mathsf{SFunctionTag}(\mathsf{tsk}, \cdot)$ and the rest are answered using $\mathsf{DFunctionTag}(\cdot)$. Note that if the adversary makes $q$ such queries then $\mathsf{Game}^0_{2\to3}$ is identical to $\mathsf{Game}_2$ and $\mathsf{Game}^q_{2\to3}$ is identical to $\mathsf{Game}_3$. To switch from $\mathsf{Game}^i_{2\to3}$ to $\mathsf{Game}^{i+1}_{2\to3}$ we introduce further sub-hybrids as follows:
  1. $\mathsf{Game}^{i.1}_{2\to3}$: At the very beginning of the game, when choosing $\mathsf{tsk} = (\mathsf{sk}, \mathsf{key})$ change from choosing the PRF key as $\mathsf{key} \leftarrow \mathsf{KeyGen}(1^\lambda)$ to choosing

     $$t_0 \leftarrow \{0,1\}^\lambda, \quad \mathsf{eqKey} \leftarrow \mathsf{Sim}_1(1^\lambda, t_0), \quad r^* \leftarrow \{0,1\}^\ell, \quad \mathsf{key} \leftarrow \mathsf{Sim}_2(\mathsf{eqKey}, r^*).$$

     Then use the value $t_0$ to generate the $i$'th function-tag query $(t_0, t_1, t_2)$. $\mathsf{Game}^i_{2\to3}$ *is computationally indistinguishable from* $\mathsf{Game}^{i.1}_{2\to3}$ *by SEPRF security.*

2. $\mathsf{Game}^{i.2}_{2\to 3}$: Choose $t_0, \mathsf{eqKey}$ at the beginning of the game as before, but do not choose $r^*, \mathsf{key}$ yet. For all function-tag queries before the $i$'th one, use $\mathsf{eqKey}$ instead of $\mathsf{key}$ to answer the query. When answering the $i$'th pre-challenge function-tag query $(t_0, t_1, t_2)$, use the value $t_0$ sampled previously. When answering the challenge input-tag query later, choose $\widetilde{C} \leftarrow \{0,1\}^\ell$, $r^* \leftarrow t_1 \oplus \widetilde{C}$, $\mathsf{key} \leftarrow \mathsf{Sim}_2(\mathsf{eqKey}, r^*)$.
   $\mathsf{Game}^{i.1}_{2\to 3}$ *is identically distributed to* $\mathsf{Game}^{i.2}_{2\to 3}$ *by the correctness of the SEPRF which says that* $\mathsf{PRF}(\mathsf{eqKey}, t_0') = \mathsf{PRF}(\mathsf{key}, t_0')$ *for all* $t_0' \neq t_0$, *and if* $t_0' = t_0$ *is ever chosen before the i'th query then the game outputs* 0 *in either case. Note that* $r^*$ *is still uniform and independent of* $t_1$ *so defining* $r^* = t_1 \oplus \widetilde{C}$ *is the same as* $r^* \leftarrow \{0,1\}^\ell$.
3. $\mathsf{Game}^{i.3}_{2\to 3}$: When answering the challenge input-tag query, instead of choosing $\widetilde{C} \leftarrow \{0,1\}^\ell$, we now choose $u \leftarrow \{0,1\}^\lambda$, $\widetilde{C} \leftarrow \mathsf{SimCircuit}(\mathsf{sk}, x, u)$.
   $\mathsf{Game}^{i.2}_{2\to 3}$ *is identically distributed to* $\mathsf{Game}^{i.3}_{2\to 3}$ *by the blindness property of blind garbled circuits and the fact that* $u \leftarrow \{0,1\}^\lambda$ *is chosen randomly.*
4. $\mathsf{Game}^{i.4}_{2\to 3}$: When answering the challenge input-tag query, instead of choosing $\widetilde{C} \leftarrow \mathsf{SimCircuit}(\mathsf{sk}, x, u)$, we now choose $\widetilde{C} \leftarrow \mathsf{GCircuit}(\mathsf{sk}, C^*_{f_i,u,t_2})$ where $f_i$ is the function chosen in the $i$'th function-tag query.
   $\mathsf{Game}^{i.3}_{2\to 3}$ *is computationally indistinguishable from* $\mathsf{Game}^{i.4}_{2\to 3}$ *by the semi-adaptive simulation security of the garbled circuit. The reduction does not know the garbling key* $\mathsf{sk}$ *but is responsible for incorporating the equivocal PRF. It uses its oracle to* $\mathsf{GCircuit}(\mathsf{sk}, \cdot)$ *to answer all calls to* $\mathsf{SFunctionTag}(\mathsf{tsk}, \cdot)$. *During the challenge input-tag query for input* $x$, *the reduction hands* $x$ *to its challenger to get* $\widetilde{x}$. *It then hands the challenger the circuit* $C^*_{f_i,u,t_2}$ *and gets* $\widetilde{C}$. *It uses the values* $\widetilde{x}, \widetilde{C}$ *to correctly answer the challenge input-tag query. If* $\widetilde{C} \leftarrow \mathsf{SimCircuit}(\mathsf{sk}, x, u)$ *then the game is identical to* $\mathsf{Game}^{i.3}_{2\to 3}$ *and if* $\widetilde{C} \leftarrow \mathsf{GCircuit}(\mathsf{sk}, C^*_{f_i,u,t_2})$ *then the game is identical to* $\mathsf{Game}^{i.4}_{2\to 3}$. *Here we rely on the fact that* $f_i(x) = 0$ *to ensure that* $C^*_{f_i,u,t_2}(x) = u$.
5. $\mathsf{Game}^{i.5}_{2\to 3}$: Instead of choosing $t_1 \leftarrow \{0,1\}^\lambda$ in the $i$'th function-tag query and then waiting to choose $\widetilde{C} \leftarrow \mathsf{GCircuit}(\mathsf{sk}, C^*_{f_i,u,t_2})$, $r^* \leftarrow t_1 \oplus \widetilde{C}$, $\mathsf{key} \leftarrow \mathsf{Sim}_2(\mathsf{eqKey}, r^*)$ in the input-tag query, we now choose $r^* \leftarrow \{0,1\}^\ell$, $\mathsf{key} \leftarrow \mathsf{Sim}_2(\mathsf{eqKey}, r^*)$ at the very beginning of the game and then during the $i$'th function-tag query choose $\widetilde{C} \leftarrow \mathsf{GCircuit}(\mathsf{sk}, C^*_{f_i,u,t_2})$ and set $t_1 = \widetilde{C} \oplus \mathsf{PRF}(\mathsf{key}, t_0)$. Furthermore, we now use $\mathsf{key}$ instead of $\mathsf{eqKey}$ to answer all function-tag queries before the $i$'th one.
   $\mathsf{Game}^{i.4}_{2\to 3}$ *is identically distributed to* $\mathsf{Game}^{i.5}_{2\to 3}$. *The changes before the "furthermore" are just syntactic. In both cases* $t_1, r^*$ *are random subject to* $t_1 \oplus r^* = \widetilde{C}$. *The "furthermore" part is identical by the correctness of the SEPRF which says that* $\mathsf{PRF}(\mathsf{eqKey}, t_0') = \mathsf{PRF}(\mathsf{key}, t_0')$ *for all* $t_0' \neq t_0$, *and if* $t_0' = t_0$ *is ever chosen before the i'th query then the game outputs*

   0 *in either case. Recall this rule about outputting 0 when $t_0$ values are repeated was adopted in* $\mathsf{Game}_1$.
6. $\mathsf{Game}^{i+1}_{2\to 3}$: This is identical to $\mathsf{Game}^{i.5}_{2\to 3}$, except that, instead of choosing

$$t_0 \leftarrow \{0,1\}^\lambda,\ \mathsf{eqKey} \leftarrow \mathsf{Sim}_1(1^\lambda, t_0),\ r^* \leftarrow \{0,1\}^\ell,\ \mathsf{key} \leftarrow \mathsf{Sim}_2(\mathsf{eqKey}, r^*)$$

   at the beginning of the game we now just choose $\mathsf{key} \leftarrow \mathsf{KeyGen}(1^\lambda)$ at the very beginning and wait to choose $t_0$ until the $i$'th function-tag query. $\mathsf{Game}^{i.5}_{2\to 3}$ *is computationally indistinguishable from* $\mathsf{Game}^{i+1}_{2\to 3}$ *by SEPRF security.*

Therefore the combination of the above hybrids shows that for each $i$: $\mathsf{Game}^{i}_{2\to 3}$ is computationally indistinguishable from $\mathsf{Game}^{i+1}_{2\to 3}$ and therefore $\mathsf{Game}_2$ is computationally indistinguishable from $\mathsf{Game}_3$.

– $\mathsf{Game}_4$ For all *post-challenge function-tag queries*, switch to answering them using $\mathsf{SFunctionTag}(\mathsf{tsk}, \cdot)$ instead of $\mathsf{DFunctionTag}(\cdot)$. Indistinguishability follows via a sequence of internal hybrids $\mathsf{Game}^{i}_{3\to 4}$ where the first $i$ post-challenge function-tag queries are answered using $\mathsf{SFunctionTag}(\mathsf{tsk}, \cdot)$ and the rest are answered using $\mathsf{DFunctionTag}(\cdot)$. Note that if the adversary makes $q$ such queries then $\mathsf{Game}^{0}_{3\to 4}$ is identical to $\mathsf{Game}_3$ and $\mathsf{Game}^{q}_{3\to 4}$ is identical to $\mathsf{Game}_4$. To switch from $\mathsf{Game}^{i}_{3\to 4}$ to $\mathsf{Game}^{i+1}_{3\to 4}$ we introduce further sub-hybrids as follows (essentially a simpler version of the sub-hybrids needed to go from $\mathsf{Game}_2$ to $\mathsf{Game}_3$ since we do not need to equivocate the SEPRF here):
  - $\mathsf{Game}^{i.1}_{3\to 4}$: We change how the $i$'th post-challenge function-tag query is answered from choosing $t_1 \leftarrow \{0,1\}^\ell$ to choosing $u \leftarrow \lambda$, $\widetilde{C} \leftarrow \mathsf{SimCircuit}(\mathsf{sk}, x, u)$ and setting $t_1 := \widetilde{C} \oplus \mathsf{PRF}(\mathsf{key}, t_0)$. We still choose $t_2 \leftarrow \{0,1\}^\lambda$ uniformly at random.
    $\mathsf{Game}^{i}_{3\to 4}$ *is distributed identically to* $\mathsf{Game}^{i.1}_{3\to 4}$ *by the blindness property of blind garbled circuits and the fact that $u \leftarrow \{0,1\}^\lambda$ is chosen randomly, which ensures that $\widetilde{C}$ is uniformly random over $\{0,1\}^\ell$.*
  - $\mathsf{Game}^{i+1}_{3\to 4}$: We change how the $i$'th post-challenge function-tag query with function $f_i$ is answered from $\widetilde{C} \leftarrow \mathsf{SimCircuit}(\mathsf{sk}, x, u)$ to choosing $\widetilde{C} \leftarrow \mathsf{GCircuit}(\mathsf{sk}, C^*_{f_i,u,t_2})$.
    $\mathsf{Game}^{i.1}_{3\to 4}$ *is computationally indistinguishable from* $\mathsf{Game}^{i+1}_{3\to 4}$ *by the semi-adaptive simulation security of the garbled circuit. The reduction does not know* $\mathsf{sk}$. *During the challenge input-tag query for input $x$, the reduction hands $x$ to its challenger to get $\widetilde{x}$ and uses it to answer the challenge input-tag query. It uses its oracle to* $\mathsf{GCircuit}(\mathsf{sk}, \cdot)$ *to answer all calls to* $\mathsf{SFunctionTag}(\mathsf{tsk}, \cdot)$ *aside from the i'th post-challenge function-tag after the input-tag query. For the i'th post-challenge function-tag query it picks the challenge circuit $C^*_{f,u,t_2}$ and gets $\widetilde{C}$ from its oracle which it uses to answer that call. If $\widetilde{C} \leftarrow$* $\mathsf{SimCircuit}(\mathsf{sk}, x, u)$ *then the game is identical to* $\mathsf{Game}^{i.1}_{3\to 3}$ *and if $\widetilde{C} \leftarrow$* $\mathsf{GCircuit}(\mathsf{sk}, C^*_{f_i,u,t_2})$ *then the game is identical to* $\mathsf{Game}^{i+1}_{3\to 4}$. *Here we rely on the fact that $f_i(x) = 0$* to ensure that *$C^*_{f_i,u,t_2}(x) = u$.*

Therefore the combination of the above hybrids shows that for each $i$: $\mathsf{Game}_{3\to4}^{i}$ is computationally indistinguishable from $\mathsf{Game}_{3\to4}^{i+1}$ and therefore $\mathsf{Game}_3$ is computationally indistinguishable from $\mathsf{Game}_4$.

– $\mathsf{Game}_5$: This is the game $\mathsf{FunTagGame}^1$ which outputs $\mathcal{A}^{\mathsf{SInputTag}(1^\lambda,\cdot),\mathsf{SFunctionTag}(1^\lambda,\cdot)}(1^\lambda)$. This is the same as $\mathsf{Game}_4$ except that we "undo" the change from $\mathsf{Game}_1$: if the oracle $\mathsf{SFunctionTag}(1^\lambda,\cdot)$ ever samples a value $t_0$ that was already used in the response to a previous query, we continue as usual instead of defining the output of the game to be 0.
*It is easy to see that* $\mathsf{Game}_4$ *and* $\mathsf{Game}_5$ *are statistically indistinguishable since* $t_0 \leftarrow \{0,1\}^\lambda$ *is chosen randomly each time.*

The above sequence of hybrids shows that $\mathsf{FunTagGame}^0$ and $\mathsf{FunTagGame}^1$ are computationally indistinguishable, which proves security.

## 5 Adaptive ABE from WE via a Functional Tag System

We construct an adaptively secure ABE scheme for circuits using:

– a statistically binding commitment scheme $(\mathsf{Com.Setup}, \mathsf{Commit})$ per Definition 2,
– a statistically sound witness indistinguishable NIZK for NP $(\mathsf{NIZK.Setup}, \mathsf{Prove}, \mathsf{Verify})$ per Definition 3,
– a witness encryption scheme for NP $(\mathsf{WE.Enc}, \mathsf{WE.Dec})$ per Definition 4,
– a functional tag system for circuits $(\mathsf{DInputTag}, \mathsf{DFunctionTag}, \mathsf{SGen}, \mathsf{SInputTag}, \mathsf{SFunctionTag}, \mathsf{Trigger})$ per Definition 6 where the tag key $\mathsf{tsk} \leftarrow \mathsf{SGen}(1^\lambda)$ is of length $|\mathsf{tsk}| = \ell(\lambda)$.

For any polynomials $s(\lambda), n(\lambda)$, let us fix the function class $\mathcal{C}_\lambda^{s,n}$ to consist of boolean circuits of size $s(\lambda)$ with $n(\lambda)$-bit input. We construct an ABE for the function class $\mathcal{C}_\lambda^{s,n}$ using a functional tag system for $\mathcal{C}_\lambda^{s,n}$ as a building block. We specify the NP relations $\mathsf{NIZK}.R, \mathsf{WE}.R$ for the NIZK and WE inside the construction.

*Construction.* The ABE scheme $(\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ is defined as follows:

– $(\mathsf{msk}, \mathsf{mpk}) \leftarrow \mathsf{Setup}(1^\lambda)$: Choose $\mathsf{NIZK.crs} \leftarrow \mathsf{NIZK.Setup}(1^\lambda)$, $\mathsf{Com.crs} \leftarrow \mathsf{Com.Setup}(1^\lambda)$, $r_0, r_1 \leftarrow \{0,1\}^\lambda$, and set

$$\mathsf{com}_0 := \mathsf{Commit}_{\mathsf{Com.crs}}(0; r_0), \mathsf{com}_1 := \mathsf{Commit}_{\mathsf{Com.crs}}(0^{\ell(\lambda)}; r_1).$$

Output $\mathsf{mpk} := (\mathsf{Com.crs}, \mathsf{NIZK.crs}, \mathsf{com}_0, \mathsf{com}_1)$, $\mathsf{msk} := r_0$.

– $\mathsf{sk}_f \leftarrow \mathsf{KeyGen}(\mathsf{msk}, f)$: Generate $\mathsf{tag}_f \leftarrow \mathsf{DFunctionTag}(1^\lambda, f)$. Give a NIZK proof

$$\pi \leftarrow \mathsf{Prove}_{\mathsf{NIZK.crs}}(\ \widetilde{x} = (\mathsf{Com.crs}, \mathsf{com}_0, \mathsf{com}_1, f, \mathsf{tag}_f)\ ,\ \widetilde{w} = r_0\ )$$

for the NP relation

$$\mathsf{NIZK}.R = \left\{ (\widetilde{x}, \widetilde{w}) \ : \ \begin{array}{l} \qquad\qquad \widetilde{x} = (\mathsf{Com.crs}, \mathsf{com}_0, \mathsf{com}_1, f, \mathsf{tag}_f) \\ \text{either } \ \widetilde{w} = r_0 \qquad\quad : \mathsf{com}_0 = \mathsf{Commit}_{\mathsf{Com.crs}}(0; r_0) \\ \text{or } \widetilde{w} = (\mathsf{tsk}, r_1, r_2) : \mathsf{com}_1 = \mathsf{Commit}_{\mathsf{Com.crs}}(\mathsf{tsk}; r_1) \\ \qquad\qquad\qquad\qquad\quad \wedge \mathsf{tag}_f = \mathsf{SFunctionTag}(\mathsf{tsk}, f; r_2) \end{array} \right\}.$$

Output $\mathsf{sk}_f = (f, \mathsf{tag}_f, \pi)$

- $\mathsf{ct} \leftarrow \mathsf{Enc}(\mathsf{mpk}, x, \mu)$: Generate $\mathsf{tag}_x \leftarrow \mathsf{DInputTag}(1^\lambda, x)$ and a witness encryption $\mathsf{WE.ct} \leftarrow \mathsf{WE.Enc}(1^\lambda, \hat{x} = (\mathsf{Com.crs}, \mathsf{NIZK.crs}, x, \mathsf{com}_0, \mathsf{com}_1, \mathsf{tag}_x), \mu)$ for the relation

$$\mathsf{WE}.R = \left\{ (\hat{x}, \hat{w}) \ : \ \begin{array}{l} \qquad \hat{x} = (\mathsf{Com.crs}, \mathsf{NIZK.crs}, \mathsf{com}_0, \mathsf{com}_1, x, \mathsf{tag}_x), \hat{w} = (f, \mathsf{tag}_f, \pi) \\ \mathsf{Verify}_{\mathsf{NIZK.crs}}(\widetilde{x} = (\mathsf{Com.crs}, \mathsf{com}_0, \mathsf{com}_1, f, \mathsf{tag}_f), \pi) = 1 \\ \wedge f(x) = 1 \wedge \mathsf{Trigger}(\mathsf{tag}_f, \mathsf{tag}_x) = 0 \end{array} \right\}.$$

Output $\mathsf{ct} = (x, \mathsf{tag}_x, \mathsf{WE.ct})$.

- $\mu := \mathsf{Dec}(\mathsf{sk}_f, \mathsf{ct})$: Output $\mu := \mathsf{WE.Dec}(\mathsf{WE.ct}, (f, \mathsf{tag}_f, \pi))$.

**Theorem 6.** *Assuming witness encryption for NP, statistically sound NIZK for NP, statistically binding commitments and a functional tag system for circuits there exists an adaptively secure ABE for circuits.*

*In particular, the above holds assuming witness encryption for NP, statistically sound NIZK for NP, and one-way functions. Alternately, the above holds assuming witness encryption for NP and any one of: (1) hardness of factoring, or (2) the decisional linear assumption in bilinear groups, or (3) the learning with errors (LWE) assumption. Lastly, the above holds just assuming evasive LWE.*

*Proof.* We show that the construction given above is an adaptively secure ABE for $\mathcal{C}_\lambda^{s,n}$ assuming the security of the components. The correctness of the ABE follows from the correctness of the WE and NIZK along with correctness (property 1) of the functional tag system. To prove adaptive security, we define a sequence of games:

- $\mathsf{Game}_0^b$: This is the ABE game $\mathsf{ABEGame}^b$ between the adversary and the challenger.
- $\mathsf{Game}_1^b$: We modify the game so that the challenger initially chooses a "smart tag system key" $\mathsf{tsk} \leftarrow \mathsf{SGen}(1^\lambda)$. When answering key queries, the challenger now samples keys $\mathsf{sk}_f = (f, \mathsf{tag}_f, \pi)$ by choosing a "smart function tag" $\mathsf{tag}_f \leftarrow \mathsf{SFunctionTag}(\mathsf{tsk}, f)$ instead of a dummy one. For the challenge ciphertext $\mathsf{ct} = (x, \mathsf{tag}_x, \mathsf{WE.ct})$, the challenger now chooses a "smart input tag" $\mathsf{tag}_x \leftarrow \mathsf{SInputTag}(\mathsf{tsk}, x)$ instead of a dummy one. The keys and the challenge ciphertext are otherwise generated the same way as previously.
  $\mathsf{Game}_0^b$ *and* $\mathsf{Game}_1^b$ *are indistinguishable by the security property (property 3) of the functional tag system. Note that in the ABE adaptive security game, the adversary can only choose attribute* $x$ *such that* $f_i(x) = 0$ *for all key queries* $f_i$*, which matches the restriction on the adversarial queries of the functional tag system.*

- $\mathsf{Game}_2^b$: We modify the game so that, when choosing $\mathsf{mpk} = (\mathsf{Com.crs}, \mathsf{NIZK.crs}, \mathsf{com}_0, \mathsf{com}_1)$, the challenger sets $\mathsf{com}_1 := \mathsf{Commit}_{\mathsf{Com.crs}}(\mathsf{tsk}; r_1)$ to be a commitment to $\mathsf{tsk}$ instead of $0^{\ell(\lambda)}$.
  $\mathsf{Game}_1^b$ *and* $\mathsf{Game}_2^b$ *are indistinguishable by the computational hiding security of the commitment scheme. Note that the commitment randomness* $r_1$ *does not appear anywhere else in the game.*
- $\mathsf{Game}_3^b$: We modify how the challenger answers key queries with keys $\mathsf{sk}_f = (f, \mathsf{tag}_f, \pi)$. In particular, the challenger now generates the proof $\pi$ as:

$$\pi \leftarrow \mathsf{Prove}_{\mathsf{NIZK.crs}}(\ \widetilde{x} = (\mathsf{Com.crs}, \mathsf{com}_0, \mathsf{com}_1, f, \mathsf{tag}_f)\ ,\ \widetilde{w} = (\mathsf{tsk}, r_1, r_2)\ )$$

  using the witness $\widetilde{w} = (\mathsf{tsk}, r_1, r_2)$ where $r_2$ is the randomness used to generate $\mathsf{tag}_f := \mathsf{SFunctionTag}(\mathsf{tsk}, f; r_2)$, instead of using the witness $\widetilde{w} = r_0$.
  $\mathsf{Game}_2^b$ *and* $\mathsf{Game}_3^b$ *are indistinguishable by witness indistinguishability security of the NIZK.*
- $\mathsf{Game}_4^b$: In this game, when choosing the master public key $\mathsf{mpk} = (\mathsf{Com.crs}, \mathsf{NIZK.crs}, \mathsf{com}_0, \mathsf{com}_1)$, the challenger now sets $\mathsf{com}_0 := \mathsf{Commit}_{\mathsf{Com.crs}}(1; r_1)$ to be a commitment to 1 instead of 0.
  $\mathsf{Game}_3^b$ *and* $\mathsf{Game}_4^b$ *are indistinguishable by the computational hiding security of the commitment scheme. Note that the commitment randomness* $r_0$ *does not appear anywhere else in the game.*
- $\mathsf{Game}_5^b$: In this game, when choosing the challenge ciphertext $\mathsf{ct} = (x, \mathsf{tag}_x, \mathsf{WE.ct})$, the challenger samples

$$\mathsf{WE.ct} \leftarrow \mathsf{WE.Enc}(1^\lambda, \hat{x} = (\mathsf{Com.crs}, \mathsf{NIZK.crs}, x, \mathsf{com}_0, \mathsf{com}_1, \mathsf{tag}_x), 0)$$

  to be an encryption of 0 rather than the bit $b$.
  $\mathsf{Game}_4^b$ *and* $\mathsf{Game}_5^b$ *are indistinguishable by WE security. Firstly, note that whenever* $\mathsf{Com.crs}$ *is binding and* $\mathsf{NIZK.crs}$ *is sound (see Definitions 2 and 3) then the statement* $\hat{x}$ *is false. To see this, assume otherwise that* $(\hat{x}, \hat{w}) \in \mathsf{WE}.R$ *for some* $\hat{w} = (f, \mathsf{tag}_f, \pi)$. *Then it must hold that* $f(x) = 1$, $\mathsf{Trigger}(\mathsf{tag}_f, \mathsf{tag}_x) = 0$ *and* $\mathsf{Verify}_{\mathsf{NIZK.crs}}(\widetilde{x} = (\mathsf{Com.crs}, \mathsf{com}_0, \mathsf{com}_1, f, \mathsf{tag}_f), \pi) = 1$. *The latter implies that there exists some* $\widetilde{w}$ such that $(\widetilde{x}, \widetilde{w}) \in \mathsf{NIZK}.R$. *Since* $\mathsf{com}_0 = \mathsf{Commit}_{\mathsf{Com.crs}}(1; r_0), \mathsf{com}_1 = \mathsf{Commit}_{\mathsf{Com.crs}}(\mathsf{tsk}; r_1)$ *this in turn implies that* $\mathsf{tag}_f = \mathsf{SFunctionTag}(\mathsf{tsk}, f; r_2)$ for some $r_2$. *But since* $\mathsf{tagtag}_x \leftarrow \mathsf{SInputTag}(\mathsf{tsk}, x)$*, the above contradicts property 2 of the functional tag system. Secondly, note that* $\mathsf{Com.crs}$ *is binding and* $\mathsf{NIZK.crs}$ *is sound with overwhelming probability, and therefore the statement* $\hat{x}$ is *false with overwhelming probability. Given the above, an adversary distinguishes* $\mathsf{Game}_4^b$ *and* $\mathsf{Game}_5^b$ *with non-negligible probability must distinguish* $\mathsf{WE.Enc}(1^\lambda, \hat{x}, 0)$ *and* $\mathsf{WE.Enc}(1^\lambda, \hat{x}, 1)$ *for a false statement* $\hat{x}$ *with non-negligible probability.*

Note that $\mathsf{Game}_5^0 \equiv \mathsf{Game}_5^1$ since the game completely ignores the bit $b$. Therefore, by the hybrid argument, we have $\mathsf{Game}_0^0$ is indistinguishable from $\mathsf{Game}_0^1$, which implies ABE security.

## References

[ABSV15] Ananth, P., Brakerski, Z., Segev, G., Vaikuntanathan, V.: From selective to adaptive security in functional encryption. In: Gennaro, R., Robshaw, M. (eds.) CRYPTO 2015, Part II. LNCS, vol. 9216, pp. 657–677. Springer, Heidelberg (2015). https://doi.org/10.1007/978-3-662-48000-7_32

[BB04] Boneh, D., Boyen, X.: Secure identity based encryption without random oracles. In: Franklin, M. (ed.) CRYPTO 2004. LNCS, vol. 3152, pp. 443–459. Springer, Heidelberg (2004). https://doi.org/10.1007/978-3-540-28628-8_27

[BF01] Boneh, D., Franklin, M.: Identity-based encryption from the Weil pairing. In: Kilian, J. (ed.) CRYPTO 2001. LNCS, vol. 2139, pp. 213–229. Springer, Heidelberg (2001). https://doi.org/10.1007/3-540-44647-8_13

[BGG+14] Boneh, D., et al.: Fully key-homomorphic encryption, arithmetic circuit ABE and compact garbled circuits. In: Nguyen, P.Q., Oswald, E. (eds.) EUROCRYPT 2014. LNCS, vol. 8441, pp. 533–556. Springer, Heidelberg (2014). https://doi.org/10.1007/978-3-642-55220-5_30

[BIOW20] Barta, O., Ishai, Y., Ostrovsky, R., Wu, D.J.: On succinct arguments and witness encryption from groups. In: Micciancio, D., Ristenpart, T. (eds.) CRYPTO 2020. LNCS, vol. 12170, pp. 776–806. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-56784-2_26

[BLSV18] Brakerski, Z., Lombardi, A., Segev, G., Vaikuntanathan, V.: Anonymous IBE, leakage resilience and circular security from new assumptions. In: Nielsen, J.B., Rijmen, V. (eds.) EUROCRYPT 2018, Part I. LNCS, vol. 10820, pp. 535–564. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-78381-9_20

[BV16] Brakerski, Z., Vaikuntanathan, V.: Circuit-ABE from LWE: unbounded attributes and semi-adaptive security. In: Robshaw, M., Katz, J. (eds.) CRYPTO 2016. LNCS, vol. 9816, pp. 363–384. Springer, Heidelberg (2016). https://doi.org/10.1007/978-3-662-53015-3_13

[CCH+19] Canetti, R., et al.: Fiat-Shamir: from practice to theory. In: Charikar, M., Cohen, E. (eds.) 51st Annual ACM Symposium on Theory of Computing, pp. 1082–1090. ACM Press (2019)

[CHK03] Canetti, R., Halevi, S., Katz, J.: A forward-secure public-key encryption scheme. In: Biham, E. (ed.) EUROCRYPT 2003. LNCS, vol. 2656, pp. 255–271. Springer, Heidelberg (2003). https://doi.org/10.1007/3-540-39200-9_16

[DG17] Döttling, N., Garg, S.: Identity-based encryption from the Diffie-Hellman assumption. In: Katz, J., Shacham, H. (eds.) CRYPTO 2017. LNCS, vol. 10401, pp. 537–569. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-63688-7_18

[FLS90] Feige, U., Lapidot, D., Shamir, A.: Multiple non-interactive zero knowledge proofs based on a single random string (extended abstract). In: 31st Annual Symposium on Foundations of Computer Science, pp. 308–317. IEEE Computer Society Press (1990)

[FWW23] Freitag, C., Waters, B., David, J.W.: How to use (plain) witness encryption: registered ABE, flexible broadcast, and more. In: Handschuh, H., Lysyanskaya, A. (eds.) CRYPTO 2023, Part IV. LNCS, vol. 14084, pp. 498–531. Springer, Heidelberg (2023). https://doi.org/10.1007/978-3-031-38551-3_16

[Gen06] Gentry, C.: Practical identity-based encryption without random oracles. In: Vaudenay, S. (ed.) EUROCRYPT 2006. LNCS, vol. 4004, pp. 445–464. Springer, Heidelberg (2006). https://doi.org/10.1007/11761679_27

[GGSW13] Garg, S., Gentry, C., Sahai, A., Waters, B.: Witness encryption and its applications. In: Boneh, D., Roughgarden, T., Feigenbaum, J. (eds.) 45th Annual ACM Symposium on Theory of Computing, pp. 467–476. ACM Press (2013)

[GKP+13] Goldwasser, S., Kalai, Y.T., Popa, R.A., Vaikuntanathan, V., Zeldovich, N.: Reusable garbled circuits and succinct functional encryption. In: Boneh, D., Roughgarden, T., Feigenbaum, J. (eds.) 45th Annual ACM Symposium on Theory of Computing, pp. 555–564. ACM Press (2013)

[GKW16] Goyal, R., Koppula, V., Waters, B.: Semi-adaptive security and bundling functionalities made generic and easy. In: Hirt, M., Smith, A. (eds.) TCC 2016, Part II. LNCS, vol. 9986, pp. 361–388. Springer, Heidelberg (2016). https://doi.org/10.1007/978-3-662-53644-5_14

[GMM17] Garg, S., Mahmoody, M., Mohammed, A.: Lower bounds on obfuscation from all-or-nothing encryption primitives. In: Katz, J., Shacham, H. (eds.) CRYPTO 2017, Part I. LNCS, vol. 10401, pp. 661–695. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-63688-7_22

[GOS06] Groth, J., Ostrovsky, R., Sahai, A.: Perfect non-interactive zero knowledge for NP. In: Vaudenay, S. (ed.) EUROCRYPT 2006. LNCS, vol. 4004, pp. 339–358. Springer, Heidelberg (2006). https://doi.org/10.1007/11761679_21

[GPSW06] Goyal, V., Pandey, O., Sahai, A., Waters, B.: Attribute-based encryption for fine-grained access control of encrypted data. In: Juels, A., Wright, R.N., di Vimercati, S.D.C. (eds.) ACM CCS 2006: 13th Conference on Computer and Communications Security, pp. 89–98. ACM Press (2006). Available as Cryptology ePrint Archive Report 2006/309

[GVW13] Gorbunov, S., Vaikuntanathan, V., Wee, H.: Attribute-based encryption for circuits. In: Boneh, D., Roughgarden, T., Feigenbaum, J. (eds.) 45th Annual ACM Symposium on Theory of Computing, pp. 545–554. ACM Press (2013)

[HJO+16] Hemenway, B., Jafargholi, Z., Ostrovsky, R., Scafuro, A., Wichs, D.: Adaptively secure garbled circuits from one-way functions. In: Robshaw, M., Katz, J. (eds.) CRYPTO 2016, Part III. LNCS, vol. 9816, pp. 149–178. Springer, Heidelberg (2016). https://doi.org/10.1007/978-3-662-53015-3_6

[JLS21] Jain, A., Lin, H., Sahai, A.: Indistinguishability obfuscation from well-founded assumptions. In: Khuller, S., Williams, V.V. (eds.) 53rd Annual ACM Symposium on Theory of Computing, pp. 60–73. ACM Press (2021)

[LOS+10] Lewko, A., Okamoto, T., Sahai, A., Takashima, K., Waters, B.: Fully secure functional encryption: attribute-based encryption and (hierarchical) inner product encryption. In: Gilbert, H. (ed.) EUROCRYPT 2010. LNCS, vol. 6110, pp. 62–91. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-13190-5_4

[LW14] Lewko, A., Waters, B.: Why proving HIBE systems secure is difficult. In: Nguyen, P.Q., Oswald, E. (eds.) EUROCRYPT 2014. LNCS, vol. 8441, pp. 58–76. Springer, Heidelberg (2014). https://doi.org/10.1007/978-3-642-55220-5_4

[Nao91] Naor, M.: Bit commitment using pseudorandomness. J. Cryptol. **4**(2), 151–158 (1991)

[PRV12] Parno, B., Raykova, M., Vaikuntanathan, V.: How to delegate and verify in public: verifiable computation from attribute-based encryption. In: Cramer, R. (ed.) TCC 2012. LNCS, vol. 7194, pp. 422–439. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-28914-9_24

[PS19] Peikert, C., Shiehian, S.: Noninteractive zero knowledge for NP from (plain) learning with errors. In: Boldyreva, A., Micciancio, D. (eds.) CRYPTO 2019. LNCS, vol. 11692, pp. 89–114. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-26948-7_4

[Reg05] Regev, O.: On lattices, learning with errors, random linear codes, and cryptography. In: Gabow, H.N., Fagin, R. (eds.) 37th Annual ACM Symposium on Theory of Computing, pp. 84–93. ACM Press (2005)

[Sha84] Shamir, A.: Identity-based cryptosystems and signature schemes. In: Blakley, G.R., Chaum, D. (eds.) CRYPTO 1984. LNCS, vol. 196, pp. 47–53. Springer, Heidelberg (1985). https://doi.org/10.1007/3-540-39568-7_5

[SW05] Sahai, A., Waters, B.: Fuzzy identity-based encryption. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 457–473. Springer, Heidelberg (2005). https://doi.org/10.1007/11426639_27

[Tsa19] Tsabary, R.: Fully secure attribute-based encryption for $t$-CNF from LWE. In: Boldyreva, A., Micciancio, D. (eds.) CRYPTO 2019. LNCS, vol. 11692, pp. 62–85. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-26948-7_3

[Tsa22] Tsabary, R.: Candidate witness encryption from lattice techniques. In: Dodis, Y., Shrimpton, T. (eds.) CRYPTO 2022, Part I. LNCS, vol. 13507, pp. 535–559. Springer, Heidelberg (2022). https://doi.org/10.1007/978-3-031-15802-5_19

[VWW22] Vaikuntanathan, V., Wee, H., Wichs, D.: Witness encryption and null-IO from evasive LWE. In: Agrawal, S., Lin, D. (eds.) ASIACRYPT 2022, Part I. LNCS, vol. 13791, pp. 195–221. Springer, Heidelberg (2022). https://doi.org/10.1007/978-3-031-22963-3_7

[Wat05] Waters, B.: Efficient identity-based encryption without random oracles. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 114–127. Springer, Heidelberg (2005). https://doi.org/10.1007/11426639_7

[Wat09] Waters, B.: Dual system encryption: realizing fully secure IBE and HIBE under simple assumptions. In: Halevi, S. (ed.) CRYPTO 2009. LNCS, vol. 5677, pp. 619–636. Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-642-03356-8_36

[Wat15] Waters, B.: A punctured programming approach to adaptively secure functional encryption. In: Gennaro, R., Robshaw, M. (eds.) CRYPTO 2015, Part II. LNCS, vol. 9216, pp. 678–697. Springer, Heidelberg (2015). https://doi.org/10.1007/978-3-662-48000-7_33

# Limits on Adaptive Security for Attribute-Based Encryption

Zvika Brakerski and Stav Medina[(✉)]

Weizmann Institute of Science, Rehovot, Israel
medina.stav@gmail.com

**Abstract.** This work addresses the long quest for proving full (adaptive) security for attribute-based encryption (ABE). We show that in order to prove full security in a black-box manner, the scheme must be "irregular" in the sense that it is impossible to "validate" secret keys to ascertain consistent decryption of ciphertexts. This extends a result of Lewko and Waters (Eurocrypt 2014) that was only applicable to straight-line proofs (without rewinding). Our work, therefore, establishes that it is impossible to circumvent the irregularity property using creative proof techniques, so long as the adversary is used in a black-box manner.

As a consequence, our work provides an explanation as to why some lattice-based ABE schemes cannot be proven fully secure, even though no known adaptive attacks exist.

## 1 Introduction

An Attribute-Based Encryption scheme (ABE) [SW05,GPSW06] is one that allows fine-grained access to encrypted data by issuing multiple secret keys, each with its own permissions, and protecting the privacy of ciphertext even against colluding unauthorized parties. More explicitly, an ABE scheme consists of a global pair of "master" public-key (also known as the public parameters of the scheme) and secret-key, where the former is used to encrypt messages, and the latter is used to issue individual decryption keys. Messages are encrypted, using the public parameters, with respect to an *attribute* $x$ (for our purposes $x \in \{0,1\}^n$). Secret keys are generated using the master secret-key, with respect to predicate functions $f$, so that $\mathrm{SK}_f$ can decrypt all ciphertexts with attributes $x$ for which $f(x) = 1$.[1] The security requirement is *collusion resilience*. Namely, even if an attacker has as many $\mathrm{SK}_{f_i}$

[1] The description here is for a variant known as Key-Policy ABE (KP-ABE). There is another variant known as Ciphertext-Policy ABE (CP-ABE) where encryption is with respect to $f$, and secret-key generation is with respect to $x$. The distinction is not very important for our purposes, so we adopt the KP-ABE notation throughout this manuscript.

For the full and most up to date version of this work, see https://eprint.iacr.org/2023/952. Authors are supported by the Israel Science Foundation (Grant No. 3426/21), and by the Horizon Europe Research and Innovation Program via ERC Project ACQUA (Grant 101087742).

E. Boyle and M. Mahmoody (Eds.): TCC 2024, LNCS 15366, pp. 91–123, 2025.
https://doi.org/10.1007/978-3-031-78020-2_4

as they want, if $f_i(x) = 0$ for all $i$, then the attacker cannot decrypt ciphertexts with attribute $x$. ABE schemes that support sufficiently rich function classes (even the class of shallow circuits or the class of boolean formulae) are known to exist only under two types of cryptographic assumptions: assumptions on groups with bilinear maps [SW05, GPSW06, OSW07, Wat11, LOS+10, LW12, KL15, CGKW18, KW20, GW20, LL20] and lattice assumptions [AFV11, ABV+12, Boy13, GVW13, BGG+14].

ABE proved to be a useful primitive for many purposes (see [LOS+10, AFV11, DDM15, GKW17, DGP21] for just a few samples among many). However, proving security for ABE is a challenging task. In the security reduction, the ABE adversary is used to violate the underlying hardness assumption. Since the ABE adversary is allowed to request multiple keys $\mathrm{SK}_{f_i}$, the proof needs to be designed so that such keys can be generated and fed to the adversary, where at the same time, there is a challenge of the underlying assumption that remains unsolved in the eyes of the reduction algorithm. Therefore, in many cases, security is proved in a relaxed model, which is known as *selective security.* In this model, the adversary needs to declare, ahead of time, the value $x^*$ on which it wishes to violate security [CHK03]. This allows the reduction to design the public parameters so that it is possible to generate $\mathrm{SK}_{f_i}$ for which $f_i(x^*) = 0$. However, this naturally restricts the attacker's power since, in an actual attack, the adversary may be exposed to some $\mathrm{SK}_{f_i}$ and only then choose $x^*$. Protecting against the latter is known as full security, or as *adaptive security*, to emphasize the possibility of the aforementioned adaptive attacks. An intermediate notion, semi-adaptive security, allows $x^*$ to be chosen after seeing the public parameters but before seeing any actual key. This latter notion appears to be more similar to selective than to adaptive security and can be achieved in similar ways to the selective case [BV16, GKW16].

Adaptive security is notoriously hard to prove. Intuitively, this is because the reduction needs to be prepared to "feed" the adversary with a key of their choice. For example, the reduction does not know whether the adversary will ask for a key for a function $f$ or for its complement. Therefore, in a sense, the reduction should have the ability to decrypt any ciphertext without the adversary's help. Indeed, until recently, fully secure ABE was *only* known to exist under assumptions on bilinear maps, mostly using the dual-system technique of Waters and its successors [Wat09].[2] Recently, Tsabary [Tsa19] showed an approach towards full security in the lattice regime, but only for a very restricted class of functions $f$ (in particular, this is still not known for general shallow circuits or for the class of boolean formulae).

Nevertheless, as hard as it is to prove adaptive security, actual adaptive attacks are not so common. In fact, we are not aware of adaptive attacks against the lattice-based schemes of [GVW13, BGG+14]. It is, therefore, quite puzzling

[2] We note that there is a way to generically upgrade selective to adaptive security, at the cost of a $2^n$ factor degradation in security, by simply guessing the value of $x^*$. This is known as *complexity leveraging.* However, the cost is prohibitive in many cases, and furthermore, this method cannot be applied if $n$ is not a-priori bounded.

that we need to apply involved techniques and lose a lot of functionality (as of yet) in order to prove this property.

Lewko and Waters [LW14] tried to formalize the above intuition on the hardness of proving adaptive security. They noticed that since the reduction needs to be able to produce keys that essentially violate the security of any individual ciphertext, one can *extract from the reduction itself* information that allows violating the hardness assumption, thus trivializing the proof. This extraction is done via *rewinding*. At a high level, we consider an algorithm that runs the reduction as an adversary, asks for a key for the function $f_0$, and then *rewinds* the reduction to the point before the key was queried. It then asks for a different key $f_1$ and claims to violate security for some $x^*$ for which $f_0(x^*) = 1$ and $f_1(x^*) = 0$. Indeed, security is violated by using $\mathrm{SK}_{f_0}$ that was obtained in the rewound part of the execution. The current thread of the reduction "thinks" that only $f_1$ was queried, thereby assuming it is interacting with a legitimate, successful ABE adversary. This should lead to the reduction of violating the hardness assumption in polynomial time.

One has to be careful when applying this argument (especially given that adaptive security via black-box reductions *is* possible to achieve in some cases). In order for it to work, the reduction should not notice that the challenge ciphertext is being decrypted by a key that was obtained in another thread. Therefore, $\mathrm{SK}_{f_0}$ should decrypt ciphertexts in a "canonical" manner that does not expose the origins of the key. Lewko and Waters, therefore, defined a criterion for secret keys and ciphertexts, essentially requiring that it is not possible to distinguish different decryptions of a ciphertext, even if they were obtained using different keys (so long as all secret keys and ciphertext involved pass a public validation procedure). This is a very natural property, and thus the Lewko-Waters result has the following very strong implication. In order to achieve provable adaptive security, one must forgo the ability to validate secret keys and ciphertexts to ensure that decryption is done in a consistent manner. We are not aware of a setting where this "checkability" property is explicitly required, but we imagine that the ability to validate keys and ciphertexts may be desirable in a multi-user system.

Existing methods for constructing fully secure ABE do this by making their scheme impossible to validate. This applies to Waters's aforementioned dual-system technique, where the proof utilizes "semi-functional" keys and ciphertexts, which differ in functionality from "regular" keys, thus inherently violating the checkability property. Tsabary's approach relies on generating a special ciphertext that some policy-accepting honestly-generated secret keys cannot decrypt. Therefore, in the security proof, the challenge ciphertext would decrypt to different values if the attacker attempted to decrypt it with different (policy-accepting) secret keys. Here again, it is inherently impossible to come up with a validation procedure for the scheme. We note that one can also derive adaptively-secure ABE from a stronger notion known as (adaptively-secure) Functional Encryption (FE) which is intimately related to program obfuscation [GGH+13]. Known constructions of adaptively-secure FE [Wat15, ABSV15] use a so-called "punctured programs" technique which, similarly to dual-systems inherently violate checkability.

An important limitation of the Lewko-Waters result is that it does not apply when the reduction itself rewinds the adversary. Indeed, rewinding is a very common proof technique in cryptography. For example, the reduction, upon receiving a request to provide a key for $f_0$, may rewind the adversary and test it on public parameters that the reduction generated by itself in order to see that it actually manages to solve ABE "dummy challenges" before actually providing it with keys with respect to the "real" public parameters.

If we then try to apply the outline above, our algorithm tries to rewind the reduction, but then the reduction, in turn, attempts to rewind the attacker. Our algorithm, therefore, needs to pretend to have been rewound and solve the dummy challenges, which again requires rewinding the reduction. Rewinding seems to complicate the above outline significantly, and indeed, Lewko and Waters only applied their techniques to straight-line reductions – ones that do not use rewinding. This still leaves hope that maybe, if we are clever enough about proof techniques, we can come up with a fully secure ABE scheme that is as simple as our existing selective schemes. In fact, perhaps it is even possible to prove adaptive security for [GVW13,BGG+14] using a sufficiently clever reduction.

## 1.1 Our Results

Our main result is to extend the result of Lewko and Waters to handle rewinding reductions. Along the way, we introduce a simpler notion of *checkability*. Therefore, our result shows that the Lewko-Waters argument cannot be circumvented by clever proof techniques (so long as the adversary is used in a black-box manner), and necessarily, there is a cost for the ability to prove adaptive security. Both in terms of naturalness and possibly performance.

Whereas Lewko and Waters considered security proofs that consisted of a simple black-box reduction to a non-interactive hardness assumption, we allow the underlying hardness assumption to be interactive and merely require that the number of communication rounds is a priori bounded by a fixed polynomial. This is a natural and essentially necessary requirement since otherwise we could take the adaptive security of the scheme to be the underlying hardness assumption, and trivially the adaptive security reduces to itself.

As an implication of our main theorem, we show that (the delegatable version of)[3] the celebrated lattice-based scheme of [BGG+14] cannot be proven adaptively secure, at least without modifications. To this end, we observe that the security definition of an ABE scheme does not involve the decryption algorithm. Indeed, an attacker receives keys and a challenge ciphertext and attempts to

[3] In the delegatable version, the secret key contains a lattice trapdoor rather than a single vector. See Sect. 1.2 for further explanation and discussion on the implications for the non-delegatable version.

recover the message that has been encrypted. The entire security game is conducted without any party being "instructed" to use the decryption algorithm.[4] We show that these schemes have an alternative decryption algorithm and that, with respect to this decryption algorithm, it is possible to validate secret keys and ciphertexts. Therefore, our result can be applied to rule out adaptive security reductions for this scheme. An important takeaway here is that in order to rule out validation, one must consider *all possible decryption circuits* (that decrypt correctly) and effectively show that it is impossible to validate secret keys and ciphertexts with respect to all of them. We then discuss some modifications to the aforementioned scheme and their impact on the ability to validate. Our conclusion is that apparently a more radical change, as per [Tsa19], may be required in order to be able to prove adaptive security in the lattice setting.

## 1.2 Technical Overview

To prove our main theorem, we consider a Turing reduction $\mathcal{R}$ from some intractability assumption $\mathcal{C}$ to violating the adaptive security of some ABE scheme. The assumption $\mathcal{C}$ is stated in terms of an interactive game (following the framework of [Nao03]). The reduction asserts that if there existed a successful ABE adversary $\mathcal{A}$, then $\mathcal{R}$ could use it in a black-box manner in order to break the assumption $\mathcal{C}$. Following [LW14], the underlying idea of our proof is to use the reduction $\mathcal{R}$ directly in order to break the assumption $\mathcal{C}$, without the help of an actual ABE attacker. In order to do so, we wish to efficiently emulate an ABE attacker for $\mathcal{R}$, in a way that $\mathcal{R}$ would not be able to distinguish from a real attacker.

Adaptive security is defined by an interactive game between the reduction $\mathcal{R}$, i.e. the challenger, and the attacker $\mathcal{A}$. The game begins with the challenger declaring the public parameters of the ABE scheme. Then, the attacker can make key queries to the challenger for secret keys of predicates of its choice. Next, it declares a challenge attribute $x^*$, and receives from the challenger a challenge ciphertext $\mathrm{CT}_{x^*}$ encrypted w.r.t. $x^*$ (say, encrypting either the message 0 or 1). Afterward, $\mathcal{A}$ can make more key queries until finally, it sends a guess of which message was encrypted in $\mathrm{CT}_{x^*}$. The adversary wins the game if it has not received any secret key that accepts $x^*$ and the guess has been correct. We say that an adversary breaks the security assumption if it wins the game with a noticeable advantage compared to randomly guessing.

Recall that in order to emulate an attacker for $\mathcal{R}$, our algorithm should be able to solve possible "dummy challenges" and properly decrypt ciphertexts provided by the reduction. Naively, this could be achieved by rewinding the reduction and extracting additional secret keys. However, as we explained above, two issues arise: (1) First, the reduction may notice when the simulated attacker decrypts the challenge ciphertext using a secret key that was extracted by rewinding the reduction, and then behave unexpectedly as a result,

[4] This property is not unique to ABE, and it also occurs in standard CPA secure encryption.

so we would not be able to use the reduction to violate $\mathcal{C}$. (2) Second, once we allow the reduction to rewind the attacker, the naive strategy could lead to many “nested” rewindings of the reduction and the attacker, which in turn may result in an exponential running time. Furthermore, when considering a general assumption $\mathcal{C}$, we have to ensure that rewinding the reduction does not affect the interaction between $\mathcal{R}$ and $\mathcal{C}$, since $\mathcal{C}$ itself is not in our control and we are unable to rewind it.

In [LW14], Lewko and Waters addressed the first obstacle and defined a *checkability* criterion for ABE schemes, which states that each secret key and each ciphertext can be validated to ensure “canonical” decryption. Namely, the checkability property requires that a valid ciphertext is decrypted to the same message using any valid secret key. This way, the simulated attacker can validate the challenge ciphertext and the secret key used for decryption, and the reduction would not detect which secret key was used. We present a simpler checkability requirement, which states that each secret key can be validated and that the decryption of ciphertext using any valid secret key results in the same decryption distribution. This property is sufficient to ensure consistent decryption by applying the same attacker strategy. Note that every scheme that satisfies Lewko and Waters's checkability requirement can be trivially transformed to satisfy our checkability requirement by validating the ciphertext at the beginning of the decryption procedure and outputting $\perp$ if found invalid. We further observe that the implementation of the decryption procedure of an ABE scheme does not play any role in the adaptive security proof. Therefore, if we modified the decryption procedure of an ABE scheme so it would satisfy our checkability property and show it cannot be proved adaptively secure, then we would conclude that the original scheme could not be proven adaptively secure as well.

To overcome the second obstacle, we adopt the more delicate rewinding technique introduced by Pass [Pas11] in the context of witness-hiding special-sound protocols for unique relations. Pass used the rewinding technique in order to obtain a sufficient amount of interaction transcripts, which are essentially proofs of some statement $x$ with different suffixes. The transcripts are then fed to the special-soundness extractor in order to recover a witness for $x$. We employ similar tools in order to rewind an ABE challenger, obtain multiple keys with respect to the same public parameters, and use them for the purpose of decrypting the challenge ciphertext.

Adopting Pass's terminology, we define the notion of a *slot*, a “time window” during the execution of $\mathcal{R}$. This window “opens” when the emulated ABE attacker makes a key query to obtain a secret key for some predicate function, and it “closes” right after the reduction responds with one and the attacker validates it. Intuitively, this is the shortest time frame that could be rewound in order to extract additional validated secret keys from the reduction. As previously described, due to the recursive nature of rewinding both the reduction and the simulated attacker, the reduction may rewind the attacker at some point during a slot, so another “nested” slot may open. We consider a slot to be “good” for rewinding if, between the time that it opens and the time that it closes, there is no communication between $\mathcal{R}$ and $\mathcal{C}$, and in addition, $\mathcal{R}$ does not rewind the attacker "too many times" within the slot. We also require the extracted key to be a valid one. Intuitively, the first condition ensures that rewinding the slot does not disturb the interaction between $\mathcal{R}$ and $\mathcal{C}$, and the second condition allows us to bound the recursion depth and limit the growth in runtime due to possible nested slots being rewound. Once we encounter a good slot in the execution, we rewind it several times to extract multiple secret keys. The precise criterion for a slot being good is determined with respect to the maximal running time of the reduction and the recursive depth of the slot. This, combined with the fact that $\mathcal{R}$ is an efficient algorithm and cannot make "too many" queries to the attacker, allows us to ensure the existence of good slots while maintaining the bound on the running time of the emulation. Note that an ABE attacker may query the challenger (embodied by the reduction in our case) many times. It suffices for us that just a fraction of the slots induced by these queries is good because once a slot is good, we can rewind it many times and obtain many secret keys that will allow us to decrypt the challenge ciphertext.

Being able to rewind the reduction is a necessary step toward simulating an attacker, but we must also argue that the rewound reduction will indeed provide us with a valid secret key that will decrypt $x^*$. To this end, we need to design the function class for which we make queries, as well as the challenge attribute $x^*$. We aim for these values to meet the following conditions: On the one hand, the secret keys queried during the "mainline" execution of $\mathcal{R}$ should not accept the selected challenge attribute $x^*$. On the other hand, to successfully decrypt the challenge, at least one of the keys from the rewound execution should accept $x^*$. Finally, we must meet these two conditions in a way such that the reduction would not be able to detect when it is being rewound; otherwise, it may abort.

We approach this challenge as follows: The challenge attribute $x^*$ is sampled uniformly at random from the set of possible attributes, and each secret key is sampled randomly from a specific set of functions, exactly the same way during the rewound queries as during the "mainline" ones. The set of functions is constructed from a pairwise independent hash family, so we expect the fraction of any subset of attributes covered by a uniformly random function from the set to be close to its expectation (a property also known as mixing). This is a key ingredient in our analysis to ensure any attribute has a high enough probability of being covered by the secret keys we extract by rewinding (this is true even when the reduction may choose not to disclose a fraction of the keys, as we explain shortly). The probability of a function in the set covering some attribute is chosen to be small enough so that with high enough probability, the uniformly random challenge $x^*$ is not covered by the functions queried during the "mainline" execution of $\mathcal{R}$, but large enough so that with high enough probability, $x^*$ is covered by at least one key that was extracted from a rewound execution. By specifying the functions to be queried, we require the ABE scheme to support them as predicates. Fortunately, since pairwise hash families can be implemented by $\mathrm{NC}^1$ circuits [IKOS08], the requirement is commonly satisfied.

We must pay delicate care when computing the probability of $x^*$ being covered by a secret key that was extracted by rewinding since we are not allowed to make any presumptions regarding the behavior of the reduction. In particular, $\mathcal{R}$ may refuse to provide secret keys for certain queries in an unexpected way. However, since $\mathcal{R}$ is a reduction from some hardness assumption $\mathcal{C}$ to an attacker, it should be impossible for $\mathcal{R}$ to violate the assumption $\mathcal{C}$ without the help of the attacker. In other words, the reduction must rely on the attacker breaking adaptive security, so it must cooperate and "play" according to the security game protocol with a high enough probability. Otherwise, it would fail to violate $\mathcal{C}$ with a noticeable advantage. For the same reason, if the reduction would only accept a negligible portion of the possible challenge attributes, it would fail to successfully violate $\mathcal{C}$ with a noticeable advantage. This behavior is a generalization of a method known as *complexity leveraging*, in which the challenger guesses the challenge attribute $x^*$ in advance and rejects any other challenge. The method is used to upgrade a selectively secure scheme to be adaptively secure generically, but at the cost of exponential degradation in security, which, as explained, makes it irrelevant to our scenario.

*Implications to Lattice-Based ABE Schemes.* We now turn our attention to applying our result to the lattice-based ABE scheme of Boneh et al. [BGG+14]. This scheme supports policy functions represented by boolean circuits of a polynomially a-priori bounded depth (the depth bound is a parameter in the initialization of the scheme), and security is based on the hardness of the Learning with Errors problem (LWE) [Reg05].

We start by outlining the high-level structure of the scheme. In particular, we consider the version that supports key delegation (see discussion on other variants and on other lattice-based ABE schemes at the end of this outline). The public parameters are designed so that each input attribute $x$ is associated with a lattice $A_x$ (we do not define what a lattice is since this is a high-level overview, but we will explain all properties that are relevant to our outline). This lattice can be publicly derived from the public parameters of the scheme. A ciphertext with respect to $x$ consists of a noisy vector that is close to the lattice $A_x$. More explicitly, $A_x$ is represented as a matrix in $\mathbb{Z}_q^{n\times m}$, and the ciphertext consists of a vector of the form $c = sA_x + e \pmod q$, where $s$ is a uniform vector in $\mathbb{Z}_q^n$, and the noise $e$ is sampled from a distribution over short vectors. The message to be encrypted is then encoded by adding an "offset" to $c$, which depends on the message. Importantly, being able to recover $e$ from $c$ suffices in order to recover the message $m$, so the exact encoding procedure is immaterial for our purposes. In terms of key generation, every possible predicate function $f$ is also associated with a (public) lattice $A_f$. In this version of [BGG+14], the secret key for the predicate $f$ consists of a *trapdoor* for the lattice $A_f$. A lattice trapdoor can take many forms (that are interchangeable). The simplest one, perhaps, is a "short basis" for the so-called dual lattice. The details are immaterial, but the important property is that given a trapdoor $T_A$ for a lattice $A$ and given a vector $c = sA + e \pmod q$, where $\|e\| \leq B$ (for some parameter $B$ that relates to the "quality" of the trapdoor), it is possible to recover $e$ (and furthermore this $e$ is unique). We refer to this operation as *decoding* (due to the similarity of decoding in error correcting codes). Thus, secret keys allow decoding with respect to $A_f$, but ciphertexts are encoded with respect to different lattices $A_x$.

The ingenious component of the scheme is a mechanism that allows, for every $f, x$ s.t. $f(x) = 1$, to come up with a low-norm matrix $H = H_{f,x}$ s.t. $A_f = A_x H_{f,x}$ (we do not define what we mean by a norm of a matrix, one can think about its spectral norm, for example). This means that given $c = sA_x + e \pmod{q}$, it is possible to multiply by $H$ and obtain $c' = sA_f + e' \pmod{q}$, where $e'$ has a possibly higher norm than $e$, but the degradation of the norm is bounded and respective to the norm of $H$. This new $c'$ can be decoded using $T_{A_f}$ in order to decrypt ciphertexts. (We did not explain why recovering $e'$ suffices for decryption, but this can be done.)

To apply our theorem to this scheme, we require the following observation, which follows from the analysis of trapdoor properties in the literature, e.g. [MP12]. It can be shown that $H_{f,x}$ can also be used to translate a trapdoor for $A_f$ into a trapdoor for $A_x$, with somewhat lower quality.[5] This means that an alternative decryption procedure would be to deduce the trapdoor for $A_x$ and then decode the vector $c$ directly.

For our purposes, we wish to ensure that two different keys, with respect to $f_1$ or $f_2$, will decrypt all ciphertext in the same way. We, therefore, take the following strategy. Given a secret key for a function $f$, we first check that its "quality" as a trapdoor $T_{A_f}$ for $A_f$ is good enough. That is, the honest key generation is guaranteed to produce trapdoors of a certain quality, and when we are given a candidate trapdoor, we check that it is indeed of this quality. Then, given a vector $c$, we derive a trapdoor $T_{A_x}$ for $A_x$, and use it to decode $c$. This is the end of the "standard" decryption procedure, but our "validated" decryption has additional steps. Note that it is quite possible that $c$ is not a legitimate ciphertext at all, and perhaps the outcome $e$ that we obtained is "garbage" and does not actually describe the difference from a nearby lattice point. In such a case, different trapdoors could lead to different "garbage" which contradicts the consistency property we are interested in. Therefore, once we recover the noise vector $e$, we check whether $(c - e)$ is indeed of the form $sA_x$ (i.e., is a vector in the lattice $A_x$) and also that $\|e\|$ is sufficiently short (in this case, that its norm is consistent with the norm of a noise vector that is selected by the honest encryption procedure). If either check fails, we return $\perp$, otherwise, we return $e$ (or rather, the message $m$ that is induced by the value of $e$). The important observation is that any trapdoor for $A_x$ that has been derived by taking a trapdoor for some $A_f$ of the prescribed quality and converting it into a trapdoor for $A_x$ using $H_{f,x}$ should be able to decode "legal" values of $e$. Therefore, if we get $c$ with an "illegal" value of $e$, then we always output $\perp$, and if the value of $e$ is "legal", then it should be correctly decoded.

[5] This is a general property. If $A = BH$, and $H$ is short, then given $H$, a trapdoor for $A$ implies a trapdoor for $B$, with the "quality" of the trapdoor degrading respective to the norm of $H$.

This concludes our alternative decryption algorithm for the [BGG+14] scheme, for which every validated secret key will produce the same output on any given input ciphertext $c$. Therefore, by our main result, this scheme cannot have a black-box proof of security.

If we wish to construct an adaptively secure scheme, we need to eliminate the checkability property. One direction that comes to mind is to degenerate the [BGG+14] scheme so that it is no longer possible to obtain a trapdoor for $A_x$. We could try, for example, to change the secret keys of the scheme so that they no longer contain a trapdoor for $A_f$. Indeed, such a modification is possible, where the secret key for $f$ is modified to only contain a single short vector from the dual lattice rather than a basis. The non-delegatable version of the [BGG+14] scheme indeed works in this way. The predecessor of the [BGG+14] scheme, namely the [GVW13] scheme, also has a similar structure to the non-delegatable [BGG+14], where the secret key does not allow obtaining a full trapdoor for $A_x$, but only a partial trapdoor (a number of short vectors in the dual lattice, that do not form a basis).

In order for this approach to indeed allow an adaptive security proof, the scheme needs to have properties that seem quite implausible. In particular, it would still be possible to apply our result to obtain a barrier if, instead of querying just one $f$, the attacker can query many different $f$'s that all accept the same value $x$. This would allow obtaining a large number of short vectors in the dual lattice of $A_x$, thus obtaining a trapdoor for $x$, which would allow for canonical decryption as described above. Due to the convoluted structure of $H_{f,x}$, it is hard to prove that a full basis will be generated in this way, but it seems very implausible that this will not be the case. The situation that we are considering is querying the key generation process on multiple functions $f$ and finding out that on all of the $x$'s that we consider (except a negligible fraction), if we take the dual-lattice vectors for $A_x$ that are generated by all $f$'s that accept $x$, it holds that they all fall into a proper subspace and do not form a full rank set. Note that the degrees of freedom of the key generation are fairly limited, and the number of $x$'s we can consider is far greater than the number of vectors produced by the key generation. Still, coming up with proof is non-trivial, and we leave it as an open problem. Nevertheless, any attempt to prove adaptive security would require proving the opposite, which seems quite difficult.

Our conclusion, therefore (and others may draw their own), is that one has to deviate from the known methods in order to achieve adaptive security. Indeed, Tsabary's construction [Tsa19] achieves this, for a very limited function class, by significantly deviating from the above blueprint so as to make it impossible to validate the decryption process.

## 1.3 Paper Organization

Some preliminaries and notation, along with ABE-related definitions are provided in Sect. 2. Our lower bound argument appears in Sect. 3. The technical details of our lattice instantiation, which are outlined in the overview above, are provided Sect. 4.

## 2 Preliminaries

### 2.1 Basic Definitions

Let $n$ be a natural number. We denote by $1^n$ the unary expansion of $n$, that is, the concatenation of $n$ 1's. We also denote $[n] \overset{\text{def}}{=} \{1, \ldots, n\}$.

**Definition 21** *(Negligible Function). A function $f : \mathbb{N} \to \mathbb{R}_{\geq 0}$ is said to be negligible if for all $c$ there exists $N$ such that $f(n) < n^{-c}$ for all $n > N$. We denote by negl$(\cdot)$ a negligible function.*

**Definition 22** *(Computational Indistinguishability). Let $\{X_\lambda\}_{\lambda \in \mathbb{N}}$ and $\{Y_\lambda\}_{\lambda \in \mathbb{N}}$ be two distribution ensembles. We say they are computationally indistinguishable if for any probabilistic polynomial-time algorithm $A$, it holds that*

$$\left| \Pr_{x \leftarrow X_\lambda}[A(x) = 1] - \Pr_{x \leftarrow Y_\lambda}[A(x) = 1] \right| = \mathit{negl}(\lambda) \tag{1}$$

**Definition 23** *(Statistical Distance). Let $X$ and $Y$ be two random variables over a finite domain $\Omega$, we define their statistical distance by*

$$D(X, Y) \overset{\mathit{def}}{=} \frac{1}{2} \sum_{\omega \in \Omega} |\Pr[X = \omega] - \Pr[Y = \omega]| \tag{2}$$

**Definition 24** *(Statistical Indistinguishability). Let $\{X_\lambda\}_{\lambda \in \mathbb{N}}$ and $\{Y_\lambda\}_{\lambda \in \mathbb{N}}$ be two distribution ensembles over a finite domain $\Omega$. We say they are statistically indistinguishable if $D(X_\lambda, Y_\lambda) = \mathit{negl}(\lambda)$.*

**Definition 25** *(Pairwise-Independent Hash Functions). A family $H$ of functions $h : [n] \to [m]$ is called pairwise independent if for every $i, j \in [n]$ such that $i \neq j$ and every $x, y \in [m]$ it holds that*

$$\Pr_{h \overset{\$}{\leftarrow} H}[h(i) = x \wedge h(j) = y] = \frac{1}{m^2} \tag{3}$$

### 2.2 Algorithms

The following algorithms have various alternative definitions. We present the definitions relevant to our analysis and results here and assume familiarity with Turing machines.

**Definition 26** *(Probabilistic Algorithm). A probabilistic algorithm is a Turing machine that receives an auxiliary random tape as input.*

**Definition 27** *(Interactive Algorithm and Interactive Protocol). An interactive algorithm, or interactive machine, is a Turing machine that has two additional communication tapes, a read-only one and a write-only one. An interactive protocol consists of two interactive machines $\pi = (A, B)$ such that $A$'s write-only communication tape is $B$'s read-only communication tape, and $A$'s read-only communication tape is $B$'s write-only communication tape. The machines take turns (also called "rounds") in being active, each turn ends with the active machine either halting or sending a message to the other machine.*

Given a pair of interactive algorithms $A$ and $B$ that interact according to some interactive protocol on some common input $x$, we denote by $\langle A, B\rangle(x)$ the distribution of the output of $B$ after interacting with $A$ on the common input.

**Definition 28** *(Oracle Machine). An oracle machine is a Turing machine with access to another machine, called the oracle. The access is implemented using an additional tape, called oracle tape, and two special states, ASK and RESPONSE. The oracle machine may enter the ASK state, which invokes an execution of an oracle on the input received through the oracle tape. The contents of the oracle tape are then replaced with the output of the oracle, and the state is changed to RESPONSE. Let $A$ be an oracle machine and $B$ an oracle; we denote $A^B$ the execution of $A$ with oracle access to $B$.*

**Remark 21** *(Relation between Oracle Machines and Interactive Machines). Throughout the paper, we treat interactive machines and oracle machines similarly, according to the following equivalency: Consider an interactive protocol $(A, B)$. The execution of the machines interactively is equivalent to executing $A$ as an oracle machine with oracle access to a stateless version of machine $B$ (i.e., machine $B$ without a state register), where every oracle query contains the partial transcript of the interactive protocol, and the output of the oracle is $B$'s next message according to the interactive protocol. Similarly, given an oracle machine $A$ with oracle access to $B$, we can consider an interactive protocol $(A, B)$ that contains the oracle queries made by $A$ to $B$ and the corresponding responses.*

**Remark 22** *(Rewinding an Oracle). Consider an interactive protocol $(A, B)$ and the corresponding execution of oracle machine $A$ with oracle access to $B$. Machine $A$ can "rewind" $B$ by making a query to $B$ with input that is a strict prefix of the partial transcript of the interactive protocol, i.e., "rewind" $B$ to a previous state in the interaction.*

**Definition 29** *(Decision Problem). A decision problem is a problem that is solved by classifying inputs to an output which is either 0 or 1.*

**Definition 210** *(Black-box Reduction). A black-box reduction from a decision problem $A$ to a decision problem $B$ is a Turing machine that solves problem $A$ given oracle access to a machine that solves problem $B$.*

## 2.3 Intractability Assumption

In the following definition, we formally describe the notion of intractability assumption to model a problem that is assumed to be "hard to solve". The assumption $\mathcal{C}$ is stated in terms of an interactive game between a challenger and an adversary (following the framework of [Nao03]).

**Definition 211** *($r(\cdot)$-round Intractability Assumption with Threshold $t(\cdot)$). An $r(\cdot)$-round intractability assumption with threshold $t(\cdot)$ is an interactive probabilistic decision problem $\mathcal{C}$, called the challenger, that interacts with another* *algorithm $\mathcal{A}$, called the attacker, such that: (1) both algorithms take as input $1^\lambda$ where $\lambda$ is the security parameter, and (2) the interaction is a-priori bounded by $r(\lambda)$ rounds. We define the advantage of the attacker $\mathcal{A}$ with respect to $\mathcal{C}$ as*

$$Adv(\mathcal{A}) \overset{def}{=} \left|\Pr\left[\langle \mathcal{A},\mathcal{C}\rangle \left(1^\lambda\right) = 1\right] - t(\lambda)\right| \tag{4}$$

*$\mathcal{C}$ is associated with a computational assumption that states that for any polynomial-time attacker $\mathcal{A}$ there exists a negligible function $\mu(\cdot)$ such that for all $\lambda \in \mathbb{N}$*

$$Adv(\mathcal{A}) \leq \mu(\lambda) \tag{5}$$

*We say that a polynomial-time attacker $\mathcal{A}$ breaks the assumption $\mathcal{C}$ with non-negligible advantage $p$ if $Adv(\mathcal{A}) = p$.*

## 2.4 Attribute-Based Encryption

**Definition 212** *(Key-Policy Attribute-Based Encryption Scheme). Let $\mathcal{X}$ be a set of objects and $\mathcal{F}$ be a class of functions of the form $f : \mathcal{X} \to \{0,1\}$. A key-policy Attribute Encryption (KP-ABE) scheme for attribute set $\mathcal{X}$ and policy class $\mathcal{F}$ is a tuple of probabilistic polynomial-time algorithms $\mathcal{S}$ = (Setup, Encrypt, KeyGen, Decrypt) as follows:*

- ***Setup**$(1^\lambda) \to$ PP, MSK* *The setup algorithm takes the security parameter $\lambda$ as input. It outputs the public parameters PP of the scheme and a master secret key MSK.*
- ***Encrypt**$(x, M, PP) \to CT_x$* *The encryption algorithm takes in an attribute $x \in \mathcal{X}$, a message $M \in \{0,1\}$ and public parameters PP. It outputs a ciphertext $CT_x$, which is an encryption of $M$ under $x$. Assume w.l.o.g. that $CT_x$ contains $x$.*
- ***KeyGen**$(MSK, f, PP) \to SK_f$* *The key generation algorithm takes in the master secret key MSK, a policy $f \in \mathcal{F}$ and public parameters PP. It outputs a secret key $SK_f$ for $f$. Assume w.l.o.g. that $SK_f$ contains $f$.*
- ***Decrypt**$(CT_x, SK_f, PP) \to M$* *The decryption algorithm takes in a ciphertext $CT_x$, a secret $SK_f$ and public parameters PP. It outputs a message $M \in \{0,1\}$.*

**Remark 23.** *Another variant of ABE is Ciphertext-Policy ABE (CP-ABE), where ciphertexts are associated with policies and secret keys with attributes. The distinction is immaterial for our purposes, so we adopt the notation of KP-ABE scheme and simply refer to it as ABE for convenience.*

**Definition 213** *((Perfect) Correctness of KP-ABE). Let $\mathcal{S}$ = (Setup, Encrypt, KeyGen, Decrypt) be a key-policy ABE scheme for attribute set $\mathcal{X}$ and policy class $\mathcal{F}$. We say that $\mathcal{S}$ is (perfectly) correct if the following holds: Let (PP, MSK) = Setup$(1^\lambda)$, $f \in \mathcal{F}$, $x \in \mathcal{X}$ and $M \in \{0,1\}$ and suppose $f(x) = 1$. Denote $CT_x = Encrypt(x, M, PP)$ and $SK_f = KeyGen(MSK, f, PP)$. It holds that*

$$Decrypt(CT_x, SK_f, PP) = M \tag{6}$$

**Definition 214** *(Adaptive Security of KP-ABE). Let $\mathcal{S}$ = (Setup, Encrypt, KeyGen, Decrypt) be a key-policy ABE scheme for attribute set $\mathcal{X}$ and policy class $\mathcal{F}$. We define adaptive security to be the intractability assumption that consists of the following interactive "game":*

1. ***Setup Phase*** *The challenger runs Setup($1^\lambda$) and sends PP to the adversary.*
2. ***Key Query Phase I*** *The adversary makes key queries for policies in $\mathcal{F}$ of her choice: that is, in each key query, the adversary sends a policy of her choice to the challenger, the challenger runs the KeyGen algorithm to produce a secret key and sends it to the adversary.*
3. ***Challenge Phase*** *The adversary declares two messages $M_0, M_1$ and a challenge attribute $x^* \in \mathcal{X}$. The challenger samples a uniformly random bit $b \in \{0,1\}$, computes $CT_{x^*} = Encrypt(x^*, M_b, PP)$ and sends the result to the adversary.*
4. ***Key Query Phase II*** *Same as Key Query Phase I.*
5. ***Guess*** *The adversary sends a guess $b'$ for the bit $b$.*

*The output of the game is defined as follows: If every queried policy $f$ satisfies $f(x^*) = 0$, the output is the adversary's guess $\tilde{b} = b'$; otherwise, the output is a uniformly random bit $\tilde{b} \xleftarrow{\$} \{0,1\}$. The threshold of the intractability assumption is $1/2$, so the advantage of the adversary is $\left|\Pr[\tilde{b} = b] - 1/2\right|$.*

**Remark 24.** *We observe that the adaptive security of an ABE scheme depends only on the Setup, KeyGen, and Encrypt procedures and not on the Decrypt procedure. Therefore, any two ABE schemes that differ only in their implementations of the decryption must either both be adaptively secure or both not.*

Next, we formally define a condition on an ABE scheme that essentially requires the scheme to support a set of policies derived from a pairwise-independent hash family. Intuitively, a random sample of policies from such a set has a significant probability of accepting a uniformly random attribute. This property will be used in our proof to ensure that an attacker that requests secret keys for many policies in that set can use them to decrypt a challenge ciphertext with a large enough probability.

**Definition 215** *(Pairwise-Friendliness of ABE). Let $\mathcal{S}$ be an ABE scheme for attribute set $\mathcal{X}$ and policy class $\mathcal{F}$. We say that $\mathcal{S}$ is pairwise friendly if for every $a, b \in \mathbb{N}$ such that $a > b$ and $a = O(\log|\mathcal{X}|)$, there exists a pairwise independent hash family $H = \{h : \mathcal{X} \to [a]\}$ so that the following holds: For every $h \in H$, the function $f_h : \mathcal{X} \to \{0,1\}$ defined by*

$$f_h(x) = 1 \iff h(x) \leq b \tag{7}$$

*is in $\mathcal{F}$.*

**Remark 25.** *For every $n \in \mathbb{N}$ and $m \leq n$ there exists a pairwise independent hash family $H = \{h : \{0,1\}^n \to \{0,1\}^m\}$ such that every $h \in H$ can be computed by an $NC^1$ circuit [IKOS08].*

**Remark 26.** *Following the previous remark, let $\mathcal{S}$ be an ABE scheme with attribute set $\mathcal{X}$ and policy class $\mathcal{F}$ such that $\mathcal{F}$ contains the class of functions with depth-d circuits for $d = O(\log|\mathcal{X}|)$, then $\mathcal{S}$ is pairwise friendly.*

The following definitions describe our checkability criterion for an ABE scheme. Intuitively, the checkability condition requires a validation procedure for secret keys, which ensures that validated keys decrypt any ciphertext the same. In particular, it makes it impossible for a challenger to distinguish which key was used to decrypt a challenge ciphertext. More precisely, checkability correctness requires that honestly generated keys are indeed valid (according to the validation procedure), and checkability soundness requires that any two valid keys decrypt any ciphertext the same way.

**Definition 216** *(Checkability of ABE). We say that an ABE scheme is "checkable" if it has an additional algorithm:*

- ***SKCheck***$(PP, SK, f) \rightarrow \{True, False\}$ *The ciphertext checking algorithm takes in public parameters PP, a key SK and a policy $f \in \mathcal{F}$. It outputs either True or False.*

**Definition 217** *(Checkability Correctness of KP-ABE). Let $\mathcal{S}$ = (Setup, Encrypt, KeyGen, Decrypt, SKCheck) be a KP-ABE scheme for attribute set $\mathcal{X}$ and policy class $\mathcal{F}$. We say that $\mathcal{S}$ satisfies the checkability-correctness property if the following holds: Let (PP, MSK) = Setup$(1^\lambda)$, $f \in \mathcal{F}$, and $SK_f = KeyGen(MSK, f, PP)$, then $SKCheck(PP, SK_f, f) = True$.*

**Definition 218** *(Checkability Soundness of KP-ABE). Let $\mathcal{S}$ = (Setup, Encrypt, KeyGen, Decrypt, SKCheck) be a KP-ABE scheme for attribute set $\mathcal{X}$ and policy class $\mathcal{F}$. We say that $\mathcal{S}$ satisfies the checkability-soundness property if the following holds: Let PP, $SK_1, SK_2$ and $f_1, f_2 \in \mathcal{F}$. Let $CT_x$, i.e. a ciphertext claimed to be encrypted w.r.t. attribute $x \in \mathcal{X}$,[6] s.t. $f_1(x) = f_2(x) = 1$. If $SKCheck(PP, SK_1, f_1) = SKCheck(PP, SK_2, f_2) = True$, then*

$$Decrypt(CT_x, SK_1, PP) = Decrypt(CT_x, SK_2, PP) \tag{8}$$

**Remark 27.** *To obtain our result, it suffices to assume a looser condition on a checkable ABE scheme: Instead of requiring that any two policy keys decrypt the ciphertext exactly the same, it suffices to require that for any two policy keys, the distributions of the decryption outputs are computationally indistinguishable.*

## 3 The Main Theorem and Proof

Informally, our main theorem asserts the following: Any pairwise-friendly, checkable ABE scheme cannot be proven adaptively secure by constructing a black-box reduction that reduces an intractability assumption to breaking the security of

[6] Recall our convention (Definition 212) that ciphertexts contain their attribute as a part of their description.

the scheme, even if the reduction is rewinding. We prove that if such a reduction existed, it would be possible to construct an efficient algorithm that violates the intractability assumption by using only the reduction, without requiring a successful ABE adversary, thereby leading to a contradiction.

**Theorem 31.** *Let $\lambda$ a security parameter and $n = poly(\lambda)$. Let $\mathcal{S}$ be a pairwise friendly and checkable ABE scheme with attribute set $\mathcal{X} = \{0,1\}^n$ and policy class $\mathcal{F}$. Let $\mathcal{C}$ be an $r(\cdot)$-round intractability assumption with threshold $t(\cdot)$, where $r, t$ are polynomials. Suppose that for every polynomial $l(\cdot)$ there exists a black-box reduction $\mathcal{R}$ such that the following holds: If $\mathcal{R}$ is given oracle access to an attacker $\mathcal{A}$ that makes $l(\cdot)$ key queries and has a non-negligible advantage in the adaptive security game of $\mathcal{S}$, then $\mathcal{R}^{\mathcal{A}}$ has non-negligible advantage w.r.t. the assumption $\mathcal{C}$.*

*Let $l(\lambda) = \omega(n(\lambda)+r(\lambda))$ and a corresponding reduction $\mathcal{R}$. Denote $\mathcal{A}$ to be a hypothetical attacker that has a non-negligible advantage in the adaptive security game of $\mathcal{S}$. Then there exists a polynomial-time algorithm $\mathcal{B}$ and a negligible function $\mu(\cdot)$ such that*

$$\left|\Pr\left[\langle \mathcal{R}^{\mathcal{A}}, \mathcal{C}\rangle\left(1^{\lambda}\right) = 1\right] - \Pr\left[\langle \mathcal{B}^{\mathcal{R}}, \mathcal{C}\rangle\left(1^{\lambda}\right) = 1\right]\right| \leq \mu(\lambda) \qquad (9)$$

*In particular, $\mathcal{B}^{\mathcal{R}}$ is a polynomial-time algorithm that has a non-negligible advantage w.r.t. the assumption $\mathcal{C}$.*

In the remainder of the section, we prove the theorem:

*Proof.* Let $\lambda$ denote the security parameter and suppose there exist $\mathcal{S}$ and $\mathcal{C}$ as described in the theorem. Let $l(\lambda) = \omega(n(\lambda) + r(\lambda))$ and let $\mathcal{R}$ be the corresponding reduction as described in the theorem. By definition of $\mathcal{R}$, if there were an attacker $\mathcal{A}$ with non-negligible advantage in the security game of $\mathcal{S}$, and $\mathcal{R}$ would be given oracle access to $\mathcal{A}$, then $\mathcal{R}$ could use $\mathcal{A}$ during an interaction with $\mathcal{C}$ to gain non-negligible advantage w.r.t. $\mathcal{C}$. We will prove that the oracle access to $\mathcal{A}$ can be simulated in a way that preserves $\mathcal{R}$'s advantage w.r.t. $\mathcal{C}$, even if it does not have oracle access to an actual attacker $\mathcal{A}$.

More explicitly, we will construct a machine $\mathcal{B}$ that has oracle access to the reduction $\mathcal{R}$ and simulates the interaction between $\mathcal{C}$ and $\mathcal{R}^{\mathcal{A}}$ for a properly defined $\mathcal{A}$ so that $\mathcal{B}$ has non-negligible advantage w.r.t. $\mathcal{C}$. To simulate the interaction accurately, $\mathcal{B}$ emulates an oracle access to an attacker $\mathcal{A}$ for $\mathcal{R}$, making it appear as if $\mathcal{A}$ has a non-negligible advantage in the adaptive security game of $\mathcal{S}$. Furthermore, the expected running time of $\mathcal{B}$ is polynomial, and in particular, the emulation of the attacker is efficient.

First, we introduce an inefficient hypothetical attacker $\mathcal{A}$ with a non-negligible advantage in the adaptive security game of $\mathcal{S}$. $\mathcal{A}$ can be used by $\mathcal{R}$ so that $\mathcal{R}^{\mathcal{A}}$ has non-negligible advantage w.r.t. the assumption $\mathcal{C}$. Second, we describe the algorithm $\mathcal{B}$ that has oracle access to the reduction $\mathcal{R}$, but instead of giving $\mathcal{R}$ an oracle access to an actual attacker, it simulates an attacker for $\mathcal{R}$ that is indistinguishable from $\mathcal{A}$ by $\mathcal{R}$. As we will show, $\mathcal{B}$ simulates $\mathcal{A}$ by rewinding the reduction $\mathcal{R}$ and exploiting the fact that $\mathcal{R}$ runs in polynomial time and cannot make too many queries to $\mathcal{A}$. Finally, we analyze the running time and advantage of $\mathcal{B}^{\mathcal{R}}$ to prove that there exists a polynomial-time algorithm that breaks the assumption $\mathcal{C}$ with non-negligible advantage.

To summarize notations:

| Parameter | Size | Meaning |
|---|---|---|
| $n$ | $n(\lambda) = \text{poly}(\lambda)$ | The dimension of $\mathcal{X} = \{0,1\}^n$, the attribute set of $\mathcal{S}$ on input $1^\lambda$. |
| $r$ | $r(\lambda) = \text{poly}(\lambda)$ | The number of communication rounds in the intractability assumption $\mathcal{C}$ on input $1^\lambda$. |
| $t$ | $t(\lambda) = \text{poly}(\lambda)$ | The threshold associated with the intractability assumption $\mathcal{C}$ on input $1^\lambda$. |
| $l$ | $l(\lambda) = \omega(n(\lambda) + r(\lambda))$ | The number of key queries made by the attacker $\mathcal{A}$ on input $1^\lambda$. |
| $M$ | $M(\lambda) = \text{poly}(\lambda)$ | The bound on the running time of $\mathcal{R}$ on input $1^\lambda$. |
| $m$ | $m(\lambda) > 8l(\lambda)$ | A parameter of $\mathcal{B}$ to be defined later. |
| $k$ | $k(\lambda) = \omega(l(\lambda)\log\lambda)$ | A parameter of $\mathcal{B}$ to be defined later. |

Recall that $\mathcal{S}$ is pairwise friendly, and let $H = \{h : \{0,1\}^n \to [m]\}$ be the pairwise independent hash family such that for every $h \in H$, the function $f_h$ defined by $f_h(x) = 1 \iff h(x) \leq \frac{m}{8l}$ is in $\mathcal{F}$.

### 3.1 Hypothetical Attacker $\mathcal{A}$

The hypothetical attacker algorithm performs as follows:

```
Algorithm $\mathcal{A}(1^\lambda)$
1. Receive PP from the challenger.
2. Initialize $F = \emptyset$ the set of functions to be queried during the key query
   phase and their corresponding keys.
3. Make $l$ key queries, for each one:
   (a) Sample a uniformly random $h \xleftarrow{\$} H$ and make a key query by sending
       the policy $f \overset{\text{def}}{=} f_h$ to the challenger.
   (b) Receive $\text{SK}_f$ and update $F \leftarrow F \cup \{(f, \text{SK}_f)\}$.
   (c) Run SKCheck(PP, $\text{SK}_f$, $f$), if the output is False then abort.
```

4. Declare two messages $M_0 = 0$ and $M_1 = 1$. Sample a uniformly random challenge attribute $x^* \xleftarrow{\$} \{0,1\}^n$ and send it to the challenger. Receive a ciphertext $\mathrm{CT}_{x^*}$.
5. Brute-force search for $h' \in H$ such that $f_{h'}(x^*) = 1$. If no such $h'$ exists, then abort.
6. Iterate over all possible secret keys and check for every key SK if $\mathrm{SKCheck}(\mathrm{PP}, \mathrm{SK}, f_{h'}) = \mathrm{True}$. If so, stop the iteration. If no such key was found, then abort.
7. Decrypt $\mathrm{CT}_{x^*}$ using SK and return the result.

Similarly to [Pas11], we formally equip $\mathcal{A}$ with access to a random oracle which is used to generate the random coins used by $\mathcal{A}$. This will allow us to consider rewindings of $\mathcal{R}$ more conveniently. We may think of $\mathcal{A}$ as a distribution over attackers, each defined by an instantiation of the random oracle.

*Success Probability.* We show that $\mathcal{A}$ breaks the adaptive security of $\mathcal{S}$:

**Claim 31.** *$\mathcal{A}$ wins the adaptive security game of $\mathcal{S}$ with advantage $\geq \frac{7}{16}$.*

*Proof.* The proof is available in the eprint version of the paper: https://eprint.iacr.org/2023/952

## 3.2 Algorithm $\mathcal{B}$

*Overview.* Recall that we wish to construct an algorithm $\mathcal{B}$ that simulates oracle access to an attacker $\mathcal{A}$ for $\mathcal{R}$ without having access to an actual attacker. In such a simulation, $\mathcal{R}$ initiates an interaction with the attacker by first sending the public parameters; then they interact according to the adaptive security game of $\mathcal{S}$ until finally, the attacker sends a guess for which message was encrypted in the challenge ciphertext. Naively, $\mathcal{B}$ could simulate the attacker by rewinding $\mathcal{R}$ to a previous state of the interaction, extracting an additional secret key, and using it to decrypt the challenge ciphertext. We emphasize that the extracted secret key should be validated by the attacker using the SKCheck procedure to ensure canonical decryption, as guaranteed by the checkability soundness of the scheme. A major problem with this naive approach is that $\mathcal{R}$ can make "intertwined" queries to $\mathcal{A}$ for many different public parameters. Therefore, if $\mathcal{B}$ would rewind $\mathcal{R}$ whenever it would be required to decrypt a challenge ciphertext, we could get an exponential blow-up in running time[7]. We resolve this by having $\mathcal{B}$ rewind the reduction $\mathcal{R}$ only under certain conditions and have a more delicate rewinding process, as we will describe shortly.

Before diving into a formal description of the rewinding process, we provide a high-level intuition. Consider the tape of the Turing machine $\mathcal{B}^{\mathcal{R}}$, i.e., the execution of $\mathcal{B}$ given oracle access to $\mathcal{R}$. At a high level, $\mathcal{B}$ initiates an execution of $\mathcal{R}$, forwards all communication between $\mathcal{C}$ and $\mathcal{R}$, and whenever $\mathcal{R}$ tries to access the attacker oracle, $\mathcal{B}$ emulates the attacker for $\mathcal{R}$. Under certain conditions to

[7] A similar problem was presented in the context of concurrent zero-knowledge in [DNS04].

be specified later, $\mathcal{B}$ "forks" the execution into two parallel executions, a process which could be thought of as "duplicating" the machine tape, and rewinds $\mathcal{R}$ in the forked execution to a previous state. In other words, $\mathcal{B}$ makes a copy of the current state of execution and then rewinds the copied execution so that it would continue differently from the original one. We think of the relation between the original execution and the duplicated one as "parent" and "child", respectively. $\mathcal{B}$ then continues running the child execution until a certain event occurs, when it terminates it (and all its child executions if they exist), and finally continues the parent execution.

More precisely, taking inspiration from Pass's work [Pas11], we define the notion of a "slot", denoted by $s$, to be a time window within the execution of $\mathcal{R}$ that "opens" just before the simulated attacker sends a policy to the reduction, and "closes" right after the reduction sends back a corresponding secret key and the simulated attacker validates it using SKCheck. Whenever a slot closes, $\mathcal{B}$ decides whether to rewind $\mathcal{R}$ back to the opening of the slot, depending on three conditions that determine if the slot is "good":

1. Between the time the slot $s$ opened and the time it closed, $\mathcal{R}$ did not send (and thus did not receive) any external message to (or from) $\mathcal{C}$.
2. Between the time the slot $s$ opened and the time it closed, the number of other slots that opened is "small", where "small" will be defined below.
3. The received key is valid, i.e., the result of SKCheck was True.

Whenever such a slot $s$ closes, $\mathcal{B}$ "duplicates" the execution, rewinds $\mathcal{R}$ in the duplicated execution back to the opening of the slot, and sends a different policy than the one sent in the original execution. $\mathcal{B}$ runs the duplicated execution until the slot either closes or stops being "good", and then terminates it (along with all its child executions if they exist). $\mathcal{B}$ repeats this process of duplicating the execution and rewinding the slot several times to extract several keys until finally, $\mathcal{B}$ returns to the original execution and continues running it. We highlight that the rewinding process could be recursive – recall that $\mathcal{R}$ might make intertwined queries, thus, there might be a slot that opens and closes within another slot.

Next, we describe this procedure formally.

*The Algorithm.* We will use the notion of a machine state, or simply a state, to formally describe the control flow of the algorithm. Intuitively, we could think of a state as a pointer to the machine tape of $\mathcal{B}$. W.l.o.g., we assume a state includes a record of all the messages sent and received by $\mathcal{C}$, $\mathcal{B}$ and $\mathcal{R}$, up to the point that the state points to.

We define a state $v$ to be $d$-good with respect to a previous state $u$ if: (1) $\mathcal{R}$ does not attempt to send messages to $\mathcal{C}$ during the time between $u$ and $v$, and (2) the number of slots that open between $u$ and $v$ is at most $\frac{M}{n^d}$. We define a slot to be the time window whose opening is a state right before $\mathcal{B}$ makes sends key-query, and closing is a state right after the respective key that was received is checked using SKCheck. We observe that any slot can be specified by its opening state, and the opening defines a distribution over the possible closings of the slot (depending on the key that was queried and the respective response). An execution-instance of a slot is a pair $(u, v)$ where $u$ is the slot-opening and $v$ is a specific slot-closing. We say that an execution-instance $(u, v)$ of slot $s$ is $d$-good if the closing of $s$, $v$, is $d$-good with respect to its opening, $u$, and the result of SKCheck at the end of the slot is True.

The algorithm we describe is $\mathcal{B}^{\mathcal{R}}$, that is, the algorithm $\mathcal{B}$ given oracle access to the reduction $\mathcal{R}$. The formal description uses the definition of the hypothetical attacker $\mathcal{A}$ and a recursive procedure SIM that simulates the attacker oracle for $\mathcal{R}$.

Recall that the interactive protocol between the reduction and the attacker oracle begins with the reduction sending public parameters to the attacker. Since $\mathcal{R}$ can make queries to the attacker that correspond to intertwined interaction transcripts, we associate each message with the public parameters that initiated the corresponding transcript. Similarly, we associate each slot with the public parameters that initiated the transcript that contains the policy message that "opened" the slot.

```
Algorithm $\mathcal{B}^{\mathcal{R}}(1^\lambda)$
1. Initialize a global set $\tilde{F} = \emptyset$.
2. Receive a message from $\mathcal{C}$.
3. Initiate an execution of the reduction oracle $\mathcal{R}$, and send the message
   received from $\mathcal{C}$ to $\mathcal{R}$.
4. Run $\mathrm{SIM}^{\mathcal{R}}(1^\lambda, 0, 0, 0)$.
```

```
Algorithm $\mathrm{SIM}^{\mathcal{R}}(1^\lambda, d, u, v)$
On input the recursive depth $d$, a state $u$, and a state $v$:
1. Check for the following mutually exclusive conditions and perform
   accordingly:
   (a) If $d = 0$ and $\mathcal{R}$ attempts to send a message to $\mathcal{C}$, forward the message
       and feed $\mathcal{R}$ the response received from $\mathcal{C}$. Note that only at recursive
       depth $d = 0$ the reduction $\mathcal{R}$ can interact with $\mathcal{C}$.
   (b) If $d > 0$ and $v$ is not a $d$-good state with respect to $u$, return $\perp$.
   (c) If $d > 0$, $u$ is an opening of a slot $s$ that closes at $v$, and $(u, v)$ is
       a $d$-good execution-instance of $s$, then return $(\mathrm{PP}, f, \mathrm{SK}_f)$ where $f$
       and $\mathrm{SK}_f$ are the policy and secret key retrieved during the slot $s$ and
       PP is the corresponding public parameters.
   (d) If $v$ is the closing of a slot $\tilde{s}$ that opened at state $\tilde{u}$ which is strictly
       after $u$, and $(\tilde{u}, v)$ is a $(d + 1)$-good execution-instance of $\tilde{s}$: Repeat
       the following $k(n)$ times:
         i. Let $r = \mathrm{SIM}(1^\lambda, d + 1, \tilde{u}, \tilde{u})$. emphasize that this is where
            the rewinding occurs – another execution of SIM is forked and
            rewound to the previous state $\tilde{u}$.)
        ii. If $r \neq \perp$, store $r = (\mathrm{PP}, f, \mathrm{SK}_f)$ in $\tilde{F}$.
2. If none of the above conditions were satisfied, check if $\mathcal{R}$ is sending a
   message to the attacker:
   (a) If $\mathcal{R}$ is sending a challenge ciphertext, do nothing and continue.
```

   (b) If $\mathcal{R}$ is sending a secret key SK as a response to a key query for policy $f$ with respect to public parameters PP:
      i. Run SKCheck(PP, SK, $f$).
      ii. If the output is False and $d = 0$ then abort. If the output is False and $d > 0$ then return $\perp$.
3. If none of the conditions of (1) and (2) were satisfied, check if according to the interactive protocol between $\mathcal{R}$ and the hypothetical attacker $\mathcal{A}$, $\mathcal{R}$ is expecting to receive a response message from the attacker:
   (a) If $\mathcal{R}$ is expecting to receive a decryption of a ciphertext associated with public parameters PP, then perform as follows: Let $\mathrm{CT}_{x^*}$ be the ciphertext and $x^*$ the challenge under which the ciphertext was encrypted;
      i. Search $\tilde{F}$ for a tuple that has the same PP and also satisfies $f(x^*) = 1$.
      ii. If there exists such a tuple, use $\mathrm{SK}_f$ to decrypt $\mathrm{CT}_{x^*}$ and send the result.
      iii. Otherwise, send a uniformly random guess.
   (b) If $\mathcal{R}$ is expecting to receive a policy, then respond as the attacker oracle would; that is, sample a uniformly random $h \xleftarrow{\$} H$ and send $f_h$.
   (c) If $\mathcal{R}$ is expecting to receive a $M_0$, $M_1$ and a challenge attribute, then respond as the hypothetical attacker oracle would; that is,
      i. Declare two messages $M_0 = 0$ and $M_1 = 1$.
      ii. Sample a uniformly random $x^* \xleftarrow{\$} \{0,1\}^n$ and send it to $\mathcal{R}$.
4. Update $v$ to be the current state (that includes all messages up to the current point) and return $\mathrm{SIM}^{\mathcal{R}}\left(1^\lambda, d, u, v\right)$.

## 3.3 Running Time

Consider a variant of $\mathcal{B}$, $\tilde{\mathcal{B}}$, such that whenever $\tilde{\mathcal{B}}$ is required to decrypt a challenge ciphertext without having retrieved a suitable secret key, $\tilde{\mathcal{B}}$ magically gets a key that decrypts the ciphertext (we can think of this as brute-force searching for a key in the same manner as the hypothetical attacker, but without counting the brute-force search into the runtime). We note that this change does not change the runtime of the algorithm. We observe that $\tilde{\mathcal{B}}$ always responds the same as the hypothetical attacker $\mathcal{A}$.

**Lemma 31.** *There exists some polynomial $p(\cdot)$ such that the running time of $\tilde{\mathcal{B}}^{\mathcal{R}}\left(1^\lambda\right)$ is bounded by $p(\lambda)$.*

*Proof.* We use a recursion tree to describe how $\tilde{\mathcal{B}}$ executes and rewinds $\mathcal{R}$: The root of the tree is the initial (and single) execution of $\mathcal{R}$ at level $d = 0$. Whenever $\tilde{\mathcal{B}}$ forks a child execution at recursive level $d$, that is, what we previously described as "duplicating the tape", it is translated into a new node at level $d + 1$ which is a child of the node from which it was forked from at level $d$.

As previously described, this occurs whenever $\tilde{\mathcal{B}}$ encounters a $(d+1)$-good slot instance (i.e., a $(d+1)$-good execution-instance of a slot) during an execution at recursive level $d$, which $\tilde{\mathcal{B}}$ then rewinds in the forked execution.

First, by definition of the algorithm $\tilde{\mathcal{B}}$, the depth of the recursion tree is bounded by a constant $c = \log_\lambda M \cdot \theta(1)$. Second, at each execution of $\mathcal{R}$ at recursive level $d$, $\mathcal{R}$ opens at most $M$ slots, so there are at most $M$ points from which $\tilde{\mathcal{B}}$ may fork child executions. Third, also by definition of $\mathcal{B}$, each slot is forked and rewound $k$ times. Combining these observations, we get that each execution node in the recursion tree has at most $M$ slots that have $k$ children each, so overall a maximal number of $Mk$ child executions. Therefore, the overall size of the recursion tree is bounded by $(Mk)^{c+1}$, thus the runtime of $\tilde{\mathcal{B}}^{\mathcal{R}}$ is bounded by a polynomial of $\lambda$ as well.

To conclude a bound on the expected running time of $\mathcal{B}$ from the bound on the running time of $\tilde{\mathcal{B}}$, we first make the following assumption: Suppose that the probability of $\mathcal{B}$ being required to decrypt a ciphertext without having retrieved a suitable key during an execution of $\mathcal{B}^{\mathcal{R}}$ is bounded by a negligible function of $\lambda$. Given this assumption, the transcript of the interaction between $\mathcal{B}^{\mathcal{R}}$ with $\mathcal{C}$ is indistinguishable from the transcript of the interaction between $\tilde{\mathcal{B}}^{\mathcal{R}}$ with $\mathcal{C}$. Therefore, under this assumption, the expected running time of $\mathcal{B}$ is bounded by a polynomial as well. The assumption is proven in the next section, in which we analyze the success probability of $\mathcal{B}$ breaking the intractability assumption $\mathcal{C}$.

## 3.4 Success Probability

In order to analyze the success probability of $\mathcal{B}^{\mathcal{R}}$, we compare the transcript of the interaction between $\mathcal{C}$ and $\mathcal{B}^{\mathcal{R}}$ with the transcript of the interaction between $\mathcal{C}$ and $\mathcal{R}^{\mathcal{A}}$. We show that the distributions of those transcripts are indistinguishable, therefore, the probability that $\mathcal{C}$ outputs 1 is the same in both scenarios except for some negligible probability. In other words, we show that there exists some negligible function $\mu(\cdot)$ such that

$$\Pr\left[\langle \mathcal{B}^{\mathcal{R}}, \mathcal{C}\rangle\left(1^\lambda\right) = 1\right] \geq \Pr\left[\langle \mathcal{R}^{\mathcal{A}}, \mathcal{C}\rangle\left(1^\lambda\right) = 1\right] - \mu(\lambda) \tag{10}$$

By the definition of $\mathcal{B}$, it forwards all messages from $\mathcal{C}$ to the (single) execution of $\mathcal{R}$ at recursive level $d = 0$ and vice versa. For simplicity, denote the execution of $\mathcal{R}$ at level $d = 0$ by $\mathcal{R}_0$. If $\mathcal{B}$ would perfectly simulate the hypothetical attacker $\mathcal{A}$ for $\mathcal{R}$, that is, respond to all queries made by $\mathcal{R}$ to the attacker oracle exactly the same as $\mathcal{A}$ would, then $\mathcal{R}_0$ would behave exactly the same as $\mathcal{R}^{\mathcal{A}}$ (all other recursive calls at level $d > 0$ would be irrelevant to $\mathcal{R}_0$), leading to the transcripts in both scenarios having exactly the same distribution. Although this is not necessarily the case, we show that the probability that $\mathcal{B}$ responds differently from $\mathcal{A}$ is negligible, so even though the transcripts are not identically distributed, they are indeed indistinguishable.

By the definitions of the hypothetical attacker $\mathcal{A}$ and the algorithm $\mathcal{B}$, they respond exactly the same to all queries that $\mathcal{R}$ makes to the attacker oracle, with the exception of when $\mathcal{R}$ requires the attacker oracle to provide a guess for which message was encrypted in the challenge ciphertext. When that happens, then the hypothetical attacker should be able to decrypt by brute-force searching for a decrypting secret key, whereas the simulated attacker might fail to obtain such a key and respond differently than $\mathcal{A}$. We argue that the probability that $\mathcal{B}$ fails to decrypt a ciphertext is negligible.

Denote:

- $E_1$ the event that $\mathcal{B}$ is required to decrypt a ciphertext $\mathrm{CT}_x$ associated with some public parameters PP without having previously rewound at least $n$ slots associated with the same PP.
- $E_2$ the event that $\mathcal{B}$ is required to decrypt a ciphertext $\mathrm{CT}_x$ associated with some public parameters PP, at least $n$ slots associated with the same PP were successfully rewound, but every $(\mathrm{PP}, f, \mathrm{SK}_f) \in \tilde{F}$ (with the same PP) satisfies $f(x) = 0$.

If neither $E_1$ nor $E_2$ occur, then $\mathcal{B}$ necessarily obtains a decrypting key and successfully decrypts the challenge ciphertext.

**Claim 32.** *The probability that $E_1$ happens in an execution between $\mathcal{B}^{\mathcal{R}}$ and $\mathcal{C}$ is 0.*

*Proof.* The proof of this claim follows similar guidelines as presented in [Pas11]. We first observe that every time an instance of a game between $\mathcal{R}$ and the simulated attacker reaches the challenge phase, and $\mathcal{R}$ sends a challenge ciphertext associated with public parameters PP, then it must have sent $l(\lambda) = \omega(n + r) = \omega(2n + r + 1)$ secret keys[8] associated with the same PP (i.e., as part of the same game). Moreover, those keys must have been valid ones, otherwise, the simulated attacker would have stopped responding and never reached the guessing phase.

Consider an occurrence of $\mathcal{B}$ being required to send a guess as part of a game associated with public parameters PP, and the $l$ slots associated with the same PP. These slots may be distributed over several nodes in the recursion tree. Since the recursive depth of the simulation is bounded by a constant $c$, there must be some recursion level $d$ such that the number of these slots at level $d$ is at least $\frac{l}{c}$. For sufficiently large $\lambda$, $\frac{l}{c} \geq 2n + r + 1$. Since $r$ bounds the total number of external messages, we conclude that in at least $2n + 1$ of the slots, there is no communication between $\mathcal{R}$ and $\mathcal{C}$. By the design of the rewinding process, the number of slot openings during any slot $s$ at level $d$ is bounded by $\frac{M}{n^d}$, so there must be at least $n$ of the $l/c$ slots that have no more than $\frac{M}{n^{d+1}}$ inner slot openings. We conclude that at least $n$ of the slots associated with PP are $(d + 1)$-good, and thus can be rewound, as desired.

**Claim 33.** *There exists some negligible function $\mu_2(\cdot)$ such that the probability that $E_2$ happens in an execution between $\mathcal{B}^{\mathcal{R}}$ and $\mathcal{C}$ on input $1^\lambda$ is bounded by $\mu_2(\lambda)$.*

[8] It will be more convenient to think about $(2n + r + 1)$ as a fundamental quantity as we see below.

*Proof.* Consider a query that $\mathcal{R}$ makes to the attacker and the respective security game, and assume that $\mathcal{B}$ is required to decrypt a challenge ciphertext. By Claim 32, there are at least $n$ relevant slots that were rewound. Let $s_1, \ldots, s_n$ be those slots. For every $i \in [n]$ we let $W_i \subseteq H$ be the set of hash functions $h \in H$ such that querying a secret key for a policy $f_h$ in slot $s_i$ results in the reduction responding with a valid key $\mathrm{SK}_{f_h}$ with probability at least $(1-\alpha)$, where $\alpha$ is a parameter that we will choose shortly. Note that the sets $W_1, \ldots, W_n$ are random variables that depend on the random coins used by $\mathcal{R}$ and $\mathcal{B}$.

Denote by $E_3$ the event that $W_1, \ldots, W_n$ are all smaller than $|H|/4$. Since $\mathcal{B}$ samples the policies in the key queries uniformly at random, and the game reaches the challenge phase only if the reduction responds with valid keys, the probability that $\mathcal{B}$ is required to decrypt and $E_3$ occurs is bounded by

$$\begin{aligned}\Pr[(\mathcal{B} \text{ required to decrypt}) \wedge E_3] &\leq \prod_{i=1}^{n} \left[\frac{|W_i|}{|H|} \cdot 1 + \left(1 - \frac{|W_i|}{|H|}\right)(1-\alpha)\right] \\ &\leq \prod_{i=1}^{n} \left[(1-\alpha) + \frac{|W_i|}{|H|} \cdot \alpha\right] < \prod_{i=1}^{n} \left(1 - \frac{3}{4}\alpha\right)\end{aligned} \tag{11}$$

Choosing $\alpha = 1/4$, we get

$$\Pr[(\mathcal{B} \text{ required to decrypt}) \wedge E_3] \leq \left(\frac{13}{16}\right)^n \tag{12}$$

Thus, this probability is bounded by a negligible function.

From now on, we consider the case in which $E_3$ does not occur, that is, there is a slot $s_i$ during the query phase that was successfully rewound and satisfies $|W_i| \geq |H|/4$. To analyze the probability that $\mathcal{B}$ fails to retrieve a key that decrypts the challenge ciphertext, we use the following additional notations:

- Let $T = \{1, \ldots, \lfloor \frac{m}{4l} \rfloor\}$, which is the set of values such that $h(x) \in T \iff f_h(x) = 1$.
- Let $W = W_i$, that is, the set of hash functions $h \in H$ such that querying $f_h$ in slot $s_i$ results in the reduction responding with a valid key with probability at least $1 - \alpha$ (where $\alpha = 1/4$).
- For all $\delta$, let $S_\delta = \left\{ x \in \{0,1\}^n \;\middle|\; \Pr_{h \xleftarrow{\$} W}[h(x) \in T] < \delta \right\}$, i.e., the set of all challenges such that if we sample a uniformly random $h \xleftarrow{\$} W$, the probability that $h(x) \in T$ is $< \delta$.
- For all $\delta$ and all $Y \subseteq H$ let $X_Y^\delta$ be the random variable $|h^{-1}(T) \cap S_\delta|$ when sampling a uniformly random $h \xleftarrow{\$} Y$.

Before diving further into the technical details, we provide a brief intuition. By definition of $S_\delta$, this is intuitively a set of "bad challenges" – challenges that have a small probability ($< \delta$) to be covered by a random hash function in $W$. Note that the smaller $\delta$ is, the smaller the set $S_\delta$ is. Since the attacker samples random hash functions in $H$, which is a pairwise independent family, the fraction of $S_\delta$ that is covered by a uniformly random $h \in H$ is close to its expectation with high probability. Therefore, for small $\delta$, we expect to find only a small fraction of the functions in $H$ whose intersection with $S_\delta$ is very small. These are intuitively "bad functions" because their contribution to the coverage of $S_\delta$, the set of "bad challenges", is little.

To compute the probability of covering a random challenge attribute $x^*$, we will compute lower and upper bounds on the expected coverage of $S_\delta$ by a random function in $W$, and extract a trade-off between the size of $S_\delta$ and $1/\delta$. The lower bound computation will assume a worst-case scenario in which all of the "bad functions" are in $W$ and exploit the fact that $W$ is a large enough fraction of $H$ to not be affected "too much" by the "bad functions" (when $\delta$ is chosen to be small enough).

As we will see, we can choose $\delta$ such that $\delta = 1/\mathrm{poly}(\lambda)$ and the size of $S_\delta$ is negligible. From this, we will conclude that a random challenge attribute is covered by the keys retrieved from the $k$ rewindings of $s_i$ with all but negligible probability.

We continue with a formal analysis:

**Lemma 32.** *For every $\delta$ we have*

$$\delta > \frac{|T|}{2m}\left(1 - \frac{4|H|m}{|W||T||S_\delta|}\right) \tag{13}$$

*Proof.* The proof is available in the eprint version of the paper: https://eprint.iacr.org/2023/952

Next, we use Lemma 32 and substitute the bounds $|W| \geq \frac{|H|}{4}$ and $\frac{|T|}{m} \geq \frac{1}{8l}$. We get that for every $\delta$

$$\delta > \frac{1}{16l}\left(1 - \frac{2^7 \cdot l}{|S_\delta|}\right) \tag{14}$$

Setting $\delta' = \frac{1}{32l}$ and substituting into (14),

$$\frac{1}{32l} > \frac{1}{16l}\left(1 - \frac{2^7 \cdot l}{|S_{\delta'}|}\right) \Rightarrow |S_{\delta'}| < 2^8 \cdot l \tag{15}$$

For sufficiently large $\lambda$ it holds that $l < 2^{n/2-8}$, thus $|S_{\delta'}| < 2^{n/2}$.

Next, we upper bound the probability of not covering a uniformly random $x^* \overset{\$}{\leftarrow} \{0,1\}^n$ by $\tilde{F}$, that is, the probability that $h(x^*) \notin T$ for every $f_h \in \tilde{F}$.

The remaining probabilistic calculation, which concludes the proof of this claim, is available in the eprint version of the paper: https://eprint.iacr.org/2023/952

By the two previous claims, the attacker simulated by $\mathcal{B}$ is indistinguishable by $\mathcal{R}$ from the hypothetical attacker $\mathcal{A}$, therefore the transcript of the interaction between $\mathcal{C}$ and $\mathcal{R}_0$ is indistinguishable from the transcript of the interaction between $\mathcal{C}$ and $\mathcal{R}^{\mathcal{A}}$, and so

$$\Pr\left[\langle \mathcal{B}^{\mathcal{R}}, \mathcal{C}\rangle\left(1^\lambda\right) = 1\right] = \Pr\left[\langle \mathcal{R}^{\mathcal{A}}, \mathcal{C}\rangle\left(1^\lambda\right) = 1\right] - \mathrm{negl}(\lambda) \tag{16}$$

Combining with the runtime analysis, we conclude that the expected running time of $\mathcal{B}$ is bounded by a polynomial function of $\lambda$, thus by Markov's inequality, we can truncate $\mathcal{B}$ to run in strictly polynomial time while preserving its non-negligible advantage w.r.t. the assumption $\mathcal{C}$. Finally, we conclude that there exists a polynomial-time machine such that, if given oracle access to $\mathcal{R}$, has a non-negligible advantage w.r.t. the assumption $\mathcal{C}$. This completes the proof.

## 4 The Case of Lattice-Based ABE

As an example of applying our framework, we consider the celebrated [BGG+14] KP-ABE candidate and show that a its delegatable version conforms with the conditions of our main theorem. The [BGG+14] scheme has been proven selectively secure based on the hardness of Learning with Errors (LWE), and while we are not aware of it being conjectured adaptively secure, we do not know of concrete adaptive attacks. We consider a variant of the scheme where function secret keys consist of lattice trapdoors. This version can be adapted to our framework fairly straightforwardly.

### 4.1 Lattice Cryptography Background

We start by presenting a few necessary definitions of lattice cryptography on the subjects of LWE and lattice trapdoors, which are required in order to describe the ABE scheme formally.

**Definition 41 (Decisional $\boldsymbol{LWE}_{n,m,q,\chi}$).** *Let $\lambda$ be a security parameter, $n = n(\lambda)$, $m = m(\lambda)$ and $q = q(\lambda)$ be integers, and $\chi = \chi(\lambda)$ be a noise distribution over $\mathbb{Z}$. The $(n, m, q, \chi)$-LWE decision problem is to distinguish between the following two distributions: Letting $A \overset{\$}{\leftarrow} \mathbb{Z}_q^{n\times m}$, $s \overset{\$}{\leftarrow} \mathbb{Z}_q^n$, $e \leftarrow \chi^m$, $u \overset{\$}{\leftarrow} \mathbb{Z}_q^m$, the first distribution is $(A, A^T s + e)$ and the second is $(A, u)$.*

**Definition 42 (Gadget Matrix).** *We define the "gadget matrix" by $G = g \otimes I_n \in \mathbb{Z}_q^{n\times n\lceil \log q\rceil}$ where $g = (1, 2, 4, \ldots, 2^{\lceil \log q\rceil - 1}) \in \mathbb{Z}_q^{\lceil \log q\rceil}$. We define the inverse of the gadget matrix function $G^{-1} : \mathbb{Z}_q^{n\times m} \to \{0,1\}^{n\lceil \log q\rceil\times n}$ which expends each entry $a \in \mathbb{Z}_q$ of the input matrix into a column of size $\lceil \log q\rceil$ which is a binary representation of $a$, so for any matrix $A \in \mathbb{Z}_q^{n\times m}$ it holds that $G \cdot G^{-1}(A) = A$.*

**Definition 43 (Matrix Norms).** *Let $T \in \mathbb{Z}^{n\times m}$ be a matrix and $\tilde{T}$ the result of applying Gram-Schmidt orthogonalization to the columns of $T$. We define the GS-norm $\|T\|_{GS}$ as the $l_2$ length of the longest column of $\tilde{T}$. We let $\|T\|_2$ be the operator norm of $T$ defined by $\|T\|_2 = \sup_{\|x\|=1} \|Tx\|$.*

The following are properties of lattice trapdoors, see [BGG+14] for references.

**Lemma 41.** *Let $m, n, q > 0$ be integers with $q$ prime.*

- *There is an efficient randomized algorithm TrapGen*$(1^n, 1^m, q)$ *that when* $m = \Theta(n \log q)$, *outputs a full-rank matrix* $A \in \mathbb{Z}_q^{n\times m}$ *along with a basis* $T_A \in \mathbb{Z}^{m\times m}$ *for*
$$\Lambda_q^\perp(A) = \{z \in \mathbb{Z}^m \mid A \cdot z = 0 \mod q\}$$
*such that* $A$ *is statistically indistinguishable from a uniformly-sampled matrix and* $\|T_A\|_{GS} = O(\sqrt{n \log q})$, *with all but negligible probability.*
- *There is an efficient algorithm TrapExtend*$(A, B, T_A)$ *that given a full-rank matrix* $A \in \mathbb{Z}_q^{n\times m}$, *a basis* $T_A$ *of* $\Lambda_q^\perp(A)$, *and* $B \in \mathbb{Z}_q^{n\times m}$, *outputs a basis* $T_{[A|B]}$ *of* $\Lambda_q^\perp([A \mid B])$ *such that* $\|T_{[A|B]}\|_{GS} = \|T_A\|_{GS}$.
- *There is a randomized algorithm SampleD*$(A, D, T_A, \sigma)$ *that given a full-rank matrix* $A \in \mathbb{Z}_q^{n\times m}$, *a basis* $T_A$ *of* $\Lambda_q^\perp(A)$, *a matrix* $D \in \mathbb{Z}_q^{n\times k}$, *and* $\sigma = \|T_A\|_{GS}\cdot\omega(\sqrt{\log m})$, *outputs a random matrix* $R \in \mathbb{Z}^{m\times k}$ *such that* $AR = D$, *and* $\|R^T\|_2 < m\sigma$ *with all but negligible probability.*

**Lemma 42.** *Let* $A \in \mathbb{Z}_q^{n\times m}$ *be a full rank matrix with a basis* $T_A \in \mathbb{Z}^{m\times m}$ *for* $\Lambda_q^\perp(A)$, *and let* $B \in \mathbb{Z}_q^{n\times k}$ *and* $C \in \mathbb{Z}_q^{k\times m}$ *such that* $A = BC$. *Let* $D \in \mathbb{Z}_q^{m\times k}$ *such that* $AD = B$ *(which necessarily exists since* $A$ *has a trapdoor* $T_A$*). Then the matrix* $T_B = [CT_A \mid I - CD]$ *is a full-rank set of vectors in* $\Lambda_q^\perp(B)$ *such that* $\|T_B\|_{GS} = \|CT_A\|_{GS}$.

*Proof.* We can immediately verify that $BT_B = 0$, thus $T_B \in \Lambda_q^\perp(B)$. $T_B$ is also full-rank since
$$T_B \begin{bmatrix} T_A^{-1}D \\ I \end{bmatrix} = I \tag{17}$$

For the same reason, the GS-norm of $T_B\|_{\text{GS}}$ is the same as $[CT_A \mid I]$, which is $\|CT_A\|_{\text{GS}}$, as desired.

*Key-Homomorphic Evaluation.* Let $f$ be a boolean circuit of depth $d$ computing a function $\{0,1\}^k \to \{0,1\}$, and assume that $f$ contains only NAND gates. We "translate" the operation of $f$ into a computation on matrices: We associate with every input wire of $f$ a matrix $A_i$, and for every other wire we assign a matrix recursively as follows: Let $A_\alpha, A_\beta$ be the matrices of the input wires, then the output wire is associated with the matrix $A_\gamma = A_\alpha \cdot G^{-1}(A_\beta) - G$. Note that for every input values $x_\alpha, x_\beta \in \{0,1\}$,
$$\begin{aligned}[A_\alpha + x_\alpha G \mid A_\beta + x_\beta G] \cdot \begin{bmatrix} G^{-1}(A_\beta) \\ -x_\alpha I \end{bmatrix} &= A_\gamma + (1 - x_\alpha x_\beta)G \\ &= A_\gamma + \text{NAND}(x_\alpha, x_\beta)G\end{aligned} \tag{18}$$

Denote by $A_f$ the matrix of the output wire of $f$. We define Eval$(f, (A_1, \ldots, A_k))$ to be the procedure that takes as inputs $f$ and $\boldsymbol{A} = (A_1, \ldots, A_k)$ and outputs $A_f$. Note that for input wires $x_1, \ldots, x_k$, the homomorphic evaluation satisfies
$$[A_1 + x_1 G \mid \cdots \mid A_k + x_k G] \cdot H_{f,x,\boldsymbol{A}} = A_f + f(x_1, \ldots, x_k)G \tag{19}$$

for some short matrix $H_{f,x,\boldsymbol{A}} \in \mathbb{Z}^{mk\times m}$ that has norm $O(n \log q)^{O(d)}$.

### 4.2 The [BGG+14] Scheme

Next, we describe the properties of [BGG+14] scheme and show their sufficiency for applying our theorem. For simplicity, we assume that the policies of the scheme accept attributes if and only if $f(x) = 0$ (instead of $f(x) = 1$).

Let $\lambda$ denote the security parameter, the parameters of the scheme are an integer $n = n(\lambda)$, a prime $q = q(\lambda)$, an integer $m = \Theta(n \log q)$, a noise distribution $\chi = \chi(\lambda)$ over $\mathbb{Z}$, and $d = d(\lambda)$. The noise distribution is chosen to be $\chi_{\max}$-bounded, that is, its support is in $[-\chi_{\max}, \chi_{\max}]$. The attribute set is $\mathcal{X} = \{0,1\}^k$ for $k$ which is given as input, and the class of policies $\mathcal{F}$ is the class of functions with depth-$d$ circuits.

The public parameters of the scheme are matrices $A_0, A_1, \ldots, A_k, D \in \mathbb{Z}_q^{n\times m}$. Every attribute $x$ is associated with a public key $A_x \in \mathbb{Z}_q^{n\times m}$ computed by

$$A_x \overset{\text{def}}{=} [A_0 \mid A_1 + x_1 G \mid \cdots \mid A_k + x_k G] \in \mathbb{Z}_q^{n\times m(k+1)} \tag{20}$$

Every predicate $f$ is also associated with a public key $A_f \in \mathbb{Z}_q^{n\times m}$, computed by $\mathrm{Eval}(f, (A_1, \ldots, A_k))$.

At a high level, the encryption procedure of the scheme is a variant of dual Regev encryption [Reg05], so a ciphertext encrypted under a public key $A \in \mathbb{Z}_q^{n\times l}$ is essentially a noisy vector close to the lattice spanned by $A$, and has the form $c^T = s^T A + e^T$ where $s \overset{\$}{\leftarrow} \mathbb{Z}_q^n$ is a uniformly random vector and $e \in \mathbb{Z}^l$ is a noise vector sampled from a distribution over short vectors. A message is encoded into the ciphertext by adding an "offset" that depends on the message. A secret key SK for $A$ is a lattice trapdoor $T_A$, i.e., a low-norm basis for the dual lattice $\Lambda_q^{\perp}(A)$. The trapdoor can be used to decrypt a ciphertext so long as the norm of its noise vector $e$ is small enough.

Formally, the encryption of a message $M \in \{0,1\}^m$ under public key $A_x$ is

$$\begin{aligned}
\mathrm{CT}_x^T = \quad & \left[c_0^T \mid c_1^T \mid \cdots \mid c_k^T \mid c_{\text{out}}^T\right] \\
= \quad & s^T \left[A_0 \mid x_1 G + A_1 \mid \cdots \mid x_k G + A_k \mid D\right] \\
& + \left[e_0^T \mid e_1^T \mid \cdots \mid e_k^T \mid e_{\text{out}}^T + \lceil q/2 \rceil M^T\right]
\end{aligned} \tag{21}$$

where $s \overset{\$}{\leftarrow} \mathbb{Z}^n$ and $e_0, \ldots, e_k, e_{\text{out}} \leftarrow \chi^m$.

To decrypt a ciphertext $\mathrm{CT}_x$ using a key $\mathrm{SK}_f$ for which $f(x) = 0$, one first homomorphically evaluates the ciphertext by applying a publicly known low-norm matrix $H_{f,x,\boldsymbol{A}} \in \mathbb{Z}^{mk\times m}$ (described in the key-homomorphic evaluation) that satisfies

$$[A_1 + x_1 G \mid \cdots \mid A_k + x_k G] H_{f,x,\boldsymbol{A}} = A_f + f(x) G \in \mathbb{Z}_q^{n\times m} \tag{22}$$

The result of evaluating a ciphertext with respect to policy $f$ is an encryption of the original message under the public matrix $[A_0 \mid A_f + f(x)G]$ as follows:

$$\begin{aligned}
c_f^T &= \left[c_1^T \mid \cdots \mid c_k^T\right] H_{f,x,\boldsymbol{A}} \\
&= s^T \left[x_1 G + A_1 \mid \cdots \mid x_k G + A_k\right] H_{f,x,\boldsymbol{A}} + \left[e_1^T \mid \cdots \mid e_k^T\right] H_{f,x,\boldsymbol{A}} \\
&= s^T A_f + \left[e_1^T \mid \cdots \mid e_k^T\right] H_{f,x,\boldsymbol{A}}
\end{aligned} \tag{23}$$

It holds that $\left\|H_{f,x,\boldsymbol{A}}^T\right\|_2 \le \Delta$ where $\Delta$ is a parameter of the scheme.

The secret key for a predicate $f$ is a trapdoor $T_f$ for $[A_0 \mid A_f]$, whose GS-norm is $\rho = O(\sqrt{n \log q})$. The trapdoor is used to sample a matrix $R \in \mathbb{Z}_q^{2m \times m}$ such that $[A_0 \mid A_f]R = D$ and $\|R^T\|_2 < 2m\rho\sigma$ for $\sigma$ that is a parameter of the scheme. Decryption is computed by

$$\begin{aligned} c_{\text{out}}^T - \left[c_0^T \mid c_f^T\right] R = \quad & s^T D + e_{\text{out}}^T + \lceil q/2 \rceil M^T \\ & - \left[s^T A_0 + e_0^T \mid s^T A_f + \left[e_1^T \mid \cdots \mid e_k^T\right] H_{f,x,\boldsymbol{A}}\right] R \\ = \quad & \lceil q/2 \rceil M^T + e_{\text{out}}^T - \left[e_0^T \mid \left[e_1^T \mid \cdots \mid e_k^T\right] H_{f,x,\boldsymbol{A}}\right] R \end{aligned} \tag{24}$$

We observe that the largest coordinate of the noise vector

$$e_{\text{out}}^T - \left[e_0^T \mid \left[e_1^T \mid \cdots \mid e_k^T\right] H_{f,x,\boldsymbol{A}}\right] R$$

is bounded by $\chi_{\max} + 2\, m\rho\sigma k\Delta\chi_{\max}$. The parameters of the scheme are chosen such that this bound is small enough compared to $q/4$, so one can round the result to extract $M$.

## 4.3 Applying Theorem 31

First, following Remark 26, the parameters of the scheme can be chosen to satisfy pairwise friendliness. Second, we claim that there exist an alternative decryption procedure, denote Decrypt', and a procedure SKCheck, such that the ABE scheme equipped with (SKCheck, Decrypt') satisfies the checkability property. Proving this claim is sufficient to apply our theorem to the scheme equipped with (SKCheck, Decrypt'), and by Remark 24 we conclude that the original scheme cannot be proved adaptively secure as well.

The implementation of SKCheck is relatively straightforward: Given the public parameters, a secret key SK, and a predicate $f$, verify that SK is a basis for $\Lambda_q^{\perp}([A_0 \mid A_f])$ and that its GS-norm is smaller than $\rho$. By definition of the scheme, we immediately get that any honestly-generated secret key passes the check, as desired.

Our strategy for the alternative decryption procedure is the following: Let $\text{SK}_f$ be a (valid) secret key for policy $f$ and let $\text{CT}_x$ be a ciphertext such that $f(x) = 0$. Note that

$$A_x \begin{bmatrix} I & 0 \\ 0 & H_{f,x,\boldsymbol{A}} \end{bmatrix} = [A_0 \mid A_f] \tag{25}$$

So by Lemma 42, the secret key, which is a trapdoor $T_f$ for $[A_0 \mid A_f]$, can be used to obtain a trapdoor $T_x$ such that $\|T_x\|_{\text{GS}} \le \Delta\rho$. We use the "secret key" $\text{SK}_x = T_x$ for $A_x$ to "decode" and decrypt the ciphertext $\text{CT}_x$ according to the following procedure. The procedure takes in a ciphertext, a secret key and public parameters.

**Decode**$(\text{CT}_x, \text{SK}_x, \text{PP})$: Denote the components of the ciphertext by $c_0, \ldots, c_k$, $c_{\text{out}}$ as defined in (21). Denote $\rho'$ the GS-norm of $\text{SK}_x$. Use $T_x = \text{SK}_x$ to obtain a matrix $R \in \mathbb{Z}^{m(k+1)\times m}$ such that $A_x R = D$ and $\|R^T\|_2 \leq (k+1)m\rho'\sigma$. Compute $c_{\text{out}}^T - \left[c_0^T \mid \cdots \mid c_k^T\right] R$ and round the result to extract $M \in \{0,1\}^m$ such that

$$\lceil q/2 \rceil M^T = \text{round}\left(c_{\text{out}}^T - \left[c_0^T \mid \cdots \mid c_k^T\right] R\right) \tag{26}$$

Use $T_x$ again to obtain a basis $R'$ for $\Lambda_q^{\perp}([A_x \mid D])$ such that $\|R'\|_{\text{GS}} \leq \rho'$. Compute the vector

$$y^T = \left[c_0^T \mid c_1^T \mid \cdots \mid c_k^T \mid c_{\text{out}}^T - \lceil q/2 \rceil M^T\right] R' \tag{27}$$

Lift $y$ to its canonical representative $\tilde{y} \in \left[-\frac{q}{2}, \frac{q}{2}\right)^{m(k+2)}$, compute $R'^{-1}$ over the rationals, and compute $z^T = \tilde{y}^T R'^{-1}$. Let $z_0, z_1, \ldots, z_k, z_{\text{out}} \in \mathbb{Z}^m$ such that $z = (z_0, z_1, \ldots, z_k, z_{\text{out}})$. Check that the coordinates of $z_0, z_1, \ldots, z_k, z_{\text{out}}$ are smaller than $\chi_{\max}$. If not, output $\perp$, otherwise, output $M$.

We claim that the new procedure satisfies the ABE correctness requirement, that is, the result of the Decode procedure on honestly generated input is the message $M$ encrypted in the input ciphertext. We prove this formally in the following lemma:

**Lemma 43.** *For any $M \in \{0,1\}^m$, $x \in \{0,1\}^k$, honestly generated public parameters PP, honestly generated public key $A_x$ for $x$, and honestly generated secret key $T_f$ for for a predicate $f$ such that $f(x) = 0$, it holds that*

$$\textit{Decrypt}(\textit{Encrypt}(x, M, PP), T_x, PP) = M \tag{28}$$

*Proof.* The proof is available in the eprint version of the paper: https://eprint.iacr.org/2023/952

Finally, we prove the following lemma to conclude the checkability of the alternative decryption:

**Lemma 44.** *Let $T_1, T_2$ be two trapdoors for $A_x$ such that $\|T_1\|_{GS}, \|T_1\|_{GS} \leq \Delta\rho$. For any ciphertext $CT_x$ it holds that Decode$(CT_x, T_1, PP) =$ Decode$(CT_x, T_2, PP)$.*

*Proof.* The proof is available in the eprint version of the paper: https://eprint.iacr.org/2023/952

## References

[ABSV15] Ananth, P., Brakerski, Z., Segev, G., Vaikuntanathan, V.: From selective to adaptive security in functional encryption. In: Gennaro, R., Robshaw, M. (eds.) CRYPTO 2015, Part II. LNCS, vol. 9216, pp. 657–677. Springer, Heidelberg (2015). https://doi.org/10.1007/978-3-662-48000-7_32

[ABV+12] Agrawal, S., Boyen, X., Vaikuntanathan, V., Voulgaris, P., Wee, H.: Functional encryption for threshold functions (or Fuzzy IBE) from lattices. In: Fischlin, M., Buchmann, J., Manulis, M. (eds.) PKC 2012. LNCS, vol. 7293, pp. 280–297. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-30057-8_17

[AFV11] Agrawal, S., Freeman, D.M., Vaikuntanathan, V.: Functional encryption for inner product predicates from learning with errors. In: Lee, D.H., Wang, X. (eds.) ASIACRYPT 2011. LNCS, vol. 7073, pp. 21–40. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-25385-0_2

[BGG+14] Boneh, D., et al.: Fully key-homomorphic encryption, arithmetic circuit ABE and compact garbled circuits. In: Nguyen, P.Q., Oswald, E. (eds.) EUROCRYPT 2014. LNCS, vol. 8441, pp. 533–556. Springer, Heidelberg (2014). https://doi.org/10.1007/978-3-642-55220-5_30

[Boy13] Boyen, X.: Attribute-based functional encryption on lattices. In: Sahai, A. (ed.) TCC 2013. LNCS, vol. 7785, pp. 122–142. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-36594-2_8

[BV16] Brakerski, Z., Vaikuntanathan, V.: Circuit-ABE from LWE: unbounded attributes and semi-adaptive security. In: Robshaw, M., Katz, J. (eds.) CRYPTO 2016, Part III. LNCS, vol. 9816, pp. 363–384. Springer, Heidelberg (2016). https://doi.org/10.1007/978-3-662-53015-3_13

[CGKW18] Chen, J., Gong, J., Kowalczyk, L., Wee, H.: Unbounded ABE via bilinear entropy expansion, revisited. In: Nielsen, J.B., Rijmen, V. (eds.) EUROCRYPT 2018. LNCS, vol. 10820, pp. 503–534. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-78381-9_19

[CHK03] Canetti, R., Halevi, S., Katz, J.: A forward-secure public-key encryption scheme. In: Biham, E. (ed.) EUROCRYPT 2003. LNCS, vol. 2656, pp. 255–271. Springer, Heidelberg (2003). https://doi.org/10.1007/3-540-39200-9_16

[DDM15] Datta, P., Dutta, R., Mukhopadhyay, S.: Compact attribute-based encryption and signcryption for general circuits from multilinear maps. In: Biryukov, A., Goyal, V. (eds.) INDOCRYPT 2015. LNCS, vol. 9462, pp. 3–24. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-26617-6_1

[DGP21] Delerablée, C., Gouriou, L., Pointcheval, D.: Key-policy ABE with delegation of rights. IACR Cryptol. ePrint Arch., p. 867 (2021). https://eprint.iacr.org/2021/867

[DNS04] Dwork, C., Naor, M., Sahai, A.: Concurrent zero-knowledge. J. ACM **51**(6), 851–898 (2004). https://doi.org/10.1145/1039488.1039489

[GGH+13] Garg, S., Gentry, C., Halevi, S., Raykova, M., Sahai, A., Waters, B.: Candidate indistinguishability obfuscation and functional encryption for all circuits. In: 54th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2013, 26-29 October, 2013, Berkeley, CA, USA, pp. 40–49. IEEE Computer Society, 2013. Full version in [GGH+16]. https://doi.org/10.1109/FOCS.2013.13

[GGH+16] Garg, S., Gentry, C., Halevi, S., Raykova, M., Sahai, A., Waters, B.: Candidate indistinguishability obfuscation and functional encryption for all circuits. SIAM J. Comput. **45**(3), 882–929 (2016). http://dx.doi.org/10.1137/14095772X

[GKW16] Goyal, R., Koppula, V., Waters, B.: Semi-adaptive security and bundling functionalities made generic and easy. In: Hirt, M., Smith, A. (eds.) TCC 2016, Part II. LNCS, vol. 9986, pp. 361–388. Springer, Heidelberg (2016). https://doi.org/10.1007/978-3-662-53644-5_14

[GKW17] Goyal, R., Koppula, V., Waters, B.: Lockable obfuscation. In: Umans, C. (ed.) 58th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2017, Berkeley, CA, USA, October 15–17, 2017, pp. 612–621. IEEE Computer Society (2017). https://doi.org/10.1109/FOCS.2017.62

[GPSW06] Goyal, V., Pandey, O., Sahai, A., Waters, B.: Attribute-based encryption for fine-grained access control of encrypted data. In: Juels, A., Wright, R.N., De Capitani di Vimercati, S. (eds.) Proceedings of the 13th ACM Conference on Computer and Communications Security, CCS 2006, Alexandria, VA, USA, Ioctober 30 - November 3, 2006, pp. 89–98. ACM (2006). https://doi.org/10.1145/1180405.1180418

[GVW13] Gorbunov, S., Vaikuntanathan, V., Wee, H.: Attribute-based encryption for circuits. In: Boneh, D., Roughgarden, T., Feigenbaum, J. (eds.) Symposium on Theory of Computing Conference, STOC'13, Palo Alto, CA, USA, June 1–4, 2013, pp. 545–554. ACM (2013). https://doi.org/10.1145/2488608.2488677

[GW20] Gong, J., Wee, H.: Adaptively secure ABE for DFA from $k$-lin and more. In: Canteaut, A., Ishai, Y. (eds.) EUROCRYPT 2020. LNCS, vol. 12107, pp. 278–308. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-45727-3_10

[IKOS08] Ishai, Y., Kushilevitz, E., Ostrovsky, R., Sahai, A.: Cryptography with constant computational overhead. In: Dwork, C. (ed.) Proceedings of the 40th Annual ACM Symposium on Theory of Computing, Victoria, British Columbia, Canada, May 17–20, 2008, pp. 433–442 (2008). ACM (2008). https://doi.org/10.1145/1374376.1374438

[KL15] Kowalczyk, L., Lewko, A.B.: Bilinear entropy expansion from the decisional linear assumption. In: Gennaro, R., Robshaw, M. (eds.) CRYPTO 2015, Part II. LNCS, vol. 9216, pp. 524–541. Springer, Heidelberg (2015). https://doi.org/10.1007/978-3-662-48000-7_26

[KW20] Kowalczyk, L., Wee, H.: Compact adaptively secure ABE for NC1 from k-lin. J. Cryptol. **33**(3), 954–1002 (2020). https://doi.org/10.1007/s00145-019-09335-x

[LL20] Lin, H., Luo, J.: Compact adaptively secure ABE from $k$-lin: beyond $\mathsf{NC}^1$ and towards $\mathsf{NL}$. In: Canteaut, A., Ishai, Y. (eds.) EUROCRYPT 2020, Part III. LNCS, vol. 12107, pp. 247–277. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-45727-3_9

[LOS+10] Lewko, A., Okamoto, T., Sahai, A., Takashima, K., Waters, B.: Fully secure functional encryption: attribute-based encryption and (hierarchical) inner product encryption. In: Gilbert, H. (ed.) EUROCRYPT 2010. LNCS, vol. 6110, pp. 62–91. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-13190-5_4

[LW12] Lewko, A., Waters, B.: New proof methods for attribute-based encryption: achieving full security through selective techniques. In: Safavi-Naini, R., Canetti, R. (eds.) CRYPTO 2012. LNCS, vol. 7417, pp. 180–198. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-32009-5_12

[LW14] Lewko, A., Waters, B.: Why proving HIBE systems secure is difficult. In: Nguyen, P.Q., Oswald, E. (eds.) EUROCRYPT 2014. LNCS, vol. 8441, pp. 58–76. Springer, Heidelberg (2014). https://doi.org/10.1007/978-3-642-55220-5_4

[MP12] Micciancio, D., Peikert, C.: Trapdoors for lattices: simpler, tighter, faster, smaller. In: Pointcheval, D., Johansson, T. (eds.) EUROCRYPT 2012. LNCS, vol. 7237, pp. 700–718. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-29011-4_41

[Nao03] Naor, M.: On cryptographic assumptions and challenges. In: Boneh, D. (ed.) CRYPTO 2003. LNCS, vol. 2729, pp. 96–109. Springer, Heidelberg (2003). https://doi.org/10.1007/978-3-540-45146-4_6

[OSW07] Ostrovsky, R., Sahai, A., Waters, B.: Attribute-based encryption with non-monotonic access structures. In: Ning, S., De Capitani di Vimercati, S., Syverson, P.F. (eds.) Proceedings of the 2007 ACM Conference on Computer and Communications Security, CCS 2007, Alexandria, Virginia, USA, October 28–31, 2007, pp. 195–203. ACM (2007). https://doi.org/10.1145/1315245.1315270

[Pas11] Pass, R.: Limits of provable security from standard assumptions. In: Fortnow, L., Vadhan, S.P. (eds.) Proceedings of the 43rd ACM Symposium on Theory of Computing, STOC 2011, San Jose, CA, USA, 6–8 June 2011, pp. 109–118. ACM (2011). https://doi.org/10.1145/1993636.1993652

[Reg05] Regev, O.: On lattices, learning with errors, random linear codes, and cryptography. In: Gabow, H.N., Fagin, R. (eds.) Proceedings of the 37th Annual ACM Symposium on Theory of Computing, Baltimore, MD, USA, May 22–24, 2005, pp. 84–93. ACM (2005). https://doi.org/10.1145/1060590.1060603

[SW05] Sahai, A., Waters, B.: Fuzzy identity-based encryption. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 457–473. Springer, Heidelberg (2005). https://doi.org/10.1007/11426639_27

[Tsa19] Tsabary, R.: Fully secure attribute-based encryption for $t$-CNF from LWE. Cryptology ePrint Archive, Paper 2019/365 (2019). https://eprint.iacr.org/2019/365

[Wat09] Waters, B.: Dual system encryption: realizing fully secure IBE and HIBE under simple assumptions. In: Halevi, S. (ed.) CRYPTO 2009. LNCS, vol. 5677, pp. 619–636. Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-642-03356-8_36

[Wat11] Waters, B.: Ciphertext-policy attribute-based encryption: an expressive, efficient, and provably secure realization. In: Catalano, D., Fazio, N., Gennaro, R., Nicolosi, A. (eds.) PKC 2011. LNCS, vol. 6571, pp. 53–70. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-19379-8_4

[Wat15] Waters, B.: A punctured programming approach to adaptively secure functional encryption. In: Gennaro, R., Robshaw, M. (eds.) CRYPTO 2015. LNCS, vol. 9216, pp. 678–697. Springer, Heidelberg (2015). https://doi.org/10.1007/978-3-662-48000-7_33

# Tighter Adaptive IBEs and VRFs: Revisiting Waters' Artificial Abort

Goichiro Hanaoka[1], Shuichi Katsumata[1,2], Kei Kimura[3], Kaoru Takemure[1,2](✉), and Shota Yamada[1]

[1] AIST, Tokyo, USA
{hanaoka-goichiro,yamada-shota}@aist.go.jp
[2] PQShield, Oxford, UK
{shuichi.katsumata,kaoru.takemure}@pqshield.com
[3] Kyushu University, Fukuoka, Japan
kkimura@inf.kyushu-u.ac.jp

**Abstract.** One of the most popular techniques to prove adaptive security of identity-based encryptions (IBE) and verifiable random functions (VRF) is the *partitioning technique*. Currently, there are only two methods to relate the adversary's advantage and runtime $(\epsilon, \mathsf{T})$ to those of the reduction's $(\epsilon_{\mathsf{proof}}, \mathsf{T}_{\mathsf{proof}})$ using this technique: One originates to Waters (Eurocrypt 2005) who introduced the famous *artificial abort* step to prove his IBE, achieving $(\epsilon_{\mathsf{proof}}, \mathsf{T}_{\mathsf{proof}}) = (O(\epsilon/Q), \mathsf{T} + O(Q^2/\epsilon^2))$, where $Q$ is the number of key queries. Bellare and Ristenpart (Eurocrypt 2009) provide an alternative analysis for the same scheme removing the artificial abort step, resulting in $(\epsilon_{\mathsf{proof}}, \mathsf{T}_{\mathsf{proof}}) = (O(\epsilon^2/Q), \mathsf{T} + O(Q))$. Importantly, the current reductions all loose quadratically in $\epsilon$.

In this paper, we revisit this two decade old problem and analyze proofs based on the partitioning technique through a new lens. For instance, the Waters IBE can now be proven secure with $(\epsilon_{\mathsf{proof}}, \mathsf{T}_{\mathsf{proof}}) = (O(\epsilon^{3/2}/Q), \mathsf{T} + O(Q))$, breaking the quadratic dependence on $\epsilon$. At the core of our improvement is a finer estimation of the failing probability of the reduction in Waters' original proof relying on artificial abort. We use Bonferroni's inequality, a tunable inequality obtained by cutting off higher order terms from the equality derived by the inclusion-exclusion principle.

Our analysis not only improves the reduction of known constructions but also opens the door for new constructions. While a similar improvement to Waters IBE is possible for the lattice-based IBE by Agrawal, Boneh, and Boyen (Eurocrypt 2010), we can slightly tweak the so-called partitioning function in their construction, achieving $(\epsilon_{\mathsf{proof}}, \mathsf{T}_{\mathsf{proof}}) = (O(\epsilon/Q), \mathsf{T} + O(Q))$. This is a much better reduction than the previously known $(O(\epsilon^3/Q^2), \mathsf{T} + O(Q))$. We also propose the first VRF with proof and verification key sizes sublinear in the security parameter under the standard $d$-LIN assumption, while simultaneously improving the reduction cost compared to all prior constructions.

E. Boyle and M. Mahmoody (Eds.): TCC 2024, LNCS 15366, pp. 124–155, 2025.
https://doi.org/10.1007/978-3-031-78020-2_5

# 1 Introduction

## 1.1 Background

In security proofs for cryptographic primitives, we often face conflicting requirements. For instance, when proving security of a signature scheme, the reduction needs to simulate signatures upon adversary's signing queries and to extract a solution to a computationally hard problem from the forgery. On first glance, such a proof seems to indicate that a reduction can simply simulate an adversary internally: It simulates a forgery instead of running the adversary and extracts the solution from it, contradicting the hardness of the problem. The *partitioning technique* resolves this apparent paradox. The message space is divided into controlled and uncontrolled sets. The reduction can only simulate signatures for controlled messages, while a forgery is only useful if it's for an uncontrolled message. Since a message can only be either controlled or uncontrolled, the paradox is resolved. This technique has been useful outside the simple application of signatures, and in particular, has been central to show *adaptive* security of more advanced primitives such as identity-based encryption (IBE) [1–4,7,8,12,24,25,30,32–34] and verifiable random function (VRF) with large input spaces [16,19,21–24,26,28,33].[1]

A proof relying on the partitioning technique comes in two steps. The *first step* consists of constructing a scheme that secretly partitions the challenge space in controlled and uncontrolled sets during the security proof. This is typically done by implicitly computing a bespoke keyed function $\mathsf{F}$ inside the scheme. In the context of signatures, this *partitioning function* $\mathsf{F}(\mathsf{M})$ is secretly computed during the signing algorithm, where $\mathsf{F}(\mathsf{M}) = 1$ (resp. 0) indicates that $\mathsf{M}$ is included in the controlled (resp. uncontrolled) set. For the reduction, the probability that the adversarial queries are consistent with the partition made by $\mathsf{F}$ needs to be high enough. Specifically, the probability that (i) $\mathsf{F}(\mathsf{M}^{(i)}) = 1$ for all messages $(\mathsf{M}^{(i)})_{i\in[Q]}$ queried to the signing oracle and (ii) $\mathsf{F}(\mathsf{M}^*) = 0$ for the forgery message $\mathsf{M}^*$ must be noticeable. Below, we denote this probability as $\gamma(\mathbf{M})$, where $\mathbf{M} := (\mathsf{M}^{(1)}, \cdots, \mathsf{M}^{(Q)}, \mathsf{M}^*)$. The *second step* is to lower bound the advantage $\epsilon_{\mathsf{proof}}$ of the reduction using the advantage of the adversary $\epsilon$. This step is trivial when reducing a hard search problem to a search type security game (e.g., unforgeability of a signature scheme) as we have a simple lower bound $\epsilon_{\mathsf{proof}} \geq \gamma_{\mathsf{min}}\epsilon$, where $\gamma_{\mathsf{min}} = \min_{\mathbf{M}} \gamma(\mathbf{M})$. However, such a simple bound no longer holds when reducing a hard decisional problem to a decisional security game, those considered by IBEs and VRFs. Studying the partitioning technique in this non-trivial setting is the main focus of our work.[2]

To the best of our knowledge, there are only two solutions to the second step of the partitioning technique. The first solution originates to Waters [30], who

[1] Other techniques to achieve adaptive security relying on specific algebraic structures (e.g., dual system encryption) exists. See Sect. 1.3 for more details.

[2] Looking ahead, the difficulty stems from the fact that in a decisional security game, the adversary may have a *negative* advantage conditioned on $\mathbf{M}$. We refer to Sect. 2.1 for the details. .

identified this non-triviality when proving security of his IBE. His main observation was that it suffices to *efficiently approximate* $\gamma(\mathsf{ID})$ to lower bound $\epsilon_{\mathsf{proof}}$, where we replace $\mathsf{M}$ with $\mathsf{ID}$ to be consistent with our IBE explanation. Namely, he used the Monte Carlo method to approximate $\gamma(\mathsf{ID})$ and completed the reduction using the notorious *artificial abort* step; a counterintuitive step where the reduction sometimes aborts the simulation and outputs a random guess, even if the simulation is successful (i.e., the adversarial queries lie in the correct constrained and unconstrained sets). While this solved the elusive problem of using the partitioning technique for decisional security games, the main caveat was that performing artificial abort incurred a huge runtime loss due to the Monte Carlo method. Denoting the runtime of the reduction and adversary by $\mathsf{T}_{\mathsf{proof}}$ and $\mathsf{T}$, respectively, we have $(\epsilon_{\mathsf{proof}}, \mathsf{T}_{\mathsf{proof}}) = (O(\epsilon/Q), \mathsf{T} + O(Q^2/\epsilon^2))$, where $Q$ is the number of key queries made by the adversary.[3] The second solution is due to Bellare and Ristenpart [5]. They showed that if the value of $\gamma(\mathsf{ID})$ for all $\mathsf{ID}$ lie in a narrow enough interval, the artificial abort step by Waters can be removed from the reduction. Specifically, the reduction no longer needs to run the costly Monte Carlo method. To satisfy this condition on $\gamma(\mathsf{ID})$, they further proposed a new partitioning function $\mathsf{F}$. Altogether, they achieve a better reduction with $(\epsilon_{\mathsf{proof}}, \mathsf{T}_{\mathsf{proof}}) = (O(\epsilon^2/Q), \mathsf{T} + O(Q))$, shaving off a factor $Q$ in total. However, notice the advantage $\epsilon_{\mathsf{proof}}$ becomes worser than Waters due to the modification they made to the partitioning function $\mathsf{F}$. Importantly, both solutions still have a reduction loss quadratic in $\epsilon$.

Surprisingly, this analysis of $(\epsilon_{\mathsf{proof}}, \mathsf{T}_{\mathsf{proof}})$, i.e., the second step of the partitioning technique, has not seen any improvement for over 15 years. Indeed, all previously cited IBEs [1–4,7,8,12,24,25,30,32–34] and VRFs [16,19,21–24,26,28,33] have proofs based on the partitioning technique that rely either on a Waters-style analysis or a Bellare-Ristenpart-style analysis—most of the improvements come from designing a better partitioning function $\mathsf{F}$ with a compatible scheme, i.e., improving the first step of the partitioning technique. This motivates us with the following question:

*Can we achieve a better reduction cost for proofs based on the partitioning technique? That is, is there a better analysis than those by Waters* [30] *and Bellare and Ristenpart* [5]*?*

## 1.2 Our Contributions

In this paper, we answer the above question affirmatively by proposing a new analysis for proofs based on the partitioning technique. Using our analysis, we improve the reduction cost of many of the aforementioned IBEs and VRFs *without* any modification to the construction. For example, Waters IBE can now be proven secure with $(\epsilon_{\mathsf{proof}}, \mathsf{T}_{\mathsf{proof}}) = (O(\epsilon^{3/2}/Q), \mathsf{T} + O(Q))$, breaking the quadratic dependence on $\epsilon$. We further obtain the same reduction cost for the

[3] Throughout the introduction, we ignore factors only depending on the security parameter $\kappa$ and focus on the adversarially dependent $Q$ and $\epsilon$.

lattice-based Agrawal-Boneh-Boyen (ABB) IBE [3], where the known reduction was quite loose, only achieving $(\epsilon_{\mathsf{proof}}, \mathsf{T}_{\mathsf{proof}}) = (O(\epsilon^3/Q^2), \mathsf{T} + O(Q))$.[4]

Our analysis not only improves the reduction of known constructions but also opens the door for new constructions. Concretely, we construct an IBE and VRF with novel properties.

- By slightly tweaking the ABB IBE construction, we obtain an IBE with a reduction $(\epsilon_{\mathsf{proof}}, \mathsf{T}_{\mathsf{proof}}) = \big(O(\epsilon^{1+\frac{1}{d-1}}/Q), \mathsf{T} + O(Q)\big)$, where $d \geq 3$ is a tunable positive integer that roughly dictates the length of the public parameter. When $d = 3$, we recover the ABB IBE, modulo the small difference in how an identity $\mathsf{ID}$ is hashed to matrices. By setting $d = \omega(1)$, we achieve $(\epsilon_{\mathsf{proof}}, \mathsf{T}_{\mathsf{proof}}) = (O(\epsilon/Q), \mathsf{T} + O(Q))$, which can be thought of as an *ideal* reduction for a partitioning based proof, matching the lower bound for the (black-box) reduction for Waters IBE [17].
- We propose the first VRF achieving sublinear verification key and proof sizes (in the security parameter) under the standard $d$-LIN assumption. Previous VRFs only achieved this under non-static $q$-type assumptions. In fact, we propose two VRFs, where one achieves an $\omega(1)$ proof size at the cost of increasing the verification key size slightly compared to the other. Moreover, the two VRFs enjoy a reduction of $(\epsilon_{\mathsf{proof}}, \mathsf{T}_{\mathsf{proof}}) = \big(O(\epsilon^{1.5}/Q), \mathsf{T} + O(Q)\big)$ and $\big(O(\epsilon^{1+\frac{\mu}{2}}/Q^{\mu}), \mathsf{T} + O(Q)\big)$ for an arbitrary constant $\mu > 1$, respectively. All prior reductions of VRFs with either sublinear verification key or proof sizes only achieve $(\epsilon_{\mathsf{proof}}, \mathsf{T}_{\mathsf{proof}}) = \big(O(\epsilon^{1+\mu}/Q^{\mu}), \mathsf{T} + O(Q)\big)$, or worse. We refer to Table 2 in Sect. 6.2 for the detailed comparison.

At the core of our technical contribution is a new framework for partitioning that interpolates the analysis of Waters and Bellare-Ristenpart in a way that we achieve the best of both worlds. Recall Waters [30] used the naive Monte Carlo method to approximate $\gamma(\mathsf{ID})$. While this leads to a good approximation, it suffers from longer runtime of $O(Q^2/\epsilon^2)$. In contrast, Bellare and Ristenpart [5] show that if $\gamma(\mathsf{ID})$ for all $\mathsf{ID}$ lie within a narrow enough interval, the expensive approximation step can be removed. This intuitively requires that a fixed value $\tilde{\gamma}$ can be used as a good enough approximation for $\gamma(\mathsf{ID})$ for *all* $\mathsf{ID}$. To realize this restrictive condition, they have to change the partitioning function $\mathsf{F}$, leading to a worser advantage $\epsilon_{\mathsf{proof}} = O(\epsilon^2/Q)$.

In our work, we resurrect Waters' artificial abort step, where we approximate $\gamma(\mathsf{ID})$ for *each* $\mathsf{ID}$, rather than requiring a single approximation $\tilde{\gamma}$ that works for $\gamma(\mathsf{ID})$ for *all* $\mathsf{ID}$ as Bellare-Ristenpart. This provides us with greater flexibility in selecting the partitioning function $\mathsf{F}$ compared to Bellare-Ristenpart and opens up the potential for achieving a higher advantage $\epsilon_{\mathsf{proof}}$. To this end, we require an improved approximation for $\gamma(\mathsf{ID})$ in comparison to Bellare and Ristenpart, as well as an efficient algorithm for computing this approximation in comparison to the Monte-Carlo method by Waters. For a better approximation of $\gamma(\mathsf{ID})$, we use Bonferroni's inequality [10], a tunable inequality obtained by cutting of

[4] To be precise, we modify the partitioning function used in ABB-IBE in a superficial manner, so technically speaking, it is no longer an identical scheme (see Sect. 2.7).

higher order terms from the equality derived by the inclusion-exclusion principle. The evaluation of $\gamma(\mathsf{ID})$ by Bellare and Ristenpart, which uses union bound, can be seen as an application of the special case of Bonferroni's inequality. Now, computing an approximation of $\gamma(\mathsf{ID})$ depends on the concrete choice of the partitioning function $\mathsf{F}$. In one case, used by Waters IBE, we need to solve certain counting problem efficiently. For this purpose, we use *generating functions*—a standard tool in enumerative combinatorics but seldom used in cryptography. This part may be of independent interest.

Given that the second step of the partitioning technique remains independent of the underlying primitives (e.g., IBE or VRF) and algebraic structures (e.g., pairings or lattices), we abstract it as a partitioning function *with approximation*, an extension of the partitioning function due to Yamada [33]. We extend the prior definition by augmenting it with an efficient algorithm that estimates $\gamma(\mathsf{ID})$. We revisit partitioning functions implicitly used in previous works [3,27,30], observing that they fit within our abstraction.

Lastly, our new analysis indicates that it is beneficial to choose a partitioning function $\mathsf{F}$ that allows to nicely and efficiently approximate $\gamma(\mathsf{ID})$. This leads to new ideas to improve the first step of the partitioning technique. For example, we show that by slightly tweaking the partitioning function $\mathsf{F}$ used in ABB IBE, we can efficiently compute the Bonferroni's inequality at a much higher order, allowing for a better approximation of $\gamma(\mathsf{ID})$. Further details are given in Sect. 2.

## 1.3 Related Works

We refer to the full version for related works on IBEs and VRFs.

**Related Works on Partitioning Techniques.** Many prior works focus on the first step of the partitioning technique, namely, designing of the partitioning function $\mathsf{F}$ and compatible algebraic structures. Concrete examples are for instance the admissible hash function [8,12,13,27], Waters hash [5,18,19,30] and its variant [3,11], and others [2,32–34]. These partitioning functions lead to IBEs and VRFs with various tradeoffs between efficiency, underlying assumption, and tightness when combined with suitable algebraic structures. We refer to the full version for more information.

Several works abstract out the algebraic structure that is compatible with the partitioning. In particular, Hofheinz and Kiltz [18] introduce the notion of programmable hash functions on pairing groups, which abstracts out the properties of Waters hash [30]. They show new applications along with novel asymptotic analysis. Zhang, Chen, and Zhang [34] extend the notion of programmable hash function to the lattice settings. Importantly though, all the above works on IBEs and VRFs rely either on the Waters-style analysis or Bellare-Ristenpart-style analysis to argue the second step of the partitioning technique[5], possibly resulting in a sub-optimal reduction.

[5] The work by Hofheinz and Kiltz [18] does not explicitly consider an application to IBEs. However, we can use their framework in the context of IBE and this requires the heavy artificial abort step in the reduction.

## 2 Technical Overview

We provide an overview of our techniques. Due to page limitation, the overview for our construction of VRF can be found in the full version of this paper.

### 2.1 The Difficulty

Let us review the proof of Waters IBE [30] based on the partitioning technique and observe the non-triviality of it. In his proof, the reduction algorithm for DBDH perfectly simulates the security game for an adversary $\mathsf{A}$ against the IBE scheme until it reaches the point where it cannot continue the simulation anymore and has to abort the reduction. The probability that the simulation is not successful only depends on the sequence of identities $\mathbf{ID} = (\mathsf{ID}^*, \mathsf{ID}^{(1)}, \ldots, \mathsf{ID}^{(Q)})$, where $\mathsf{ID}^* \in \{0,1\}^\ell$ is the challenge identity for which the challenge ciphertext is generated and $\mathsf{ID}^{(1)}, \ldots, \mathsf{ID}^{(Q)} \in \{0,1\}^\ell$ are identities for which key queries were made. Let us analyze a naive reduction that outputs the same bit as $\mathsf{A}$ when the simulation is successful and outputs a random bit when the simulation fails.

We denote the advantage of the adversary $\mathsf{A}$ by $\epsilon$, the probability that $\mathsf{A}$ makes the sequence of queries $\mathbf{ID}$ by $p(\mathbf{ID})$, and its advantage conditioned on the sequence of queries $\mathbf{ID}$ by $\epsilon(\mathbf{ID})$. We have

$$\frac{1}{2} + \epsilon = \sum_{\mathbf{ID}} p(\mathbf{ID}) \left( \frac{1}{2} + \epsilon(\mathbf{ID}) \right) = \frac{1}{2} + \sum_{\mathbf{ID}} p(\mathbf{ID}) \epsilon(\mathbf{ID}),$$

where the sum is taken over all possible $\mathbf{ID}$. Denoting the probability of the simulation being successful by $\gamma(\mathbf{ID})$,[6] the advantage of the reduction algorithm against DBDH can be evaluated as

$$\sum_{\mathbf{ID}} p(\mathbf{ID}) \left( \gamma(\mathbf{ID}) \left( \frac{1}{2} + \epsilon(\mathbf{ID}) \right) + \frac{1 - \gamma(\mathbf{ID})}{2} \right) - \frac{1}{2} = \sum_{\mathbf{ID}} \gamma(\mathbf{ID}) p(\mathbf{ID}) \epsilon(\mathbf{ID}). \quad (1)$$

In Waters' proof, it is shown that $\gamma(\mathbf{ID}) \geq 1/\mathsf{poly}$ for all possible $\mathbf{ID}$. It is tempting to conclude the proof by claiming the above advantage is non-negligible conditioned on $\epsilon$ being non-negligible. However, this intuition turns out to be false and this is precisely the reason why artificial abort was introduced in [30]. As an illustrating example, consider an adversary who yields only two types of query sequences $\mathbf{ID}_A$ and $\mathbf{ID}_B$. We further assume $p(\mathbf{ID}_A) = p(\mathbf{ID}_B) = 1/2$, $\gamma(\mathbf{ID}_A) = 1/3$, $\gamma(\mathbf{ID}_B) = 2/3$, $\epsilon(\mathbf{ID}_A) = 2/5$, and $\epsilon(\mathbf{ID}_B) = -1/5$. Even though the adversary $\mathsf{A}$ has advantage $1/10$, the advantage of the reduction algorithm is 0, meaning that it guesses the challenge bit no better than randomly guessing.

The reason why the above problem occurs is that $\epsilon(\mathbf{ID})$ can be negative for some $\mathbf{ID}$. When the "weight" on $\epsilon(\mathbf{ID})$ changes from $p(\mathbf{ID})$ to $p(\mathbf{ID})\gamma(\mathbf{ID})$ due

[6] We differentiate "not aborting" and "simulation being successful", since we will later introduce artificial abort, where the simulator aborts even if the simulation is successful. We note that in the naive reduction described here, this distinction is irrelevant.

to the failure of the simulation, the negative $\epsilon(\mathsf{ID})$ may be amplified to cancel out the positive $\epsilon(\mathsf{ID}')$, rendering the total sum being negligible. It is worth highlighting that this is exactly why partitioning based proofs are easier for search type games since $\epsilon(\mathsf{ID}) \geq 0$ is guaranteed by definition (see Footnote 2).

## 2.2 Artificial Abort

We then move to explain in several steps how Waters [30] resolved the above problem by introducing the artificial abort step. First, observe that if $\gamma(\mathsf{ID}) = \gamma$ holds for some fixed $\gamma \geq 1/\mathsf{poly}$ for all $\mathsf{ID}$, the above naive reduction works. This is because we have the following which is non-negligible:

$$\sum_{\mathsf{ID}} \gamma(\mathsf{ID}) p(\mathsf{ID}) \epsilon(\mathsf{ID}) = \gamma \sum_{\mathsf{ID}} p(\mathsf{ID}) \epsilon(\mathsf{ID}) = \gamma\epsilon.$$

We then move to the more realistic setting where $\gamma(\mathsf{ID})$ varies with $\mathsf{ID}$. Here, we still assume that $\gamma(\mathsf{ID}) \geq \gamma_{\min}$ holds for all $\mathsf{ID}$ and for some fixed $\gamma_{\min} \geq 1/\mathsf{poly}$. For the sake of explanation, we also introduce a simplifying assumption that $\gamma(\mathsf{ID})$ can be computed efficiently given $\mathsf{ID}$. In this setting, we can make the reduction work by introducing an additional abort step (i.e., artificial abort). Namely, after having successfully completed the simulation against the adversary, the simulator evaluates $\gamma(\mathsf{ID})$ based on the sequence of queries $\mathsf{ID}$. It then aborts with probability $1 - \gamma_{\min}/\gamma(\mathsf{ID})$ and outputs a random bit. Then, the probability of the simulation not aborting is the same for all $\mathsf{ID}$, namely, $\gamma_{\min}$. We therefore can use the above analysis to conclude that the advantage of the final adversary is $\gamma_{\min}\epsilon$, which is non-negligible.

However, in reality, we do not know how to compute $\gamma(\mathsf{ID})$ efficiently. What Waters [30] did instead is to approximate the value of $\gamma(\mathsf{ID})$ by the Monte Carlo method. The simulator repeatedly chooses simulation randomness, sees if each randomness leads to a successful simulation, and uses the fraction of randomness that leads to a successful simulation as an approximation for $\gamma(\mathsf{ID})$. We do not give details of the analysis by [30] further, since it is irrelevant to the overview. We just note that the Monte Carlo method is expensive and the approximation needs time proportional to $O(Q^2/\epsilon^2)$ to compute.

## 2.3 Accuracy of Approximation

Let us discuss how the accuracy of the approximation $\gamma(\mathsf{ID})$ affects the reduction. We note that our explanation here is different from the analysis by [30] and is an extension of the analysis by Bellare and Ristenpart [5]. For the sake of easier exposition, we first show our general analysis and then explain the analysis by [5] as a special case. Let us assume that $\gamma(\mathsf{ID})$ can be approximated efficiently and deterministically. We denote the approximation for $\gamma(\mathsf{ID})$ by $\widetilde{\gamma}(\mathsf{ID})$. At the end of the simulation, the reduction algorithm aborts and outputs a random bit with probability $1 - \gamma_{\min}/\widetilde{\gamma}(\mathsf{ID})$, with the intention of making the abort probability as independent of $\mathsf{ID}$ as possible. We then discuss the advantage of the adversary. Since we have just changed the abort probability, we can see that the advantage of the reduction algorithm is obtained by replacing $\gamma(\mathsf{ID})$ in Eq. (1) with $\gamma_{\min} \cdot \gamma(\mathsf{ID})/\widetilde{\gamma}(\mathsf{ID})$, which is the probability that the reduction algorithm does not abort conditioned on the sequence of the identities in the simulation is $\mathsf{ID}$. Namely, the advantage is

$$\sum_{\mathsf{ID}} \gamma_{\min} \cdot \left(\frac{\gamma(\mathsf{ID})}{\widetilde{\gamma}(\mathsf{ID})}\right) \cdot p(\mathsf{ID})\epsilon(\mathsf{ID}) = \gamma_{\min} \cdot \left(\sum_{\mathsf{ID}} (1 + \Delta(\mathsf{ID})) \cdot p(\mathsf{ID})\epsilon(\mathsf{ID})\right),$$

where we define $\Delta(\mathsf{ID}) := \gamma(\mathsf{ID})/\widetilde{\gamma}(\mathsf{ID}) - 1$. In the following, we will argue that if $\Delta(\mathsf{ID})$ is sufficiently small, we can give a useful lower bound for the above quantity. Toward this goal, we assume $-\overline{\Delta} \le \Delta(\mathsf{ID}) \le \overline{\Delta}$ and continue the analysis. We have

$$\begin{aligned}
&\sum_{\mathsf{ID}} (1 + \Delta(\mathsf{ID})) \cdot p(\mathsf{ID})\epsilon(\mathsf{ID}) \\
&\quad = \epsilon + \sum_{\mathsf{ID}} \Delta(\mathsf{ID}) p(\mathsf{ID})\epsilon(\mathsf{ID}) \\
&\quad \ge \epsilon + \sum_{\mathsf{ID} \text{ s.t. } \epsilon(\mathsf{ID}) \ge 0} (-\overline{\Delta}) \cdot p(\mathsf{ID})\epsilon(\mathsf{ID}) + \sum_{\mathsf{ID} \text{ s.t. } \epsilon(\mathsf{ID}) < 0} \overline{\Delta} \cdot p(\mathsf{ID})\epsilon(\mathsf{ID}) \\
&\quad \ge \epsilon - 2\overline{\Delta},
\end{aligned}$$

where the first line uses $\sum_{\mathsf{ID}} p(\mathsf{ID})\epsilon(\mathsf{ID}) = \epsilon$ and the third line uses $\sum_{\mathsf{ID} \text{ s.t. } \epsilon(\mathsf{ID}) \ge 0} p(\mathsf{ID})\epsilon(\mathsf{ID}) \le 1$ and $\sum_{\mathsf{ID} \text{ s.t. } \epsilon(\mathsf{ID}) < 0} p(\mathsf{ID})\epsilon(\mathsf{ID}) \ge -1$. This analysis shows that if $\overline{\Delta} < \epsilon/3$, we have that the overall advantage of the reduction algorithm is at least $\gamma_{\min}\epsilon/3$, which is non-negligible. Recalling the definition of $\overline{\Delta}$, this means that for the reduction to work, it suffices to approximate $\gamma(\mathsf{ID})$ within an additive error no greater than $\gamma_{\min}\epsilon/3$.

We then move to explain the idea of Bellare and Ristenpart [5] as a special case of the above reduction strategy. We can regard their reduction algorithm as a special case of the above reduction, where the approximation $\widetilde{\gamma}(\mathsf{ID})$ for $\gamma(\mathsf{ID})$ is always set to be $\gamma_{\min}$, regardless of $\mathsf{ID}$. This means that the reduction algorithm never artificially aborts, since $1 - \gamma_{\min}/\widetilde{\gamma}(\mathsf{ID}) = 0$. As we have discussed above, we need to have $\overline{\Delta} < \epsilon/3$, which implies that $(1-\epsilon/3)\gamma_{\min} \le \gamma(\mathsf{ID}) \le (1+\epsilon/3)\gamma_{\min}$ for all $\mathsf{ID}$. They achieve this condition by finding a clever choice of parameters. We defer the detail to the next subsection.

## 2.4 Simulation Method of Bellare and Ristenpart [5]

To explain their idea, we have to dive into details of how $\gamma(\mathsf{ID})$ is defined for a sequence of queries $\mathsf{ID}$. Recall that in the security proof using the partitioning technique, we divide the identity space into controlled and uncontrolled sets based on a secret randomness $K$. In the security proof by [5,30], they divide the identity space by the following (partitioning) function

$$\mathsf{F}_{\mathsf{Wat}}(K, \mathsf{ID}) = \begin{cases} 0 & \text{(Meaning“uncontrolled”)} \quad \text{if} \quad K_0 + \sum_{i:\mathsf{ID}_i=1} K_i = 0 \\ 1 & \text{(Meaning“controlled”)} \quad \text{if} \quad K_0 + \sum_{i:\mathsf{ID}_i=1} K_i \ne 0. \end{cases}, \qquad (2)$$

where $K = (K_0, K_1 \dots, K_\ell)$ is the secret randomness chosen as $K_0 \xleftarrow{\$} [-\ell N, 0]$ and $K_i \xleftarrow{\$} [0, N]$ for $i \in [\ell]$, $\ell$ denotes the binary length of identities, and $\mathsf{ID}_i$ denotes the $i$-th bit of an identity $\mathsf{ID} \in \{0,1\}^\ell$. We will explain how they determine the parameter $N$ later. Recall that $\gamma(\mathbf{ID})$ is the probability that the simulation is successful. Namely, this is the probability that $\mathsf{ID}^*$ falls into the uncontrolled set and all of $\mathsf{ID}^{(1)}, \dots, \mathsf{ID}^{(Q)}$ fall into the controlled set. Denoting the event that $\mathsf{F}_{\mathsf{Wat}}(K, \mathsf{ID}^*) = 0$ holds by $\mathsf{E}^*$ and the event that $\mathsf{F}_{\mathsf{Wat}}(K, \mathsf{ID}^{(j)}) = 0$ holds for $j \in [Q]$ by $\mathsf{E}^{(j)}$, we have

$$\begin{aligned} \gamma(\mathbf{ID}) &= \Pr[\mathsf{E}^* \wedge \neg\mathsf{E}^{(1)} \cdots \wedge \neg\mathsf{E}^{(Q)}] \\ &= \Pr[\mathsf{E}^*] - \Pr[\mathsf{E}^* \wedge (\mathsf{E}^{(1)} \vee \cdots \vee \mathsf{E}^{(Q)})] \\ &= \Pr[\mathsf{E}^*] - \Pr[(\mathsf{E}^* \wedge \mathsf{E}^{(1)}) \vee \cdots \vee (\mathsf{E}^* \wedge \mathsf{E}^{(Q)})], \end{aligned} \tag{3}$$

where the probability is taken over the choice of $K$.

Since it is straightforward to see $\Pr[\mathsf{E}^*] = 1/(\ell N + 1)$, getting approximation for $\gamma(\mathbf{ID})$ boils down to getting approximation for $\Pr[(\mathsf{E}^* \wedge \mathsf{E}^{(1)}) \vee \cdots \vee (\mathsf{E}^* \wedge \mathsf{E}^{(Q)})]$. They use the union bound to upper bound the term and give a trivial lower bound 0, which results in the following inequality:

$$\Pr[\mathsf{E}^*] - \underbrace{\sum_{j \in [Q]} \Pr[\mathsf{E}^* \wedge \mathsf{E}^{(j)}]}_{= \mathrm{Approximation error}} \le \gamma(\mathbf{ID}) \le \Pr[\mathsf{E}^*]. \tag{4}$$

Recall that they use fixed $\gamma_{\min}$ as an approximation for $\gamma(\mathbf{ID})$ for all $\mathbf{ID}$ and for the reduction to work, the approximation should be within additive error of $\gamma_{\min}\epsilon/3$. To achieve this guarantee, they adjust the parameter so that the approximation error term $\sum_{j \in [Q]} \Pr[\mathsf{E}^* \wedge \mathsf{E}^{(j)}]$ is as small as possible. Let us introduce the parameter $\delta$, which is defined as $\delta := \Pr[\mathsf{E}^*]$. We can easily see that $\delta$ can be controlled by adjusting the parameter $N$ and $\Pr[\mathsf{E}^{(j)}] = \delta$ for $j \in [Q]$ holds. For the sake of simplicity of the explanation, we introduce an oversimplifying assumption that these events are pair-wise independent, meaning that $\Pr[\mathsf{E}^{(j)} \wedge \mathsf{E}^*] = \delta^2$ and $\Pr[\mathsf{E}^{(j)} \wedge \mathsf{E}^{(k)}] = \delta^2$. We then upper bound the error term as

$$\sum_{j \in [Q]} \Pr[\mathsf{E}^* \wedge \mathsf{E}^{(j)}] \le Q\delta^2.$$

What remains is to choose $\gamma_{\min}$ and $\delta$ so that $\gamma_{\min} \le \delta - Q\delta^2$ and $Q\delta^2 \le \gamma_{\min}\epsilon/3$ hold. The latter inequality implies $Q\delta^2 \lll \gamma_{\min}$ and the former then implies that we can take $\gamma_{\min} = \delta/2$ for example. Then, the latter implies $Q\delta^2 \le \delta\epsilon/6$, which in turn implies $\delta \le \epsilon/6Q$. Therefore, we set $\delta = \Theta(\epsilon/Q)$ and then the advantage of the reduction algorithm is $\gamma_{\min}\epsilon/3 = \Theta(\delta\epsilon) = \Theta(\epsilon^2/Q)$.[7]

[7] Due to the simplifying assumption, the bound here does not exactly correspond to that given in [5]. More formally, we have an extra cost of $O(1/\ell)$ in the final advantage. Similar remark applies to other analyses that appear in the overview.

## 2.5 More Sophisticated Approximation

Our idea to improve the reduction algorithms of previous works [5,30] is to approximate $\gamma(\mathsf{ID})$ by a more sophisticated analysis. In particular, we approximate the term $\Pr[(\mathsf{E}^* \wedge \mathsf{E}^{(1)}) \vee \cdots \vee (\mathsf{E}^* \wedge \mathsf{E}^{(Q)})]$ in Eq. (3) by Bonferroni's inequalities[8], rather than the union bound. Namely, we have

$$\sum_{j \in [Q]} \Pr[\mathsf{E}_2^{(j)}] - \sum_{1 \le j < k \le Q} \Pr[\mathsf{E}_2^{(j)} \wedge \mathsf{E}_2^{(k)}] \le \Pr[\mathsf{E}_2^{(1)} \vee \cdots \vee \mathsf{E}_2^{(Q)}] \le \sum_{j \in [Q]} \Pr[\mathsf{E}_2^{(j)}],$$

where we denote $\mathsf{E}_2^{(j)} := \mathsf{E}^* \wedge \mathsf{E}^{(j)}$ for notational convenience in the above. Plugging the above equation into Eq. (3), we obtain

$$\begin{aligned} \Pr[\mathsf{E}^*] - \sum_{j \in [Q]} \Pr[\mathsf{E}^* \wedge \mathsf{E}^{(j)}] \le \gamma(\mathsf{ID}) \le \Pr[\mathsf{E}^*] - \sum_{j \in [Q]} \Pr[\mathsf{E}^* \wedge \mathsf{E}^{(j)}] \\ + \underbrace{\sum_{1 \le j < k \le Q} \Pr[\mathsf{E}^* \wedge \mathsf{E}^{(j)} \wedge \mathsf{E}^{(k)}]}_{=} \text{Approximationerror}, \end{aligned} \tag{5}$$

where we use $\Pr[\mathsf{E}_2^{(j)} \wedge \mathsf{E}_2^{(k)}] = \Pr[\mathsf{E}^* \wedge \mathsf{E}^{(j)} \wedge \mathsf{E}^{(k)}]$. We then use $\Pr[\mathsf{E}^*] - \sum_{j \in [Q]} \Pr[\mathsf{E}^* \wedge \mathsf{E}^{(j)}]$ as the approximation for $\gamma(\mathsf{ID})$, i.e., $\widetilde{\gamma}(\mathsf{ID}) := \Pr[\mathsf{E}^*] - \sum_{j \in [Q]} \Pr[\mathsf{E}^* \wedge \mathsf{E}^{(j)}]$. While for this to be useful, we have to show that we can efficiently compute $\Pr[\mathsf{E}^* \wedge \mathsf{E}^{(j)}]$, we simply assume this is possible and defer the detail to Sect. 2.6. Now, observe that the approximation error can be bounded by $\sum_{1 \le j < k \le Q} \Pr[\mathsf{E}^* \wedge \mathsf{E}^{(j)} \wedge \mathsf{E}^{(k)}]$. We analyze this term by introducing again an oversimplifying assumption that the events $\mathsf{E}^*, \mathsf{E}^{(1)}, \ldots, \mathsf{E}^{(Q)}$ are 3-wise independent, meaning that any conjunction of 3 of them happens with probability $\delta^3$. Using this, we bound the above approximation error term by $Q(Q-1)\delta^3/2 \le Q^2\delta^3$. By our condition on the approximation error, we need to satisfy

$$Q^2\delta^3 \le \gamma_{\min}\epsilon/3.$$

We also have to set $\gamma_{\min}$ so that it is smaller than the leftmost term in Eq. (5). We have $\sum_{j \in [Q]} \Pr[\mathsf{E}^* \wedge \mathsf{E}^{(j)}] = Q\delta^2$ by our assumption of pairwise independence and thus the condition is equivalent to

$$\gamma_{\min} \le \delta - Q\delta^2.$$

By a similar analysis explained in the previous subsection, we can take $\gamma_{\min} = \delta/2$. We then have $Q^2\delta^3 \le \delta\epsilon/6$, resulting in $\delta \le \epsilon^{1/2}/6Q$. By setting $\delta = \Theta(\epsilon^{1/2}/Q)$, the advantage of the reduction algorithm becomes $\gamma_{\min}\epsilon/3 = \Theta(\delta\epsilon) = \Theta(\epsilon^{1.5}/Q)$, improving the result of [5]. The reason for this improvement is our fine-grained approximation of $\gamma(\mathsf{ID})$ compared to [5] based on Bonferroni's inequality. By representing the approximation error as a higher order polynomial of $\delta$, we can chose a larger $\delta$ (i.e., $\Theta(\epsilon^{0.5}/Q)$ as opposed to $\Theta(\epsilon/Q)$), leading to a better advantage.

[8] Bonferroni's inequalities are the inequalities obtained by cutting off higher order terms from the equality derived from inclusion-exclusion principle. See the full version for the formal statement.

### 2.6 Computing the Probability Efficiently

Two things are missing from the above explanation. First, in the above analysis, we assumed that the events $\mathsf{E}^*$, $\mathsf{E}^{(1)}, \ldots, \mathsf{E}^{(Q)}$ are 3-wise independent. Unfortunately, this assumption is not true. However, we can show that $\Pr[\mathsf{E}^* \wedge \mathsf{E}^{(j)} \wedge \mathsf{E}^{(k)}] = \Theta(\ell^2\delta^3)$ for $j \neq k$, which is still useful for the analysis. We defer the details on how to prove this to the main body of the paper. The other more important detail missing from the above explanation is how to compute $\Pr[\mathsf{E}^* \wedge \mathsf{E}^{(j)}]$ efficiently for $j \in [Q]$. The rest of this subsection will be devoted on explaining how to do it. Let us define $S := \{i \in [\ell] : \mathsf{ID}_i^* = 1\}$ and $T := \{i \in [\ell] : \mathsf{ID}_i^{(j)} = 1\}$. Then, by the definition of $\mathsf{E}^*$ and $\mathsf{E}^{(j)}$, we have

$$\begin{aligned} \Pr[\mathsf{E}^* \wedge \mathsf{E}^{(j)}] &= \Pr\left[K_0 + \sum_{i \in S} K_i = 0 \;\wedge\; K_0 + \sum_{i \in T} K_i = 0\right] \\ &= \frac{1}{\ell N + 1} \cdot \Pr\left[\sum_{i \in S} K_i = \sum_{i \in T} K_i\right], \end{aligned} \tag{6}$$

where the probability is taken over the randomness of $K_0 \xleftarrow{\$} [-\ell N, 0]$ and $K_i \xleftarrow{\$} [0, N]$ for $i \in [\ell]$. Without loss of generality, we can assume that $S \cap T = \emptyset$. Furthermore, we can assume that $S = [n_S]$ and $T = [n_S + 1, n_S + n_T]$, where $n_S = \#S$ and $n_T = \#T$ with $n_S \leq n_T$. Toward computing the above probability, we introduce a function $R$, defined as

$$R_n(\alpha) := \#\left\{0 \leq K_i \leq N \;:\; \sum_{i \in [n]} K_i = \alpha\right\}.$$

Using the notation, we continue the analysis from Eq. (6). We have

$$\begin{aligned} \Pr[\mathsf{E}^* \wedge \mathsf{E}^{(j)}] &= \frac{1}{(\ell N + 1)(N+1)^{n_S + n_T}} \\ &\cdot \#\left\{K_i \in [0, N] \text{ for } i \in [n_S + n_T] : \sum_{i \in [n_S]} K_i = \sum_{i \in [n_T]} K_i\right\} = \frac{1}{(\ell N + 1)(N+1)^{n_S + n_T}} \\ &\cdot \sum_{\alpha=0}^{n_S N} \#\left\{K_i \in [0, N] \text{ for } i \in [n_S + n_T] : \sum_{i \in [n_S]} K_i = \sum_{i \in [n_T]} K_i = \alpha\right\} \\ &= \frac{1}{(\ell N + 1)(N+1)^{n_S + n_T}} \cdot \sum_{\alpha=0}^{n_S N} R_{n_S}(\alpha) R_{n_T}(\alpha). \end{aligned}$$

At this point, the problem of estimating the probability boils down to the problem of computing the summation $\sum_{\alpha=0}^{n_S N} R_{n_S}(\alpha) R_{n_T}(\alpha)$. We emphasize that the algorithm needs to run in at most poly-logarithmic time in $N$. Otherwise, our final reduction algorithm will add an additive overhead $Q \cdot \mathsf{poly}(N) = Q \cdot \mathsf{poly}(Q, 1/\epsilon)$ to the running time, which ruins the merit of having a larger distinguishing advantage for the reduction algorithm compared to [5].

The most natural approach for solving the problem would be to take a dynamic programming approach to compute $R_n(\alpha)$ for $n \in \{n_S, n_T\}$ and $\alpha \in [0, n_T]$ and then compute the summation. This approach is problematic in two-folds: First, computing $R_n(\alpha)$ by dynamic programming requires $\mathsf{poly}(N)$ time, which is too slow. Furthermore, even if $R_\ell(\alpha)$ can be computed efficiently, we have to compute the summation of $n_S N$ terms, which requires $\mathsf{poly}(N)$ time if we follow the straightforward approach. Luckily, there is an elegant solution to the first problem of computing $R_n(\alpha)$ efficiently using the powerful machinery of *generating functions* [31], which is a standard tool in enumerative combinatorics. Furthermore, we can show the equation

$$\sum_{\alpha=0}^{n_S N} R_{n_S}(\alpha) R_{n_T}(\alpha) = R_{n_S+n_T}(n_T N) \tag{7}$$

again using generating functions and therefore the summation can be computed efficiently. We defer the detail of how to compute $R_\ell(\alpha)$ to the main body and explain how to prove the equation here.

Before the proof, let us define a useful notation. For a polynomial $f(\mathsf{Z}) = \sum_i a_i \mathsf{Z}^i$ with $\mathsf{Z}$ being indeterminate, we denote $[\mathsf{Z}^j]f(\mathsf{Z})$ as the $j$-th coefficient of $f(\mathsf{Z})$, namely, $a_j$. We then observe that $R_n(\alpha)$ equals to $[\mathsf{Z}^\alpha](1+\mathsf{Z}+\mathsf{Z}^2+\cdots \mathsf{Z}^N)^n$. This can be seen by expanding the multiplication and observing that to yield the term $\mathsf{Z}^\alpha$, we have to choose $\mathsf{Z}^{K_i}$ from the $i$-th factor so that their sum $K_1 + \cdots K_N$ equals to $\alpha$. We also observe that $R_n(\alpha) = R_n(nN - \alpha)$, which can be seen by comparing the coefficients of the left and right hands of the equality $(1 + \mathsf{Z} + \mathsf{Z}^2 + \cdots \mathsf{Z}^N)^n = \mathsf{Z}^{nN}(1 + \mathsf{Z}^{-1} + \cdots \mathsf{Z}^{-N})^n$. We finally observe that for polynomials $f(\mathsf{Z})$ and $g(\mathsf{Z})$ and an integer $n$ with $n \geq \deg(f)$, we have

$$[\mathsf{Z}^n](f(\mathsf{Z}) \cdot g(\mathsf{Z})\ ) = \sum_{i=0}^{\deg(f)} [\mathsf{Z}^i]f(\mathsf{Z}) \cdot [\mathsf{Z}^{n-i}]g(\mathsf{Z}).$$

Equipped with the observations, we are now ready to prove Eq. (7). We have:

$$\begin{aligned}
\sum_{\alpha=0}^{n_S N} R_{n_S}(\alpha) R_{n_T}(\alpha) &= \sum_{\alpha=0}^{n_S N} R_{n_S}(\alpha) R_{n_T}(n_T N - \alpha) \\
&= \sum_{\alpha=0}^{n_S N} [\mathsf{Z}^\alpha](1 + \mathsf{Z} + \mathsf{Z}^2 + \cdots + \mathsf{Z}^N)^{n_S} \cdot [\mathsf{Z}^{n_T N - \alpha}](1 + \mathsf{Z} + \mathsf{Z}^2 + \cdots + \mathsf{Z}^N)^{n_T} \\
&= [\mathsf{Z}^{n_T N}](1 + \mathsf{Z} + \mathsf{Z}^2 + \cdots + \mathsf{Z}^N)^{n_S+n_T} \\
&= R_{n_S+n_T}(n_T N)
\end{aligned}$$

as desired.

## 2.7 Partitioning for Lattices

From here on, we shift our focus and analyze different partitioning strategies. Let us start with a variant of $\mathsf{F}_{\mathsf{Wat}}$ defined in Eq. (2). While the partitioning strategy specified by the function $\mathsf{F}_{\mathsf{Wat}}$ can in principle be used in the lattice setting, it requires a super-polynomial size modulus $q$ for the underlying scheme, since $q$ should be larger than the parameter $N$, which is polynomially related to $Q$ (and $1/\epsilon$). To refrain from using a superpolynomial modulus $q$, Boyen [11] proposed a variant of Waters' partitioning function suitable for the lattice setting, later used for proving the security of ABB IBE [3]. Our formal analysis reveals that their analysis suffers from a large reduction loss of $\gamma_{\mathsf{min}}\epsilon = O(\epsilon^3/Q^2)$. We show that a more natural adaptation of the Waters' partitioning function to the lattice setting gives us a reduction with $\gamma_{\mathsf{min}}\epsilon = O(\epsilon^2/Q)$, even with the Bellare-Ristenpart-style analysis. This variant is essentially identical to Boyen's partitioning function but fixing some superfluous components. Importantly, this is only a superficial difference and keeps the efficiency of the original ABB IBE unchanged. We then show that this can be further improved to $\gamma_{\mathsf{min}}\epsilon = O(\epsilon^{1.5}/Q)$ by our analysis using Bonferroni's inequality. Lastly, with a more noticeable tweak to the partitioning function, we can achieve $\gamma_{\mathsf{min}}\epsilon = O(\epsilon^{1+1/d}/Q)$ for an arbitrary $d > 2$ or even $\gamma_{\mathsf{min}}\epsilon = O(\epsilon/Q)$, where this tweak results in slightly modifying the ABB IBE.

More concretely, we define our partitioning function $\mathsf{F}_{\mathsf{ParWat}}(K, \mathsf{x})$ as follows:

$$\mathsf{F}_{\mathsf{ParWat}}(K, \mathsf{ID}) = \begin{cases} 0 & K_0 + \sum_{i:\mathsf{ID}_i=1} K_i = \mathbf{0}_c \pmod q \\ 1 & \text{otherwise} \end{cases},$$

where $K = (K_0, K_1, \ldots, K_\ell) \in (\mathbb{Z}_q^c)^\ell$ and $c$ is a parameter that will be defined later. $K_0, K_1, \ldots, K_\ell$ are chosen uniformly at random from $\mathbb{Z}_q^c$. We note that here, $q$ is a small polynomially bounded prime.

**Bellare-Ristenpart-Style Analysis.** We then analyze $\gamma(\mathsf{ID})$. Let us start with a Bellare-Ristenpart-style analysis, where we use a fixed value $\gamma_{\mathsf{min}}$ for the estimation of $\gamma(\mathsf{ID})$. Denoting the event that $\mathsf{F}_{\mathsf{ParWat}}(K, \mathsf{ID}^*) = 0$ holds by $\mathsf{E}^*$ and the event that $\mathsf{F}_{\mathsf{ParWat}}(K, \mathsf{ID}^{(j)}) = 0$ holds for $j \in [Q]$ by $\mathsf{E}^{(j)}$, Eq. (4) can be shown to hold by the same analysis as in Sect. 2.4. We then proceed to bound the error term $\sum_{j\in[Q]} \Pr[\mathsf{E}^* \wedge \mathsf{E}^{(j)}]$. While this requires a bit of work for the case of $\mathsf{F}_{\mathsf{Wat}}$, it is straightforward here. Concretely, we have

$$\Pr[\mathsf{E}^*] = \frac{1}{q^c}, \qquad \Pr[\mathsf{E}^* \wedge \mathsf{E}^{(j)}] = \frac{1}{q^{2c}}$$

for all $j \in [Q]$. Here, the former equation is straightforward to see by the fact that $K_0$ is distributed uniformly at random over $\mathbb{Z}_q^c$. To see the latter equation, we observe

$$K_0 + \sum_{i:\mathsf{ID}_i=1} K_i = (1, \mathsf{ID}) \cdot (K_0^\top, K_1^\top, \ldots, K_\ell^\top)^\top,$$

where we regard $\mathsf{ID} \in \{0,1\}^\ell$ as a row vector with dimension $\ell$ and $(K_0^\top, K_1^\top, \ldots, K_\ell^\top)^\top \in \mathbb{Z}_q^{(\ell+1)\times c}$ is a matrix obtained by regarding each $K_i$ as a row vector and concatenating them vertically. When $\mathsf{ID}^*$ and $\mathsf{ID}^{(j)}$ are distinct, $(1, \mathsf{ID}^*)$ and $(1, \mathsf{ID}^{(j)})$ are linearly independent and thus the pair $(K_0 + \sum_{i:\mathsf{ID}_i^*=1} K_i, K_0 + \sum_{i:\mathsf{ID}_i^{(j)}=1} K_i)$ are distributed uniformly at random over $\mathbb{Z}_q^{2c}$, implying the above equation.

From the above analysis, we can see that the error term in Eq. (4) can be bounded by $Q \cdot q^{-2c}$. It remains to choose $\gamma_{\mathsf{min}}$ and $c$ so that $\gamma_{\mathsf{min}} \leq q^{-c} - Qq^{-2c}$ and $Q \cdot q^{-2c} \leq \gamma_{\mathsf{min}}\epsilon/3$ hold. Combining these inequalities, we have $Q \cdot q^{-2c} \leq q^{-c}\epsilon/3$. To satisfy this, we choose $c = \log_q(3Q/\epsilon)$, which leads to the reduction cost $\gamma_{\mathsf{min}}\epsilon = \Theta(\epsilon^2/Q)$.

**Our Improved Analysis.** We then move to explain our finer-grained analysis using Bonferroni's inequality. Now, by the same analysis as Sect. 2.5 using Bonferroni's inequality, we can derive Eq. (5). We then set $\widetilde{\gamma}(\mathbf{ID}) := \Pr[\mathsf{E}^*] - \sum_{j \in [Q]} \Pr[\mathsf{E}^* \wedge \mathsf{E}^{(j)}] = q^{-c} - Qq^{-2c}$. Unlike Sect. 2.5, we can directly compute $\widetilde{\gamma}(\mathbf{ID})$. We then bound the error term $\sum_{j,k} \Pr[\mathsf{E}^* \wedge \mathsf{E}^{(j)} \wedge \mathsf{E}^{(k)}]$ in Eq. (5). We have

$$\Pr[\mathsf{E}^* \wedge \mathsf{E}^{(j)} \wedge \mathsf{E}^{(k)}] = q^{-3c}$$

for each $j, k$, since we can prove that the vectors $(1, \mathsf{ID}^*)$, $(1, \mathsf{ID}^{(j)})$, and $(1, \mathsf{ID}^{(k)})$ are linearly independent for mutually distinct $\mathsf{ID}^*$, $\mathsf{ID}^{(j)}$, and $\mathsf{ID}^{(k)}$. This allows us to bound the error term by $Qq^{-3c}$. It remains to choose $\gamma_{\mathsf{min}}$ and $c$ so that $\gamma_{\mathsf{min}} \leq q^{-c} - Qq^{-2c}$ and $Q^2 \cdot q^{-3c} \leq \gamma_{\mathsf{min}}\epsilon/3$ hold. Combining these inequalities, we have $Q \cdot q^{-3c} \leq q^{-c}\epsilon/3$. To satisfy this, we choose $c = \log_q(3Q/\sqrt{\epsilon})$, which leads to the reduction cost $\gamma_{\mathsf{min}}\epsilon = \Theta(\epsilon^{1.5}/Q)$. This improves the bound based on the Bellare-Ristenpart-style analysis by a factor of $\epsilon^{1/2}$.

**Going Beyond $\gamma_{\mathsf{min}}\epsilon = O(\epsilon^{1.5}/Q)$.** A natural question would be whether we can go beyond $\gamma_{\mathsf{min}}\epsilon = O(\epsilon^{1.5}/Q)$ using Bonferroni's inequality with higher order terms. This could be possible if we had $\Pr[\mathsf{E}^* \wedge \mathsf{E}^{(j_1)} \wedge \cdots \wedge \mathsf{E}^{(j_{d-1})}] = q^{-cd}$ for $d \geq 4$. However, unfortunately, this does not hold already for $d = 4$. We therefore change the function a bit so that

$$\mathsf{F}_{\mathsf{ParWat}}(K, \mathsf{ID}) = \begin{cases} 0 & K_0 + \sum_{i : h_{d\text{-}\mathsf{wise}}(\mathsf{ID})_i = 1} K_i = \mathbf{0}_c \pmod q \\ 1 & \text{otherwise} \end{cases},$$

where the only change we add is that we hash the identity by a hash function $h_{d\text{-}\mathsf{wise}} : \{0,1\}^\ell \to \{0,1\}^{L_d}$. For the hash function, we require the property that $(1, h_{d\text{-}\mathsf{wise}}(\mathsf{ID}_1)), \ldots, (1, h_{d\text{-}\mathsf{wise}}(\mathsf{ID}_d))$ are linearly independent over $\mathbb{Z}_q$ for mutually distinct $\mathsf{ID}_1, \ldots, \mathsf{ID}_d$. Let us postpone the construction of such a hash function to the main body. Assuming that we have such a hash function, we are now able to prove $\Pr[\mathsf{E}^* \wedge \mathsf{E}^{(j_1)} \wedge \cdots \wedge \mathsf{E}^{(j_{d-1})}] = q^{-cd}$ by the same linear algebraic discussion we have done. We can then approximate the value of $\gamma(\mathbf{ID})$ within error $Q^{d-1}q^{-cd}$ and this leads to the improved reduction cost of $\gamma_{\mathsf{min}}\epsilon = \Theta(\epsilon^{1+1/(d-1)}/Q)$. Furthermore, by setting $d = \omega(1)$, we can completely eliminate the dependence of $\gamma_{\mathsf{min}}$ on $\epsilon$ to achieve $\gamma_{\mathsf{min}} = O(1/Q)$, which leads to $\gamma_{\mathsf{min}}\epsilon = O(\epsilon/Q)$. Note that the above change in the partitioning function increases the size of the key $K$, since the output length $L_d$ of $h_{d\text{-}\mathsf{wise}}$ is about $d\ell$, which is $d$ times longer than the input.

## 2.8 Partitioning Based on Substring Matching

Here, we demonstrate that our technique can lead to tighter analysis also for the partitioning based on the substring matching [27], which has been a useful tool in constructing adaptively secure IBEs and VRFs [6,8,12,24,26,33]. Here, we focus on the application to IBE, though our analysis is applicable to VRF as well, as is done in Sect. 6. To describe the partitioning function, we introduce an error correcting code $\mathsf{Encode} : \{0,1\}^\ell \to \{0,1\}^n$ with relative distance $0 < c < 1/2$ and output length $n$.[9] Then, the identity space $\{0,1\}^\ell$ is partitioned as follows:

$$\mathsf{F}_{\mathsf{SSM}}(K, \mathsf{ID}) = \begin{cases} 0 & \text{if} \quad \sigma_i = \mathsf{Encode}(\mathsf{ID})_{I_i} \quad \forall i \in [\eta] \\ 1 & \text{otherwise} \end{cases}, \tag{8}$$

where the secret information $K$ is in the form of $K = \{(I_i, \sigma_i)\}_{i\in[\eta]}$ and $I_i \in [n]$ and $\sigma_i \in \Sigma$ for all $i \in [n]$, with $\eta$ being a parameter that will be chosen later. We choose $I = (I_i)_{i\in[\eta]}$ so that $I$ constitutes a random subset of $[n]$ and $\sigma_i \xleftarrow{\$} \{0,1\}$ for each $i$.

**Bellare-Ristenpart-Style Analysis.** We then analyze $\gamma(\mathsf{ID})$. Let us start with a Bellare-Ristenpart-style analysis. Denoting the event that $\mathsf{F}_{\mathsf{SSM}}(K, \mathsf{ID}^*) = 0$ holds by $\mathsf{E}^*$ and the event that $\mathsf{F}_{\mathsf{SSM}}(K, \mathsf{ID}^{(j)}) = 0$ holds for $j \in [Q]$ by $\mathsf{E}^{(j)}$, Eq. (4) can be shown to hold by the same analysis as in Sect. 2.4. We then proceed to bound the error term $\sum_{j\in[Q]} \Pr[\mathsf{E}^* \wedge \mathsf{E}^{(j)}]$. Noting that it is straightforward to see $\Pr[\mathsf{E}^*] = 2^{-\eta}$, we evaluate $\Pr[\mathsf{E}^* \wedge \mathsf{E}^{(j)}]$:

$$\begin{aligned}
\Pr[\mathsf{E}^* \wedge \mathsf{E}^{(j)}] &= \Pr\left[\left(\mathsf{Encode}(\mathsf{ID}^*)_{I_i} = \sigma_i \ \forall i \in [\eta]\right) \wedge I \subseteq \underbrace{\{k : \mathsf{Encode}(\mathsf{ID}^*)_k = \mathsf{Encode}(\mathsf{ID}^{(j)})_k\}}_{:=J}\right] \\
&= \Pr\left[\left(\mathsf{Encode}(\mathsf{ID}^*)_{I_i} = \sigma_i \ \forall i \in [\eta']\right) \ \middle|\ I \subseteq J\}\right] \cdot \Pr[I \subseteq J\}] \\
&= 2^{-\eta} \cdot \prod_{i=0}^{\eta-1} \left(\frac{\#J - i}{n-i}\right) \\
&\le 2^{-\eta} \cdot \prod_{i=0}^{\eta-1} \left(\frac{(1-c)n - i}{n-i}\right) \\
&\le 2^{-\eta}(1-c)^\eta
\end{aligned} \tag{9}$$

where the third equation follows from $\sigma_i \xleftarrow{\$} \{0,1\}$ and by the fact that $I$ is a random subset of $[n]$ and the first inequality follows from the fact that the relative distance of $\mathsf{Encode}$ is $c$, which in turn implies $J \le (1-c)n$. From the above analysis, we can see that the error term in Eq. (4) can be bounded by $Q \cdot 2^{-\eta}(1-c)^\eta$. It remains to choose $\gamma_{\mathsf{min}}$ and $\eta$ so that $\gamma_{\mathsf{min}} \le 2^{-\eta} - Q2^{-\eta}(1-c)^\eta$ and $Q \cdot 2^{-\eta}(1-c)^\eta \le \gamma_{\mathsf{min}}\epsilon/3$ hold. Combining these inequalities, we have $Q \cdot 2^{-\eta}(1-c)^\eta \le 2^{-\eta}\epsilon/3$. To satisfy this, we choose $\eta = \log_{1/1-c}(3Q/\epsilon)$, which leads to the reduction cost $\gamma_{\mathsf{min}}\epsilon = \Theta(\epsilon^{1+\mu}/Q^\mu)$, where $\mu = 1/(\log 1/(1-c))$. Note that we have $\mu > 1$ and by approaching $c$ to $1/2$, it is possible to make $\mu$ approach to 1 as closely as one wants. The security proofs for many IBE and

[9] Many previous works (e.g., [8,12]) primarily focus on the encoding function and call it "admissible hash". In this paper, we use the term partitioning based on substring matching following [6,26].

VRF schemes [21,24,26,33] essentially depend on the above analysis and derive the above reduction cost.

**Our Improved Analysis.** We then move to explain our finer-grained analysis. Now, by the same analysis as Sect. 2.5 using Bonferroni's inequality, we can derive Eq. (5). We then set $\widetilde{\gamma}(\mathbf{ID}) := \Pr[\mathsf{E}^*] - \sum_{j \in [Q]} \Pr[\mathsf{E}^* \wedge \mathsf{E}^{(j)}]$. Similarly to Sect. 2.7, it is straightforward to compute $\widetilde{\gamma}(\mathbf{ID})$ efficiently, since we can use Eq. (9) to compute each of $\Pr[\mathsf{E}^* \wedge \mathsf{E}^{(j)}]$. We then bound the error term $\sum_{j,k} \Pr[\mathsf{E}^* \wedge \mathsf{E}^{(j)} \wedge \mathsf{E}^{(k)}]$ in Eq. (5). For doing that, we need an extra property for $\mathsf{Encode}$ that we call the *small triple overlap property*. Namely, we need for an arbitrary but mutually distinct $\mathsf{x}_1, \mathsf{x}_2, \mathsf{x}_3 \in \{0,1\}^\ell$ to satisfy,

$$\#\left\{\iota \in [n] : \mathsf{Encode}(\mathsf{x}_1)_\iota = \mathsf{Encode}(\mathsf{x}_2)_\iota = \mathsf{Encode}(\mathsf{x}_3)_\iota\right\} \le (1-c)^2 n.$$

We defer the construction of such code to the end of this subsection and continue the analysis. We now bound each of $\Pr[\mathsf{E}^* \wedge \mathsf{E}^{(j)} \wedge \mathsf{E}^{(k)}]$:

$$\begin{aligned}
&\Pr[\mathsf{E}^* \wedge \mathsf{E}^{(j)} \wedge \mathsf{E}^{(k)}] \\
&\quad = \Pr\Bigg[ (\mathsf{Encode}(\mathsf{ID}^*)_{I_i} = \sigma_i \ \forall i \in [\eta]) \wedge I \\
&\qquad \subseteq \underbrace{\{\iota : \mathsf{Encode}(\mathsf{ID}^*)_\iota = \mathsf{Encode}(\mathsf{ID}^{(j)})_\iota = \mathsf{Encode}(\mathsf{ID}^{(k)})_\iota\}}_{:=L} \Bigg] \\
&\quad = \Pr\left[(\mathsf{Encode}(\mathsf{x})_{I_i} = \sigma_i \ \forall i \in [\eta]) \mid I \subseteq L\right] \cdot \Pr[I \subseteq L] \\
&\quad = 2^{-\eta} \cdot \prod_{i=0}^{\eta-1} \left( \frac{\#L - i}{n - i} \right) \\
&\quad \le 2^{-\eta} \cdot \prod_{i=0}^{\eta-1} \left( \frac{(1-c)^2 n - i}{n - i} \right) \\
&\quad \le 2^{-\eta} (1-c)^{2\eta}.
\end{aligned}$$

where we use the small triple overlap property in the first inequality. From the above analysis, we can see that the error term in Eq. (5) can be bounded by $Q^2 2^{-\eta}(1-c)^{2\eta}$. It remains to choose $\gamma_{\mathsf{min}}$ and $\eta$ so that $\gamma_{\mathsf{min}} \le 2^{-\eta} - Q \cdot 2^{-\eta}(1-c)^\eta$ and $Q^2 \cdot 2^{-\eta}(1-c)^{2\eta} \le \gamma_{\mathsf{min}}\epsilon/3$ hold. From the both inequalities, we can derive $Q^2(1-c)^{2\eta} \le \epsilon/3$. We then take $\eta = \log_{1/1-c}(3Q/\sqrt{\epsilon})$, which leads to the reduction cost $\gamma_{\mathsf{min}}\epsilon = \Theta(\epsilon^{1+\mu/2}/Q^\mu)$, where $\mu = \log 1/(1-c)$. Note that we can make $\mu$ approach 1 as closely as one wants by approaching $c$ to 1/2. This improves the reduction cost of previous works $\gamma_{\mathsf{min}}\epsilon = \Theta(\epsilon^{1+\mu}/Q^\mu)$ by a factor of $\epsilon^{\mu/2}$. Again, the reason why the improvement is possible is that we use the finer-grained approximation of $\gamma(\mathbf{ID})$ using Bonferroni's inequality to represent the error terms as a higher order polynomial of $(1-c)$. This allows us to take $\eta$ smaller, which leads to better advantage.

**Instantiating** $\mathsf{Encode}$**.** We then discuss how to instantiate $\mathsf{Encode}$. Unfortunately, we do not know explicit constructions of a function with the small triple overlap property, where an explicit construction refers to a deterministic algorithm that takes as input $n$, $\ell$, and $\mathsf{ID}$ and outputs $\mathsf{Encode}(\mathsf{ID})$. Instead, we observe that a randomly chosen 3-wise independent hash function satisfies this property with overwhelming probability under specific parameter settings. Therefore, in applications to IBEs/VRFs, we choose a random 3-wise independent hash function and append it to the public parameters as the description of $\mathsf{Encode}$. The description size of $\mathsf{Encode}$ is much shorter than any other part of the parameters in our application and does not harm the efficiency of the construction. In addition, we observe that this does not harm the security of the constructions either. We defer to the details to the main body.

**Polynomial-Size Alphabet Variant.** Finally, we discuss the variant of the function $\mathsf{F}_{\mathsf{SSM}}$ with polynomial-size alphabets, where the underlying encoding function has codewords with a polynomial-size alphabet. Namely, we have $\mathsf{Encode} : \{0,1\}^\ell \to \Sigma^n$ for a polynomial size $\Sigma$, rather than $\Sigma = \{0,1\}$. While many previous works primarily focused on binary encoding functions when constructing IBEs/VRFs [8,12,24,33], Kohl [26] showed that using encoding functions with a polynomial-size alphabet can be useful when constructing VRF schemes with a compact proof size. While she uses Reed-Solomon code, we replace it with a 3-wise independent hash function. Since a 3-wise independent hash function achieves larger relative distance $c$ than the Reed-Solomon code (w.h.p), using it is quite beneficial. We can improve the overall parameter size of her construction even if we have to add the description of $\mathsf{Encode}$ to the public parameter as the description size is small. Furthermore, we can also improve the reduction cost because of the larger relative distance of the 3-wise independent hash function. On top of the improvement described above, we can further apply our finer-grained analysis to the variant with polynomial-size alphabet, since the underlying encoding function satisfies the small triple overlap property. This leads to a tighter analysis that achieves $\gamma_{\mathsf{min}}\epsilon = O(\epsilon^{1.5}/Q)$.

## 3 A Finer Grained Analysis of the Artificial Abort Paradigm

Our main technical contribution is to provide a more fine grained analysis of Bellare and Ristenpart [5] by further relying on the artificial abort paradigm [30]. In this section, we divorce the artificial abort paradigm from security proofs of a particular cryptographic primitive. Instead, we provide a statistical theorem that extracts the essence of the paradigm. Looking ahead, in Sect. 4, we will relate the following statistical theorem to concrete cryptographic primitives using a tool called *partitioning function with approximation*. This allows for a more modular proof of IBE and VRF schemes, as we illustrate in Sects. 5 and 6.

**Theorem 1.** *Let $\mathcal{T}$ be a finite set named the* transcript *space. Let $\mathcal{D} : \{0,1\} \times \{0,1\} \times \mathcal{T} \to [0,1]$ be an arbitrary distribution. Let $\gamma_{\mathsf{min}} > 0$ be a positive real and $\gamma : \mathcal{T} \to [0,1]$ and $\widetilde{\gamma} : \mathcal{T} \to [0,1]$ be functions such that $\gamma(\mathsf{T}) \geq \widetilde{\gamma}(\mathsf{T}) \geq \gamma_{\mathsf{min}}$ for all* transcripts *$\mathsf{T} \in \mathcal{T}$.*

*Consider a distribution* $\mathcal{D}^* : \{0,1\} \times \{0,1\} \times \mathcal{T}$ *defined through the following procedure:*

1. *Sample* $(\mathsf{coin}, \widehat{\mathsf{coin}}, \mathsf{T}) \xleftarrow{\$} \mathcal{D}$.
2. *With probability* $\gamma(\mathsf{T})$*, set* $\mathsf{coin}' \leftarrow \widehat{\mathsf{coin}}$ *and with probability* $1 - \gamma(\mathsf{T})$*, sample a uniformly random* $\mathsf{coin}' \xleftarrow{\$} \{0,1\}$*. The later event is called* $\mathsf{Bad}$*. If* $\neg\mathsf{Bad}$*, it further executes Item 3.*
3. *With probability* $1 - \gamma_{\mathsf{min}}/\widetilde{\gamma}(\mathsf{T})$*, replace* $\mathsf{coin}'$ *with a uniformly random* $\mathsf{coin}' \xleftarrow{\$} \{0,1\}$*. This event is called* $\mathsf{AAbort}$*, short for* artificial abort*.*
4. *Output* $(\mathsf{coin}, \mathsf{coin}', \mathsf{T})$.

*Lastly, define*

$$\epsilon = \left| \Pr_{(\mathsf{coin}, \widehat{\mathsf{coin}}, \mathsf{T}) \xleftarrow{\$} \mathcal{D}} \left[ \widehat{\mathsf{coin}} = \mathsf{coin} \right] - \frac{1}{2} \right| \quad \textit{and} \quad \epsilon^* = \left| \Pr_{(\mathsf{coin}, \mathsf{coin}', \mathsf{T}) \xleftarrow{\$} \mathcal{D}^*} \left[ \mathsf{coin}' = \mathsf{coin} \right] - \frac{1}{2} \right|.$$

*Then, if* $|\gamma(\mathsf{T}) - \widetilde{\gamma}(\mathsf{T})| < \frac{\gamma_{\mathsf{min}}}{3} \cdot \epsilon$ *holds for all transcripts* $\mathsf{T} \in \mathcal{T}$*, we have* $\epsilon^* > \frac{\gamma_{\mathsf{min}}}{3} \cdot \epsilon$*.*

We refer to the full version for the proof. Here, we explain some intuition of the theorem. In the context of security proofs, $\mathsf{coin}$ denotes the random challenge bit sampled by the challenger and $\widehat{\mathsf{coin}}$ denotes the guess output by the adversary $\mathsf{A}$. The advantage of $\mathsf{A}$ is thus $\epsilon$. $\mathsf{Bad}$ denotes the typical event that the reduction fails. For example, in the context of IBE schemes, $\mathsf{Bad}$ can denote the event that the reduction cannot answer the key-extraction query or cannot simulate the challenge ciphertext. In such a case, since the reduction cannot properly simulate the game for $\mathsf{A}$, it will output a random $\mathsf{coin}'$ as $\mathsf{A}$'s output. $\mathsf{AAbort}$ is the more interesting event. In this case, while the reduction is able to simulate $\mathsf{A}$ till the end of the game and obtains $\widehat{\mathsf{coin}}$, it will ignore this and output a random $\mathsf{coin}'$ with some probability. The term *artificial* abort stems from the fact that the reduction is ignoring $\mathsf{A}$'s output even if it might be the case $\mathsf{coin} = \widehat{\mathsf{coin}}$. While counter intuitive, The artificial abort paradigm states that the reduction's advantage can degrade by at most a factor $\gamma_{\mathsf{min}}/3$. In other words, the quality of the reduction is dictated by how large $\gamma_{\mathsf{min}}$ can be; the larger the $\gamma_{\mathsf{min}}$, the better the reduction is.

*Remark 1 (Comparison with Prior Work).* As briefly mentioned in the introduction, the proof of Bellare and Ristenpart [5] can be seen as a special case of our Theorem 1. Their proof fixes the approximation function $\widetilde{\gamma}(\mathsf{T}) := \gamma_{\mathsf{min}}$ for all $\mathsf{T} \in \mathcal{T}$. Effectively, this is a special class of reduction *without* performing artificial aborts. As we see in the later sections, a tighter security proof is achieved by fine-tuning $\widetilde{\gamma}(\mathsf{T})$ and tactically performing artificial aborts. While we did not chose to do so, we can generalize our Theorem 1 to capture the proof of Waters [30] as well. Recall that in his proof, $\widetilde{\gamma}(\mathsf{T})$ is not a fixed value but rather a probabilistic value defined through the Monte Carlo method. Accordingly, $|\gamma(\mathsf{T}) - \widetilde{\gamma}(\mathsf{T})| < \frac{\gamma_{\mathsf{min}}}{3} \cdot \epsilon$ will only be satisfied with some probability. As we did not obtain new results with this generalization, we intentionally kept our definition simple to only capture [5].

## 4 Partitioning Function with Approximation

In this section, we introduce a tool called *partitioning function with approximation* allowing us to naturally use the finer grained artificial abort paradigm in Theorem 1 to prove tighter security of a wide class of cryptographic primitives.

### 4.1 Overview

A partitioning function *without* approximation was first introduced by Yamada [33]. Let us use IBE schemes as a representative example to get a flavor of this tool. A partitioning function allows the reduction to secretly partition the identity space into two sets of exponential size: the reduction can answer key-extraction queries on one set and embed a hard problem into the challenge ciphertext on the other set. The partition is made in a meticulous manner so that there is a noticeable probability that the adversary's key-extraction queries and the challenge identity fall in the correct sets. Looking at Theorem 1, the probability that the partitioning fails (e.g., the reduction cannot answer the key-extraction query) is denoted as $\mathsf{Bad}$, occurring with probability $1 - \gamma(\mathbf{ID})$, where $\mathbf{ID}$ is the sequence of identities queried by the adversary. Most prior works using (explicitly or implicitly) partitioning functions [6,21,25,32,33] rely on the analysis of Bellare and Ristenpart [5]. They approximate $\gamma(\mathbf{ID})$ by the trivial lower bound $\widetilde{\gamma}(\mathbf{ID}) = \gamma_{\mathsf{min}}$, in which case the probability of an artificial abort $\mathsf{AAbort}$ occurring becomes $1 - \gamma_{\mathsf{min}}/\widetilde{\gamma}(\mathbf{ID}) = 0$. Consequently, as explained in the technical overview, the reduction has to rely on a small $\gamma_{\mathsf{min}}$. As it is clear from Theorem 1, a smaller $\gamma_{\mathsf{min}}$ results in a worser reduction.

It is worth recalling that we cannot choose an arbitrary approximation $\widetilde{\gamma}(\mathbf{ID})$, say $\widetilde{\gamma}(\mathbf{ID}) = \gamma(\mathbf{ID})$, as $\widetilde{\gamma}(\mathbf{ID})$ needs to be *efficiently* computable. This is because the reduction must compute $1 - \gamma_{\mathsf{min}}/\widetilde{\gamma}(\mathbf{ID})$ to perform the artificial abort.

We propose four partitioning functions allowing to *efficiently* approximate $\gamma(\mathbf{ID})$ better than $\gamma_{\mathsf{min}}$. Each partitioning function has different characteristics and can be embedded into a wide class of cryptographic primitives with different algebraic properties. An overview of the partitioning functions with approximation can be found in the following Table 1. One of the four partitioning functions $\mathsf{F}_{\mathsf{ParWat}}$ is new to this work. $\mathsf{F}_{\mathsf{SSM}}$, $\mathsf{F}_{\mathsf{Wat}}$, and $\mathsf{F}_{\mathsf{Boy}}$ appear in [27], [30], and [3], respectively. The novelty of our work is proving that each of $\mathsf{F}_{\mathsf{SSM}}$, $\mathsf{F}_{\mathsf{Wat}}$, and $\mathsf{F}_{\mathsf{Boy}}$ has a corresponding efficiently computable approximation $\widetilde{\gamma}(\mathbf{ID})$ better than $\gamma_{\mathsf{min}}$, where in the case of $\mathsf{F}_{\mathsf{SSM}}$, we have to use specific error correcting codes in order for our analysis to work. Here, we present and analyze $\mathsf{F}_{\mathsf{ParWat}}$ and refer to the full version for details on the other functions. A concrete example of how to use our partitioning function with approximation along with Theorem 1 is given in Sect. 5 and 6.

### 4.2 Definition of Partitioning Function with Approximation

We define a partitioning function *with* approximation. The definition is based on [33], where we extend the original definition to capture a finer grained approximation of $\gamma$. We recover the original definition by setting $\widetilde{\gamma}(\mathbf{x}) = \gamma_{\mathsf{min}}$.

**Table 1.** Different Types of Partitioning Function and their Quality of $\gamma_{\mathsf{min}}$.

| Partitioning Function | $\gamma_{\mathsf{min}}$ with [5] Analysis | $\gamma_{\mathsf{min}}$ with Fine-tuned Analysis | Misc. |
|---|---|---|---|
| $\mathsf{F}_{\mathsf{Wat}}$ | $O(\epsilon/\ell Q)$ | $O(\sqrt{\epsilon}/\ell Q)$ | pairing: IBEs and VRFs<br>lattice: IBE with exp. modulus $q$ |
| $\mathsf{F}_{\mathsf{Boy}}$ | $O(\epsilon^2/Q^2)$ | $O(\epsilon/Q^2)$ | lattice IBEs |
| $\mathsf{F}_{\mathsf{ParWat}}$ (Sect. 4.3) | $O(\epsilon/qQ)$ | $O(\epsilon^{1/d}/qQ)^{\dagger}$ | lattice IBEs |
| $\mathsf{F}_{\mathsf{SSM}}$ | $O((\epsilon/Q)^{\mu})$ | $O((\sqrt{\epsilon}/Q)^{\mu})$ | pairing and lattice IBEs & VRFs |
| $\mathsf{F}_{\mathsf{SSM}}$ | $O((\epsilon/\ell Q)^{1+1/\nu})^{\ddagger}$ | $O(\sqrt{\epsilon}/\ell^{\nu} Q)$ | pairing and lattice IBEs & VRFs |

The table shows four different partitioning functions. A black (resp., gray) entry shows that the corresponding bound is proven in our work (resp., previous work). The column "$\gamma_{\mathsf{min}}$ with [5] Analysis" shows lower bounds on $\gamma_{\mathsf{min}}$ derived from Bellare-Ristenpart-style analysis, where $\widetilde{\gamma}(\mathbf{x})$ is a fixed value that does not depend on $\mathbf{x}$. The column "$\gamma_{\mathsf{min}}$ with Fine-tuned Analysis" shows lower bounds on $\gamma_{\mathsf{min}}$ derived from our fine-tuned analysis, where $\widetilde{\gamma}$ can be dependent on the input $\mathbf{x}$. For $\mathsf{F}_{\mathsf{SSM}}$, "Binary" (resp., "Poly") represents the case where the underlying error correcting code is instantiated over binary (resp., polynomial size) alphabet. In the table, $\ell$ is the length of the input, $q$ is the size of the modulus used in the lattice based constructions, and $d$ is an integer that can be set arbitrarily, which is determined by the underlying hash functions. The constants $\mu > 1$ and $1 \geq \nu > 0$ are determined by the underlying error correcting codes and can be set arbitrarily.
† By choosing $d = \omega(1)$, we can achieve $\gamma_{\mathsf{min}} = O(1/q\kappa Q)$, which removes the dependency on $\epsilon$ altogether.
‡ The bound here is due to Kohl [26]. We can improve the bound to $O(\epsilon/\ell^{\nu} Q)$ using our error correcting code. We refer to the full version for the details.

**Definition 1 (Partitioning Function with Approximation).** *Let* $\mathsf{F} = \left\{\mathsf{F}_{\kappa} : \mathcal{K}_{\kappa} \times \{0,1\}^{\ell(\kappa)} \to \{0,1\}\right\}_{\kappa \in \mathbb{N}}$ *be an ensemble of function families. We say that* $\mathsf{F}$ *is a* $(\gamma_{\mathsf{min}}, T_{\mathsf{F}}, T_{\mathsf{approx}})$-partitioning function, *if there exists an efficient algorithm* $\mathsf{PrtSmp}(1^{\kappa}, Q, \epsilon)$, *which takes as input a polynomially bounded* $Q = Q(\kappa) \in \mathbb{N}$ *and a noticeable* $\epsilon = \epsilon(\kappa) \in (0, 1/2]$ *and outputs a* partitioning key $K$ *such that:*

1. *There exists* $\kappa_0 \in \mathbb{N}$ *such that*
$$\Pr\left[K \in \mathcal{K}_{\kappa} \ : \ K \xleftarrow{\$} \mathsf{PrtSmp}\left(1^{\kappa}, Q(\kappa), \epsilon(\kappa)\right)\right] = 1$$
*for all* $\kappa > \kappa_0$. *Here,* $\kappa_0$ *may depend on functions* $Q(\kappa)$ *and* $\epsilon(\kappa)$.
2. *For a vector* $\mathbf{x} := (\mathsf{x}^*, \mathsf{x}^{(1)}, \ldots, \mathsf{x}^{(Q)}) \in (\{0,1\}^{\ell})^{Q+1}$, *let us define* $\gamma(\kappa, \mathbf{x})$ *as*
$$\gamma(\kappa, \mathbf{x}) := \Pr\left[\mathsf{F}(K, \mathsf{x}^{(1)}) = \cdots = \mathsf{F}(K, \mathsf{x}^{(Q)}) = 1 \right.$$
$$\left. \wedge\, \mathsf{F}(K, \mathsf{x}^*) = 0 : K \xleftarrow{\$} \mathsf{PrtSmp}\left(1^{\kappa}, Q(\kappa), \epsilon(\kappa)\right)\right].$$
*For* $\lambda > \lambda_0$, *there exist* $\gamma_{\mathsf{min}}(\kappa)$ *and* $\widetilde{\gamma}(\kappa, \mathbf{x})$ *that depend on* $Q(\kappa)$ *and* $\epsilon(\kappa)$ *such that for all distinct* $\mathsf{x}^{(1)}, \ldots, \mathsf{x}^{(Q)}, \mathsf{x}^* \in \{0,1\}^{\ell}$, *the following hold:*
$$\gamma(\kappa, \mathbf{x}) \geq \gamma_{\min}(\kappa), \ \ \widetilde{\gamma}(\kappa, \mathbf{x}) \geq \gamma_{\min}(\kappa), \ \ |\gamma(\kappa, \mathbf{x}) - \widetilde{\gamma}(\kappa, \mathbf{x})| < \frac{\gamma_{\min}(\kappa)}{3} \cdot \epsilon. \quad (10)$$

*The probability is taken over the choice of* $K \xleftarrow{\$} \mathsf{PrtSmp}(1^\lambda, Q(\lambda), \epsilon(\lambda))$.

3. *For* $\lambda > \lambda_0$*, there exists an algorithm that takes* $\kappa, Q$*,* $\epsilon$*, and* $\mathbf{x}$ *as input and computes* $\gamma_{\mathsf{min}}(\kappa)$ *and* $\widetilde{\gamma}(\kappa, \mathbf{x})$ *in time* $T_{\mathsf{approx}}(\kappa, Q, \epsilon)$*. Moreover, for all* $\kappa > \kappa_0$*,* $K \in \mathcal{K}$ *and* $\mathsf{x} \in \{0,1\}^\ell$*,* $\mathsf{F}(K, \mathsf{x})$ *can be computed in time* $T_{\mathsf{F}}(\kappa)$*.*

We may drop the subscript $\lambda$ and denote $\mathsf{F}$, $\mathcal{K}$, and $\mathcal{X}$ for the sake of simplicity.

## 4.3 A New Partitioning Function for Lattices

Here, we present a new partitioning function $\mathsf{F}_{\mathsf{ParWat}}$ that can be used in place of $\mathsf{F}_{\mathsf{Boy}}$ [3]. Compared to $\mathsf{F}_{\mathsf{Boy}}$, it achieves better parameter (i.e., larger $\gamma_{\mathsf{min}}$) and thus leads to better reduction costs in the corresponding applications. $\mathsf{F}_{\mathsf{ParWat}}$ can be viewed as performing parallel repetition of the Waters partitioning function $\mathsf{F}_{\mathsf{Wat}}$ with a twist, using a (perfect) *d-wise linearly independent hash function.*

Let $\mathsf{H}_n^{\mathsf{frd}} : \mathbb{Z}_q^n \to \mathbb{Z}_q^{n\times n}$ a full-rank difference encoding (see the full version for the formal definition). For any integers $d$ and $L_d = L(d)$, let $h_{d\text{-}\mathsf{wise}} : \{0,1\}^\ell \to \{0,1\}^{L_d}$ be a $d$-wise linearly independent hash function over $\mathbb{Z}_q$, that is, for any distinct $(\mathsf{x}_i)_{i\in[d]} \in (\{0,1\}^\ell)^d$, $(h_{d\text{-}\mathsf{wise}}(\mathsf{x}_i))_{i\in[d]}$ is linearly independent over $\mathbb{Z}_q$. For $d = 3$, we can define $h_{d\text{-}\mathsf{wise}}(\mathsf{x}) = (1, \mathsf{x})$, since as we show in the full version, the map $\mathsf{x} \mapsto (1, \mathsf{x})$ is 3-wise linearly independent over $\mathbb{Z}_p$ for any primer $p \geq 3$. We show how to construct such a $d$-wise linearly independent hash function for $d > 3$ in the full version. We then define our partitioning function $\mathsf{F}_{\mathsf{ParWat}}$ as follows:

$$\mathsf{F}_{\mathsf{ParWat}}(K, \mathsf{x}) = \begin{cases} 0 \ \sum_{i:h_{d\text{-}\mathsf{wise}}(\mathsf{x})_i=1} \mathsf{H}_n^{\mathsf{frd}}(K_i) = \mathbf{0}_{n\times n} \pmod q \\ 1 \text{ otherwise} \end{cases}$$

where $K := (K_1, \ldots, K_{L_d}) \in \mathcal{K} := (\mathbb{Z}_q^n)^{L_d}$, $\mathsf{x} \in \{0,1\}^\ell$, and $h_{d\text{-}\mathsf{wise}}(\mathsf{x})_i$ is the $i$-th bit of the hashed identity $h_{d\text{-}\mathsf{wise}}(\mathsf{x}) \in \{0,1\}^{L_d}$.

For this function, we have the following theorem.

**Theorem 2.** *Let* $n = n(\kappa), \ell = \ell(\kappa), q = q(\kappa), d = d(\kappa)$ *be integers such that* $q$ *is a prime and* $d \geq 3$ *is odd. Let* $h_{d\text{-}\mathsf{wise}} : \{0,1\}^\ell \to \{0,1\}^{L_d}$ *be a d-wise linearly independent hash function over* $\mathbb{Z}_q$*. Let* $\epsilon = \epsilon(\kappa)$ *be a noticeable function in* $(0, 1/2]$*,* $Q = Q(\kappa)$ *be a polynomially bounded positive integer, let* $k$ *be the smallest integer such that* $q^k \geq 2 \cdot Q \cdot \epsilon^{-\frac{1}{d-1}}$*. Then,* $\mathsf{F}_{\mathsf{ParWat}}$ *is a* $(\gamma_{\mathsf{min}}, T_{\mathsf{F}}, T_{\mathsf{approx}})$*-partitioning function such that*

$$\gamma_{\mathsf{min}} = \frac{1}{q^k} + \sum_{t\in[d-2]} (-1)^t \cdot \binom{Q}{t} \cdot \frac{1}{q^{(t+1)k}}, \ T_{\mathsf{F}} = L_d \cdot \mathsf{poly}(\kappa), \ \textit{and } T_{\mathsf{approx}} = \mathsf{poly}(\kappa),$$

*where* $\mathsf{poly}(\kappa)$ *is a fixed polynomial independent from* $Q$ *and* $\epsilon$*. In particular, this implies* $\gamma_{\mathsf{min}} \geq \frac{\epsilon^{\frac{1}{d-1}}}{4q\cdot Q}$ *and we have* $\gamma_{\mathsf{min}} \geq \frac{1}{4\kappa q\cdot Q}$ *if we set* $d = \omega(1)$*.*

*Proof.* We first define the algorithm $\mathsf{PrtSmp}(1^\kappa, Q, \epsilon)$.

$\mathsf{PrtSmp}(1^\kappa, Q, \epsilon) \to K$**:** It takes as input a security parameter $1^\kappa$, a polynomial bounded $Q = Q(\kappa)$, and a noticeable $\epsilon = \epsilon(\kappa) \in (0, 1/2]$. It computes the smallest integer such $q^k \geq 2 \cdot Q \cdot \epsilon^{-\frac{1}{d-1}}$ and samples $K \xleftarrow{\$} (\mathbb{Z}_q^k \times \{0\}^{n-k})^{L_d} \subseteq (\mathbb{Z}_q^n)^{L_d}$ and returns $K$.

It is clear that $\mathsf{PrtSmp}$ terminates in polynomial time. Below, we show that $\mathsf{PrtSmp}$ satisfies the three properties in Definition 1.

**First Property.** It is clear that $K \in \mathcal{K} := (\mathbb{Z}_q^n)^{L_d}$. Since the output $K$ of $\mathsf{PrtSmp}$ is always included in $\mathcal{K}$, $\mathsf{PrtSmp}$ satisfies the first property.

**Second Property.** For any $\mathsf{x}$, denote $\mathsf{E}(\mathsf{x})$ as the event $\mathsf{F}_{\mathsf{ParWat}}(K, \mathsf{x}) = 0$. Then, for $\mathbf{x} = (\mathsf{x}^*, \mathsf{x}^{(1)}, \ldots, \mathsf{x}^{(Q)})$, we define $\gamma(\kappa, \mathbf{x})$ as

$$\gamma(\kappa, \mathbf{x}) := \Pr\left[\neg\mathsf{E}(\mathsf{x}^{(1)}) \wedge \cdots \wedge \neg\mathsf{E}(\mathsf{x}^{(Q)}) \wedge \mathsf{E}(\mathsf{x}^*)\right]$$

where the probability is taken over the choice of $K \xleftarrow{\$} \mathsf{PrtSmp}\,(1^\kappa, Q, \epsilon)$.

Further define $\gamma_{\mathsf{min}}$ and $\widetilde{\gamma}(\mathbf{x})$ as

$$\gamma_{\mathsf{min}} := \frac{1}{q^k} + \sum_{t \in [d-2]} (-1)^t \cdot \binom{Q}{t} \cdot \frac{1}{q^{(t+1)k}}$$

$$\widetilde{\gamma}(\mathbf{x}) := \Pr[\mathsf{E}(\mathsf{x}^*)] + \sum_{t \in [d-2]} (-1)^t \cdot \left( \sum_{1 \leq j_1 < \cdots < j_t \leq [Q]} \Pr\left[\mathsf{E}(\mathsf{x}^*) \wedge \bigwedge_{k \in [t]} \mathsf{E}(\mathsf{x}^{(j_k)})\right]\right). \tag{11}$$

Below, we show that $\gamma(\mathbf{x})$, $\gamma_{\mathsf{min}}$, and $\widetilde{\gamma}(\mathbf{x})$ satisfy the three inequalities in Definition 1, Item 2. We first make a simplifying observation: notice that for any $\mathbf{x} \in \{0,1\}^\ell$, $\mathsf{F}_{\mathsf{ParWat}}(K, \mathsf{x}) = 0$ implies $\sum_{i : h_{d\text{-wise}}(\mathsf{x})_i = 1} K_i = \mathbf{0}_n \in \mathbb{Z}_q^n$ since $\mathsf{H}_n^{\mathsf{frd}}$ is linearly homomorphic and $\mathbf{0}_n$ is the only vector that gets mapped to $\mathbf{0}_{n \times n}$ by $\mathsf{H}_n^{\mathsf{frd}}$. Moreover, since each entry of $K_1, \cdots, K_n$ is distributed independently of each other, we can analyze the probability that $\sum_{i : h_{d\text{-wise}}(\mathsf{x})_i = 1} K_i = \mathbf{0}_n$ entry-wise. That is, for any $\mathsf{x} \in \{0,1\}^\ell$, we have $\Pr[\sum_{i : h_{d\text{-wise}}(\mathsf{x})_i = 1} K_i = \mathbf{0}_n] = \prod_{\nu \in [n]} \Pr\left[\sum_{i : h_{d\text{-wise}}(\mathsf{x})_i = 1} K_i[\nu] = 0\right] = \prod_{\nu \in [k]} \Pr\left[\sum_{i : h_{d\text{-wise}}(\mathsf{x})_i = 1} K_i[\nu] = 0\right]$, where $K_i[\nu]$ denotes the $\nu$-th entry of $K_i$ and the last equality follows from $K_i \in \mathbb{Z}_q^k \times \{0\}^{n-k}$ for all $i \in [L_d]$. For any $\mathsf{x}$ and $\nu \in [k]$, let us denote $\mathsf{E}_\nu(\mathsf{x})$ to be the event $\sum_{i : h_{d\text{-wise}}(\mathsf{x})_i = 1} K_i[\nu] = 0$ for $K \xleftarrow{\$} \mathsf{PrtSmp}(1^\lambda, Q, \epsilon)$. Then, from the above argument, $\mathsf{E}(\mathsf{x}) = \wedge_{\nu \in [k]} \mathsf{E}_\nu(\mathsf{x})$ defines the event $\mathsf{F}_{\mathsf{ParWat}}(K, \mathsf{x}) = 0$.

Now, let us first focus on the first inequality: $\gamma(\mathbf{x}) \geq \gamma_{\mathsf{min}}$. Notice that if $\gamma(\mathbf{x}) \geq \widetilde{\gamma}(\mathbf{x})$, then the second inequality in Definition 1, Item 2 implies the first inequality. Since we will show the second inequality later, we only need to show

$\gamma(\mathbf{x}) \geq \widetilde{\gamma}(\mathbf{x})$.

$$\begin{aligned}\gamma(\mathbf{x}) &= \Pr[\mathsf{E}(\mathsf{x}^*) \wedge \neg\mathsf{E}(\mathsf{x}^{(1)}) \wedge \cdots \wedge \neg\mathsf{E}(\mathsf{x}^{(Q)})] \\ &= \Pr[\mathsf{E}(\mathsf{x}^*)] - \Pr[\mathsf{E}(\mathsf{x}^*) \wedge \neg(\neg\mathsf{E}(\mathsf{x}^{(1)}) \wedge \cdots \wedge \neg\mathsf{E}(\mathsf{x}^{(Q)}))] \\ &= \Pr[\mathsf{E}(\mathsf{x}^*)] - \Pr[\mathsf{E}(\mathsf{x}^*) \wedge (\mathsf{E}(\mathsf{x}^{(1)}) \vee \cdots \vee \mathsf{E}(\mathsf{x}^{(Q)}))] \\ &= \Pr[\mathsf{E}(\mathsf{x}^*)] - \Pr[(\mathsf{E}(\mathsf{x}^*) \wedge \mathsf{E}(\mathsf{x}^{(1)})) \vee \cdots \vee (\mathsf{E}(\mathsf{x}^*) \wedge \mathsf{E}(\mathsf{x}^{(Q)}))] \\ &\geq \Pr[\mathsf{E}(\mathsf{x}^*)] + \sum_{t\in[d-2]} (-1)^t \cdot \left( \sum_{1\leq j_1<\cdots<j_t\leq[Q]} \Pr\left[ \bigwedge_{k\in[t]} \left( \mathsf{E}(\mathsf{x}^*) \wedge \mathsf{E}(\mathsf{x}^{(j_k)}) \right] \right) \right) \\ &= \Pr[\mathsf{E}(\mathsf{x}^*)] + \sum_{t\in[d-2]} (-1)^t \cdot \left( \sum_{1\leq j_1<\cdots<j_t\leq[Q]} \Pr\left[ \mathsf{E}(\mathsf{x}^*) \wedge \bigwedge_{k\in[t]} \mathsf{E}(\mathsf{x}^{(j_k)}) \right] \right) = \widetilde{\gamma}(\mathbf{x}),\end{aligned} \tag{12}$$

where the third equation follows from the De Morgan's laws and the inequality follows from the Bonferroni inequality and the fact that $d$ is odd. In the above, we also assume implicitly that $d \leq Q$; if $d = Q$, then the above will be an equality rather than an inequality. Thus, $\gamma(\mathbf{x}) \geq \widetilde{\gamma}(\mathbf{x})$ as desired.

We next show the second inequality: $\widetilde{\gamma}(\mathbf{x}) \geq \gamma_{\mathsf{min}}$. Using the fact that $h_{d\text{-}\mathsf{wise}}$ is a $d$-wise linearly independent hash over $\mathbb{Z}_q$, for any distinct $(\mathsf{x}_i)_{i\in[d]}$, $(h_{d\text{-}\mathsf{wise}}(\mathsf{x}_i))_{i\in[d]}$ is linearly independent over $\mathbb{Z}_q$. As we show in the full version, we have the following for every $t \in [d]$ and $\nu \in [k]$:

$$\Pr\left[\bigwedge_{i\in[t]} \mathsf{E}_\nu(\mathsf{x}_i)\right] = \frac{1}{q^t} \quad \Rightarrow \quad \Pr\left[\bigwedge_{i\in[t]} \mathsf{E}(\mathsf{x}_i)\right] = \frac{1}{q^{tk}}, \tag{13}$$

where the implication holds from $\mathsf{E}(\mathsf{x}) = \wedge_{\nu\in[k]}\mathsf{E}_\nu(\mathsf{x})$ and the independence of $\mathsf{E}_\nu(\mathsf{x})$ for distinct $\nu$'s. Plugging this into Eq. (12), we have

$$\widetilde{\gamma}(\mathbf{x}) = \frac{1}{q^k} + \sum_{t\in[d-2]} (-1)^t \cdot \binom{Q}{t} \cdot \frac{1}{q^{(t+1)k}} = \gamma_{\mathsf{min}}.$$

This establishes the second inequality.

Finally, we show the third inequality: $|\gamma(\kappa, \mathbf{x}) - \widetilde{\gamma}(\kappa, \mathbf{x})| < \frac{\gamma_{\min}(\kappa)}{3} \cdot \epsilon$. Following a similar argument made to derive Eq. (12), we can establish

$$\gamma(\mathbf{x}) \leq \Pr[\mathsf{E}(\mathsf{x}^*)] + \sum_{t\in[d-1]} (-1)^t \cdot \left( \sum_{1\leq j_1<\cdots<j_t\leq[Q]} \Pr\left[ \mathsf{E}(\mathsf{x}^*) \wedge \bigwedge_{k\in[t]} \mathsf{E}(\mathsf{x}^{(j_k)}) \right] \right),$$

where the only difference is that we use the Bonferroni inequality to upper bound, rather than lower bound, $\gamma(\mathbf{x})$. This implies

$$|\gamma(\mathbf{x}) - \tilde{\gamma}(\mathbf{x})| \leq \sum_{1\leq j_1<\cdots<j_{d-1}\leq[Q]} \Pr\left[ \mathsf{E}(\mathsf{x}^*) \wedge \bigwedge_{k\in[d-1]} \mathsf{E}(\mathsf{x}^{(j_k)}) \right] = \binom{Q}{d-1} \cdot \frac{1}{q^{dk}},$$

where the right equality holds from Eq. (13).

It remains to show the following inequality for the third inequality.

$$\binom{Q}{d-1} \cdot \frac{1}{q^{dk}} < \frac{\gamma_{\mathsf{min}}}{3} \cdot \epsilon = \frac{\epsilon}{3q^k} \left( 1 + \sum_{t \in [d-2]} (-1)^t \cdot \binom{Q}{t} \cdot \frac{1}{q^{tk}} \right). \tag{14}$$

From assumption, we have $Q \leq c \cdot q^k \cdot \epsilon^{1/(d-1)}$ for $c = 1/2$. Plugging this into the left hand side of Eq. (14), we have

$$\text{(l.h.s)} \leq \frac{Q^{d-1}}{2^{d-2} \cdot q^{dk}} \leq \frac{c^{d-1} \cdot \epsilon}{2^{d-2} \cdot q^k} = \frac{\epsilon}{2^{2d-3} \cdot q^k},$$

where the first inequality follows from the fact $(d-1)! \geq 2^{d-2}$ for $d \geq 3$. On the other hand, we have

$$\frac{\epsilon}{6q^k} \leq \frac{\epsilon}{3q^k} \cdot \left( 1 - c \cdot \epsilon^{\frac{1}{d-1}} \right) \leq \text{(r.h.s)},$$

where the first inequality follows from $c = 1/2$, $\epsilon \in (0, 1/2]$ and $\epsilon^{1/(d-1)} < 1$ for any $d \geq 3$, and the second inequality follows implicitly from the Bonferroni inequality. Thus, for any $d \geq 3$, we have Eq. (14) as desired. This establishes the third inequality.

Combining everything, $\mathsf{F}_{\mathsf{ParWat}}$ indeed satisfies the second property of Definition 1. As a concrete example, $k$ is the smallest integer such that $q^k \geq 2 \cdot Q \cdot \epsilon^{-\frac{1}{d-1}}$. Thus, we have $2 \cdot Q \cdot \epsilon^{-\frac{1}{d-1}} \geq q^{k-1}$, implying $2 \cdot q \cdot Q \cdot \epsilon^{-\frac{1}{d-1}} \geq q^k$. Combined with the lower bound $\gamma_{\mathsf{min}} \geq \frac{1}{2q^k}$ (implicitly) established above, we have $\gamma_{\mathsf{min}} > \frac{\epsilon^{\frac{1}{d-1}}}{4q \cdot Q}$ as in the theorem statement. The statement on the case of $d = \omega(1)$ is obtained by observing $\epsilon > \kappa^{-d}$ for sufficiently large $\lambda$.

**Third Property.** Finally, we show the third property of Definition 1. Notice that for any $\mathsf{x}$, we established $\widetilde{\gamma}(\mathsf{x}) = \gamma_{\mathsf{min}}$. Since $\gamma_{\mathsf{min}}$ can be computed in time $\mathsf{poly}(d, \log Q, \log(1/\epsilon))$ so can $\widetilde{\gamma}(\mathsf{x})$. Note we can upper bound $\mathsf{poly}(d, \log Q, \log(1/\epsilon)) = \mathsf{poly}(d, \kappa)$ by a fixed polynomial since $Q$ is a polynomial and $\epsilon$ is noticeable. Moreover, $\mathsf{F}_{\mathsf{Boy}}$ can be computed with $k \times L_d$ additions so we have $T_{\mathsf{F}} = L_d \cdot \mathsf{poly}(\log Q, \log(1/\epsilon))$. Similarly this is upper bounded by $L_d \cdot \mathsf{poly}(\kappa)$ for some fixed polynomial as desired.

## 5 Application to IBEs

Recall that the notion of the partitioning function [33] abstracts out the core statistical properties useful for proving security of various cryptographic primitives. Section 3, we essentially showed that if the underlying partitioning function admits good enough approximation for the quantity $\gamma$, then we can achieve better reduction costs in various security proofs than those obtained by existing techniques [5,30]. Then, in Sect. 4, we showed that new and existing partitioning functions indeed admit good enough approximations. These arguments are divorced from the underlying cryptographic primitives and algebraic structures. In this section, we apply the tools we developed in Sects. 3 and 4 to the specific context of IBE. This allows us to prove improved reduction costs for Waters IBE [30] and Agrawal-Boneh-Boyen IBE [3] and also yields a new scheme with good reduction costs. To formally prove these results in a unified manner, we show a template of the security proof for IBE that uses partitioning functions. We then prove the security of the respective IBE schemes using the template.

### 5.1 Application to Waters IBE

Here, we apply our framework to Waters IBE [30]. His IBE achieves the unique property of having short ciphertext consisting only of 2 group elements and security under the standard DBDH assumption or even under the CBDH assumption if we slightly modify it using Goldreich-Levin's hardcore bit function [14] (See the full version and [25]). For Waters IBE, we improve the reduction cost from $O(\epsilon^2/Q\ell)$ to $O(\epsilon^{1.5}/Q\ell)$, where by reduction cost we mean the advantage of the DBDH solving algorithm obtained by a $(t, Q, \epsilon)$-adversary against the IBE. Here, we ignore the difference between the running time of the DBDH solving algorithms, since they are $t + Q \cdot \mathsf{poly}(\kappa)$ in both cases and their difference can be ignored in most of the interesting parameters settings. More formally, we obtain the following theorem:

**Theorem 3.** *If there is an $(t_{\mathsf{A}}, Q, \epsilon_{\mathsf{A}})$-adversary $\mathsf{A}$ against the $\mathsf{IND\text{-}CPA}$ security of the Waters IBE scheme, there is an adversary $\mathsf{B}$ that breaks the DBDH problem with advantage $\epsilon_{\mathsf{B}}$ and $t_{\mathsf{B}}$ such that*

$$\epsilon_{\mathsf{B}} > \frac{\epsilon_{\mathsf{A}}^{1.5}}{21Q\ell}, \quad t_{\mathsf{B}} = t_{\mathsf{A}} + O(Q \cdot \ell^2) \cdot \mathsf{poly}(\kappa) \tag{15}$$

*where $Q \le p\sqrt{\epsilon_{\mathsf{A}}}/\ell\sqrt{3}$ and $\mathsf{poly}(\kappa)$ is roughly the overhead incurred by the running the simulated algorithms compared to the real $(\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Encrypt})$ algorithms.*

The proof of the theorem can be obtained by observing that the original proof of Waters IBE follows the template of partitioning-based reduction for IBE and plugging in our analysis on $\mathsf{F}_{\mathsf{Wat}}$ into our template. We provide the proof of the theorem and necessary background, including the description of the Waters IBE scheme and partitioning-based reduction for the scheme, in the full version.

### 5.2 Applications to ABB IBE and Its Variant

Here, we apply our framework to ABB IBE [3], which is one of the most important lattice IBE schemes, since it achieves the shortest ciphertext size and computational efficiency among the existing schemes. Conventionally, the reduction cost for ABB IBE was considered to be $O(\epsilon^2/qQ)$, employing the partitioning strategy based on $\mathsf{F}_{\mathsf{Boy}}$. However, as we note in the full version, our formal analysis reveals that they are only lower bounded by $O(\epsilon^3/Q^2)$, which is much worse.

Using our new analysis on $\mathsf{F}_{\mathsf{Boy}}$, we can improve it to be $O(\epsilon^2/Q^2)$. Furthermore, by using our analysis on new partitioning function $\mathsf{F}_{\mathsf{ParWat}}$ with $d = 3$, this can be further improved to be $O(\epsilon^{1.5}/qQ)$. More formally, we obtain the following theorem:

**Theorem 4.** *If there is an $(t_{\mathsf{A}}, Q, \epsilon_{\mathsf{A}})$-adversary $\mathsf{A}$ against the $\mathsf{IND\text{-}CPA}$ security of the ABB IBE scheme, there is an adversary $\mathsf{B}$ that breaks the LWE problem with advantage $\epsilon_{\mathsf{B}}$ and $t_{\mathsf{B}}$ such that*

$$\epsilon_{\mathsf{B}} > \frac{\epsilon_{\mathsf{A}}^{1.5}}{12qQ} - \mathsf{negl}(\kappa), \quad t_{\mathsf{B}} = t_{\mathsf{A}} + Q \cdot \mathsf{poly}(\kappa) \tag{16}$$

*where $q^n \geq 2 \cdot Q/\sqrt{\epsilon_{\mathsf{A}}}$ holds for dimension $n$ of the scheme and $\mathsf{poly}(\kappa)$ is roughly the overhead incurred by the running the simulated algorithms compared to the real $(\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Encrypt})$ algorithms.*

We also consider a variant of ABB IBE, where we hash an identity using $d$-wise linearly independent hash function and then use it as a new identity in ABB IBE scheme. Roughly speaking, $d$-extended ABB IBE has a master public key size that is $d$-times longer than the original ABB IBE and has almost the same ciphertext size. We call it $d$-extended ABB IBE scheme. For $d$-extended ABB IBE, we can achieve better reduction cost of $O(\epsilon^{1+\frac{1}{d-1}}/qQ)$ using the power of $\mathsf{F}_{\mathsf{ParWat}}$ for arbitrarily chosen odd $d$.

**Theorem 5.** *If there is an $(t_{\mathsf{A}}, Q, \epsilon_{\mathsf{A}})$-adversary $\mathsf{A}$ against the $\mathsf{IND\text{-}CPA}$ security of the $d$-extended ABB IBE scheme for odd integer $d \geq 3$, there is an adversary $\mathsf{B}$ that breaks the LWE problem with advantage $\epsilon_{\mathsf{B}}$ and $t_{\mathsf{B}}$ such that*

$$\epsilon_{\mathsf{B}} > \frac{\epsilon_{\mathsf{A}}^{1+\frac{1}{d-1}}}{12qQ} - \mathsf{negl}(\kappa), \quad t_{\mathsf{B}} = t_{\mathsf{A}} + Q \cdot \mathsf{poly}(\kappa).$$

*In particular, if we have $d \geq \omega(1)$, we have*

$$\epsilon_{\mathsf{B}} > \frac{\epsilon_{\mathsf{A}}}{12q\kappa Q} - \mathsf{negl}(\kappa), \quad t_{\mathsf{B}} = t_{\mathsf{A}} + Q \cdot \mathsf{poly}(\kappa)$$

*where $q^n \geq 2 \cdot Q \cdot \epsilon^{-\frac{1}{d-1}}$ holds for dimension $n$ of the scheme and $\mathsf{poly}(\kappa)$ is roughly the overhead incurred by the running the simulated algorithms compared to the real $(\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Encrypt})$ algorithms.*

Note that Theorem 4 is a special case of Theorem 5, since $d$-extended ABB scheme with $d = 3$ equals to the ABB scheme. The proof of Theorem 5 can be obtained by showing that $d$-extended ABB IBE admits partitioning-based reduction and plugging in our analysis on $\mathsf{F}_{\mathsf{ParWat}}$ in Sect. 4.3 into our template. We provide the formal proof of the theorems and necessary background, including the description of ABB and $d$-extended ABB IBE schemes, in the full version.

## 6 Application to VRFs

In this section, we apply the tools we developed in Sects. 3 and 4 to VRF. Similarly to the case of IBE (Sect. 5), we prepare a security proof template that allows us to prove the security of VRF using partitioning function with approximation in a modular manner. The template is provided in the full version. However, unlike Sect. 5, we do not focus on applying our framework to existing schemes. Rather, we construct a new VRF scheme and then apply our framework to the scheme. The new VRF scheme subsumes the previous schemes in terms of asymptotic space efficiency and security at the same time, in the sense that it is proven secure under the standard $d$-LIN assumption with tighter reductions. We refer to Table 2 for the overview.

### 6.1 Our New Short VRF

Here, we propose new construction of VRF with short parameters. Our scheme achieves the best space efficiency among the existing schemes and enjoys the security proof under a static assumption at the same time. Our construction is based on the construction proposed by Kohl [26], but we substantially improve the space efficiency by adding a new twist to the scheme. We then proceed to prove the security of the scheme based on our framework. Our framework yields tighter reduction cost compared to the conventional analyses.

In Fig. 1, we give the description of our new VRF scheme. We refer to the full version for details on the notations. For the construction, we need an error correcting code $\mathsf{Encode} : \{0,1\}^\ell \to \Sigma^n$ for an alphabet $\Sigma$ and an injective map $\mathsf{Inj} : [n] \times \Sigma \to [n_1] \times [n_2]$. In order to be able to define such an injective map, we need to have $n|\Sigma| \le n_1 n_2$, where $|\Sigma|$ is the size of the alphabet. We will typically set $n_1 = n_2 = \lceil \sqrt{n|\Sigma|} \rceil$ to achieve the smallest verification key size. For the construction, we use the map $\mathsf{S} : \{0,1\}^\ell \to 2^{[n_1]\times[n_2]}$ defined as

$$\mathsf{S}(\mathsf{x}) := \{\ \mathsf{Inj}(i, \mathsf{Encode}(\mathsf{x})_i) : i \in [n]\ \},$$

where $\mathsf{Encode}(\mathsf{x})_i \in \Sigma$ denotes the $i$-th symbol of $\mathsf{Encode}(\mathsf{x}) \in \Sigma^n$. We can instantiate $\mathsf{Encode}$ by the binary or non-binary error correcting codes (see the full version). As we will discuss in Sect. 6.2, different choice of error correcting codes leads to trade-offs between the efficiency and the reduction loss. The construction is parameterized by $d$ and is secure under the $d$-LIN assumption similarly to [26]. We typically choose $d$ to be small constant like $d = 2$ or $d = 3$.

We prove correctness, unique provability, and pseudorandomness of our new VRF scheme in the full version. The pseudorandomness of our VRF scheme is proven by using our security proof template. Specifically, we show that our scheme has the partitioning-based reduction associated with the partitioning function with approximation $\mathsf{F}_{\mathsf{SSM}}$ and plugging in our analysis on $\mathsf{F}_{\mathsf{SSM}}$ into the template.

```
Gen(1^λ)
1 : $\mathcal{G} \xleftarrow{\$} \mathsf{GrpGen}(1^\lambda)$
2 : $\mathbf{M}_{i,j} \xleftarrow{\$} \mathbb{Z}_p^{d \times d}$ for $i \in [\eta]$ and $j \in [n_1]$
3 : $\mathbf{N}_{i,k} \xleftarrow{\$} \mathbb{Z}_p^{d \times d}$ for $i \in [\eta]$ and $k \in [n_2]$
4 : $\mathbf{u} \xleftarrow{\$} \mathbb{Z}_p^d \backslash \{\mathbf{0}_d\}, \mathbf{w} \xleftarrow{\$} (\mathbb{Z}_p^*)^d$
5 : $\mathsf{vk} := (\mathcal{G}, [\mathbf{u}], [\mathbf{w}], \{[\mathbf{M}_{i,j}]\}_{i \in [\eta], j \in [n_1]}, \{[\mathbf{N}_{i,k}]\}_{i \in [\eta], k \in [n_2]})$
6 : $\mathsf{sk} := (\mathcal{G}, \mathbf{u}, \mathbf{w}, \{\mathbf{M}_{i,j}\}_{i \in [\eta], j \in [n_1]}, \{\mathbf{N}_{i,k}\}_{i \in [\eta], k \in [n_2]})$
7 : return $(\mathsf{vk}, \mathsf{sk})$
```

```
Eval(sk, x)
1 : parse $\mathsf{sk} \leftarrow (\mathcal{G}, \mathbf{u}, \mathbf{w}, \{\mathbf{M}_{i,j}\}_{i,j}, \{\mathbf{N}_{i,k}\}_{i,k})$
2 : Compute $\mathsf{S}(\mathsf{x}) \subseteq [n_1] \times [n_2]$
3 : Compute $\mathbf{P}_i := \sum_{(j,k) \in \mathsf{S}(\mathsf{x})} \mathbf{M}_{i,j} \mathbf{N}_{i,k}$ for $i \in [\eta]$
4 : Compute $\mathbf{v}_i := \left( \prod_{\iota=1}^{i} \mathbf{P}_\iota \right)^\top \mathbf{u}$ for $i \in [\eta]$
5 : $\mathbf{z} := \mathbf{v}_\eta \oslash \mathbf{w}$
6 : $\mathsf{y} := [\langle \mathbf{z}, \mathbf{1}_d \rangle], \ \pi := (\{[\mathbf{v}_i], [\mathbf{P}_i]\}_{i \in [\eta]}, [\mathbf{z}])$
7 : return $(\mathsf{y}, \pi)$
```

```
Verify(vk, x, y, π)
1 : Check that vk is in the following form and output 0 otherwise:
    $\mathsf{vk} = (\mathcal{G}, [\mathbf{u}] \in \mathbb{G}^d, [\mathbf{w}] \in \mathbb{G}^d, \{[\mathbf{M}_{i,j}] \in \mathbb{G}^{d \times d}\}_{i \in [\eta], j \in [n_1]}, \{[\mathbf{N}_{i,k}] \in \mathbb{G}^{d \times d}\}_{i \in [\eta], k \in [n_2]})$
    such that $\mathsf{GrpVfy}(\mathcal{G}) = 1$
2 : Check that y and $\pi$ are in the following form and output 0 otherwise:
    $\mathsf{y} \in \mathbb{G}, \quad \pi = (\left\{ [\mathbf{v}_i] \in \mathbb{G}^d, [\mathbf{P}_i] \in \mathbb{G}^{d \times d} \right\}_{i \in [\eta]}, [\mathbf{z}] \in \mathbb{G}^d)$
3 : Check whether the following equations hold for all $i \in [\eta]$ and output 0 otherwise:
    $e([\mathbf{I}_d], [\mathbf{P}_i]) \stackrel{?}{=} \prod_{(j,k) \in \mathsf{S}(\mathsf{x})} e([\mathbf{M}_{i,j}], [\mathbf{N}_{i,k}]), \quad e([\mathbf{I}_d], [\mathbf{v}_i]) \stackrel{?}{=} e(\left[\mathbf{P}_i^\top\right], [\mathbf{v}_{i-1}]), \quad$ where $\mathbf{v}_0 := \mathbf{u}$
4 : Check whether the following equations hold and output 0 otherwise:
    $[\mathbf{z}] \odot [\mathbf{w}] \stackrel{?}{=} [\mathbf{1}_d] \odot [\mathbf{v}_\eta], \quad \mathsf{y} \stackrel{?}{=} [\langle \mathbf{z}, \mathbf{1}_d \rangle]$
5 : return 1
```

**Fig. 1.** Our VRF Scheme.

## 6.2 Comparison

Here, we discuss our new VRF scheme constructed in Sect. 6.1 and compare it with previous schemes. We refer to Table 2 for the overview. For comparison, we focus on the schemes that achieve "all the desired properties" [16]. Namely, we require the construction to have exponential-sized input space, adaptive security (i.e., both evaluation queries and the challenge query can be made adaptively), and security under non-interactive assumption. Here, we narrow the focus further and discuss constructions that are proven secure under a static assumption. We therefore do not include constructions that are proven secure under $q$-type assumptions [9,20,21,24,28,33] or even stronger assumptions [22,23] in the table. However, we mention that our construction achieves asymptotic efficiency that matches that of [24], which is based on $q$-type assumptions. We also do not include the construction of VRFs from general assumptions that are quite inefficient [6,15]. As we can see from the table, we achieve the best parameter size and reduction costs at the same time. In particular, compared to [26], our scheme substantially reduces the verification key size and improves the reduction cost at the same time, while maintaining the compact proof size. This improvement is obtained by the combination of our usage of error correcting codes and the

**Table 2.** Comparison of VRF Schemes with All The Desired Properties Based on Standard Assumptions.

| Schemes | $\lvert\mathsf{vk}\rvert$ | $\lvert\pi\rvert$ | Reduction Cost |
|---|---|---|---|
| | (# of $\mathbb{G}$) | (# of $\mathbb{G}$) | |
| [16] | $O(\kappa)$ | $O(\kappa)$ | $\epsilon^{1+\mu}/\kappa Q^{\mu}$ |
| [29] | $O(\kappa)$ | $O(\kappa)$ | $\epsilon^{1+\mu}/\kappa Q^{\mu}$ |
| [26] (binary) | $\omega(\kappa \log \kappa)$ | $\omega(\log \kappa)$ | $\epsilon^{1+\mu}/\omega(\log \kappa) Q^{\mu}$ |
| [26] (polynomial) | $\omega(\lambda^{2+2\nu})$ | $\omega(1)$ | $\epsilon^{2+1/\nu}/\omega(1)\kappa^{1+\nu}Q^{1+1/\nu}$ |
| Section 6.1 (binary) | $\omega(\sqrt{\kappa \log \kappa})$ | $\omega(\log \kappa)$ | $\epsilon^{1/2+\mu}/\omega(\log \kappa) Q^{\mu}$ |
| Section 6.1 (polynomial) | $\omega(\lambda^{1/2+5\nu/2})$ | $\omega(1)$ | $\epsilon^{3/2}/\omega(1)\kappa^{\nu} Q$ |

We compare VRF schemes with all the desired properties proven secure under a static assumption. The constructions in the table are all proven secure under the $d$-LIN assumption. $\lvert\mathsf{vk}\rvert$ and $\lvert\pi\rvert$ represent the size of the verification keys and the size of the proofs, respectively. To measure $\lvert\mathsf{vk}\rvert$ and $\lvert\pi\rvert$, we count the number of group elements. $Q$ and $\epsilon$ denote the number of evaluation queries and the advantage, respectively. $\mathsf{poly}(\lambda)$ represents fixed polynomial that does not depend on $Q$ and $\epsilon$. To measure the reduction cost, we show the advantage of the algorithm that solves the problem constructed from the adversary against the corresponding VRF scheme. We measure the reduction cost by employing the technique of Bellare and Ristenpart [5] for all the prior scheme and use our fine-tuned analysis for our schemes. In the table, $\mu$ and $\nu$ are arbitrary constants with $\mu > 1$ and $0 < \nu \leq 1$, respectively.

change in the underlying algebraic structure of the scheme. We refer to the full version for more discussion.

**Acknowledgement.** The first and the last author were partly supported by JST AIP Acceleration Research JPMJCR22U5. The third author is partially supported by JSPS KAKENHI Grant Number JP21K17700. The last author was supported by JST CREST Grant Number JPMJCR22M1.

## References

1. Abdalla, M., Catalano, D., Fiore, D.: Verifiable random functions: relations to identity-based key encapsulation and new constructions. J. Cryptol. **27**(3), 544–593 (2014). https://doi.org/10.1007/s00145-013-9153-x
2. Abla, P., Liu, F.-H., Wang, H., Wang, Z.: Ring-based identity based encryption – asymptotically shorter MPK and tighter security. In: Nissim, K., Waters, B. (eds.) TCC 2021, Part III. LNCS, vol. 13044, pp. 157–187. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-90456-2_6
3. Agrawal, S., Boneh, D., Boyen, X.: Efficient lattice (H)IBE in the standard model. In: Gilbert, H. (ed.) EUROCRYPT 2010. LNCS, vol. 6110, pp. 553–572. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-13190-5_28

4. Apon, D., Fan, X., Liu, F.H.: Vector encoding over lattices and its applications. Cryptology ePrint Archive, Report 2017/455 (2017). https://eprint.iacr.org/2017/455
5. Bellare, M., Ristenpart, T.: Simulation without the artificial abort: simplified proof and improved concrete security for Waters' IBE scheme. In: Joux, A. (ed.) EUROCRYPT 2009. LNCS, vol. 5479, pp. 407–424. Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-642-01001-9_24
6. Bitansky, N.: Verifiable random functions from non-interactive witness-indistinguishable proofs. In: Kalai, Y., Reyzin, L. (eds.) TCC 2017, Part II. LNCS, vol. 10678, pp. 567–594. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-70503-3_19
7. Boneh, D.: Simplified OAEP for the RSA and Rabin functions. In: Kilian, J. (ed.) CRYPTO 2001. LNCS, vol. 2139, pp. 275–291. Springer, Heidelberg (2001). https://doi.org/10.1007/3-540-44647-8_17
8. Boneh, D., Boyen, X.: Secure identity based encryption without random oracles. In: Franklin, M. (ed.) CRYPTO 2004. LNCS, vol. 3152, pp. 443–459. Springer, Heidelberg (2004). https://doi.org/10.1007/978-3-540-28628-8_27
9. Boneh, D., Montgomery, H.W., Raghunathan, A.: Algebraic pseudorandom functions with improved efficiency from the augmented cascade. In: Al-Shaer, E., Keromytis, A.D., Shmatikov, V. (eds.) ACM CCS 2010, pp. 131–140. ACM Press (Oct 2010). https://doi.org/10.1145/1866307.1866323
10. Bonferroni, C.: Teoria statistica delle classi e calcolo delle probabilita. Pubblicazioni del R Istituto Superiore di Scienze Economiche e Commericiali di Firenze **8**, 3–62 (1936)
11. Boyen, X.: Lattice mixing and vanishing trapdoors: a framework for fully secure short signatures and more. In: Nguyen, P.Q., Pointcheval, D. (eds.) PKC 2010. LNCS, vol. 6056, pp. 499–517. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-13013-7_29
12. Cash, D., Hofheinz, D., Kiltz, E., Peikert, C.: Bonsai trees, or how to delegate a lattice basis. In: Gilbert, H. (ed.) EUROCRYPT 2010. LNCS, vol. 6110, pp. 523–552. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-13190-5_27
13. Freire, E.S.V., Hofheinz, D., Paterson, K.G., Striecks, C.: Programmable hash functions in the multilinear setting. In: Canetti, R., Garay, J.A. (eds.) CRYPTO 2013, Part I. LNCS, vol. 8042, pp. 513–530. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-40041-4_28
14. Goldreich, O., Levin, L.A.: A hard-core predicate for all one-way functions. In: 21st ACM STOC, pp. 25–32. ACM Press (1989). https://doi.org/10.1145/73007.73010
15. Goyal, R., Hohenberger, S., Koppula, V., Waters, B.: A generic approach to constructing and proving verifiable random functions. In: Kalai, Y., Reyzin, L. (eds.) TCC 2017, Part II. LNCS, vol. 10678, pp. 537–566. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-70503-3_18
16. Hofheinz, D., Jager, T.: Verifiable random functions from standard assumptions. In: Kushilevitz, E., Malkin, T. (eds.) TCC 2016, Part I. LNCS, vol. 9562, pp. 336–362. Springer, Heidelberg (2016). https://doi.org/10.1007/978-3-662-49096-9_14
17. Hofheinz, D., Jager, T., Knapp, E.: Waters signatures with optimal security reduction. In: Fischlin, M., Buchmann, J., Manulis, M. (eds.) PKC 2012. LNCS, vol. 7293, pp. 66–83. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-30057-8_5

18. Hofheinz, D., Kiltz, E.: Programmable hash functions and their applications. In: Wagner, D. (ed.) CRYPTO 2008. LNCS, vol. 5157, pp. 21–38. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-85174-5_2
19. Hohenberger, S., Waters, B.: Realizing hash-and-sign signatures under standard assumptions. In: Joux, A. (ed.) EUROCRYPT 2009. LNCS, vol. 5479, pp. 333–350. Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-642-01001-9_19
20. Hohenberger, S., Waters, B.: Constructing verifiable random functions with large input spaces. In: Gilbert, H. (ed.) EUROCRYPT 2010. LNCS, vol. 6110, pp. 656–672. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-13190-5_33
21. Jager, T.: Verifiable random functions from weaker assumptions. In: Dodis, Y., Nielsen, J.B. (eds.) TCC 2015, Part II. LNCS, vol. 9015, pp. 121–143. Springer, Heidelberg (2015). https://doi.org/10.1007/978-3-662-46497-7_5
22. Jager, T., Kurek, R., Niehues, D.: Efficient adaptively-secure IB-KEMs and VRFs via near-collision resistance. In: Garay, J.A. (ed.) PKC 2021, Part I. LNCS, vol. 12710, pp. 596–626. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-75245-3_22
23. Jager, T., Niehues, D.: On the real-world instantiability of admissible hash functions and efficient verifiable random functions. In: Paterson, K.G., Stebila, D. (eds.) SAC 2019. LNCS, vol. 11959, pp. 303–332. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-38471-5_13
24. Katsumata, S.: On the untapped potential of encoding predicates by arithmetic circuits and their applications. In: Takagi, T., Peyrin, T. (eds.) ASIACRYPT 2017, Part III. LNCS, vol. 10626, pp. 95–125. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-70700-6_4
25. Katsumata, S., Yamada, S.: Partitioning via non-linear polynomial functions: more compact IBEs from ideal lattices and bilinear maps. In: Cheon, J.H., Takagi, T. (eds.) ASIACRYPT 2016, Part II. LNCS, vol. 10032, pp. 682–712. Springer, Heidelberg (2016). https://doi.org/10.1007/978-3-662-53890-6_23
26. Kohl, L.: Hunting and gathering – verifiable random functions from standard assumptions with short proofs. In: Lin, D., Sako, K. (eds.) PKC 2019, Part II. LNCS, vol. 11443, pp. 408–437. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-17259-6_14
27. Lysyanskaya, A.: Unique signatures and verifiable random functions from the DH-DDH separation. In: Yung, M. (ed.) CRYPTO 2002. LNCS, vol. 2442, pp. 597–612. Springer, Heidelberg (2002). https://doi.org/10.1007/3-540-45708-9_38
28. Niehues, D.: Verifiable random functions with optimal tightness. In: Garay, J.A. (ed.) PKC 2021, Part II. LNCS, vol. 12711, pp. 61–91. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-75248-4_3
29. Roşie, R.: Adaptive-secure VRFs with shorter keys from static assumptions. In: Camenisch, J., Papadimitratos, P. (eds.) CANS 2018. LNCS, vol. 11124, pp. 440–459. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-00434-7_22
30. Waters, B.: Efficient identity-based encryption without random oracles. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 114–127. Springer, Heidelberg (2005). https://doi.org/10.1007/11426639_7
31. Wilf, H.S.: Generating functionology. 3rd edn. A K Peters/CRC Press, Boca Raton (2005)
32. Yamada, S.: Adaptively secure identity-based encryption from lattices with asymptotically shorter public parameters. In: Fischlin, M., Coron, J.-S. (eds.) EUROCRYPT 2016, Part II. LNCS, vol. 9666, pp. 32–62. Springer, Heidelberg (2016). https://doi.org/10.1007/978-3-662-49896-5_2

33. Yamada, S.: Asymptotically compact adaptively secure lattice IBEs and verifiable random functions via generalized partitioning techniques. In: Katz, J., Shacham, H. (eds.) CRYPTO 2017, Part III. LNCS, vol. 10403, pp. 161–193. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-63697-9_6
34. Zhang, J., Chen, Yu., Zhang, Z.: Programmable hash functions from lattices: short signatures and IBEs with small key sizes. In: Robshaw, M., Katz, J. (eds.) CRYPTO 2016, Part III. LNCS, vol. 9816, pp. 303–332. Springer, Heidelberg (2016). https://doi.org/10.1007/978-3-662-53015-3_11

# Distributed Broadcast Encryption from Lattices

Jeffrey Champion(✉) and David J. Wu

University of Texas at Austin, Austin, TX, USA
jchampion@utexas.edu

**Abstract.** A broadcast encryption scheme allows a user to encrypt a message to $N$ recipients with a ciphertext whose size scales sublinearly with $N$. While broadcast encryption enables succinct encrypted broadcasts, it also introduces a strong trust assumption and a single point of failure; namely, there is a central authority who generates the decryption keys for all users in the system. Distributed broadcast encryption offers an appealing alternative where there is a one-time (trusted) setup process that generates a set of public parameters. Thereafter, users can independently generate their own public keys and post them to a public-key directory. Moreover, anyone can broadcast an encrypted message to any subset of user public keys with a ciphertext whose size scales sublinearly with the size of the broadcast set. Unlike traditional broadcast encryption, there are no long-term secrets in distributed broadcast encryption and users can join the system at any time (by posting their public key to the public-key directory).

Previously, distributed broadcast encryption schemes were known from standard pairing-based assumptions or from powerful tools like indistinguishability obfuscation or witness encryption. In this work, we provide the first distributed broadcast encryption scheme from a falsifiable lattice assumption. Specifically, we rely on the $\ell$-succinct learning with errors (LWE) assumption introduced by Wee (CRYPTO 2024). Previously, the only lattice-based candidate for distributed broadcast encryption goes through general-purpose witness encryption, which in turn is only known from the *private-coin* evasive LWE assumption, a strong and *non-falsifiable* lattice assumption. Along the way, we also describe a more direct construction of broadcast encryption from lattices.

## 1 Introduction

Suppose a user wants to encrypt a message to a set of users $S$. With vanilla public-key encryption, the encrypter would separately encrypt the message under each user's public key, and then broadcast the set of $|S|$ ciphertexts. Each user can read the message by decrypting their respective ciphertext in the broadcast. In this case, the size of the encrypted broadcast scales *linearly* with the size of the set $|S|$. Broadcast encryption [FN93] provides an elegant approach for achieving *succinct* encrypted broadcasts. With broadcast encryption, the

E. Boyle and M. Mahmoody (Eds.): TCC 2024, LNCS 15366, pp. 156–189, 2025.
https://doi.org/10.1007/978-3-031-78020-2_6

encrypter can encrypt a message to an arbitrary set of $S$ users with a ciphertext whose length scales *sublinearly* with $|S|$. However, broadcast encryption achieves this savings at a cost of introducing a central *trusted* authority that generates the public parameters for the scheme as well as each user's individual decryption key. Broadcast encryption thus has built-in key escrow, and indeed, if the central authority is ever compromised, then the attacker learns the secret keys for every single user in the system. This is in direct contrast to the setting with public-key encryption where each user generates their own cryptographic keys. A natural question is whether we can achieve the efficiency advantages of broadcast encryption *without* relying on a trusted centralized authority.

*Distributed Broadcast Encryption.* To circumvent the key escrow problem implicit in broadcast encryption, several works have introduced the notion of *distributed broadcast encryption* [WQZDF10, BZ14]. Distributed broadcast encryption is a hybrid between public-key encryption and broadcast encryption. Like the setting of public-key encryption, users in distributed broadcast encryption generate their own public/secret key-pairs and then post their public keys to a public-key directory (i.e., a public bulletin board). Anyone can encrypt a message to an arbitrary collection of public keys with a ciphertext whose size scales sublinearly with the size of the broadcast set (much like in traditional broadcast encryption). Note that in distributed broadcast encryption, we assume the encrypter and the decrypter know the set of public keys associated with a ciphertext (similar to how in broadcast encryption, both the encrypter and the decrypter know the set of users associated with the broadcast). While there is no trusted authority in distributed broadcast encryption, we do allow for a one-time trusted sampling of a set of public parameters. The trusted setup only needs to be performed once (e.g., using multiparty computation) and the same set of public parameters can be shared across multiple schemes. There are no long-term secrets in the scheme following the initial setup process. Thus, distributed broadcast encryption (and its generalizations) provide an elegant way to combine the decentralized, trustless nature of public-key encryption with the efficiency benefits of broadcast encryption.

To date, distributed broadcast encryption is known from indistinguishability obfuscation [BZ14], witness encryption [FWW23], as well as assumptions over bilinear groups [WQZDF10, KMW23, GKPW24]. The work of [FWW23] also shows how to generic construct a distributed broadcast encryption scheme from a registered attribute-based encryption (ABE) scheme; several recent works have shown how to construct registered ABE from pairing-based assumptions [HLWW23, ZZGQ23, GLWW24, AT24]. Among these constructions, the only one from plausibly post-quantum assumptions is the one based on witness encryption, which can be constructed using lattice assumptions [Tsa22, VWW22]—specifically, the evasive learning with errors (LWE) assumption [Wee22, Tsa22]. However, evasive LWE is a strong *non-falsifiable* lattice assumption, and moreover, existing constructions of witness encryption rely on a *private-coin* version of evasive LWE. As noted in [VWW22], there are (heuristic) obfuscation-based counter-examples for the general version of private-coin evasive LWE, so the status of private-coin evasive LWE remains unsettled. A natural goal then is to obtain simpler and more direct constructions of distributed broadcast encryption from (preferably falsifiable) lattice assumptions. An even better objective would be to obtain distributed broadcast encryption from the plain LWE assumption, but to date, even the simpler notion of centralized broadcast encryption from LWE remains a long-standing open problem. Existing centralized broadcast encryption schemes from lattice assumptions either lack a security proof [BV22], or rely on new lattice assumptions such as (public-coin) evasive LWE [Wee22] or $\ell$-succinct LWE [Wee24].

*This Work.* In this work, we give the first distributed broadcast encryption scheme from a *falsifiable* lattice assumption. Specifically, we rely on the $\ell$-succinct LWE assumption recently introduced by Wee [Wee24] for constructing broadcast encryption and succinct attribute-based encryption. The $\ell$-succinct LWE assumption essentially asserts that $(\mathbf{A}, \mathbf{s}^{\mathsf{T}}\mathbf{A} + \mathbf{e}^{\mathsf{T}})$ is pseudorandom even given a trapdoor for the related matrix $\mathbf{V} = [\mathbf{I}_\ell \otimes \mathbf{A} \mid \mathbf{U}]$ where $\mathbf{A} \xleftarrow{\text{R}} \mathbb{Z}_q^{n \times m}$, $\mathbf{s} \xleftarrow{\text{R}} \mathbb{Z}_q^n$, $\mathbf{e} \leftarrow \chi^m$, $\mathbf{U} \xleftarrow{\text{R}} \mathbb{Z}_q^{\ell n \times m}$, and $\chi$ is an error distribution. We provide more details in Sect. 1.1. The $\ell$-succinct LWE assumption is a falsifiable assumption and is implied by (public-coin) evasive LWE (in combination with LWE). Variants of this assumption (adapted to the setting of short integer solutions) have also been used in recent constructions of succinct functional commitments [ACL+22, WW23a, CLM23, BCFL23, WW23b, FMN23]. We summarize our results with the following informal theorem and provide a comparison to previous distributed broadcast encryption schemes in Table 1.

**Theorem 1 (Informal).** *Let $\lambda$ be a security parameter and $N$ be a bound on the number of users. Then, under the $\ell$-succinct LWE assumption (with $\ell \geq N \cdot O(\lambda \log N)$), there exists a distributed broadcast encryption scheme that supports up to $N$ users with the following properties:*

- *The public parameters consist of a structured string of size $N^2 \cdot \mathsf{poly}(\lambda, \log N)$.*
- *Each user's public key has size $O(N\lambda \log^2 N)$ and secret key has size $O(\lambda \log^2 N)$.*
- *An encryption to a set of $S \subseteq [N]$ users has size $O(\lambda \log^2 N)$.*
- *Encryption and decryption with respect to a set $S$ take time $|S| \cdot \mathsf{poly}(\lambda, \log N)$. Moreover, if the set $S$ is known in advance, we can precompute a set-dependent encryption key $\mathsf{pk}_S$; encrypting to the set $S$ then requires $\mathsf{poly}(\lambda, \log N)$ time. Similarly, each user $i \in S$ can also precompute a set-dependent decryption key $\mathsf{sk}_{S,i}$; decrypting a ciphertext associated with $S$ then requires $\mathsf{poly}(\lambda, \log N)$ time.*

*Open Problems.* Our work gives the first distributed broadcast encryption scheme from a falsifiable lattice assumption. Our scheme has a quadratic-size CRS. An interesting open problem is to obtain a distributed broadcast encryption scheme with a linear (or even sublinear-size) CRS from a falsifiable lattice assumption. Schemes with linear-size public parameters are known from bilinear

**Table 1.** Comparison with existing distributed broadcast encryption schemes. For each scheme, we report the size of the public parameters $\mathsf{pp}$, the user public/secret key-pair $(\mathsf{pk}, \mathsf{sk})$, and the ciphertext $\mathsf{ct}$ as a function of the number of users $N$, and the size of the broadcast set $|S|$. For simplicity of comparison, we suppress $\mathsf{poly}(\lambda, \log N)$ factors, where $\lambda$ is the security parameter. For each scheme, we also indicate whether the public parameters $\mathsf{pp}$ (if required) can be generated using a *transparent* setup procedure (TP), and whether it is (plausibly) post-quantum secure (PQ). The first two rows describe generic *non-succinct* approaches of using public-key encryption (PKE) or registration-based encryption (RBE) [GHMR18] to separately encrypt to each user in the broadcast set. We write $i\mathcal{O}$ to denote indistinguishability obfuscation [BGI+01, GGH+13]. The parameter $\ell$ in $\ell$-succinct LWE must satisfy $\ell \geq N \cdot O(\lambda \log N)$.

| Scheme | Assumption | $\lvert\mathsf{pp}\rvert$ | $\lvert\mathsf{pk}\rvert$ | $\lvert\mathsf{sk}\rvert$ | $\lvert\mathsf{ct}\rvert$ | TP | PQ |
|---|---|---|---|---|---|---|---|
| Generic | public-key encryption | – | 1 | 1 | $\lvert S\rvert$ | ✓ | ✓ |
| Generic | registration-based encryption | 1 | 1 | 1 | $\lvert S\rvert$ | ✓ | ✓ |
| [WQZDF10] | bilinear Diffie-Hellman exponent | $N$ | $N^2$ | $N$ | 1 | ✓ | ✗ |
| [BZ14] | $i\mathcal{O}$ + one-way function | – | 1 | 1 | 1 | ✓ | ✗ |
| [FWW23]∗ | witness encryption + LWE | 1 | 1 | 1 | 1 | ✓ | ✓ |
| [KMW23] | bilinear Diffie-Hellman exponent | $N$ | $N$ | 1 | 1 | ✗ | ✗ |
| [KMW23] | $k$-$\mathsf{Lin}$ (pairing group) | $N^2$ | $N$ | 1 | 1 | ✗ | ✗ |
| [GKPW24] | generic bilinear group | $N$ | $N$ | 1 | 1 | ✗ | ✗ |
| **This work** | $\ell$-succinct LWE | $N^2$ | $N$ | 1 | 1 | ✗ | ✓ |

∗ The work of [FWW23] also describe a generic approach for constructing distributed broadcast encryption from any registered attribute-based encryption (ABE) scheme. A number of recent works have shown how to construct registered ABE from bilinear maps [HLWW23, ZZGQ23, GLWW24, AT24]. Since these generic instantiations do not improve upon other the other bilinear-map-based constructions already shown in the table, we omit these for simplicity of comparison.

maps (under either the bilinear Diffie-Hellman exponent assumption or in the generic bilinear group model) [KMW23, GKPW24]. Another interesting question is to construct a distributed broadcast encryption scheme that is able to support an a priori unbounded number of users. Currently, this is only known from witness encryption and indistinguishability obfuscation. Note that if we alternatively impose a bound on the size of the broadcast set, then the transformation of [GLWW23] can be used to obtain a scheme that supports an arbitrary number of users (but where each ciphertext can only target a bounded subset of users).

*On the $\ell$-Succinct LWE Assumption.* Security of our lattice-based distributed broadcast encryption scheme relies on the $\ell$-succinct LWE assumption recently introduced by Wee [Wee24]. Prior to this work, distributed broadcast encryption was known from witness encryption [FWW23], which can be built from evasive LWE [Tsa22, VWW22]. Since both approaches rely on non-standard lattice assumptions, it is natural to ask whether it is worthwhile to study constructions from $\ell$-succinct LWE if we already have one from evasive LWE. We provide a brief discussion here and refer to [Wee24, §1.4] for additional perspectives.

First, unlike evasive LWE, the $\ell$-succinct LWE assumption is falsifiable. The $\ell$-succinct LWE assumption is also implied by (public-coin) evasive LWE together with plain LWE (c.f., [Wee24, §6.2]), so formally, $\ell$-succinct LWE is a *weaker* assumption than evasive LWE. On the other hand, evasive LWE is non-falsifiable, and must be carefully-formulated to avoid counter-examples. In particular, there are obfuscation-based counter-examples for the general version of *private-coin* evasive LWE [Wee22, VWW22]. Existing constructions of witness encryption based on evasive LWE [Tsa22, VWW22] all rely on private-coin versions of evasive LWE. Note that the known counter-examples for private-coin evasive LWE pertain only to the most general version of the assumption, and not to the specific distributions needed by [Tsa22, VWW22].

A second advantage of the $\ell$-succinct LWE assumption over evasive LWE is that it is "instance-independent." We reduce to the same assumption irrespective of the adversary. In contrast, when reducing security to evasive LWE, the matrices in the pre- and post-conditions are typically functions of the adversary (specifically, the queries that the adversary makes). Formally, this is captured by defining a sampling algorithm based on the adversary. So even though the evasive LWE post-condition itself is a falsifiable assumption, there is typically a *different* post-condition for each adversary. As such, when analyzing security, we are relying on a family of computational assumptions (one for each adversary) as opposed to a single instance-independent assumption (that applies to all adversaries). Since $\ell$-succinct LWE is falsifiable and instance-independent, the $\ell$-succinct LWE assumption provides a concrete target for cryptanalysis, especially compared to evasive LWE.

There has also recently been a proliferation of new (falsifiable) lattice assumptions. Most of these correspond to some variant of the short integer solutions (SIS) problem or the LWE problem with hints [ACL+22, WW23b, CLM23, BCFL23, WW23a, FMN23, AFLN24]; see [Alb24] for a survey and comparison. Essentially, these assumptions assert that SIS or LWE is hard with respect to a matrix $\mathbf{A}$ even given some structured preimage $\mathbf{A}^{-1}(\mathbf{P})$ for some matrix $\mathbf{P}$. Among these, the $\ell$-succinct SIS assumption is weaker (up to polynomial losses in the parameters) than assumptions like $\mathsf{BASIS}_{\mathsf{struct}}$ or $k$-$R$-$\mathsf{ISIS}$ assumptions considered in many of the aforementioned works. From this perspective, we believe $\ell$-succinct SIS and $\ell$-succinct LWE to be an appealing assumption to use when studying new lattice-based constructions.

Finally, if we compare our distributed broadcast encryption scheme directly to the one based on witness encryption, we obtain a much more direct construction (conceptually similar to classic pairing-based broadcast encryption schemes [BGW05, GW09]). For instance, the witness encryption approach makes heavy non-black-box use of cryptographic objects (specifically, the witness encryption scheme is applied to a function-binding hash function, which itself relies on leveled homomorphic encryption to construct). In contrast, our approach directly realizes the broadcast functionality and does not need any kind of homomorphic encryption machinery. We believe this to be a significant conceptual benefit of our approach.

## 1.1 Technical Overview

In this section, we provide a high-level overview of our approach for constructing distributed broadcast encryption from lattices.

*Notation.* We write $D_{\mathbb{Z},\sigma}$ to denote the discrete Gaussian distribution over $\mathbb{Z}$ with width parameter $\sigma > 0$. For a matrix $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ and a target vector $\mathbf{t} \in \mathbb{Z}_q^n$, we write $\mathbf{A}^{-1}(\mathbf{t})$ to denote a random variable $\mathbf{x} \leftarrow D_{\mathbb{Z},\sigma}^m$ conditioned on $\mathbf{A}\mathbf{x} = \mathbf{t}$. We can efficiently sample from $\mathbf{A}^{-1}(\mathbf{t})$ given a trapdoor for the matrix $\mathbf{A}$. To simplify the description in this overview, we use *curly underlines* to suppress small noise terms. Namely, we write $\utilde{\mathbf{s}^\mathsf{T}\mathbf{A}}$ to denote $\mathbf{s}^\mathsf{T}\mathbf{A} + \mathbf{e}^\mathsf{T}$ where $\mathbf{e}$ is a small error vector.

*Distributed Broadcast Encryption.* Next, we recall the syntax of a distributed broadcast encryption scheme [WQZDF10,BZ14]:

- **Setup:** In distributed broadcast encryption, there is an initial (trusted) setup algorithm that samples a set of public parameters $\mathsf{pp}$. Similar to [WQZDF10, KMW23, GKPW24], we assume an a priori bound $N$ on the maximum number of users, and allow the size of the public parameters to scale with $N$.
- **Key-generation:** In distributed broadcast encryption, each user has a distinct index $i \in [N]$. Using the public parameters $\mathsf{pp}$, user $i$ can generate a public/secret key-pair $(\mathsf{pk}_i, \mathsf{sk}_i)$. Typically, user $i$ would post the public key $\mathsf{pk}_i$ to the public key directory. As noted in Sect. 1.2, the notion of flexible broadcast encryption [FWW23] eliminates the need for a user index (i.e., users simply generate a public/secret key-pair). The work of [GLWW23] show how to generically transform a distributed broadcast encryption into a flexible broadcast encryption scheme. In this work, we just focus on the simpler notion of distributed broadcast encryption.
- **Encryption:** The encryption algorithm takes the public parameters $\mathsf{pp}$, a set of public keys $\{\mathsf{pk}_i\}_{i \in S}$, the message $\mu$, and outputs the ciphertext $\mathsf{ct}$.
- **Decryption:** The decryption algorithm takes a ciphertext, the public parameters $\mathsf{pp}$, the associated set of public keys $\{\mathsf{pk}_i\}_{i \in S}$, the secret key $\mathsf{sk}_i$ for $i \in S$, and outputs the message.

The security requirement says that an encryption of $\mu$ to a set of public keys $\{\mathsf{pk}_i\}_{i \in S}$ should computationally hide $\mu$ from an adversary who only sees the public parameters $\mathsf{pp}$ and the public keys $\{\mathsf{pk}_i\}_{i \in S}$ of the users in the broadcast set. We say the scheme is selectively secure if the adversary has to declare the indices $S \subseteq [N]$ of the honest users at the beginning of the security game *before* it sees the public keys, and that it is adaptively secure if the adversary can choose the set $S$ after seeing each user's public key (and selectively corrupting a subset of their keys). In this work, we are only able to prove selective security of our scheme; it is an interesting question to construct an adaptively secure distributed broadcast encryption scheme from lattice assumptions.[1]

*Starting Point: A (centralized) Broadcast Encryption Scheme.* We begin by describing a simple (centralized) broadcast encryption scheme for $N$ users. While previous lattice-based broadcast encryption schemes [BV22, Wee22, Wee24] start by constructing a ciphertext-policy ABE scheme with succinct ciphertexts, we take a more direct approach which notably does *not* rely on any of the homomorphic evaluation machinery typically seen in lattice-based ABE schemes. In turn, our approach more readily extends to support distributed key generation. The structure of our construction can be viewed as a lattice-based version of the pairing-based broadcast encryption scheme from [GW09, GKW18]. We describe our approach below:

– **Setup:** The master public key $\mathsf{mpk}$ for the broadcast encryption scheme is a tuple
$$\left(\mathbf{A}, \mathbf{B}, \mathbf{p}, \mathbf{W}_1, \ldots, \mathbf{W}_N, \mathbf{r}_1, \ldots, \mathbf{r}_N, \left\{\mathbf{A}^{-1}(\mathbf{W}_i \mathbf{r}_j)\right\}_{i \neq j}\right).$$
Here, $\mathbf{A}, \mathbf{B}, \mathbf{W}_i \xleftarrow{\text{R}} \mathbb{Z}_q^{n \times m}$, $\mathbf{p} \xleftarrow{\text{R}} \mathbb{Z}_q^n$, and $\mathbf{r}_i \leftarrow D_{\mathbb{Z},\sigma}^m$. The secret key for user $i \in [N]$ is
$$\mathsf{sk}_i = \mathbf{A}^{-1}(\mathbf{p} + \mathbf{B}\mathbf{r}_i + \mathbf{W}_i \mathbf{r}_i).$$
– **Encryption:** To encrypt a bit $\mu \in \{0,1\}$ to a set $S \subseteq [N]$, the encrypter samples an LWE secret $\mathbf{s} \xleftarrow{\text{R}} \mathbb{Z}_q^n$ and computes $\mathbf{W}_S = \sum_{j \in S} \mathbf{W}_j$. The ciphertext is
$$\mathsf{ct}_S = \left(\underset{\sim}{\mathbf{s}^{\mathsf{T}}\mathbf{A}}\ ,\ \underset{\sim}{\mathbf{s}^{\mathsf{T}}(\mathbf{B} + \mathbf{W}_S)}\ ,\ \underset{\sim}{\mathbf{s}^{\mathsf{T}}\mathbf{p} + \mu \cdot \lfloor q/2 \rceil}\right),$$
where we write $\lfloor \cdot \rceil$ to denote the function that rounds to the nearest integer.
– **Decryption:** Decryption relies on the fact that when $i \in S$, we have
$$\begin{aligned}\underset{\sim}{\mathbf{s}^{\mathsf{T}}\mathbf{A}}\left(\mathsf{sk}_i + \sum_{j \in S \setminus \{i\}} \mathbf{A}^{-1}(\mathbf{W}_j \mathbf{r}_i)\right) &\approx \mathbf{s}^{\mathsf{T}}\mathbf{p} + \mathbf{s}^{\mathsf{T}}\mathbf{B}\mathbf{r}_i + \mathbf{s}^{\mathsf{T}}\mathbf{W}_i\mathbf{r}_i + \sum_{j \in S \setminus \{i\}} \mathbf{s}^{\mathsf{T}}\mathbf{W}_j \mathbf{r}_i \\ &= \mathbf{s}^{\mathsf{T}}\mathbf{p} + \mathbf{s}^{\mathsf{T}}\mathbf{B}\mathbf{r}_i + \mathbf{s}^{\mathsf{T}}\mathbf{W}_S \mathbf{r}_i,\end{aligned}$$
where $\mathbf{A}^{-1}(\mathbf{W}_j \mathbf{r}_i)$ are the "cross-terms" from the master public key. To decrypt, user $i$ then computes
$$\underset{\sim}{\mathbf{s}^{\mathsf{T}}\mathbf{p} + \mu \cdot \lfloor q/2 \rceil} + \underset{\sim}{\mathbf{s}^{\mathsf{T}}(\mathbf{B} + \mathbf{W}_S)}\mathbf{r}_i - \underset{\sim}{\mathbf{s}^{\mathsf{T}}\mathbf{A}}\left(\mathsf{sk}_i + \sum_{j \in S \setminus \{i\}} \mathbf{A}^{-1}(\mathbf{W}_j \mathbf{r}_i)\right) \approx \mu \cdot \lfloor q/2 \rceil,$$
and rounds to recover $\mu$.

[1] We are limited to selective security because our security proof relies on a "partitioning" argument where the reduction algorithm first programs the challenge set into the public parameters. This limitation is common to most lattice-based ABE and broadcast encryption schemes [GVW13, BGG+14, DKW21, WWW22, Wee22, HLL23, Wee24].

We can prove selective security of this construction by relying on evasive LWE [Wee22, Tsa22].[2] The evasive LWE assumption essentially asserts that if $(\utilde{\mathbf{s}^\mathsf{T}\mathbf{A}}, \utilde{\mathbf{s}^\mathsf{T}\mathbf{P}})$ is pseudorandom, then $\utilde{\mathbf{s}^\mathsf{T}\mathbf{A}}$ is pseudorandom given $\mathbf{A}^{-1}(\mathbf{P})$. As noted above, in the selective security game, the adversary begins by declaring its challenge set $S^* \subseteq [N]$. It then receives the secret keys $\mathsf{sk}_i$ for all $i \notin S^*$ and its goal is to distinguish between encryptions of $\mu_0$ and $\mu_1$ to the set $S^*$. To prove selective security from evasive LWE, we leverage a partitioning argument where the reduction programs $\mathbf{B} := \mathbf{B}^* - \mathbf{W}_{S^*}$ where $\mathbf{B}^* \xleftarrow{\textsc{r}} \mathbb{Z}_q^{n\times m}$. Under evasive LWE, the claim now boils down to showing that

$$\utilde{\mathbf{s}^\mathsf{T}\mathbf{A}}\ ,\ \utilde{\mathbf{s}^\mathsf{T}\mathbf{B}^*}\ ,\ \utilde{\mathbf{s}^\mathsf{T}\mathbf{p}}\ ,\ \big\{\utilde{\mathbf{s}^\mathsf{T}\mathbf{W}_i\mathbf{r}_j}\big\}_{i\neq j}\ ,\ \big\{\utilde{\mathbf{s}^\mathsf{T}(\mathbf{p} + (\mathbf{B}^* - \mathbf{W}_{S^*})\mathbf{r}_i + \mathbf{W}_i\mathbf{r}_i)}\big\}_{i\notin S^*}$$

is pseudorandom. Since the details of this proof is immaterial to our subsequent construction (and analysis), we omit the formal details in this overview.[3]

*Distributed Key Generation.* To extend to distributed broadcast encryption, we partition the public parameters for the centralized broadcast encryption described above into two sets of components: one that is sampled by the initial (trusted) setup, and one that is sampled by each individual user:

- The components $(\mathbf{A}, \mathbf{B}, \mathbf{p}, \mathbf{r}_1, \ldots, \mathbf{r}_N)$ are part of the public parameters for the distributed broadcast encryption scheme.
- The matrices $\mathbf{W}_i$ as well as the cross terms $\mathbf{A}^{-1}(\mathbf{W}_i\mathbf{r}_j)$ for $j \neq i$ will be chosen by user $i$. Namely, the $i^{\text{th}}$ user's public key is $\mathsf{pk}_i = \big(\mathbf{W}_i, \{\mathbf{A}^{-1}(\mathbf{W}_i\mathbf{r}_j)\}_{j\neq i}\big)$. The decryption key for user $i$ is still $\mathsf{sk}_i = \mathbf{A}^{-1}(\mathbf{p} + \mathbf{B}\mathbf{r}_i + \mathbf{W}_i\mathbf{r}_i)$. At this point, it is *unclear* how user $i$ samples these components since it does *not* (and cannot) have a trapdoor for $\mathbf{A}$.

Observe that the public parameters $\mathsf{pp}$ together with any collection of public keys $\{\mathsf{pk}_i\}_{i\in S}$ now define a set of public parameters for the centralized broadcast encryption scheme (for $|S|$ users). Correctness now follows immediately. It suffices to build a mechanism for users to sample their public and secret keys *without* knowledge of a trapdoor for $\mathbf{A}$.

[2] Note that the security of our distributed broadcast encryption scheme will ultimately be based on the $\ell$-succinct LWE assumption [Wee24], which is a falsifiable assumption that is implied by evasive LWE. However, we do not know how to prove security of this particular centralized broadcast encryption scheme from $\ell$-succinct LWE. This is because our distributed broadcast encryption scheme will use a modified key-generation algorithm (described below).

[3] One approach is to first argue that $\utilde{\mathbf{s}^\mathsf{T}\mathbf{W}_i\mathbf{r}_j}$ is pseudorandom for all $i, j \in [N]$. Since $\mathbf{r}_j$ is short, we can use noise smudging to argue that $\utilde{\mathbf{s}^\mathsf{T}\mathbf{W}_i\mathbf{r}_j} \approx \utilde{(\mathbf{s}^\mathsf{T}\mathbf{W}_i + \mathbf{e}^\mathsf{T})}\mathbf{r}_j$, for a small error vector $\mathbf{e}$. Then, by LWE (with secret $\mathbf{s}$), this is indistinguishable from $\utilde{\mathbf{t}_i^\mathsf{T}}\mathbf{r}_i$, where $\mathbf{t}_i \xleftarrow{\textsc{r}} \mathbb{Z}_q^m$. We can now appeal to LWE again (with secret $\mathbf{t}_i$) to argue that this is pseudorandom. Since $\mathbf{r}_j$ is short, this step would rely on the analysis from [BLMR13].

*Sampling Public Keys.* To complete the construction, we need a way for a user to sample a public key $\mathsf{pk}_i = \left(\mathbf{W}_i, \left\{\mathbf{A}^{-1}(\mathbf{W}_i\mathbf{r}_j)\right\}_{j\neq i}\right)$ together with a secret key $\mathsf{sk}_i = \mathbf{A}^{-1}(\mathbf{p}+\mathbf{B}\mathbf{r}_i+\mathbf{W}_i\mathbf{r}_i)$ *without* a trapdoor for $\mathbf{A}$. For simplicity, consider first the simpler goal of sampling a fresh $\mathbf{W}_i \in \mathbb{Z}_q^{n\times m}$ together with short vectors $\mathbf{y}_j \in \mathbb{Z}_q^m$ where $\mathbf{A}\mathbf{y}_j = \mathbf{W}_i\mathbf{r}_j$ for all $j \in [N]$. To facilitate this, we can publish a collection of random matrices $\mathbf{Z}_1, \ldots, \mathbf{Z}_k \in \mathbb{Z}_q^{n\times m}$ in the public parameters together with their preimages $\mathbf{A}^{-1}(\mathbf{Z}_i\mathbf{r}_j)$ for all $i \in [k]$ and $j \in [N]$. The user can now pick a (short) vector $\mathbf{d} \in \mathbb{Z}_q^k$ and define $\mathbf{W}_i := \sum_{\tau\in[k]} d_\tau \mathbf{Z}_\tau$. Moreover, if $\mathbf{d}$ is short, then $\sum_{\tau\in[k]} d_\tau \mathbf{A}^{-1}(\mathbf{Z}_\tau\mathbf{r}_j)$ is a short preimage of $\sum_{\tau\in[k]} d_\tau \mathbf{Z}_\tau \mathbf{r}_j = \mathbf{W}_i\mathbf{r}_j$ for all $i, j \in [N]$. In essence, the public parameters contain $k$ public/secret key-pairs and the user samples their key by taking a random linear combination of the fixed keys in the public parameters. The hope then is that the user's public key $\mathbf{W}_i = \sum_{\tau\in[k]} d_\tau \mathbf{Z}_\tau$ and cross-terms $\sum_{\tau\in[k]} d_\tau \mathbf{A}^{-1}(\mathbf{Z}_\tau\mathbf{r}_j)$ hide the linear combination $\mathbf{d}$ the user used to generate their public/secret key-pair. While a Gaussian leftover hash lemma [AGHS13, AR16] can plausibly be used to show that the cross-terms are statistically close to $\mathbf{A}^{-1}(\sum_{\tau\in[k]} d_\tau \mathbf{Z}_\tau \mathbf{r}_j)$, we opt for a more direct approach inspired by recent constructions of functional commitments [WW23a, WW23b]. Namely, we publish a *full* trapdoor to facilitate direct sampling of the cross terms $\mathbf{A}^{-1}(\sum_{\tau\in[k]} d_\tau \mathbf{Z}_\tau \mathbf{r}_j)$ and the secret key. This approach is also more conducive to proving security from the $\ell$-succinct LWE assumption.

*Publishing a Trapdoor for a Related Matrix.* Instead of publishing short preimages $\mathbf{A}^{-1}(\mathbf{Z}_i\mathbf{r}_j)$ in the public parameters, we give out a *full* trapdoor for a matrix related to $\mathbf{A}$ in the public parameters. In particular, we define the matrix

$$\mathbf{V} = \left[\begin{array}{ccc|ccc} \mathbf{A} & & & -\mathbf{Z}_1\mathbf{r}_1 & \cdots & -\mathbf{Z}_k\mathbf{r}_1 \\ & \ddots & & \vdots & \ddots & \vdots \\ & & \mathbf{A} & -\mathbf{Z}_1\mathbf{r}_N & \cdots & -\mathbf{Z}_k\mathbf{r}_N \end{array}\right] = \left[\begin{array}{ccc|c} \mathbf{A} & & & -\mathbf{Z}(\mathbf{I}_k \otimes \mathbf{r}_1) \\ & \ddots & & \vdots \\ & & \mathbf{A} & -\mathbf{Z}(\mathbf{I}_k \otimes \mathbf{r}_N) \end{array}\right] \in \mathbb{Z}_q^{nN\times(mN+k)}, \tag{1.1}$$

where $\mathbf{Z} = [\mathbf{Z}_1 \mid \cdots \mid \mathbf{Z}_k] \in \mathbb{Z}_q^{n\times mk}$. Suppose we sample

$$\begin{bmatrix} \mathbf{y}_1 \\ \vdots \\ \mathbf{y}_N \\ \mathbf{d} \end{bmatrix} \leftarrow \mathbf{V}^{-1}(\mathbf{u}_i \otimes (\mathbf{p} + \mathbf{B}\mathbf{r}_i)) \in \mathbb{Z}_q^{mN+k}, \tag{1.2}$$

where $\mathbf{u}_i \in \mathbb{Z}_q^N$ denotes the $i^{\text{th}}$ canonical basis vector, and each $\mathbf{y}_j \in \mathbb{Z}_q^m$ and $\mathbf{d} \in \mathbb{Z}_q^k$. This means

$$\left[\begin{array}{ccc|c} \mathbf{A} & & & -\mathbf{Z}(\mathbf{I}_k \otimes \mathbf{r}_1) \\ & \ddots & & \vdots \\ & & \mathbf{A} & -\mathbf{Z}(\mathbf{I}_k \otimes \mathbf{r}_N) \end{array}\right] \cdot \begin{bmatrix} \mathbf{y}_1 \\ \vdots \\ \mathbf{y}_N \\ \mathbf{d} \end{bmatrix} = \begin{bmatrix} \mathbf{0}^n \\ \vdots \\ \mathbf{0}^n \\ \mathbf{p}+\mathbf{B}\mathbf{r}_i \\ \mathbf{0}^n \\ \vdots \\ \mathbf{0}^n \end{bmatrix} \in \mathbb{Z}_q^{nN}. \tag{1.3}$$

Next, by the mixed product rule for tensor (Kronecker) products (Eq. (2.1)), we can also write

$$\mathbf{Z}(\mathbf{I}_k \otimes \mathbf{r}_j)\mathbf{d} = \mathbf{Z}(\mathbf{I}_k \otimes \mathbf{r}_j)(\mathbf{d} \otimes \ 1) = \mathbf{Z}(\mathbf{d} \otimes \mathbf{I}_m)(1 \otimes \mathbf{r}_j) = \mathbf{Z}(\mathbf{d} \otimes \mathbf{I}_m)\mathbf{r}_j.$$

Define $\mathbf{W}_i := \mathbf{Z}(\mathbf{d} \otimes \mathbf{I}_m)$. Then, Eq. (1.3) says that for all $i \neq j$,

$$\forall j \neq i : \mathbf{A}\mathbf{y}_j - \mathbf{Z}(\mathbf{I}_k \otimes \mathbf{r}_j)\mathbf{d} = \mathbf{0}^n \implies \mathbf{A}\mathbf{y}_j = \mathbf{W}_i\mathbf{r}_j$$

and

$$\mathbf{A}\mathbf{y}_i - \mathbf{Z}(\mathbf{I}_k \otimes \mathbf{r}_i)\mathbf{d} = \mathbf{p} + \mathbf{B}\mathbf{r}_i \implies \mathbf{A}\mathbf{y}_i = \mathbf{p} + \mathbf{B}\mathbf{r}_i + \mathbf{W}_i\mathbf{r}_i.$$

These are the *same* relations for the public parameters and the secret key as in the centralized broadcast encryption scheme. Moreover, when $\mathbf{A} \xleftarrow{\text{R}} \mathbb{Z}_q^{n \times m}$ and $m \geq O(n \log q)$, the distribution of $\mathbf{d}$ output by Eq. (1.2) is distributed according to a discrete Gaussian. This follows implicitly from the Gaussian preimage sampling algorithm from [GPV08]; we also refer to [WW23b, §2] for a formal proof. Correspondingly then, when $k \geq O(nm \log q)$, the distribution of $\mathbf{W}_i = \mathbf{Z}(\mathbf{d} \otimes \mathbf{I}_m)$ is statistically close to uniform. Moreover the distribution of cross-terms $\mathbf{y}_j$ is distributed exactly according to $\mathbf{A}^{-1}(\mathbf{W}_i\mathbf{r}_j)$. As such, the public keys sampled using this procedure precisely coincide with the distribution in the original centralized broadcast encryption scheme. Putting all the pieces together, we now describe the full distributed broadcast encryption scheme:

- **Setup:** The public parameters $\mathsf{pp}$ consists of

$$\mathsf{pp} = (\mathbf{A}, \mathbf{B}, \mathbf{p}, \mathbf{r}_1, \ldots, \mathbf{r}_N, \mathbf{Z}, \mathsf{td}_{\mathbf{V}}),$$

  where $\mathbf{A}, \mathbf{B} \xleftarrow{\text{R}} \mathbb{Z}_q^{n \times m}$, $\mathbf{p} \xleftarrow{\text{R}} \mathbb{Z}_q^n$, and $\mathbf{r}_1, \ldots, \mathbf{r}_N \leftarrow D_{\mathbb{Z},\sigma}^m$ exactly as in the centralized broadcast encryption scheme. The additional components $\mathbf{Z}$ and $\mathsf{td}_{\mathbf{V}}$ are sampled as $\mathbf{Z} \xleftarrow{\text{R}} \mathbb{Z}_q^{n \times mk}$ and $\mathsf{td}_{\mathbf{V}}$ is a (random) trapdoor for the matrix $\mathbf{V}$ in Eq. (1.1).
- **Key generation:** To generate a public/secret key pair for an index $i \in [N]$, the user uses the trapdoor $\mathsf{td}_{\mathbf{V}}$ to sample $(\mathbf{y}_1, \ldots, \mathbf{y}_N, \mathbf{d})$ according to Eq. (1.2). It computes $\mathbf{W}_i = \mathbf{Z}(\mathbf{d} \otimes \mathbf{I}_m)$ and defines the public key to be $\mathsf{pk}_i = \big(\mathbf{W}_i, \{\mathbf{y}_j\}_{j \neq i}\big)$ and the secret key to be $\mathsf{sk}_i = \mathbf{y}_i$. As shown previously, for all $j \neq i$, it holds that $\mathbf{A}\mathbf{y}_j = \mathbf{W}_i\mathbf{r}_j$ and $\mathbf{A}\mathbf{y}_i = \mathbf{p} + \mathbf{B}\mathbf{r}_i + \mathbf{W}_i\mathbf{r}_i$.
- **Encryption and decryption:** These are the same as in the centralized broadcast encryption scheme. Specifically, the combination of the public parameters $\mathsf{pp}$ with the individual user public keys $\{\mathsf{pk}_i\}_{i \in [N]}$ can be viewed as a set of public parameters for the centralized broadcast encryption scheme. Since each user's secret key satisfies the same invariant as the centralized scheme, correctness follows as before.

We give the formal description in Sect. 3.1.

*N-Structured LWE.* To prove security, we rely on the $N$-structured LWE assumption which asserts that

$$(\mathbf{A}, \mathbf{s}^{\mathsf{T}}\utilde{\mathbf{A}}, \mathbf{Z}, \mathbf{r}_1, \ldots, \mathbf{r}_N, \mathsf{td}_{\mathbf{V}}) \approx (\mathbf{A}, \mathbf{v}^{\mathsf{T}}, \mathbf{Z}, \mathbf{r}_1, \ldots, \mathbf{r}_N, \mathsf{td}_{\mathbf{V}}), \tag{1.4}$$

where $\mathbf{V}$ is the matrix in Eq. (1.1), $\mathsf{td}_{\mathbf{V}}$ is a random trapdoor for $\mathbf{V}$, and $\mathbf{A} \xleftarrow{\text{R}} \mathbb{Z}_q^{n \times m}$, $\mathbf{s} \xleftarrow{\text{R}} \mathbb{Z}_q^n$, $\mathbf{v} \xleftarrow{\text{R}} \mathbb{Z}_q^m$, $\mathbf{Z} \xleftarrow{\text{R}} \mathbb{Z}_q^{n \times mk}$, and $\mathbf{r}_1, \ldots, \mathbf{r}_N \leftarrow D_{\mathbb{Z},\sigma}^m$. Later on, we will show that the $N$-structured LWE assumption follows from the $\ell$-succinct LWE assumption recently introduced by Wee [Wee24]. We discuss both assumptions at the end of this section.

*Proof Strategy.* We now provide a sketch of our security proof, and specifically, how the reduction algorithm simulates the key-generation queries. In the selective security game, the adversary begins by committing to the set of indices $S^* \subseteq [N]$ associated with the challenge ciphertext. The reduction algorithm obtains $(\mathbf{A}, \mathbf{v}^{\mathsf{T}}, \mathbf{Z}, \mathbf{r}_1, \ldots, \mathbf{r}_N, \mathsf{td}_{\mathbf{V}})$ from the $\ell$-structured LWE challenger. It uses $\mathbf{A}, \mathbf{Z}, \mathbf{r}_1, \ldots, \mathbf{r}_N, \mathsf{td}_{\mathbf{V}}$ as the corresponding components of the public parameters for the distributed broadcast encryption scheme. The question is how the reduction algorithm simulates the public keys $\mathsf{pk}_i = (\mathbf{W}_i, \{\mathbf{y}_{i,j}\}_{j \neq i})$ for the honest users and how it simulates the challenge ciphertext. Suppose for a moment that the reduction algorithm *knew* the $\mathbf{W}_i$ for each index $i \in S^*$ in the challenge set. Then, it would be able to compute $\mathbf{W}_{S^*} = \sum_{i \in S^*} \mathbf{W}_i$ and set $\mathbf{B} = \mathbf{A}\mathbf{H} - \mathbf{W}_{S^*}$, $\mathbf{p} = \mathbf{A}\mathbf{h}$ where $\mathbf{H} \xleftarrow{\text{R}} \{0,1\}^{m \times m}$ and $\mathbf{h} \xleftarrow{\text{R}} \{0,1\}^m$. In this case, the reduction could define the challenge ciphertext to be

$$\mathsf{ct}_{S^*} = \big(\mathbf{v}^{\mathsf{T}}, \mathbf{v}^{\mathsf{T}}\mathbf{H}, \mathbf{v}^{\mathsf{T}}\mathbf{h} + \mu \cdot \lfloor q/2 \rceil\big).$$

If $\mathbf{v}^{\mathsf{T}} = \utilde{\mathbf{s}^{\mathsf{T}}\mathbf{A}}$, then we have

$$\mathsf{ct}_{S^*} = \big(\utilde{\mathbf{s}^{\mathsf{T}}\mathbf{A}}, \utilde{\mathbf{s}^{\mathsf{T}}\mathbf{A}\mathbf{H}}, \utilde{\mathbf{s}^{\mathsf{T}}\mathbf{A}\mathbf{h} + \mu \cdot \lfloor q/2 \rceil}\big) = \big(\utilde{\mathbf{s}^{\mathsf{T}}\mathbf{A}}, \utilde{\mathbf{s}^{\mathsf{T}}(\mathbf{B} + \mathbf{W}_{S^*})}, \utilde{\mathbf{s}^{\mathsf{T}}\mathbf{p} + \mu \cdot \lfloor q/2 \rceil}\big),$$

which is distributed according to the real scheme. If $\mathbf{v}$ is a random vector, then by the leftover hash lemma, the challenge ciphertext is uniformly random and security holds.

The problem with this approach is that the reduction algorithm *cannot* choose $\mathbf{W}_i$ arbitrarily. Recall that $\mathbf{W}_i$ is a component of the public key, and in the real scheme, is derived by first sampling $(\mathbf{y}_{i,1}, \ldots, \mathbf{y}_{i,N}, \mathbf{d}_i)$ from $\mathbf{V}^{-1}(\mathbf{u}_i \otimes (\mathbf{p} + \mathbf{B}\mathbf{r}_i))$ according to Eq. (1.3) and then setting $\mathbf{W}_i = \mathbf{Z}(\mathbf{d}_i \otimes \mathbf{I}_m)$. Here, we immediately run into a circularity issue. The reduction algorithm needs to know $\mathbf{W}_i$ in order to program the challenge set $S^*$ into $\mathbf{B}$, but sampling $\mathbf{W}_i$ seemingly requires that $\mathbf{B}$ is already fixed!

Thus, the reduction algorithm needs an alternative method for simulating the honest users' public keys. The observation is simple: the public key $\mathsf{pk}_i$ for an index $i \in S^*$ only depends on $\mathbf{y}_{i,j}$ for $j \neq i$ and $\mathbf{d}$; importantly, $\mathsf{pk}_i$ does not depend on the value of $\mathbf{y}_{i,i}$. Indeed, $\mathbf{y}_{i,i}$ is the secret key for user $i$ which is not revealed to the adversary and also *cannot* be known to the reduction. Thus, in the reduction, instead of sampling $\mathbf{y}_{i,i}$ so that $\mathbf{A}\mathbf{y}_{i,i} = \mathbf{p} + \mathbf{B}\mathbf{r}_i + \mathbf{W}_i\mathbf{r}_i$ as in the real scheme, the reduction algorithm simply samples $\mathbf{y}_{i,i}$ so that $\mathbf{A}\mathbf{y}_{i,i} = \mathbf{W}_i\mathbf{r}_i$. In other words, the reduction algorithm samples $(\mathbf{y}_{i,1}, \ldots, \mathbf{y}_{i,N}, \mathbf{d}_i)$ from $\mathbf{V}^{-1}(\mathbf{0}^{nN})$. By the structure of $\mathbf{V}$ (see Eq. (1.1)), we can show that sampling from this distribution does *not* affect the marginal distributions of $\mathbf{y}_{i,j}$ for $j \neq i$ and $\mathbf{d}$. As such, this does not affect the adversary's view. With this modified sampling procedure, the reduction algorithm is able to sample the $\mathbf{W}_i$ components of each public key (independently of $\mathbf{B}$), and then program $\mathbf{W}_{S^*} = \sum_{i \in S^*} \mathbf{W}_i$ into the public parameters (as described above). We provide the full details in Sect. 3.1.

*$N$-structured LWE and $\ell$-succinct LWE.* The above reduction relies on the $N$-structured LWE assumption (Eq. (1.4)) which essentially asserts hardness of LWE given a trapdoor for the related matrix $\mathbf{V}$ used in our construction. We can relate this assumption to the recently introduced $\ell$-succinct LWE assumption [Wee24] which asserts that

$$(\mathbf{A}, \underset{\sim}{\mathbf{s}^\mathsf{T}\mathbf{A}}, \mathbf{U}, \mathsf{td}_\mathbf{V}) \approx (\mathbf{A}, \mathbf{v}^\mathsf{T}, \mathbf{U}, \mathsf{td}_\mathbf{V}), \tag{1.5}$$

where $\mathbf{A} \xleftarrow{\text{R}} \mathbb{Z}_q^{n \times m}$, $\mathbf{s} \xleftarrow{\text{R}} \mathbb{Z}_q^n$, $\mathbf{v} \xleftarrow{\text{R}} \mathbb{Z}_q^m$, $\mathbf{U} \xleftarrow{\text{R}} \mathbb{Z}_q^{\ell n \times m}$, $\mathbf{V} = [\mathbf{I}_\ell \otimes \mathbf{A} \mid \mathbf{U}]$ and $\mathsf{td}_\mathbf{V}$ is a random trapdoor for the matrix $\mathbf{V}$. The $\ell$-succinct LWE assumption is a falsifiable assumption and moreover, Wee showed that it is implied by the (public-coin) evasive LWE assumption [Wee24]. Wee also showed how to leverage $\ell$-succinct LWE to construct an ABE scheme with succinct ciphertexts, which in particular, implies a (centralized) broadcast encryption scheme with short ciphertexts (and long public parameters). The analogous $\ell$-succinct short integer solutions (SIS) assumption (i.e., SIS is hard with respect to $\mathbf{A}$ given a trapdoor for $\mathbf{V}$) has been used to construct succinct functional commitments [WW23a]. As shown in [Wee24], the $\ell$-succinct SIS assumption is the least-structured or weakest among the multitude of structured lattice assumptions (e.g., $\mathsf{BASIS}_{\mathsf{struct}}$ [WW23b] or $k$-$R$-$\mathsf{ISIS}$ [ACL+22]) that have been introduced in recent years.

In Sect. 4, we show that if the $\ell$-succinct LWE assumption holds with parameter $\ell \geq N \cdot O(\lambda \log N)$, then the $N$-structured LWE assumption also holds, provided that the width parameter $k$ (i.e., the number of blocks in $\mathbf{Z}$) is at least $k \geq O(nm \log q)$. While it may appear that the $N$-structured LWE assumption gives out a trapdoor for a more structured matrix than the $N$-succinct LWE assumption, we show here that they are very similar. We illustrate this with a simple example. In the following description, we write $[\mathbf{I}_N \otimes \mathbf{A} \mid \mathbf{M}_{\mathbf{Z},\mathbf{R}}]$ (for $\mathbf{R} = [\mathbf{r}_1 \mid \cdots \mid \mathbf{r}_N]$) to denote the matrix $\mathbf{V}$ from Eq. (1.1) and $[\mathbf{I}_N \otimes \mathbf{A} \mid \mathbf{U}]$ to denote the matrix $\mathbf{V}$ from the $\ell$-succinct LWE assumption (Eq. (1.5)). In particular,

$$\mathbf{M}_{\mathbf{Z},\mathbf{R}} \coloneqq \begin{bmatrix} -\mathbf{Z}(\mathbf{I}_k \otimes \mathbf{r}_1) \\ \vdots \\ -\mathbf{Z}(\mathbf{I}_k \otimes \mathbf{r}_N) \end{bmatrix} \in \mathbb{Z}_q^{nN \times k} \quad \text{and} \quad \mathbf{U} = \begin{bmatrix} \mathbf{U}_1 \\ \vdots \\ \mathbf{U}_\ell \end{bmatrix} \in \mathbb{Z}_q^{n\ell \times m}.$$

Suppose we sample $(\mathbf{y}_1, \ldots, \mathbf{y}_\ell, \mathbf{r})$ from $[\mathbf{I}_\ell \otimes \mathbf{A} \mid \mathbf{U}]^{-1}(\mathbf{0}^{n\ell})$, where $\mathbf{y}_i \in \mathbb{Z}_q^m, \mathbf{r} \in \mathbb{Z}_q^m$. This is statistically indistinguishable from sampling

$$\mathbf{r} \leftarrow D_{\mathbb{Z},\sigma}^m \quad \text{and} \quad \forall i \in [\ell] : \mathbf{y}_i \leftarrow \mathbf{A}^{-1}(-\mathbf{U}_i \mathbf{r}). \tag{1.6}$$

On the other hand, suppose we sample $(\hat{\mathbf{y}}_1, \ldots, \hat{\mathbf{y}}_N, \hat{\mathbf{d}})$ from $[\mathbf{I}_N \otimes \mathbf{A} \mid \mathbf{M}_{\mathbf{Z},\mathbf{R}}]^{-1}$ $(\mathbf{0}^{nN})$, where $\hat{\mathbf{y}}_i \in \mathbb{Z}_q^m$ and $\hat{\mathbf{d}} \in \mathbb{Z}_q^k$. This is statistically indistinguishable from sampling

$$\hat{\mathbf{d}} \leftarrow D_{\mathbb{Z},\sigma}^k \quad \text{and} \quad \forall j \in [N] : \hat{\mathbf{y}}_i \leftarrow \mathbf{A}^{-1}(\mathbf{Z}(\hat{\mathbf{d}} \otimes \mathbf{I}_m)\mathbf{r}_i). \tag{1.7}$$

Since $\hat{\mathbf{d}} \leftarrow D_{\mathbb{Z},\sigma}^k$, when $k \geq O(nm \log q)$, the marginal distribution of $\mathbf{Z}(\hat{\mathbf{d}} \otimes \mathbf{I}_m)$ is statistically close to uniform. If we define $\mathbf{U} := \mathbf{Z}(\hat{\mathbf{d}} \otimes \mathbf{I}_m)$, then the distribution in Eq. (1.7) becomes

$$\hat{\mathbf{d}} \leftarrow D_{\mathbb{Z},\sigma}^k \quad \text{and} \quad \forall j \in [N] : \hat{\mathbf{y}}_i \leftarrow \mathbf{A}^{-1}(\mathbf{U}\mathbf{r}_i). \tag{1.8}$$

This is a similar "cross-term" structure as in Eq. (1.6), except with the roles of $\mathbf{U}$ and $\mathbf{r}$ interchanged (i.e., the same $\mathbf{r}$ is used for all $i \in [\ell]$ in Eq. (1.6) while the same $\mathbf{U}$ is used for all $i \in [N]$ in Eq. (1.8)). By "transposing" a collection of preimages sampled as in Eq. (1.6), we can transform them into a collection of preimages distributed as in Eq. (1.8). To simulate the $\mathbf{Z}$ and $\hat{\mathbf{d}}$ components that determine the $\mathbf{U}$ matrix in Eq. (1.8), we rely on preimage sampling techniques. We provide a formal reduction in Sect. 4 (Theorem 8).

## 1.2 Additional Related Work

*Decentralized broadcast encryption.* An alternative approach for solving the key-escrow problem in broadcast encryption is to rely on an *interactive* key-generation process. This is referred to as *decentralized broadcast encryption* [PPS12]. Namely, when a new user joins the system, the users in the system runs an MPC protocol with the existing users to obtain their secret key (and existing users obtain an updated key). In distributed broadcast encryption, key-generation is non-interactive and we do not require users to be cognizant of other users in the system.

*Flexible Broadcast Encryption.* In distributed broadcast encryption, each user's public key is actually associated with a slot index $i \in [N]$. Moreover, a user can only encrypt to a set of public keys if they occupy different slots. The work of [FWW23] introduced a stronger notion of *flexible* broadcast encryption where it is possible to encrypt to an *arbitrary* set of public keys without any slot restrictions. In the same work, the authors showed how to construct flexible broadcast encryption using witness encryption (together with a function-binding hash function). Recently, the work of [GLWW23] showed a generic compiler from distributed broadcast encryption to flexible broadcast encryption using combinatoric tools. The work of [GKPW24] also provides a direct construction of flexible broadcast encryption from pairings.

*Registration-Based Cryptography.* Distributed broadcast encryption falls into the more general umbrella of "registration-based cryptography" [GHMR18], which seeks to remove the trusted authority from advanced encryption schemes like identity-based encryption (IBE) [GHMR18, GHM+19, GV20, CES21, GKMR23, DKL+23, FKdP23], attribute-based encryption (ABE) [HLWW23, FWW23, ZZGQ23, GLWW24, AT24], functional encryption (FE) [FFM+23, DPY23], and traitor tracing [BLM+24]. Broadly speaking, the goal in each of these settings is to replace the trusted key-issuing authority with a public bulletin board where users can post their own public keys (that they themselves sample). Moreover a (transparent) key curator can then aggregate the individual public keys into a single short set of public parameters. The work of [FWW23] also shows how to compile any registered ABE scheme (that supports a single attribute and the always-accept policy) into a distributed broadcast encryption scheme (with *succinct* ciphertexts). Existing constructions of registered ABE either rely on indistinguishability obfuscation [HLWW23], witness encryption [FWW23], or pairing-based assumptions [HLWW23, ZZGQ23, GLWW24, AT24].

## 2 Preliminaries

Throughout this work, we write $\lambda$ to denote the security parameter. For a positive integer $n \in \mathbb{N}$, we write $[n] := \{1, \ldots, n\}$. We write $\mathsf{poly}(\lambda)$ to denote a fixed polynomial in $\lambda$. We write $\mathsf{negl}(\lambda)$ to denote a function that is negligible in $\lambda$: namely, $o(\lambda^{-c})$ for all $c \in \mathbb{N}$. We say an event occurs with overwhelming probability if the probability of its complement occurring is negligible. For functions $f = f(\lambda)$ and $g = g(\lambda)$, we write $f \geq O(g)$ to denote that there exists a fixed function $g' \in O(g)$ such that $f(\lambda) \geq g'(\lambda)$ for all $\lambda \in \mathbb{N}$. We say an algorithm is efficient if it runs in probabilistic polynomial time in the length of its input. For two ensembles of distributions $\mathcal{D}_1 = \{\mathcal{D}_{1,\lambda}\}_{\lambda \in \mathbb{N}}$ and $\mathcal{D}_2 = \{\mathcal{D}_{2,\lambda}\}_{\lambda \in \mathbb{N}}$ indexed by a security parameter, we say they are computationally indistinguishable if no efficient algorithm can distinguish them except with $\mathsf{negl}(\lambda)$ probability. We say they are statistically indistinguishable if the statistical distance between them is $\mathsf{negl}(\lambda)$. We write $\mathcal{D}_1 \overset{c}{\approx} \mathcal{D}_2$ (resp., $\mathcal{D}_1 \overset{\mathrm{R}}{\leftarrow} \mathcal{D}_2$) if $\mathcal{D}_1$ and $\mathcal{D}_2$ are computationally (resp., statistically) indistinguishable. Throughout this work, we will use bold uppercase letters (e.g., $\mathbf{A}$, $\mathbf{B}$) to denote matrices and bold lowercase letters (e.g., $\mathbf{u}$, $\mathbf{v}$) to denote vectors. We use non-boldface letters (e.g., $v_1, \ldots, v_n$) to refer their components. For a dimension $n \in \mathbb{N}$, we write $\mathbf{I}_n \in \mathbb{Z}^{n \times n}$ to denote the identity matrix of dimension $n$. Throughout, we write $\|\cdot\|$ to denote the $\ell_\infty$ norm.

*Tensor Products.* For matrices $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ and $\mathbf{B} \in \mathbb{Z}_q^{k \times \ell}$, we write $\mathbf{A} \otimes \mathbf{B} \in \mathbb{Z}_q^{nk \times m\ell}$ to denote their tensor (Kronecker) product. For matrices $\mathbf{A}, \mathbf{B}, \mathbf{C}, \mathbf{D}$ where the products $\mathbf{AC}$ and $\mathbf{BD}$ are well-defined, then

$$(\mathbf{A} \otimes \mathbf{B})(\mathbf{C} \otimes \mathbf{D}) = (\mathbf{AC}) \otimes (\mathbf{BD}). \tag{2.1}$$

We now recall a generalization of the leftover hash lemma along with a simple corollary that will be useful in our analysis.

**Lemma 1 (Generalized Leftover Hash Lemma** [ABB10, Lemma 13, adapted]**).** *Let $n, m, q$ be integers such that $m \geq 2n \log q$ and $q > 2$ is prime. Then, for all fixed vectors $\mathbf{e} \in \mathbb{Z}_q^m$ and all $k = \mathsf{poly}(n)$, the statistical distance between the following distributions is $\mathsf{negl}(n)$:*

$$\left\{ (\mathbf{A}, \mathbf{AR}, \mathbf{e}^{\mathsf{T}}\mathbf{R}) : \begin{matrix} \mathbf{A} \overset{\mathrm{R}}{\leftarrow} \mathbb{Z}_q^{n \times m} \\ \mathbf{R} \overset{\mathrm{R}}{\leftarrow} \{0,1\}^{m \times k} \end{matrix} \right\} \text{ and } \left\{ (\mathbf{A}, \mathbf{U}, \mathbf{e}^{\mathsf{T}}\mathbf{R}) : \begin{matrix} \mathbf{A} \overset{\mathrm{R}}{\leftarrow} \mathbb{Z}_q^{n \times m}, \mathbf{U} \overset{\mathrm{R}}{\leftarrow} \mathbb{Z}_q^{n \times k} \\ \mathbf{R} \overset{\mathrm{R}}{\leftarrow} \{0,1\}^{m \times k} \end{matrix} \right\}.$$

**Corollary 1 (Column Space of Random Matrix** [GPV08, Lemma 5.1]**).** *Let $n, m, q$ be lattice parameters where $q$ is prime and $m \geq 2n \log q$. Then, for all but a $q^{-n}$ fraction of matrices $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$, the columns of $\mathbf{A}$ generate $\mathbb{Z}_q^n$.*

*Discrete Gaussians and Gadget Matrices.* We write $D_{\mathbb{Z},\sigma}$ to denote the discrete Gaussian distribution over $\mathbb{Z}$ with width parameter $\sigma > 0$. For a matrix $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ and a target vector $\mathbf{t} \in \mathbb{Z}_q^n$ in the column-space of $\mathbf{A}$, we write $\mathbf{A}_\sigma^{-1}(\mathbf{t})$ to denote a random variable $\mathbf{x} \leftarrow D_{\mathbb{Z},\sigma}^m$ conditioned on $\mathbf{A}\mathbf{x} = \mathbf{t} \bmod q$. We extend $\mathbf{A}_\sigma^{-1}$ to matrices by applying $\mathbf{A}_\sigma^{-1}$ to each column of the input. For positive integers $n, q \in \mathbb{N}$, let $\mathbf{G}_n = \mathbf{I}_n \otimes \mathbf{g}^\intercal \in \mathbb{Z}_q^{n \times m'}$ be the gadget matrix [MP12] where $\mathbf{I}_n$ is the identity matrix of dimension $n$, $\mathbf{g}^\intercal = [1, 2, \ldots, 2^{\lfloor \log q \rfloor}]$, and $m' = n(\lfloor \log q \rfloor + 1)$. We also recall some basic properties of the discrete Gaussian distribution.

**Lemma 2 (Gaussian Tail Bound** [MP12, Lemma 2.6, adapted]**).** *Let $n, m, q$ be lattice parameters where $m \geq 2n \log q$. Sample $\mathbf{A} \xleftarrow{\text{R}} \mathbb{Z}_q^{n \times m}$. Then, for all $\sigma > \log m$ and all vectors $\mathbf{t} \in \mathbb{Z}_q^n$ in the span of $\mathbf{A}$,*

$$\Pr[\|\mathbf{u}\| > \sqrt{m}\sigma : \mathbf{u} \leftarrow \mathbf{A}_\sigma^{-1}(\mathbf{t})] \leq O(2^{-m}).$$

*For the particular case of the discrete Gaussian over the integers and any $\lambda \in \mathbb{N}$,*

$$\Pr[|x| > \sqrt{\lambda}\sigma : x \leftarrow D_{\mathbb{Z},\sigma}] \leq 2^{-\lambda}.$$

**Lemma 3 (Gaussian Samples** [GPV08, adapted]**).** *Let $n, m, q, \sigma$ be lattice parameters such that $\sigma \geq \log m$, $m \geq 2n \log q$, and $q$ is prime. Then the statistical distance between the following distributions is at most $\mathsf{negl}(n)$:*

$$\left\{(\mathbf{A}, \mathbf{x}, \mathbf{A}\mathbf{x}) : \mathbf{A} \xleftarrow{\text{R}} \mathbb{Z}_q^{n \times m}, \mathbf{x} \leftarrow D_{\mathbb{Z},\sigma}^m\right\} \text{ and } \left\{(\mathbf{A}, \mathbf{x}, \mathbf{t}) : \begin{array}{c} \mathbf{A} \xleftarrow{\text{R}} \mathbb{Z}_q^{n \times m} \\ \mathbf{t} \xleftarrow{\text{R}} \mathbb{Z}_q^n, \mathbf{x} \leftarrow \mathbf{A}_\sigma^{-1}(\mathbf{t}) \end{array}\right\}.$$

*Basis Extension and Lattice Trapdoors.* We will also use the following lemma characterizing the distribution of $[\mathbf{A} \mid \mathbf{B}]^{-1}(\cdot)$. We give the statement from [WW23b], which follows immediately from earlier works on preimage sampling and basis delegation [GPV08, CHKP10, MP12]. Finally, we recall the notion of a gadget trapdoor [MP12].

**Lemma 4 (Marginal of Gaussian Preimages** [WW23b]**).** *Let $n, m, q$ be lattice parameters where $m \geq 2n \log q$ and $q$ is prime. Let $\mathbf{B} \in \mathbb{Z}_q^{n\ell \times k}$ where $\ell, k = \mathsf{poly}(n, \log q)$. Let $\mathbf{C} = [\mathbf{I}_\ell \otimes \mathbf{A} \mid \mathbf{B}] \in \mathbb{Z}_q^{n\ell \times (m\ell + k)}$. Then for all target vectors $\mathbf{t} \in \mathbb{Z}_q^{n\ell}$ and all width parameters $s \geq \log(\ell m)$, the statistical distance between the following distributions is $\mathsf{negl}(n)$:*

$$\left\{\mathbf{v} : \mathbf{A} \xleftarrow{\text{R}} \mathbb{Z}_q^{n \times m}, \mathbf{v} \leftarrow \mathbf{C}_s^{-1}(\mathbf{t})\right\} \text{ and } \left\{\begin{bmatrix} \mathbf{v}_1 \\ \mathbf{v}_2 \end{bmatrix} : \begin{array}{c} \mathbf{A} \xleftarrow{\text{R}} \mathbb{Z}_q^{n \times m}, \mathbf{v}_2 \leftarrow D_{\mathbb{Z},s}^k \\ \mathbf{v}_1 \leftarrow (\mathbf{I}_\ell \otimes \mathbf{A})_s^{-1}(\mathbf{t} - \mathbf{B}\mathbf{v}_2) \end{array}\right\}.$$

**Lemma 5 (Gadget Trapdoor** [Ajt96, GPV08, MP12]**).** *Let $n, m, q$ be lattice parameters with $m \geq 3n \log q$. Then there exists efficient algorithms $(\mathsf{TrapGen}, \mathsf{SamplePre})$ with the following syntax:*

- $\mathsf{TrapGen}(1^n, q, m) \to (\mathbf{A}, \mathbf{R})$: *On input the lattice dimension $n$, the modulus $q$, and the number of samples $m$, the trapdoor-generation algorithm outputs a matrix $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ together with a trapdoor $\mathbf{R} \in \mathbb{Z}_q^{m \times m'}$ where $m' = n(\lfloor \log q \rfloor + 1)$.*
- $\mathsf{SamplePre}(\mathbf{A}, \mathbf{R}, \mathbf{t}, \sigma) \to \mathbf{x}$: *On input a matrix $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$, a trapdoor $\mathbf{R} \in \mathbb{Z}_q^{m \times m'}$, a target vector $\mathbf{t} \in \mathbb{Z}_q^n$, and a Gaussian width parameter $\sigma$, the preimage-sampling algorithm outputs a vector $\mathbf{x} \in \mathbb{Z}_q^m$.*

*Moreover, the above algorithms satisfy the following properties:*

- **Trapdoor distribution:** *If $(\mathbf{A}, \mathbf{R}) \leftarrow \mathsf{TrapGen}(1^n, q, m)$ and $\mathbf{A}' \xleftarrow{\text{R}} \mathbb{Z}_q^{n \times m}$, then $\Delta(\mathbf{A}, \mathbf{A}') = \mathsf{negl}(n)$. Moreover, $\mathbf{A}\mathbf{R} = \mathbf{G}_n \in \mathbb{Z}_q^{n \times m'}$ and $\|\mathbf{R}\| = 1$.*
- **Preimage sampling:** *For all matrices $\mathbf{R} \in \mathbb{Z}_q^{m \times m'}$, parameters $\sigma > 0$, and all target vectors $\mathbf{t} \in \mathbb{Z}_q^n$ in the column span of $\mathbf{A}$, the output $\mathbf{x} \leftarrow \mathsf{SamplePre}(\mathbf{A}, \mathbf{R}, \mathbf{t}, \sigma)$ satisfies $\mathbf{A}\mathbf{x} = \mathbf{t}$.*
- **Preimage distribution:** *Suppose $\mathbf{R}$ is a gadget trapdoor for $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ (i.e., $\mathbf{A}\mathbf{R} = \mathbf{G}_n$). Then, for all $\sigma \geq m\|\mathbf{R}\| \log n$, and all target vectors $\mathbf{t} \in \mathbb{Z}_q^n$, the statistical distance between the following distributions is at most $\mathsf{negl}(n)$:*

$$\{\mathbf{x} \leftarrow \mathsf{SamplePre}(\mathbf{A}, \mathbf{R}, \mathbf{t}, \sigma)\} \quad \textit{and} \quad \{\mathbf{x} \leftarrow \mathbf{A}_\sigma^{-1}(\mathbf{t})\}.$$

*Learning with Errors and $\ell$-Succinct LWE.* The learning with errors (LWE) assumption [Reg05] with parameters $(n, m, q, \sigma)$ states that the distribution of $(\mathbf{A}, \mathbf{s}^\mathsf{T}\mathbf{A} + \mathbf{e}^\mathsf{T})$ is computationally indistinguishable from $(\mathbf{A}, \mathbf{v}^\mathsf{T})$ when $\mathbf{A} \xleftarrow{\text{R}} \mathbb{Z}_q^{n \times m}$, $\mathbf{s} \xleftarrow{\text{R}} \mathbb{Z}_q^n$, $\mathbf{e} \leftarrow D_{\mathbb{Z},\sigma}^m$, and $\mathbf{v} \xleftarrow{\text{R}} \mathbb{Z}_q^m$. Many recent works [ACL+22, WW23b, BCFL23, WW23a, CLM23, FMN23, Wee24] have introduced falsifiable variants of the LWE assumption (or the dual problem of short integer solutions (SIS)) which conjecture that the LWE (or SIS) problem with respect to $\mathbf{A}$ is hard even given a trapdoor for a matrix *related* to $\mathbf{A}$. In this work, we use the $\ell$-succinct LWE assumption introduced by Wee [Wee24], which asserts that LWE is hard with respect to $\mathbf{A}$ even given a trapdoor for the matrix $[\mathbf{I}_\ell \otimes \mathbf{A} \mid \mathbf{U}]$ where $\mathbf{U} \xleftarrow{\text{R}} \mathbb{Z}_q^{n\ell \times m}$. As discussed in [Wee24], the $\ell$-succinct LWE assumption is weaker than many of the other recently-proposed structured lattice assumptions (specifically, the LWE analogs of $k$-$R$-ISIS [ACL+22, BCFL23] and $\mathsf{BASIS}_{\mathsf{struct}}$ [WW23a, FMN23]). It is also implied by assumptions like the evasive LWE assumption [Wee22, Tsa22]. We now give the formal statement of the assumption:

**Assumption 2 ($\ell$-Succinct LWE [Wee24]).** Let $\lambda$ be a security parameter and let $n = n(\lambda), m = m(\lambda), q = q(\lambda), \sigma = \sigma(\lambda)$ be lattice parameters. Let $s = s(\lambda)$ be a Gaussian width parameter and $\ell = \ell(\lambda)$ be a dimension. We say that the $\ell$-succinct LWE assumption with parameters $(n, m, q, \sigma, s)$ holds if for all efficient adversaries $\mathcal{A}$, there exists a negligible function $\mathsf{negl}(\cdot)$ such that for all $\lambda \in \mathbb{N}$:

$$|\Pr[\mathcal{A}(\mathbf{A}, \mathbf{s}^\mathsf{T}\mathbf{A} + \mathbf{e}^\mathsf{T}, \mathbf{U}, \mathbf{T}) = 1] - \Pr[\mathcal{A}(\mathbf{A}, \mathbf{v}^\mathsf{T}, \mathbf{U}, \mathbf{T}) = 1]| = \mathsf{negl}(\lambda),$$

where $\mathbf{A} \xleftarrow{\text{R}} \mathbb{Z}_q^{n \times m}$, $\mathbf{s} \xleftarrow{\text{R}} \mathbb{Z}_q^n$, $\mathbf{e} \leftarrow D_{\mathbb{Z},\sigma}^m$, $\mathbf{v} \xleftarrow{\text{R}} \mathbb{Z}_q^m$, $\mathbf{U} \xleftarrow{\text{R}} \mathbb{Z}_q^{n\ell \times m}$, and $\mathbf{T} \leftarrow [\mathbf{I}_\ell \otimes \mathbf{A} \mid \mathbf{U}]_s^{-1}(\mathbf{G}_{n\ell})$.[4]

In other words, we require that LWE is hard with respect to $\mathbf{A}$ even given a fresh gadget trapdoor $\mathbf{T}$ for a related matrix $[\mathbf{I}_\ell \otimes \mathbf{A} \mid \mathbf{U}]$.

## 2.1 Distributed Broadcast Encryption

We now define the notion of distributed broadcast encryption.

**Definition 1 (Distributed Broadcast Encryption** [BZ14, KMW23]**).** *Let $\lambda$ be the security parameter and $N$ be the number of users. An $N$-user distributed broadcast encryption scheme is a tuple of efficient algorithms* $(\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{IsValid}, \mathsf{Enc}, \mathsf{Dec})$ *with the following syntax:*

- $\mathsf{Setup}(1^\lambda, 1^N) \to \mathsf{pp}$*: On input the security parameter $\lambda$ and the number of users $N$, the setup algorithm outputs the public parameters* $\mathsf{pp}$*.*
- $\mathsf{KeyGen}(\mathsf{pp}, i) \to (\mathsf{pk}_i, \mathsf{sk}_i)$*: On input the public parameters* $\mathsf{pp}$ *and an index $i \in [N]$, the key-generation algorithm outputs a public key and secret key* $(\mathsf{pk}_i, \mathsf{sk}_i)$*.*
- $\mathsf{IsValid}(\mathsf{pp}, i, \mathsf{pk}_i) \to b$*: On input the public parameters* $\mathsf{pp}$*, an index $i \in [N]$, and a public key* $\mathsf{pk}_i$*, the validity-checking algorithm outputs a bit $b \in \{0, 1\}$.*
- $\mathsf{Enc}(\mathsf{pp}, \{(i, \mathsf{pk}_i)\}_{i \in S}, \mu) \to \mathsf{ct}$*: On input the public parameters* $\mathsf{pp}$*, a collection of public keys* $\mathsf{pk}_i$ *and a message $\mu \in \{0, 1\}$, the encryption algorithm outputs a ciphertext* $\mathsf{ct}$*.*
- $\mathsf{Dec}(\mathsf{pp}, \{(i, \mathsf{pk}_i)\}_{i \in S}, \mathsf{ct}, (j, \mathsf{sk}_j)) \to \mu$*: On input the public parameters* $\mathsf{pp}$*, a collection of public keys* $\mathsf{pk}_i$*, a ciphertext* $\mathsf{ct}$*, and a secret key* $\mathsf{sk}_j$ *for an index $j$, the decryption algorithm outputs a message $\mu \in \{0, 1\}$.*

*We require that* $(\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{IsValid}, \mathsf{Enc}, \mathsf{Dec})$ *satisfy the following properties:*

- ***Correctness:*** *For a security parameter $\lambda \in \mathbb{N}$, a bound $N$ on the number of users, and an adversary $\mathcal{A}$, we define the correctness experiment as follows:*
  - *The challenger samples* $\mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda, 1^N)$ *and gives* $\mathsf{pp}$ *to $\mathcal{A}$.*
  - *The adversary specifies a target index $j \in [N]$. The challenger responds by computing* $(\mathsf{pk}_j, \mathsf{sk}_j) \leftarrow \mathsf{KeyGen}(\mathsf{pp}, j)$*. It gives* $\mathsf{pk}_j$ *to the adversary $\mathcal{A}$.*
  - *The adversary outputs a set $S \subseteq [N]$, a collection of public keys* $\mathsf{pk}_i$ *for $i \in S \setminus \{j\}$, and a message $\mu \in \{0, 1\}$.*
  - *The challenger checks that $j \in S$ and that* $\mathsf{IsValid}(\mathsf{pp}, i, \mathsf{pk}_i) = 1$ *for each $i \in S \setminus \{j\}$ and outputs $b = 1$ if not. Otherwise, the challenger computes* $\mathsf{ct} \leftarrow \mathsf{Enc}(\mathsf{pp}, \{(i, \mathsf{pk}_i)\}_{i \in S}, \mu)$ *and* $\mu' \leftarrow \mathsf{Dec}(\mathsf{pp}, \{(i, \mathsf{pk}_i)\}_{i \in S}, \mathsf{ct}, (j, \mathsf{sk}_j))$*. It outputs $b = 1$ if $\mu = \mu'$ and $b = 0$ otherwise.*

  *We say that the scheme is correct if for all $\lambda, N \in \mathbb{N}$ and all adversaries $\mathcal{A}$, there exists a negligible function* $\mathsf{negl}(\cdot)$ *such that for all $\lambda \in \mathbb{N}$, $\Pr[b = 1] \geq 1 - \mathsf{negl}(\lambda)$ in the correctness experiment.*

[4] Note that this distribution is only well defined when $\mathbf{G}_{n\ell}$ is in the image of $[\mathbf{I}_\ell \otimes \mathbf{A} \mid \mathbf{U}]$. Thus, when $\mathbf{G}_{n\ell}$ is not in the image, we set $\mathbf{T} = \perp$. Accordingly, taking $m \geq 2n \log q$ ensures that this event occurs with negligible probability (Corollary 1).

– ***Verifiable keys:*** *For all $\lambda, N \in \mathbb{N}$, and all indices $i \in [N]$, it holds that*

$$\Pr\left[\mathsf{IsValid}(\mathsf{pp}, i, \mathsf{pk}_i) = 1 : \begin{array}{c} \mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda, 1^N) \\ (\mathsf{pk}_i, \mathsf{sk}_i) \leftarrow \mathsf{KeyGen}(\mathsf{pp}, i) \end{array}\right] \geq 1 - \mathsf{negl}(\lambda).$$

– ***Selective security:*** *For a security parameter $\lambda$, a bound $N$ on the number of users, and a bit $b \in \{0,1\}$, we define the selective security game between an adversary $\mathcal{A}$ and a challenger as follows:*
  - *On input the security parameter $1^\lambda$ and the number of users $1^N$, the adversary outputs a challenge set $S^* \subseteq [N]$.*
  - *The challenger samples $\mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda, 1^N)$ and $(\mathsf{pk}_i, \mathsf{sk}_i) \leftarrow \mathsf{KeyGen}(\mathsf{pp}, i)$ for $i \in S^*$. It also computes $\mathsf{ct}_b \leftarrow \mathsf{Enc}(\mathsf{pp}, \{\mathsf{pk}_i\}_{i \in S^*}, b, S^*)$ and sends $\big(\mathsf{pp}, \{\mathsf{pk}_i\}_{i \in S^*}, \mathsf{ct}_b\big)$ to $\mathcal{A}$.*
  - *At the end of the game, algorithm $\mathcal{A}$ outputs $b' \in \{0,1\}$, which is the output of the experiment.*

  *We say the distributed broadcast encryption scheme is selectively secure if for all polynomials $N = N(\lambda)$, and all efficient adversaries $\mathcal{A}$, there exists a negligible function $\mathsf{negl}(\cdot)$ such that for all $\lambda \in \mathbb{N}$,*

$$|\Pr[b' = 1 \mid b = 1] - \Pr[b' = 1 \mid b = 0]| = \mathsf{negl}(\lambda) \tag{2.2}$$

  *in the selective security game. We say that the scheme is selectively secure for up to $N$ users if Eq. (2.2) holds for the specific value of $N$.*

– ***Short ciphertexts:*** *There exists a fixed polynomial $\mathsf{poly}(\cdot)$ such that for all $\lambda, N \in \mathbb{N}$, all subsets $S \subseteq [N]$, all public parameters $\mathsf{pp}$ in the support of $\mathsf{Setup}(1^\lambda, 1^N)$, all key-pairs $(\mathsf{pk}_i, \mathsf{sk}_i)$ in the support of $\mathsf{KeyGen}(\mathsf{pp}, i)$ for $i \in S$, all messages $\mu \in \{0,1\}$, and all ciphertexts $\mathsf{ct}$ in the support of $\mathsf{Enc}(\mathsf{pp}, \{\mathsf{pk}_i\}_{i \in S}, \mu, S)$, it holds that $|\mathsf{ct}| \leq \mathsf{poly}(\lambda + \log N)$.*

*Remark 1 (Encrypting Long Messages).* Definition 1 considers the (simple) setting where the ciphertext encrypts a single bit. It is straightforward to support encrypting longer messages by composing with a symmetric encryption scheme. Namely, to encrypt a message $\mu \in \{0,1\}^m$, the encryption algorithm samples a symmetric key $k \in \{0,1\}^{\mathsf{poly}(\lambda)}$, encrypts the bits of $k$ using the broadcast encryption scheme, and then encrypts $\mu$ using the symmetric key $k$. The size of the overall ciphertext is then $|\mu| + \mathsf{poly}(\lambda, \log N)$.

## 3 Distributed Broadcast Encryption from Lattices

In this section, we give our construction of a selectively-secure distributed broadcast encryption scheme. We begin by introducing an intermediate assumption called the $\ell$-structured LWE assumption which we will use in our security analysis. Then, in Sect. 4, we show that our intermediate assumption follows from the $\ell'$-succinct LWE assumption (Definition 2) for $\ell' \geq \ell \cdot O(n \log q)$, where $n, q$ are lattice parameters.

**Assumption 3 ($\ell$-Structured LWE).** Let $\lambda$ be a security parameter and $n = n(\lambda), m = m(\lambda), q = q(\lambda), \sigma = \sigma(\lambda)$ be lattice parameters. Let $s = s(\lambda)$ be a Gaussian width parameter. Let $k = k(\lambda)$ and $\ell = \ell(\lambda)$ be dimension parameters. We say that the $\ell$-structured LWE assumption with parameters $(n, m, q, \sigma, s, k)$ holds if for all efficient adversaries $\mathcal{A}$, there exists a negligible function $\mathsf{negl}(\cdot)$ such that for all $\lambda \in \mathbb{N}$:

$$|\Pr[\mathcal{A}(\mathbf{A}, \mathbf{s}^{\mathsf{T}}\mathbf{A} + \mathbf{e}^{\mathsf{T}}, \mathbf{Z}, \mathbf{R}, \mathbf{T}) = 1] - \Pr[\mathcal{A}(\mathbf{A}, \mathbf{u}^{\mathsf{T}}, \mathbf{Z}, \mathbf{R}, \mathbf{T}) = 1]| = \mathsf{negl}(\lambda),$$

where $\mathbf{A} \xleftarrow{\mathrm{R}} \mathbb{Z}_q^{n \times m}$, $\mathbf{s} \xleftarrow{\mathrm{R}} \mathbb{Z}_q^n$, $\mathbf{e} \leftarrow D_{\mathbb{Z},\sigma}^m$, $\mathbf{u} \xleftarrow{\mathrm{R}} \mathbb{Z}_q^m$, $\mathbf{Z} = [\mathbf{Z}_1 \mid \cdots \mid \mathbf{Z}_k] \xleftarrow{\mathrm{R}} \mathbb{Z}_q^{n \times mk}$, $\mathbf{R} = [\mathbf{r}_1 \mid \cdots \mid \mathbf{r}_\ell] \leftarrow D_{\mathbb{Z},s}^{m \times \ell}$, $\mathbf{T} \leftarrow (\mathbf{V}_{\ell,k})_s^{-1}(\mathbf{G}_{n\ell})$, and

$$\mathbf{V}_{\ell,k} = \left[\begin{array}{ccc|ccc} \mathbf{A} & & & -\mathbf{Z}_1\mathbf{r}_1 & \cdots & -\mathbf{Z}_k\mathbf{r}_1 \\ & \ddots & & \vdots & \ddots & \vdots \\ & & \mathbf{A} & -\mathbf{Z}_1\mathbf{r}_\ell & \cdots & -\mathbf{Z}_k\mathbf{r}_\ell \end{array}\right] = \left[\begin{array}{ccc|c} \mathbf{A} & & & -\mathbf{Z}(\mathbf{I}_k \otimes \mathbf{r}_1) \\ & \ddots & & \vdots \\ & & \mathbf{A} & -\mathbf{Z}(\mathbf{I}_k \otimes \mathbf{r}_\ell) \end{array}\right] \in \mathbb{Z}_q^{n\ell \times (m\ell + k)}. \tag{3.1}$$

Similar to $\ell$-succinct LWE, we require that LWE is hard with respect to $\mathbf{A}$ even given a trapdoor $\mathbf{T}$ for a related matrix $\mathbf{V}_{\ell,k}$. While the right side of the matrix $\mathbf{V}_{\ell,k}$ appears significantly more structured than the random matrix $\mathbf{U}$ in the $\ell$-succinct LWE assumption (Definition 2), we show in Sect. 4 that this assumption is implied by the $\ell'$-succinct LWE assumption (when $\ell' \geq \ell \cdot O(n \log q)$ and $k \geq 3nm \log q$).

*Parameter Setting.* Similar to $\ell$-succinct LWE, we only consider instantiations with $m \geq O(n \log q)$ so that the matrix $\mathbf{A}$ spans $\mathbb{Z}_q^n$ with overwhelming probability and the $\mathbf{Z}_i$ matrices have sufficient width. We additionally note that the $\ell$-structured LWE assumption is false when $k$ is too small. In this setting, the adversary can use the trapdoor to repeatedly sample $(\mathbf{V}_{\ell,k})_s^{-1}(\mathbf{0})$. By Lemma 4, these preimages include samples from $\mathbf{A}_s^{-1}(\mathbf{Z}(\mathbf{d} \otimes \mathbf{I}_m)\mathbf{r}_i)$ where $\mathbf{d} \leftarrow D_{\mathbb{Z},s}^k$. When $k$ is too small, collisions in the value of $\mathbf{d}$ will arise with noticeable probability. Such a collision immediately gives a short vector $\mathbf{x}$ such that $\mathbf{A}\mathbf{x} = \mathbf{0}$ (which immediately breaks LWE with respect to $\mathbf{A}$). Thus, we require $k = \omega(\log n)$ to ensure that collisions are unlikely to occur. In our setting, we will only consider $k \geq O(nm \log q)$. In this case, the marginal distribution of $\mathbf{Z}(\mathbf{d} \otimes \mathbf{I}_m)$ is statistically close to uniform by Lemma 3. For this parameter reigme, we can in fact show that the $\ell$-structured LWE assumptions holds under the $\ell'$-succinct LWE for $\ell' \geq \ell \cdot O(n \log q)$. We provide this reduction in Sect. 4.

## 3.1 Distributed Broadcast Encryption from $\ell$-Structured LWE

In this section, we describe our distributed broadcast encryption scheme from $\ell$-structured LWE, where $\ell = N$ is the bound on the number of users in the system.

**Construction 4 (Distributed Broadcast Encryption).** Let $\lambda \in \mathbb{N}$ be a security parameter, $N \in \mathbb{N}$ be the number of users, and $n = n(\lambda, N), m = m(\lambda, N), q = q(\lambda, N), \sigma = \sigma(\lambda, N)$ be lattice parameters. Let $s_0 = s_0(\lambda, N), s_1 = s_1(\lambda, N)$ be Gaussian width parameters, $k = k(\lambda, N)$ be a dimension, and $\beta = \beta(\lambda, N)$ be a norm bound. We construct our distributed broadcast encryption scheme $(\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{IsValid}, \mathsf{Enc}, \mathsf{Dec})$ as follows:

- $\mathsf{Setup}(1^\lambda, 1^N)$: On input the security parameter $\lambda$ and the bound on the number of users $N$, the setup algorithm proceeds as follows:
  1. Sample $(\mathbf{A}, \mathbf{T_A}) \leftarrow \mathsf{TrapGen}(1^n, q, m)$, $\mathbf{B} \xleftarrow{\text{R}} \mathbb{Z}_q^{n \times m}$, $\mathbf{p} \xleftarrow{\text{R}} \mathbb{Z}_q^n$.
  2. For each $i \in [k]$, sample $\mathbf{Z}_i \xleftarrow{\text{R}} \mathbb{Z}_q^{n \times m}$ and let $\mathbf{Z} = [\mathbf{Z}_1 \mid \cdots \mid \mathbf{Z}_k] \in \mathbb{Z}_q^{n \times mk}$. For each $i \in [N]$, sample $\mathbf{r}_i \leftarrow D_{\mathbb{Z}, s_0}^m$.
  3. Sample $\mathbf{T_V} \leftarrow \mathsf{SamplePre}\big(\mathbf{V}_{N,k}, \left[\begin{smallmatrix} \mathbf{I}_N \otimes \mathbf{T_A} \\ \mathbf{0} \end{smallmatrix}\right], \mathbf{G}_{nN}, s_0\big)$, where

$$\mathbf{V}_{N,k} = \left[\begin{array}{ccc|c} \mathbf{A} & & & -\mathbf{Z}(\mathbf{I}_k \otimes \mathbf{r}_1) \\ & \ddots & & \vdots \\ & & \mathbf{A} & -\mathbf{Z}(\mathbf{I}_k \otimes \mathbf{r}_N) \end{array}\right] \in \mathbb{Z}_q^{nN \times (mN+k)}. \tag{3.2}$$

  Output $\mathsf{pp} = (\mathbf{A}, \mathbf{B}, \mathbf{p}, \mathbf{Z}, \{\mathbf{r}_i\}_{i \in [N]}, \mathbf{T_V})$.
- $\mathsf{KeyGen}(\mathsf{pp}, i)$: On input the public parameters $\mathsf{pp} = (\mathbf{A}, \mathbf{B}, \mathbf{p}, \mathbf{Z}, \{\mathbf{r}_i\}_{i \in [N]}, \mathbf{T_V})$ and an index $i \in [N]$, the key-generation algorithm samples

$$\begin{bmatrix} \mathbf{y}_1 \\ \vdots \\ \mathbf{y}_N \\ \mathbf{d} \end{bmatrix} \leftarrow \mathsf{SamplePre}(\mathbf{V}_{N,k}, \mathbf{T_V}, \mathbf{u}_i \otimes (\mathbf{p} + \mathbf{B}\mathbf{r}_i), s_1), \tag{3.3}$$

  where $\mathbf{u}_i \in \{0,1\}^N$ is the the $i^{\text{th}}$ standard basis vector, $\mathbf{y}_i \in \mathbb{Z}^m$ for each $i \in [N]$, and $\mathbf{d} \in \mathbb{Z}^k$. It sets $\mathbf{W} = \mathbf{Z}(\mathbf{d} \otimes \mathbf{I}_m) \in \mathbb{Z}_q^{n \times m}$ and outputs the public key $\mathsf{pk} = (\mathbf{W}, \{\mathbf{y}_j\}_{j \neq i})$ and the secret key $\mathsf{sk} = \mathbf{y}_i$.
- $\mathsf{IsValid}(\mathsf{pp}, i, \mathsf{pk}_i)$: On input the parameters $\mathsf{pp} = (\mathbf{A}, \mathbf{B}, \mathbf{p}, \mathbf{Z}, \{\mathbf{r}_i\}_{i \in [N]}, \mathbf{T_V})$, an index $i \in [N]$, and a public key $\mathsf{pk}_i = (\mathbf{W}_i, \{\mathbf{y}_{i,j}\}_{j \neq i})$, the validity-checking algorithm outputs 1 if the following holds:

$$\forall j \neq i : \mathbf{A}\mathbf{y}_{i,j} = \mathbf{W}_i \mathbf{r}_j \quad \text{and} \quad \|\mathbf{y}_{i,j}\| \leq \beta.$$

  Otherwise, the algorithm outputs 0.
- $\mathsf{Enc}(\mathsf{pp}, \{(j, \mathsf{pk}_j)\}_{j \in S}, \mu)$: On input $\mathsf{pp} = (\mathbf{A}, \mathbf{B}, \mathbf{p}, \mathbf{Z}, \{\mathbf{r}_i\}_{i \in [N]}, \mathbf{T_V})$, a collection of public keys $\mathsf{pk}_j = (\mathbf{W}_j, \{\mathbf{y}_{j,j'}\}_{j' \neq j})$ for each $j \in S$, and a message $\mu \in \{0,1\}$, the encryption algorithm samples $\mathbf{s} \xleftarrow{\text{R}} \mathbb{Z}_q^n$, $\mathbf{e} \leftarrow D_{\mathbb{Z},\sigma}^m$, $\mathbf{H} \xleftarrow{\text{R}} \{0,1\}^{m \times m}$, and $\mathbf{h} \xleftarrow{\text{R}} \{0,1\}^m$. It computes $\mathbf{W}_S = \sum_{j \in S} \mathbf{W}_j$ and outputs

$$\mathsf{ct} = \big(\mathbf{s}^\mathsf{T}\mathbf{A} + \mathbf{e}^\mathsf{T}\ ,\ \mathbf{s}^\mathsf{T}(\mathbf{B} + \mathbf{W}_S) + \mathbf{e}^\mathsf{T}\mathbf{H}\ ,\ \mathbf{s}^\mathsf{T}\mathbf{p} + \mathbf{e}^\mathsf{T}\mathbf{h} + \mu \cdot \lfloor q/2 \rfloor\big).$$

- $\mathsf{Dec}(\mathsf{pp}, \{(j, \mathsf{pk}_j)\}_{j \in S}, \mathsf{ct}, (i, \mathsf{sk}_i))$: On input $\mathsf{pp} = (\mathbf{A}, \mathbf{B}, \mathbf{p}, \mathbf{Z}, \{\mathbf{r}_i\}_{i \in [N]}, \mathbf{T}_{\mathbf{V}})$, a collection of public keys $\mathsf{pk}_j = (\mathbf{W}_j, \{\mathbf{y}_{j,j'}\}_{j' \neq j})$ for each $j \in S$, a ciphertext $\mathsf{ct} = (\mathbf{c}_1^\intercal, \mathbf{c}_2^\intercal, c_3)$, and a secret key $\mathsf{sk}_i = \mathbf{y}_{i,i} \in \mathbb{Z}_q^m$ for an index $i$, the decryption algorithm computes

$$z = c_3 + \mathbf{c}_2^\intercal \mathbf{r}_i - \mathbf{c}_1^\intercal \left( \mathbf{y}_{i,i} + \sum_{j \in S \setminus \{i\}} \mathbf{y}_{j,i} \right) \in \mathbb{Z}_q,$$

and outputs $\lfloor z \rceil$ where $\lfloor z \rceil$ outputs 0 if $-q/4 \leq z < q/4$ and 1 otherwise.

**Theorem 5 (Verifiable Keys).** *Suppose $q$ is prime, $n \geq \lambda$, $m \geq 2n \log q$, $s_0 \geq (mN + k) \log(nN)$, $s_1 \geq (mN + k)\sqrt{m} s_0 \log(nN)$, and $\beta \geq \sqrt{m} s_1$. Then, Construction 4 has verifiable keys.*

*Proof.* Let $\lambda, N \in \mathbb{N}$ and $i \in [N]$. Let $\mathsf{pp} = (\mathbf{A}, \mathbf{B}, \mathbf{p}, \mathbf{Z}, \{\mathbf{r}_i\}_{i \in [N]}, \mathbf{T}_{\mathbf{V}}) \leftarrow \mathsf{Setup}(1^\lambda, 1^N)$, and sample $(\mathsf{pk}_i, \mathsf{sk}_i) \leftarrow \mathsf{KeyGen}(\mathsf{pp}, i)$. Then, we can write

$$\mathsf{pk}_i = \big(\mathbf{W}_i, \{\mathbf{y}_{i,j}\}_{j \neq i}\big) \quad \text{and} \quad \mathsf{sk}_i = \mathbf{y}_{i,i}.$$

We now show that $\mathsf{IsValid}(\mathsf{pp}, i, \mathsf{pk}_i) = 1$ with overwhelming probability:

- Since $s_0 \geq (mN + k) \log(nN)$, by Lemma 5, the distribution of $\mathbf{T}_{\mathbf{V}}$ is statistically close to $(\mathbf{V}_{N,k})_{s_0}^{-1}(\mathbf{G}_{nN})$. Since $m \geq 2n \log q$ and $q$ is prime, by Lemmas 4 and 2, we have that $\|\mathbf{T}_{\mathbf{V}}\| \leq \sqrt{m} s_0$ with overwhelming probability.
- Since $s_1 \geq (mN + k)\sqrt{m} s_0 \log(nN)$, by Lemma 5, the distribution of $\mathbf{y}_{i,1}, \ldots, \mathbf{y}_{i,N}, \mathbf{d}_i$ output by Eq. (1.2) is statistically close to sampling from $(\mathbf{V}_{N,k})_{s_1}^{-1}(\mathbf{u}_i \otimes (\mathbf{p} + \mathbf{B}\mathbf{r}_i))$. By construction of $\mathbf{V}_{N,k}$ (see Eq. (3.2)) and using Eq. (2.1), this means that for all $j \neq i$

$$\mathbf{0} = \mathbf{A}\mathbf{y}_{i,j} - \mathbf{Z}(\mathbf{I}_k \otimes \mathbf{r}_j)\mathbf{d}_i = \mathbf{A}\mathbf{y}_{i,j} - \mathbf{Z}(\mathbf{I}_k \otimes \mathbf{r}_j)(\mathbf{d}_i \otimes 1) = \mathbf{A}\mathbf{y}_{i,j} - \mathbf{Z}(\mathbf{d}_i \otimes \mathbf{I}_m)\mathbf{r}_j.$$

  By definition of $\mathsf{KeyGen}$, it sets $\mathbf{W}_i = \mathbf{Z}(\mathbf{d}_i \otimes \mathbf{I}_m)$. Correspondingly, this means that

$$\mathbf{A}\mathbf{y}_{i,j} = \mathbf{Z}(\mathbf{d}_i \otimes \mathbf{I}_m)\mathbf{r}_j = \mathbf{W}_i \mathbf{r}_j.$$

- By Lemmas 2 and 4, $\|\mathbf{y}_{i,j}\| \leq \sqrt{m} s_1 \leq \beta$ with overwhelming probability.

Thus, $\mathsf{IsValid}(\mathsf{pp}, i, \mathsf{pk}_i) = 1$ holds with overwhelming probability.

**Theorem 6 (Correctness).** *Suppose the modulus $q$ is prime, $m \geq 2n \log q$, $s_0 \geq (mN + k) \log(nN)$, $s_1 \geq (mN + k)\sqrt{m} s_0 \log(nN)$, $\beta \geq \sqrt{m} s_1$, and $q \geq 4\sqrt{nm}\sigma(1 + N\beta + \sqrt{nm} s_0)$. Then, Construction 4 satisfies correctness.*

*Proof.* Let $\mathsf{pp} = (\mathbf{A}, \mathbf{B}, \mathbf{p}, \mathbf{Z}, \{\mathbf{r}_i\}_{i \in [N]}, \mathbf{T}_{\mathbf{V}}) \leftarrow \mathsf{Setup}(1^\lambda, 1^N)$. Take any index $i \in [N]$, and let $(\mathsf{pk}_i, \mathsf{sk}_i) \leftarrow \mathsf{KeyGen}(\mathsf{msk}, i)$. Write $\mathsf{pk}_i = \big(\mathbf{W}_i, \{\mathbf{y}_{i,j}\}_{j \neq i}\big)$ and $\mathsf{sk}_i = \mathbf{y}_{i,i}$. By the same analysis as in the proof of Theorem 5, we have $\|\mathbf{y}_{i,i}\| \leq \beta$ and $\mathbf{A}\mathbf{y}_{i,i} - \mathbf{Z}(\mathbf{d}_i \otimes \mathbf{I}_m)\mathbf{r}_i = \mathbf{p} + \mathbf{B}\mathbf{r}_i$. Since $\mathbf{W}_i = \mathbf{Z}(\mathbf{d}_i \otimes \mathbf{I}_m)$, this means that

$$\mathbf{A}\mathbf{y}_{i,i} = \mathbf{p} + \mathbf{B}\mathbf{r}_i + \mathbf{W}_i \mathbf{r}_i \tag{3.4}$$

Take any set $S \subseteq [N]$ and any collection of public keys $\{\mathsf{pk}_j\}_{j \in S \setminus \{i\}}$ where $\mathsf{pk}_j$ satisfies $\mathsf{IsValid}(\mathsf{pp}, i, \mathsf{pk}_i) = 1$. This means that for all $j \in S \setminus \{i\}$,

$$\mathbf{A}\mathbf{y}_{j,i} = \mathbf{W}_j \mathbf{r}_i \quad \text{and} \quad \|\mathbf{y}_{j,i}\| \leq \beta. \tag{3.5}$$

Take any message $\mu \in \{0,1\}$ and let $\mathsf{ct} = (\mathbf{c}_1^\intercal, \mathbf{c}_2^\intercal, c_3) \leftarrow \mathsf{Enc}(\mathsf{pp}, \{\mathsf{pk}_i\}_{i \in S}, \mu, S)$. Let $\mathbf{s} \in \mathbb{Z}_q^n, \mathbf{e} \in \mathbb{Z}_q^m, \mathbf{H} \in \{0,1\}^{m \times m}, \mathbf{h} \in \{0,1\}^m$ be the components sampled by encryption. Consider the output of the algorithm $\mathsf{Dec}(\mathsf{pp}, \{(i, \mathsf{pk}_i)\}_{i \in S}, \mathsf{ct}, (j, \mathsf{sk}_j))$. First,

$$\mathbf{c}_1^\intercal \left( \mathbf{y}_{i,i} + \sum_{j \in S \setminus \{i\}} \mathbf{y}_{j,i} \right) = \mathbf{s}^\intercal \mathbf{A} \mathbf{y}_{i,i} + \sum_{j \in S \setminus \{i\}} \mathbf{s}^\intercal \mathbf{A} \mathbf{y}_{j,i} + \underbrace{\mathbf{e}^\intercal \mathbf{y}_{i,i} + \sum_{j \in S \setminus \{i\}} \mathbf{e}^\intercal \mathbf{y}_{j,i}}_{\tilde{e}_1}.$$

Combined with Eqs. 3.4 and 3.5, this becomes

$$\begin{aligned} \mathbf{c}_1^\intercal \left( \mathbf{y}_{i,i} + \sum_{j \in S \setminus \{i\}} \mathbf{y}_{j,i} \right) &= \mathbf{s}^\intercal (\mathbf{p} + \mathbf{B}\mathbf{r}_i + \mathbf{W}_i \mathbf{r}_i) + \sum_{j \in S \setminus \{i\}} \mathbf{s}^\intercal \mathbf{W}_j \mathbf{r}_i + \tilde{e}_1 \\ &= \mathbf{s}^\intercal (\mathbf{p} + \mathbf{B}\mathbf{r}_i + \mathbf{W}_S \mathbf{r}_i) + \tilde{e}_1, \end{aligned}$$

using the fact that $\mathbf{W}_S = \sum_{j \in S} \mathbf{W}_j$ and $i \in S$. Next,

$$c_3 + \mathbf{c}_2^\intercal \mathbf{r}_i = \mu \cdot \lfloor q/2 \rfloor + \mathbf{s}^\intercal \mathbf{p} + \mathbf{s}^\intercal (\mathbf{B} + \mathbf{W}_S) \mathbf{r}_i + \underbrace{\mathbf{e}^\intercal \mathbf{h} + \mathbf{e}^\intercal \mathbf{H} \mathbf{r}_i}_{\tilde{e}_2}.$$

Putting everything together, we have

$$c_3 + \mathbf{c}_2^\intercal \mathbf{r}_i - \mathbf{c}_1^\intercal \left( \mathbf{y}_{i,i} + \sum_{j \in S \setminus \{i\}} \mathbf{y}_{j,i} \right) = \mu \cdot \lfloor q/2 \rfloor - \tilde{e}_1 + \tilde{e}_2.$$

It suffices to show that $|\tilde{e}_1 - \tilde{e}_2| < q/4$. We show this holds with overwhelming probability:

- Since $\mathsf{Enc}$ samples $\mathbf{e} \leftarrow D_{\mathbb{Z},\sigma}^m$, by Lemma 2, with overwhelming probability, $\|\mathbf{e}\| \leq \sqrt{n}\sigma$.
- Since $\|\mathbf{y}_{j,i}\| \leq \beta$ for all $j \in S$, it follows that $|\mathbf{e}^\intercal \mathbf{y}_{j,i}| \leq \sqrt{n} m \beta \sigma$. Thus,

$$|\tilde{e}_1| \leq \sum_{j \in S} |\mathbf{e}^\intercal \mathbf{y}_{j,i}| \leq N \sqrt{n} m \beta \sigma.$$

- Next, $\mathbf{H} \in \{0,1\}^{m \times m}$ and $\mathbf{h} \in \{0,1\}^m$ so $|\mathbf{e}^\intercal \mathbf{h}| \leq \sqrt{n} m \sigma$ and $\|\mathbf{e}^\intercal \mathbf{H}\| \leq \sqrt{n} m \sigma$. Since $\mathbf{r}_i \leftarrow D_{\mathbb{Z},s_0}^m$, by Lemma 2, with overwhelming probability $\|\mathbf{r}_i\| \leq \sqrt{n} s_0$. Then, $|\mathbf{e}^\intercal \mathbf{H} \mathbf{r}_i| \leq n m^2 \sigma s_0$. Thus,

$$|\tilde{e}_2| \leq |\mathbf{e}^\intercal \mathbf{h}| + |\mathbf{e}^\intercal \mathbf{H} \mathbf{r}_i| \leq \sqrt{n} m \sigma (1 + \sqrt{n} m s_0).$$

Correctness holds as long as

$$q \geq 4|\tilde{e}_1 - \tilde{e}_2| \geq 4\sqrt{n} m\sigma(1 + N\beta + \sqrt{n} m s_0).$$

**Theorem 7 (Selective Security).** *Let $\lambda$ be a security parameter and $N = N(\lambda)$ be any polynomial function. Suppose $n \geq \lambda$, $m \geq 3n \log q$, $s_0 \geq (mN + k)\log(nN)$ and $s_1 \geq (mN + k)\sqrt{m} s_0 \log(nN)$. Then, under the $N$-structured LWE assumption (Assumption 3) with parameters $(n, m, q, \sigma, s_0, k)$, Construction 4 is selectively-secure for up to $N$ users.*

*Proof.* Take any polynomial $N = N(\lambda)$ and any efficient adversary $\mathcal{A}$ for the selective security game. We start by defining a sequence of hybrid experiments:

- $\mathsf{Hyb}_0^{(b)}$: This is the selective security game with challenge bit $b \in \{0,1\}$. At the beginning of the game, the adversary $\mathcal{A}$ declares the set $S^* \subseteq [N]$. The challenger then samples $\mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda, 1^N)$, $(\mathsf{pk}_i, \mathsf{sk}_i) \leftarrow \mathsf{KeyGen}(\mathsf{pp}, i)$ for each $i \in S^*$, and $\mathsf{ct}_b \leftarrow \mathsf{Enc}(\mathsf{pp}, \{\mathsf{pk}_i\}_{i \in S^*}, b, S^*)$. The challenger gives $(\mathsf{pp}, \{\mathsf{pk}_i\}_{i\in S^*}, \mathsf{ct}_b)$ to the adversary $\mathcal{A}$. To recall, the challenger samples the elements as follows:
  - The challenger starts by sampling $(\mathbf{A}, \mathbf{T}_\mathbf{A}) \leftarrow \mathsf{TrapGen}(1^n, q, m)$, $\mathbf{B} \xleftarrow{\text{R}} \mathbb{Z}_q^{n\times m}$, $\mathbf{p} \xleftarrow{\text{R}} \mathbb{Z}_q^n$, $\mathbf{Z}_1, \ldots, \mathbf{Z}_k \xleftarrow{\text{R}} \mathbb{Z}_q^{n\times m}$, and $\mathbf{r}_1, \ldots, \mathbf{r}_N \leftarrow D_{\mathbb{Z}, s_0}^m$. It sets $\mathbf{Z} = [\mathbf{Z}_1 \mid \cdots \mid \mathbf{Z}_k] \in \mathbb{Z}_q^{n \times mk}$ and $\mathbf{V}_{N,k}$ as in Eq. (3.2).
  - Next, it samples a trapdoor $\mathbf{T}_\mathbf{V} \leftarrow \mathsf{SamplePre}(\mathbf{V}_{N,k}, \left[\begin{smallmatrix}\mathbf{I}_N \otimes \mathbf{T}_\mathbf{A} \\ \mathbf{0}\end{smallmatrix}\right], \mathbf{G}_{nN}, s_0)$. The challenger sets the public parameters to be

$$\mathsf{pp} = (\mathbf{A}, \mathbf{B}, \mathbf{p}, \mathbf{Z}, \{\mathbf{r}_i\}_{i\in[N]}, \mathbf{T}_\mathbf{V}).$$

  - To generate the public key for $i \in S^*$, the challenger samples

$$\boldsymbol{\kappa}_i \leftarrow \mathsf{SamplePre}(\mathbf{V}_{N,k}, \mathbf{T}_\mathbf{V}, \mathbf{u}_i \otimes (\mathbf{p} + \mathbf{B}\mathbf{r}_i), s_1),$$

  and then parses

$$\boldsymbol{\kappa}_i = \begin{bmatrix} \mathbf{y}_{i,1} \\ \vdots \\ \mathbf{y}_{i,N} \\ \mathbf{d}_i \end{bmatrix} \in \mathbb{Z}_q^{Nm+k}, \tag{3.6}$$

  where $\mathbf{y}_{i,j} \in \mathbb{Z}_q^m$ and $\mathbf{d}_i \in \mathbb{Z}_q^k$. It sets $\mathbf{W}_i = \mathbf{Z}(\mathbf{d}_i \otimes \mathbf{I}_m)$ and $\mathsf{pk}_i = \big(\mathbf{W}_i, \{\mathbf{y}_{i,j}\}_{j\neq i}\big)$.
  - Finally, to generate the challenge ciphertext, the challenger samples $\mathbf{s} \xleftarrow{\text{R}} \mathbb{Z}_q^n$, $\mathbf{e} \leftarrow D_{\mathbb{Z},\sigma}^m$, $\mathbf{H} \xleftarrow{\text{R}} \{0,1\}^{m\times m}$, and $\mathbf{h} \xleftarrow{\text{R}} \{0,1\}^m$. It sets $\mathbf{W}_{S^*} = \sum_{j\in S^*} \mathbf{W}_j$ and constructs the challenge ciphertext as

$$\mathsf{ct}_b = (\mathbf{c}_1^\mathsf{T}, \mathbf{c}_2^\mathsf{T}, c_3) = \big(\mathbf{s}^\mathsf{T}\mathbf{A} + \mathbf{e}^\mathsf{T}\ ,\ \mathbf{s}^\mathsf{T}(\mathbf{B} + \mathbf{W}_{S^*}) + \mathbf{e}^\mathsf{T}\mathbf{H}\ ,\ \mathbf{s}^\mathsf{T}\mathbf{p} + \mathbf{e}^\mathsf{T}\mathbf{h} + b\cdot \lfloor q/2 \rfloor\big).$$

  At the end of the experiment, algorithm $\mathcal{A}$ outputs a bit $b' \in \{0,1\}$, which is the output of the experiment.
- $\mathsf{Hyb}_1^{(b)}$: Same as $\mathsf{Hyb}_0^{(b)}$, except the challenger samples $\mathbf{T}_\mathbf{V} \leftarrow (\mathbf{V}_{N,k})_{s_0}^{-1}(\mathbf{G}_{nN})$.

- $\mathsf{Hyb}_2^{(b)}$: Same as $\mathsf{Hyb}_1^{(b)}$, except for all $i \in S^*$, the challenger samples the $\boldsymbol{\kappa}_i$ component as $\boldsymbol{\kappa}_i \leftarrow (\mathbf{V}_{N,k})_{s_1}^{-1}(\mathbf{u}_i \otimes (\mathbf{p} + \mathbf{B}\mathbf{r}_i))$.
- $\mathsf{Hyb}_3^{(b)}$: Same as $\mathsf{Hyb}_2^{(b)}$, except the challenger samples $\mathbf{A} \xleftarrow{\text{R}} \mathbb{Z}_q^{n \times m}$.
- $\mathsf{Hyb}_4^{(b)}$: Same as $\mathsf{Hyb}_3^{(b)}$, except for all $i \in S^*$, the challenger first samples $\mathbf{d}_i \leftarrow D_{\mathbb{Z},s_1}^k$. Then, it sets $\mathbf{W}_i = \mathbf{Z}(\mathbf{d}_i \otimes \mathbf{I}_m)$. Finally, it samples

$$\forall j \neq i : \mathbf{y}_{i,j} \leftarrow \mathbf{A}_{s_1}^{-1}(\mathbf{W}_i \mathbf{r}_j) \quad \text{and} \quad \mathbf{y}_{i,i} \leftarrow \mathbf{A}_{s_1}^{-1}(\mathbf{p} + \mathbf{B}\mathbf{r}_i + \mathbf{W}_i \mathbf{r}_i).$$

- $\mathsf{Hyb}_5^{(b)}$: Same as $\mathsf{Hyb}_4^{(b)}$, except for all $i \in S^*$, the challenger samples the secret key component $\mathbf{y}_{i,i} \leftarrow \mathbf{A}_{s_1}^{-1}(\mathbf{W}_i \mathbf{r}_i)$.
- $\mathsf{Hyb}_6^{(b)}$: Same as $\mathsf{Hyb}_5^{(b)}$, except for all $i \in S^*$, the challenger samples the key component $\boldsymbol{\kappa}_i \leftarrow (\mathbf{V}_{N,k})_{s_1}^{-1}(\mathbf{0}^{nN})$ and the components $\mathbf{y}_{i,j}$, $\mathbf{d}_i$ are again derived from $\boldsymbol{\kappa}_i$ according to Eq. (3.6).
- $\mathsf{Hyb}_7^{(b)}$: Same as $\mathsf{Hyb}_6^{(b)}$ except for all $i \in S^*$, the challenger samples $\boldsymbol{\kappa}_i$ as $\boldsymbol{\kappa}_i \leftarrow \mathsf{SamplePre}(\mathbf{V}_{N,k}, \mathbf{T}_{\mathbf{V}}, \mathbf{0}^{nN}, s_1)$.
- $\mathsf{Hyb}_8^{(b)}$: Same as $\mathsf{Hyb}_7^{(b)}$, except the challenger sets $\mathbf{B} = \mathbf{B}^* - \mathbf{W}_{S^*}$, where $\mathbf{B}^* \xleftarrow{\text{R}} \mathbb{Z}_q^{n \times m}$.
- $\mathsf{Hyb}_9^{(b)}$: Same as $\mathsf{Hyb}_8^{(b)}$ except the challenger sets $\mathbf{B}^* = \mathbf{A}\mathbf{H}$ and $\mathbf{p} = \mathbf{A}\mathbf{h}$.
- $\mathsf{Hyb}_{10}^{(b)}$: Same as $\mathsf{Hyb}_9^{(b)}$ except the challenger samples $\mathbf{c}_1 \xleftarrow{\text{R}} \mathbb{Z}_q^m$ and sets $\mathbf{c}_2^{\mathsf{T}} = \mathbf{c}_1^{\mathsf{T}}\mathbf{H}$ and $c_3 = \mathbf{c}_1^{\mathsf{T}}\mathbf{h} + b \cdot \lfloor q/2 \rfloor$.
- $\mathsf{Hyb}_{11}^{(b)}$: Same as $\mathsf{Hyb}_{10}^{(b)}$ except the challenger samples $\mathbf{B}^* \xleftarrow{\text{R}} \mathbb{Z}_q^{n \times m}$, $\mathbf{p} \xleftarrow{\text{R}} \mathbb{Z}_q^n$, $\mathbf{c}_2 \xleftarrow{\text{R}} \mathbb{Z}_q^m$ and $c_3 \xleftarrow{\text{R}} \mathbb{Z}_q$.

We write $\mathsf{Hyb}_i^{(b)}(\mathcal{A})$ to denote the output distribution of an execution of $\mathsf{Hyb}_i^{(b)}$ with adversary $\mathcal{A}$. We now argue that each adjacent pair of distributions are indistinguishable.

**Lemma 6.** *Suppose $n \geq \lambda$, $m \geq 3n \log q$ and $s_0 \geq (mN + k)\log(nN)$. Then, $\mathsf{Hyb}_0^{(b)}(\mathcal{A}) \overset{\text{R}}{\approx} \mathsf{Hyb}_1^{(b)}(\mathcal{A})$.*

*Proof.* Since $m \geq 3n \log q$ and $s_0 \geq (mN + k)\log(nN)$, by Lemma 5, the distribution of $\mathbf{T}_{\mathbf{V}}$ in $\mathsf{Hyb}_0^{(b)}$ is statistically indistinguishable from sampling $\mathbf{T}_{\mathbf{V}} \leftarrow (\mathbf{V}_{N,k})_{s_0}^{-1}(\mathbf{G}_{nN})$.

**Lemma 7.** *Suppose $n \geq \lambda$, $m \geq 2n \log q$ and $s_1 \geq (mN + k)\sqrt{m} s_0 \log(nN)$. Then, $\mathsf{Hyb}_1^{(b)}(\mathcal{A}) \overset{\text{R}}{\approx} \mathsf{Hyb}_2^{(b)}(\mathcal{A})$.*

*Proof.* Since $m \geq 2n \log q$ and $q$ is prime, by Lemmas 2 and 4, we have that $\|\mathbf{T}_{\mathbf{V}}\| \leq \sqrt{m} s_0$ with overwhelming probability. Since $s_1 \geq (mN + k)\sqrt{m} s_0 \log(nN)$, by Lemma 5, the distribution of $\boldsymbol{\kappa}_i$ in $\mathsf{Hyb}_1^{(b)}$ is statistically close to sampling from $(\mathbf{V}_{N,k})_{s_1}^{-1}(\mathbf{u}_i \otimes (\mathbf{p} + \mathbf{B}\mathbf{r}_i))$. Since $N = \mathsf{poly}(\lambda)$, the claim now follows by a hybrid argument.

**Lemma 8.** *Suppose $n \geq \lambda$ and $m \geq 3n \log q$. Then, $\mathsf{Hyb}_2^{(b)}(\mathcal{A}) \overset{\text{R}}{\approx} \mathsf{Hyb}_3^{(b)}(\mathcal{A})$.*

*Proof.* Follows immediately by Lemma 5.

**Lemma 9.** *Suppose $n \geq \lambda$, $m \geq 2n \log q$, and $s_1 \geq \log(mN)$. Then, $\mathsf{Hyb}_3^{(b)}(\mathcal{A}) \overset{\mathrm{R}}{\leftarrow} \mathsf{Hyb}_4^{(b)}(\mathcal{A})$.*

*Proof.* By Lemma 4, for each $i \in S^*$, the distribution of $\{\mathbf{y}_{i,j}\}_{j \in [N]}$ and $\mathbf{d}_i$ in $\mathsf{Hyb}_3^{(b)}$ is statistically close to the distribution

$$\mathbf{d}_i \leftarrow D_{\mathbb{Z},s_1}^k, \forall j \neq i : \mathbf{y}_{i,j} \leftarrow \mathbf{A}_{s_1}^{-1}(\mathbf{Z}(\mathbf{I}_k \otimes \mathbf{r}_j)\mathbf{d}_i), \mathbf{y}_{i,i} \leftarrow \mathbf{A}_{s_1}^{-1}(\mathbf{p} + \mathbf{B}\mathbf{r}_i + \mathbf{Z}(\mathbf{I}_k \otimes \mathbf{r}_i)\mathbf{d}_i).$$

In $\mathsf{Hyb}_3^{(b)}$, the challenger then sets $\mathbf{W}_i = \mathbf{Z}(\mathbf{d}_i \otimes \mathbf{I}_m)$. In this case, by Eq. (2.1),

$$\mathbf{Z}(\mathbf{I}_k \otimes \mathbf{r}_j)\mathbf{d}_i = \mathbf{Z}(\mathbf{I}_k \otimes \mathbf{r}_j)(\mathbf{d}_i \otimes 1) = \mathbf{Z}(\mathbf{d}_i \otimes \mathbf{I}_m)\mathbf{r}_j = \mathbf{W}_i\mathbf{r}_j.$$

The challenger's sampling procedure in $\mathsf{Hyb}_3^{(b)}$ is thus equivalent to first sampling $\mathbf{d}_i \leftarrow D_{\mathbb{Z},s_1}^k$, then setting $\mathbf{W}_i = \mathbf{Z}(\mathbf{d}_i \otimes \mathbf{I}_m)$, and finally sampling

$$\forall j \neq i : \mathbf{y}_{i,j} \leftarrow \mathbf{A}_{s_1}^{-1}(\mathbf{W}_i\mathbf{r}_j) \quad \text{and} \quad \mathbf{y}_{i,i} \leftarrow \mathbf{A}_{s_1}^{-1}(\mathbf{p} + \mathbf{B}\mathbf{r}_i + \mathbf{W}_i\mathbf{r}_i).$$

This is the sampling procedure in $\mathsf{Hyb}_4^{(b)}$. Since $N = \mathsf{poly}(\lambda)$, the claim now follows by a hybrid argument over each $i \in S^*$.

**Lemma 10.** *The distributions $\mathsf{Hyb}_4^{(b)}(\mathcal{A})$ and $\mathsf{Hyb}_5^{(b)}(\mathcal{A})$ are identically distributed.*

*Proof.* The adversary's view in the two experiments is *independent* of $\mathbf{y}_{i,i}$ for all $i \in S^*$, so these two distributions are identical.

**Lemma 11.** *Suppose $n \geq \lambda$ and $m \geq 2n \log q$, and $s_1 \geq \log(mN)$. Then, $\mathsf{Hyb}_5^{(b)}(\mathcal{A}) \overset{\mathrm{R}}{\leftarrow} \mathsf{Hyb}_6^{(b)}(\mathcal{A})$.*

*Proof.* This follows by a similar argument as in the proof of Lemma 9. Specifically, by Lemma 4, the distribution of $\{\mathbf{y}_{i,j}\}_{j \in [N]}$ and $\mathbf{d}_i$ in $\mathsf{Hyb}_6^{(b)}$ is statistically close to the distribution

$$\mathbf{d}_i \leftarrow D_{\mathbb{Z},s_1}^k \quad \text{and} \quad \forall j \in [N] : \mathbf{y}_{i,j} \leftarrow \mathbf{A}_{s_1}^{-1}(\mathbf{Z}(\mathbf{I}_k \otimes \mathbf{r}_j)\mathbf{d}_i).$$

Then, the challenger sets $\mathbf{W}_i = \mathbf{Z}(\mathbf{d}_i \otimes \mathbf{I}_m)$. As in the proof of Lemma 9, we can write $\mathbf{W}_i\mathbf{r}_j = \mathbf{Z}(\mathbf{I}_k \otimes \mathbf{r}_j)\mathbf{d}_i$. Thus, the challenger is sampling $\mathbf{y}_{i,j} \leftarrow \mathbf{A}_{s_1}^{-1}(\mathbf{W}_i\mathbf{r}_j)$ for all $j \in [N]$. This is the distribution in $\mathsf{Hyb}_5^{(b)}$.

**Lemma 12.** *Suppose $n \geq \lambda$, $m \geq 2n \log q$ and $s_1 \geq (mN + k)\sqrt{m}s_0 \log(nN)$. Then, $\mathsf{Hyb}_6^{(b)}(\mathcal{A}) \overset{\mathrm{R}}{\leftarrow} \mathsf{Hyb}_7^{(b)}(\mathcal{A})$.*

*Proof.* Follows by the same argument as in the proof of Lemma 7.

**Lemma 13.** *The distributions $\mathsf{Hyb}_7^{(b)}(\mathcal{A})$ and $\mathsf{Hyb}_8^{(b)}(\mathcal{A})$ are identically distributed.*

*Proof.* In $\mathsf{Hyb}_7^{(b)}$ and $\mathsf{Hyb}_8^{(b)}$, none of the $\boldsymbol{\kappa}_i$ depend on $\mathbf{B}$. As such, the distribution of $\mathbf{B}$ is uniform and *independent* of $\mathbf{W}_{S^*}$ in both experiments. As such, these two distributions are identical.

**Lemma 14.** *Suppose $n \geq \lambda$, $m \geq 2n \log q$ and $q > 2$ is prime. Then,* $\mathsf{Hyb}_8^{(b)}(\mathcal{A}) \overset{s}{\approx} \mathsf{Hyb}_9^{(b)}(\mathcal{A})$.

*Proof.* By Lemma 1, for all $\mathbf{e} \in \mathbb{Z}_q^m$, the following two distributions are statistically indistinguishable:

$$(\mathbf{A}, \mathbf{AH}, \mathbf{Ah}, \mathbf{e}^\intercal \mathbf{H}, \mathbf{e}^\intercal \mathbf{h}) \quad \text{and} \quad (\mathbf{A}, \mathbf{B}^*, \mathbf{p}, \mathbf{e}^\intercal \mathbf{H}, \mathbf{e}^\intercal \mathbf{h}),$$

where $\mathbf{A} \xleftarrow{\mathrm{R}} \mathbb{Z}_q^{n\times m}, \mathbf{B}^* \xleftarrow{\mathrm{R}} \mathbb{Z}_q^{n\times m}, \mathbf{p} \xleftarrow{\mathrm{R}} \mathbb{Z}_q^n, \mathbf{H} \xleftarrow{\mathrm{R}} \{0,1\}^{m\times m}, \mathbf{h} \xleftarrow{\mathrm{R}} \{0,1\}^m$. The left and right distributions correspond to $\mathsf{Hyb}_9^{(b)}$ and $\mathsf{Hyb}_8^{(b)}$, respectively.

**Lemma 15.** *Suppose the $N$-structured LWE (Assumption 3) holds with parameters $(n, m, q, \sigma, s_0, k)$. Then, $sslo_9(\mathcal{A}) \overset{c}{\approx} \mathsf{Hyb}_{10}^{(b)}(\mathcal{A})$.*

*Proof.* Suppose there exists a bit $b \in \{0,1\}$ and an efficient adversary $\mathcal{A}$ that can distinguish between $\mathsf{Hyb}_9^{(b)}$ and $\mathsf{Hyb}_{10}^{(b)}$ with non-negligible advantage $\varepsilon > 0$. We use algorithm $\mathcal{A}$ to construct an algorithm $\mathcal{B}$ that breaks the $N$-structured LWE assumption with parameters $(n, m, q, \sigma, s_0, k)$:

1. At the start of the game, algorithm $\mathcal{B}$ receives an $N$-structured LWE challenge $(\mathbf{A}, \mathbf{c}_1^\intercal, \mathbf{Z}, \mathbf{R}, \mathbf{T}_\mathbf{V})$ from its challenger. Let $\mathbf{V}_{N,k}$ be the matrix from Eq. (3.2) formed from the components $\mathbf{Z}$ and $\mathbf{R} = [\mathbf{r}_1 \mid \cdots \mid \mathbf{r}_N]$.
2. Algorithm $\mathcal{B}$ samples $\mathbf{H} \xleftarrow{\mathrm{R}} \{0,1\}^{m\times m}$ and $\mathbf{h} \xleftarrow{\mathrm{R}} \{0,1\}^m$, then sets $\mathbf{B}^* = \mathbf{AH}, \mathbf{p} = \mathbf{Ah}, \mathbf{c}_2^\intercal = \mathbf{c}_1^\intercal \mathbf{H}$ and $c_3 = \mathbf{c}_1^\intercal \mathbf{h} + b \cdot \lfloor q/2 \rfloor$.
3. Algorithm $\mathcal{B}$ starts running algorithm $\mathcal{A}$ and receives a set $S^* \subseteq [N]$. For all $i \in S^*$, algorithm $\mathcal{B}$ samples $\boldsymbol{\kappa}_i \leftarrow \mathsf{SamplePre}(\mathbf{V}_{N,k}, \mathbf{T}_\mathbf{V}, \mathbf{0}^{nN}, s_1)$ and sets $\mathbf{W}_i = \mathbf{Z}(\mathbf{d}_i \otimes \mathbf{I}_m)$, where $\mathbf{y}_{i,j}$ and $\mathbf{d}_i$ are derived from $\boldsymbol{\kappa}_i$ as in Eq. (3.6).
4. Algorithm $\mathcal{B}$ sets $\mathbf{B} = \mathbf{B}^* - \mathbf{W}_{S^*}$, $\mathsf{pp} = (\mathbf{A}, \mathbf{B}, \mathbf{p}, \mathbf{Z}, \{\mathbf{r}_i\}_{i\in[N]}, \mathbf{T}_\mathbf{V})$, and $\mathsf{pk}_i = (\mathbf{W}_i, \{\mathbf{y}_{i,j}\}_{j\neq i})$ for each $i \in S^*$. It sets $\mathsf{ct}_b = (\mathbf{c}_1^\intercal, \mathbf{c}_2^\intercal, c_3)$. It gives $(\mathsf{pp}, \{\mathsf{pk}_i\}_{i\in S^*}, \mathsf{ct}_b)$ to $\mathcal{A}$ and outputs whatever algorithm $\mathcal{A}$ outputs.

We first show that $\mathcal{B}$ correctly simulates an execution of $\mathsf{Hyb}_9^{(b)}$ and $\mathsf{Hyb}_{10}^{(b)}$ for $\mathcal{A}$.

- The $N$-structured LWE challenger samples $\mathbf{A} \xleftarrow{\mathrm{R}} \mathbb{Z}_q^{n\times m}$, $\mathbf{Z} \xleftarrow{\mathrm{R}} \mathbb{Z}_q^{n\times mk}$, and $\mathbf{R} \leftarrow D_{\mathbb{Z},s_0}^{m\times \ell}$. Moreover, the challenger samples $\mathbf{T}_\mathbf{V} \leftarrow (\mathbf{V}_{N,k})_{s_0}^{-1}(\mathbf{G}_{nN})$, which coincides with the distribution of the public parameters in $\mathsf{Hyb}_9^{(b)}$ and $\mathsf{Hyb}_{10}^{(b)}$. Next, algorithm $\mathcal{B}$ sets $\mathbf{B} = \mathbf{AH}$ and $\mathbf{p} = \mathbf{Ah}$, so we conclude that the public parameters $\mathsf{pp}$ are perfectly simulated.
- Next, algorithm $\mathcal{B}$ samples $\boldsymbol{\kappa}_i$ using the same procedure as in $\mathsf{Hyb}_9^{(b)}$ and $\mathsf{Hyb}_{10}^{(b)}$, so the public keys are perfectly simulated.
- Consider the distribution of the challenge ciphertext:

- If $\mathbf{c}_1^\mathsf{T} = \mathbf{s}^\mathsf{T}\mathbf{A} + \mathbf{e}^\mathsf{T}$ where $\mathbf{s} \xleftarrow{\text{R}} \mathbb{Z}_q^n$ and $\mathbf{e} \leftarrow D_{\mathbb{Z},\sigma}^m$, then

$$\begin{aligned} \mathbf{c}_2^\mathsf{T} &= \mathbf{c}_1^\mathsf{T}\mathbf{H} = \mathbf{s}^\mathsf{T}\mathbf{A}\mathbf{H} + \mathbf{e}^\mathsf{T}\mathbf{H} = \mathbf{s}^\mathsf{T}\mathbf{B}^* + \mathbf{e}^\mathsf{T}\mathbf{H} = \mathbf{s}^\mathsf{T}(\mathbf{B} + \mathbf{W}_{S^*}) + \mathbf{e}^\mathsf{T}\mathbf{H} \\ c_3 &= \mathbf{c}_1^\mathsf{T}\mathbf{h} + b \cdot \lfloor q/2 \rfloor = \mathbf{s}^\mathsf{T}\mathbf{A}\mathbf{h} + \mathbf{e}^\mathsf{T}\mathbf{p} + b \cdot \lfloor q/2 \rfloor = \mathbf{s}^\mathsf{T}\mathbf{p} + \mathbf{e}^\mathsf{T}\mathbf{h} + b \cdot \lfloor q/2 \rfloor, \end{aligned}$$

  which is the distribution of the challenge ciphertext in $\mathsf{Hyb}_9^{(b)}$.
- Conversely, if $\mathbf{c}_1 \xleftarrow{\text{R}} \mathbb{Z}_q^m$. Then, algorithm $\mathcal{B}$ perfectly simulates an execution of $\mathsf{Hyb}_{10}^{(b)}$.

We conclude that algorithm $\mathcal{B}$ breaks the $N$-structured LWE problem with the same advantage $\varepsilon$.

**Lemma 16.** *If $n \geq \lambda$, $m \geq 2(n+1)\log q$, and $q > 2$ is a prime, then* $\mathsf{Hyb}_{10}^{(b)}(\mathcal{A}) \xleftarrow{\text{R}} \mathsf{Hyb}_{11}^{(b)}(\mathcal{A})$.

*Proof.* This follows from Lemma 1 applied to the matrix $\begin{bmatrix} \mathbf{A} \\ \mathbf{c}_1^\mathsf{T} \end{bmatrix} \in \mathbb{Z}_q^{(n+1)\times m}$.

**Lemma 17.** *The experiments* $\mathsf{Hyb}_{10}^{(0)}(\mathcal{A})$ *and* $\mathsf{Hyb}_{11}^{(1)}(\mathcal{A})$ *are identically distributed.*

*Proof.* By construction, the challenger's behavior in $\mathsf{Hyb}_{11}^{(b)}$ is *independent* of the challenge bit $b \in \{0,1\}$, so the adversary's view in the two distributions is identical.

Combining Lemmas 6 to 17, selective security follows by a hybrid argument.

*Parameter Instantiation.* Let $\lambda$ be a security parameter and $N$ be a bound on the number of users. We can instantiate the lattice parameters in Construction 4 to satisfy Theorems 5 to 7:

– We set the lattice dimension $n = \lambda$ and $m = O(n \log q)$.
– We set the noise parameter $\sigma = \mathsf{poly}(n)$ (such that the LWE assumption with parameters $(n, m, q, \sigma)$ holds). We set the dimension to be $k = O(nm \log q)$.
– We set $s_0 = (mN + k)\log(nN)$ and $s_1 = (mN+k)\sqrt{m}s_0 \log(nN) = (mN + k)^2\sqrt{m}\log^2(nN)$.
– Finally, we set the norm bound $\beta = \sqrt{m}s_1 = (mN+k)^2 m \log^2(nN)$ and the modulus $q$ such that

$$q \geq 4\sqrt{n}m\sigma(1 + N\beta + \sqrt{n}ms_0) = N^3 \cdot \mathsf{poly}(\lambda, \log N).$$

  In this case, $\log q = O(\log N + \log \lambda)$.

With this setting of parameters, we obtain a distributed broadcast encryption scheme with the following parameter sizes. Without loss of generality, we assume that $N \geq \lambda$.

– **Public parameter size:** The public parameters $\mathsf{pp}$ have size $|\mathsf{pp}| = N^2 \cdot \mathsf{poly}(\lambda, \log N)$.

- **Public key size:** Each user's public key $\mathsf{pk}$ consists of a matrix $\mathbf{W} \in \mathbb{Z}_q^{n \times m}$ and $N-1$ cross-terms $\mathbf{y}_j \in \mathbb{Z}_q^m$, so $|\mathsf{pk}| \leq (n+N)m \log q = O(N\lambda \log^2 N)$.
- **Secret key size:** The secret key for user $i \in [N]$ consists of a vector $\mathbf{y}_i \in \mathbb{Z}_q^m$, so $|\mathsf{sk}_i| = O(m \log q) = O(\lambda \log^2 N)$.
- **Ciphertext size:** The ciphertext for any set $S \subseteq [N]$ and message $\mu \in \{0,1\}$ consists of of $2m+1$ elements of $\mathbb{Z}_q$, so $|\mathsf{ct}| = O(\lambda \log^2 N)$.

Combined with the reduction from $\ell$-structured LWE to $\ell'$-succinct LWE (for $\ell' \geq \ell \cdot O(n \log q)$) from Sect. 4 (Theorem 8), we obtain the following corollary:

**Corollary 2 (Distributed Broadcast Encryption from $\ell$-succinct LWE).** *Let $\lambda$ be a security parameter and $N = N(\lambda)$ be any polynomial. Let $\ell \geq N \cdot O(\lambda \log N)$. Under the $\ell$-succinct LWE assumption for $\ell = N \cdot O(\lambda \log N)$ (and a polynomial modulus-to-noise ratio), there exists a selectively-secure distributed broadcast encryption scheme. Both the size of the ciphertext and a user's secret key is $O(\lambda \log^2 N)$, the size of a user's public key is $O(N\lambda \log^2 N)$, and the size of the public parameters is $N^2 \cdot \mathsf{poly}(\lambda, \log N)$.*

*Remark 2 (Precomputing Encryption and Decryption Keys).* Similar to the pairing-based constructions of distributed broadcast encryption [KMW23, GLWW23, GKPW24], we can improve the efficiency of the encryption and decryption algorithms when the broadcast set $S$ is known in advance. Specifically, we can view the matrix $\mathbf{W}_S = \sum_{i \in S} \mathbf{W}_i$ as the public key for encrypting to the set $S$; given $\mathbf{W}_S$, encrypting a message to the set $S$ just requires time $\mathsf{poly}(\lambda, \log N)$. Similarly, if user $j$ knows the broadcast set $S$ in advance, she can pre-compute her decryption component $\mathbf{y}_{j,S} := \sum_{i \in S \setminus \{j\}} \mathbf{y}_{i,j}$. Given $\mathbf{y}_{j,S}$, decryption now runs in time $\mathsf{poly}(\lambda, \log N)$. Precomputation is useful in settings where users frequently send or receive broadcasts to the *same* set $S$.

## 4 Relating $\ell$-Structured LWE and $\ell$-Succinct LWE

In this section, we formally show that the $\ell$-structured LWE assumption (Assumption 3) used in Sect. 3.1 follows under the $\ell'$-succinct LWE assumption (Assumption 2) when $\ell' \geq \ell \cdot O(n \log q)$, where $n$ is the lattice dimension and $q$ is the modulus. Essentially, our proof shows how to build a trapdoor for the $\ell$-structured LWE assumption using a trapdoor for the $\ell'$-succinct LWE assumption. We refer to Sect. 1.1 for a high-level overview of our proof strategy.

**Theorem 8 ($\ell'$-succinct LWE implies $\ell$-structured LWE).** *Let $\lambda$ be a security parameter and $n = n(\lambda), m = m(\lambda), q = q(\lambda), \sigma = \sigma(\lambda)$ be lattice parameters. Let $s = s(\lambda), s' = s'(\lambda)$ be Gaussian width parameters, and $\ell = \ell(\lambda), k = k(\lambda)$ be polynomially-bounded dimensions. Suppose $q$ is prime and the following conditions hold:*

- *$n \geq \lambda$, $m \geq 2n \log q$, $k \geq 3nm \log q$, $q \leq 2^n$;*
- *$s' \geq \log m$, $s \geq \max\{m^{3/2}(\ell'+1)s' \log(n\ell'), k \log(nm)\}$.*

*Let $\ell' = \ell n(\lfloor \log q \rfloor + 1)$. Then, the $\ell'$-succinct LWE assumption with parameters $(n, m, q, \sigma, s')$ implies the $\ell$-structured LWE with parameters $(n, m, q, \sigma, s, k)$.*

*Proof.* We show how to transform the components in an $\ell'$-succinct LWE instance into those of an $\ell$-structured LWE instance. Specifically, consider the components $(\mathbf{A}, \mathbf{U}, \mathbf{T}')$ in an $\ell'$-succinct LWE instance with parameters $(n, m, q, \sigma, s')$:

$$\mathbf{A} \xleftarrow{R} \mathbb{Z}_q^{n \times m} \quad \text{and} \quad \mathbf{U} \xleftarrow{R} \mathbb{Z}_q^{n\ell' \times m} \quad \text{and} \quad \mathbf{T}' \leftarrow [\mathbf{I}_{\ell'} \otimes \mathbf{A} \mid \mathbf{U}]_{s'}^{-1}(\mathbf{G}_{n\ell'}), \tag{4.1}$$

We show how to use these components to construct a tuple $(\mathbf{A}, \mathbf{Z}, \mathbf{R}, \mathbf{T})$ distributed according to the specification of an $\ell$-structured LWE instance with parameters $(n, m, q, \sigma, s, k)$:

$$\mathbf{A} \xleftarrow{R} \mathbb{Z}_q^{n \times m} \quad \text{and} \quad \mathbf{Z} \xleftarrow{R} \mathbb{Z}_q^{n \times mk} \quad \text{and} \quad \mathbf{R} \leftarrow D_{\mathbb{Z},s}^{m \times \ell} \quad \text{and} \quad \mathbf{T} \leftarrow (\mathbf{V}_{\ell,k})_s^{-1}(\mathbf{G}_{n\ell}), \tag{4.2}$$

where $\mathbf{V}_{\ell,k}$ is the matrix from Eq. (3.1). We construct a reduction algorithm $\mathcal{R}$ as follows:

1. On input $(\mathbf{A}, \mathbf{U}, \mathbf{T}')$, parse

$$\mathbf{U} = \begin{bmatrix} \mathbf{U}_1 \\ \vdots \\ \mathbf{U}_{\ell'} \end{bmatrix} \in \mathbb{Z}_q^{n\ell \times m}, \tag{4.3}$$

where $\mathbf{U}_i \in \mathbb{Z}_q^{n \times m}$. Next, write the gadget matrix $\mathbf{G}_{n\ell}$ as

$$\mathbf{G}_{n\ell} = \begin{bmatrix} \mathbf{v}_{1,1} & \cdots & \mathbf{v}_{1,\ell'} \\ \vdots & \ddots & \vdots \\ \mathbf{v}_{\ell,1} & \cdots & \mathbf{v}_{\ell,\ell'} \end{bmatrix} \in \mathbb{Z}_q^{n\ell \times \ell'}, \tag{4.4}$$

where $\mathbf{v}_{i,j} \in \mathbb{Z}_q^n$ for all $i \in [\ell]$ and $j \in [\ell']$. For each $i \in [\ell]$, define

$$\hat{\mathbf{v}}_i := \begin{bmatrix} \mathbf{v}_{i,1} \\ \vdots \\ \mathbf{v}_{i,\ell'} \end{bmatrix} \in \mathbb{Z}_q^{n\ell'}.$$

Then, for each $i \in [\ell]$, sample

$$\begin{bmatrix} \mathbf{x}_{i,1} \\ \vdots \\ \mathbf{x}_{i,\ell'} \\ \mathbf{r}_i \end{bmatrix} \leftarrow \mathsf{SamplePre}\big([\mathbf{I}_{\ell'} \otimes \mathbf{A} \mid \mathbf{U}], \mathbf{T}', \hat{\mathbf{v}}_i, s\big) \in \mathbb{Z}_q^{m\ell'+m}, \tag{4.5}$$

where $\mathbf{x}_{i,j}, \mathbf{r}_i \in \mathbb{Z}_q^m$.

2. Next, sample $(\mathbf{Z}', \mathbf{T}_{\mathbf{Z}'}) \leftarrow \mathsf{TrapGen}(1^{nm}, q, k)$ and
$$\mathbf{d}_j \leftarrow \mathsf{SamplePre}(\mathbf{Z}', \mathbf{T}_{\mathbf{Z}'}, \mathsf{vec}(-\mathbf{U}_j), s) \in \mathbb{Z}_q^k \tag{4.6}$$
for each $j \in [\ell']$. Here, $\mathsf{vec}(-\mathbf{U}_i) \in \mathbb{Z}_q^{nm}$ denotes the vectorization of $-\mathbf{U}_i$ (i.e., the vector obtained by vertically stacking the columns of $-\mathbf{U}_i$ from left to right).
3. For each $i \in [k]$, let $\mathbf{Z}_i \in \mathbb{Z}_q^{n \times m}$ be the matrix where $\mathsf{vec}(\mathbf{Z}_i) = \mathbf{z}'_i$ and $\mathbf{z}'_i \in \mathbb{Z}_q^{nm}$ is the $i^{\text{th}}$ column of $\mathbf{Z}'$. Let $\mathbf{Z} = [\mathbf{Z}_1 \mid \cdots \mid \mathbf{Z}_k] \in \mathbb{Z}_q^{n \times mk}$ and $\mathbf{R} = [\mathbf{r}_1 \mid \cdots \mid \mathbf{r}_\ell] \in \mathbb{Z}_q^{m \times \ell}$. Finally, let
$$\mathbf{V}_{\ell,k} = \left[\begin{array}{ccc|c} \mathbf{A} & & & -\mathbf{Z}(\mathbf{I}_k \otimes \mathbf{r}_1) \\ & \ddots & & \vdots \\ & & \mathbf{A} & -\mathbf{Z}(\mathbf{I}_k \otimes \mathbf{r}_\ell) \end{array}\right] \in \mathbb{Z}_q^{n\ell \times (m\ell + k)}, \mathbf{T} = \begin{bmatrix} \mathbf{x}_{1,1} & \cdots & \mathbf{x}_{1,\ell'} \\ \vdots & \ddots & \vdots \\ \mathbf{x}_{\ell,1} & \cdots & \mathbf{x}_{\ell,\ell'} \\ \mathbf{d}_1 & \cdots & \mathbf{d}_{\ell'} \end{bmatrix} \in \mathbb{Z}_q^{(m\ell+k) \times \ell'}. \tag{4.7}$$
Output $(\mathbf{A}, \mathbf{Z}, \mathbf{R}, \mathbf{T})$.

In the full version of this paper [CW24], we show that if the inputs $(\mathbf{A}, \mathbf{U}, \mathbf{T}')$ to the above algorithm are distributed according to Eq. (4.1), then the outputs $(\mathbf{A}, \mathbf{Z}, \mathbf{R}, \mathbf{T})$ are statistically close to the distribution in Eq. (4.2).

*SIS Variants.* Our proof for Theorem 8 shows how to use a trapdoor for the $\ell'$-succinct LWE assumption to construct a trapdoor for the $\ell$-structured LWE assumption. The same transformation would apply to show a similar reduction between the $\ell'$-succinct SIS assumption and the $\ell$-structured SIS assumption. We can also construct an analogous reduction algorithm that transforms an instance of the $\ell'$-structured LWE assumption into an instance of the $\ell$-succinct LWE assumption. Thus, up to a $O(n \log q)$ increase in the dimension (and polynomial blow-up in the noise parameters), these assumptions are essentially equivalent.

**Acknowledgments.** We thank Hoeteck Wee for many insightful discussions on distributed broadcast encryption and the $\ell$-succinct LWE assumption. We thank Brent Waters and the anonymous TCC reviewers for providing many helpful comments on the presentation. David J. Wu is supported by NSF CNS-2140975, CNS-2318701, a Microsoft Research Faculty Fellowship, and a Google Research Scholar award.

## References

[ABB10] Agrawal, S., Boneh, D., Boyen, X.: Efficient lattice (H)IBE in the standard model. In: Gilbert, H. (ed.) EUROCRYPT 2010. LNCS, vol. 6110, pp. 553–572. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-13190-5_28

[ACL+22] Albrecht, M.R., Cini, V., Lai, R.W.F., Malavolta, R.W.F., Thyagarajan, S.A.K.: Lattice-based SNARKs: publicly verifiable, preprocessing, and recursively composable - (extended abstract). In: Dodis, Y., Shrimpton, T. (eds.) CRYPTO 2022. LNCS, vol. 13508, pp. 102–132. Springer, Cham (2022)

[AFLN24] Albrecht, M.R., Fenzi, G., Lapiha, O., Nguyen, N.K.: SLAP: succinct lattice-based polynomial commitments from standard assumptions. In: Joye, M., Leander, G. (eds.) EUROCRYPT 2024. LNCS, vol. 14657, pp. 90–119. Springer, Cham (2024). https://doi.org/10.1007/978-3-031-58754-2_4

[AGHS13] Agrawal, S., Gentry, C., Halevi, S., Sahai, A.: Discrete gaussian leftover hash lemma over infinite domains. In: Sako, K., Sarkar, P. (eds.) ASIACRYPT 2013. LNCS, vol. 8269, pp. 97–116. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-42033-7_6

[Ajt96] Ajtai, M.: Generating hard instances of lattice problems (extended abstract). In: STOC (1996)

[Alb24] Albrecht, M.R.: SIS with hints zoo (2024

[AR16] Aggarwal, D., Regev, O.: A note on discrete gaussian combinations of lattice vectors. Chic. J. Theor. Comput. Sci. (2016)

[AT24] Attrapadung, N., Tomida, J.: A modular approach to registered ABE for unbounded predicates. In: Reyzin, L., Stebila, D. (eds.) CRYPTO 2024. LNCS, vol. 14922, pp. 280–316. Springer, Cham (2024)

[BCFL23] Balbás, D., Catalano, D., Fiore, D., Lai, R.W.F.: Chainable functional commitments for unbounded-depth circuits. In: Rothblum, G., Wee, H. (eds.) TCC 2023. LNCS, vol. 14371, pp. 363–393. Springer, Cham (2023). https://doi.org/10.1007/978-3-031-48621-0_13

[BGG+14] Boneh, D., et al.: Fully key-homomorphic encryption, arithmetic circuit ABE and compact garbled circuits. In: Nguyen, P.Q., Oswald, E. (eds.) EUROCRYPT 2014. LNCS, vol. 8441, pp. 533–556. Springer, Heidelberg (2014). https://doi.org/10.1007/978-3-642-55220-5_30

[BGI+01] Barak, B., et al.: On the (im)possibility of obfuscating programs. In: Kilian, J. (ed.) CRYPTO 2001. LNCS, vol. 2139, pp. 1–18. Springer, Heidelberg (2001). https://doi.org/10.1007/3-540-44647-8_1

[BGW05] Boneh, D., Gentry, C., Waters, B.: Collusion resistant broadcast encryption with short ciphertexts and private keys. In: Shoup, V. (ed.) Collusion resistant broadcast encryption with short ciphertexts and private keys. LNCS, vol. 3621, pp. 258–275. Springer, Heidelberg (2005). https://doi.org/10.1007/11535218_16

[BLM+24] Branco, P., Lai, R.W.F., Maitra, M., Malavolta, G., Rahimi, A., Woo, I.K.Y.: Traitor tracing without trusted authority from registered functional encryption. IACR Cryptol. ePrint Arch. (2024)

[BLMR13] Boneh, D., Lewi, K., Montgomery, H., Raghunathan, A.: Key homomorphic PRFs and their applications. In: Canetti, R., Garay, J.A. (eds.) CRYPTO 2013. LNCS, vol. 8042, pp. 410–428. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-40041-4_23

[BV22] Brakerski, Z., Vaikuntanathan, V.: Lattice-inspired broadcast encryption and succinct ciphertext-policy ABE. In: ITCS (2022)

[BZ14] Boneh, D., Zhandry, M.: Multiparty key exchange, efficient traitor tracing, and more from indistinguishability obfuscation. In: Garay, J.A., Gennaro, R. (eds.) CRYPTO 2014. LNCS, vol. 8616, pp. 480–499. Springer, Heidelberg (2014). https://doi.org/10.1007/978-3-662-44371-2_27

[CES21] Cong, K., Eldefrawy, K., Smart, N.P.: Optimizing registration based encryption. In: Cryptography and Coding (2021)

[CHKP10] Cash, D., Hofheinz, D., Kiltz, E., Peikert, C.: Bonsai trees, or how to delegate a lattice basis. In: Gilbert, H. (ed.) EUROCRYPT 2010. LNCS, vol. 6110, pp. 523–552. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-13190-5_27

[CLM23] Cini, V., Lai, R.W.F., Malavolta, G.: Lattice-based succinct arguments from vanishing polynomials - (extended abstract). In: Handschuh, H., Lysyanskaya, A. (eds.) CRYPTO 2023. LNCS, vol. 14082, pp. 72–105. Springer, Cham (2023). https://doi.org/10.1007/978-3-031-38545-2_3

[CW24] Champion, J., Wu, D.J.: Distributed broadcast encryption from lattices. IACR Cryptol. ePrint Arch. (2024). https://eprint.iacr.org/2024/1417.pdf

[DKL+23] Döttling, N., Kolonelos, D., Lai, R.W.F., Lin, C., Malavolta, G., Rahimi, A.: Efficient laconic cryptography from learning with errors. In: Hazay, C., Stam, M. (eds.) EUROCRYPT 2023. LNCS, vol. 14006, pp. 417–446. Springer, Cham (2023). https://doi.org/10.1007/978-3-031-30620-4_14

[DKW21] Datta, P., Komargodski, I., Waters, B.: Decentralized multi-authority ABE for DNFs from LWE. In: Canteaut, A., Standaert, F.-X. (eds.) EUROCRYPT 2021. LNCS, vol. 12696, pp. 177–209. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-77870-5_7

[DPY23] Datta, P., Pal, T., Yamada, S.: Registered FE beyond predicates: (attribute-based) linear functions and more. Cryptology ePrint Archive (2023)

[FFM+23] Francati, D., Friolo, D., Maitra, M., Malavolta, G., Rahimi, A., Venturi, D.: Registered (inner-product) functional encryption. In: Guo, J., Steinfeld, R. (eds.) ASIACRYPT 2023. LNCS, vol. 14442, pp. 98–133. Springer, Singapore (2023). https://doi.org/10.1007/978-981-99-8733-7_4

[FKdP23] Fiore, D., Kolonelos, D., de Perthuis, P.: Cuckoo commitments: registration-based encryption and key-value map commitments for large spaces. In: ASIACRYPT 2023. LNCS, vol. 14442, pp. 166–200. Springer, Singapore (2023). https://doi.org/10.1007/978-981-99-8733-7_6

[FMN23] Fenzi, G., Moghaddas, H., Nguyen, N.K.: Lattice-based polynomial commitments: towards asymptotic and concrete efficiency. IACR Cryptol. ePrint Arch. (2023)

[FN93] Fiat, A., Naor, M.: Broadcast encryption. In: Stinson, D.R. (ed.) CRYPTO 1993. LNCS, vol. 773, pp. 480–491. Springer, Heidelberg (1994). https://doi.org/10.1007/3-540-48329-2_40

[FWW23] Freitag, C., Waters, B., David, J.W.: How to use (plain) witness encryption: registered ABE, flexible broadcast, and more. In: CRYPTO 2023. LNCS, vol. 14084, pp. 498–531. Springer, Cham (2023). https://doi.org/10.1007/978-3-031-38551-3_16

[GGH+13] Garg, S., Gentry, C., Halevi, S., Raykova, M., Sahai, A., Waters, B.: Candidate indistinguishability obfuscation and functional encryption for all circuits. In: FOCS (2013)

[GHM+19] Garg, S., Hajiabadi, M., Mahmoody, M., Rahimi, A., Sekar, S.: Registration-based encryption from standard assumptions. In: Lin, D., Sako, K. (eds.) PKC 2019. LNCS, vol. 11443, pp. 63–93. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-17259-6_3

[GHMR18] Garg, S., Hajiabadi, M., Mahmoody, M., Rahimi, A.: Registration-based encryption: removing private-key generator from IBE. In: Beimel, A., Dziembowski, S. (eds.) TCC 2018. LNCS, vol. 11239, pp. 689–718. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-03807-6_25

[GKMR23] Glaeser, N., Kolonelos, D., Malavolta, G., Rahimi, A.: Efficient registration-based encryption. In: ACM CCS (2023)

[GKPW24] Garg, S., Kolonelos, D., Policharla, G.-V., Wang, M.: Threshold encryption with silent setup. In: Reyzin, L., Stebila, D. (eds.) CRYPTO 2024. LNCS, vol. 14926, pp. 352–386. Springer, Cham (2021). https://doi.org/10.1007/978-3-031-68394-7_12

[GKW18] Gay, R., Kowalczyk, L., Wee, H.: Tight adaptively secure broadcast encryption with short ciphertexts and keys. In: Catalano, D., De Prisco, R. (eds.) SCN 2018. LNCS, vol. 11035, pp. 123–139. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-98113-0_7

[GLWW23] Garg, R., Lu, G., Waters, B., Wu, D.J.: Realizing flexible broadcast encryption: how to broadcast to a public-key directory. In: ACM CCS (2023)

[GLWW24] Garg, R., Lu, G., Waters, B., Wu, D.J.: Reducing the CRS size in registered ABE systems. In: Reyzin, L., Stebila, D. (eds.) CRYPTO 2024. LNCS, vol. 14922, pp. 143–177. Springer, Cham (2024). https://doi.org/10.1007/978-3-031-68382-4_5

[GPV08] Gentry, C., Peikert, C., Vaikuntanathan, V.: Trapdoors for hard lattices and new cryptographic constructions. In: STOC (2008)

[GV20] Goyal, R., Vusirikala, S.: Verifiable registration-based encryption. In: Micciancio, D., Ristenpart, T. (eds.) CRYPTO 2020. LNCS, vol. 12170, pp. 621–651. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-56784-2_21

[GVW13] Gorbunov, S., Vaikuntanathan, V., Wee, H.: Attribute-based encryption for circuits. In: STOC (2013)

[GW09] Gentry, C., Waters, B.: Adaptive security in broadcast encryption systems (with short ciphertexts). In: Joux, A. (ed.) EUROCRYPT 2009. LNCS, vol. 5479, pp. 171–188. Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-642-01001-9_10

[HLL23] Hsieh, Y.-C., Lin, H., Luo, J.: Attribute-based encryption for circuits of unbounded depth from lattices. In: FOCS (2023)

[HLWW23] Hohenberger, S., Lu, G., Waters, B., Wu, D.J.: Registered attribute-based encryption. In: Hazay, C., Stam, M. (eds.) EUROCRYPT 2023. LNCS, vol. 14006, pp. 511–542. Springer, Cham (2023). https://doi.org/10.1007/978-3-031-30620-4_17

[KMW23] Kolonelos, D., Malavolta, G., Wee, H.: Distributed broadcast encryption from bilinear groups. In: Guo, J., Steinfeld, R. (eds.) ASIACRYPT 2023. LNCS, vol. 14442, pp. 407–441. Springer, Cham (2023). https://doi.org/10.1007/978-981-99-8733-7_13

[MP12] Micciancio, D., Peikert, C.: Trapdoors for lattices: simpler, tighter, faster, smaller. In: Pointcheval, D., Johansson, T. (eds.) EUROCRYPT 2012. LNCS, vol. 7237, pp. 700–718. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-29011-4_41

[PPS12] Phan, D.H., Pointcheval, D., Strefler, M.: Decentralized dynamic broadcast encryption. In: Visconti, I., De Prisco, R. (eds.) SCN 2012. LNCS, vol. 7485, pp. 166–183. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-32928-9_10

[Reg05] Regev, O.: On lattices, learning with errors, random linear codes, and cryptography. In: STOC (2005)

[Tsa22] Tsabary, R.: Candidate witness encryption from lattice techniques. In: Dodis, Y., Shrimpton, T. (eds.) CRYPTO 2022. LNCS, vol. 13507, pp. 535–559. Springer, Cham (2022). https://doi.org/10.1007/978-3-031-15802-5_19

[VWW22] Vaikuntanathan, V., Wee, H., Wichs, D.: Witness encryption and Null-IO from evasive LWE. In: Agrawal, S., Lin, D. (eds.) ASIACRYPT 2022. LNCS, vol. 13791, pp. 195–221. Springer, Cham (2022). https://doi.org/10.1007/978-3-031-22963-3_7

[Wee22] Wee, H.: Optimal broadcast encryption and CP-ABE from evasive lattice assumptions. In: Dunkelman, O., Dziembowski, S. (eds.) EUROCRYPT 2022. LNCS, vol. 13276, pp. 217–241. Springer, Cham (2022). https://doi.org/10.1007/978-3-031-07085-3_8

[Wee24] Wee, H.: Circuit ABE with $poly(\text{depth}, \lambda)$-sized ciphertexts and keys from lattices. In: Reyzin, L., Stebila, D. (eds.) CRYPTO 2024. LNCS, vol. 14922, pp. 178–209. Springer, Cham (2024). https://doi.org/10.1007/978-3-031-68382-4_6

[WQZDF10] Wu, Q., Qin, B., Zhang, L., Domingo-Ferrer, J.: Ad hoc broadcast encryption. In: ACM CCS (2010)

[WW23a] Wee, H., Wu, D.J.: Lattice-based functional commitments: fast verification and cryptanalysis. In: Guo, J., Steinfeld, R. (eds.) ASIACRYPT 2023. LNCS, vol. 14442, pp. 201–235. Springer, Singapore (2023)

[WW23b] Wee, H., Wu, D.J.: Succinct vector, polynomial, and functional commitments from lattices. In: Hazay, C., Stam, M. (eds.) EUROCRYPT 2023. LNCS, vol. 14006, pp. 385–416. Springer, Cham (2023). https://doi.org/10.1007/978-3-031-30620-4_13

[WWW22] Waters, B., Wee, H., Wu, D.J.: Multi-authority ABE from lattices without random oracles. In: Kiltz, E., Vaikuntanathan, V. (eds.) TCC 2022. LNCS, vol. 13747, pp. 651–679. Springer, Cham (2022). https://doi.org/10.1007/978-3-031-22318-1_23

[ZZGQ23] Zhu, Z., Zhang, K., Gong, J., Qian, H.: Registered ABE via predicate encodings. In: Guo, J., Steinfeld, R. (eds.) ASIACRYPT 2023. LNCS, vol. 14442, pp. 66–97. Springer, Singapore (2023). https://doi.org/10.1007/978-981-99-8733-7_3

# Quantum II and Separations

# Unclonable Commitments and Proofs

Vipul Goyal[1,2](✉), Giulio Malavolta[3], and Justin Raizes[2]

[1] NTT Research, Sunnyvale, CA, USA
[2] Carnegie Mellon University, Pittsburgh, PA, USA
vipul@cmu.edu
[3] Bocconi University, Milan, Italy

**Abstract.** Non-malleable cryptography, proposed by Dolev, Dwork, and Naor (SICOMP '00), has numerous applications in protocol composition. In the context of proofs, it guarantees that an adversary who receives a proof cannot maul it into another valid proof. However, non-malleable cryptography (particularly in the non-interactive setting) suffers from an important limitation: An attacker can always copy the proof and resubmit it to another verifier (or even multiple verifiers).

In this work, we *prevent even the possibility of copying the proof as it is*, by relying on quantum information. We call the resulting primitive *unclonable proofs*, making progress on a question posed by Aaronson. We also consider the related notion of *unclonable commitments*. We introduce formal definitions of these primitives that model security in various settings of interest. We also provide a near tight characterization of the conditions under which these primitives are possible, including a rough equivalence between unclonable proofs and public-key quantum money.

## 1 Introduction

Non-malleable cryptography studies cryptographic primitives that (provably) cannot be mauled in a "controlled" way. For instance, non-malleable zero-knowledge [DDN00] considers protocols between a prover and a verifier, where the prover wants to convince the verifier of the validity of a certain NP-statement $x$. At the same time, non-malleability guarantees that the verifier cannot act as a *man-in-the-middle*, i.e., it cannot use the interaction with the prover to convince a *different* verifier (unless they were able to create their proof from scratch). However, traditional non-malleable cryptographic protocols (particularly the non-interactive ones) suffer from an important limitation: A man-in-the-middle can always *copy* the messages of the prover and forward them to the new verifier. This attack is clearly unavoidable, and it is typically ruled out by weakening the non-malleability guarantee (for instance, by only requiring non-malleability across different NP-statements, or by associating each participant with a *tag* and defining non-malleability to hold only across different tags). In this work, we ask the following question:

Can we construct proofs that (provably) cannot be cloned?

© International Association for Cryptologic Research 2025
E. Boyle and M. Mahmoody (Eds.): TCC 2024, LNCS 15366, pp. 193–224, 2025.
https://doi.org/10.1007/978-3-031-78020-2_7

While this question is clearly hopeless classically, quantum information behaves in a fundamentally different manner. The no-cloning theorem [WZ82] offers the tantalizing possibility that one may be able to overcome the above limitation leveraging quantum information. In fact, recent years have seen a surge of success in constructing cryptographic primitives with *unclonability* guarantees, such as quantum money [Wie83, Aar09, AC12, FGH+12, Zha21], unclonable encryption [BL20, AK21, AKL+22], signature tokens [BS17, CLLZ21], revocable decryption keys [KN22, AKN+23, APV23, BGG+23], unclonable pseudorandom functions [CLLZ21], to mention a few. However, to the best of our knowledge, the notion of unclonable proofs (and even the more basic notion of unclonable commitments) has not been formally studied. In fact, this same question was posed by Aaronson as open problem.[1] The purpose of this work is to make progress on this front.

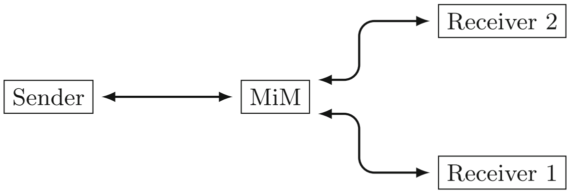

**Fig. 1.** The Commitment/Proof Cloning Experiment

*Unclonable Commitments and Proofs.* In this work we initiate the formal study of unclonable commitments and unclonable proofs.[2] More specifically, we consider a man-in-the-middle (MiM) interacting in a *left* session with an honest sender, and in two *right* sessions with two honest receivers. This is illustrated in Fig. 1. In these settings we require that:

– (Commitments) The message committed to in one of the right sessions is independent of the message committed to in the left session. In particular, consider two (non-communicating) distinguishers which each receive one of the messages that were committed to in the right sessions, then attempt to guess whether the left session is a commitment to $m_0$ or to $m_1$. We require that they cannot *simultaneously* succeed with negligible advantage over random guessing, i.e. with probability $1/2 + \text{negl}(\lambda)$.
– (Proofs: Simulation-Extractability) If the right receivers both accept statements $\widetilde{x}_1$ and $\widetilde{x}_2$, then MiM must know a witness for at least one of the two statements. This is formalized using a simulator-extractor which simultaneously simulates MiM's view *without* the witness for the left session and, if both right sessions accept, extracts a witness for either $\widetilde{x}_1$ and $\widetilde{x}_2$.

[1] https://scottaaronson.blog/?p=2903.
[2] We refer to both computationally and statistically sound protocols as proofs in this work.

- (Proofs: Simulation-Soundness) Even if $\mathsf{MiM}$ receives a simulated proof for any statement $x$ (potentially not in the language $\mathcal{L}$) in the left session, it cannot convince two honest verifiers to accept statements $\widetilde{x}_1 \notin \mathcal{L}$ and $\widetilde{x}_2 \notin \mathcal{L}$ in the right sessions.

We explicitly mention here that the notion of unclonability is in principle meaningful for proofs that do not satisfy any zero-knowledge guarantee. However, there must be a sense in which the witness is hidden, as otherwise the verifier could violate the above properties by simply recovering the witness and computing any number of honest proofs.

*Application: Unclonable Credentials.* As an interesting application, we mention that unclonable proofs/arguments can be used to convert any classical credential into an *unclonable credential.* Starting from any publicly-verifiable credential (such as a digital signature), one can simply commit to such a credential and produce an unclonable proof of the validity of such a credential. The resulting state can be then publicly verified and it can (provably) not be cloned. This can be used, for instance, to building uncloneable smartcards or tokens to grant access to a restricted area, or to log into a particular device. Contrary to existing unclonable smartcards, unclonable credentials provide provably secure guarantees against an adversary that can even tamper with the hardware.

## 1.1 Our Results

As the main contributions of this work, we lay the definitional groundwork for unclonable commitments and proofs, and we study under which computational assumptions we can build this primitives. We consider two settings: Same-protocol unclonability and strong unclonability.

*Same-Protocol Unclonability.* In these settings, we require that the above unclonability guarantees hold when the same verification protocol is performed in all three sessions. For commitments, we obtain three different constructions with tradeoffs between their assumptions and properties.

**Theorem 1 (Unclonable Commitments - Informal).** *We obtain the following constructions of unclonable commitments.*

- *Assuming public-key quantum money and non-interactive non-malleable commitments, there exist non-interactive unclonable commitments.*
- *Assuming post-quantum non-malleable commitments, there exist interactive post-quantum unclonable commitments.*
- *Assuming non-interactive post-quantum non-malleable commitments for one left session and four right sessions, there exist non-interactive unclonable commitments.*

The assumption of non-malleable commitments is somewhat minimal, since unclonability is a strengthening of non-malleability. On the other hand, we show that non-interactive unclonable proofs require (seemingly) stronger assumptions.

**Theorem 2 (Non-Interactive Unclonable Proofs - Informal).** *Assuming non-interactive zero knowledge and either one-way functions or public key encryption, non-interactive unclonable proofs exist* if and only if *public key quantum money exists.*

If we are willing to allow interaction, then we can construct an unclonable proof with only classical computation.

**Theorem 3 (Post-Quantum Unclonable Proofs - Informal).** *Assuming post-quantum one-way functions, there exist (interactive) post-quantum unclonable proofs.*

Additionally, we study the setting where the adversary receives $k$ commitments/proofs in the left sessions and attempts to clone them to produce $k + r$ commitments/proofs in the right sessions, which we call *many-many unclonability.* Our constructions also obtain many-many unclonability, if instantiated with a commitment/proof where non-malleability holds even for many left and right sessions (concurrent non-malleability).

**Theorem 4 (Many-Many Unclonability - Informal).** *Assuming concurrent non-malleable commitments (respectively, proofs), there exist many-many unclonable commitments (respectively, proofs).*

*Strong Unclonability.* In these settings, we require that the above unclonability guarantees hold for commitments/proofs with respect to *any verification procedure.* This is a strong notion of unclonability that effectively captures the intuition that the information vanishes after the verification is performed. Unfortunately, we show that unclonable proofs do not exist in these settings, assuming the existence of commitments. In these settings we show that unclonable proofs are impossible to achieve.

**Theorem 5 (Impossibility of Strongly Unclonable Proofs - Informal).** *Assuming classical commitments, there do not exist strongly unclonable proofs, even for interactive protocols.*

The proof of the impossibility result makes use of a secure multiparty computation protocol to construct an explicit attack. To the best of our knowledge, this is the first time that techniques from secure multiparty computation have been used to analyze unclonability.

We complement the impossibility result by mentioning several possible modifications to the model that may allow sidestepping the impossibility. We explore one of them to construct strongly unclonable commitments in the quantum random oracle model (QROM), and leave the rest as open problems.

**Theorem 6 (Strongly Unclonable Commitments - Informal).** *In the QROM, there unconditionally exist strongly unclonable commitments against right protocols which are statistically binding.*

## 2 Technical Overview

Since unclonability is closely related to non-malleability, we begin by recalling this notion and discussing its relation with unclonability. In a non-malleability experiment, an adversarial MiM interacts with a sender in a left session and with a receiver in the right session. This is similar to the scenario illustrated in Fig. 1, except that there is only one right receiver. Since it is impossible to prevent the MiM from directly forwarding the messages between the two sessions, non-malleability is commonly defined with respect to identities, or tags, to force differences between the two sessions. If the MiM uses a different tag in the right session than in the left session, then we may require one of the following guarantees:

- (Commitments [DDN00, PR05b]) The message committed to in the right session is independent of the message committed to in the left session. In particular, the joint view of the MiM and the value committed to in the right session cannot be used to distinguish whether the left session is a commitment to $m_0$ or to $m_1$.
- (Proofs: Simulation-Extraction [PR05b]) The MiM must know a witness for the statement $\widetilde{x}$ in the right session. This is formalized using a simulator-extractor which simultaneously simulates MiM's view *without* the witness for the left session and, if the right session accepts, extracts a witness for $\widetilde{x}$.
- (Proofs: Simulation-Soundness [Sah99]) Even if the MiM receives a simulated proof for any statement $x$ (potentially not in the language $\mathcal{L}$) in the left session, it cannot convince an honest verifier to accept a false statement $\widetilde{x} \notin \mathcal{L}$ in the right session.

We emphasize that these guarantees *only* hold if the left and right sessions use different tags.

### 2.1 Definitions: Unclonable Commitments

Unclonable commitments remove the different-tag restriction by considering *two* right sessions, instead of one. Although the MiM can still forward one of the right sessions, we require that the messages $\widetilde{m_1}$ and $\widetilde{m_2}$ committed to in the right sessions cannot simultaneously be dependent on the message $m$ committed to in the left session. We formalize this by requiring that no adversary wins the following game with probability greater than $1/2 + \text{negl}$.

1. The challenger commits to either $m_0$ or $m_1$ in the left session, and the MiM commits in the right sessions. Afterwards, the MiM splits its internal state into two registers $\mathsf{MiM}_1$ and $\mathsf{MiM}_2$.
2. Two non-communicating distinguishers $D_1$ and $D_2$ each receive a residual register $\mathsf{MiM}_1$ (respectively $\mathsf{MiM}_2$) and the value $\widetilde{m}_1$ (respectively, $\widetilde{m}_2$) of the commitment in right session one (respectively, two). The adversary wins if both distinguishers correctly guess which message the challenger committed to.

To ensure that $\widetilde{m_1}$ and $\widetilde{m_2}$ are well-defined, we define unclonability for statistically binding commitments. This allows us to define the value of a commitment using a computationally-*un*bounded extractor that acts on the receiver's state. The case of computationally bounded unclonable commitments is less straightforward. Even in the less demanding setting of classical non-malleability, definitions with respect to computational binding are nuanced since it is difficult to define a meaningful "value committed in the right session". Nevertheless, our paradigm can in principle be applied to any notion of non-malleability for computational binding, such as non-malleability with respect to opening [DIO98] or non-malleability with respect to replacement [Goy11], simply by using the corresponding notion of a commitment's value. For example, non-malleability with respect to opening defines a commitment's value to be the receiver's output at the end of the opening phase.

*Relation to Unclonable Encryption.* The definition of unclonable commitments has many similarities to unclonable encryption [BL20], but has a few key differences. In unclonable encryption, the adversary splits a ciphertext $\mathsf{Enc}(k, m_b)$ into two registers $\mathcal{B}$ and $\mathcal{C}$. Then, adversaries $B$ and $C$ are given $\mathcal{B}$ and $\mathcal{C}$, respectively, along with the secret key $k$, and try to guess the bit $b$ simultaneously without communicating. Philosophically, unclonable encryption captures the scenario where an eavesdropping adversary wants to collect information about a ciphertext in the hopes of later learning the key.

In contrast, unclonable commitments philosophically captures a scenario where an eavesdropping adversary not only wishes to leak information about the committed message, but to actually make use of the leaked information before it is revealed. Because the adversary is attempting a more difficult task, we can hope to construct unclonable commitments from weaker assumptions. Indeed, indistinguishable-unclonable encryption is currently only known in the random oracle model [AKL+22]. In contrast, we show that when the same protocol is used in all three sessions, it is possible to construct non-interactive unclonable commitments from any non-interactive non-malleable commitment (see Sect. 4.2). In this sense, unclonable commitments are weaker than unclonable encryption.

However, the two notions are technically incomparable. We note that unclonable encryption may in principle be unconditionally possible, whereas unclonable commitments necessarily require computational assumptions. Even with computational assumptions, it is not clear how to transform an unclonable encryption scheme into an unclonable commitment. Indeed, existing techniques for adding computational assumptions to unclonable encryption (e.g. to create unclonable public-key encryption) make crucial use of a "fake-key" property that allows the encrypter to open a ciphertext to any message it wants [AK21]. This technique is fundamentally incompatible with the binding requirement of a commitment scheme.

*Relation to* [BC23]. Recently, Broadbent and Culf proposed a different definition of unclonable commitments. In their definition, the committer Alice interacts with a receiver Bob, while an eavesdropper Eve sees (and can interfere with) their messages. After the commitment phase, Alice and Bob run a check phase. During the opening phase, which happens after the check phase, Bob and Eve cannot directly communicate. Their definition guarantees that if Alice accepts in the check phase, then at the end of the opening phase Eve has no information about Alice's message. Here, Eve plays a similar role to the one that MiM plays in our definition. There are several core differences between our definitions.

- The first difference is that their definition takes place in the interactive setting due to the extra check phase, whereas our definition also makes sense for non-interactive settings.
- The second is that their definition is for statistically hiding (and hence computationally binding) commitments, whereas ours is for computationally hiding, statistically binding commitments.
- A third difference is that their scheme provides no guarantees if Alice rejects during the check phase, or if Eve can communicate with Bob during the opening phase. In this case, it is possible that Eve and Bob both hold information about Alice's message. In contrast, our guarantee always holds (even if the left session aborts during one of the two phases).

To exemplify the difference between the two approaches, consider the following toy scenario: Alice has found a proof that P=NP and wants to commit to the proof to the Clay institute (Bob), to later claim the 1M$ prize. Note that our constructions allow Alice to make a non-interactive commitment. Eve intercepts the commitment, and she tries to clone it and submit as her own commitment to Bob (who is distributing the prize). Our definition prevents this attack since Eve's commitment is not allowed to depend on that of Alice. On the other hand, Broadbent and Culf's definition does not necessarily defend against this attack: This is because their definition only applies if Eve and Bob do not communicate at all during the opening phase. In this example, Eve can intercept Alice's opening, create her own opening based on that, and submit to Bob.

## 2.2 Definitions: Unclonable Proofs

We now turn our attention to defining unclonable proofs. Existing notions of unclonability, such as unclonable encryption [BL20] or even copy-protection [Aar09] attempt to prevent an adversary in possession of a cloned state from learning new information. *However, these notions do not make sense in the context of proofs, since it is always possible to generate a fresh proof using the witness.* To fill this gap, we present two incomparable notions of unclonable proofs which are inspired by the non-malleable proof literature.

*Extraction-Unclonability.* The first notion aims to realize the intuition that the *only* way for the MiM to generate two accepting proofs is to generate one of them using the witness. In other words, the MiM must "know" at least one witness. In the classical setting of non-malleability, this is formalized by the notion of simulation-extraction.

In the setting of unclonability, we consider a cloning experiment where the MiM receives a proof of a statement $x$ in the left session and sends proofs in two right sessions (see Fig. 1). This experiment outputs MiM's final view, as well as the statements $\widetilde{x}_1$ and $\widetilde{x}_2$ in the two right sessions and whether the verifiers accept in those sessions. Extraction-unclonability requires the existence of a simulator $\mathcal{S}(x)$ which receives as input the statement $x$ to be proved in the left session (but not a witness) and outputs two things. First, it must output a view $\tau$ which is indistinguishable from adversary's view at the end of the real cloning experiment. Second, it must output two potential witnesses $\widetilde{w}_1$ and $\widetilde{w}_2$. If the verifiers accept in both right sessions of $\tau$, then at least one of $\widetilde{w}_1$ and $\widetilde{w}_2$ must be a valid witness for $\widetilde{x}_1$ and $\widetilde{x}_2$, respectively. We note that since witness relations are efficiently checkable, it is possible to efficiently determine which of the two right sessions extraction succeeded in, and which session was potentially forwarded.

*Soundness-Unclonability.* A more nuanced adversary might attempt to directly break soundness of both right sessions. Although extraction-unclonability prevents this by extracting at one least valid witness if the left session proves a true statement $x \in \mathcal{L}$, it does not provide any guarantees if the left session proves a *false* statement $x \notin \mathcal{L}$. This nuance is reminiscent of the difference between simulation-extraction and simulation-soundness in the setting of classical non-malleability, which [JP14] shows are incomparable.

In fact, this scenario arises naturally in cryptographic proofs. Consider a "paired commitment" where the left prover commits twice to the same value $v$ and proves in zero knowledge that the two commitments are consistent. A simple strategy to prove security of this paired commitment is to use a hybrid argument where first we simulate the zero-knowledge proof, then switch the commitments one at a time to a new value $v'$. Observe that in the middle hybrid, the left commitments are to *different* values $v$ and $v'$, yet the zero-knowledge proof is still being simulated. If the receiver (the MiM) were to engage as a prover in a second proof during this hybrid, then the results could be catastrophic, since there would be no guarantee of soundness. Thus, we could not simultaneously rely on the hiding of the paired commitment and the soundness of a second proof.

To address this issue, we extend simulation-soundness to the setting of unclonability. Consider the same proof cloning experiment as for extraction-unclonability. Simulation-soundness requires the existence of a simulator $\mathcal{S}(x)$ which receives as input the statement $x$ to be proved in the left session (but not a witness), then outputs a view $\tau$ which is indistinguishable from the adversary's view at the end of the proof cloning experiment. Furthermore, if $\widetilde{x}_1 \notin \mathcal{L}$ and $\widetilde{x}_2 \notin \mathcal{L}$ in $\tau$, then at most one of the receivers accept in the right sessions of $\tau$.

*Application: Unclonable Credentials.* We expand on the previously mentioned application of unclonable credentials and briefly discuss why both extraction-unclonability and soundness-unclonability suffice for it. Recall that to construct an unclonable credential (say for an employee's badge), one can classically commit to a signature of the badge number $b$, then produce an unclonable proof that the commitment contains a valid signature of $b$.

If the proof were to satisfy extraction-unclonability, then any copier would be able to produce a valid signature without receiving one in the clear. This immediately violates the unforgeability of the signature scheme.

On the other hand, if the proof were to satisfy soundness-unclonability, then we could consider a series of hybrid experiments where $b$ becomes an invalid credential in general. First, the unclonable proof is simulated. Second, the badge holds a commitment to $\perp$, instead of a commitment to a signature on $b$. Finally, the signing key is "punctured", so that no valid signature on $b$ exists [BSW16]. In the final hybrid, any proof of a signature for $b$ must be breaking soundness, so soundness-unclonability guarantees that the adversary cannot create two accepting badges for $b$. Since this experiment is indistinguishable from the real world, it also cannot create an acceptable second badge for $b$ in the real world.

## 2.3 Same-Protocol Unclonability

In same-protocol unclonability, we consider the case where the right sessions use the same commitment/proof protocol as the left session. For example, this scenario could be useful when many users are interacting with the same central entity, who uses a standardized cryptographic suite.

*Unclonable Tag-Generation.* A non-malleable commitment has similar security to an unclonable one, except that its security guarantee only holds when the right and left sessions use different tags. Thus, if there was a way to guarantee that one of the two right sessions uses a different tag than the left session, we would be able to rely on non-malleability in that session.

This suggests a natural primitive which we call *unclonable tag-generation*. In a tag-generation protocol, a sender and a receiver interact to agree on a string $\mathsf{tag}$. Unclonability is defined using a game where an adversary $\mathsf{MiM}$ participates in three simultaneous sessions: a left session where it acts as the receiver, and two right sessions where it acts as the sender. Unclonability guarantees that at least one of the two right sessions outputs a different tag than the left session, unless all three sessions output a failure symbol $\perp$.

Armed with this primitive, there is a simple construction of unclonable commitments or unclonable proofs. First, agree on $\mathsf{tag}$ using an unclonable tag-generation protocol. Then, execute a non-malleable commitment using $\mathsf{tag}$. If both the tag-generation protocol and the non-malleable commitment are non-interactive, then the two messages can be sent in parallel. This yields a non-interactive unclonable commitment. Unclonable proofs can be constructed similarly from non-malleable proofs.

*Constructing Unclonable Tag-Generation.* A natural idea for unclonable tag-generation is to use the strong unclonability properties of public key quantum money (PKQM). This yields a simple, non-interactive protocol. The sender sends a serial number, banknote pair $(s, |\$_s\rangle)$ to the receiver. The tag is $s$. The verifier checks that the banknote matches $s$ before outputting $s$. Since no $\mathsf{MiM}$ can create two copies of $|\$_s\rangle$, it can only use $s$ as a tag in a single right session. Unfortunately, public key quantum money is currently only known from strong or poorly-understood assumptions such as indistinguishability obfuscation [AC12, FGH+12, Zha21].

It is immediate to see that in the interactive setting, a *classical* solution is possible. The key observation is that two different receivers may act differently, as long as they send at least one message. First, the receiver sends a uniformly random message $r$. The sender samples a signature key pair $(\mathsf{sk}, \mathsf{vk})$, then signs $r$. It sends $\mathsf{vk}$ and the signature back to the receiver, who verifies the signature. The tag is the verification key $\mathsf{vk}$. Notice that in the unclonability experiment, the left sender will only sign a single $r$. However, the two right sessions will use different $r$'s with high probability. Thus, in order to break unclonability, the adversary $\mathsf{MiM}$ must forge one of the two signatures. By combining this idea with a post-quantum non-malleable commitment/proof (e.g. [ABG+21, BLS22, LPY23]), it is even possible to construct a post-quantum unclonable commitment/proof. The downside of this approach is that it is interactive, which precludes constructing non-interactive unclonable primitives.

As a result of these ideas, we obtain two constructions of both unclonable commitments and unclonable proofs. One construction is non-interactive, but requires public key quantum money. The other is post-quantum, but requires two rounds of communication.

*Non-interactive Unclonable Commitments without PKQM.* In the case of commitments, it is possible to generate the tags in a more flexible way. Since a commitment has both a commitment phase and an opening phase, the tag can be generated in the commitment phase, then confirmed in the opening phase. Unless the adversary can confirm the same tag in both opening phases, it is unable to clone a commitment. This delayed confirmation allows us to use techniques from private-key quantum money, which exists unconditionally [Wie83].

This idea yields our third construction of unclonable commitments, which is non-interactive and can be based only on post-quantum non-interactive non-malleable commitments. As mentioned previously, this is an almost minimal assumption, since unclonability is a strengthening of non-malleability. To generate the tag, the committer samples a random Wiesner state $|x\rangle_\theta$, then commits to $(x, \theta)$ using a post-quantum non-interactive non-malleable commitment scheme with tag 0. Call this commitment $\mathsf{com}_{\mathsf{tag}}$. Then, to commit to a message $m$, the committer computes a non-interactive non-malleable commitment to $m$, using $\mathsf{com}_{\mathsf{tag}}$ as the tag. To open the commitment, the committer opens both non-malleable commitments and the verifier checks that $\mathsf{com}_{\mathsf{tag}}$ matches the Wiesner state.

Since the tag $\mathsf{com}_{\mathsf{tag}}$ is statistically binding, it information-theoretically determines a Wiesner state. Thus, in order to use the same tag in all three sessions, the adversarial MiM must clone a Wiesner state *before* its description is revealed. Otherwise, one of the right sessions will contain an invalid commitment. On the other hand, if the adversary uses a different tag in one of the right sessions, then non-malleability guarantees that the value committed inside that session is independent of the value committed in the left session.

However, this intuitive proof fails against a more subtle attack. An adversary may attempt to selectively generate invalid commitments according to the left message $m$. To do this, it generates an independent Wiesner state $|\widetilde{x}\rangle_{\widetilde{\theta}}$. Then, it mauls the commitment to $m$ so that its right tag commitment $\widetilde{\mathsf{com}}_{\mathsf{tag}}$ matches the Wiesner state if and only if $m$ is a pre-determined message. Crucially, the non-malleability of $\mathsf{com}_{\mathsf{tag}}$ prevents this type of attack. Since $\widetilde{\mathsf{com}}_{\mathsf{tag}}$ must use a different tag (0) from the commitment to $m$, the value committed inside it must be independent of $m$. Combining this with the intuitive proof from before shows that in at least one session, the adversary must either make an invalid commitment or commit to a value which is independent of $m$, and that which case happens must also be independent of $m$. Therefore this session is independent of $m$ in general.

*Unclonable NIZKs Imply PKQM.* The reader may observe that the two ideas for unclonable commitments from weaker assumptions both made use of multiple messages to confirm the tag. In our third construction of commitments, we make use of the opening phase to delay the confirmation, giving non-interactive unclonable commitments. However, some primitives, such as proofs, only consist of a single phase. Given this, it is natural to wonder whether non-interactive unclonability for such primitives inherently require public key quantum money. We show that in the case of unclonable non-interactive zero knowledge (NIZKs), the answer is yes; unclonable NIZKs do imply public key quantum money, assuming one-way functions. When combining this with the previous construction of unclonable NIZKs from public key quantum money, we arrive at a loose equivalence between the two.

To construct public key quantum money, we use unclonable NIZKs and commitments. To mint a note, the bank first samples a common reference string (CRS) for the NIZK. Since the bank's only goal is to prevent cloning of banknotes, it may be trusted to do this honestly. Then, it generates a commitment $\mathsf{com}_0$ to 0, which will act as the serial number. Finally, it proves using the unclonable NIZK that $\mathsf{com}_0$ is a commitment to 0. Intuitively, the unclonability of the NIZK guarantees that any adversary which produces two banknotes must be able to find a witness for $\mathsf{com}_0$ being a commitment to 0, just given $\mathsf{com}_0$. However, this would violate the security of the commitment! It is also possible to show the security of the scheme using the related notion of NIZK soundness-unclonability, which guarantees that the adversary cannot convince both right verifiers to accept a statement $\widetilde{x} \notin \mathcal{L}$, even if it receives a simulated proof for a statement $x$ (potentially not in $\mathcal{L}$). We defer the details to the technical section.

## 2.4 Strong Unclonability

It is also interesting to consider the setting where the right protocols may be *different* from the left protocol. For example, this scenario could be useful when a user wishes to place sealed bids in auctions from multiple auction houses, who might each prefer their own cryptographic libraries.

*Negative Results.* It is simple to see that strongly unclonable NIZKs are not possible; an adversary can simply ask the verifier to return the NIZK after they are done, then reuse the returned NIZK. We further show that even *interactive* strongly unclonable proofs are not possible.

Intuitively, an adversary can attack a candidate proof by engaging in a single session of a secure multiparty computation (MPC) protocol (for quantum functionalities) with the two verifiers. The MPC protocol emulates a single verifier for the proof. This ideal verifier will always accept the (honest) proof from the left session, so both right verifiers will also accept. We emphasize that the verifiers do not have a direct channel between them, so they must send messages to each other via the adversary. Since the adversary controls the channel between the two verifiers, the MPC protocol must be secure against a dishonest majority in order to guarantee that the right protocols are sound.

One can also imagine a similar attack for commitments. However, we only define unclonable commitments with respect to statistical binding. In order for the right protocols, which are defined using the MPC, to be statistically binding, the MPC must guarantee the correct output even when an *unbounded* adversary controls the majority of the parties.

*Positive Results: Commitments.* One option for avoiding the impossibility is to use a random oracle (the QROM model). In a dishonest majority setting, an MPC cannot hope to query the QROM, preventing the attack. We show how to construct a strongly unclonable commitment scheme in the QROM model, where unclonability holds if the two right protocols are statistically binding.

To commit to a message $m$, encrypt $m$ using an unclonable encryption scheme, then commit to the encryption key $k$ using the random oracle. Unclonable encryption can be constructed in the QROM [AKL+22]. As long as $k$ is statistically bound, there is only one possible decryption of the unclonable ciphertext, giving statistical binding. Crucially, random oracles allow commitments which are *both* statistically binding and statistically hiding. The former property allows statistical binding. The latter property ensures $k$ can be removed from the left commitment without affecting the values committed to in the right commitments. Since the right commitments must be statistically binding, they each uniquely determine a value $\widetilde{m}_r$. However, if $\widetilde{m}_1$ and $\widetilde{m}_2$ simultaneously depended on $m$, then this would violate the security of the underlying unclonable encryption scheme.

## 2.5 Concurrent Work

In an independent and concurrent work, Jawale and Khurana [JK23] also introduced a notion of unclonable NIZKs and showed that it is roughly equivalent to public-key quantum money. Their notion of *unclonable extractability* is very similar to our notion of *extraction-unclonability*, which supports it as a natural definition of unclonability. They also apply unclonable NIZKs to construct unclonable signatures of knowledge, which we do not consider. On the other hand, we additionally define and construct unclonable commitments, extend our definitions to the interactive case, and investigate the possibility of achieving security against verifiers who do not follow the honest verification procedure (strong unclonability).

### 2.6 Paper Organization

We define and construct unclonable tag-generation in Sect. 3. We define unclonable commitments in Sect. 4.1 and construct them in Sect. 4.2. We define unclonable proofs in Sect. 5.1 and construct them in Sect. 5.2. We prove their equivalence to public key quantum money in Sect. 5.3. We present our negative results for strong unclonability in Sect. 6 and the positive results in Sect. 7.

## 3 Unclonable Tag-Generation

In order to construct unclonable primitives, we start by constructing a primitive called unclonable tag-generation protocols. Informally, a tag-generation protocol guarantees that no adversary can receive a generated tag in a left session, then force the same tag in two other sessions of the same protocol. This is a useful property to combine with non-malleable primitives, which only guarantee security if the tags in the left and right sessions are different. Together, they give a very natural construction of unclonable primitives: determine the tag using an unclonable tag-generation protocol, then run the corresponding non-malleable primitive.

### 3.1 Definition

*Tag-Generation Protocols.* A tag-generation protocol is a (potentially interactive) protocol between a sender and a receiver. At the end of the protocol, the sender and the receiver output the same string $\mathsf{tag}$.

*Tag Cloning Game.* The tag cloning game $\mathsf{Cl\text{-}TagGen}(1^\lambda)$ uses a tag-generation protocol $\mathsf{TagGen}$. It involves three session: a "left" session between an honest sender $S_L$ and a QPT adversarial man-in-the-middle $\mathsf{MiM}$, and two right sessions between $\mathsf{MiM}$ and an honest right receiver, respectively named $R_1$ and $R_2$. The sessions may be interleaved arbitrarily. The game is played as follows:

1. **Tag-Generation:** Each session runs an instance of $\mathsf{TagGen}$.
2. **Output:** Let $\mathsf{tag}$ be $S_L$'s output, let $\widetilde{\mathsf{tag}}_1$ be $R_1$'s output, and let $\widetilde{\mathsf{tag}}_2$ be $R_2$'s output. The adversary wins (output 1) if $\mathsf{tag} = \widetilde{\mathsf{tag}}_1 = \widetilde{\mathsf{tag}}_2 \neq \bot$. Otherwise the game outputs 0.

**Definition 1 (Unclonability for Tag-Generation).** *A tag-generation scheme* $\Pi$ *is* unclonable *if for all QPT* $\mathsf{MiM}$*,*

$$\Pr[\mathsf{Cl\text{-}TagGen}(1^\lambda) = 1] \leq \mathrm{negl}$$

## 3.2 Constructions

*A Classical Interactive Protocol.* We observe that even classically, two different receivers may act differently. Thus, we may use interaction to realize a classical unclonable tag-generation protocol.

1. The receiver samples a uniform bitstring $r \leftarrow \{0,1\}^{\lambda}$, then sends it to the sender.
2. The sender samples a digital signature key pair $(\mathsf{sk}, \mathsf{vk})$. It computes $\sigma \leftarrow \mathsf{Sign}(\mathsf{sk}, r)$, then sends $(\mathsf{vk}, \sigma)$ to the receiver.
3. **Output:** The sender outputs $\mathsf{vk}$. If $\mathsf{Verify}(\mathsf{vk}, \sigma, r)$ accepts, the receiver outputs $\mathsf{vk}$. Otherwise it outputs $\perp$.

**Fig. 2.** Post-Quantum Unclonable Tag-Generation

**Lemma 1 (Interactive Classical Unclonable Tag-Generation).** *Assuming post-quantum one-way functions, there exists a two-round, post-quantum tag-generation protocol which is unclonable.*

*Proof.* The construction is given in Fig. 2. Note that post-quantum digital signatures can be constructed from post-quantum one-way functions. Say that $\mathsf{MiM}$ violated unclonability, and consider an execution where $\mathsf{tag} = \widetilde{\mathsf{tag}}_1 = \widetilde{\mathsf{tag}}_2 \neq \perp$. By assumption, such executions occur with noticeable probability. Since no tag is equal to $\perp$, it must be the case that $\mathsf{Verify}(\widetilde{\mathsf{tag}}_1, \widetilde{\sigma}_1, \widetilde{r}_1)$ and $\mathsf{Verify}(\widetilde{\mathsf{tag}}_2, \widetilde{\sigma}_2, \widetilde{r}_2)$ both accepted. Furthermore, $\widetilde{r}_1 \neq \widetilde{r}_2$ with probability $1 - 2^{-\lambda}$ over the randomness of $R_1$ and $R_2$. Therefore either $r \neq \widetilde{r}_1$ or $r \neq \widetilde{r}_2$. Without loss of generality, say it is $\widetilde{r}_i$. Thus, given $\mathsf{MiM}$, we could break the unforgeability of digital signatures by querying the unforgeability challenger on $r$, using the returned signature $\sigma$ to finish the unclonability game, and outputting $\widetilde{\sigma}_i$.

Unfortunately, using an interactive tag-generation protocol to build an unclonable protocol results in an interactive protocol. In order to construct non-interactive unclonable primitives, we will need a non-interactive tag-generation protocol.

*Unclonable Tag-Generation From Public Key Quantum Money.* Public key quantum money scheme has a very strong unclonability property. This gives a very simple and natural unclonable tag-generation protocol.

**Lemma 2 (Unclonable Tag-Generation from PKQM).** *Assuming public key quantum money, there exists a non-interactive unclonable tag-generation protocol.*

1. The sender generates a banknote $|\$\rangle$ with serial number $s$. It sends both to the receiver.
2. **Output:** The sender outputs $s$. If $\mathsf{Verify}(|\$\rangle, s)$ accepts, the receiver outputs $s$. Otherwise it outputs $\perp$.

**Fig. 3.** Non-Interactive Unclonable Tag-Generation from Public Key Quantum Money

*Proof.* The construction is given in Fig. 3. Say that some $\mathsf{MiM}$ violated unclonability. Then with noticeable probability, $s = \widetilde{s}_1 = \widetilde{s}_2$ and both $|\widetilde{\$_1}\rangle$ and $|\widetilde{\$_2}\rangle$ are valid banknotes for $s$. This is a direct contradiction of the security of public key quantum money.

## 4 Unclonable Commitments

### 4.1 Definitions

Informally, an unclonable commitment guarantees that no man-in-the-middle can receive a commitment to $m$, then commit to related messages in two other sessions of the same commitment protocol. This is very similar to the guarantee provided by non-malleable commitments. The key difference is that an unclonable commitment does not require that the committers in the three sessions use distinct identities, or even that they send different messages. As a direct consequence, *unclonability is meaningful even if the adversary attempts to directly forward the left session.*

*Commitment Cloning Game.* The commitment cloning game $\mathsf{Cl\text{-}Com}(1^\lambda, m_0, m_1)$ uses a statistically binding commitment protocol $(\mathsf{Com}, \mathsf{Open})$. It involves three session: a "left" session between an honest committer $C_L$ and an adversarial man-in-the-middle $\mathsf{MiM}$, and two right sessions between $\mathsf{MiM}$ and an honest right receiver, respectively named $R_1$ and $R_2$. The sessions may be interleaved arbitrarily. Additionally, it uses two QPT distinguishers $D_1$ and $D_2$. The game is played as follows:

1. **Commitment:** In the left session, $C_L$ samples a random bit $b$, then commits to $m_b$ using $\mathsf{Com}$. Simultaneously in the right sessions, $\mathsf{MiM}$ interacts with $R_1$ and $R_2$ as the committer in two executions of $\mathsf{Com}$. At the end of the execution, $\mathsf{MiM}$ splits its internal state into two registers $\mathsf{MiM}_1$ and $\mathsf{MiM}_2$.
2. **Output:** Let $\mathsf{Ext}$ be the statistical binding extractor for $(\mathsf{Com}, \mathsf{Open})$. Let $\mathcal{R}_1$ and $\mathcal{R}_2$ be the internal registers of $R_1$ and $R_2$, respectively. Compute $(\mathcal{R}_1, \widetilde{\mathcal{M}}_1) \leftarrow \mathsf{Ext}(\mathcal{R}_1)$ and $(\mathcal{R}_2, \widetilde{\mathcal{M}}_2) \leftarrow \mathsf{Ext}(\mathcal{R}_2)$, where registers $\widetilde{\mathcal{M}}_1$ and $\widetilde{\mathcal{M}}_2$ contains the extracted messages for right session one and two, respectively. Compute $D_1(\mathsf{MiM}_1, \widetilde{\mathcal{M}}_1)$ and $D_2(\mathsf{MiM}_2, \widetilde{\mathcal{M}}_2)$, then measure their respective output bits $b_1$ and $b_2$. The adversary wins (output 1) if $b_1 = b_2 = b$, and otherwise the game outputs 0.

**Definition 2 (Unclonability for Commitments).** *A commitment scheme* $(\mathsf{Com}, \mathsf{Open})$ *is* unclonable *if for all (non-uniform) QPT adversaries* $\mathsf{MiM}$*, all (non-uniform) QPT distinguisher pairs* $(D_1, D_2)$ *and all message pairs* $(m_0, m_1) \in \{0,1\}^{\ell}$*,*

$$\Pr[\mathsf{Cl\text{-}Com}(1^{\lambda}, m_0, m_1) = 1] \leq \frac{1}{2} + \mathrm{negl}(\lambda)$$

It is also interesting to consider a stronger notion where the right sessions may use different (statistically binding) commitment schemes than the left session. The corresponding game $\mathsf{Cl\text{-}Com}_{\mathsf{strong}}(1^{\lambda}, m_0, m_1)$ takes the same form as $G(1^{\lambda}, m_0, m_1)$, except the right sessions use commitment schemes $(\widetilde{\mathsf{Com}}_1, \widetilde{\mathsf{Open}}_1)$ and $(\widetilde{\mathsf{Com}}_2, \widetilde{\mathsf{Open}}_2)$. These right commitments must satisfy statistical binding.

**Definition 3 (Strong Unclonability).** *A commitment scheme* $(\mathsf{Com}_L, \mathsf{Open}_L)$ *is* strongly unclonable *if for all right commitment schemes* $(\widetilde{\mathsf{Com}}_1, \widetilde{\mathsf{Open}}_1)$ *and* $(\widetilde{\mathsf{Com}}_2, \widetilde{\mathsf{Open}}_2)$*, every non-uniform QPT adversary* $\mathsf{MiM}$*, every (non-uniform) QPT distinguisher pair* $(D_1, D_2)$ *and all message pairs* $(m_0, m_1) \in \{0,1\}^n$*,*

$$\Pr[\mathsf{Cl\text{-}Com}_{\mathsf{strong}}(1^{\lambda}, m_0, m_1) = 1] \leq \frac{1}{2} + \mathrm{negl}(\lambda)$$

Since strong unclonability is a property about the composition of *different* protocols, it is much more difficult to achieve. We examine this notion further in Sect. 7.

*Many-Many Unclonability.* A natural extension is to consider an adversary which receives commitments in $k$ left sessions and attempts to produce $k+r$ commitments to related values in the right sessions. Similar settings have been previously studied in the context of program-copy protection [Aar09, LLQZ22]. In the case of commitments, we additionally aim to generalize the notion of concurrent non-malleable commitments [DDN00, PR05a]. In concurrent non-malleable commitments, the *joint* values of all right commitments whose tags differ from the left commitments must be independent of the *joint* values of all left commitments. Intuitively, many-many unclonability guarantees that if there are $k$ left sessions, then at most $k$ right sessions are correlated with them; the others must be independent. The many-many commitment cloning game $\mathsf{Cl\text{-}Com}_{k,r}(1^{\lambda}, \boldsymbol{m}_0, \boldsymbol{m}_1)$ for $k$ left sessions and $k+r$ right sessions is played as follows:

1. **Commitment:** In the left sessions, the challenger samples a random bit $b$, then commits to $\boldsymbol{m}_{b,i}$ for each $i \in [k]$. Simultaneously in the right sessions, $\mathsf{MiM}$ interacts as the committer with the $k+r$ honest receivers. At the end of this phase, $\mathsf{MiM}$ chooses a partition $P = (P_1, \ldots, P_{k+1})$ of $[k+r]$ and splits its state into $k+1$ registers $\mathsf{MiM}_1, \ldots, \mathsf{MiM}_{k+1}$.
2. **Output:** For every $i \in [k+1]$, compute $(\mathcal{R}_i, \widetilde{\mathcal{M}}_i) \leftarrow \mathsf{Ext}(\mathcal{R}_i)$, then compute $D_i(\mathsf{MiM}_i, \widetilde{\mathcal{M}}_i)$ and measure its output bit $b_i$. The adversary wins (output 1) if $b_i = b$ for every $i \in [k+1]$. Otherwise the experiment outputs 0.

**Definition 4 (Many-Many Unclonability for Commitments).** *A commitment scheme* $(\mathsf{Com}, \mathsf{Open})$ *is* many-many unclonable *if for all* $k = \text{poly}(\lambda)$, $r = \text{poly}(\lambda)$, *all (non-uniform) QPT adversaries* $\mathsf{MiM}$, *all (non-uniform) QPT distinguishers* $D_1, \ldots, D_{k+1}$, *and all message vectors* $(\boldsymbol{m}_0, \boldsymbol{m}_1) \in \{0,1\}^{nk}$,

$$\Pr[\mathsf{Cl\text{-}Com}_{k,r}(1^\lambda, \boldsymbol{m}_0, \boldsymbol{m}_1) = 1] \leq \frac{1}{2} + \text{negl}(\lambda)$$

## 4.2 Constructions

A simple construction for unclonable commitments is to use an unclonable tag-generation protocol to generate a tag for a non-malleable commitment. This guarantees that one of the two right sessions uses a different tag than the left session. In this case, we can rely on the security of the non-malleable commitment.

*Tools.* The construction uses the following tools.
- A non-malleable commitment $(\mathsf{NMCom}, \mathsf{NMOpen})$ with respect to commitment.
- An unclonable tag-generation protocol $\mathsf{TagGen}$.

$\underline{\mathsf{UCom}(1^\lambda, m)}$*:* The committer and receiver do the following:

1. Compute a tag $\mathsf{tag} \leftarrow \mathsf{TagGen}$. If $\mathsf{tag} = \bot$, abort.
2. Run the non-malleable commitment $\langle \mathsf{NMCom}_C(m), \mathsf{NMCom}_R \rangle(\mathsf{tag})$.
   If the same party sends the last message of $\mathsf{TagGen}$ and the first message of $\langle \mathsf{NMCom}_C(m), \mathsf{NMCom}_R \rangle(\mathsf{tag})$, they may send these messages at the same time.

$\underline{\mathsf{UOpen}(1^\lambda)}$*:* Run the opening phase of the non-malleable commitment $\langle \mathsf{NMOpen}_C, \mathsf{NMOpen}_R \rangle(\mathsf{tag})$.

**Fig. 4.** An Unclonable Commitment

**Theorem 7.** *Assuming* $n_1$*-round non-malleable commitments and* $n_2$*-round unclonable tag-generation protocols, there exists an unclonable commitment with* $n_1 + n_2$ *or* $n_1 + n_2$ *or* $n_1 + n_2 - 1$ *rounds.*

*Proof.* The construction is given in Fig. 4. Hiding is immediate from the underlying non-malleable commitment. Binding and hiding follow directly from $\mathsf{NMCom}$.

We show unclonability by reducing to the non-malleability of $(\mathsf{NMCom}, \mathsf{NMOpen})$. The non-malleability adversary $\mathsf{MiM}_{\mathsf{nm}}$ internally runs the cloning adversary $\mathsf{MiM}_U$. It internally runs $\mathsf{TagGen}$ in all three sessions to determine the tags $\mathsf{tag}$, $\widetilde{\mathsf{tag}}_1$, and $\widetilde{\mathsf{tag}}_2$. By the unclonability of $\mathsf{TagGen}$, $\widetilde{\mathsf{tag}}_r \neq \mathsf{tag}$ for some $r$.

$\mathsf{MiM}_{\mathsf{nm}}$ chooses $\mathsf{tag}$ as the tag for the external left session and $\widetilde{\mathsf{tag}}_r$ as the tag for the external right session. It then plays the non-malleability experiment by forwarding messages between the external sessions and their respective internal sessions, while internally emulating an honest interaction in the unchosen internal right session. Finally, the non-malleability distinguisher $D_{\mathsf{nm}}$ simply runs the cloning distinguisher $D_{U,r}$ for session $r$. Observe that since $\widetilde{\mathsf{tag}}_r \neq \mathsf{tag}$, $D_{\mathsf{nm}}$ receives the same extracted value as $D_{U,r}$ with overwhelming probability. Thus, if the cloning distinguishers simultaneously succeed with probability $1/2+\epsilon$, then $D_{\mathsf{nm}}$ succeeds with probability $\geq 1/2+\epsilon-\text{negl}(\lambda)$. □

If the non-malleable commitment is *concurrent* non-malleable, then the same construction is *many-many* unclonable.

**Theorem 8.** *Assuming $n_1$-round concurrent non-malleable commitments and $n_2$-round unclonable tag-generation protocols, there exists a many-many unclonable commitment with $n_1+n_2$ or $n_1+n_2-1$ rounds.*

*Proof.* Consider an execution of the unclonable commitment game with $k$ left sessions and $k+r$ right sessions. Due to the unclonability of $\mathsf{TagGen}$, for each left session there is at most one right session with the same identity $\mathsf{tag} \neq \perp$, except with negligible probability. This can be reduced to the $(1 \to 2)$ unclonability of the tag-generation by randomly selecting one of the $k = \text{poly}(\lambda)$ left sessions and two of the $k+r = \text{poly}(\lambda)$ right sessions. Therefore at least one of the $k+1$ sets of right sessions does not include any sessions whose tags match a left session. Many-many unclonability can be reduced to the concurrent non-malleability of $(\mathsf{NMCom}, \mathsf{NMOpen})$ similarly to the proof of Theorem 7; identify these sessions at the end of the commitment phase, then run the corresponding cloning distinguisher.

When combining the construction in Fig. 4 with the post-quantum tag-generation protocol and post-quantum non-malleable commitments, we get a *post-quantum* construction of unclonable commitments. Post-quantum non-malleable commitments were first introduced by [ABG+21]. Later work improved the assumptions to only require post-quantum one-way functions [BLS22], and even achieved a constant-round version [LPY23].

**Corollary 1 (Post-Quantum Interactive Unclonable Commitments).** *Assuming post-quantum one-way functions, there exist constant-round post-quantum unclonable commitments.* □

*Proof.* This is immediate from Theorem 7, Lemma 1 (two-round post-quantum unclonable tag-generation from one-way functions), and [LPY23].

**Corollary 2 (Unclonable Commitments from PKQM).** *Assuming non-interactive non-malleable commitments and public key quantum money, there exist non-interactive unclonable commitments.*

*Proof.* This is immediate from Theorem 7 and Lemma 2 (non-interactive tag-generation from public key quantum money). □

We note that non-interactive non-malleable commitments with respect to commitment[3] can be constructed in the CRS model from any non-malleable public key encryption (PKE) scheme with perfect correctness. Non-malleable PKE can be constructed from any PKE with security against chosen-plaintext attacks [CDMW18]. Although [CDMW18] proved only classical security, we observe that their construction is also post-quantum secure.

We now return to the construction sketch of non-interactive non-malleable commitments. The CRS consists of a public key. To commit to a message, encrypt it under the public key in the CRS, and to open, reveal the randomness. Perfect correctness guarantees binding, since there is a unique message that was encrypted. Although non-malleable PKE is typically defined *without* a tag, a tag $t$ can be generically added by committing to $t \| m$ instead of $m$ [PR05b].

**Non-interactive Construction from Weaker Assumptions.** Unfortunately, public key quantum money is currently only known from strong assumptions such as post-quantum indistinguishability obfuscation [AC12, FGH+12, Zha21]. In order to build an unclonable commitment from weaker assumptions, we will make use of the fact that commitments have both a commitment phase and an opening phase. This structure allows for the tag to be generated in the commitment phase, then "confirmed" in the opening phase. The delayed verification allows for the usage of techniques from private-key quantum money in order to guarantee that the tag cannot be cloned.

**Theorem 9.** *If there exist post-quantum non-interactive commitments which are* $(1,4)$*-concurrent non-malleable with respect to commitment, then there exist non-interactive unclonable commitments.*

*Proof (Sketch).* The construction is given in Fig. 5. Hiding follows directly from $(\mathsf{NMCom}, \mathsf{NMOpen})$. The statistical binding extractor extracts from $\mathsf{nmcom}_{\mathsf{tag}}$ and $\mathsf{nmcom}_{\mathsf{msg}}$ in unbounded time, then measures whether $|\psi\rangle$ is consistent with the extracted messages.

Unclonability reduces to the concurrent non-malleability of $(\mathsf{NMCom}, \mathsf{NMOpen})$, with four right sessions. If $\mathsf{MiM}_U$, $D_{U,1}$, $D_{U,2}$ win the cloning game with probability $1/2 + \varepsilon$, then we can construct a non-malleability adversary $\mathsf{MiM}_{\mathsf{nm}}$ and distinguisher $D_{\mathsf{nm}}$ which win the concurrent non-malleability game with probability $1/2 + \varepsilon - \mathrm{negl}(\lambda)$. In particular, $\mathsf{MiM}_{\mathsf{nm}}$ can maul $\mathsf{nmcom}_{\mathsf{msg}}$. At a high level, $\mathsf{MiM}_{\mathsf{nm}}$ picks $(x, \theta)$, receives $\mathsf{nmcom}_{\mathsf{msg}}$ from the challenger, and internally runs $\mathsf{MiM}_U$ to obtain two non-malleable commitments from each of the right session: $\widetilde{\mathsf{nmcom}}_{\mathsf{tag},1}$, $\widetilde{\mathsf{nmcom}}_{\mathsf{msg},1}$, $\widetilde{\mathsf{nmcom}}_{\mathsf{tag},1}$, and $\widetilde{\mathsf{nmcom}}_{\mathsf{msg},1}$.

[3] Non-interactive non-malleable commitments were previously studied in [DIO98]. However, they achieve non-malleability with respect to opening, which is weaker than non-malleability with respect to commitment.

*Tools.* The construction uses a post-quantum non-interactive commitment $(\mathsf{NMCom}, \mathsf{NMOpen})$ which is concurrent non-malleable with respect to commitment.

$\underline{\mathsf{UCom}_C(1^\lambda, m)}$*:* The committer does the following.

1. Sample $x, \theta \leftarrow \{0,1\}^\lambda$.
2. Compute $(\mathsf{nmcom}_{\mathsf{tag}}, d_{\mathsf{tag}}) \leftarrow \mathsf{NMCom}_C(\mathsf{tag} = 0, (x, \theta))$.
3. Compute $(\mathsf{nmcom}_{\mathsf{msg}}, d_{\mathsf{msg}}) \leftarrow \mathsf{NMCom}_C(\mathsf{tag} = \mathsf{nmcom}_{\mathsf{tag}}, m)$.
4. Send $(\mathsf{nmcom}_{\mathsf{tag}}, \mathsf{nmcom}_{\mathsf{msg}}, |x\rangle_\theta)$ to the receiver.

$\underline{\mathsf{UCom}_R(1^\lambda, (\mathsf{nmcom}_{\mathsf{tag}}, \mathsf{nmcom}_{\mathsf{msg}}, |\psi\rangle))}$*:* Reject the commitment if $\mathsf{nmcom}_{\mathsf{tag}} = 0$.

$\underline{\mathsf{UOpen}_C(1^\lambda)}$*:* The committer sends the openings $(d_{\mathsf{tag}}, (x, \theta))$ and $(d_{\mathsf{msg}}, m)$ to the receiver.

$\underline{\mathsf{UOpen}_R(1^\lambda, (\mathsf{nmcom}_{\mathsf{tag}}, \mathsf{nmcom}_{\mathsf{msg}}, |\psi\rangle), ((d_{\mathsf{tag}}, (x, \theta)), (d_{\mathsf{msg}}, m))}$*:* The receiver does the following:

1. Verify the openings to $\mathsf{nmcom}_{\mathsf{tag}}$ and $\mathsf{nmcom}_{\mathsf{msg}}$. If either reject, output $\perp$. Otherwise, continue.
2. Measure $H^\theta |\psi\rangle$ in the basis $\theta$. If the result is $x$, output $m$. Otherwise, output $\perp$.

**Fig. 5.** A Non-Interactive Unclonable Commitment from Post-Quantum Non-Malleable Commitments

At the end of the commitment phase, the non-malleability experiment provides the committed values $\widetilde{v}_{\mathsf{tag},1}$, $\widetilde{v}_{\mathsf{msg},1}$, $\widetilde{v}_{\mathsf{tag},2}$, and $\widetilde{v}_{\mathsf{msg},2}$ to $D_{\mathsf{nm}}$ for each of these commitments, each of which is replaced with $\perp$ if that non-malleable commitment uses $\mathsf{nmcom}_{\mathsf{tag}}$ as its tag. $D_{\mathsf{nm}}$ then attempts to use these values to learn at least one of the right committed values $\widetilde{v}_1$ or $\widetilde{v}_2$. If it can, then it can run the corresponding cloning distinguisher to win the non-malleability game. Note that since cloning a commitment requires *both* distinguishers to win simultaneously, using one of them is enough.

If one of the two right sessions uses *different* tags from the left session, learning the value for that session is trivial. On the other hand, if both right sessions use the same tags as the left session, then one of them used the tag invalidly, or else $\mathsf{MiM}_U$ has cloned $|x\rangle_\theta$ from the left session. Since $\mathsf{MiM}_{\mathsf{nm}}$ has $(x, \theta)$, it can check which of the two sessions used the tag invalidly; this session must be a commitment to $\perp$. See the full version for more details. □

The construction in Fig. 5 is also *many-many* unclonable (Definition 4).

**Theorem 10.** *If there exist post-quantum non-interactive commitments which are concurrent non-malleable with respect to commitment, then there exist non-interactive many-many unclonable commitments.*

*Proof (Sketch).* The proof is very similar to the proof of Theorem 9, so we omit the full details. At a high level, we define a $\mathsf{MiM}_{\mathsf{nm}}$ which outputs two non-malleable commitments for each right session of the cloning game. The distinguisher $D_{\mathsf{nm}}$ attempts to extract the values of the unclonable commitments using the values revealed in the non-malleability game. If it succeeds in extracting all values in at least one distinguishing set, it runs the corresponding cloning distinguisher on the extracted values. $D_{\mathsf{nm}}$ loses the distinguishing game *only* if it fails to extract one value from every distinguishing set or if at least one of the cloning distinguishers would guess incorrectly. By assumption, the latter occurs with probability $\leq 1/2 - \epsilon$.

The crux of the argument is to show that $D_{\mathsf{nm}}$ successfully extracts every value from some distinguishing set with overwhelming probability. If this is the case, then $D_{\mathsf{nm}}$ wins the non-malleability game with probability $\geq 1/2 + \epsilon - \text{negl}(\lambda)$, which violates the security of $(\mathsf{NMCom}, \mathsf{NMOpen})$. It is possible to show that for every left commitment in the cloning game, at most one right commitment in the cloning game validly uses the same tag, except with negligible probability. Since there are $k$ left commitments and $k+1$ distinguishing sets, at least one distinguishing set only contains commitments which do *not* validly use any tags from a left session. The values revealed by the non-malleability game suffice to extract every value from this set. □

## 5 Unclonable Proofs

### 5.1 Definition

Intuitively, an unclonable proof should guarantee that an adversary cannot use a simulated left proof to produce two new proofs without knowing either witness. This is very similar to the related notion of non-malleable zero knowledge. However, a key difference is unclonability does not require the right proofs to be different than the left proof for its guarantee to hold. Thus, *unclonability is meaningful even if the adversary attempts to directly forward the left proof.*

*Proof Cloning Experiment.* The proof cloning experiment $\mathsf{Cl\text{-}Arg}(1^\lambda, x, w, \rho)$ uses a proof $\langle \mathsf{Prove}, \mathsf{Verify} \rangle$ for a language $\mathcal{L}$ and is parameterized by a statement-witness pair $(x, w) \in R_{\mathcal{L}}$ and a quantum advice string $\rho$. It involves three sessions: a "left" session between an honest left prover $P_L(x, w)$ and an adversarial man-in-the-middle $\mathsf{MiM}(\rho)$, and two right sessions between $\mathsf{MiM}$ and an honest right verifier, respectively named $V_1$ and $V_2$. The sessions may be interleaved arbitrarily. In the left session, $P_L$ attempts to convince $\mathsf{MiM}$ that $x \in \mathcal{L}$, using $\langle \mathsf{Prove}, \mathsf{Verify} \rangle$. In the right sessions, $\mathsf{MiM}$ chooses statements $\widetilde{x_1}$ and $\widetilde{x_2}$ adaptively, then attempts to convince $V_1$ and $V_2$ that $\widetilde{x}_1 \in \mathcal{L}$ and $\widetilde{x}_2 \in \mathcal{L}$, respectively, using the same proof system. $\mathsf{Cl\text{-}Arg}(1^\lambda, x, w, \rho)$ outputs a tuple $(\tau_{\mathsf{MiM}}, (\widetilde{x}_1, \widetilde{a}_1), (\widetilde{x}_2, \widetilde{a}_2))$ consisting of the view of $\mathsf{MiM}$, the statements $\widetilde{x}_1$, $\widetilde{x}_2$ argued in the right sessions, and the acceptance bits $\widetilde{a}_1$, $\widetilde{a}_2$ of the two right verifiers (where 1 is accept).

**Definition 5 (Simulation-Extraction-Unclonability).** *A proof system* $\langle \mathsf{Prove}, \mathsf{Verify} \rangle$ *is* SE-unclonable *if there exists a QPT simulator-extractor* $\mathsf{SE}$, *which outputs a tuple* $((\tau_{\mathsf{MiM}}, (\widetilde{x}_1, \widetilde{a}_1), (\widetilde{x}_2, \widetilde{a}_2)), \widetilde{w_1}, \widetilde{w_2})$, *such that for every (non-uniform) QPT adversary* $\mathsf{MiM}$ *with quantum advice* $\rho$ *and every* $(x, w) \in R_{\mathcal{L}}$,

1. ***Zero Knowledge.*** $\mathsf{Cl\text{-}Arg}(1^\lambda, x, w, \rho)$ *is computationally indistinguishable from the first output* $(\tau_{\mathsf{MiM}}, (\widetilde{x}_1, \widetilde{a}_1), (\widetilde{x}_2, \widetilde{a}_2))$ *of* $\mathsf{SE}(1^\lambda, x, \rho)$.
2. ***Witness Extraction.*** *If* $\mathsf{SE}(1^\lambda, x, \rho)$ *outputs acceptance bits* $\widetilde{a}_1 = \widetilde{a}_2 = 1$, *then at least one of the following holds, except with negligible probability:* $(\widetilde{x}_1, \widetilde{w}_1) \in \mathsf{Rel}_{\mathcal{L}}$ *or* $(\widetilde{x}_2, \widetilde{w}_2) \in \mathsf{Rel}_{\mathcal{L}}$.

It is also interesting to consider the case where the right sessions may use different proof systems $\langle \widetilde{\mathsf{Prove}_1}, \widetilde{\mathsf{Verify}_1} \rangle$ and $\langle \widetilde{\mathsf{Prove}_2}, \widetilde{\mathsf{Verify}_2} \rangle$ than the left session. Note that these proof systems may be for different languages $\widetilde{\mathcal{L}}_1$ and $\widetilde{\mathcal{L}}_2$ than the left proof system. We denote the corresponding experiment as $\mathsf{Cl\text{-}Arg}_{\mathsf{strong}}(1^\lambda, x, w, \rho)$.

**Definition 6 (Strong SE-Unclonability).** *A proof system* $\langle \mathsf{Prove}, \mathsf{Verify} \rangle$ *is* strongly SE-unclonable *if there exists a QPT simulator-extractor* $\mathsf{SE}$ *which outputs a tuple* $((\tau_{\mathsf{MiM}}, (\widetilde{x}_1, \widetilde{a}_1), (\widetilde{x}_2, \widetilde{a}_2)), \widetilde{w_1}, \widetilde{w_2})$, *such that for every (non-uniform) QPT adversary* $\mathsf{MiM}$ *with quantum advice* $\rho$ *and every* $(x, w) \in R_{\mathcal{L}}$,

1. ***Zero Knowledge.*** $\mathsf{Cl\text{-}Arg}_{\mathsf{strong}}(1^\lambda, x, w, \rho)$ *is computationally indistinguishable from the first output of* $\mathsf{SE}(1^\lambda, x, \rho)$.
2. ***Witness Extraction.*** *If* $\mathsf{SE}(1^\lambda, x, \rho)$ *outputs acceptance bits* $\widetilde{a}_1 = \widetilde{a}_2 = 1$, *then at least one of the following holds, except with negligible probability:* $(\widetilde{w}_1, \widetilde{x}_1) \in \mathsf{Rel}_{\widetilde{\mathcal{L}}_1}$ *or* $(\widetilde{w}_2, \widetilde{x}_2) \in \mathsf{Rel}_{\widetilde{\mathcal{L}}_2}$.

Since strong SE-unclonability is a property about the composition of *different* protocols, it is much more difficult to achieve. We explore this notion further in Sect. 6.

*Soundness-Unclonability.* An alternative notion only requires that at least one of the two right sessions retains its soundness, even if the left session is simulated for a potentially false statement. Note that in this case, the adversary can violate the soundness of one right session simply by forwarding the left session.

**Definition 7 (Soundness Unclonability).** *A proof* $\langle \mathsf{Prove}, \mathsf{Verify} \rangle$ *is* soundness-unclonable *if there exists a QPT simulator* $\mathcal{S}$ *such that for every QPT* $\mathsf{MiM}$ *with quantum advice* $\rho$,

1. ***Zero Knowledge.*** *For every* $(x, w) \in \mathsf{Rel}_{\mathcal{L}}$, $\mathcal{S}(1^\lambda, x, \rho)$ *is computationally indistinguishable from* $\mathsf{Cl\text{-}Arg}(1^\lambda, x, w, \rho)$.
2. ***Simulation-Soundness.*** *For a simulated execution* $\nu$, *let* $\mathsf{SV}(\nu)$ *be the number of right sessions* $i$ *where* $R_i$ *outputs accept, but* $\widetilde{x}_i \notin \mathcal{L}$, *i.e. a soundness violation occurs. For every* $x \in \{0,1\}^n$,

$$\Pr[\mathsf{SV}(\nu) > 1 : \nu \leftarrow \mathcal{S}(1^\lambda, x, \rho)] = \mathrm{negl}(\lambda)$$

We may additionally consider the strong variant, which permits any proof in the right sessions. The relation between SE-unclonability and soundness-unclonability seems similar to the relation between simulation-extraction [PR05b] and simulation-soundness [Sah99] in non-malleable zero knowledge, which [JP14] shows are incomparable.

*Many-Many Unclonability.* A natural extension is to consider an adversary which sees $k$ proofs in the left sessions and attempts to produce $k+r$ proofs in the right sessions, for polynomial $k$ and $r$. Such settings have previously been studied in the context of program copy-protection [Aar09, LLQZ22]. Intuitively, $k \to k+r$ unclonability should guarantee that the adversary must know at least $r$ witnesses (or that at least $r$ sessions are sound). We denote the corresponding game as $\mathsf{Cl\text{-}Arg}_{k,r}$.

**Definition 8 (Many-Many SE-Unclonability).** *A proof system* $\langle \mathsf{Prove}, \mathsf{Verify} \rangle$ *is* many-many SE-unclonable *if for every* $k = \text{poly}(\lambda)$*,* $r = \text{poly}(\lambda)$*, there exists a QPT simulator-extractor* $\mathsf{SE}_{k,r}$ *which outputs a tuple* $((\rho_{\mathsf{MiM}}, (\widetilde{x}_i, \widetilde{a}_i)_{i\in[k+r]}), (\widetilde{w}_i)_{i\in[k+r]})$*, such that for every (non-uniform) QPT adversary* $\mathsf{MiM}$ *with quantum advice* $\rho$ *and for every* $(x, w) \in R_{\mathcal{L}}$*,*

1. ***Zero Knowledge.*** $\mathsf{Cl\text{-}Arg}_{k,r}(1^\lambda, x, w, \rho)$ *is computationally indistinguishable from the first output of* $\mathsf{SE}_{k,r}(1^\lambda, x, \rho)$*.*
2. ***Witness Extraction.*** *For a simulated execution* $\nu$*, let* $\mathsf{Acc}(\nu) = \sum_{i=1}^{k+r} \widetilde{a}_i$ *be the number of right sessions which output accept. Let* $W(\nu, (\widetilde{w}_i)_{i\in[k+r]})$ *be the number of right sessions* $i$ *such that* $(\widetilde{x}_i, \widetilde{w}_i) \in \mathsf{Rel}_{\mathcal{L}}$*. Then*

$$\Pr[\mathsf{Acc}(\nu) > W(\nu, (\widetilde{w}_i)_{i\in[k+r]}) + k : (\nu, (\widetilde{w}_i)_{i\in[k+r]}) \leftarrow \mathsf{SE}_{k,r}(1^\lambda, x, \rho)] = \text{negl}(\lambda)$$

**Definition 9 (Many-Many Soundness Unclonability).** *A proof* $\langle \mathsf{Prove}, \mathsf{Verify} \rangle$ *is* many-many soundness-unclonable *if for every* $k = \text{poly}(\lambda)$ *and* $r = \text{poly}(\lambda)$*, there exists a QPT simulator* $\mathcal{S}$ *such that for every QPT* $\mathsf{MiM}$ *with quantum advice* $\rho$*,*

1. ***Zero Knowledge.*** *For every* $(x, w) \in \mathsf{Rel}_{\mathcal{L}}$*,* $\mathcal{S}(1^\lambda, x, \rho)$ *is computationally indistinguishable from* $\mathsf{Cl\text{-}Arg}_{k,r}(1^\lambda, x, w, \rho)$*.*
2. ***Simulation-Soundness.*** *For a simulated execution* $\nu$*, let* $\mathsf{SV}(\nu)$ *be the number of right sessions* $i$ *where* $R_i$ *outputs accept, but* $\widetilde{x}_i \notin \mathcal{L}$*, i.e. a soundness violation occurs. For every* $x \in \{0,1\}^n$*,*

$$\Pr[\mathsf{SU}(\nu) > k : \nu \leftarrow \mathcal{S}(1^\lambda, x, \rho)] = \text{negl}(\lambda)$$

## 5.2 Constructions

**Theorem 11.** *Assuming* $n_1$*-round unclonable tag-generation protocols and* $n_2$*-round simulation-sound proofs, there exist soundness-unclonable proofs with* $n_1 + n_2$ *or* $n_1 + n_2 - 1$ *rounds.*

*Tools.* The construction uses the following tools.
- A simulation-extractable (respectively, simulation-sound) proof $\langle \mathsf{NMProve}, \mathsf{NMVerify} \rangle$.
- An unclonable tag-generation protocol $\mathsf{TagGen}$.

$\underline{\langle \mathsf{UProve}, \mathsf{UVerify} \rangle(1^\lambda, x)}$*:* The committer and receiver do the following:

1. Run the tag-generation protocol $\mathsf{TagGen}$. Let $\mathsf{tag}$ be the output of the protocol.
2. Run the proof $\langle \mathsf{NMProve}, \mathsf{NMVerify} \rangle(\mathsf{tag}, x)$ using $\mathsf{tag}$ as the non-malleable tag.

*Round Reduction Optimization.* If the same party sends the last message of $\mathsf{TagGen}$ and the first message of $\langle \mathsf{Prove}, \mathsf{Verify} \rangle(\mathsf{tag}, x)$, they may send the messages at the same time.

**Fig. 6.** An SE-Unclonable (respectively, Simulation-Sound) Proof

*Proof.* The construction is given in Fig. 6. Soundness is immediate from the underlying proof. The simulator constructs an adversary for the zero knowledge property of the underlying simulation-sound proof by internally emulating the two right verifiers. Consider an execution of the simulator where $\widetilde{V_1}$ and $\widetilde{V_2}$ respectively accept. Say that $\mathsf{TagGen}$ output $\mathsf{tag}$ in the left session and $\widetilde{\mathsf{tag}}_1$, $\widetilde{\mathsf{tag}}_2$ in the right sessions in this execution. By the security of $\mathsf{TagGen}$, $\mathsf{tag} \neq \widetilde{\mathsf{tag}}_b$ for some $b \in \{1, 2\}$. Otherwise, one of the right tags is $\perp$, and so the corresponding right receiver would have rejected the proof. By the simulation-soundness of $\langle \mathsf{Prove}, \mathsf{Verify} \rangle$, right session $b$ is sound. Therefore $\widetilde{x}_b \in \mathcal{L}$, except with negligible probability, since $R_b$ accepted. □

**Theorem 12.** *Assuming $n_1$-round unclonable tag-generation protocols and $n_2$-round simulation-extractable proofs, there exist SE-unclonable proofs with $n_1+n_2$ or $n_1+n_2-1$ rounds.*

*Proof (Sketch).* The construction is given in Fig. 6. Soundness is immediate from the underlying proof. To show SE-unclonability, we construct the simulator-extractor $\mathsf{SE}_U$. We start by constructing a halfway simulator-extractor $\mathsf{SE}_{U,1/2}$ which will succeed in simulating the adversary's view, but will only extract a valid witness with probability $1/2 \pm \mathrm{negl}(\lambda)$. Given an unclonability adversary $\mathsf{MiM}^*_U$, we will define an adversary $\mathsf{MiM}_{\mathsf{nm}}$ for the non-malleability game with $\langle \mathsf{NMProve}, \mathsf{NMVerify} \rangle$. We will then use $\mathsf{MiM}_{\mathsf{nm}}$ to define $\mathsf{SE}_{U,1/2}$.

$\mathsf{MiM}_{\mathsf{nm}}$ uniformly samples $b \leftarrow \{1, 2\}$. Set $\bar{b}$ to be the unchosen value. It internally runs the unclonability experiment using $\mathsf{MiM}_U$ by internally simulating an honest right verifier $R_{\bar{b}}$ and forwarding the other sessions to the external non-malleability experiment.

$\mathsf{SE}_{U,1/2}$ runs the non-malleability simulator-extractor $\mathsf{SE}_{\mathsf{nm}}$ on $\mathsf{MiM}_{\mathsf{nm}}$ to obtain $\mathsf{MiM}_{\mathsf{nm}}$'s view and a witness $\widetilde{w}_b$. Whenever the tags for the forwarded sessions do not match, i.e. $\widetilde{\mathsf{tag}}_b \neq \mathsf{tag}$, and both internal right sessions accept,

$\mathsf{SE}_{U,1/2}$ succeeds in extracting a witness. This happens with probability 1/2, so Watrous rewinding can be used to amplify the extraction success rate to $1 - \text{negl}(\lambda)$.

Full details of the proof can be found in the full version. □

As it turns out, the construction in Fig. 6 is also *many-many* unclonable if it is instantiated with a concurrent non-malleable proof.

**Theorem 13.** *Assuming $n_1$-round unclonable tag-generation protocols and $n_2$-round concurrent simulation-sound proofs, there exist many-many soundness-unclonable proofs with $n_1 + n_2$ or $n_1 + n_2 - 1$ rounds.*

*Proof.* In a tag-generation experiment where a man-in-the-middle acts as the receiver in $k$ left sessions and as the sender in $k + r$ right sessions, for each left session there is at most one right session with the same output $\mathsf{tag} \neq \perp$, except with negligible probability. In particular, at least $r$ right sessions have tags which either do not belong to any left session or are $\perp$, except with negligible probability. This can be reduced to the $(1 \to 2)$ unclonability of the tag-generation by randomly selecting one of the $k = \text{poly}(\lambda)$ left sessions and two of the $k + r = \text{poly}(\lambda)$ right sessions.

If $k + r'$ right sessions accept, then these sessions all have tags which are not $\perp$. By the previous property, $r'$ of these tags also do not belong to any left session. Therefore the concurrent simulation-soundness of $\langle \mathsf{NMProve}, \mathsf{NMVerify} \rangle$ implies that the statements in these sessions belong to the NP language. □

**Theorem 14.** *Assuming $n_1$-round unclonable tag-generation protocols and $n_2$-round concurrent simulation-extractable proofs, there exist many-many SE-unclonable proofs with $n_1 + n_2$ or $n_1 + n_2 - 1$ rounds.*

*Proof.* Define an adversary $\mathsf{MiM}_{\mathsf{nm}}$ for the underlying non-malleable proof $\langle \mathsf{NMProve}, \mathsf{NMVerify} \rangle$ which internally interacts with the cloning adversary $\mathsf{MiM}_U$. $\mathsf{MiM}_{\mathsf{nm}}$ internally runs the tag-generation procedure in each session to determine the tags, then forwards the messages of $\langle \mathsf{NMProve}, \mathsf{NMVerify} \rangle$ between $\mathsf{MiM}_U$ and the external non-malleability game. The simulator-extractor for the unclonable proof runs the simulator-extractor for the underlying non-malleable proof $\langle \mathsf{NMProve}, \mathsf{NMVerify} \rangle$ on $\mathsf{MiM}_{\mathsf{nm}}$. Recall from the proof of Theorem 13 that at least $r$ right sessions have tags which either do not belong to any left session or are $\perp$, except with negligible probability. If $k + r'$ right sessions accept, then at least $r'$ of these tags are not $\perp$ and do not belong to any left session. The concurrent simulator-extractor for $\langle \mathsf{NMProve}, \mathsf{NMVerify} \rangle$ extracts witnesses for these sessions, except with negligible probability. □

**Corollary 3.** *Assuming post-quantum one-way functions, there exist post-quantum interactive soundness-unclonable proofs and post-quantum interactive simulation-sound proofs.*

*Proof.* This is immediate from Theorem 12, Theorem 11, Lemma 1, and the fact that post-quantum one-way functions imply simulation-sound proofs [GLM23].

Post-quantum simulation-extractable proofs are implied by post-quantum bounded-concurrent secure two party computation, which [GLM23] also constructs from post-quantum one-way functions.

□

**Corollary 4.** *Assuming simulation-extractable (respectively, simulation-sound) NIZKs and public key quantum money, there exist SE-unclonable (respectively, simulation-sound) NIZKs.*

*Proof.* This is immediate from Theorem 12, Theorem 11, and Lemma 2. Note that the round-reduction optimization can be applied to make the construction non-interactive. □

Simulation-sound NIZKs can be constructed in the common reference string (CRS) model from any NIZK using one-way functions [Sah99]. Furthermore, a simulation-sound NIZK can be compiled to be simulation-extractable using public key encryption, again in the CRS model. We observe that this construction also holds against quantum adversaries. We sketch a brief justification in the full version.

## 5.3 Relation to Public Key Quantum Money

As we saw in the previous section, it is possible to construct unclonable zero-knowledge proofs using public-key quantum money and a non-malleable zero-knowledge proof. If the non-malleable proof is non-interactive, then so is the resulting unclonable proof. Since NIZKs are inherently non-interactive, it is natural to wonder whether an unclonable NIZK can be used to construct public-key quantum money. We show in this section that this is indeed the case.

**Theorem 15.** *If one-way functions and simulation-extractable (respectively, simulation-sound) NIZKs exist, then the existence of SE-unclonable NIZKs (respectively, soundness-unclonable NIZKs) is equivalent to the existence of public-key quantum money.*

*Proof.* We prove the forward implication in Lemma 3 and the backward implication in Lemma 4 □

**Lemma 3.** *If post-quantum one-way functions and either SE-unclonable NIZKs or soundness-unclonable NIZKs exist, then public key quantum money also exists.*

*Proof.* We construct a quantum money mini-scheme, which can be transformed into a fully-fledged public key quantum money scheme using signatures. To generate a banknote, first honestly generate a common reference string $\mathsf{crs}$ for the NIZK.[4] Next, sample a post-quantum, non-interactive, statistically binding com-

[4] In general, the bank can be trusted, since its only goal is to prevent the duplication of banknotes. This differs from quantum *lightning* [Zha19], where an adversary may mint its own notes and wants to produce two with the same serial number.

mitment $\mathsf{com} \leftarrow \mathsf{Com}(0)$.[5] Finally, sample an unclonable NIZK $|\pi\rangle$ for the language $\mathcal{L}_0$ of commitments to 0 and instance $\mathsf{com}$. The serial number is $\mathsf{com}$ and the quantum money state is $|\pi\rangle$. $\mathsf{Verify}_{\mathsf{QM}}(\mathsf{com}, |\pi\rangle)$ outputs the result of the NIZK verification procedure $\mathsf{Verify}_{\mathsf{NIZK}}(\mathsf{com}, |\pi\rangle)$.

Consider the following hybrid experiments, where a challenger generates a quantum money state and an adversary attempts to clone it. The probability of the adversary successfully cloning in each of these hybrids is computationally close to the others.

- $H_0$: This is the original experiment. The experiment outputs whether the adversary successfully clones, i.e. whether $\mathsf{Verify}_{\mathsf{QM}}$ accepts both states.
- $H_1$: The same as the original experiment, except $|\pi\rangle$ is simulated. Indistinguishability from $H_0$ reduces to the zero knowledge property of the NIZK.
- $H_2$: The same as $H_1$, except $\mathsf{com} \leftarrow \mathsf{Com}(1)$ is a commitment to 1 instead of to 0. Indistinguishability from $H_1$ reduces to the hiding of $\mathsf{Com}$.

Say that in $H_2$, the adversary produces two states $|\widetilde{\pi}_1\rangle$ and $|\widetilde{\pi}_2\rangle$ such that $\mathsf{Verify}_{\mathsf{QM}}(\mathsf{com}, |\widetilde{\pi}_1\rangle)$ and $\mathsf{Verify}_{\mathsf{QM}}(\mathsf{com}, |\widetilde{\pi}_2\rangle)$ both output accept. Since the quantum money verification procedure just runs the NIZK verification procedure, $\mathsf{Verify}_{\mathsf{NIZK}}(\mathsf{com}, |\widetilde{\pi}_r\rangle)$ also accepts for both $r = 1, 2$. However, since $\mathsf{Com}$ is statistically binding, $\mathsf{com} \notin \mathcal{L}_0$. If the NIZK was SE-unclonable, this is a contradiction, since the simulator-extractor cannot extract a valid witness for either right session. Similarly, this is also a contradiction if the NIZK was instead soundness-unclonable, since $\mathsf{MiM}$ breaks soundness in both right sessions. Therefore the adversary cannot clone the quantum money state in $H_2$, except with $\text{negl}(\lambda)$ probability. Since the cloning success probability is computationally close in $H_0$ and $H_2$, no computationally bounded adversary can clone banknotes, except with $\text{negl}(\lambda)$ probability. □

As previously discussed at the end of Sect. 5.2, simulation-sound NIZKs can be constructed from any one-way function and any NIZK. Additionally, simulation-extractable NIZKs can be constructed from any public-key encryption scheme and any NIZK. Thus we obtain the following corollary of Theorem 15.

**Corollary 5.** *If one-way functions and NIZKs exist, then the existence of soundness-unclonable NIZKs is equivalent to the existence of public-key quantum money. Furthermore, if public-key encryption and NIZKs exist, then the existence of SE-unclonable NIZKs is equivalent to the existence of public-key quantum money.*

## 6 Strong Unclonability: Negative Results

We begin with a very simple attack ruling out strongly unclonable NIZKs.

[5] These can be constructed in the CRS model from post-quantum one-way functions, e.g. using Naor's commitment scheme [Nao90].

**Theorem 16.** *There do not exist strongly SE-unclonable or soundness-unclonable NIZKs.*

*Proof.* As an explicit attack, MiM first forwards the left NIZK to $R_1$. $R_1$ verifies it, then returns it to MiM. By correctness of the NIZK and the Gentle Measurement Lemma [Win99], the returned state is statistically close to the original NIZK. Finally, MiM sends the returned NIZK to $R_2$ and $R_2$ verifies it. □

*Interactive Zero Knowledge.* Next, we rule out even *interactive* strongly unclonable proofs. At a high level, the man-in-the-middle and two right verifiers will agree to run an MPC which allows them to act as a single verifier for the left session. It is important to note that $V_1$ and $V_2$ do *not* have an authenticated channel. Instead, they must pass messages to each other through the man-in-the-middle. Since the man-in-the-middle may tamper with these messages, the MPC must be secure against a dishonest majority in order for the resulting proof to be sound.

**Theorem 17.** *Assuming stateful secure multiparty computation for quantum functionalities with dishonest majority, there do not exist strongly SE-unclonable proofs in the plain model, except for languages in BQP. Furthermore, there do not exist strongly soundness-unclonable proofs in the plain model.*[6]

*Remark.* Commitments imply MPC with dishonest majority [BCKM21]. Statefulness can be generically and unconditionally added by the ideal functionality authenticating and encrypting messages to itself. We sketch a construction for this in the full version.

*Proof.* We describe an explicit attack. Define a stateful MPC protocol between three parties $P_1$, $P_2$, and $P_3$, where $P_3$ provides input. The ideal functionality acts as the verifier in the left proof system. In other words, it interprets the input as a message from the prover in the left proof, then computes and outputs the verifier's next message according to the left proof system. At the end of the protocol, it outputs the ideal verifier's output to all parties. This is possible since the ideal verifier outputs a classical bit.

In right session 1, $V_1$ and MiM interact in an execution of this MPC protocol, where $V_1$ controls $P_1$ and MiM controls both $P_2$ and $P_3$. In right session 2, $V_2$ and MiM interact in an execution of this MPC protocol, where $V_2$ controls $P_2$ and MiM controls both $P_1$ and $P_3$. In both sessions, MiM, acting as $P_3$, is supposed to input messages from the left proof system into the MPC.

*Claim.* Both right proof systems are computationally sound.

*Proof.* This is immediate from the security of the MPC against a dishonest majority and the soundness of the left proof system. □

[6] We note that the attack given in the proof uses computationally sound right protocols. Definition 6 permits any right proofs, whether computationally or statistically sound. In principle, we could design a weaker definition which only considers statistically sound right protocols, and the attack would not apply.

To carry out the attack, $\mathsf{MiM}$ forwards messages between the sessions. More explicitly, consider round $i$ of both sessions. $\mathsf{MiM}$ receives messages $m_i^{1\to 2}$ and $m_i^{1\to 3}$ from $V_1$, as well as messages $m_i^{2\to 1}$ and $m_i^{2\to 3}$ from $V_2$. It computes messages $m_i^{3\to 1}$ and $m_i^{3\to 2}$ honestly according to the MPC protocol. Then, it sends $m_i^{2\to 1}$ and $m_i^{3\to 1}$ to $V_1$ and sends $m_i^{1\to 2}$ and $m_i^{3\to 2}$ to $V_2$.

Observe that due to the message forwarding, $\mathsf{MiM}$, $V_1$, and $V_2$ are effectively participating in a single execution of the MPC protocol. Thus, whenever the ideal verifier (which is emulated by the MPC) would accept a statement $x \in \mathcal{L}$, both $V_1$ and $V_2$ will accept the same statement $x \in \mathcal{L}$.

*Claim.* The left proof cannot be soundness-unclonable.

*Proof.* Consider an execution where the left proof is simulated for a *false* statement. Then the right protocols both accept the same false statement, breaking soundness. □

*Claim.* If the left proof is SE-unclonable for a language $\mathcal{L}$, then $\mathcal{L} \in \mathsf{BQP}$.

*Proof.* To decide a statement $x$, run the simulator-extractor on $x$ and $\mathsf{MiM}$ to obtain a witness $w$. Check whether $w$ is a witness for $x \in \mathcal{L}$. If $x \in \mathcal{L}$, then both $V_1$ and $V_2$ will accept. Then, by definition of SE-unclonability, the simulator-extractor must output a valid witness for at least one of the two right sessions. Note that the simulator-extractor is still well-defined for $x \notin \mathcal{L}$, but by definition, it cannot output a valid witness. Since $\mathsf{MiM}$ is defined independently of $x$, the simulator-extractor thus decides $\mathcal{L}$. □

□

## 7 Strong Unclonability: Positive Results

### 7.1 Commitments: Strong Unclonability with Respect to Statistical Binding

Our construction is in the QROM and uses any unclonable-indistinguishable encryption scheme $(\mathsf{UGen}, \mathsf{UEnc}, \mathsf{UDec})$ with information-theoretic unclonability as a black box. Such schemes are known in the QROM, and constructing them in the plain model is an open question [AKL+22]. We note that using a random oracle bypasses the impossibility result, since a dishonest-majority MPC cannot correctly implement random oracle calls.

**Theorem 18.** *There exists a non-interactive commitment scheme in the QROM which is unclonable with respect to any (right) statistically binding commitment scheme. This holds unconditionally against any adversary which makes polynomially-many queries to the random oracle.*

The construction is given in Fig. 7 see the full version for the proof.

```
$\underline{\mathsf{UCom}_C(1^\lambda, m)}$: The sender does the following:
  1. Sample an unclonable encryption key $k \leftarrow \mathsf{UGen}(1^\lambda)$.
  2. Encrypt $|\psi\rangle \leftarrow \mathsf{UEnc}(k, m)$.
  3. Sample $r \leftarrow \{0,1\}^\lambda$, then query the random oracle to get $h \leftarrow H(r, k)$.
  4. Send $(|\psi\rangle, h)$ to the receiver.

$\underline{\mathsf{UOpen}_C(1^\lambda)}$: The sender sends $(r, k)$ to the receiver.

$\underline{\mathsf{UOpen}_R(1^\lambda, (|\psi\rangle, h))}$: The receiver does the following:
  1. Query the random oracle to get $h' \leftarrow H(r, k)$. If $h \neq h'$, output $\perp$. Otherwise
     continue.
  2. Decrypt and output $m \leftarrow \mathsf{UDec}(k, |\psi\rangle)$.
```

**Fig. 7.** A Strongly Unclonable Commitment with Respect to Statistical Binding

**Acknowledgements.** G.M. is supported by the European Research Council through an ERC Starting Grant (Grant agreement No. 101077455, ObfusQation).

J.R. is supported in part by the NSF award 1916939, DARPA SIEVE program, a gift from Ripple, a DoE NETL award, a JP Morgan Faculty Fellowship, a PNC center for financial services innovation award, and a Cylab seed funding award.

## References

[Aar09] Aaronson, S.: Quantum copy-protection and quantum money. In: Proceedings of the 24th Annual IEEE Conference on Computational Complexity. CCC 2009, Paris, France, 15–18 July 2009, pp. 229–242. IEEE Computer Society (2009)

[ABG+21] Agarwal, A., Bartusek, J., Goyal, V., Khurana, D., Malavolta, G.: Post-quantum multi-party computation. In: Canteaut, A., Standaert, F.-X. (eds.) EUROCRYPT 2021. LNCS, vol. 12696, pp. 435–464. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-77870-5_16

[AC12] Aaronson, S., Christiano, P.: Quantum money from hidden subspaces. In: Karloff, H.J., Pitassi, T. (eds.) 44th ACM STOC, pp. 41–60. ACM Press (2012)

[AK21] Ananth, P., Kaleoglu, F.: Unclonable encryption, revisited. In: Nissim, K., Waters, B. (eds.) TCC 2021. LNCS, vol. 13042, pp. 299–329. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-90459-3_11

[AKL+22] Ananth, P., Kaleoglu, F., Li, X., Liu, Q., Zhandry, M.: On the feasibility of unclonable encryption, and more. In: Dodis, Y., Shrimpton, T. (eds.) CRYPTO 2022, Part II. LNCS, vol. 13508, pp. 212–241. Springer, Heidelberg (2022). https://doi.org/10.1007/978-3-031-15979-4_8

[AKN+23] Agrawal, S., Kitagawa, F., Nishimaki, R., Yamada, S., Yamakawa, T.: Public key encryption with secure key leasing. In: Hazay, C., Stam, M. (eds.) EUROCRYPT 2023, Part I. Lecture Notes in Computer Science, vol. 14004, pp. 581–610. Springer, Cham (2023). https://doi.org/10.1007/978-3-031-30545-0_20

[APV23] Ananth, P., Poremba, A., Vaikuntanathan, V.: Revocable cryptography from learning with errors. Cryptology ePrint Archive, Report 2023/325 (2023). https://eprint.iacr.org/2023/325

[BC23] Broadbent, A., Culf, E.: Uncloneable cryptographic primitives with interaction. CoRR, abs/2303.00048 (2023)

[BCKM21] Bartusek, J., Coladangelo, A., Khurana, D., Ma, F.: One-way functions imply secure computation in a quantum world. In: Malkin, T., Peikert, C. (eds.) CRYPTO 2021. LNCS, vol. 12825, pp. 467–496. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-84242-0_17

[BGG+23] Bartusek, J., et al.: Obfuscation and outsourced computation with certified deletion. Cryptology ePrint Archive, Report 2023/265 (2023). https://eprint.iacr.org/2023/265

[BL20] Broadbent, A., Lord, S.: Uncloneable quantum encryption via oracles. In: Flammia, S.T. (ed.) 15th Conference on the Theory of Quantum Computation, Communication and Cryptography. TQC 2020, 9–12 June 2020, Riga, Latvia, vol. 158. LIPIcs, pp. 4:1–4:22. Schloss Dagstuhl - Leibniz-Zentrum für Informatik (2020)

[BLS22] Bitansky, N., Lin, H., Shmueli, O.: Non-malleable commitments against quantum attacks. In: Dunkelman, O., Dziembowski, S. (eds.) EUROCRYPT 2022, Part III. LNCS, vol. 13277, pp. 519–550. Springer, Cham (2022). https://doi.org/10.1007/978-3-031-07082-2_19

[BS17] Ben-David, S., Sattath, O.: Quantum tokens for digital signatures. Cryptology ePrint Archive, Report 2017/094 (2017). https://eprint.iacr.org/2017/094

[BSW16] Bellare, M., Stepanovs, I., Waters, B.: New negative results on differing-inputs obfuscation. In: Fischlin, M., Coron, J.-S. (eds.) EUROCRYPT 2016, Part II. LNCS, vol. 9666, pp. 792–821. Springer, Heidelberg (2016). https://doi.org/10.1007/978-3-662-49896-5_28

[CDMW18] Choi, S.G., Dachman-Soled, D., Malkin, T., Wee, H.: A black-box construction of non-malleable encryption from semantically secure encryption. J. Cryptol. **31**(1), 172–201 (2018)

[CLLZ21] Coladangelo, A., Liu, J., Liu, Q., Zhandry, M.: Hidden cosets and applications to unclonable cryptography. In: Malkin, T., Peikert, C. (eds.) CRYPTO 2021. LNCS, vol. 12825, pp. 556–584. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-84242-0_20

[DDN00] Dolev, D., Dwork, C., Naor, M.: Nonmalleable cryptography. SIAM J. Comput. **30**(2), 391–437 (2000)

[DIO98] Di Crescenzo, G., Ishai, Y., Ostrovsky, R.: Non-interactive and non-malleable commitment. In: 30th ACM STOC, pp. 141–150. ACM Press (1998)

[FGH+12] Farhi, E., Gosset, D., Hassidim, A., Lutomirski, A., Shor, P.W.: Quantum money from knots. In: Goldwasser, S. (ed.) ITCS 2012, pp. 276–289. ACM, January 2012

[GLM23] Goyal, V., Liang, X., Malavolta, G.: On concurrent multi-party quantum computation. In: Handschuh, H., Lysyanskaya, A. (eds.) CRYPTO 2023. LNCS, vol. 14085, pp. 129–161. Springer, Cham (2023). https://doi.org/10.1007/978-3-031-38554-4_5

[Goy11] Goyal, V.: Constant round non-malleable protocols using one way functions. In: Fortnow, L., Vadhan, S.P. (eds.) 43rd ACM STOC, pp. 695–704. ACM Press (2011)

[JK23] Jawale, R., Khurana, D.: Unclonable non-interactive zero-knowledge. IACR Cryptology ePrint Archive, p. 1532 (2023)

[JP14] Jain, A., Pandey, O.: Non-malleable zero knowledge: black-box constructions and definitional relationships. In: Abdalla, M., De Prisco, R. (eds.) SCN 2014. LNCS, vol. 8642, pp. 435–454. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-10879-7_25

[KN22] Kitagawa, F., Nishimaki, R.: Functional encryption with secure key leasing. In: Agrawal, S., Lin, D. (eds.) ASIACRYPT 2022. LNCS, vol. 13794, pp. 569–598. Springer, Cham (2022). https://doi.org/10.1007/978-3-031-22972-5_20

[LLQZ22] Liu, J., Liu, Q., Qian, L., Zhandry, M.: Collusion resistant copy-protection for watermarkable functionalities. In: Kiltz, E., Vaikuntanathan, V. (eds.) TCC 2022. LNCS, vol. 13747, pp. 294–323. Springer, Cham (2022). https://doi.org/10.1007/978-3-031-22318-1_11

[LPY23] Liang, X., Pandey, O., Yamakawa, T.: A new approach to post-quantum non-malleability. In: 64th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2023, Santa Cruz, CA, USA, 6–9 November 2023. IEEE (2023)

[Nao90] Naor, M.: Bit commitment using pseudo-randomness. In: Brassard, G. (ed.) CRYPTO 1989. LNCS, vol. 435, pp. 128–136. Springer, New York (1990). https://doi.org/10.1007/0-387-34805-0_13

[PR05a] Pass, R., Rosen, A.: Concurrent non-malleable commitments. In: 46th Annual IEEE Symposium on Foundations of Computer Science (FOCS 2005), 23–25 October 2005, Pittsburgh, PA, USA, Proceedings, pp. 563–572. IEEE Computer Society (2005)

[PR05b] Pass, R., Rosen, A.: New and improved constructions of non-malleable cryptographic protocols. In: Gabow, H.N., Fagin, R. (eds.) 37th ACM STOC, pp. 533–542. ACM Press (2005)

[Sah99] Sahai, A.: Non-malleable non-interactive zero knowledge and adaptive chosen-ciphertext security. In: 40th FOCS, pp. 543–553. IEEE Computer Society Press (1999)

[Wie83] Wiesner, S.: Conjugate coding. SIGACT News **15**(1), 78–88 (1983)

[Win99] Winter, A.J.: Coding theorem and strong converse for quantum channels. IEEE Trans. Inf. Theory **45**(7), 2481–2485 (1999)

[WZ82] Wootters, W.K., Zurek, W.H.: A single quantum cannot be cloned. Nature **299**(5886), 802–803 (1982)

[Zha19] Zhandry, M.: Quantum lightning never strikes the same state twice. In: Ishai, Y., Rijmen, V. (eds.) EUROCRYPT 2019. LNCS, vol. 11478, pp. 408–438. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-17659-4_14

[Zha21] Zhandry, M.: White box traitor tracing. In: Malkin, T., Peikert, C. (eds.) CRYPTO 2021. LNCS, vol. 12828, pp. 303–333. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-84259-8_11

# Unclonable Cryptography with Unbounded Collusions and Impossibility of Hyperefficient Shadow Tomography

Alper Çakan[1(✉)] and Vipul Goyal[1,2]

[1] Carnegie Mellon University, Pittsburgh, PA, USA
acakan@cs.cmu.edu

[2] NTT Research, Sunnyvale, CA, USA
vipul@vipulgoyal.org

**Abstract.** Quantum no-cloning theorem gives rise to the intriguing possibility of quantum copy protection where we encode a program or functionality in a quantum state such that a user in possession of $k$ copies cannot create $k+1$ copies, for any $k$. Introduced by Aaronson (CCC'09) over a decade ago, copy protection has proven to be notoriously hard to achieve. Previous work has been able to achieve copy-protection for various functionalities only in restricted models: (i) in the bounded collusion setting where $k \rightarrow k+1$ security is achieved for a-priori fixed collusion bound $k$ (in the plain model with the same computational assumptions as ours, by Liu, Liu, Qian, Zhandry [QIP'23]), or, (ii) only $k \rightarrow 2k$ security is achieved (relative to a structured quantum oracle, by Aaronson [CCC'09]).

In this work, we give the first *unbounded* collusion-resistant (i.e. multiple-copy secure) copy-protection schemes, answering the long-standing open question of constructing such schemes, raised by multiple previous works starting with Aaronson (CCC'09).

More specifically, we obtain the following results.

– We construct (i) public-key encryption, (ii) public-key functional encryption, (iii) signature and (iv) pseudorandom function schemes whose keys are copy-protected against unbounded collusions in the plain model (i.e. without any idealized oracles), assuming (post-quantum) subexponentially secure $i\mathcal{O}$ and LWE.
– We show that any unlearnable functionality can be copy-protected against unbounded collusions, relative to a classical oracle.
– As a corollary of our results, we rule out the existence of *hyperefficient quantum shadow tomography*,
  • even given non-black-box access to the measurements, assuming subexponentially secure $i\mathcal{O}$ and LWE, or,
  • unconditionally relative to a quantumly accessible *classical* oracle,

  and hence answer an open question by Aaronson (STOC'18).

We obtain our results through a novel technique which uses identity-based encryption to construct multiple copy secure copy-protection

E. Boyle and M. Mahmoody (Eds.): TCC 2024, LNCS 15366, pp. 225–256, 2025.
https://doi.org/10.1007/978-3-031-78020-2_8

schemes from 1-copy $\rightarrow$ 2-copy secure schemes. We believe our technique is of independent interest.

Along the way, we also obtain the following results.

– We define and prove the security of new collusion-resistant monogamy-of-entanglement games for *coset states*.
– We construct a classical puncturable functional encryption scheme whose master secret key can be punctured at all functions $f$ such that $f(m_0) \neq f(m_1)$. This might also be of independent interest.

**Keywords:** Quantum Cryptography · Unclonable Cryptography · Shadow Tomography · Copy Protection

## 1 Introduction

The no-cloning principle, a fundamental implication of quantum mechanics, shows that arbitrary unknown quantum states cannot be copied. This simple principle allows us to imagine applications that are classically impossible. Indeed, it has found a wide range of applications in cryptography, starting with the work of Wiener [30] where he puts forward the notion of *quantum money*, where we imagine that there is a bank producing quantum states, called *banknotes*, that are secure against counterfeiting: any (malicious) user in possession of $k$ banknotes for any $k$ cannot produce $k+1$ *authentic* banknotes. The interesting notion of quantum banknotes (i.e., unclonable authenticatable quantum states) also led Aaronson [1] to pose the following question:

Can we use quantum information to *copy-protect functionalities/programs*, where user(s) in possession of some number of copies of a program $P$ cannot produce more working copies?

In more detail, we want to achieve the following. A vendor encodes a functionality[1] into a quantum state, and a user in possession of such a state can use it to evaluate the functionality any number of times, and we want to achieve $a \rightarrow b$ copy-protection: any malicious user(s) in possession of $a$ such copies of the program cannot produce $b$ working copies. Similar to quantum money, this is an impossible feat in a classical world since classical information can be readily copied any amount of times. Therefore, in a classical world, once you are given a single working copy of the program, you can make any number of copies of it.

Perhaps surprisingly, [1] showed copy-protection using quantum information is indeed possible: relative to a *structured*[2] *quantum oracle*, any *unlearnable* program can be copy-protected in a way that is $k \rightarrow k+r$ secure (for any [polynomial] $k$ and some $r > k$). That is, in the construction of [1], the adversary

[1] For example, a proprietary software or a decryption program/key of an encryption system that is used to distribute encrypted content.

[2] The oracle used in this construction takes as input a function and a value, evaluates the function on the value, or takes as input a function and outputs a Haar random state associated with it.

is prevented from doubling their number of working copies. Later, Aaronson et al. [7] showed that relative to a *classical* structured oracle (that depends on the program being copy protected) model, any unlearnable program can be copy-protected, but this time only in the $\mathsf{1\text{-}copy} \rightarrow \mathsf{2\text{-}copy}$ setting.

*Copy-Protecting Decryption Keys, PRFs and Signing Keys.* In a related line of work, Georgiou and Zhandry [16] started the study of *single-decryptor encryption*, that is, copy-protection for decryption functionality (i.e. secret keys) of a public-key encryption (PKE) scheme where an adversary tries to create $k+1$ working decryption keys given only $k$ copy-protected keys. More formally, in this model, a *pirate* adversary obtains the classical public key and $k$ copy-protected quantum secret keys of the scheme. Then, it produces $k+1$ *freeloader* adversaries that are possibly entangled but not communicating[3], and these freeloaders are presented with classical challenge ciphertexts. We require that they cannot all succeed in decrypting simultaneously. [16] also gave a $1 \rightarrow 2$ secure copy-protection scheme relative to a structured oracle. Later, Coladangelo et al. [13] showed how to construct a $1 \rightarrow 2$ copy-protected public-key encryption using *coset states*, this time in the plain model, assuming quantum hardness of LWE, (post-quantum) subexponentially secure indistinguishability obfuscation and one-way functions. They also construct $1 \rightarrow 2$ copy-protection schemes for pseudorandom functions (PRF), based on the same assumptions. Liu et al. [21] constructed *bounded* collusion-resistant PKE and PRF schemes, by showing through an elegant proof that the $k$-way parallel repetitions of the schemes of [13] are bounded $k \rightarrow k+1$ copy-protection secure. Further, they also construct a *bounded* $k \rightarrow k+1$ copy-protection secure scheme for the signing keys of a signature scheme. However, for all schemes of [21], the collusion-bound $k$ is fixed during setup, the sizes of the schemes grow (linearly) with the bound $k$ and the copy-protected key generation is stateful.

*Collusion-Resistant Copy-Protection.* Unfortunately, none of the previous work satisfy the most-general notion of unbounded collusion-resistant copy-protection where we require $k \rightarrow k+1$ security for all polynomials $k$ (that is not known and hence the size of the scheme does not depend on it).

In particular, all schemes of [7,13,21] can easily be broken when the adversary obtains multiple copies. Any 2 users (in case of the first two works) or $k+1$ users (for the fixed $k$ value, in case of [21])[4] with copy-protected keys can create an *anonymous classical* program/string (which can be copied/distributed any number of times) that can be used to decrypt any ciphertext in case of encryption schemes, or evaluate/sign any input in case of general programs, PRFs and signatures. The only other scheme, that of [1], is only $k \rightarrow 2k$ secure rather than $k \rightarrow k+1$, and more importantly, since it relies on structured quantum

[3] If they were allowed to communicate, one freeloader could hold the secret key and all the other freeloaders would simply send their challenge ciphertexts to him to decrypt and send back the result.

[4] We re-emphasize that the size of the scheme (e.g. ciphertext and public-key sizes) grows with the set $k$ value, so it cannot be set arbitrarily large.

oracles, it cannot even be heuristically instantiated since we do not have any (even candidate) constructions of general-purpose quantum circuit obfuscation. In fact, [21] argues that even any extension of the scheme of [1] would require such obfuscation, since it uses Haar random states and there is evidence that these states cannot be *classically verified* [20,21].

We believe that the security guarantees of the previous work [7,13,21] are very unrealistic in the age of the Internet: the users can actually mount the anonymous attacks described above through classical channels, by simply measuring their key and sending the classical measurement result to other parties or posting it online!

*Computational Complexity of Shadow Tomography.* Lastly, aside from theoretical interest in the unbounded collusion setting in and of itself, we note that it is a theoretically important problem also due to its intimate connection to the computational complexity of another important problem, *shadow tomography* [3].

The above state of affairs leaves open the following natural question also raised explicitly in several previous works [1,4,13,21]:

Can we use quantum information to construct unbounded collusion-resistant copy-protection schemes?

In this work, we answer the above question positively, in the plain model, with computational assumptions matching the previous work.

## 1.1 Our Results

In this work, we resolve the long-standing open problem of constructing fully collusion-resistant copy-protection schemes by constructing such schemes for public-key encryption, public-key functional encryption, signatures and pseudorandom functions, all in the plain model.

*Copy-Protecting Decryption Keys.* We construct encryption schemes where the secret keys are copy-protected.

**Theorem 1.** *Assuming post-quantum subexponentially secure indistinguishability obfuscation and subexponentially secure LWE, there exists a public-key encryption scheme with fully collusion-resistant copy-protected secret keys.*

Our computational assumptions above match[5] the assumptions made by [13] to achieve $1 \rightarrow 2$ copy-protection and those made by [21] to achieve $k \rightarrow k+1$ bounded collusion-resistant copy-protected public-key encryption schemes.

[5] More specifically, our assumptions exactly match the assumptions made by [21], but [13] assumes polynomially secure LWE whereas we assume subexponentially secure LWE. We emphasize that [13] still assume subexponentially secure $i\mathcal{O}$ and subexponentially secure one-way functions.

**Theorem 2.** *Assuming post-quantum subexponentially secure indistinguishability obfuscation and subexponentially secure LWE, there exists a public-key* functional *encryption scheme with fully collusion-resistant copy-protected secret keys.*

Prior to our work, the only construction of functional encryption with copy-protected secret keys (given by Kitagawa and Nishimaki [19]) was in the $1 \rightarrow 2$ copy-protection setting, based on assumptions same as ours, and in a weaker security model where no key queries were allowed after seeing the challenge ciphertext. Furthermore, on top of matching the assumptions previous work used for constructing copy-protected public-key encryption, the $i\mathcal{O}$ assumption we make for our copy-protected FE scheme can be considered necessary since functional encryption is known to be equivalent to indistinguishability obfuscation (up to subexponential security loss) [10].

Since functional encryption can be used to construct identity-based encryption [28] and attribute-based encryption [17,25] in a straightforward manner, our work also gives the first identity-based encryption and attribute-based encryption schemes with collusion-resistant copy-protected secret keys. Through copy-protected identity-based encryption, we can also obtain *unclonable identity cards*, first suggested by [1].

*Copy-Protecting PRF and Signature Keys.* We also construct copy-protection schemes for a family of pseudorandom functions (PRF) and signing keys of a signature scheme.

**Theorem 3.** *Assuming post-quantum subexponentially secure indistinguishability obfuscation and subexponentially secure LWE, there exists a PRF and a signature scheme with fully collusion-resistant copy-protected keys.*

We refer the reader to the full version [11] for the full constructions and the proofs.

*Copy-Protecting All Unlearnable Functionalities.* We also show how to copy-protect any unlearnable functionality, relative to a classical oracle.

**Theorem 4.** *Assuming post-quantum subexponentially secure one-way functions, for any unlearnable functionality, there exists a fully collusion-resistant copy-protection scheme relative to an efficient* classical *oracle.*

This supersedes[6] both [1], which uses a structured quantum oracle and only satisfies $k \rightarrow 2k$ copy-protection, and [7] which uses a structured classical oracle but only satisfies $1 \rightarrow 2$ copy-protection. We refer the reader to the full version [11] for the construction and the proofs.

[6] Note that Theorem 4 and similar results of [1,6] cannot be securely instantiated in the plain model for all unlearnable functionalities, since [8] proves that there exists an unlearnable functionality that cannot be copy-protected in the plain model.

*Impossibility of Hyperefficient Shadow Tomograph.* Shadow tomography, introduced by Aaronson [3], is the following task: Given many copies of a mixed state $\rho$ and a list of binary measurements $\{E_1, \ldots, E_M\}$, estimate the acceptance probabilities $\mathrm{Tr}(E_i\rho)$ of measurements $E_i$ within additive error $\varepsilon$, for all measurements $i \in [M]$. This task has important ramifications for quantum information theory, since it means that we can learn many properties of a quantum state without needing to do a full tomography of it, which necessarily requires exponentially many copies of the state [24]. It has also found many applications in cryptography, such as (i) [3] who shows that unconditional copy-protection is not possible (ii) [9] who shows that unconditional PKE cannot exist even if we allow public-keys to be quantum and (iii) [18] who shows that unconditional one-way state generators cannot exist. Lastly, shadow tomography also has connections to the question of *classical vs. quantum advice*, and the related complexity classes $\mathsf{BQP/poly}$ and $\mathsf{BQP/qpoly}$. Note that in general, and in particular in all of these applications, the measurement set is indexed by all possible strings in some support and $M$ is exponential in the security parameter or in the number of qubits. In fact, the case $M = \mathsf{poly}(\lambda)$ can be trivially solved in polynomial $(M/\varepsilon^2)$ time with polynomially many copies, by estimating each $\mathrm{Tr}(E_i\rho)$ for $i \in [M]$ simply by actually performing the measurements $E_i$ multiple times on separate copies.

[3] showed that shadow tomography can be performed in a sample-efficient manner; using $\mathsf{poly}(n, \log M, \frac{1}{\varepsilon})$ copies of an $n$-qubit state $\rho$, however, their scheme is not *computationally efficient*, with time complexity $\tilde{O}(M)$[7]. In light of above, they posed the following as an open question: is *hyperefficient* shadow tomography possible? That is, is it possible to perform shadow tomography with time complexity $\mathsf{poly}(n, \log M, \frac{1}{\varepsilon})$? Note that in this case, we ask that the set of measurements $\{E_i\}_{i\in M}$ be implemented by a uniform quantum algorithm $E$ that on input $i, \tau$, applies the measurement $E_i$ to the state $\tau$. We will be given this quantum circuit $E$ as input and we are asked to output a quantum circuit $C$ such that $C(i)$ estimates $\mathrm{Tr}(E_i\rho)$ for all $i$.[8]

Previously, hyperefficient shadow tomography was ruled out only relative to quantum oracles [3,5,20], where we only get oracle access to the measurement circuit $E$. Through a generic attack on copy-protection schemes using shadow tomography given by [3,27], a corollary of our results is the impossibility of hyperefficient shadow tomography, answering the open question of [3].

**Corollary 1.** *Assuming post-quantum subexponentially secure indistinguishability obfuscation and LWE, there does not exist a hyperefficient shadow tomography algorithm.*

[7] As noted above, $M$ is exponential in the security parameter or in the number of qubits.

[8] Without these assumptions, even reading the descriptions of all measurements or outputting all the estimates would take $\Omega(M)$ time.

**Corollary 2.** *Assuming post-quantum subexponentially secure one-way functions, relative to a* classical *oracle, there cannot exist a hyperefficient shadow tomography algorithm.*

We note that making computational assumptions is necessary, since, hyperefficient shadow tomography is possible given access to PP oracle.[9]

*Technical Contributions and Additional Results.* An important contribution of our work is a novel technique to construct collusion-resistant copy-protection schemes which relies on using identity-based encryption (IBE). We use this technique in all of our constructions and we believe it to be of independent interest. Our technique could be considered an analogue of the technique of using digital signatures to construct full-fledged (i.e. collusion-resistant) quantum money from single banknote schemes [4,15,22]. We also define and prove the security of new collusion-resistant *monogamy-of-entanglement* games [13,14] for coset states to prove the security of our schemes. We refer the reader to the full version [11] for the formal statements and the proof.

Finally, using the techniques we employ to prove the security of our functional encryption scheme, we also give a construction of a classical functional encryption scheme where the master secret key can be punctured such that the resulting master key allows issuing keys only for functions $f$ that satisfy $f(m_0) = f(m_1)$. This allows us to remove the interaction/key queries after the challenge ciphertext in the usual functional encryption security game, since the adversary can issue their own keys using the punctured master secret key. This might also be of independent interest.

**Theorem 5.** *Assuming subexponentially secure indistinguishability obfuscation and one-way functions, there exists a functional encryption scheme whose master secret key can be punctured at all functions $f$ such that $f(m_0) \neq f(m_1)$.*

We refer the reader to the full version [11] for the constructions and the proofs.

## 2 Technical Overview

### 2.1 Public-Key Encryption with Copy-Protected Secret Keys

Let us first describe our model, which is the same as previous work [1,13,16,21]. We consider a public-key encryption scheme with classical ciphertexts, a classical public-key and an additional (quantum) algorithm QKeyGen. The copy-protected key generation algorithm QKeyGen, on input the classical secret key, outputs a reusable quantum state that can be used to correctly decrypt any number of ciphertexts. For security, we will require that a user with $k$ copy-protected secret keys cannot create $k+1$ keys. More formally, in an *anti-piracy game* for public-key encryption, we have an adversary, called a *pirate*. This adversary is given the public key $pk$, and then for any (polynomial) number of rounds, it queries for

[9] We thank an anonymous reviewer for pointing out this remark.

quantum copy-protected secret keys. After it is done, it outputs pairs of challenge messages $(m_\ell^0, m_\ell^1)_{\ell \in [k+1]}$ and $k+1$ (possibly entangled) *freeloader* adversaries, where $k$ is the number of copy-protected keys it has queried. Then, the challenger samples challenge bits $b_\ell$, and presents each freeloader with $\mathsf{Enc}(pk, m_\ell^{b_\ell})$. The freeloaders output their predictions $b'_\ell$, and the adversary wins if $b'_\ell = b_\ell$ for all $\ell \in [k+1]$. We require that no efficient adversary can win with probability better than $1/2 + \mathsf{negl}(\lambda)$. The baseline success probability is $1/2$, since the pirate adversary can output $k$ of its keys to the first $k$ freeloaders, and let the last freeloader randomly guess the challenge bit $b_{k+1}$.

$1 \to 2$ **Copy-Protection Secure Construction of Coladangelo et al.** [13] As a warm-up, we will recall the $1 \to 2$ copy-protection secure construction based on coset states, given by [13], which also forms the base of our construction.

A coset state [13,29] is a state of the form $\sum_{a \in A} (-1)^{\langle s', a \rangle} |a + s\rangle =: |A_{s,s'}\rangle$ where $A \subseteq \mathbb{F}_2^n$ is a subspace and $s, s' \in \mathbb{F}_2^n$. [13,14] showed that coset states satisfy a property called *strong monogamy-of-entanglement (MoE)*, which is as follows. Consider the following game between an adversary tuple $\mathcal{A}_0, \mathcal{A}_1, \mathcal{A}_2$ and a challenger. Challenger uniformly at random samples a subspace $A \subseteq \mathbb{F}_2^n$ of dimension $n/2$ and elements $s, s' \in \mathbb{F}_2^n$, and submits $|A_{s,s'}\rangle$ and the obfuscated programs[10] $i\mathcal{O}(A+s), i\mathcal{O}(A^\perp + s')$ to the adversary $\mathcal{A}_0$. Then, the adversary $\mathcal{A}_0$ outputs two (entangled) registers $\mathsf{R}_1, \mathsf{R}_2$, for $\mathcal{A}_1, \mathcal{A}_2$. Then, $\mathcal{A}_1, \mathcal{A}_2$ receive their registers and also the description of the subspace $A$ (but not the vectors $s, s'$ of course). Finally, $\mathcal{A}_1$ is required to output a vector in $A + s$ and $\mathcal{A}_2$ is required to output a vector in $A^\perp + s'$. Strong MoE property says that no efficient adversary can win this game with non-negligible probability. In a variation used implicitly by [13] and later formalized in a different context by [12], we present $\mathcal{A}_0$ with multiple, say $c$ many, independent coset states (called a *coset state tuple*) and the corresponding membership checking programs, and require that $\mathcal{A}_1, \mathcal{A}_2$ each output vectors in $A_i + s_i$ or $A_i^\perp + s'_i$ for all $i \in [c]$, depending on random challenge strings $r^1, r^2 \in \{0,1\}^c$ presented to them. By a reduction to the original version, it can be shown that no efficient adversary can win this game with non-negligible probability. We call this variation the multi-challenge version.

Now, we move onto the copy-protected public-key encryption construction of [13]. During setup, we sample a coset tuple $(A_i, s_i, s'_i)_{i \in [c]}$. The coset state tuple $|A_{i,s_i,s'_i}\rangle_{i \in [c(\lambda)]}$ becomes the copy-protected quantum secret key, and we output $pk = (i\mathcal{O}(A_i + s), i\mathcal{O}(A_i^\perp + s'_i))_{i \in [c(\lambda)]}$ as the public key. Finally, to encrypt a message $m$, we sample a random string $r$ and an indistinguishability obfuscation $\mathsf{OP} \leftarrow i\mathcal{O}(\mathsf{PCt}_{pk,r,m})$, where $\mathsf{PCt}_{pk,r,m}$ is a program that takes is input vectors $(v_i)_{i \in [c]}$ and, checks if they are in correct cosets with respect to $r$. That is, we require $v \in A_i + s_i$ if the $i$-th bit of $r$ is 0 and $v \in A_i^\perp + s'_i$ if it is 1. The program $\mathsf{PCt}_{pk,r,m}$ outputs the message $m$ if and only if the vectors pass the test. We output $(\mathsf{OPCt}, r)$ as the ciphertext. To decrpyt a message, we simply apply QFT (quantum Fourier transform) to our coset state tuple at indices where $(r)_i = 1$.

[10] Here, we overload the notation to let $A + s$ also denote the program that takes as input a vector $v$ and outputs 1 if $v \in A + s$, and 0 if not, and similarly for $A^\perp + s$.

Then, it is easy to see that running $\mathsf{OPCt}$ coherently on our key and measuring the result gives us $m$ with probability 1[11].

On a high level, the security follows by multi-challenge MoE game, since the two freeloaders, to decrypt their ciphertexts, must be querying the programs $\mathsf{PCt}^{(1)}, \mathsf{PCt}^{(2)}$ at the correct vectors with respect to $r_1, r_2$ respectively, which is exactly the challenge in the MoE game. The proof is more involved since (i) $i\mathcal{O}$ is used rather than ideal oracles and (ii) the freeloaders can be entangled. We discuss this further in the upcoming sections.

**Challenges for Collusion-Resistant Copy-Protection.** First, we note that the construction of [13] is trivially insecure when the adversary is given two copies of the secret key: The adversary can measure one copy of the state $|A_{i,s_i,s'_i}\rangle$ in the computational basis and the other copy in the Hadamard basis, thus obtaining vectors $v_i \in A_i + s_i$ and $w_i \in A_i^{\perp} + s'_i$ for all $i \in [c]$. Using these vectors, one can decrypt any ciphertext and since these vectors are classical information, the pirate adversary can indeed produce any number of working secret keys. Thus, the scheme only satisfies $1 \to 2$ unclonability.

One natural solution, argued by [21], is to try and employ quantum states that already possess a collusion-resistant unclonability guarantee, such as Haar random states or their computational neighbor, pseudorandom states. This is indeed the approach employed by [1] to achieve $k \to 2k$ copy-protection relative to a structured quantum oracle. However, the problem is that there is no known way of verifying such states or employing these states to construct a copy-protection scheme without the use of quantum oracles, and there is evidence that this is an inherent property of such states [20,21].

Another natural solution, used by [21], is to *independently* sample a new coset state tuple $|A^{(j)}_{i,s_i,s'_i}\rangle_{i\in[c(\lambda)]}$ whenever a copy-protected secret key is requested rather than giving out the same key state multiple times. In this case, the ciphertext program also takes as input the index $j$ of the key the decryption procedure is using, and verifies the input vectors with respect to that coset tuple. Therefore, we need to include the corresponding obfuscated membership checking programs for each possible key in the public-key, since otherwise the ciphertexts would not be decryptable by that key. Therefore, we can only have $k$ different key states for a fixed $k$ chosen during setup (which is when $pk$ is created). Therefore, the construction of [21] only achieves $k \to k+1$ copy-protection where the collusion-bound $k$ needs to be known at the time of setup, and the size of the scheme (public key, ciphertexts) grows with $k$, since the scheme basically consists of $k$ independent instances of the $1 \to 2$ secure scheme of [13]. Furthermore, similar to the scheme of [13], this scheme becomes trivially insecure once given $k + 1$ keys, since we will have obtained one of the coset state tuples twice.

[11] By Gentle Measurement Lemma [2], this also means that we can revert the quantum key back to its original state after decrypting a ciphertext.

**Our Solution: Pseudorandom Coset States and Identity-Based Encryption.** As discussed above, if we are sampling independent coset states for each copy-protected key query, we need to have an a-priori bound on the number of different keys. In the unbounded setting, since there are exponentially many cosets, it is not possible to verify all possible cosets using a polynomial size public key $pk$.

Our solution to this is to *compress* the public-key by using pseudorandom coset states rather than truly random ones. We sample a PRF key $K$ and include it in the classical secret key. Then, whenever we need to sample a copy-protected quantum secret key using our classical secret key, we sample a random identity string $id$ from $\{0,1\}^\lambda$ and then sample a coset state tuple using the randomness $F(K, id)$. Our public-key will be an obfuscated program $\mathsf{OPMem}_K$ (with PRF key $K$ embedded) that takes in an $id$, some vectors $(v_i)_{i\in[c]}$ and a *basis* $r$, and verifies the vectors $(v_i)_{i\in[c]}$ with respect to $r$ and the coset tuple associated with $id$. We now have a polynomial size public-key that allows us to verify any possible (honest) coset state tuple.

A high level intuition for security is as follows, where for now we assume we use ideal oracles instead of $i\mathcal{O}$. By PRF security, the adversary's view is indistinguishable from having obtained $k$ independent coset state tuples since for any efficient adversary that obtains any (polynomial) number of quantum secret keys, they will all have unique identity strings with overwhelming probability. Note that we still need to argue that one cannot produce $k+1$ working keys from $k$ independent coset state tuples, which we discuss how to argue in Sect. 2.2.

However, in reality, we are using $i\mathcal{O}$ and not ideal oracles. Now, the first problem is that, the coset state tuples that the adversary obtains during key query phase are no longer pseudorandom, since the adversary does not only have query access to the PRF but rather has the PRF key $K$ inside $pk$. A standard solution when using PRFs and indistinguishability obfuscation is to puncture the PRF key at some inputs. Let $id_1, \dots, id_k$ be the identity strings of the $k$ copy-protected keys obtained by the adversary. We can try to puncture the PRF key at $id_1, \dots, id_k$, but this would make the size of our public-key dependent on $k$. A much more important problem is that the adversary is not required to run $\mathsf{PCt}$ on only one of $id_i$, and in fact, $\mathsf{PCt}$ might be leaking[12] $m$. Or, the adversary somehow might be obtaining the hidden message $m$ by running it on some unrelated identity $id$ and vectors that pass the verification of $\mathsf{PMem}$ for $id$[13]. The latter is because the adversary has access to $K$ in some form (i.e. inside $\mathsf{PMem}$), therefore, it might be somehow obtaining $F(K, id)$ for some $id$. To rule this possibility out, we would need to puncture the PRF key at all strings in $\{0,1\}^\lambda$!

To solve this problem and to puncture the PRF key only at few points, we first want to make sure that the adversary can obtain the hidden message $m$ only by running $\mathsf{PCt}$ on an identity string associated with one of the copy-protected keys it did obtain. To ensure this, we use the following approach based on identity-

[12] Since we are not using black-box obfuscation for $\mathsf{PCt}$.
[13] Since we are not using black-box obfuscation for $\mathsf{PMem}$.

based encryption (IBE). When $\mathsf{PCt}$ is queried on some $id$ and some vectors $(u_j)_j$, after verifying that the vectors are in the correct cosets with respect to $id$ and $r$, the program $\mathsf{PCt}$ outputs an $\mathsf{IBE}$ encryption of $m$ under the identity $id$, rather than $m$ in the clear. We will also change our copy-protected key generation algorithm to output the $\mathsf{IBE}$ secret key associated with $id$. Now, we will be able to argue that if an adversary is able to decrypt a ciphertext and obtain $m$, then it must have obtained $\mathsf{IBE.Enc}(pk, id_i, m)$ for some $id_i$. This is because by the security of $\mathsf{IBE}$, the adversary cannot decrypt ciphertexts under identities other than $id_1, \ldots, id_k$ - the only identities for which it has obtained the IBE secret keys. Above in turn means that the adversary must have run $\mathsf{PCt}$ on $id_i$ and the correct vectors for the coset tuple associated with $id_i$. In essence, we are forcing the adversary the clone one of the original copy-protected secret keys rather than coming up with a new key. Hence, we will eventually reduce to the MoE security of the coset state tuple associated with some $id_i$. Now, we need to only puncture the PRF key at (at most) $k$ points! This is still too many.[14] However, we observe the following: the adversary obtains $k$ secret keys $sk_{id_1}, \ldots, sk_{id_k}$ of the IBE scheme while there are $k+1$ freeloaders. Hence, by pigeonhole principle, two of the $k+1$ freeloaders must be using the same key $sk_{id_i}$ for some $i \in [k]$, and hence, the same coset state tuple - the one associated with $id_i$. As a result, we will only need to puncture the PRF key $K$ at $id_i$.

## 2.2 Proving Security

In this section, we give a high-level overview of the security proof of our public-key encryption construction. Our goal is to reduce the security of our scheme to the monogamy-of-entanglement game, which we will do so by extracting coset vectors $(v_i)_{i \in [c]}$ from the freeloader adversaries. On a high level, our proof uses ideas from [7,13] for simultaneous extraction from entangled adversaries and from [21] to extract the correct coset vectors from multiple freeloaders through a pigeonhole argument. The security proof of our functional encryption construction follows similarly and we refer the reader to the full version [11] for details.

Note that in general, applying an extraction (which is essentially a measurement) on one of the freeloader adversaries might irreversibly damage the other ones since they are entangled. We will first make the testing of the freeloaders projective, which will allow us to argue that we can extract vectors from entangled adversaries since (i) repeating a projective measurement always gives the same outcome and does not change the state, (ii) acting (e.g. extracting) on some part of a state, informally, does not change the behaviour of projective measurements on the other part *too much*. Now, let us briefly discuss *projective implementations*, introduced by Zhandry [31]. Let $\mathcal{E} = \{\mathcal{E}_1, \mathcal{E}_0 = I - \mathcal{E}\}$ be a

[14] Remember that when obfuscating a program using $i\mathcal{O}$, all programs that we will move between must be of the same size. Thus, if we are puncturing the PRF key at $k$ points, our initial obfuscated public-key program needs to be padded to a size that depends on $k$.

binary POVM. [31] shows that there is a *projective* measurement (indexed by a finite subset of $\mathbb{R}_{0 \leq \cdot \leq 1}$) denoted $\mathsf{PI}(\mathcal{E})$ such that the following procedure has the same output distribution as applying $\mathcal{E}$ to $\rho$, for any state $\rho$.[15]

1. Apply $\mathsf{PI}(\mathcal{E})$ to $\rho$ obtain a value $p \in [0, 1]$.
2. Output 1 with probability $p$.

Essentially, the projective implementation estimates the probability that $\mathcal{E}$ *accepts* $\rho$, and does so through a projective measurement. Note that $\mathsf{PI}(\mathcal{E})$ in general is inefficient, however, it can be approximated efficiently [31]. We will ignore this issue in this section - see the full proof for details.

In our anti-piracy game, we assume that the pirate adversary outputs each freeloader as $(U, \sigma)$ where $U$ is a unitary and $\sigma$ is some quantum state. We interpret this as a quantum circuit[16] (with some hardwired quantum state) that takes in a challenge ciphertext and outputs a prediction $b'$. The challenger executes the freeloader using an appropriate universal quantum circuit. Now, let $\mathcal{D}$ be a ciphertext distribution and let $(U_i, \mathsf{R}_i)$ be a freeloader output by the pirate adversary (where $\mathsf{R}_i$ denotes the register containing the quantum part), and consider the following measurement on $\mathsf{R}_i$.

1. Sample $b \leftarrow \{0, 1\}$.
2. Sample $ct \leftarrow \mathcal{D}(m_i^b)$.
3. Execute $U(ct_i, \mathsf{R}_i)$, measure the first qubit of the output registers in computational basis to obtain $b'$.
4. Output 1 if $b' = b$.

When we set $\mathcal{D}$ to be the honest ciphertext distribution where we encrypt $m$ as $\mathsf{PKE.Enc}(pk, m)$, we see that the above measurement exactly corresponds to the testing of the freeloader in the anti-piracy game. Now, consider a modified game (parameterized by some inverse polynomial $\gamma(\lambda)$) where instead of performing this measurement directly, the challenger performs its projective implementation $\mathsf{PI}_{\mathcal{D}}$, and the adversary is said to win if the output is $> 1/2 + \gamma(\lambda)$ for all $k + 1$ freeloaders. Essentially, we are estimating the success probabilities of the freeloaders and comparing it to the baseline. Note that since $\mathsf{PI}_{\mathcal{D}}$ is projective, once we apply it and obtain a value $p$, the post-measurement state will again give $p$ when its tested again for $\mathcal{D}$. [13] proves that this modified game is stronger: it implies the security of the original anti-piracy game. Hence, we will prove security with respect to this stronger game.[17]

Now, we move onto a sketch of the security proof of our scheme. The idea is to test the freeloaders with respect to multiple challenge ciphertext distributions to

[15] We can equivalently say that the expected value of $\mathsf{PI}(\mathcal{E}) \cdot \rho$ is $\mathrm{Tr}[\mathcal{E}_1 \rho]$.

[16] Note that while $U$ is a unitary, this definition is enough to capture general quantum circuits since the adversary can also include empty workspace qubits inside $\sigma$, along with some quantum information obtained from the copy-protected keys.

[17] There is a caveat here that we need to prove security with respect to this game for all inverse polynomial $\gamma(\lambda)$ so that it implies security with respect to the original game.

pinpoint two freeloaders that use the same coset state tuple, and then extracting coset vectors from them and violating its $1 \to 2$ MoE security. Let us assume that an adversary wins the (modified) anti-piracy game (with probability $1/p(\lambda)$ where $p(\cdot)$ is a polynomial), meaning that applying $\mathsf{PI}_{\mathcal{D}}$ yields $> 1/2 + \gamma(\lambda)$ for all $k+1$ freeloaders simultaneously with probability $> 1/p(\lambda)$. We define ciphertext distributions $\mathcal{D}_j$, for all $j \in \{0, 1, \ldots, 2^\lambda\}$, representing all possible identity strings in $\{0,1\}^\lambda$ (plus, the dummy upper bound $2^\lambda$). We define $\mathcal{D}_j$ so that an encryption of a message $m$ is $(i\mathcal{O}(\mathsf{PCt}^j), r)$ where $\mathsf{PCt}^j$ is the program that works as the honest ciphertext program if the input $id$ satisfies $id \geq j$, and otherwise it replaces its hardcoded message $m$ with $\top$ at the beginning. Observe that $\mathcal{D}_0$ corresponds to the honest ciphertext distribution, since $id < 0$ is never satisfied. Similarly, $\mathcal{D}_{2^\lambda}$ corresponds to the dummy ciphertext distribution where the message is not actually contained in the ciphertext. Now, consider the following thought experiment. We apply the measurements $\mathsf{PI}_{\mathcal{D}_i}$ sequentially from $j = 0$ to $j = 2^\lambda$, to all $k+1$ freeloaders. Let $q_{\ell,j}$ denote the outcomes for each freeloader $\ell \in [k+1]$. Intuitively, a non-negligible jump/gap between $q_{\ell,j}$ and $q_{\ell,j+1}$ for $j \in \{0, \ldots, 2^\lambda - 1\}$ will mean that the freeloader $\ell$ is querying the ciphertext program at some vectors that are correct for the coset tuple associated with $j$. Since $\mathcal{D}_0$ is the honest ciphertext distribution of this scheme, the step $j = 0$ corresponds to the original security game and hence we get $q_{\ell,0} > 1/2 + \gamma$ for all $\ell \in [k+1]$ by assumption. We will also have $q_{\ell,2^\lambda} \leq 1/2$ for all $\ell \in [k+1]$ since the step $j = 2^\lambda$ corresponds to the ciphertext distribution $\mathcal{D}_{2^\lambda}$ that does not actually contain the message, and therefore no freeloader[18] can succeed with probability better than $1/2$ against $\mathcal{D}_{2^\lambda}$. Previous works [4,21] use a pigeonhole principle to reduce $k \to k+1$ security to $1 \to 2$ unclonability security, where they conclude that two freeloaders must have a large gap between $|q_{\ell,j} - q_{\ell,j+1}|$ at the same jump point $j$, meaning that they are utilizing the same coset state tuple [21] or two quantum money banknotes must come from the same initial banknote [4]; where they randomly guess this critical index and place the $1 \to 2$ challenge there. However, the problem in our case is that the possible jump points $j_\ell$ are in $\{0, 1, \ldots, 2^\lambda - 1\}$, whereas we only have $k+1$ freeloaders. This creates a multitude of problems: (i) we cannot conclude that there will be a non-negligible jump since the average step between $q_{\ell,0}$ and $q_{\ell,2^\lambda}$ is $\gamma/2^\lambda$, which is negligible, (ii) even if there is a non-negligible jump, we cannot apply the pigeonhole principle to guarantee that there is a pair of freeloaders $\ell, \ell'$ that have the jump index $j_\ell = j_{\ell'}$ since we have $2^\lambda$ slots for $k+1$ freeloaders. Further, note that even if both of the previous concerns worked out and the two freeloaders' non-negligible jump indices coincide, we cannot actually test the freeloaders with respect to all $\mathcal{D}_j$ to find it or randomly guess it since there are exponentially many possibilities. However, a careful reader might guess that thanks to the $\mathsf{IBE}$ security, the challenge ciphertext distributions above actually *collapse* around $k$ points: $j = id_1, \ldots, id_k$, the identity strings of the secret keys obtained by

[18] Here, we are talking about any freeloader program/state, not necessarily the initial ones, since the state of the freeloaders has changed since we already applied the previous tests $\mathsf{PI}_{\mathcal{D}_j}$ for $j = 0, \ldots, 2^\lambda - 1$.

the adversary. That is, we claim that jumps can only happen at indices $j$ that correspond to some $id_i$. The reason is that, informally, the difference between $\mathcal{D}_j$ and $\mathcal{D}_{j+1}$ only occurs when the obfuscated ciphertext program is evaluated at $id = j$, in which case the output is $\mathsf{IBE}$ encryptions of $m$ and $\top$ respectively, both under the identity $j$. However, if $j$ is not one of $id_i$, then the different outputs of these programs will be $\mathsf{IBE}$ ciphertexts that are indistinguishable to the adversary, by the security of $\mathsf{IBE}$. Therefore, no freeloader can detect this change, and there cannot be a jump between $q_{\ell,j}$ and $q_{\ell,j+1}$. This (i) allows us to conclude that for each freeloader there must be a $\gamma/k$ jump (which is non-negligible) at one of $j = id_1, \dots, id_k$, and (ii) since the jump points are now all in $\{id_1, \dots, id_k\}$, we can apply a pigeonhole argument to say that there is two freeloaders have the same jump point since there are $k+1$ freeloaders with $k$ jump slots. However, note that the ciphertext programs are only $i\mathcal{O}$ programs and not ideal oracles, therefore, the above argument is only informal and needs to be proven. Overall, while the above intuitions are the crux of our technique, formalizing these requires care and the full proof delicately intertwines all these observations, whilst also dealing with further technical problems. We refer the reader to the full version [11] for the full proof. We also prove a new results on *collusion-resistant MoE for pseudorandom coset states* that is needed in our proof. We refer the reader to the full version [11] for this result.

## 2.3 Public-Key Functional Encryption with Copy-Protected Functional Keys

In the setting of functional encryption, we now have functional keys, where a functional key for a function $f$ allows one to obtain $f(m)$ given the encryption $\mathsf{Enc}(m)$, and nothing else. Similar to PKE (Sect. 2.1), for functional encryption with copy-protected keys, we require that an adversary that obtains $k$ copy-protected (functional) keys cannot create $k+1$ working keys (for any functions). We also allow the adversary to obtain classical functional keys. We move onto our construction. The starting point is our public-key encryption scheme. To generate a quantum copy-protected key for a function $f$, we sample a random $id$ as before, but now we generate the coset tuple using the randomness $F(K, id||f)$ rather than $F(K, id)$. Basically, the coset states are now associated with both the function $f$ and a random $id$. We note that the random identity is still required, since we allow the adversary to query for multiple copy-protected keys for the same function $f$. We also change our ciphertexts so that they now output an $\mathsf{IBE}$ encryption of $f(m)$ under the identity $id||f$, rather than outputting an encryption of $m$.

**Proving Security: Building and Using Puncturable Functional Encryption.** Proof of security for our FE scheme will be similar to the proof of our PKE scheme (Sect. 2.2). In particular, we will now identify identity string-function pairs (associated with the functional keys) with elements of $\{0, 1, \dots, 2^\lambda \cdot 2^\lambda\}$, and have ciphertext distributions $\mathcal{D}_i$ for all such elements. While we have $2^{2\lambda}$

jump points, similar to PKE, we can argue that they can occur only at $k$ points: $j = id_1||f_1, \ldots, id_k||f_k$, where $f_1, \ldots, f_k$ are the functions the pirate adversary has queried in copy-protected mode and $id_1, \ldots, id_k$ are the associated identity strings in the same order. The reason is that, for other values, either (i) the adversary will not have the IBE secret key for the identity $id||g$ (meaning that it has not queried for the function $g$), or (ii) we will have $g(m_0) = g(m_1)$ (if it has queried for the function $g$ in the classical mode). Thus, $\mathcal{D}_i$ and $\mathcal{D}_{i+1}$ will be indistinguishable at points other than ones listed above; either by IBE security or since the $\mathsf{PCt}$ will output $g(m_0) = g(m_1)$ in both distributions.

There is one caveat left. As discussed before, in our copy-protection security proofs, we crucially rely on *projective implementations* [31] to estimate the success of the freeloader adversaries for the task where they are given an encryption of $m^b$ with random $b \leftarrow \{0, 1\}$ and they output a prediction $b'$ for it. This allows us to simultaneously extract vectors from two entangled freeloader adversaries. While projective implementations are in general inefficient, [31] also gives an efficient algorithm (called *approximated projective implementation*) that approximates it well, using a technique similar to the celebrated witness-preserving $\mathsf{QMA}$ amplification result of [23]. Crucially, we note that above decryption process between the challenger and freeloader, for which we estimate the success probability, is non-interactive (i.e., not single round). However, in a copy-protected functional encryption security game, the freeloader adversaries will be allowed to query for more functional keys after they receive their challenge ciphertexts, for any polynomial number of rounds. Therefore, we will not able to use the approximated projective implementation as-is to estimate the success probability of a freeloader adversary for functional encryption. While one solution might be to try and generalize approximate projective implementations to interactive procedures, given that the original technique of [23] also only applies to $\mathsf{QMA}$ (which is single round), this might be a challenging task.

We side-step the issue above using a classical solution. We define a variation of our scheme where the challenger gives the freeloader adversaries a *punctured master secret key pmsk* along with their challange ciphertext. This punctured key has the challenge messages $m^0, m^1$ chosen by the adversary hardcoded, and it takes in a function $f$ and outputs the secret key for $f$ if $f(m^0) = f(m^1)$. Then, since the freeloader adversaries can simulate (using this punctured key *pmsk*) themselves any key queries that they want to make after seeing the challenge ciphertext, we remove the interaction between the freeloaders and the challenger. As a result, we are again able to use approximate projective implementations in our technique.

The only remaining challenge is making sure that our functional encryption construction is still secure when the adversaries obtain this punctured master secret key, which is an obfuscated program that contains the master secret key *msk*. While *pmsk* will only answer the queries on functions that the adversary was allowed to query for anyways, the problem is that we are using indistinguishability obfuscation rather than black-box obfuscation to compute *psmk*. To resolve this issue, we upgrade our FE construction to use an identity-based encryption scheme with puncturable master secret keys. In such a scheme, we are able to produce a master secret key that can issue identity keys for any identity other than the identity it was punctured at. When we are proving the security of our functional encryption scheme, we will construct hybrids corresponding to all possible $id||f$. Moving between each hybrid, we only need to rely on the security of $\mathsf{IBE}$ at this identity. Therefore, in our security proof, we will not only use a puncturing argument inside our obfuscated ciphertext program $\mathsf{PCt}$, but we will also puncture the $\mathsf{IBE}$ master secret key inside $pmsk$ at $id||f$. Thus, we will be able to rely on the security of $\mathsf{IBE}$ even when the adversary has $pmsk$. We refer the reader to the full version [11] for the full proof.

## 2.4 PRFs and Signature Schemes with Copy-Protected Secret Keys

Let us first describe the setting. In the case of PRFs, we imagine a quantum key generation algorithm that, given the PRF key $K$, can generate copy-protected keys that can be used to evaluate the PRF $F(K, \cdot)$ any number of times. For copy-protection, we require that given $k$ such keys, the adversary cannot create $k+1$ freeloaders that can distinguish $F(K, x)$ versus a random string from the co-domain of $F$, given uniformly at random $x$.[19] In the case of signatures, we have copy-protected re-usable signing keys that can sign any message. Similar to above, given $k$ such keys, pirate outputs $k+1$ freeloaders, and we ask the freeloaders to sign random messages.[20]

Our signature scheme will be the same as our PRF scheme, where the signature on $m$ will be the PRF evaluation $F(K, m)$, with the difference from the PRF scheme being that we will also have a verification key. Similar to the signature scheme construction of Sahai and Waters [26], the verification key will be an obfuscated program that verifies a message-signature pair $(m, \sigma)$ by checking $f(\sigma) = f(F(K, m))$ where $f$ is a one-way function. Due to these similarities, we only discuss our signature scheme here. Both constructions in full with security proofs can be found in the full version [11].

In our signature scheme, a copy-protected signing key will consist of two parts: (i) a coset state tuple generated using the randomness $F(K'', id)$ for random $id$, similar to our PKE scheme; (ii) an obfuscated signing program $\mathsf{PSign}_K$. The program $\mathsf{PSign}_K$ will take as input a message $m$, along with $id$ and vectors $(v_i)_{i \in [c]}$. Similar to the ciphertext programs $\mathsf{PCt}$ in our PKE construction, $\mathsf{PSign}_K$ will verify that the vectors $(v_i)_{i \in [c]}$ are in the correct cosets $A_i + s_i$ or $A_i^{\perp} + s'_i$, depending on the $i$-th bit of $m$, where the tuple $(A_i, s_i, s'_i)$ is associated with $id$. Informally, since the challenge messages $m^1, m^2$ are random, similar to $r^1, r^2$ in the PKE case, we will be able to violate the monogamy-of-entanglement game, given freeloader adversaries that can sign these message - arriving at a

[19] Note that $x$ being randomized and being revealated after the splitting is required, since otherwise the pirate can evaluate the PRF before splitting into freeloaders, and it can simply give the classical evaluation results to the freeloaders.

[20] As in the case of PRFs, known/deterministic messages can be signed before the split by the pirate, hence, random challenge messages are required.

contradiction. However, since we are using $i\mathcal{O}$ and not black-box obfuscation to obfuscate $\mathsf{PSign}_K$, some information about $K$ might leaking, allowing the adversary to sign messages without querying the program with correct vectors. Similar to [13,21], we use the *hidden trigger technique* of [26] to solve this issue and reduce the security of our signature scheme to that of our copy-protected PKE scheme.

Hidden triggers, introduced by [26] to construct deniable encryption, is a sparse set of inputs that can be efficiently sampled and are pseudorandom, even given a program that *uses* these inputs. In the case of [13,21], their set of hidden trigger inputs are special encodings of the ciphertexts of their copy-protected PKE scheme. Using this technique, they embed a separate thread in $\mathsf{PSign}_K$ that detects if the message $m$ is a trigger input, and in that case, executes the embedded ciphertext program $\mathsf{PCt}$ (which is a PKE encryption of $F(K, m)$) in this input instead of normal execution. This allows them to reduce the task of finding the signature $F(K, m)$ for a message $m$ to the task of decrypting a PKE ciphertext encrypting $F(K, m)$ (hence reducing security to their PKE scheme), by undetectably replacing the random challenge messages to be signed with hidden triggers.

In our case, two new issues arise. First, as discussed, to achieve collusion-resistance, our PKE ciphertext programs crucially output $\mathsf{IBE}$ ciphertexts upon successful coset vector verification, meaning that they are randomized programs, which makes it more challenging to encode them as hidden triggers. We solve this issue as follows. Inside the ciphertext program, we expand the hidden signature $F(K, m)$ using a PRG, and we use part of the expanded string as a PRF key to supply randomness to $\mathsf{IBE.Encrypt}$.

Secondly, the previous work [13,21] crucially rely on puncturing the PRF key $K$ at all the challenge points $m_1, \ldots, m_{k+1}$, to replace these challenge messages with hidden trigger inputs and utilize the hidden thread in $\mathsf{PSign}$. However, in our case, we would have to puncture PRF key at $k+1$ points since we have $k+1$ freeloaders/challenges, where $k$ is not a-priori bounded - this is not possible since the sizes of the punctured key and the obfuscated programs would need to grow with $k$. We solve this issue by making our hidden trigger inputs *publicly generatable*, that is, by arguing that hidden trigger inputs are indistinguishable from uniform strings even given a program that generates these inputs (which needs to include the PRF key $K$). This allows us to only prove that a single challenge message is indistinguishable from a single hidden trigger input, and then we simply rely on the hybrid lemma to conclude the same result for any number of challenge messages, since the trigger inputs can now be generated by the adversary itself during the hybrid lemma argument. We use a new *prefix-puncturing* argument for the PRF key $K$ to achieve publicly-generatable hidden triggers for our scheme. The full proof is technical, we refer the reader to the full version [11] for the full constructions and the full proofs.

### 2.5 Impossibility of Hyperefficient Shadow Tomography

As discussed in the introduction, an important corollary of our result is the impossibility of hyperefficient shadow tomography. Suppose a shadow tomography procedure exists. We now describe a generic attack on copy-protection, given by [3] and adapted to the case of copy-protecting decryption keys by [27], that uses shadow tomography. Let $s(\lambda)$ be the size of the ciphertexts of a public-key encryption scheme $\mathsf{PKE}$ with collusion-resistant copy-protected secret keys, for 1-bit messages. Then, define the set of measurements $\{E_{ct}\}_{ct \in \{0,1\}^{s(\lambda)}}$ as follows: $E_{ct}$ is the binary measurement $\mathsf{PKE.Dec}(\cdot, ct)$. That is, given a state (which will be a copy-protected secret key in our case), $E_{ct}$ is the binary measurement implemented by running $\mathsf{PKE.Dec}$ on $\rho$ and accepting if it outputs 1. Then, it is easy to see that once we obtain the estimates of the acceptance probabilities of all $E_{ct}$ for the state $\rho$ where $\rho$ is the copy-protected secret key, when we are given a ciphertext $ct$, we can simply use this estimate to tell if $ct$ is an encryption of 1 or 0, since $E_{ct}$ would *accept* $\rho$ if $ct$ is an encryption of 1, and reject it otherwise. Since these estimates are classical values, given some number of keys we can perform shadow tomography and then we can create any number of decryption programs.

The attack above is used by [3,27] to conclude that *unconditional* collusion-resistant copy-protection is impossible, since [3] gives a shadow tomography procedure that uses polynomially many copies of a state $\rho$, however, the procedure takes exponential time. Now, the question is, does there exist a *hyperefficient* shadow tomography procedure? We observe that the measurement set $\{E_{ct}\}_{ct \in \{0,1\}^{s(\lambda)}}$ above is actually implemented by a uniform algorithm: $\mathsf{PKE.Dec}(\cdot, \cdot)$. Hence, if there exists a hyperefficient shadow tomography procedure, it would output (a classical description of) a quantum circuit that can estimate all $E_{ct}$, given time and number of copies that are both $\mathsf{poly}\left(|\rho|, \log\left(2^{s(\lambda)}\right)\right) = \mathsf{poly}(\lambda)$. However, this would break the security of our collusion resistant copy-protected PKE scheme, since we can query for sufficiently many keys, perform the shadow tomography and freely distribute the resulting classical information. Thus, we conclude that hyperefficient shadow tomography is not possible. We refer the reader to the full version [11] for the formal statements and proofs.

## 3 Public-Key Encryption with Copy-Protected Secret Keys

In this section, we define public-key encryption with copy-protected secret keys. Then, we give our construction based on coset states and prove it secure.

### 3.1 Definitions

**Definition 1 (Public-key Encryption with Copy-Protected Secret Keys).** *A public-key encryption scheme with copy-protected secret keys consists of the following efficient algorithms.*

- $\mathsf{KeyGen}(1^\lambda)$: *Takes in the security parameter, output a classical secret key $sk$ and a public key $pk$.*
- $\mathsf{QKeyGen}(sk)$: *Takes as input the classical secret key and outputs a quantum secret key.*
- $\mathsf{Enc}(pk, m)$: *Takes in the public key and a message $m \in \mathcal{M}$, outputs and encryption of $m$.*
- $\mathsf{Dec}(\mathsf{R}_{\mathsf{dec}}, ct)$: *Takes in a quantum secret key and a ciphertext, outputs a message or $\perp$.*

*We require correctness*[21] *and* CPA security*.*

*Correctness. For all messages $m \in \mathcal{M}$,*

$$\Pr\left[\mathsf{Dec}(\mathsf{R}_{\mathsf{dec}}, ct) = m : \begin{array}{c} pk, sk \leftarrow \mathsf{Setup}(1^\lambda) \\ \mathsf{R}_{\mathsf{dec}} \leftarrow \mathsf{QKeyGen}(sk) \\ ct \leftarrow \mathsf{Enc}(pk, m) \end{array}\right] = 1.$$

*CPA Security. For any stateful QPT adversary $\mathcal{A}$,*

$$\Pr\left[\mathcal{A}(ct) = b : \begin{array}{c} pk, sk \leftarrow \mathsf{Setup}(1^\lambda) \\ m_0, m_1 \leftarrow \mathcal{A}(pk, 1^\lambda) \\ b \leftarrow \{0,1\} \\ ct \leftarrow \mathsf{Enc}(pk, m_b) \end{array}\right] \leq \frac{1}{2} + \mathsf{negl}(\lambda).$$

As observed by [13], correctness of the scheme along with Almost As Good As New Lemma [2] means that we can implement decryption in a way such that the quantum secret key is not disturbed. Thus, we can reuse the key to decrypt any number of times.

Following prior work, we will use two different security notions, regular anti-piracy and strong anti-piracy. The former will be the natural security notion while the latter definition is easier to work with when proving security. Both of our definitions follow [13,21], with the strengthening that we allow unbounded[22] number of key queries and we also allow the adversary to choose different challenge messages for each freeloader.

Now, we move onto the first definition. In this definition, the pirate (or *splitting*) adversary queries for copy-protected keys for any number of rounds. Then, if it has queried for $k$ keys, it outputs $k+1$ freeloaders, which are unitaries along with hardwired quantum states. More precisely, it outputs a $(k+1)$-partite (possibly entangled) register $\mathsf{R}_{\mathsf{adv}}$ and unitaries $U_\ell$. Then, the challenger presents these freeloaders with challenge ciphertexts, and the adversary wins if all freeloaders correctly predict the challenges. Below, we write $\mathsf{U}_{quantum}$ to denote the quantum universal circuit $\mathsf{U}_{quantum}((U, \rho), x)$ that takes in a unitary $U$ and a state $\rho$, and simulates the *induced* quantum circuit on input $x$ (i.e. computes $U(\rho, x)$),

[21] While our schemes satisfy perfect correctness, i.e., correctness with probability 1, some work relax the definition to $1 - \mathsf{negl}(\lambda)$.

[22] Still polynomial since the adversary is QPT.

and finally measures the first output qubit in the computational basis. We note that the freeloaders being unitaries is not restrictive and actually captures general quantum circuits since the hardwired quantum state $(\mathsf{R}_{\mathsf{adv}})_\ell$ can include[23] workspace qubits initialized to zeroes.

**Definition 2 (CPA-Style Regular $\gamma$-Anti-Piracy Security).** *Let* $\mathsf{PKE}$ *be a public key encryption scheme with copy-protected secret keys. Consider the following game between the challenger and an adversary* $\mathcal{A}$.

$\underline{\mathsf{PKEAntiPiracy}(\lambda, \mathcal{A})}$

1. *The challenger runs* $sk, pk \leftarrow \mathsf{PKE.Setup}(1^\lambda)$ *and submits* $pk$ *to the adversary.*
2. *For multiple rounds,* $\mathcal{A}$ *makes quantum key queries. For each query, the challenger generates a key as* $\mathsf{R} \leftarrow \mathsf{PKE.QKeyGen}(sk)$ *and submits* $\mathsf{R}$ *to the adversary.*
3. $\mathcal{A}$ *outputs a* $(k+1)$*-partite register* $\mathsf{R}_{\mathsf{adv}}$*, unitaries* $\{U_\ell\}_{\ell \in [k+1]}$ *and challenge messages* $\{m_\ell^0, m_\ell^1\}_{\ell \in [k+1]}$*, where* $k$ *is the number of queries it made.*
4. *The challenger executes the following for each* $\ell \in [k+1]$.
   1. $b_\ell \leftarrow \{0,1\}$.
   2. $ct_\ell \leftarrow \mathsf{PKE.Enc}(pk, m_\ell^{b_\ell})$.
   3. $b'_\ell \leftarrow \mathsf{U}_{quantum}(U_\ell, \mathsf{R}_{\mathsf{adv}}[\ell], ct_\ell)$.
   4. *Check if* $b'_\ell = b_\ell$.
5. *The challenger outputs 1 if and only if all the checks pass.*

*We say that* $\mathsf{PKE}$ *satisfies* $\gamma$*-anti-piracy security if for any QPT adversary* $\mathcal{A}$,

$$\Pr[\mathsf{PKEAntiPiracy}(\lambda, \mathcal{A}) = 1] \leq \frac{1}{2} + \gamma(\lambda) + \mathsf{negl}(\lambda).$$

*We ignore writing* $\gamma$ *when* $\gamma = 0$.

We can also define a stronger notion called CPA-style strong $\gamma$-anti-piracy [13], where $\gamma$ is a parameter - we refer the reader to the full version [11] for the formal definition. CPA-style regular $\gamma$-anti-piracy for any inverse polynomial $\gamma$ implies regular CPA security and regular $\gamma$-anti-piracy for $\gamma = 0$ [11,13].

## 3.2 Construction

In this section, we present our construction. Assume the existence of following primitives where we set $\nu(\lambda) = 2^{-6\lambda} \cdot 2^{-8\lambda^{0.3 C_{\mathsf{MoE.Coll}}}}$.

- $i\mathcal{O}$, indistinguishability obfuscation scheme that is $\nu(\lambda)$-secure against $2^{5\lambda} \cdot 2^{8\lambda^{0.3 C_{\mathsf{MoE.Coll}}}}$-time adversaries,
- $\mathsf{IBE}$, identity-based encryption scheme for the identity space $\mathcal{ID} = \{0,1\}^\lambda$ that is $\nu(\lambda)$-secure against $2^{5\lambda} \cdot 2^{8\lambda^{0.3 C_{\mathsf{MoE.Coll}}}}$-time adversaries,

[23] It will also include some quantum information that the pirate adversary has produced from the copy-protected keys.

– $F_1$, puncturable PRF family with input length $\lambda$ and output length same as the size of the randomness used by $\mathsf{CosetGen}$ that is $\nu(\lambda)$-secure against $2^{5\lambda} \cdot 2^{8\lambda^{0.3C_{\mathsf{MoE.Coll}}}}$-time adversaries,
– $F_2$, puncturable PRF family with input length $\lambda$ and output length same as the size of the randomness used by $\mathsf{IBE.Enc}$ that is $\nu(\lambda)$-secure against $2^{5\lambda} \cdot 2^{8\lambda^{0.3C_{\mathsf{MoE.Coll}}}}$-time adversaries,
– $\mathsf{CCObf}$, compute-and-compare obfuscation for $2^{-\lambda^{0.2\cdot C_{\mathsf{MoE.Coll}}}}$-unpredictable distributions that is $2^{-2\lambda-1} \cdot 2^{-2\lambda^{0.3C_{\mathsf{MoE.Coll}}}}$-secure against $2^{3\lambda} \cdot 2^{2\lambda^{0.3C_{\mathsf{MoE.Coll}}}}$-time adversaries,

A remark is in order regarding our assumptions. We note that all of our assumptions above can be based on any subexponential $i\mathcal{O}$ and LWE assumption. For example, if we have an $i\mathcal{O}$ scheme that is $2^{-\lambda^{c_1}}$-secure against $2^{\lambda^{c_2}}$-time adversaries; in our construction we implicitly initiate it with security parameter $\lambda^{c'}$ where $c' = \max\{0.3C_{\mathsf{MoE.Coll}}/c_1, 0.3C_{\mathsf{MoE.Coll}}/c_2\}$. While this might require larger padding for obfuscated circuits, this is still within polynomial factors. The same applies for the other primitives. Thus, our assumptions can be based solely on subexponential hardness for any exponent, since we can always scale the security parameter by a polynomial factor when instantiating the underlying primitives.

Set $L(\lambda) = \lambda$ and therefore $c_L(\lambda) = 24\cdot\lambda^3$. We also assume that all obfuscated programs in the construction and in the proof are appropriately padded.

We now give our construction for public-key encryption with copy-protected secret keys.

$\underline{\mathsf{PKE.Setup}(1^\lambda)}$

1. Sample a PRF key $K_1 \leftarrow F_1.\mathsf{KeyGen}(1^\lambda)$.
2. Sample $cpk, csmk \leftarrow \mathsf{IBE.Setup}(1^\lambda)$.
3. Sample $\mathsf{OPMem} \leftarrow i\mathcal{O}(\mathsf{PMem}_{K_1})$, where $\mathsf{PMem}_{K_1}$ is the following program.

```
$\underline{\mathsf{PMem}_{K_1}(id, u_1, \ldots, u_{c_L(\lambda)}, r)}$
**Hardcoded:** $K_1$
(a) $(A_i, s_i, s'_i)_{i\in[c_L(\lambda)]} \leftarrow \mathsf{CosetGen}(1^{L(\lambda)+\lambda}; F_1(K_1, id))$.
(b) For each $i \in [c_L(\lambda)]$, check if $u_i \in A_i + s_i$ if $(r)_i = 0$ and check
    if $u_i \in A_i^\perp + s'_i$ if $(r)_i = 1$. If any of the checks fail, output 0 and
    terminate.
(c) Output 1.
```

4. Set $pk = (cpk, \mathsf{OPMem})$ and $sk = (cmsk, K_1)$.
5. Output $(pk, sk)$.

$\underline{\mathsf{PKE.QKeyGen}(sk)}$

1. Parse $(cmsk, K_1) = sk$.
2. Sample $id \leftarrow \{0,1\}^\lambda$.
3. $(A_i, s_i, s'_i)_{i\in[c_L(\lambda)]} = \mathsf{CosetGen}(1^{L(\lambda)+\lambda}; F_1(K_1, id))$.
4. $ck \leftarrow \mathsf{IBE.KeyGen}(cmsk, id)$.

5. Output $\left(|A_{i,s_i,s'_i}\rangle\right)_{i\in[c_L(\lambda)]}, ck, id$.

```
PKE.Enc(pk, m)
1. Parse $(cpk, \mathsf{OPMem}) = pk$.
2. Sample $r \leftarrow \{0,1\}^{c_L(\lambda)}$.
3. Sample a PRF key $K_2$ for $F_2$ as $K_2 \leftarrow F_2.\mathsf{KeyGen}(1^\lambda)$.
4. Sample $\mathsf{OPCt} \leftarrow i\mathcal{O}(\mathsf{PCt}_{\mathsf{OPMem},cpk,K_2,r,m})$, where $\mathsf{PCt}_{\mathsf{OPMem},cpk,K_2,r,m}$ is the
   following program.

   $\mathsf{PCt}_{\mathsf{OPMem},cpk,K_2,r,m}(id, u_1, \ldots, u_{c_L(\lambda)})$
   Hardcoded: $\mathsf{OPMem}, cpk, K_2, r, m$
   (a) Run $\mathsf{OPMem}(id, u_1, \ldots, u_{c_L(\lambda)}, r)$. If it outputs 0, output $\perp$ and terminate.
   (b) Output $\mathsf{IBE.Enc}(cpk, id, m; F_2(K_2, id))$.

5. Output $(\mathsf{OPCt}, r)$.
```

```
PKE.Dec(R_key, ct)
1. Parse $((\mathsf{R}_i)_{i\in[c_L(\lambda)]}, ck, id) = \mathsf{R}_{\mathsf{key}}$ and $(\mathsf{OPCt}, r) = ct$.
2. For indices $i \in [c_L(\lambda)]$ such that $(r)_i = 1$, apply $H^{\otimes\kappa(L(\lambda)+\lambda)}$ to $\mathsf{R}_i$.
3. Run the program $\mathsf{OPCt}$ coherently on $id$ and $(\mathsf{R}_i)_{i\in[c_L(\lambda)]}$.
4. Measure the output register and denote the outcome by $cct$.
5. Output $\mathsf{IBE.Dec}(ck, cct)$.
```

Correctness with probability 1 follows in a straightforward manner from the correctness of the underlying schemes. We claim that the construction is also secure.

**Theorem 6.** PKE *satisfies strong $\gamma$-anti-piracy for any inverse polynomial $\gamma$.*

*Proof.* We refer the reader to the full version [11] for the proof.

When we instantiate the assumed building blocks with known constructions, we get the following corollary.

**Corollary 3.** *Assuming subexponentially secure $i\mathcal{O}$ and subexponentially secure LWE, there exists a public-key encryption scheme that satisfies anti-piracy security against unbounded collusion.*

*Proof.* IBE can be constructed based on $i\mathcal{O}$ and one-way functions. $F_1$ and $F_2$ can be constructed based on one-way functions. Which in turn can be obtained from LWE. CCObf can be constructed based on $i\mathcal{O}$ and LWE.

## 4 Public-Key Functional Encryption with Copy-Protected Functional Keys

In this section, we define functional encryption with copy-protected functional keys. Then, we give a construction based on coset states and prove it secure.

### 4.1 Definitions

An informal overview of our security model is as follows. The piracy adversary will be allowed to adaptively query for classical (i.e., not copy-protected) and copy-protected (i.e. quantum) functional keys. At the end of this first query phase, the adversary will produce a pair of challenge messages $m^0, m^1$ and $k+1$ registers (*freeloaders*) where $k$ is the number of copy-protected keys obtained by it. After this split, the challenger presents the freeloaders each with a challenge ciphertext. Finally, after receiving the challenge ciphertexts, freeloaders can query for more functional keys, and they output their guess at the end.

We will also require the following for the challenge message pair $m^0, m^1$ and the functions queried. First, we require that $f(m^0) = f(m^1)$ for all functions $f$ queried by the pirate in the *classical mode*. This is required since, otherwise, the pirate can give all the freeloaders the classical key $sk_f$, and they can decrypt their challenge ciphertexts with this key to distinguish $\mathsf{Enc}(m_0)$ vs $\mathsf{Enc}(m_1)$. Second, for the same reason as above, we require that a freeloader can query a key for $f$ only if $f(m^0) = f(m^1)$. Note that these requirements are the same as the classical FE game. Importantly, we will not require anything for functional keys that were obtained in the *copy-protected mode* by the pirate adversary before the split. Thus, our security guarantee will allow $k$ out of the $k+1$ freeloaders to possibly use these copy-protected functional keys to decrypt their challenge ciphertexts. However, it should not be possible for all $k+1$ registers to use these copy-protected keys simultaneously.

We also define our model so that copy-protected functional keys are generated given only a classical functional key, without any extra information. Therefore, we do not need to separately require that a copy-protected key for $f$ allows no more than obtaining $f(m)$ given $\mathsf{Enc}(m)$, which is already implied by the regular functional encryption security.

**Definition 3 (Public-key Functional Encryption with Copy-Protected Secret Keys).** *A public-key functional encryption scheme with copy-protected secret keys is a public-key functional encryption scheme with the following additional algorithm and guarantee.*

- $\mathsf{QKeyGen}(fk)$*: Takes as input a classical functional key, outputs a quantum secret key.*

*We require correctness*[24] *for the quantum functional keys.*

*Correctness. For all messages $m \in \mathcal{M}$,*

$$\Pr\left[\mathsf{Dec}(\mathsf{R}_{\mathsf{dec}}, ct) = f(m) : \begin{array}{c} pk, msk \leftarrow \mathsf{Setup}(1^\lambda) \\ sk_f \leftarrow \mathsf{KeyGen}(msk, f) \\ \mathsf{R}_f \leftarrow \mathsf{QKeyGen}(sk_f) \\ ct \leftarrow \mathsf{Enc}(pk, m) \end{array}\right] = 1.$$

[24] While our schemes satisfy perfect correctness, i.e., correctness with probability 1, some work relax the definition to $1 - \mathsf{negl}(\lambda)$.

As discussed in Sect. 3, correctness of the scheme along with As Good As New Lemma [2] means that we can implement decryption in a way such that the quantum functional key is not disturbed. Thus, we can reuse the key to decrypt any number of times.

Similar to public-key encryption, we define a CPA-style anti-piracy security definition. We refer the reader to the full version [11] for the formal definition.

## 4.2 Construction

In section, we give our construction of a functional encryption scheme with copy-protected keys for the class of functions $\mathfrak{F}$ defined as all circuits that are of size at most $Q(\lambda)$, where $Q(\lambda)$ is any fixed polynomial. The construction is highly similar to our public-key encryption construction. The main difference is that a functional key for a function $f$ will consist of an $\mathsf{IBE}$ key for $id||f$ where $id$ is a random string.

Assume the existence of following primitives where we set $\nu(\lambda) = 2^{-5\lambda - Q(\lambda)} \cdot 2^{-8\lambda^{0.3C_{\mathsf{MoE.Coll}}}}$.

- $i\mathcal{O}$, indistinguishability obfuscation scheme that is $\nu(\lambda)$-secure against $2^{5\lambda} \cdot 2^{8\lambda^{0.3C_{\mathsf{MoE.Coll}}}}$-time adversaries,
- $\mathsf{IBE}$, identity-based encryption scheme with puncturable master secret keys and deterministic $\mathsf{KeyGen}$ that satisfies strong punctured key correctness, for the identity space $\mathcal{ID} = \{0,1\}^{Q(\lambda)+\lambda}$ that is $\nu(\lambda)$-secure against $2^{5\lambda} \cdot 2^{8\lambda^{0.3C_{\mathsf{MoE.Coll}}}}$-time adversaries,
- $F_1$, puncturable PRF family with input length $Q(\lambda) + \lambda$ and output length same as the size of the randomness used by $\mathsf{CosetGen}$ that is $\nu(\lambda)$-secure against $2^{5\lambda} \cdot 2^{8\lambda^{0.3C_{\mathsf{MoE.Coll}}}}$-time adversaries,
- $F_2$, puncturable PRF family with input length $Q(\lambda) + \lambda$ and output length same as the size of the randomness used by $\mathsf{IBE.Enc}$ that is $\nu(\lambda)$-secure against $2^{5\lambda} \cdot 2^{8\lambda^{0.3C_{\mathsf{MoE.Coll}}}}$-time adversaries,
- $\mathsf{CCObf}$, compute-and-compare obfuscation for $2^{-\lambda^{0.2 \cdot C_{\mathsf{MoE.Coll}}}}$-unpredictable distributions that is $2^{-2\lambda-1} \cdot 2^{-2\lambda^{0.3C_{\mathsf{MoE.Coll}}}}$-secure against $2^{3\lambda} \cdot 2^{2\lambda^{0.3C_{\mathsf{MoE.Coll}}}}$-time adversaries,

Similar to our public-key encryption scheme, while we assume exponential security of the above primitives for specific exponents, these assumptions can be based only on subexponential hardness for some exponent, since we can always scale the security parameter by a polynomial factor.

Also, set $L(\lambda) = Q(\lambda) + \lambda$ and hence $c_L(\lambda) = 3 \cdot (Q(\lambda) + 2\lambda)^3$.

We now give our construction. Below, assume that all programs that are obfuscated are appropriately padded.

$\underline{\mathsf{FE.Setup}(1^\lambda)}$

1. Sample a PRF key $K_1 \leftarrow F_1.\mathsf{KeyGen}(1^\lambda)$.
2. Sample $cpk, csmk \leftarrow \mathsf{IBE.Setup}(1^\lambda)$.
3. Sample $\mathsf{OPMem} \leftarrow i\mathcal{O}(\mathsf{PMem}_{K_1})$, where $\mathsf{PMem}_{K_1}$ is the following program.

```
$\underline{\mathsf{PMem}_{K_1}(id||f, u_1, \ldots, u_{c_L(\lambda)}, r)}$
Hardcoded: $K_1$
(a) $(A_i, s_i, s'_i)_{i \in [c_L(\lambda)]} \leftarrow \mathsf{CosetGen}(1^{L(\lambda)+\lambda}; F_1(K_1, id||f))$.
(b) For each $i \in [c_L(\lambda)]$, check if $u_i \in A_i + s_i$ if $(r)_i = 0$ and check
    if $u_i \in A_i^{\perp} + s'_i$ if $(r)_i = 1$. If any of the checks fail, output 0 and
    terminate.
(c) Output 1.
```

4. Set $pk = (cpk, \mathsf{OPMem})$, $msk = (cmsk, K_1)$.
5. Output $(pk, msk)$.

$\underline{\mathsf{FE.KeyGen}(msk, f)}$

1. Parse $(cmsk, K_1) = msk$.
2. Sample $id \leftarrow \{0,1\}^{\lambda}$.
3. Sample $ck \leftarrow \mathsf{IBE.KeyGen}(cmsk, id||f)$.
4. $(A_i, s_i, s'_i)_{i \in [c_L(\lambda)]} = \mathsf{CosetGen}(1^{L(\lambda)+\lambda}; F_1(K_1, id||f))$.
5. Output $(ck, id, f, (A_i, s_i, s'_i)_{i \in [c_L(\lambda)]})$.

$\underline{\mathsf{FE.QKeyGen}(fk)}$

1. Parse $(ck, id, f, (A_i, s_i, s'_i)_{i \in [c_L(\lambda)]}) = fk$.
2. Output $\left(|A_{i,s_i,s'_i}\rangle\right)_{i \in [c_L(\lambda)]}, ck, id, f$.

$\underline{\mathsf{FE.Enc}(pk, m)}$

1. Parse $(cpk, \mathsf{OPMem}) = pk$.
2. Sample $r \leftarrow \{0,1\}^{c_L(\lambda)}$.
3. Sample a PRF key $K_2$ for $F_2$ as $K_2 \leftarrow F_2.\mathsf{KeyGen}(1^{\lambda})$.
4. Sample $\mathsf{OPCt} \leftarrow i\mathcal{O}(\mathsf{PCt}_{\mathsf{OPMem}, cpk, K_2, r, m})$, where $\mathsf{PCt}_{\mathsf{OPMem}, cpk, K_2, r, m}$ is the following program.

```
$\underline{\mathsf{PCt}_{\mathsf{OPMem}, cpk, K_2, r, m}(id||f, u_1, \ldots, u_{c_L(\lambda)})}$
Hardcoded: $\mathsf{OPMem}, cpk, K_2, r, m$
(a) Run $\mathsf{OPMem}(id||f, u_1, \ldots, u_{c_L(\lambda)}, r)$. If it outputs 0, output $\perp$ and
    terminate.
(b) Output $\mathsf{IBE.Enc}(cpk, id||f, f(m); F_2(K_2, id||f))$.
```

5. Output $(\mathsf{OPCt}, r)$.

$\underline{\mathsf{FE.Dec}(\mathsf{R}_{\mathsf{key}}, ct)}$

1. Parse $((\mathsf{R}_i)_{i \in [c_L(\lambda)]}, ck, id, f) = \mathsf{R}_{\mathsf{key}}$ and $(\mathsf{OPCt}, r) = ct$.
2. For indices $i \in [c_L(\lambda)]$ such that $(r)_i = 1$, apply $H^{\otimes \kappa(L(\lambda)+\lambda)}$ to $\mathsf{R}_i$.
3. Run the program $\mathsf{OPCt}$ coherently on $id, f$ and $(\mathsf{R}_i)_{i \in [c_L(\lambda)]}$.
4. Measure the output register and denote the outcome by $cct$.
5. Output $\mathsf{IBE.Dec}(ck, cct)$.

Correctness with probability 1 follows in a straightforward manner from the correctness of the underlying schemes. We claim that the construction is also secure.

**Theorem 7.** $\mathsf{FE}$ *satisfies $\gamma$-anti-piracy for any inverse polynomial $\gamma$.*

*Proof.* We refer the reader to the full version [11] for the proof

When we instantiate the assumed primitives with known constructions, we get the following corollary.

**Corollary 4.** *Assuming subexponentially secure $i\mathcal{O}$ and subexponentially secure LWE, there exists a public-key functional encryption scheme that satisfies anti-piracy security against unbounded collusion.*

*Proof.* We refer the reader to the full version [11] for the proof.

## 5 Signature Scheme with Copy-Protected Keys

In this section, we define signature schemes with copy-protected signing keys. Then, we give our construction based on coset states and prove it secure.

### 5.1 Definitions

**Definition 4 (Signature Scheme with Copy-Protected Secret Keys).** *A signature scheme with copy-protected secret keys consists of the following efficient algorithms.*

– $\mathsf{KeyGen}(1^\lambda)$*: Takes in the security parameter, output a classical signing key $sk$ and a classical verification key $vk$.*
– $\mathsf{QKeyGen}(sk)$*: Takes as input the classical signing key and outputs a quantum signing key.*
– $\mathsf{Sign}(\mathsf{R}_{\mathsf{sk}}, m)$*: Takes in a quantum signing key and a message $m$, outputs a classical signature on $m$.*
– $\mathsf{Ver}(vk, m, sig)$*: Takes in the verification key, a message $m \in \mathcal{M}$ and a claimed signature $sig$ on $m$, outputs $1$ (accept) or $0$ (reject).*

*We require correctness.*

*Correctness. For all messages $m \in \mathcal{M}$,*

$$\Pr\left[\mathsf{Ver}(vk, sig) = 1 : \begin{array}{r} sk, vk \leftarrow \mathsf{Setup}(1^\lambda) \\ \mathsf{R}_{\mathsf{sk}} \leftarrow \mathsf{QKeyGen}(sk) \\ sig \leftarrow \mathsf{Sign}(\mathsf{R}_{\mathsf{sk}}, m) \end{array}\right] \geq 1 - \mathsf{negl}(\lambda).$$

**Definition 5 (Pseudodeterministic Signatures).** *A signature scheme is said to be* pseudodeterministic *if for any value of $sk, vk$ in the support induced by* $\mathsf{KeyGen}$*, for any message $m \in \mathcal{M}$, there exists a fixed signature $sig_{sk,vk,m}$ such that*

$$\Pr\left[sig = sig_{sk,vk,m} : \begin{array}{r} sk, vk \leftarrow \mathsf{Setup}(1^\lambda) \\ \mathsf{R}_{\mathsf{sk}} \leftarrow \mathsf{QKeyGen}(sk) \\ sig \leftarrow \mathsf{Sign}(\mathsf{R}_{\mathsf{sk}}, m) \end{array}\right] \geq 1 - \mathsf{negl}(\lambda).$$

As observed by [21], a pseudodeterministic signature scheme, along with As Good As New Lemma [2] means that we can implement the signing in a way such that the quantum secret key is only negligibly disturbed. Thus, we can reuse the key to sign any polynomial number of times. Our scheme (Sect. 5.2) will be pseudodeterministic.

We now define anti-piracy security for signature schemes, similar to our PKE definition (Definition 2).

**Definition 6 (Anti-Piracy Security for Signature Schemes).** *Let* $\mathsf{DS}$ *be a signature scheme with copy-protected secret keys. Consider the following game between the challenger and an adversary* $\mathcal{A}$. $\underline{\mathsf{SignatureAntiPiracy}(\lambda, \mathcal{A})}$

1. *The challenger runs* $sk, vk \leftarrow \mathsf{DS.Setup}(1^\lambda)$ *and submits* $vk$ *to the adversary.*
2. *For multiple rounds,* $\mathcal{A}$ *makes quantum key queries. For each query, the challenger generates a key as* $\mathsf{R} \leftarrow \mathsf{DS.QKeyGen}(sk)$ *and submits* $\mathsf{R}$ *to the adversary.*
3. $\mathcal{A}$ *outputs a* $(k+1)$*-partite register* $\mathsf{R}_{\mathsf{adv}}$ *and freeloader unitaries* $\{U_\ell\}_{\ell \in [k+1]}$ *where* $k$ *is the number of queries it made.*
4. *The challenger executes the following for each* $\ell \in [k+1]$.
   (a) $m_\ell \leftarrow \mathcal{M}$.
   (b) $sig_\ell \leftarrow \mathsf{U}_{quantum}(U_\ell, \mathsf{R}_{\mathsf{adv}}[\ell], m_\ell)$.
   (c) *Check if* $\mathsf{DS.Ver}(vk, m_\ell, sig_\ell) = 1$.
5. *The challenger outputs 1 if and only if all the checks pass.*

*We say that* $\mathsf{DS}$ *satisfies anti-piracy security if for any QPT adversary* $\mathcal{A}$,

$$\Pr[\mathsf{SignatureAntiPiracy}(\lambda, \mathcal{A}) = 1] \leq \mathsf{negl}(\lambda).$$

## 5.2 Construction

In this section, we present our construction. Assume the existence of following primitives where we set $\nu(\lambda) = 2^{-6\lambda} \cdot 2^{-8\lambda^{0.3 C_{\mathsf{MoE.Coll}}}}$.

- $F$, prefix puncturable extracting PRF with error $2^{-\lambda-1}$ for min-entropy $s_2(\lambda) + s_3(\lambda)$, with input length $m(\lambda)$ and output length $n(\lambda)$,
- $i\mathcal{O}$, indistinguishability obfuscation scheme that is $\nu(\lambda)$-secure against $2^{5\lambda} \cdot 2^{8\lambda^{0.3 C_{\mathsf{MoE.Coll}}}}$-time adversaries,
- $\mathsf{IBE}$, identity-based encryption scheme for the identity space $\mathcal{ID} = \{0,1\}^\lambda$ that is $\nu(\lambda)$-secure against $2^{5\lambda} \cdot 2^{8\lambda^{0.3 C_{\mathsf{MoE.Coll}}}}$-time adversaries,
- $F_1$, puncturable PRF family with input length $\lambda$ and output length same as the size of the randomness used by $\mathsf{CosetGen}$, that is $\nu(\lambda)$-secure against $2^{5\lambda} \cdot 2^{8\lambda^{0.3 C_{\mathsf{MoE.Coll}}}}$-time adversaries,
- $F_2$, puncturable PRF family with input length $\lambda$ and output length same as the size of the randomness used by $\mathsf{IBE.Enc}$ that is $\nu(\lambda)$-secure against $2^{5\lambda} \cdot 2^{8\lambda^{0.3 C_{\mathsf{MoE.Coll}}}}$-time adversaries,[25]

[25] We also assume that $F_2$ has uniformly random keys (when not punctured), that is, the key generation algorithm $F_2.\mathsf{KeyGen}$ simply samples and outputs a uniformly random string. This is satisfied by the puncturable PRF constructions based on one-way functions we are using.

- $\mathsf{CCObf}$, compute-and-compare obfuscation for $2^{-\lambda^{0.2 \cdot C_{\mathsf{MoE.Coll}}}}$-unpredictable distributions that is $2^{-2\lambda-1} \cdot 2^{-2\lambda^{0.3C_{\mathsf{MoE.Coll}}}}$-secure against $2^{3\lambda} \cdot 2^{2\lambda^{0.3C_{\mathsf{MoE.Coll}}}}$-time adversaries,
- $F_3$, puncturable statistically injective PRF with error probability $2^{-\lambda}$ with input length $s_3(\lambda)$ and output length $s_2(\lambda)$,
- $F_4$, puncturable PRF with input length $s_2(\lambda)$ and output length $s_3(\lambda)$,
- $G_1$, a pseudorandom generator with input length $n(\lambda)$ and output length $n(\lambda)$ plus the key size of the PRF $F_2$,
- $G_2$, a pseudorandom generator with input length $s_1(\lambda)/2$ and output length $s_1(\lambda)$,
- $G_3$, a pseudorandom generator with input length $\lambda$ and output length $2 \cdot \lambda$,
- $f$, a subexponentially secure injective one-way function with input space $\{0,1\}^{n(\lambda)}$.

We also set the parameters from above as follows:

- $n(\lambda) = \lambda$,
- $s_1(\lambda) = c_L(\lambda)$,
- $s_3(\lambda) - s_1(\lambda) - 2\lambda$ to be larger than the size of the obfuscations (of the program $Q$) that will be used in the proof,
- $s_2(\lambda) \geq 2 \cdot s_3(\lambda) + \lambda$,
- $s_2(\lambda) + s_3(\lambda) \geq n(\lambda) + 2\lambda + 4$,
- $m(\lambda) = s_1(\lambda) + s_2(\lambda) + s_3(\lambda)$.

As in our other schemes, while some of our security assumptions above are exponential with specific exponents, all of these assumptions can be based solely on subexponential hardness for any exponent, since we can always scale the security parameter by a polynomial factor when instantiating the underlying primitives.

Set $L(\lambda) = \lambda$ and therefore $c_L(\lambda) = 24 \cdot \lambda^3$. We also assume that all obfuscated programs in the construction and in the proof are appropriately padded.

We now give our signature scheme with copy-protected signing keys, for the message space $\mathcal{M} = \{0,1\}^{m(\lambda)}$. $\underline{\mathsf{DS.Setup}(1^\lambda)}$

1. Sample PRF keys $K \leftarrow F.\mathsf{KeyGen}(1^\lambda)$ and $K_i \leftarrow F_i.\mathsf{KeyGen}(1^\lambda)$ for $i \in \{1,3,4\}$.
2. Sample $cpk, csmk \leftarrow \mathsf{IBE.Setup}(1^\lambda)$.
3. Sample $\mathsf{OPVer} \leftarrow i\mathcal{O}(\mathsf{PVer})$ where $\mathsf{PVer}$ is the following program.

```
$\underline{\mathsf{PVer}(m, sig)}$
**Hardcoded:** $K, K_3, K_4$ **Hidden Trigger Check**
(a) Parse $m_1||m_2||m_3 = m$ with $|m_i| = s_i$.
(b) Compute $m_1'||\mathsf{OQ}'||r' = F_4(K_4, m_2) \oplus m_3$.
(c) Check if $m_1' = m_1$ and $m_2 = F_3(K_3, m_1'||\mathsf{OQ}'||r')$. If so, treat $\mathsf{OQ}'$
    as a classical circuit, output $\mathsf{OQ}'(\mathsf{mode} = \mathsf{verify}, sig||0^{c_L(\lambda)\cdot\lambda})$ and
    terminate.
    **Normal Mode**
```

```
(d) Parse $y||K'_2 = G_1(F(K,m))$ with $|y| = n(\lambda)$.
(e) Output 1 if $f(sig) = f(y)$. Otherwise, output 0.
```

4. Sample $\mathsf{OPMem} \leftarrow i\mathcal{O}(\mathsf{PMem}_{K_1})$, where $\mathsf{PMem}_{K_1}$ is the following program.

```
$\mathsf{PMem}_{K_1}(id, u_1, \ldots, u_{c_L(\lambda)}, x)$
Hardcoded: $K_1$
(a) $(A_i, s_i, s'_i)_{i \in [c_L(\lambda)]} \leftarrow \mathsf{CosetGen}(1^{L(\lambda)+\lambda}; F_1(K_1, id))$.
(b) For each $i \in [c_L(\lambda)]$, check if $u_i \in A_i + s_i$ if $(x)_i = 0$ and check
    if $u_i \in A_i^{\perp} + s'_i$ if $(x)_i = 1$. If any of the checks fail, output 0 and
    terminate.
(c) Output 1.
```

5. Sample $\mathsf{OPEval} \leftarrow i\mathcal{O}(\mathsf{PEval})$, where $\mathsf{PEval}$ is the following program.[26]

```
$\mathsf{PEval}(m, id, u_1, \ldots, u_{c_L(\lambda)})$
Hardcoded: $\mathsf{OPMem}, cpk, K, K_3, K_4$ Hidden Trigger Check
(a) Parse $m_1||m_2||m_3 = m$ with $|m_i| = s_i$.
(b) Compute $m'_1||\mathsf{OQ}'||r' = F_4(K_4, m_2) \oplus m_3$.
(c) Check if $m'_1 = m_1$ and $m_2 = F_3(K_3, m'_1||\mathsf{OQ}'||r')$. If so, treat $\mathsf{OQ}'$
    as a classical circuit, output $\mathsf{OQ}'(\mathsf{mode} = \mathsf{eval}, id, u_1, \ldots, u_{c_L(\lambda)})$ and
    terminate.Normal Mode
(d) Run $\mathsf{OPMem}(id, u_1, \ldots, u_{c_L(\lambda)}, m_1)$. If it outputs 0, output $\perp$ and
    terminate.
(e) Parse $y||K'_2 = G_1(F(K,m))$ with $|y| = n(\lambda)$.
(f) Output $\mathsf{IBE.Enc}(cpk, id, y; F_2(K'_2, id))$.
```

6. Set $vk = \mathsf{OPVer}$ and $sk = (cmsk, cpk, K_1, \mathsf{OPEval})$.
7. Output $(vk, sk)$.

$\underline{\mathsf{DS.QKeyGen}(sk)}$

1. Parse $(cmsk, cpk, K_1, \mathsf{OPEval}) = sk$.
2. Sample $id \leftarrow \{0,1\}^{\lambda}$.
3. $(A_i, s_i, s'_i)_{i \in [c_L(\lambda)]} = \mathsf{CosetGen}(1^{L(\lambda)+\lambda}; F_1(K_1, id))$.
4. $ck \leftarrow \mathsf{IBE.KeyGen}(cmsk, id)$.
5. Output $\left(|A_{i,s_i,s'_i}\rangle\right)_{i \in [c_L(\lambda)]}, ck, id, \mathsf{OPEval}$.

$\underline{\mathsf{DS.Sign}(\mathsf{R}_{\mathsf{key}}, m)}$

1. Parse $((\mathsf{R}_i)_{i \in [c_L(\lambda)]}, ck, id, \mathsf{OPEval}) = \mathsf{R}_{\mathsf{key}}$.
2. Parse $m_1||m_2||m_3 = m$ with $|m_i| = s_i$.

[26] Note that it is also possible to put the coset generation PRF key $K_1$ directly inside $\mathsf{OPEval}$ due to the $i\mathcal{O}$ security. However, we elect to use $\mathsf{OPMem}$ to preserve the similarities to our PKE construction.

3. For indices $i \in [c_L(\lambda)]$ such that $(m_0)_i = 1$, apply $H^{\otimes \kappa(L(\lambda)+\lambda)}$ to $\mathsf{R}_i$.
4. Run the program $\mathsf{OPEval}$ coherently on $m, id$ and $(\mathsf{R}_i)_{i \in [c_L(\lambda)]}$.
5. Measure the output register and denote the outcome by $cct$.
6. Output $\mathsf{IBE.Dec}(ck, cct)$.

$\underline{\mathsf{DS.Ver}(vk, m, sig)}$

1. Parse $\mathsf{OPVer} = vk$.
2. Output $\mathsf{OPVer}(m, sig)$.

We claim that the construction is correct and secure.

**Theorem 8.** $\mathsf{DS}$ *satisfies correctness (Definition 4) and psuedodeterminism (Definition 5), and hence reusability.*

**Theorem 9.** $\mathsf{DS}$ *satisfies selective*[27] *message existential unforgeability security.*

**Theorem 10.** $\mathsf{DS}$ *satisfies anti-piracy security (Definition 6).*

We refer the reader to the full version [11] for the proofs. When we instantiate the assumed building blocks with known constructions, we get the following corollary.

**Corollary 5.** *Assuming subexponentially secure $i\mathcal{O}$ and subexponentially secure LWE, there exists a signature scheme that satisfies anti-piracy security against unbounded collusion.*

*Proof.* We refer the reader to the full version [11] for the proof.

**Acknowledgments.** We thank Jiahui Liu for useful discussions. A.C. was supported by the following grants of V.G.: NSF award 1916939, DARPA SIEVE program, a gift from Ripple, a DoE, NETL award, a JP Morgan Faculty Fellowship, a PNC center for financial services innovation award, and a Cylab seed funding award.

## References

1. Aaronson, S.: Quantum copy-protection and quantum money. In: 2009 24th Annual IEEE Conference on Computational Complexity, pp. 229–242 (2009). https://doi.org/10.1109/CCC.2009.42
2. Aaronson, S.: The complexity of quantum states and transformations: from quantum money to black holes (2016). https://doi.org/10.48550/ARXIV.1607.05256, https://arxiv.org/abs/1607.05256
3. Aaronson, S.: Shadow tomography of quantum states. SIAM J. Comput. **49**(5), STOC18-368-STOC18-394 (2018). https://doi.org/10.1137/18M120275X

[27] It can also be made by adaptively secure by complexity leveraging and slightly changing the parameters of the underlying primitives, since we are already assuming subexponentially secure primitives.

4. Aaronson, S., Christiano, P.: Quantum money from hidden subspaces. In: Proceedings of the Forty-Fourth Annual ACM Symposium on Theory of Computing. STOC '12, pp. 41–60. Association for Computing Machinery, New York, NY, USA (2012). https://doi.org/10.1145/2213977.2213983
5. Aaronson, S., Kuperberg, G.: Quantum versus classical proofs and advice. In: Twenty-Second Annual IEEE Conference on Computational Complexity (CCC'07), pp. 115–128. IEEE (2007)
6. Aaronson, S., Liu, J., Liu, Q., Zhandry, M., Zhang, R.: New approaches for quantum copy-protection (2020)
7. Aaronson, S., Liu, J., Liu, Q., Zhandry, M., Zhang, R.: New approaches for quantum copy-protection. In: Malkin, T., Peikert, C. (eds.) CRYPTO 2021. LNCS, vol. 12825, pp. 526–555. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-84242-0_19
8. Ananth, P., La Placa, R.L.: Secure software leasing. In: Canteaut, A., Standaert, F.-X. (eds.) EUROCRYPT 2021. LNCS, vol. 12697, pp. 501–530. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-77886-6_17
9. Barooti, K., et al.: Public-key encryption with quantum keys. In: Rothblum, G., Wee, H. (eds.) TCC 2023. LNCS, vol. 14372, pp. 198–227. Springer, Cham (2023). https://doi.org/10.1007/978-3-031-48624-1_8
10. Bitansky, N., Vaikuntanathan, V.: Indistinguishability obfuscation from functional encryption. J. ACM **65**(6) (2018). https://doi.org/10.1145/3234511
11. Çakan, A., Goyal, V.: Unclonable cryptography with unbounded collusions (2023). https://eprint.iacr.org/2023/1841
12. Çakan, A., Goyal, V., Liu-Zhang, C.D., Ribeiro, J.: Unbounded leakage-resilience and leakage-detection in a quantum world. Cryptology ePrint Archive, Paper 2023/410 (2023), https://eprint.iacr.org/2023/410
13. Coladangelo, A., Liu, J., Liu, Q., Zhandry, M.: Hidden cosets and applications to unclonable cryptography. In: Malkin, T., Peikert, C. (eds.) CRYPTO 2021. LNCS, vol. 12825, pp. 556–584. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-84242-0_20
14. Culf, E., Vidick, T.: A monogamy-of-entanglement game for subspace coset states. Quantum **6**, 791 (2022)
15. Farhi, E., Gosset, D., Hassidim, A., Lutomirski, A., Shor, P.: Quantum money from knots. In: Proceedings of the 3rd Innovations in Theoretical Computer Science Conference. ITCS '12, pp. 276–289. Association for Computing Machinery, New York, NY, USA (2012). https://doi.org/10.1145/2090236.2090260
16. Georgiou, M., Zhandry, M.: Unclonable decryption keys. Cryptology ePrint Archive, Paper 2020/877 (2020). https://eprint.iacr.org/2020/877
17. Goyal, V., Pandey, O., Sahai, A., Waters, B.: Attribute-based encryption for fine-grained access control of encrypted data. In: Proceedings of the 13th ACM Conference on Computer and Communications Security, pp. 89–98 (2006)
18. Khurana, D., Tomer, K.: Commitments from quantum one-wayness (2024)
19. Kitagawa, F., Nishimaki, R.: Functional encryption with secure key leasing. In: Agrawal, S., Lin, D. (eds.) ASIACRYPT 2022. LNCS, vol. 13794, pp. 569–598. Springer, Cham (2022). https://doi.org/10.1007/978-3-031-22972-5_20
20. Kretschmer, W.: Quantum pseudorandomness and classical complexity. In: 16th Conference on the Theory of Quantum Computation, Communication and Cryptography (TQC 2021). Schloss Dagstuhl-Leibniz-Zentrum für Informatik (2021)
21. Liu, J., Liu, Q., Qian, L., Zhandry, M.: Collusion resistant copy-protection for watermarkable functionalities. Cryptology ePrint Archive, Paper 2022/1429 (2022). https://eprint.iacr.org/2022/1429

22. Lutomirski, A., et al.: Breaking and making quantum money: toward a new quantum cryptographic protocol (2009)
23. Marriott, C., Watrous, J.: Quantum arthur-merlin games. In: Proceedings. 19th IEEE Annual Conference on Computational Complexity, pp. 275–285 (2004). https://doi.org/10.1109/CCC.2004.1313850
24. O'Donnell, R., Wright, J.: Efficient quantum tomography. In: Proceedings of the Forty-Eighth Annual ACM Symposium on Theory of Computing, pp. 899–912 (2016)
25. Sahai, A., Waters, B.: Fuzzy identity-based encryption. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 457–473. Springer, Heidelberg (2005). https://doi.org/10.1007/11426639_27
26. Sahai, A., Waters, B.: How to use indistinguishability obfuscation: deniable encryption, and more. In: Proceedings of the Forty-Sixth Annual ACM Symposium on Theory of Computing. STOC '14, pp. 475–484. Association for Computing Machinery, New York, NY, USA (2014). https://doi.org/10.1145/2591796.2591825
27. Sattath, O., Wyborski, S.: Uncloneable decryptors from quantum copy-protection (2022)
28. Shamir, A.: Identity-based cryptosystems and signature schemes. In: Blakley, G.R., Chaum, D. (eds.) CRYPTO 1984. LNCS, vol. 196, pp. 47–53. Springer, Heidelberg (1985). https://doi.org/10.1007/3-540-39568-7_5
29. Vidick, T., Zhang, T.: Classical proofs of quantum knowledge. In: Canteaut, A., Standaert, F.-X. (eds.) EUROCRYPT 2021. LNCS, vol. 12697, pp. 630–660. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-77886-6_22
30. Wiesner, S.: Conjugate coding. SIGACT News **15**(1), 78-88 (1983). https://doi.org/10.1145/1008908.1008920
31. Zhandry, M.: Schrödinger's pirate: how to trace a quantum decoder. In: Pass, R., Pietrzak, K. (eds.) TCC 2020. LNCS, vol. 12552, pp. 61–91. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-64381-2_3

# Quantum Key-Revocable Dual-Regev Encryption, Revisited

Prabhanjan Ananth[1], Zihan Hu[2](✉), and Zikuan Huang[3]

[1] UCSB, Santa Barbara, USA
prabhanjan@cs.ucsb.edu
[2] EPFL, Lausanne, Switzerland
zihan.hu@epfl.ch
[3] Tsinghua University, Beijing, China
hzk21@mails.tsinghua.edu.cn

**Abstract.** Quantum information can be used to achieve novel cryptographic primitives that are impossible to achieve classically. A recent work by Ananth, Poremba, Vaikuntanathan (TCC 2023) focuses on equipping the dual-Regev encryption scheme, introduced by Gentry, Peikert, Vaikuntanathan (STOC 2008), with key revocation capabilities using quantum information. They further showed that the key-revocable dual-Regev scheme implies the existence of fully homomorphic encryption and pseudorandom functions, with both of them also equipped with key revocation capabilities. Unfortunately, they were only able to prove the security of their schemes based on new conjectures and left open the problem of basing the security of key revocable dual-Regev encryption on well-studied assumptions.

In this work, we resolve this open problem. Assuming polynomial hardness of learning with errors (over sub-exponential modulus), we show that key-revocable dual-Regev encryption is secure. As a consequence, for the first time, we achieve the following results:

- Key-revocable public-key encryption and key-revocable fully-homomorphic encryption satisfying classical revocation security and based on polynomial hardness of learning with errors. Prior works either did not achieve classical revocation or were based on sub-exponential hardness of learning with errors.
- Key-revocable pseudorandom functions satisfying classical revocation from the polynomial hardness of learning with errors. Prior works relied upon unproven conjectures.

## 1 Introduction

Leveraging fundamental principles of quantum information to achieve cryptographic notions, that are otherwise impossible to achieve classically, is an exciting research direction. In the past few years, a dizzying variety of quantum cryptographic primitives, termed as *unclonable* primitives, have been studied. Underlying the unclonable primitives is the no-cloning principle of quantum mechanics [WZ82, Die82] which states that quantum states, unlike classical strings, cannot be copied. The recent surge in the development of unclonable primitives has

E. Boyle and M. Mahmoody (Eds.): TCC 2024, LNCS 15366, pp. 257–288, 2025.
https://doi.org/10.1007/978-3-031-78020-2_9

resulted in innovative approaches to tackle many real-world security challenges, including protection against anti-piracy [Aar09], privacy concerns in blockchain technology [AGKZ20], and provable deletion of cryptographic data from the web [BI20, BL20].

We focus on the task of securely leasing or revoking cryptographic keys using the tools of quantum information. Before precisely stating the problem that we set out to address, let us consider two scenarios: (a) Imagine a manager needing to temporarily delegate their duties, including access to sensitive encrypted data, to their subordinate by sharing cryptographic keys. The challenge is ensuring the subordinate's access is revoked upon the manager's return, a task that is impossible to achieve with classical keys, (b) If a cryptographic key is stolen from a device, unless the attacker has left a trace, it becomes challenging to detect such an attack and report it.

Quantum information presents a unique approach of tackling both of the above aforementioned problems.

Our Focus. A major focus of our work is on protecting decryption keys. Specifically, we focus on the popular dual-Regev public-key encryption scheme of [GPV08] (also, referred to as the GPV encryption scheme), which has inspired the design of many lattice-based cryptographic primitives [BGG+14, Mah18, BDGM20, Qua20]. A key-revocable dual-Regev public-key encryption scheme, first introduced in [APV23], is the same as the dual-Regev scheme except that we have the additional guarantee that the decryption keys can alternately be represented as quantum states. Any user in possession of the quantum decryption key can decrypt ciphertexts just the way he would have been able to do if he had a classical decryption key. The security guarantee stipulates that once the user returns the quantum decryption key, they will lose the ability to decrypt ciphertexts and in particular, we require that the semantic security of dual-Regev encryption still hold. We refer the reader to Sect. 1.1 for a more detailed description of the key-revocable dual Regev public-key encryption scheme.

Key-Revocable Security of Dual-Regev: Motivation. Proving the security of key-revocable dual-Regev encryption could lead to adding key revocation capabilities to other cryptographic primitives. Indeed, [APV23] showed that key-revocable dual-Regev encryption can be leveraged to prove the existence of fully homomorphic encryption and pseudorandom functions equipped with key revocation capabilities. The structure of dual-Regev encryption was crucially exploited in these applications.

There is also an aesthetic reason behind studying this problem. Dual-Regev public-key encryption is an elegant construction that is taught in most graduate classes on lattice-based cryptography. Understanding whether it satisfies key-revocable security is a natural theoretical question.

The work of [APV23] attempted to prove the key-revocable security of dual-Regev encryption. Unfortunately, they were only able to prove the security of this construction based on a new unfounded conjecture. They leave the problem of proving the key-revocable security of dual Regev encryption on concrete computational assumptions as an important open problem. In this same work, inspired by the literature on certified deletion [BI20, HMNY21, BK22], they define a stronger property called classical revocation: instead of the user being asked to return the state, they are only asked to return a classical string that certifies that the quantum decryption key has been deleted. After the state has been deleted, as before, we require the semantic security of dual-Regev encryption to still hold. [APV23] relied upon yet another new conjecture to show that dual-Regev encryption satisfied classical key-revocation security. The reliance on both these conjectures makes the current state of affairs rather unsatisfactory. [APV23] left open the problem of basing key-revocation security of dual-Regev encryption on well-studied cryptographic assumptions.

*Main Result.* In this work, we resolve this open problem. We show the following:

**Theorem 1.** *Assuming polynomial hardness of learning with errors over sub-exponential modulus*[1]*, dual-Regev encryption is key-revocable. Moreover, this scheme satisfies the classical revocation property.*

*Applications.* By combining the above theorem with the applications of key-revocable dual-Regev encryption in [APV23], we obtain the following results:

MAIN APPLICATION: We present the first result of *key-revocable pseudorandom functions* based on the polynomial hardness of learning with errors and also simultaneously satisfies classical revocation property. Prior work by [APV23] relied upon unproven conjectures.

OTHER APPLICATIONS: We also achieve other applications that are in some aspects better than the previous works.

1. We present the first result of *key-revocable public-key encryption* that is based on polynomial hardness of learning with errors and simultaneously satisfies classical revocation property. Prior works by [AKN+23, CGJL23] satisfied one but not the other.
2. We present the first result of *key-revocable fully homomorphic encryption* that is based on polynomial hardness of learning with errors and simultaneously satisfies classical revocation property. Prior work by [CGJL23] achieved this result from sub-exponential hardness of learning with errors.

<u>MAIN TECHNICAL CONTRIBUTION:</u> At the heart of our result is a new search-to-decision reduction that reduces a quantum distinguisher that breaks the semantic security of dual-Regev encryption into a quantum adversary that can solve an inhomogeneous short integer solution (ISIS) problem. Our search-to-decision reduction is qualitatively different from [APV23] who rely upon Goldreich-Levin reduction over large finite fields. In addition to the fact that [APV23] relies

[1] By aggressively setting the parameters, it would suffice to just assume polynomial hardness of learning with errors over quasi-polynomial modulus.

upon a conjecture, their reduction necessarily[2] incurs a loss that is inversely proportional to $q$, where $q$ is the size of the field. Since they need to set $q$ to be sub-exponential in the security parameter, this means that their reduction suffers from sub-exponential loss. On the other hand, our ISIS solver only incurs inverse polynomial loss, independent of $q$.

Related Works: It would be remiss not to discuss two other related prior works.

Chardouvelis, Goyal, Jain, Liu [CGJL23] present instantiations of key-revocable public-key encryption and fully homomorphic encryption. Moreover, their schemes satisfy classical key-revocation security[3]. *There are two advantages of our work over theirs:*

– *They do not have any results on pseudorandom functions,*
– *They assume sub-exponential hardness of learning with errors whereas we only assume polynomial hardness of learning with errors.*

Besides that, our work **fundamentally** differs from their work, both in terms of constructions and its analysis. Let us begin by highlighting the differences in the construction.

At a high level, our construction is the same as the dual-Regev public-key encryption scheme except for the quantum decryption key, whereas [CGJL23] builds a new encryption scheme inspired by noisy trapdoor claw-free functions (NTCF) introduced by [BCM+21]. Specifically, they repeat many instantiations of NTCFs in parallel and use that to build a key-revocable public-key encryption scheme. The NTCF itself is instantiated using the (original) Regev public-key encryption scheme.

Even the overall approach in the analysis is quite different: we do a reduction from decision LWE to SIS whereas they do a search-to-decision reduction for LWE itself. In the analysis, we use Gaussian collapsing lemma [Por22] and introduce a new lemma, lemma 6 which is distinct in our work. Given the fact that the constructions are very different, unsurprisingly, the implementation details also vary quite a bit in both the works. For instance, since they do parallel repetition, their extraction method is more complicated since the adversary could have broken any one of the instantiations.

Agrawal, Kitagawa, Nishimaki, Yamada, Yamakawa [AKN+23] present an instantiation of key-revocable public-key encryption based on the existence of any post-quantum secure public-key encryption scheme. They also present other key-revocable notions, such as functional encryption, that are not covered in this work. *There are two advantages of our work over theirs:*

– *They do not prove the classical key revocation security of their scheme,*

[2] Their starting point is the classical Goldreich Levin reduction over finite fields by Dodis et al. [DGT+10]. This reduction already suffers from a loss that is inversely proportional to $q$.

[3] In fact, they satisfy a much stronger property where the communication with the user can be completely classical.

– *They also do not provide any positive results on either fully homomorphic encryption or pseudorandom functions.*

Both the works, [CGJL23] and [AKN+23], come up with arguably more involved constructions of key-revocable public-key encryption which make it unwieldy to extend their techniques to get new applications.

## 1.1 Technical Overview

In this section, we give an overview of the main ideas and techniques underlying our proofs.

*Key-Revocable Dual-Regev Public-Key Encryption.* We first recall the key-revocable dual-Regev constructions from [APV23]. This part has been reproduced verbatim from their work.

– $\mathsf{KeyGen}(1^\lambda)$: Sample a matrix $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$ and a short trapdoor basis $\mathsf{td}_{\mathbf{A}}$ for it. The (quantum) decryption key is a Gaussian superposition of ISIS solutions, which is generated by the following procedure: Create a Gaussian superposition of short vectors $\mathbf{x}$, compute the image $\mathbf{A}\cdot\mathbf{x} \pmod q$ in the second register to get

$$|\psi\rangle = \sum_{\mathbf{x}\in\mathbb{Z}_q^m} \rho_\sigma(\mathbf{x})|\mathbf{x}\rangle \otimes |\mathbf{A}\cdot\mathbf{x} \pmod q\rangle$$

where $\rho_\sigma(\mathbf{x}) = \exp(-\pi\|\mathbf{x}\|^2/\sigma^2)$ is the Gaussian measure, for some $\sigma > 0$, and measure the second register to the Gaussian coset state

$$|\psi_{\mathbf{y}}\rangle = \sum_{\substack{\mathbf{x}\in\mathbb{Z}_q^m\\ \mathbf{A}\mathbf{x}=\mathbf{y} \pmod q}} \rho_\sigma(\mathbf{x})|\mathbf{x}\rangle$$

for some measurement outcome $\mathbf{y} \in \mathbb{Z}_q^n$.
Finally we set $\mathsf{PK} = (\mathbf{A}, \mathbf{y})$, $\mathsf{MSK} = \mathsf{td}_{\mathbf{A}}$ and $\rho_{\mathsf{SK}} = |\psi_{\mathbf{y}}\rangle$.

– $\mathsf{Enc}(\mathsf{PK}, \mu)$: To encrypt a bit $\mu \in \{0, 1\}$, sample a random string $\mathbf{s} \xleftarrow{\$} \mathbb{Z}_q^n$ together with discrete Gaussian errors $\mathbf{e} \in \mathbb{Z}^m$ and $e' \in \mathbb{Z}$, and output a classical ciphertext $\mathsf{CT}$ given by

$$\mathsf{CT} = \left(\mathbf{s}^\intercal\mathbf{A} + \mathbf{e}^\intercal, \mathbf{s}^\intercal\mathbf{y} + e' + \mu\cdot\lfloor\frac{q}{2}\rfloor\right) \in \mathbb{Z}_q^m \times \mathbb{Z}_q.$$

– $\mathsf{Dec}(\rho_{\mathsf{SK}}, \mathsf{CT})$: First apply the unitary $U : |\mathbf{x}\rangle|0\rangle \to |\mathbf{x}\rangle|\mathsf{CT}\cdot(-\mathbf{x}, 1)^\intercal\rangle$ on input $\rho_{\mathsf{SK}} \otimes |0\rangle\langle 0|$, and then measure the second register in the computational basis. Because $\rho_{\mathsf{SK}}$ is supposed to be the Gaussian coset state $|\psi_{\mathbf{y}}\rangle$, which is a superposition of short vector $\mathbf{x}$ subject $\mathbf{A}\cdot\mathbf{x} = \mathbf{y}$, we obtain an approximation of $\mu\cdot\lfloor\frac{q}{2}\rfloor$ from which we can recover $\mu$.

– $\mathsf{Revoke}(\mathsf{PK}, \mathsf{MSK}, \rho)$ : Apply the projective measurement $\{|\psi_{\mathbf{y}}\rangle\langle\psi_{\mathbf{y}}|, I - |\psi_{\mathbf{y}}\rangle\langle\psi_{\mathbf{y}}|\}$ onto $\rho$ using the master secret key $\mathsf{td}_{\mathbf{A}}$[4]. Output $\mathsf{Valid}$ if the measurement succeeds, and output $\mathsf{Invalid}$, otherwise.

Consider an efficient adversary $\mathcal{A}$. It receives as input a state $|\psi_{\mathbf{y}}\rangle$ from the challenger and computes a state $\rho_{\mathrm{R},\mathrm{Aux}}$ on two registers R and Aux. Subsequently, the adversary returns system R to the challenger, while retaining system Aux as quantum advice for subsequent steps. Informally speaking, we say that the above scheme is secure if $\mathcal{A}$ wins both of the following events simultaneously only with negligible probability:

– $\mathsf{Revoke}$ on the system R outputs $\mathsf{Valid}$.
– Using Aux, $\mathcal{A}$ can distinguish $\left(\mathbf{s}^{\intercal}\mathbf{A} + \mathbf{e}^{\intercal}, \mathbf{s}^{\intercal}\mathbf{y} + e' + \lfloor \frac{q}{2} \rfloor\right)$ versus $(\mathbf{s}^{\intercal}\mathbf{A} + \mathbf{e}^{\intercal}, \mathbf{s}^{\intercal}\mathbf{y} + e')$

*Starting Point.* Inspired by [APV23], we undertake the following approach. Suppose there did exist an efficient adversary $\mathcal{A}$ that is successful in violating the security of the above construction. We reduce $\mathcal{A}$ into an SIS solver $\mathcal{B}$, which is described as follows: it first runs $\mathcal{A}$ on input $(\mathbf{A}, \mathbf{y}, |\psi_{\mathbf{y}}\rangle)$ to obtain a state $\rho$ on two registers R and Aux. Then, $\mathcal{B}$ needs to be cleverly designed in such a way that it recovers a short vector $\mathbf{x_0}$ from R and a short vector $\mathbf{x_1}$ from Aux satisfying the following properties:

– $\mathbf{A}\mathbf{x_0} = \mathbf{y}$, $\mathbf{A}\mathbf{x_1} = \mathbf{y}$ and,
– $\mathbf{x_0} \neq \mathbf{x_1}$.

Once both the vectors $\mathbf{x_0}$ and $\mathbf{x_1}$ are recovered then it simply sets the SIS solution to be $\mathbf{x_0} - \mathbf{x_1}$.

While [APV23] set out on this route, they only managed to show such a reduction based on a new conjecture. The core reason behind this is the fact that it is challenging to be able to *simultaneously* recover two *distinct* short solutions from *two potentially entangled* registers R and Aux. An attempt to recover $\mathbf{x_0}$ from R could invariably disturb the part of the state on Aux such that it is no longer possible to recover $\mathbf{x_1}$. Any approach we undertake should tackle this challenge.

*Our Approach.* We propose a three-step approach to prove the security of key-revocable dual-Regev encryption based on learning with errors.

– Step 1. In the first step, we transform the intermediate state $\rho$ (on R and Aux) produced by $\mathcal{A}$ into a "good state" $\rho_{\mathsf{good}}$. This step doesn't need to always succeed. We require two guarantees here: (a) this step aborts with probability bounded away from 1 and, (b) conditioned on not abort, the output of this step is a good state $\rho_{\mathsf{good}}$ such that the revocation on R succeeds with non-negligible probability and Step 2 works.

[4] [APV23] showed how to implement this projective measurement efficiently with the trapdoor $\mathsf{td}_{\mathbf{A}}$.

– STEP 2. Suppose the output of Step 1 is $\rho_{\mathsf{good}}$. We require that as long as $\rho_{\mathsf{good}}$ is a good state then, from AUX, we should be able to recover a short vector $\mathbf{x_1}$ such that $\mathbf{A}\mathbf{x_1} = \mathbf{y}$. More importantly, we should be able to recover $\mathbf{x_1}$ with overwhelming probability.
– STEP 3. We recover a short vector $\mathbf{x_0}$ from the register R such that $\mathbf{A}\mathbf{x_0} = \mathbf{y}$. Our hope is that $\mathbf{x_0}$ and $\mathbf{x_1}$ are distinct and if this is the case then $\mathbf{x_0} - \mathbf{x_1}$ is a non-trivial short solution in the kernel of $\mathbf{A}$.

The easiest step to realize is Step 3. Suppose we have the guarantee that we can recover $\mathbf{x_1}$ from AUX with overwhelming probability. By invoking *almost as good as new* lemma [Aar16], we can show that the state $\rho$ after Step 2 is not disturbed by much. This means that Revoke still succeeds on R with inverse polynomial probability. This further implies that measuring the register R yields a short vector $\mathbf{x_0}$. Then using a simplified analysis of [APV23], we can argue that $\mathbf{x_0} \neq \mathbf{x_1}$, completing the proof.

We focus our attention on implementing Steps 1 and 2. Our main technical contribution will lie in Step 2.

IMPLEMENTING STEP 1: To implement Step 1, we rely upon the threshold implementation technique introduced by Zhandry [Zha20]. Threshold implementation is a technique employed to get an estimate of the success probability of a POVM on a state. In our context, we employ this technique to test whether the adversary acting upon AUX register of $\rho$ is successful in violating the security of key-revocable dual-Regev encryption scheme. Formally, we define the threshold implementation operator $\mathsf{TI}_{\frac{1}{2}+\gamma}$, where $\gamma$ is some inverse polynomial, with the following properties:

1. $\mathsf{TI}_{\frac{1}{2}+\gamma}$ is akin to a projector-like operator, collapsing the state to a $\gamma$-good state $\rho_{\mathsf{good}}$ capable of distinguishing between $(\mathbf{s}^\intercal\mathbf{A} + \mathbf{e}^\intercal, \mathbf{s}^\intercal\mathbf{y} + e')$ and $(\mathbf{u}, r)$ for $\mathbf{u}, r$ being sampled uniformly randomly with probability $2\gamma$ (referred to as a "$\gamma$-good state") when $\mathsf{TI}_{\frac{1}{2}+\gamma}$ outputs 1, or to some other state when $\mathsf{TI}_{\frac{1}{2}+\gamma}$ outputs 0.
2. For a successful adversary, applying $\mathsf{TI}_{\frac{1}{2}+\gamma}$ with an inverse polynomial $\gamma$ on $\rho_{\text{AUX}}$ results in an output of 1 with noticeable probability.
3. Upon applying $\mathsf{TI}_{\frac{1}{2}+\gamma}$ again on a $\gamma$-good state, it yields an output of 1 with probability 1.

To summarize, as long as $\mathcal{A}$ is a successful adversary, $\mathsf{TI}_{\frac{1}{2}+\gamma}$ collapses $\rho$ into a good state $\rho_{\mathsf{good}}$ with inverse polynomial probability.

IMPLEMENTING STEP 2: As mentioned earlier, implementing Step 2 is our main technical contribution.

It was already shown by [APV23] that $\mathbf{x_1}$ can be extracted from AUX. However, the success probability of their extraction mechanism was only inverse polynomial which is insufficient for our purpose. Instead, we completely depart from [APV23] and propose a novel extraction method. This high-level approach is inspired by [CGJL23] although they study for a completely different construction.

At a high level, our extractor proceeds by guessing each entry of $\mathbf{x_1}$, where $\mathbf{x_1}$ is a short solution mapping $\mathbf{A}$ to $\mathbf{y}$, one coordinate at a time. For each coordinate, we try all possible values and using the distinguisher, test which of our guesses was correct. Recall that there are exponentially many short vectors that map $\mathbf{A}$ to $\mathbf{y}$. But once we apply the Gaussian collapsing lemma [Por22, LMZ23], we can replace the state $|\psi_{\mathbf{y}}\rangle$ with $|\mathbf{x_1}\rangle$. While recovering, say, the $i^{th}$ coordinate of $\mathbf{x_1}$, we use the distinguisher on $\textsc{Aux}$ to figure out whether the guess for the $i^{th}$ coordinate was correct or not. However, this has to be handled with care. Since the distinguisher has quantum auxiliary advice, we cannot keep hoping to run the distinguisher again and again. After the first run, the state of the distinguisher could be damaged making it useless for future iterations. So we need to come up with a mechanism to check if a guess is correct or not while maintaining the quantum state. Making crucial use of threshold implementation along with techniques from lattice-based cryptography, we show how to implement this.

Our extractor is described as follows:

1. Initialize $\mathbf{x} = \mathbf{0}$ as the output register.
2. For each position $i \in [m]$ and each guess $g_i$, we test whether the $i$-th entry $\mathbf{x_1}$ is $g_i$ by:
   (a) Applying $\mathsf{TI}_{\frac{1}{2}+\gamma'}(i, g_i)$ on system $\textsc{Aux}$, where $\mathsf{TI}_{\frac{1}{2}+\gamma'}(i, g_i)$ is a threshold implementation that 'tests' whether the state is $\gamma'$-good at distinguishing between $\left(\mathbf{s}^\intercal\mathbf{A} + \mathbf{e}^\intercal + c \cdot \hat{\mathbf{i}}, \mathbf{s}^\intercal\mathbf{y} + c \cdot g_i + e'\right)$ (where $c \xleftarrow{\$} \mathbb{Z}_q$ and $\hat{\mathbf{i}}$ is the unit vector on the $i$-th dimension) and $(\mathbf{u}, r)$ (where $\mathbf{u} \xleftarrow{\$} \mathbb{Z}_q^m, r \xleftarrow{\$} \mathbb{Z}_q$).
   (b) If the output is 1, set $\mathbf{x}_i = g_i$.
   (c) If the output is 0, skip to the next iteration.
3. Output $\mathbf{x}$.

We argue that our extractor outputs $\mathbf{x}_1$ with nearly perfect probability if $\mathsf{TI}_{\frac{1}{2}+\gamma}$ on $\rho_{\textsc{Aux}}$ outputs 1. Zhandry [Zha20] demonstrates that for two threshold implementations concerning computationally indistinguishable tasks (e.g., distinguishing $(\mathbf{s}^\intercal\mathbf{A} + \mathbf{e}^\intercal, \mathbf{s}^\intercal\mathbf{y} + e')$ from $(\mathbf{u}, r)$, and distinguishing $(\mathbf{u}, \mathbf{u}^\intercal\mathbf{x}_1 + e')$ from $(\mathbf{u}, r)$), their outputs are closely related. Now, considering each guess $g_i$ for position $i$:

– If the guess is correct (i.e., the $i$-th entry of $\mathbf{x}_1$ is $g_i$), the distribution $\left(\mathbf{s}^\intercal\mathbf{A} + \mathbf{e}^\intercal + c \cdot \hat{\mathbf{i}}, \mathbf{s}^\intercal\mathbf{y} + c \cdot g_i + e'\right)$ is computationally indistinguishable from the distribution $(\mathbf{u}, \mathbf{u}^\intercal\mathbf{x}_1 + e')$, and thus also from $(\mathbf{s}^\intercal\mathbf{A} + \mathbf{e}^\intercal, \mathbf{s}^\intercal\mathbf{y} + e')$. Given $\rho'_{\textsc{Aux}}$ is a $\gamma$-good state, $\mathsf{TI}_{\frac{1}{2}+\gamma'}(i, g_i)$ outputs 1 with $1 - \mathsf{negl}$ probability if all other threshold implementations are ignored (i.e., applied $\mathsf{TI}_{\frac{1}{2}+\gamma'}(i, g_i)$ just after $\mathsf{TI}_{\frac{1}{2}+\gamma}$).
– If the guess is incorrect, the distribution $\left(\mathbf{s}^\intercal\mathbf{A} + \mathbf{e}^\intercal + c \cdot \hat{\mathbf{i}}, \mathbf{s}^\intercal\mathbf{y} + c \cdot g_i + e'\right)$ is computationally indistinguishable from $(\mathbf{u}, r)$. Consequently, any state provides no advantage as advice, and $\mathsf{TI}_{\frac{1}{2}+\gamma'}$ outputs 1 with $\mathsf{negl}$ probability.

Finally, we apply the quantum union bound to all measurements to demonstrate that the probability of no error occurring during our testing procedure is $1 - \mathsf{negl}$.

In the above proof, we omitted a major issue. Recall that in Step 1, we implement threshold implementation to project the state $\rho$ onto a good state $\rho_{\mathsf{good}}$. Moreover, this threshold implementation is designed to check if the adversary can distinguish between the distributions $(\mathbf{s}^\intercal \mathbf{A} + \mathbf{e}^\intercal, \mathbf{s}^\intercal \mathbf{y} + e')$ and $(\mathbf{u}, r)$. As discussed above, at some point, in the intermediate hybrids we need to change these distributions. Once we switch the distributions, the threshold implementation might only work with negligible probability. Our hope, in some cases invoking learning with errors, is to argue that this does not happen. However, it is not clear how to carry out this reduction. After all, the threshold implementation as defined by [Zha20] operates on a superposition of exponentially many samples from a distribution and so, given just one sample from a distribution, it is not possible to perform threshold implementation. We present a useful lemma (in Sect. 5) where we argue that operationally, the guarantees of threshold implementation (including the output and the residual state) are not affected when one distribution is replaced with another computationally indistinguishable distribution.

## 2 Preliminaries

We use standard notations throughout this work. We assume that the reader is familiar with quantum computing and lattices. We refer the reader to the full version [AHH24] for a complete presentation of this section.

### 2.1 Quantum Computing

We will use the following lemma.

**Lemma 1 (Quantum Union Bound, [Gao15]).** *Let $\mathcal{H}$ be a Hilbert space. Let $\rho \in \mathcal{D}(\mathcal{H})$ be a state and let $\boldsymbol{\Pi}_1, \ldots, \boldsymbol{\Pi}_n \geq 0$ be sequence of (orthogonal) projections acting on $\mathcal{H}$. Suppose that, for every $i \in [n]$, it holds that $\mathrm{Tr}[\boldsymbol{\Pi}_i \rho] = 1 - \varepsilon_i$, for $\varepsilon_i \in [0,1]$. Then, if we sequentially measure $\rho$ with projective measurements $\{\boldsymbol{\Pi}_1, \mathbf{I} - \boldsymbol{\Pi}_1\}, \ldots, \{\boldsymbol{\Pi}_n, \mathbf{I} - \boldsymbol{\Pi}_n\}$, the probability that all measurements succeed is at least*

$$\mathrm{Tr}[\boldsymbol{\Pi}_n \cdots \boldsymbol{\Pi}_1 \rho \boldsymbol{\Pi}_1 \cdots \boldsymbol{\Pi}_n] \geq 1 - 4 \sum_{i=1}^{n} \varepsilon_i.$$

### 2.2 Lattices and Cryptography

We adapt notations from [APV23] and keep it same as much as we can. The following subsection is copied verbatim from [APV23].

In this work, we mainly consider *q-ary lattices* $\Lambda$ that that satisfy $q\mathbb{Z}^m \subseteq \Lambda \subseteq \mathbb{Z}^m$, for some integer modulus $q \geq 2$. Specifically, we consider the lattice generated by a matrix $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ for some $n, m \in \mathbb{N}$ that consists of all vectors which are perpendicular to the rows of $\mathbf{A}$, namely

$$\Lambda_q^{\perp}(\mathbf{A}) = \{\mathbf{x} \in \mathbb{Z}^m : \mathbf{A} \cdot \mathbf{x} = \mathbf{0} \pmod{q}\}.$$

For any *syndrome* $\mathbf{y} \in \mathbb{Z}_q^n$ in the column span of $\mathbf{A}$, we also consider the coset $\Lambda_q^{\mathbf{y}}(\mathbf{A})$ given by

$$\Lambda_q^{\mathbf{y}}(\mathbf{A}) = \{\mathbf{x} \in \mathbb{Z}^m : \mathbf{A} \cdot \mathbf{x} = \mathbf{y} \pmod{q}\} = \Lambda_q^{\perp}(\mathbf{A}) + \mathbf{c},$$

where $\mathbf{c} \in \mathbb{Z}^m$ is an arbitrary integer solution to the equation $\mathbf{A}\mathbf{c} = \mathbf{y} \pmod{q}$.

**Definition 1 (Truncated discrete Gaussian distribution).** *Let $m \in \mathbb{N}$, $q \geq 2$ be an integer modulus and let $\sigma > 0$ be a parameter. Then, the* truncated *discrete Gaussian distribution $D_{\mathbb{Z}_q^m,\sigma}$ with finite support $\{\mathbf{x} \in \mathbb{Z}^m \cap (-\frac{q}{2}, \frac{q}{2}]^m : \|\mathbf{x}\| \leq \sigma\sqrt{m}\}$ is defined as the density*

$$D_{\mathbb{Z}_q^m,\sigma}(\mathbf{x}) = \frac{\rho_\sigma(\mathbf{x})}{\sum\limits_{\mathbf{y} \in \mathbb{Z}_q^m, \|\mathbf{y}\| \leq \sigma\sqrt{m}} \rho_\sigma(\mathbf{y})}.$$

*where $\rho_\sigma(\mathbf{x}) := \exp(-\pi\|\mathbf{x}\|^2/\sigma^2)$ is the Gaussian distribution.*

We will use the following two results.

**Lemma 2 (Noise smudging, [DGT+10]).** *Let $y, \sigma > 0$. Then, the statistical distance between the distribution $D_{\mathbb{Z},\sigma}$ and $D_{Z,\sigma} + y$ is at most $y/\sigma$.*

We use the following technical lemma on the min-entropy of the truncated discrete Gaussian distribution, which we prove below.

**Lemma 3 (min-entropy of the truncated discrete Gaussian, [APV23], Lemma 2.10).** *Let $n \in \mathbb{N}$ and let $q$ be a prime with $m \geq 2n \log q$. Let $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ be a matrix whose columns generate $\mathbb{Z}_q^n$. Then, for any $\sigma \geq \omega(\sqrt{\log m})$, there exists a negligible $\varepsilon(m)$ such that*

$$\max_{\mathbf{y} \in \mathbb{Z}_q^n} \max_{\substack{\mathbf{x} \in \mathbb{Z}_q^m, \|\mathbf{x}\| \leq \sigma\sqrt{m} \\ \mathbf{A}\mathbf{x} = \mathbf{y} \pmod{q}}} \left\{ \frac{\rho_\sigma(\mathbf{x})}{\sum\limits_{\substack{\mathbf{z} \in \mathbb{Z}_q^m, \|\mathbf{z}\| \leq \sigma\sqrt{m} \\ \mathbf{A}\mathbf{z} = \mathbf{y} \pmod{q}}} \rho_\sigma(\mathbf{z})} \right\} \leq 2^{-m+1} \cdot \frac{1+\varepsilon}{1-\varepsilon}.$$

**Theorem 2 (Gaussian-collapsing property, [Por22], Theorem 4).** *Let $n \in \mathbb{N}$ and $q$ be a prime with $m \geq 2n \log q$, each parameterized by $\lambda \in \mathbb{N}$. Let $\sqrt{8m} < \sigma < q/\sqrt{8m}$. Then, the following samples are computationally indistinguishable* *assuming the quantum hardness of decisional* $\mathsf{LWE}^m_{n,q,\alpha q}$*, for any noise ratio* $\alpha \in (0,1)$ *with relative noise magnitude* $1/\alpha = \sigma \cdot 2^{o(n)}$ :

$$\left(\mathbf{A} \xleftarrow{\$} \mathbb{Z}_q^{n\times m},\ |\psi_{\mathbf{y}}\rangle = \sum_{\substack{\mathbf{x}\in\mathbb{Z}_q^m \\ \mathbf{A}\mathbf{x}=\mathbf{y}}} \rho_\sigma(\mathbf{x})\,|\mathbf{x}\rangle,\ \mathbf{y}\in\mathbb{Z}_q^n\right) \approx_c \left(\mathbf{A} \xleftarrow{\$} \mathbb{Z}_q^{n\times m},\ |\mathbf{x}_0\rangle,\ \mathbf{A}\cdot\mathbf{x}_0 \in \mathbb{Z}_q^n\right)$$

*where* $(|\psi_{\mathbf{y}}\rangle, \mathbf{y}) \leftarrow \mathsf{GenGauss}(\mathbf{A},\sigma)$ *and where* $\mathbf{x}_0 \sim D_{\mathbb{Z}_q^m,\frac{\sigma}{\sqrt{2}}}$ *is a (truncated) discrete Gaussian distribution.*

## 2.3 Threshold Implementation and Its Approximate Version

In the subsection, we review some techniques called *Threshold Implementation* [ALL+21], which is a simple extension of Projective Implementation [Zha20].

**Theorem 3 (Threshold implementation, [ALL+21]).** *Let* $\gamma \in (0,1)$ *be a parameter and let* $\mathcal{P} = (P,Q)$ *be a two-outcome POVM, where* $P$ *has an eigenbasis* $\{|\psi_i\rangle\}$ *with associated eigenvalues* $\{\lambda_i\}$*. Then, there exists a projective threshold implementation* $(\mathsf{TI}_\gamma(\mathcal{P}), \mathsf{I} - \mathsf{TI}_\gamma(\mathcal{P}))$ *such that*

– $\mathsf{TI}_\gamma(\mathcal{P})$ *projects a quantum state into the subspace spanned by* $\{|\psi_i\rangle\}$ *whose eigenvalues* $\lambda_i$ *satisfy the property* $\lambda_i \leq \gamma$*.*
– $\mathsf{I}-\mathsf{TI}_\gamma(\mathcal{P})$ *projects a quantum state into the subspace spanned by* $\{|\psi_i\rangle\}$ *whose eigenvalues* $\lambda_i$ *satisfy the property* $\lambda_i > \gamma$*.*

Unfortunately, the threshold implementation can, in general, not be efficiently computable. However, inspired by the work of Marriott and Watrous [MW05], Zhandry [Zha20] showed that the approximate version of the threshold implementation can be implemented efficiently as long as the POVM is a mixture of projective measurements. We first review the definition of mixture of projective measurements.

**Definition 2 (Mixture of projective measurements).** *Let* $\mathcal{P} = \{\mathcal{P}_i\}_{i\in\mathcal{I}}$ *be a collection of binary outcome projective measurements* $\mathcal{P}_i = (P_i, Q_i)$ *over the same Hilbert space* $\mathcal{H}$*, and suppose that* $P_i$ *corresponds to outcome* 1 *and* $Q_i$ *corresponds to outcome* 0*. Let* $D$ *be a distribution over the index set* $\mathcal{I}$*. Then,* $\mathcal{P}_D = (P_D, Q_D)$ *is the following mixture of projective measurements:*

$$P_D = \sum_{i\in\mathcal{I}} \Pr[i \leftarrow D]\, P_i \qquad \textit{and} \qquad Q_D = \sum_{i\in\mathcal{I}} \Pr[i \leftarrow D]\, Q_i.$$

In other words, $\mathcal{P}_D$ is the same as first sampling $i$ according to the distribution $D$, and then applying the projective measurements $\mathcal{P}_i$.

For any mixture of projective measurements $\mathcal{P}_D$, the approximate threshold implementation satisfies the following properties.

**Lemma 4 (Approximate threshold implementation, Theorem 6.2 in [Zha20] and Corollary 1 in [ALL+21]).** *Let $\mathcal{P}_D = (P_D, Q_D)$ be a binary outcome POVM over Hilbert space $\mathcal{H}$ that is a mixture of projective measurements over some distribution $D$. Let $\varepsilon, \delta, \gamma \in (0, 1)$. Then, there exists an efficient binary-outcome quantum algorithm $\mathsf{ATI}^{\varepsilon,\delta}_{\mathcal{P},D,\gamma}$, interpreted as the POVM element corresponding to outcome 1, such that the following holds:*

- *For all quantum states $\rho$, $\mathrm{Tr}[\mathsf{ATI}^{\varepsilon,\delta}_{\mathcal{P},D,\gamma-\varepsilon}\, \rho] \geq \mathrm{Tr}[\mathsf{TI}_\gamma(\mathcal{P}_D)\, \rho] - \delta$.*
- *For all quantum states $\rho$, it holds that $\mathrm{Tr}[\mathsf{TI}_{\gamma-2\varepsilon}(\mathcal{P}_D)\, \rho'] \geq 1 - 2\delta$, where $\rho'$ is the post-measurement state which results from applying the measurement $\mathsf{ATI}^{\varepsilon,\delta}_{\mathcal{P},D,\gamma}$ to $\rho$ and obtaining outcome 1.*
- *The expected running time to implement $\mathsf{ATI}^{\varepsilon,\delta}_{\mathcal{P},D,\gamma}$ is proportional to $\mathrm{poly}(1/\varepsilon, \log(1/\delta))$, the time it takes to implement $P_D$, and the time it takes to sample from $D$.*

## 3 Definition: Key-Revocable Public-Key Encryption

A key-revocable public-key encryption is a type of public-key encryption. Consider the case where the secret key holder wishes to temporarily give the secret key to a third party and later wants to take it back while maintaining the security i.e. the third party upon taken its key away, can't decrypt any message later. This is impossible in the classical case since the third party can always copy the secret key locally. But we may achieve this functionality by representing the secret key as a quantum state.

**Definition 3 (Key-Revocable Public-Key Encryption [APV23]).** *A key-revocable public-key encryption scheme consists of efficient algorithms* $(\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Revoke})$*, where* $\mathsf{Enc}$ *is a* $\mathsf{PPT}$ *algorithm and* $\mathsf{KeyGen}, \mathsf{Dec}, \mathsf{Revoke}$ *are* $\mathsf{QPT}$ *algorithms defined as follows:*

- $\mathsf{KeyGen}(1^\lambda)$*: given as input a security parameter $\lambda$, output a public key* $\mathsf{PK}$*, a master secret key* $\mathsf{MSK}$ *and a quantum decryption key* $\rho_{\mathsf{SK}}$*.*
- $\mathsf{Enc}(\mathsf{PK}, \mu)$*: given a public key* $\mathsf{PK}$ *and plaintext $\mu \in \{0, 1\}$, output a ciphertext* $\mathsf{CT}$*.*
- $\mathsf{Dec}(\rho_{\mathsf{SK}}, \mathsf{CT})$*: given a decryption key* $\rho_{\mathsf{SK}}$ *and ciphertext* $\mathsf{CT}$*, output a message $y$.*
- $\mathsf{Revoke}(\mathsf{PK}, \mathsf{MSK}, \rho_{\mathrm{R}})$*: given as input a master secret key* $\mathsf{MSK}$*, a public key* $\mathsf{PK}$ *and quantum state* $\rho_{\mathrm{R}}$*, output* $\mathsf{Valid}$ *or* $\mathsf{Invalid}$*.*

*Correctness of Decryption.* For $\mu \in \{0, 1\}$, the following holds:

$$\mathsf{Pr}\left[\mu \leftarrow \mathsf{Dec}(\rho_{\mathsf{SK}}, \mathsf{CT}) : \begin{smallmatrix} (\mathsf{PK}, \mathsf{MSK}, \rho_{\mathsf{SK}}) \leftarrow \mathsf{KeyGen}(1^\lambda) \\ \mathsf{CT} \leftarrow \mathsf{Enc}(\mathsf{PK}, \mu) \end{smallmatrix}\right] \geq 1 - \mathsf{negl}.$$

*Correctness of Revocation.* The following holds:

$$\mathsf{Pr}\left[\mathsf{Valid} \leftarrow \mathsf{Revoke}(\mathsf{PK}, \mathsf{MSK}, \rho_{\mathsf{SK}}) : (\mathsf{PK}, \mathsf{MSK}, \rho_{\mathsf{SK}}) \leftarrow \mathsf{KeyGen}(1^\lambda)\right] \geq 1 - \mathsf{negl}.$$

### 3.1 Security Definition

The security captures the case where the adversary is given the key and later taken back. After that, if the key passes the revocation check the adversary is asked to play a CPA like game that it is given either the ciphertext of a chosen message or a random message. The adversary wins if it can distinguish between these two cases.

**Definition 4.** *A key-revocable public-key encryption scheme $\Sigma = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Revoke})$ is $(\epsilon, \delta)$-secure if, for every* QPT *adversary $\mathcal{A}$ with*

$$\mathsf{Pr}\left[\mathsf{Invalid} \leftarrow \mathsf{Expt}^{\Sigma,\mathcal{A}}(1^\lambda, b)\right] \leq \delta(\lambda)$$

*for $b \in \{0,1\}$, it holds that*

$$\left|\mathsf{Pr}\left[1 \leftarrow \mathsf{Expt}^{\Sigma,\mathcal{A}}(1^\lambda, 0)\right] - \mathsf{Pr}\left[1 \leftarrow \mathsf{Expt}^{\Sigma,\mathcal{A}}(1^\lambda, 1)\right]\right| \leq \epsilon(\lambda),$$

*where $\mathsf{Expt}^{\Sigma,\mathcal{A}}(1^\lambda, b)$ is defined as Fig. 1.*

*If $\delta(\lambda) = 1 - \frac{1}{\mathrm{poly}(\lambda)}$ and $\epsilon(\lambda) = \mathsf{negl}(\lambda)$, we simply say the key-revocable encryption scheme is secure.*

$\underline{\mathsf{Expt}^{\Sigma,\mathcal{A}}(1^\lambda, b)}$ :

*Initialization Phase:*

- The challenger runs $(\mathsf{PK}, \mathsf{MSK}, \rho_{\mathsf{SK}}) \leftarrow \mathsf{KeyGen}(1^\lambda)$ and sends $(\mathsf{PK}, \rho_{\mathsf{SK}})$ to $\mathcal{A}$.

*Revocation Phase:*

- The challenger sends the message `REVOKE` to $\mathcal{A}$.
- The adversary $\mathcal{A}$ returns a state $\rho_{\mathrm{R}}$.
- The challenger aborts if $\mathsf{Revoke}(\mathsf{MSK}, \mathsf{PK}, \rho_{\mathrm{R}})$ outputs $\mathsf{Invalid}$.

*Guessing Phase:*

- $\mathcal{A}$ submits a plaintext $\mu \in \{0,1\}$ to the challenger.
- If $b = 0$: The challenger sends $\mathsf{CT} \leftarrow \mathsf{Enc}(\mathsf{PK}, \mu)$ to $\mathcal{A}$. Else, if $b = 1$, the challenger sends $\mathsf{CT} \leftarrow \mathcal{C}$, where $\mathcal{C}$ is the ciphertext space of $\ell$ bit messages.
- Output $b_{\mathcal{A}}$ if the output of $\mathcal{A}$ is $b_{\mathcal{A}}$.

**Fig. 1.** Security Experiment

## 4 Construction: Key Revocable Dual-Regev Encryption

The construction is exactly the same as the construction in [APV23]. We include the construction here for completeness.

**Construction 4 (Key Revocable Dual-Regev Encryption [APV23]).** *Let $n, m \in \mathbb{N}$ and $q \geq 2$ be a prime, each parameterized by $\lambda \in \mathbb{N}$. Let $\alpha, \beta, \sigma > 0$ be parameters. The key-revocable public key scheme* $\mathsf{RevDual} = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Revoke})$ *consists of the following* QPT *algorithms:*

- $\mathsf{KeyGen}(1^\lambda) \rightarrow (\mathsf{PK}, \rho_{\mathsf{SK}}, \mathsf{MSK})$*: Sample* $(\mathbf{A} \in \mathbb{Z}_q^{n\times m}, \mathsf{td}_{\mathbf{A}}) \leftarrow \mathsf{GenTrap}(1^n, 1^m, q)$ *where* $\mathsf{GenTrap}$ *is the algorithm that generates the LWE matrix with its trapdoor. Then generate a Gaussian superposition* $(|\psi_{\mathbf{y}}\rangle, \mathbf{y}) \leftarrow \mathsf{GenGauss}(\mathbf{A}, \sigma)$[5] *for some* $\mathbf{y} \in \mathbb{Z}_q^n$*. Output* $\mathsf{PK} = (\mathbf{A}, \mathbf{y}), \rho_{\mathsf{SK}} = |\psi_{\mathbf{y}}\rangle$ *and* $\mathsf{MSK} = \mathsf{td}_{\mathbf{A}}$.
- $\mathsf{Enc}(\mathsf{PK}, \mu) \rightarrow \mathsf{CT}$*: to encrypt a bit* $\mu \in \{0,1\}$*, sample a random vector* $\mathbf{s} \leftarrow \mathbb{Z}_q^n$ *and errors* $\mathbf{e} \sim D_{\mathbb{Z}^m, \alpha q}$ *and* $e' \sim D_{\mathbb{Z}, \beta q}$ *and output the ciphertext pair* $\mathsf{CT} = \left(\mathbf{s}^\intercal \mathbf{A} + \mathbf{e}^\intercal \pmod q, \mathbf{s}^\intercal \mathbf{y} + e' + \mu \cdot \lfloor \frac{q}{2} \rfloor \pmod q\right) \in \mathbb{Z}_q^m \times \mathbb{Z}_q$.
- $\mathsf{Dec}(\rho_{\mathsf{SK}}, \mathsf{CT}) \rightarrow \{0,1\}$ *: to decrypt* $\mathsf{CT}$*, apply the unitary* $U : |\mathbf{x}\rangle|0\rangle \rightarrow |\mathbf{x}\rangle|\mathsf{CT} \cdot (-\mathbf{x}, 1)^\intercal\rangle$ *on input* $|\psi_{\mathbf{y}}\rangle|0\rangle$*, where* $\rho_{\mathsf{SK}} = |\psi_{\mathbf{y}}\rangle$*, and measure the second register in the computational basis. Output* $0$*, if the measurement outcome is closer to* $0$ *than to* $\lfloor \frac{q}{2} \rfloor$*, and output* $1$*, otherwise.*
- $\mathsf{Revoke}(\mathsf{MSK}, \mathsf{PK}, \rho) \rightarrow \{\top, \bot\}$ *: on input* $\mathsf{td}_{\mathbf{A}} \leftarrow \mathsf{MSK}$ *and* $(\mathbf{A}, \mathbf{y}) \leftarrow \mathsf{PK}$*, apply the measurement* $\{|\psi_{\mathbf{y}}\rangle\langle\psi_{\mathbf{y}}|, I - |\psi_{\mathbf{y}}\rangle\langle\psi_{\mathbf{y}}|\}$ *onto the state* $\rho$ *using the procedure* $\mathsf{QSampGauss}(\mathbf{A}, \mathsf{td}_{\mathbf{A}}, \mathbf{y}, \sigma)$[6]*. Output* $\top$ *if the measurement is successful, and* $\bot$ *otherwise.*

From [APV23], this construction satisfies the correctness of decryption and the correctness of revocation. In this work, we will focus on showing the construction is in fact secure.

**Theorem 5.** *Let* $n \in \mathbb{N}$ *and* $q$ *be a prime modulus with* $q = 2^{o(n)}$ *and* $m \geq 2n \log q$*, each parameterized by security parameter* $\lambda \in \mathbb{N}$*. Let* $\sqrt{8m} < \sigma < q/\sqrt{8m}$ *and let* $\alpha, \beta \in (0,1)$ *be noise ratios chosen such that* $\beta/\alpha = 2^{o(n)}$ *and* $1/\alpha = 2^{o(n)} \cdot \sigma$*. Then, assuming the polynomial hardness of* $\mathsf{LWE}^m_{n,q,\alpha q}$ *with sub-exponential modulus, the scheme* $\mathsf{RevDual} = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Revoke})$ *in Construction 4 is a secure key-revocable public-key encryption scheme according to Definition 4.*

We organize the proof of Theorem 5 in the following way:

- In Sect. 5, we prove an important property for approximate threshold implementation, which allows us to do hybrid arguments between approximate threshold implementation on computationally indistinguishable distributions.
- In Sect. 6, we present our construction for the almost perfect preimage extractor that lies in the heart of our result.
- In Sect. 7, we complete our proof of the above theorem.

[5] The detailed description of $\mathsf{GenGauss}$ can be found in [APV23].
[6] The detailed description of $\mathsf{QSampGauss}$ can be found in [APV23].

## 5 Indistinguishability on Approximate Threshold Implementation

Zhandry [Zha20] analyzed the relationship between the output distribution of $\mathsf{TI}_{\gamma_0}(\mathcal{P}_{D_0})$ and $\mathsf{TI}_{\gamma_1}(\mathcal{P}_{D_1})$ (and $\mathsf{ATI}_{\mathcal{P},D_0,\gamma_0}$ and $\mathsf{ATI}_{\mathcal{P},D_1,\gamma_1}$) for some thresholds $\gamma_0$ and $\gamma_1$ on the same state for two computationally indistinguishable distributions $D_0$ and $D_1$. However, in our work, we also care about the residual state after applying the procedures. So we give a more precise analysis below.

In this section, we show how to leverage a (possibly not efficiently constructible) quantum state $\rho$ on which $\mathsf{ATI}_{\mathcal{P},D_0,\gamma}$ and $\mathsf{ATI}_{\mathcal{P},D_1,\gamma}$ behave differently to construct a $\mathsf{QPT}$ distinguisher (with auxiliary state $\rho$) for $D_0$ and $D_1$. This can be viewed as an extension of Theorem 6.5 and Corollary 6.9 in [Zha20].

This result allows us to do hybrid arguments between $\mathsf{ATI}_{\mathcal{P},D_0,\gamma}$ and $\mathsf{ATI}_{\mathcal{P},D_1,\gamma}$ with exactly the same threshold parameter $\gamma$ for computationally indistinguishable distributions $D_0$ and $D_1$ even when an efficient quantum procedure is applied on the residual state after $\mathsf{ATI}$. Notably, it applies even when we need some classical advice to sample from $D_0$ and $D_1$, in which case, our $\mathsf{QPT}$ distinguisher additionally takes the same classical advice and distinguishes $D_0$ and $D_1$.

**Lemma 5.** *Let $\mathcal{P}$ be a collection of projective measurements indexed by some set $\mathcal{I}$. Suppose $\mathcal{P}$ can be implemented by a quantum circuit of size $|\mathcal{P}|$. Let $D_0, D_1$ be two efficiently sampleable distributions over $\mathcal{I}$. For any state $\rho \in \mathcal{D}(\mathcal{H})$, denote $(b, \rho') \leftarrow \mathsf{ATI}^{\varepsilon,\delta}_{\mathcal{P},D,\gamma}(\rho)$ be the procedure that runs $\mathsf{ATI}^{\varepsilon,\delta}_{\mathcal{P},D,\gamma}$ on state $\rho$, and gets an output $b$ and the post-measurement state $\rho'$. For any polynomial $\mu$, any quantum state $\rho$ and any (possibly quantum) predicate $h : \{0,1\} \times \mathcal{D}(\mathcal{H}) \to \{0,1\}$ with circuit size $|h|$, if*

$$\left|\Pr\left[h(b,\rho')=1|(b,\rho')\leftarrow \mathsf{ATI}^{\epsilon,\delta}_{\mathcal{P},D_0,\gamma}(\rho)\right]-\Pr\left[h(b,\rho')=1|(b,\rho')\leftarrow \mathsf{ATI}^{\epsilon,\delta}_{\mathcal{P},D_1,\gamma}(\rho)\right]\right|\geq \frac{1}{\mu(\lambda)}.$$

*Then there exists a quantum circuit $\mathcal{C}$ of size* $\mathrm{poly}(\lambda, 1/\epsilon, \log(1/\delta), \mu, |\mathcal{P}|, |h|)$ *(which only uses the quantum circuits to implement $\mathcal{P}$, $h$ and to sample $D_0, D_1$ as a black box) such that*

$$\left|\Pr\left[\mathcal{C}(\rho,x)=b : {\scriptstyle b \xleftarrow{\$} \{0,1\} \atop x\sim D_b}\right]-\frac{1}{2}\right|\geq \frac{1}{(\mu(\lambda))^3\cdot \mathrm{poly}(\lambda,1/\epsilon,\log(1/\delta))}$$

*which is an inverse polynomial if $\mu$ is a polynomial.*

*Proof.* The proof follows the same idea as the proof for Theorem 6.5 in [Zha20]. Roughly speaking, the output of $\mathsf{ATI}^{\epsilon,\delta}_{\mathcal{P},D,\gamma}$ can be approximated up to inverse polynomial additive error given only polynomial samples from $D$. We refer the reader to the full version [AHH24] for the full proof.

$\mathsf{ATI}$ may change the input state in an essential way even when it outputs 1 with overwhelming probability because $\mathsf{ATI}$ is not a projector. For example, let a pure quantum state $\rho$ be a superposition of eigenvectors (of $\mathcal{P}_D$) $|\psi_i\rangle$ whose eigenvalues $\lambda_i$ satisfy the property $\lambda_i \geq \gamma + 10\epsilon$. If we apply $\mathsf{ATI}^{\epsilon,\delta}_{\mathcal{P},D,\gamma}$ on $\rho$, we will get outcome 1 with almost certainty, but the residual state $\rho'$ may lose coherence and become closer to a mixture of $|\psi_i\rangle$.

When we know the $\mathsf{ATI}$ outputs 0 or 1 with overwhelming probability, it is a good idea to *minimize the disturbance* by purifying $\mathsf{ATI}$ and performing uncomputation, just like the famous gentle measurements. To be more precise, we consider the projective version of $\mathsf{ATI}^{\epsilon,\delta}_{\mathcal{P},D,\gamma}$. Formally, $\mathsf{ATI}^{\epsilon,\delta}_{\mathcal{P},D,\gamma}$ can be written as introducing $\text{poly}(1/\epsilon, \log(1/\delta))$ ancillas initialized as $|0\rangle$, applying a unitary $U$ on the state, and then applying a projective measurement $(|0\rangle\langle 0|, |1\rangle\langle 1|)$ on the output register of state to get the output. We will denote the binary-outcome projective measurement $(U^\dagger|0\rangle\langle 0|U, U^\dagger|1\rangle\langle 1|U)$ as $\overline{\mathsf{ATI}}^{\epsilon,\delta}_{\mathcal{P},D,\gamma}$, the projective version of $\mathsf{ATI}^{\epsilon,\delta}_{\mathcal{P},D,\gamma}$, which also has size $\text{poly}(1/\epsilon, \log(1/\delta))$. By definition, for any quantum state $\rho$, the output distribution of running $\overline{\mathsf{ATI}}$ on $\rho$ along with enough fresh ancillas is the same as the output distribution of running $\mathsf{ATI}$ on $\rho$ (but the residual states are different).

Roughly speaking, $\overline{\mathsf{ATI}}^{\epsilon,\delta}_{\mathcal{P},D,\gamma}$ does the same thing as $\mathsf{ATI}^{\epsilon,\delta}_{\mathcal{P},D,\gamma}$ except that it uncomputes intermediate results. Notice that a quantum query to function $f$ is implemented as $U_f : |x\rangle|y\rangle \to |x\rangle|y \oplus f(x)\rangle$, whose inverse is exactly $U_f$. We can use the same proof technique in Lemma 5 to show that $\overline{\mathsf{ATI}}^{\epsilon,\delta}_{\mathcal{P},D,\gamma}$ can also be approximated by polynomial classical samples from $D$ up to inverse polynomial precision and thus we can also apply hybrid arguments between $\overline{\mathsf{ATI}}_{\mathcal{P},D_0,\gamma}$ and $\overline{\mathsf{ATI}}_{\mathcal{P},D_1,\gamma}$ for computationally indistinguishable distributions $D_0$ and $D_1$. Formally,

**Lemma 6.** *Let $\mathcal{H}_{\mathrm{R}}, \mathcal{H}_{\mathrm{Aux}}$ be Hilbert spaces. Let $\mathcal{P}$ be a collection of projective measurements indexed by some set $\mathcal{I}$. Suppose $\mathcal{P}$ can be implemented by a quantum circuit of size $|\mathcal{P}|$. Let $D_0, D_1$ be two efficiently sampleable distributions over $\mathcal{I}$. For any state $\rho \in \mathcal{D}(\mathcal{H}_{\mathrm{R}})$, denote $(b, \rho') \leftarrow \overline{\mathsf{ATI}}^{\varepsilon,\delta}_{\mathcal{P},D,\gamma}(\rho)$ be the procedure that runs $\overline{\mathsf{ATI}}^{\varepsilon,\delta}_{\mathcal{P},D,\gamma}$ on state $\rho$ along with enough fresh ancillas initialized to $|0\rangle$, and gets an output $b$ and the post-measurement state $\rho'$. For any polynomial $\mu$, any quantum state $\rho$ and any (possibly quantum) predicate $h : \{0,1\} \times \mathcal{D}(\mathcal{H}_{\mathrm{Aux}}) \to \{0,1\}$ with circuit size $|h|$, if*

$$\left|\Pr\left[h(b,\rho') = 1 | (b,\rho') \leftarrow \overline{\mathsf{ATI}}^{\epsilon,\delta}_{\mathcal{P},D_0,\gamma}(\rho)\right] - \Pr\left[h(b,\rho') = 1 | (b,\rho') \leftarrow \overline{\mathsf{ATI}}^{\epsilon,\delta}_{\mathcal{P},D_1,\gamma}(\rho)\right]\right| \geq \frac{1}{\mu(\lambda)}.$$

*Then there exists a quantum circuit $\mathcal{C}$ of size $\text{poly}(\lambda, 1/\epsilon, \log(1/\delta), \mu, |\mathcal{P}|, |h|)$ (which only uses the quantum circuits to implement $\mathcal{P}$, $h$ and to sample $D_0, D_1$ as a black box) such that*

$$\left|\Pr\left[\mathcal{C}(\rho, x) = b : \underset{x \sim D_b}{b \xleftarrow{\$} \{0,1\}}\right] - \frac{1}{2}\right| \geq \frac{1}{(\mu(\lambda))^3 \cdot \text{poly}(\lambda, 1/\epsilon, \log(1/\delta))}$$

*which is an inverse polynomial if $\mu$ is a polynomial.*

*Proof.* We omit the proof as it's almost the same as the proof of Lemma 5.

## 6 Almost Perfect Extraction of Preimages

In this section, we show how to extract a short preimage of $\mathbf{y}$ with overwhelming probability, given a good (quantum) distinguisher between the distribution of a ciphertext of message $\mu$ and a uniform distribution. Our main contribution is an extraction algorithm that is guaranteed to work with *overwhelming probability*, in contrast to the extraction algorithm in [APV23] that only works with probability inversely proportional to the field size.

Since a general quantum distinguisher can be a superposition of a good distinguisher and a useless distinguisher, we use (Approximate) Threshold Implementation to (approximately) test whether a given quantum distinguisher is good before we apply the extraction algorithm. We need the following notations before we formally define what is a good quantum distinguisher.

*Threshold Implementation on a Quantum Distinguisher.* For a quantum algorithm $\mathcal{A}$ with auxiliary quantum state $\rho$, let projective measurements $\{\mathcal{P}_x^{\mathcal{A}} = (P_x^{\mathcal{A}}, Q_x^{\mathcal{A}})\}$ correspond to running $\mathcal{A}$ on $x$ and the auxiliary state $\rho$. Suppose that $P_x^{\mathcal{A}}$ corresponds to outcome 1 and $Q_x^{\mathcal{A}}$ corresponds to outcome 0.

For two distributions $D_0$ and $D_1$, denote $(D_0, D_1)$ to be the distribution of $(b, x)$ where $b \overset{\$}{\leftarrow} \{0,1\}$ and $x \sim D_b$. We say that $(\mathcal{A}, \rho)$ is a *$\gamma$-good quantum distinguisher* for distributions $D_0$ and $D_1$ with support $\mathcal{X}$ if and only if $\rho$ passes the projector $\mathsf{TI}_{1/2+\gamma}(\mathcal{P}^{\mathcal{A}}_{(D_0,D_1)})$. Here, we abuse the notation to define the POVM $\mathcal{P}^{\mathcal{A}}_{(D_0,D_1)} = (P^{\mathcal{A}}_{(D_0,D_1)}, Q^{\mathcal{A}}_{(D_0,D_1)})$[7] such that

$$P^{\mathcal{A}}_{(D_0,D_1)} = \frac{P^{\mathcal{A}}_{D_1} + Q^{\mathcal{A}}_{D_0}}{2} = \frac{\sum_{x\in\mathcal{X}} \mathsf{Pr}\left[x \leftarrow D_1\right] P_x^{\mathcal{A}} + \sum_{x\in\mathcal{X}} \mathsf{Pr}\left[x \leftarrow D_0\right] Q_x^{\mathcal{A}}}{2},$$

$$Q^{\mathcal{A}}_{(D_0,D_1)} = I - P^{\mathcal{A}}_{(D_0,D_1)}.$$

In other words, $\mathcal{P}^{\mathcal{A}}_{(D_0,D_1)} = (P^{\mathcal{A}}_{(D_0,D_1)}, Q^{\mathcal{A}}_{(D_0,D_1)})$ is the POVM measurement (where $P^{\mathcal{A}}_{(D_0,D_1)}$ corresponds to output 1 and $Q^{\mathcal{A}}_{(D_0,D_1)}$ corresponds to output 0) that on any input quantum state $\rho$,

- Sample $(b, x) \sim (D_0, D_1)$.
- Feed $x$ and the input quantum state $\rho$ into $\mathcal{A}$, which outputs a guess $b'$.
- Output 1 if $b' = b$; 0 otherwise.

We denote the approximate version of $\mathsf{TI}_{1/2+\gamma}(\mathcal{P}^{\mathcal{A}}_{(D_0,D_1)})$ as $\mathsf{ATI}^{\epsilon,\delta}_{\mathcal{P}^{\mathcal{A}},(D_0,D_1),1/2+\gamma}$. Roughly speaking, $\mathsf{ATI}^{\epsilon,\delta}_{\mathcal{P}^{\mathcal{A}},(D_0,D_1),1/2+\gamma}$ can efficiently estimate whether the algorithm $\mathcal{A}$, along with the input quantum state as auxiliary, can distinguish $D_0$ and $D_1$ with advantage at least $\gamma$. We denote the projective version of $\mathsf{ATI}^{\epsilon,\delta}_{\mathcal{P}^{\mathcal{A}},(D_0,D_1),1/2+\gamma}$ as $\overline{\mathsf{ATI}}^{\epsilon,\delta}_{\mathcal{P}^{\mathcal{A}},(D_0,D_1),1/2+\gamma}$.

[7] $\mathcal{P}^{\mathcal{A}}_{(D_0,D_1)}$ is actually a mixture of projective measurements for the distribution $(D_0, D_1)$ and a collection of binary outcome projective measurements $\mathcal{P}_{b,x} = (Q_x^{\mathcal{A}}, P_x^{\mathcal{A}})$ if $b = 0$ and $\mathcal{P}_{b,x} = (P_x^{\mathcal{A}}, Q_x^{\mathcal{A}})$ if $b = 1$.

*Some Important Distributions.* The threshold implementation will be used to test whether a quantum distinguisher works well on the following distributions. The prime modulus $q$, the noise ratios $\alpha, \beta \in (0, 1)$ and $n, m \in N$ are all fixed parameters that will be soon clear from the context. For matrix $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$, and vectors $\mathbf{y} \in \mathbb{Z}_q^n, \mathbf{x} \in \mathbb{Z}_q^m$,

- Denote $D_{\mathsf{lwe}}^{\mathbf{A},\mathbf{y}}$ to be the distribution of $(\mathbf{A}, \mathbf{y}, \mathbf{s}^\intercal\mathbf{A} + \mathbf{e}^\intercal, \mathbf{s}^\intercal\mathbf{y} + e')$ where $\mathbf{s} \xleftarrow{\$} \mathbb{Z}_q^n, \mathbf{e} \sim D_{\mathbb{Z}_q^m,\alpha q}$ and $e' \sim D_{\mathbb{Z}_q,\beta q}$.
- Denote $D_{\mathsf{unif}}^{\mathbf{A},\mathbf{y}}$ to be the distribution of $(\mathbf{A}, \mathbf{y}, \mathbf{u}^\intercal, u')$ where $\mathbf{u} \xleftarrow{\$} \mathbb{Z}_q^m$ and $u' \xleftarrow{\$} \mathbb{Z}_q$.
- Denote $\bar{D}_{\mathsf{lwe}}^{\mathbf{A},\mathbf{x}}$ to be the distribution of $(\mathbf{A}, \mathbf{A}\mathbf{x}, \mathbf{u}^\intercal, \mathbf{u}^\intercal\mathbf{x}+e')$ where $e' \sim D_{\mathbb{Z}_q,\beta q}$ and $\mathbf{u}^\intercal = \mathbf{s}^\intercal\mathbf{A} + \mathbf{e}^\intercal$ for $\mathbf{s} \xleftarrow{\$} \mathbb{Z}_q^n, \mathbf{e} \sim D_{\mathbb{Z}_q^m,\alpha q}$.
- Denote $D_{\mathsf{gl}}^{\mathbf{A},\mathbf{x}}$ to be the distribution of $(\mathbf{A}, \mathbf{A}\mathbf{x}, \mathbf{u}^\intercal, \mathbf{u}^\intercal\mathbf{x}+e')$ where $e' \sim D_{\mathbb{Z}_q,\beta q}$ and $\mathbf{u} \xleftarrow{\$} \mathbb{Z}_q^m$.

For each of the above distribution $D$, we denote $D(i, g_i)$ to be the distribution of $(\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3 + c \cdot \hat{\mathbf{i}}^\intercal, v_4 + c \cdot g_i)$ where $\hat{\mathbf{i}}$ is the unit vector with its $i^{th}$ coordinate being 1, $c \xleftarrow{\$} \mathbb{Z}_q$, and $(\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3, v_4) \sim D$. It is easy to generate a sample from $D(i, g_i)$ given $i, g_i$ and a sample from $D$. Thus if we can efficiently distinguish between $D_0(i, g_i)$ and $D_1(i, g_i)$, then on input $(i, g_i)$, we can efficiently distinguish between $D_0$ and $D_1$.

The adversary can be described as an unitary $\mathcal{A}$ acting on a Hilbert space $\mathcal{H} = \mathcal{H}_{\mathrm{A}_\lambda} \otimes \mathcal{H}_{\mathrm{B}_\lambda} = \mathcal{H}_{\mathrm{R}_\lambda} \otimes \mathcal{H}_{\mathrm{AUX}_\lambda}$.

- $H_{\mathrm{A}_\lambda}$ stores the secret key state given by the challenger.
- $H_{\mathrm{B}_\lambda}$ is initialized to a quantum advice.
- $H_{\mathrm{R}_\lambda}$ stores the state returned to the challenger.
- $H_{\mathrm{AUX}_\lambda}$ is kept by the adversary.

We will omit $\lambda$ when it is clear from the context. We show the following result.

**Theorem 6 (Almost Optimal Search-to-Decision Reduction with Quantum Auxiliary Input).** *Let $n \in \mathbb{N}$ and $q$ be a prime modulus with $q = 2^{o(n)}$ and let $m \geq 2n \log q$, each parameterized by the security parameter $\lambda \in \mathbb{N}$ such that $m \leq \mathrm{poly}(\lambda)$. Let $\sqrt{8m} < \sigma < q/\sqrt{8m}$ and let $\alpha, \beta \in (0, 1)$ be noise ratios with $\beta/\alpha = 2^{o(n)}$, $2^{-o(n)} \leq \alpha\sigma \leq \mathsf{negl}(\lambda)$ and $\alpha\sigma/\beta \leq \mathsf{negl}(\lambda)$. Let $\mathcal{A} = \{(\mathcal{A}_{\lambda,\mathbf{A},\mathbf{y}}, \nu_\lambda)\}_{\lambda\in\mathbb{N}}$ be any non-uniform quantum algorithm consisting of a family of polynomial-sized quantum circuits and polynomial-sized advice states $\nu_\lambda \in \mathcal{D}(\mathcal{H}_{B_\lambda})$ which are independent of $\mathbf{A}$ and $\mathbf{y}$.*

*Assume the decisional $\mathsf{LWE}_{n,q,\alpha q}^m$ cannot be solved by a quantum algorithm running in time $\mathrm{poly}(\lambda, \sigma)$ with distinguishing advantage $1/\mathrm{poly}(\lambda, \sigma)$. If there exist functions $\varepsilon(\lambda) = 1/\mathrm{poly}(\lambda)$, $\gamma(\lambda) = 1/\mathrm{poly}(\lambda)$, $\delta(\lambda) = 2^{-\Theta(\lambda)}$ and a* QPT *distinguisher $\mathcal{D}$ such that (Fig. 2)*

$$\Pr\left[1 \leftarrow \mathsf{SearchToDecisionExpt}^{\mathcal{A},\mathcal{D}}(1^\lambda)\right] = \varepsilon(\lambda).$$

```
SearchToDecisionExpt^{A,D}(1^λ):

1. Sample $\mathbf{A} \xleftarrow{\$} \mathbb{Z}_q^{n \times m}$.
2. Generate $(|\psi_{\mathbf{y}}\rangle, \mathbf{y}) \leftarrow \mathsf{GenGauss}(\mathbf{A}, \sigma)$.
3. Generate $\rho_{\mathrm{R,\,Aux}} \leftarrow \mathcal{A}_{\lambda,\mathbf{A},\mathbf{y}}(|\psi_{\mathbf{y}}\rangle\langle\psi_{\mathbf{y}}| \otimes \nu_\lambda)$.
4. Let two-outcome projective measurements $\{\mathcal{P}_x^{\mathcal{D}}\}$ correspond to running $\mathcal{D}$ on
   samples $x$ and the auxiliary state in Aux. We approximately test whether
   $(\mathcal{D}, \rho_{\mathrm{Aux}})$ is a $\gamma$-good quantum distinguisher between the distributions $D_{\mathsf{lwe}}^{\mathbf{A},\mathbf{y}}$ and
   $D_{\mathsf{unif}}^{\mathbf{A},\mathbf{y}}$ by running $\mathsf{ATI}^{\gamma/6,\delta}_{\mathcal{P}^{\mathcal{D}},(D_{\mathsf{lwe}}^{\mathbf{A},\mathbf{y}},D_{\mathsf{unif}}^{\mathbf{A},\mathbf{y}}),1/2+\gamma}$ on $\rho_{\mathrm{Aux}}$ and output the result.
```

**Fig. 2.** The experiment $\mathsf{SearchToDecisionExpt}^{\mathcal{A},\mathcal{D}}(1^\lambda)$.

*Then, there exists a quantum extractor $\mathcal{E}$ that takes as input $\mathbf{A}$, $\mathbf{y}$ and system* Aux *of the state $\rho_{\mathrm{R,Aux}}$ and outputs a short vector in the coset $\Lambda_q^{\mathbf{y}}(\mathbf{A})$ in time* $\mathrm{poly}(\lambda, \sigma, 1/\gamma)$ *such that*

$$\Pr\left[\mathbf{x} \in \Lambda_q^{\mathbf{y}}(\mathbf{A}) \cap \mathcal{B}^m(\mathbf{0}, \sigma\sqrt{m/2}) \ : \ \begin{array}{c} \mathbf{A} \xleftarrow{\$} \mathbb{Z}_q^{n\times m} \\ (|\psi_{\mathbf{y}}\rangle, \mathbf{y}) \leftarrow \mathsf{GenGauss}(\mathbf{A},\sigma) \\ \rho_{\mathrm{R,Aux}} \leftarrow \mathcal{A}_{\lambda,\mathbf{A},\mathbf{y}}(|\psi_{\mathbf{y}}\rangle\langle\psi_{\mathbf{y}}| \otimes \nu_\lambda) \\ 1 \leftarrow \mathsf{ATI}^{\gamma/6,\delta}_{\mathcal{P}^{\mathcal{D}},(D_{\mathsf{lwe}}^{\mathbf{A},\mathbf{y}},D_{\mathsf{unif}}^{\mathbf{A},\mathbf{y}}),1/2+\gamma}(\rho_{\mathrm{Aux}}) \\ \mathbf{x} \leftarrow \mathcal{E}(\mathbf{A},\mathbf{y},\mathrm{Aux}) \end{array}\right] \geq 1 - \mathsf{negl}(\lambda).$$

## 6.1 Construction of the Extractor

In the subsection, we formally define our quantum extractor $\mathcal{E}$. $\mathcal{E}$ takes $\mathbf{A}, \mathbf{y}$ and the quantum state in Aux as input, and does the following:

```
E(A, y, Aux):

1. Set $\mathbf{x} = \mathbf{0}$, $\epsilon = \gamma/6$, $\gamma' = \gamma - 3\epsilon = \gamma/2$.
2. For each $i = 1, 2, \cdots, m$:
       For each $g_i \in [-\sigma\sqrt{m/2}, \cdots, \sigma\sqrt{m/2}]$:
         i. Let Aux store the current state of the quantum distinguisher.
        ii. Run $\overline{\mathsf{ATI}}^{\epsilon,\delta}_{\mathcal{P}^{\mathcal{D}},(D_{\mathsf{lwe}}^{\mathbf{A},\mathbf{y}}(i,g_i),D_{\mathsf{unif}}^{\mathbf{A},\mathbf{y}}),1/2+\gamma'}$ on the residual state in register Aux
            along with enough fresh ancillas initialized to $|0\rangle$.
       iii. If it outputs 1, set $\mathbf{x}_i = g_i$, and move on to the next guess.
        iv. If it outputs 0, move on to the next guess.
3. Output $\mathbf{x}$.
```

**Fig. 3.** The quantum extractor $\mathcal{E}(\mathbf{A}, \mathbf{y}, \mathrm{Aux})$.

By construction, the extractor runs in time $\mathrm{poly}(\lambda, \sigma, m, 1/\epsilon, \log \frac{1}{\delta}) = \mathrm{poly}(\lambda, \sigma, 1/\gamma)$.

## 6.2 Analysis of the Extractor

Before we analyze the success probability of our extractor, we make crucial observations on the distributions $D^{\mathbf{A},\mathbf{y}}_{\mathsf{lwe}}$, $D^{\mathbf{A},\mathbf{y}}_{\mathsf{unif}}$, $\bar{D}^{\mathbf{A},\mathbf{x}}_{\mathsf{lwe}}$ and $D^{\mathbf{A},\mathbf{x}}_{\mathsf{gl}}$.

**Lemma 7.** *For any $\mathbf{x} \in \mathcal{B}^m(\mathbf{0}, \sigma\sqrt{m/2})$, the statistical distance between $\bar{D}^{\mathbf{A},\mathbf{x}}_{\mathsf{lwe}}$ and $D^{\mathbf{A},\mathbf{Ax}}_{\mathsf{lwe}}$ is at most $\mathsf{negl}(\lambda)$.*

*Proof.* By noise smudging (Lemma 2), the statistical distance between the distribution $D_{\mathbb{Z}_q,\beta q}$ and the distribution of $\mathbf{e}^\intercal\mathbf{x} + e'$ where $e' \sim D_{\mathbb{Z}_q,\beta q}$ and $|\mathbf{e}^\intercal\mathbf{x}| \le \alpha q\sigma m$ is at most $\alpha\sigma m/\beta$. Notice that for any $\mathbf{x} \in \mathcal{B}^m(\mathbf{0}, \sigma\sqrt{m/2})$, when $\mathbf{e}$ is sampled from $D_{\mathbb{Z}_q^m,\alpha q}$, $|\mathbf{e}^\intercal\mathbf{x}| \ge \alpha q\sigma m$ with probability at most $2^{-\Omega(\lambda)}$(from Banaszczyk's tail bound [Ban93]).

Thus the statistical distance between $\bar{D}^{\mathbf{A},\mathbf{x}}_{\mathsf{lwe}}$ and $D^{\mathbf{A},\mathbf{Ax}}_{\mathsf{lwe}}$ is at most $\alpha\sigma m/\beta + 2^{-\Omega(\lambda)}$, which by our choice of parameters, is at most $\mathsf{negl}(\lambda)$.

**Lemma 8.** *For integer $i \in [m]$ and $g_i = \mathbf{x}_i$, $D^{\mathbf{A},\mathbf{x}}_{\mathsf{gl}}(i, g_i) = D^{\mathbf{A},\mathbf{x}}_{\mathsf{gl}}$.*

*For integer $i \in [m]$ and $g_i \neq \mathbf{x}_i$, $D^{\mathbf{A},\mathbf{x}}_{\mathsf{gl}}(i, g_i) = D^{\mathbf{A},\mathbf{Ax}}_{\mathsf{unif}}$.*

*Proof.* This follows directly from the definition, so we omit the proof.

Now we are ready to prove Theorem 6.

*Proof.* To prove Theorem 6, it suffices to prove that

$$\mathsf{Pr}\left[1 \leftarrow \mathsf{Game}^{\mathcal{A},\mathcal{D}}_0\left(1^\lambda\right)\right] \le \mathsf{negl}(\lambda)$$

where $\mathsf{Game}^{\mathcal{A},\mathcal{D}}_0$ is shown in Fig. 4.

```
$\underline{\mathsf{Game}^{\mathcal{A},\mathcal{D}}_0\left(1^\lambda\right)}$:

1. Sample $\mathbf{A} \xleftarrow{\$} \mathbb{Z}_q^{n\times m}$.
2. Generate $(|\psi_\mathbf{y}\rangle, \mathbf{y}) \leftarrow \mathsf{GenGauss}(\mathbf{A}, \sigma)$.
3. Generate $\rho_{\text{R, Aux}} \leftarrow \mathcal{A}_{\lambda,\mathbf{A},\mathbf{y}}(|\psi_\mathbf{y}\rangle\langle\psi_\mathbf{y}| \otimes \nu_\lambda)$.
4. Compute $b \leftarrow \mathsf{ATI}^{\gamma/6,\delta}_{\mathcal{P}^{\mathcal{D}},(D^{\mathbf{A},\mathbf{y}}_{\mathsf{lwe}},D^{\mathbf{A},\mathbf{y}}_{\mathsf{unif}}),1/2+\gamma}(\rho_{\text{Aux}})$. Abort if $b = 0$.
5. Compute $\mathbf{x} \leftarrow \mathcal{E}(\mathbf{A}, \mathbf{y}, \text{Aux})$.
6. Output 1 if $\mathbf{x} \notin \Lambda^{\mathbf{y}}_q(\mathbf{A}) \cap \mathcal{B}^m(\mathbf{0}, \sigma\sqrt{m/2})$; Otherwise, output 0.
```

**Fig. 4.** The game $\mathsf{Game}^{\mathcal{A},\mathcal{D}}_0\left(1^\lambda\right)$.

Let's consider the following sequence of hybrid distributions.

$\mathsf{H}_0$: This is the same as the game $\mathsf{Game}_0^{\mathcal{A},\mathcal{D}}(1^\lambda)$ defined in Fig. 4.

$\mathsf{H}_1$: This is the following distribution:

1. Sample $\mathbf{A} \xleftarrow{\$} \mathbb{Z}_q^{n\times m}$.
2. Sample a Gaussian vector $\mathbf{x}' \sim D_{\mathbb{Z}_q^m,\sigma/\sqrt{2}}$ and let $\mathbf{y} = \mathbf{A}\cdot\mathbf{x}' \bmod q$.
3. Generate $\rho_{\mathrm{R},\,\textsc{Aux}} \leftarrow \mathcal{A}_{\lambda,\mathbf{A},\mathbf{y}}(|\mathbf{x}'\rangle\langle\mathbf{x}'| \otimes \nu_\lambda)$.
4. Compute $b \leftarrow \mathsf{ATI}^{\gamma/6,\delta}_{\mathcal{P}^{\mathcal{D}},(D^{\mathbf{A},\mathbf{y}}_{\mathsf{lwe}},D^{\mathbf{A},\mathbf{y}}_{\mathsf{unif}}),1/2+\gamma}(\rho_{\textsc{Aux}})$. Abort if $b=0$.
5. Compute $\mathbf{x} \leftarrow \mathcal{E}(\mathbf{A},\mathbf{y},\textsc{Aux})$.
6. Output 1 if $\mathbf{x} \notin \Lambda_q^{\mathbf{y}}(\mathbf{A}) \cap \mathcal{B}^m(\mathbf{0},\sigma\sqrt{m/2})$; Otherwise, output 0.

$\mathsf{H}_2$: This is the following distribution:

1. Sample $\mathbf{A} \xleftarrow{\$} \mathbb{Z}_q^{n\times m}$.
2. Sample a Gaussian vector $\mathbf{x}' \sim D_{\mathbb{Z}_q^m,\sigma/\sqrt{2}}$ and let $\mathbf{y} = \mathbf{A}\cdot\mathbf{x}' \bmod q$.
3. Generate $\rho_{\mathrm{R},\,\textsc{Aux}} \leftarrow \mathcal{A}_{\lambda,\mathbf{A},\mathbf{y}}(|\mathbf{x}'\rangle\langle\mathbf{x}'| \otimes \nu_\lambda)$.
4. Compute $b \leftarrow \mathsf{ATI}^{\gamma/6,\delta}_{\mathcal{P}^{\mathcal{D}},(D^{\mathbf{A},\mathbf{x}'}_{\mathsf{gl}},D^{\mathbf{A},\mathbf{y}}_{\mathsf{unif}}),1/2+\gamma}(\rho_{\textsc{Aux}})$. Abort if $b=0$.
5. Compute $\mathbf{x} \leftarrow \mathcal{E}(\mathbf{A},\mathbf{y},\textsc{Aux})$.
6. Output 1 if $\mathbf{x} \notin \Lambda_q^{\mathbf{y}}(\mathbf{A}) \cap \mathcal{B}^m(\mathbf{0},\sigma\sqrt{m/2})$; Otherwise, output 0.

$\mathsf{H}_{3,k}$: This is the following distribution which replaces $\overline{\mathsf{ATI}}^{\epsilon,\delta}_{\mathcal{P}^{\mathcal{D}},(D^{\mathbf{A},\mathbf{y}}_{\mathsf{lwe}}(i,g_i),D^{\mathbf{A},\mathbf{y}}_{\mathsf{unif}}),1/2+\gamma'}$ in $\mathcal{E}$ with $\overline{\mathsf{ATI}}^{\epsilon,\delta}_{\mathcal{P}^{\mathcal{D}},\left(D^{\mathbf{A},\mathbf{x}'}_{\mathsf{gl}}(i,g_i),D^{\mathbf{A},\mathbf{y}}_{\mathsf{unif}}\right),1/2+\gamma'}$ one by one (recall the description of $\mathcal{E}$ defined in Fig. 3).

```
1. Sample $\mathbf{A} \xleftarrow{\$} \mathbb{Z}_q^{n\times m}$.
2. Sample a Gaussian vector $\mathbf{x}' \sim D_{\mathbb{Z}_q^m,\sigma/\sqrt{2}}$ and let $\mathbf{y} = \mathbf{A}\cdot\mathbf{x}' \bmod q$.
3. Generate $\rho_{\mathrm{R},\,\textsc{Aux}} \leftarrow \mathcal{A}_{\lambda,\mathbf{A},\mathbf{y}}(|\mathbf{x}'\rangle\langle\mathbf{x}'| \otimes \nu_\lambda)$.
4. Compute $b \leftarrow \mathsf{ATI}^{\gamma/6,\delta}_{\mathcal{P}^{\mathcal{D}},(D^{\mathbf{A},\mathbf{x}'}_{\mathsf{gl}},D^{\mathbf{A},\mathbf{y}}_{\mathsf{unif}}),1/2+\gamma}(\rho_{\textsc{Aux}})$. Abort if $b=0$.
5. Set $\mathbf{x}=\mathbf{0}$, $\epsilon=\gamma/6$, $\gamma'=\gamma-3\epsilon=\gamma/2$, $t=0$.
6. For each $i=1,2,\cdots,m$:
      For each $g_i \in [-\sigma\sqrt{m/2},\cdots,\sigma\sqrt{m/2}]$:
    i. Let Aux store the current state of the quantum distinguisher. $t \leftarrow t+1$.
    ii. If $t \le k$, run $\overline{\mathsf{ATI}}^{\epsilon,\delta}_{\mathcal{P}^{\mathcal{D}},\left(D^{\mathbf{A},\mathbf{x}'}_{\mathsf{gl}}(i,g_i),D^{\mathbf{A},\mathbf{y}}_{\mathsf{unif}}\right),1/2+\gamma'}$ on the residual state in register Aux along with enough fresh ancillas initialized to $|0\rangle$. Otherwise, run $\overline{\mathsf{ATI}}^{\epsilon,\delta}_{\mathcal{P}^{\mathcal{D}},\left(D^{\mathbf{A},\mathbf{y}}_{\mathsf{lwe}}(i,g_i),D^{\mathbf{A},\mathbf{y}}_{\mathsf{unif}}\right),1/2+\gamma'}$ on the residual state in register Aux along with enough fresh ancillas initialized to $|0\rangle$ .
    iii. If it outputs 1, set $\mathbf{x}_i=g_i$, and move on to the next guess.
    iv. If it outputs 0, move on to the next guess.
7. Output 1 if $\mathbf{x} \notin \Lambda_q^{\mathbf{y}}(\mathbf{A}) \cap \mathcal{B}^m(\mathbf{0},\sigma\sqrt{m/2})$; Otherwise, output 0.
```

We now show the following:

**Lemma 9.** *Assuming the quantum hardness of* $\mathsf{LWE}^m_{n,q,\alpha q}$*, the hybrids* $\mathsf{H}_0$ *and* $\mathsf{H}_1$ *are computationally indistinguishable,*

$$\mathsf{H}_0 \approx_c \mathsf{H}_1.$$

*Proof.* This follows directly from the Gaussian-collapsing property (Theorem 2) for quantum distinguishers with auxiliary states.

**Lemma 10.** *Assuming the quantum hardness of* $\mathsf{LWE}^m_{n,q,\alpha q}$*, the hybrids* $\mathsf{H}_1$ *and* $\mathsf{H}_2$ *are computationally indistinguishable,*

$$\mathsf{H}_1 \approx_c \mathsf{H}_2.$$

*Proof.* We prove the claim by contradiction.

Suppose $\mathsf{H}_1$ and $\mathsf{H}_2$ can be distinguished by a $\mathsf{QPT}$ algorithm $\mathcal{B}$ with advantage $1/\lambda^c$ for a constant $c > 0$ and infinitely many $\lambda$. Fix one such $\lambda$.

By standard averaging argument, for at least $\frac{1}{2\lambda^c}$ fraction of $(\mathbf{A}, \mathbf{x}')$ sampled according to $\mathbf{A} \xleftarrow{\$} \mathbb{Z}_q^{n\times m}$ and $\mathbf{x}' \sim D_{\mathbb{Z}_q^m, \sigma/\sqrt{2}}$, $\mathcal{B}$ can distinguish the result of running step 3–6 of $\mathsf{H}_1$ on $(\mathbf{A}, \mathbf{x}')$, and the result of running step 3–6 of $\mathsf{H}_2$ on $(\mathbf{A}, \mathbf{x}')$ with advantage at least $\frac{1}{2\lambda^c}$. Let's call those $(\mathbf{A}, \mathbf{x}')$ good. Then from Lemma 5, there exists a quantum circuit $\mathcal{C}$ of size $\mathrm{poly}(\lambda, 1/\epsilon, \log(1/\delta))$ such that for each good $(\mathbf{A}, \mathbf{x}')$, $\mathcal{C}(\rho_{\mathrm{AUX}}, \mathbf{A}, \mathbf{x}', \cdot)$ can distinguish samples from $\left(D^{\mathbf{A},\mathbf{x}'}_{\mathsf{gl}}, D^{\mathbf{A},\mathbf{y}}_{\mathsf{unif}}\right)$ and samples from $\left(D^{\mathbf{A},\mathbf{y}}_{\mathsf{lwe}}, D^{\mathbf{A},\mathbf{y}}_{\mathsf{unif}}\right)$ with advantage at least $\frac{1}{\mathrm{poly}(\lambda, 1/\epsilon, \log(1/\delta))}$.

As we can sample $D^{\mathbf{A},\mathbf{y}}_{\mathsf{unif}}$ by ourselves and $D^{\mathbf{A},\mathbf{y}}_{\mathsf{lwe}} \approx_s \bar{D}^{\mathbf{A},\mathbf{x}'}_{\mathsf{lwe}}$ (from Lemma 7 and the choice of parameters), there exists a polynomial size quantum circuit $\mathcal{C}'$ such that for each good $(\mathbf{A}, \mathbf{x}')$, $\mathcal{C}'(\rho_{\mathrm{AUX}}, \mathbf{A}, \mathbf{x}', \cdot)$ can distinguish samples from $D^{\mathbf{A},\mathbf{x}'}_{\mathsf{gl}}$ and $\bar{D}^{\mathbf{A},\mathbf{x}'}_{\mathsf{lwe}}$ with advantage at least $1/\lambda^d$ for some constant $d > 0$.

Recall that the only difference in $D^{\mathbf{A},\mathbf{x}'}_{\mathsf{gl}}$ and $\bar{D}^{\mathbf{A},\mathbf{x}'}_{\mathsf{lwe}}$ is whether $\mathbf{u}$ is sampled according to LWE or sampled uniformly. Now let's show how to leverage the fact to break $\mathsf{LWE}^m_{n,q,\alpha q}$ using this $\mathcal{C}'$ (Algorithm 1). Notice that for all the good $(\mathbf{A}, \mathbf{x}')$, line 3 passes with noticeable probability (by averaging arguments over the eigenspaces) and the residual state after running $\mathsf{ATI}$ and obtaining outcome 1 is still a good distinguisher (by Lemma 4). So Algorithm 1 breaks decisional $\mathsf{LWE}^m_{n,q,\alpha q}$ efficiently if Lemma 10 doesn't hold.

This ends our proof of the claim.

**Lemma 11.** *Assume that the decisional* $\mathsf{LWE}^m_{n,q,\alpha q}$ *cannot be solved by a quantum algorithm running in time* $\mathrm{poly}(\lambda, \sigma)$ *with distinguishing advantage* $1/\mathrm{poly}(\lambda, \sigma)$.

*The probability that hybrid* $\mathsf{H}_{3,k}$ *outputs 1 and the probability that hybrid* $\mathsf{H}_{3,k+1}$ *outputs 1 are* $\mathsf{negl}(\lambda)/\sigma$ *close. Formally, for* $0 \le k \le \sqrt{2}\sigma m^{3/2} - 1$,

$$\left|\mathsf{Pr}\left[\mathsf{H}_{3,k+1} = 1\right] - \mathsf{Pr}\left[\mathsf{H}_{3,k} = 1\right]\right| \le \mathsf{negl}(\lambda)/\sigma$$

*Proof.* The proof is the same with the proof of Lemma 10 except that we apply Lemma 6 instead of Lemma 5 and that we set the parameter $\mu$ in Lemma 6 as $1/\mathrm{poly}(\lambda, \sigma)$ instead of $1/\mathrm{poly}(\lambda)$. We omit the proof details.

**Algorithm 1:** An algorithm to break decisional $\mathsf{LWE}^m_{n,q,\alpha q}$ if Lemma 10 doesn't hold

```
Input  : Matrix $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$ and vector $\mathbf{u} \in \mathbb{Z}_q^n$ (and quantum advice $\nu_\lambda$).
Output: 0 or 1 (guess whether $\mathbf{u}$ is sampled from uniform or according to $\mathsf{LWE}^m_{n,q,\alpha q}$)
1 Sample a vector $\mathbf{x}' \sim D_{\mathbb{Z}_q^m,\sigma/\sqrt{2}}$ and let $\mathbf{y} = \mathbf{A}\cdot\mathbf{x}' \bmod q$.
2 Generate $\rho_{\mathrm{R,\,Aux}} \leftarrow \mathcal{A}_{\lambda,\mathbf{A},\mathbf{y}}(|\mathbf{x}'\rangle\langle\mathbf{x}'| \otimes \nu_\lambda)$.
3 Test whether $\mathcal{C}'(\rho_{\mathrm{Aux}}, \mathbf{A}, \mathbf{x}', \cdot)$ can be used to distinguish samples from $D_{\mathsf{gl}}^{\mathbf{A},\mathbf{x}'}$
  and samples from $\bar{D}_{\mathsf{lwe}}^{\mathbf{A},\mathbf{x}'}$ with advantage at least $1/\lambda^d$ by running $\mathsf{ATI}^{1/\lambda^{d+1},\delta}$ on
  it with threshold $1/2 + \frac{1}{4\lambda^d}$. If the ATI outputs 0 (it is not a good distinguisher),
  output a random guess and abort.
4 Denote the residual state (if not abort) in register Aux as $\rho'_{\mathrm{Aux}}$.
5 Sample $e' \sim D_{\mathbb{Z}_q,\beta q}$.Let $\mathbf{v} := (\mathbf{A}, \mathbf{A}\mathbf{x}', \mathbf{u}^\intercal, \mathbf{u}^\intercal\mathbf{x}' + e')$.
6 Run $\mathcal{C}'(\rho'_{\mathrm{Aux}}, \mathbf{A}, \mathbf{x}', \mathbf{v})$ and output the result.
```

**Lemma 12.** $\mathsf{H}_{3,\sqrt{2}\sigma m^{3/2}}$ *outputs 1 with negligible probability.*

*Proof.* We first define $\mathsf{Game}_1^{\mathcal{A},\mathcal{D}}(1^\lambda)$ in Fig. 5. It is the same as $\mathsf{H}_{3,\sqrt{2}\sigma m^{3/2}}$ except that it will output 1 if $\mathbf{x} \neq \mathbf{x}'$ (which is implied by $\mathbf{x} \notin \Lambda_q^{\mathbf{y}}(\mathbf{A}) \cap \mathcal{B}^m(\mathbf{0}, \sigma\sqrt{m/2})$), so to prove Lemma 12, it suffices to prove that $\mathsf{Game}_1^{\mathcal{A},\mathcal{D}}(1^\lambda)$ outputs 1 with negligible probability.

```
$\underline{\mathsf{Game}_1^{\mathcal{A},\mathcal{D}}(1^\lambda)}$:

1. Sample $\mathbf{A} \xleftarrow{\$} \mathbb{Z}_q^{n\times m}$.
2. Sample a Gaussian vector $\mathbf{x}' \sim D_{\mathbb{Z}_q^m,\sigma/\sqrt{2}}$ and let $\mathbf{y} = \mathbf{A}\cdot\mathbf{x}' \bmod q$.
3. Generate $\rho_{\mathrm{R,\,Aux}} \leftarrow \mathcal{A}_{\lambda,\mathbf{A},\mathbf{y}}(|\mathbf{x}'\rangle\langle\mathbf{x}'| \otimes \nu_\lambda)$.
4. Compute $b \leftarrow \mathsf{ATI}^{\gamma/6,\delta}_{\mathcal{P}^{\mathcal{D}},(D_{\mathsf{gl}}^{\mathbf{A},\mathbf{x}'},D_{\mathsf{unif}}^{\mathbf{A},\mathbf{y}}),1/2+\gamma}(\rho_{\mathrm{Aux}})$. Abort if $b = 0$.
5. Set $\mathbf{x} = \mathbf{0}$, $\epsilon = \gamma/6$, $\gamma' = \gamma - 3\epsilon = \gamma/2$.
6. For each $i = 1, 2, \cdots, m$:
       For each $g_i \in [-\sigma\sqrt{m/2}, \cdots, \sigma\sqrt{m/2}]$:
         i. Let Aux store the current state of the quantum distinguisher.
        ii. Run $\overline{\mathsf{ATI}}^{\epsilon,\delta}_{\mathcal{P}^{\mathcal{D}},\left(D_{\mathsf{gl}}^{\mathbf{A},\mathbf{x}'}(i,g_i),D_{\mathsf{unif}}^{\mathbf{A},\mathbf{y}}\right),1/2+\gamma'}$ on the residual state in register Aux
            along with enough fresh ancillas initialized to $|0\rangle$.
       iii. If it outputs 1, set $\mathbf{x}_i = g_i$, and move on to the next guess.
        iv. If it outputs 0, move on to the next guess.
7. Output 1 if $\mathbf{x} \neq \mathbf{x}'$; Otherwise, output 0.
```

**Fig. 5.** The game $\mathsf{Game}_1^{\mathcal{A},\mathcal{D}}(1^\lambda)$.

Notice that in step 6, we apply a sequence of projective measurements $\overline{\mathsf{ATI}}$ and set each coordinate of $\mathbf{x}'$ based on the measurement outcomes. By Quantum Union Bound (Lemma 1), $\Pr\left[1 \leftarrow \mathsf{Game}_1^{\mathcal{A},\mathcal{D}}\left(1^\lambda\right)\right]$ can be bounded by a union of events that for $\mathbf{x}'$ sampled according to $D_{\mathbb{Z}_q^m,\sigma/\sqrt{2}}$, $\mathsf{SubGame}^{\mathcal{A},\mathcal{D}}\left(1^\lambda, i, g_i, \mathbf{x}'\right)$ outputs 1:

$$\begin{aligned}&\Pr\left[1 \leftarrow \mathsf{Game}_1^{\mathcal{A},\mathcal{D}}\left(1^\lambda\right)\right]\\ \leq & 4\sum_{i=1}^{m}\sum_{g_i=-\sigma\sqrt{m/2}}^{\sigma\sqrt{m/2}} \Pr\left[1 \leftarrow \mathsf{SubGame}^{\mathcal{A},\mathcal{D}}\left(1^\lambda, i, g_i, \mathbf{x}'\right) : \mathbf{x}' \sim D_{\mathbb{Z}_q^m,\sigma/\sqrt{2}}\right]\end{aligned}$$

where $\mathsf{SubGame}^{\mathcal{A},\mathcal{D}}\left(1^\lambda, i, g_i, \mathbf{x}'\right)$ is defined in Fig. 6.

<u>$\mathsf{SubGame}^{\mathcal{A},\mathcal{D}}\left(1^\lambda, i, g_i, \mathbf{x}'\right)$</u>:

1. Sample $\mathbf{A} \xleftarrow{\$} \mathbb{Z}_q^{n\times m}$. Let $\mathbf{y} = \mathbf{A}\cdot\mathbf{x}' \bmod q$.
2. Generate $\rho_{\mathrm{R,\,Aux}} \leftarrow \mathcal{A}_{\lambda,\mathbf{A},\mathbf{y}}(|\mathbf{x}'\rangle\langle\mathbf{x}'| \otimes \nu_\lambda)$.
3. Compute $b \leftarrow \mathsf{ATI}^{\gamma/6,\delta}_{\mathcal{P}^{\mathcal{D}},(D_{\mathsf{gl}}^{\mathbf{A},\mathbf{x}'},D_{\mathsf{unif}}^{\mathbf{A},\mathbf{y}}),1/2+\gamma}(\rho_{\mathrm{Aux}})$. Abort if $b = 0$.
4. Set $\mathbf{x} = \mathbf{0}$, $\epsilon = \gamma/6$, $\gamma' = \gamma - 3\epsilon = \gamma/2$.
5. Run $\overline{\mathsf{ATI}}^{\epsilon,\delta}_{\mathcal{P}^{\mathcal{D}},\left(D_{\mathsf{gl}}^{\mathbf{A},\mathbf{x}'}(i,g_i),D_{\mathsf{unif}}^{\mathbf{A},\mathbf{y}}\right),1/2+\gamma'}$ on the residual state in register Aux along with enough fresh ancillas initialized to $|0\rangle$.
6. Output 1 if $\left(g_i \neq \mathbf{x}'_i\right.$ and $\overline{\mathsf{ATI}}$ outputs $\left.1\right)$ or $\left(g_i = \mathbf{x}'_i\right.$ and $\overline{\mathsf{ATI}}$ outputs $\left.0\right)$; otherwise, output 0.

**Fig. 6.** The game $\mathsf{SubGame}^{\mathcal{A},\mathcal{D}}\left(1^\lambda, i, g_i, \mathbf{x}'\right)$.

Now let's show for any fixed $i, g_i, \mathbf{x}'$,

$$\Pr\left[1 \leftarrow \mathsf{SubGame}^{\mathcal{A},\mathcal{D}}\left(1^\lambda, i, g_i, \mathbf{x}'\right)\right] \leq \mathsf{negl}(\lambda)/\sigma \tag{1}$$

*Case 1:* $g_i = \mathbf{x}'_i$ Consider the residual state $\rho'_{\mathrm{Aux}}$ of running step 3 and obtaining $b = 1$. From Lemma 4, running $\mathsf{ATI}^{\epsilon,\delta}_{\mathcal{P}^{\mathcal{D}},\left(D_{\mathsf{gl}}^{\mathbf{A},\mathbf{x}'},D_{\mathsf{unif}}^{\mathbf{A},\mathbf{y}}\right),1/2+\gamma'}$ on $\rho'_{\mathrm{Aux}}$, we will obtain 1 with probability at least $1 - 3\delta$.

From Lemma 8, when $g_i = \mathbf{x}'_i$, $D_{\mathsf{gl}}^{\mathbf{A},\mathbf{x}'}(i, g_i) = D_{\mathsf{gl}}^{\mathbf{A},\mathbf{x}'}$. Thus the output distribution of running $\mathsf{ATI}^{\epsilon,\delta}_{\mathcal{P}^{\mathcal{D}},\left(D_{\mathsf{gl}}^{\mathbf{A},\mathbf{x}'},D_{\mathsf{unif}}^{\mathbf{A},\mathbf{y}}\right),1/2+\gamma'}$ on $\rho'_{\mathrm{Aux}}$ is exactly the same as that of running $\overline{\mathsf{ATI}}^{\epsilon,\delta}_{\mathcal{P}^{\mathcal{D}},\left(D_{\mathsf{gl}}^{\mathbf{A},\mathbf{x}'}(i,g_i),D_{\mathsf{unif}}^{\mathbf{A},\mathbf{y}}\right),1/2+\gamma'}$ on $\rho'_{\mathrm{Aux}}$. Therefore,

$$\Pr\left[1 \leftarrow \mathsf{SubGame}^{\mathcal{A},\mathcal{D}}\left(1^\lambda, i, g_i, \mathbf{x}'\right)\right] \leq 3\delta \leq \mathsf{negl}(\lambda)/\sigma.$$

*Case 2:* $g_i \neq \mathbf{x}'_i$ Again consider the residual state $\rho'_{\text{AUX}}$ of running step 3 and obtaining $b = 1$. From Lemma 8, when $g_i \neq \mathbf{x}'_i$, $D_{\mathsf{gl}}^{\mathbf{A},\mathbf{x}'}(i, g_i) = D_{\mathsf{unif}}^{\mathbf{A},\mathbf{A}\mathbf{x}'} = D_{\mathsf{unif}}^{\mathbf{A},\mathbf{y}}$. Thus when running $\mathsf{ATI}^{\epsilon,\delta}_{\mathcal{P}^{\mathcal{D}},(D_{\mathsf{unif}}^{\mathbf{A},\mathbf{y}}, D_{\mathsf{unif}}^{\mathbf{A},\mathbf{y}}),1/2+\gamma'}$ on $\rho'_{\text{AUX}}$, we will obtain 1 with probability exactly $\Pr\left[1 \leftarrow \mathsf{SubGame}^{\mathcal{A},\mathcal{D}}\left(1^\lambda, i, g_i, \mathbf{x}'\right)\right]$.

As $\mathcal{P}^{\mathcal{D}}_{(D_{\mathsf{unif}}^{\mathbf{A},\mathbf{y}}, D_{\mathsf{unif}}^{\mathbf{A},\mathbf{y}})}$ only has eigenvalue $1/2 < 1/2 + \gamma' - \epsilon$ (any distinguisher cannot do better than outputting a random guess when facing $D_{\mathsf{unif}}^{\mathbf{A},\mathbf{y}}$ and $D_{\mathsf{unif}}^{\mathbf{A},\mathbf{y}}$), from Lemma 4, running $\mathsf{ATI}^{\epsilon,\delta}_{\mathcal{P}^{\mathcal{D}},(D_{\mathsf{unif}}^{\mathbf{A},\mathbf{y}}, D_{\mathsf{unif}}^{\mathbf{A},\mathbf{y}}),1/2+\gamma'}$ on any state, we cannot get 1 with probability greater than $\delta$, which implies that

$$\Pr\left[1 \leftarrow \mathsf{SubGame}^{\mathcal{A},\mathcal{D}}\left(1^\lambda, i, g_i, \mathbf{x}'\right)\right] \leq \delta \leq \mathsf{negl}(\lambda)/\sigma.$$

Summing up Eq. 1 and averaging over $\mathbf{x}'$, we can get that

$$\begin{aligned}
&\Pr\left[1 \leftarrow \mathsf{Game}_1^{\mathcal{A},\mathcal{D}}\left(1^\lambda\right)\right] \\
\leq &4 \sum_{i=1}^{m} \sum_{g_i=-\sigma\sqrt{m/2}}^{\sigma\sqrt{m/2}} \Pr\left[1 \leftarrow \mathsf{SubGame}^{\mathcal{A},\mathcal{D}}\left(1^\lambda, i, g_i, \mathbf{x}'\right) : \mathbf{x}' \sim D_{\mathbb{Z}_q^m, \sigma/\sqrt{2}}\right] \\
\leq &\mathsf{negl}(\lambda),
\end{aligned}$$

which ends the proof.

Recall that $\mathsf{H}_0$ is the same as the game $\mathsf{Game}_0^{\mathcal{A},\mathcal{D}}\left(1^\lambda\right)$, Theorem 6 follows directly from Lemma 9, Lemma 10, Lemma 11 over $\Theta(\sigma m^{3/2})$ pairs of consecutive hybrids, Lemma 12 and the observation that $\mathsf{H}_2 = \mathsf{H}_{3,0}$.

## 7 Proof of Theorem 5

We prove by contradiction. Let $\mathcal{A}$ be the QPT adversary and without loss of generality, we assume that the adversary submits $\mu = 0$ and assume that

$$\Pr\left[1 \leftarrow \mathsf{Expt}^{\Sigma,\mathcal{A}}(1^\lambda, 1)\right] - \Pr\left[1 \leftarrow \mathsf{Expt}^{\Sigma,\mathcal{A}}(1^\lambda, 0)\right] = \epsilon(\lambda),$$

where $\epsilon(\lambda)$ is inverse polynomial, $\mathsf{Expt}^{\Sigma,\mathcal{A}}(1^\lambda, b)$ is defined as Fig. 1 and $\Sigma = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Revoke})$.

We decompose the adversary into two QPT algorithms $\mathcal{A}, \mathcal{D}$ where given input state $|\psi_{\mathbf{y}}\rangle$, $\mathcal{A}$ generates the state $\rho_{\text{R,AUX}}$. After returning system R to the challenger, $\mathcal{D}$ takes $\rho_{\text{AUX}}$ and responds to the challenge. Then $\mathcal{A}, \mathcal{D}$ satisfy

$$\Pr\left[1 \leftarrow \mathsf{SecurityExpt}^{\mathcal{A},\mathcal{D}}(1^\lambda, 1)\right] - \Pr\left[1 \leftarrow \mathsf{SecurityExpt}^{\mathcal{A},\mathcal{D}}(1^\lambda, 0)\right] = \epsilon(\lambda)$$

where $\mathsf{SecurityExpt}^{\mathcal{A},\mathcal{D}}$ is the experiment shown in Fig. 7, because the inefficient revocation implements $\mathsf{Revoke}(\mathsf{MSK}, \mathsf{PK}, \sigma)$.

$\underline{\mathsf{SecurityExpt}^{\mathcal{A},\mathcal{D}}\left(1^\lambda, b\right)}$:

1. Sample $\mathbf{A} \xleftarrow{\$} \mathbb{Z}_q^{n\times m}$.
2. Generate $(|\psi_{\mathbf{y}}\rangle, \mathbf{y}) \leftarrow \mathsf{GenGauss}(\mathbf{A}, \sigma)$.
3. Generate $\rho_{R,\,\text{Aux}} \leftarrow \mathcal{A}_{\lambda,\mathbf{A},\mathbf{y}}(|\psi_{\mathbf{y}}\rangle\langle\psi_{\mathbf{y}}| \otimes \nu_\lambda)$.
4. Apply inefficient revocation on system R, if it fails, output Invalid.
5. If $b = 0$, sample $(\mathbf{A}, \mathbf{y}, \mathbf{u}^\intercal, u') \sim D_{\mathsf{lwe}}^{\mathbf{A},\mathbf{y}}$. If $b = 1$, sample $(\mathbf{A}, \mathbf{y}, \mathbf{u}^\intercal, u') \sim D_{\mathsf{unif}}^{\mathbf{A},\mathbf{y}}$.
6. Run $b' \leftarrow \mathcal{D}(\mathbf{A}, \mathbf{y}, \mathbf{u}^\intercal, u', \text{Aux})$ and output $b'$.

**Fig. 7.** The experiment $\mathsf{SecurityExpt}^{\mathcal{A},\mathcal{D}}\left(1^\lambda, b\right)$.

**Lemma 13.** *For adversary $\mathcal{A}, \mathcal{D}$ that satisfy*

$$\mathsf{Pr}\left[1 \leftarrow \mathsf{SecurityExpt}^{\mathcal{A},\mathcal{D}}(1^\lambda, 1)\right] - \mathsf{Pr}\left[1 \leftarrow \mathsf{SecurityExpt}^{\mathcal{A},\mathcal{D}}(1^\lambda, 0)\right] = \epsilon(\lambda),$$

*they also satisfy*

$$\mathsf{Pr}\left[1 \leftarrow \mathsf{ATISecurityExpt}^{\mathcal{A},\mathcal{D},\gamma}(1^\lambda)\right] \geq \frac{\epsilon(\lambda)}{4} - \mathsf{negl}.$$

*for $\gamma = \frac{3\epsilon}{14}$ where $\mathsf{ATISecurityExpt}^{\mathcal{A},\mathcal{D}}$ is shown in Fig. 8.*

$\underline{\mathsf{ATISecurityExpt}^{\mathcal{A},\mathcal{D},\gamma}\left(1^\lambda, b\right)}$:

1. Sample $\mathbf{A} \xleftarrow{\$} \mathbb{Z}_q^{n\times m}$.
2. Generate $(|\psi_{\mathbf{y}}\rangle, \mathbf{y}) \leftarrow \mathsf{GenGauss}(\mathbf{A}, \sigma)$.
3. Generate $\rho_{R,\,\text{Aux}} \leftarrow \mathcal{A}_{\lambda,\mathbf{A},\mathbf{y}}(|\psi_{\mathbf{y}}\rangle\langle\psi_{\mathbf{y}}| \otimes \nu_\lambda)$.
4. Apply inefficient revocation on system R, if it fails, output Invalid.
5. Run $\mathsf{ATI}^{\frac{\gamma}{6},\delta}_{\mathcal{P}^{\mathcal{D}},\left(D_{\mathsf{lwe}}^{\mathbf{A},\mathbf{y}}, D_{\mathsf{unif}}^{\mathbf{A},\mathbf{y}}\right),\frac{1}{2}+\gamma}$ where $\delta = 2^{-\Theta(\lambda)}$ on system Aux and output the result.

**Fig. 8.** The experiment $\mathsf{ATISecurityExpt}^{\mathcal{A},\mathcal{D},\varepsilon}\left(1^\lambda, b\right)$.

*Proof.* Suppose that revocation succeeds with probability $p$. The residual state $\rho_{\text{Aux}}$ satisfies

$$\mathbb{E}\left[\mathrm{Tr}\left[\mathcal{P}^{\mathcal{D}}_{\left(D_{\mathsf{lwe}}^{\mathbf{A},\mathbf{y}}, D_{\mathsf{unif}}^{\mathbf{A},\mathbf{y}}\right)} \rho_{\text{Aux}}\right] |\text{Revocation succeeds on R}\right] \geq \frac{1}{2} + \frac{\epsilon}{2p}.$$

By averaging argument and the definition of threshold implementation Theorem 3,

$$\mathbb{E}\left[\operatorname{Tr}\left[\mathsf{TI}_{\frac{1}{2}+\frac{\epsilon}{4}}\left(\mathcal{P}^{\mathcal{D}}_{(D^{\mathbf{A},\mathbf{y}}_{\mathsf{lwe}},D^{\mathbf{A},\mathbf{y}}_{\mathsf{unif}})}\right)\rho_{\mathrm{Aux}}\right]|\text{Revocation succeeds on R}\right] \geq \frac{\epsilon}{4p}.$$

By Lemma 4, if we set $\delta = 2^{-\Theta(\lambda)}$ we have,

$$\begin{aligned}
&\mathsf{Pr}\left[\mathsf{ATI}^{\frac{\gamma}{6},\delta}_{\mathcal{P}^{\mathcal{D}},(D^{\mathbf{A},\mathbf{y}}_{\mathsf{lwe}},D^{\mathbf{A},\mathbf{y}}_{\mathsf{unif}}),\frac{1}{2}+\gamma}(\rho_{\mathrm{Aux}}) = 1|\text{Revocation succeeds on R}\right]\\
=&\mathbb{E}\left[\operatorname{Tr}\left[\mathsf{ATI}^{\frac{\gamma}{6},\delta}_{\mathcal{P}^{\mathcal{D}},(D^{\mathbf{A},\mathbf{y}}_{\mathsf{lwe}},D^{\mathbf{A},\mathbf{y}}_{\mathsf{unif}}),\frac{1}{2}+\gamma}\rho_{\mathrm{Aux}}\right]|\text{Revocation succeeds on R}\right]\\
\geq&\mathbb{E}\left[\operatorname{Tr}\left[\mathsf{TI}_{\frac{1}{2}+\frac{\epsilon}{4}}\left(\mathcal{P}^{\mathcal{D}}_{(D^{\mathbf{A},\mathbf{y}}_{\mathsf{lwe}},D^{\mathbf{A},\mathbf{y}}_{\mathsf{unif}})}\right)\rho_{\mathrm{Aux}}\right]|\text{Revocation succeeds on R}\right] - \delta\\
\geq&\frac{\epsilon}{4p} - \mathsf{negl}.
\end{aligned}$$

Using the above lemma we can construct Algorithm 2 for solving $\mathsf{SIS}^m_{n,q,\sigma\sqrt{2m}}$ problem using the adversary $\mathcal{A},\mathcal{D}$. As for our choice of parameters, the hardness of $\mathsf{LWE}^m_{n,q,\alpha q}$ implies the hardness of $\mathsf{SIS}^m_{n,q,\sigma\sqrt{2m}}$, Theorem 5 follows directly from the correctness of Algorithm 2, which we show in the following claim.

```
Algorithm 2: SIS_Solver($\mathbf{A}$)
  Input: Matrix $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$.
  Output: Vector $\mathbf{x} \in \mathbb{Z}^m$.
1 Generate a Gaussian state $(|\psi_{\mathbf{y}}\rangle, \mathbf{y}) \leftarrow \mathsf{GenGauss}(\mathbf{A},\sigma)$ with
      $|\psi_{\mathbf{y}}\rangle = \sum_{\mathbf{x}\in\mathbb{Z}_q^m,\ \mathbf{A}\mathbf{x}=\mathbf{y} \pmod q} \rho_\sigma(\mathbf{x})\,|\mathbf{x}\rangle$
  for some vector $\mathbf{y} \in \mathbb{Z}_q^n$.
2 Run $\mathcal{A}$ to generate a bipartite state $\rho_{R,\mathrm{Aux}}$ in systems $\mathcal{H}_{\mathrm{R}} \otimes \mathcal{H}_{\mathrm{AUX}}$ with
  $\mathcal{H}_{\mathrm{R}} = \mathcal{H}_q^m$.
3 Run $\mathsf{ATI}^{\gamma/6,\delta}_{\mathcal{P}^{\mathcal{D}},(D^{\mathbf{A},\mathbf{y}}_{\mathsf{lwe}},D^{\mathbf{A},\mathbf{y}}_{\mathsf{unif}}),1/2+\gamma}$ on system Aux, abort if the output is 0.
4 Run the extractor $\mathcal{E}(\mathbf{A},\mathbf{y},\mathrm{Aux})$ from Theorem 6, and let $\mathbf{x}_1 \in \mathbb{Z}_q^n$ denote
  the outcome.
5 Measure system R in the computational basis, and let $\mathbf{x}_0 \in \mathbb{Z}_q^n$ denote the
  outcome.
6 Output the vector $\mathbf{x} = \mathbf{x}_1 - \mathbf{x}_0$.
```

*Claim.* Algorithm 2 solves $\mathsf{SIS}^m_{n,q,\sigma\sqrt{2m}}$ with inverse polynomial probability when $\mathcal{A},\mathcal{D}$ is a successful adversary.

*Proof.* Suppose $\mathcal{A},\mathcal{D}$ is a successful adversary. To show that Algorithm 2 can obtain a short non-zero solution $\mathbf{x}$ we prove the following two statements:

$\underline{\mathsf{SimultExtractionExpt}^{\mathcal{A},\mathcal{D}}(1^\lambda)}$:

1. Sample $\mathbf{A} \xleftarrow{\$} \mathbb{Z}_q^{n\times m}$.
2. Generate $(|\psi_{\mathbf{y}}\rangle, \mathbf{y}) \leftarrow \mathsf{GenGauss}(\mathbf{A}, \sigma)$.
3. Generate $\rho_{R,\,\text{Aux}} \leftarrow \mathcal{A}_{\lambda,\mathbf{A},\mathbf{y}}(|\psi_{\mathbf{y}}\rangle\langle\psi_{\mathbf{y}}| \otimes \nu_\lambda)$.
4. Run $\mathsf{ATI}^{\gamma/6,\delta}_{\mathcal{P}^{\mathcal{D}},(D^{\mathbf{A},\mathbf{y}}_{\mathsf{lwe}},D^{\mathbf{A},\mathbf{y}}_{\mathsf{unif}}),1/2+\gamma}$ on system Aux, abort if the output is 0.
5. Run the extractor $\mathcal{E}(\mathbf{A}, \mathbf{y}, \text{Aux})$ from Theorem 6, and let $\mathbf{x}_1 \in \mathbb{Z}_q^n$ denote the outcome. Abort if $\mathbf{x}_1 \notin \Lambda_q^{\mathbf{y}}(\mathbf{A}) \cap \mathcal{B}^m(\mathbf{0}, \sigma\sqrt{m/2})$.
6. Apply inefficient revocation on system R, abort if it fails; Otherwise, output 1.

**Fig. 9.** The experiment $\mathsf{SimultExtractionExpt}^{\mathcal{A},\mathcal{D}}(1^\lambda)$.

– The probability that on system Aux the extractor $\mathcal{E}$ extracts a short preimage $\mathbf{x}_1$ of $\mathbf{y}$ and revocation succeeds on R is inverse polynomial

$$\mathsf{Pr}\left[\mathsf{SimultExtractionExpt}^{\mathcal{A},\mathcal{D}}(1^\lambda) = 1\right] = \frac{1}{\text{poly}(\lambda)}.$$

where $\mathsf{SimulExtractionExpt}$ is defined as Fig. 9.

– Suppose that revocation succeeds with probability $\varepsilon(\lambda)$ conditioned on the extraction being successful. Then instead of running revocation on R, if we measure register R in computational basis and obtain result $\mathbf{x}_0$, the probability that $\mathbf{x}_0$ is a short preimage of $\mathbf{y}$ that is different from $\mathbf{x}_1$ is $\varepsilon(\lambda) - \mathsf{negl}(\lambda)$ conditioned on the extraction being successful.

If both statements are true, by basic probability arguments we prove the claim.

The first statement follows from Lemma 13 and Theorem 6. Let $\mathsf{GoodDecryptor}$ denote the event that we pass the $\mathsf{ATI}$ test on step 4. Let $\mathsf{RevocationSuc}$ denote the event that the inefficient revocation succeeds on system $R$ on step 6. Let $\mathsf{ExtractionSuc}$ denote the event that $\mathbf{x}_1$ is a short preimage of $\mathbf{y}$ on step 5. Since step 4–5 and step 6 commute, by Lemma 13,

$$\mathsf{Pr}\left[\mathsf{RevocationSuc} \wedge \mathsf{GoodDecryptor}\right] = \frac{1}{\text{poly}(\lambda)}.$$

By Theorem 6,

$$\mathsf{Pr}\left[\mathsf{ExtractionSuc} \mid \mathsf{GoodDecryptor}\right] \geq 1 - \mathsf{negl}(\lambda).$$

By basic probability calculation,

$$\mathsf{Pr}\left[\mathsf{RevocationSuc} \wedge \mathsf{ExtractionSuc}\right] = \frac{1}{\text{poly}(\lambda)}.$$

Now we prove the second statement. We show that given a specific short preimage $\mathbf{x}_1$ of $\mathbf{y}$ and a state $\rho_{\mathrm{R}}$ such that revocation succeeds on R with probability $\varepsilon(\lambda)$, if we measure R under computational basis, we obtain a short preimage $\mathbf{x}_0$ of $\mathbf{y}$ that is different from $\mathbf{x}_1$ with probability $\varepsilon(\lambda) - \mathsf{negl}(\lambda)$. Define the set of short preimages $\mathcal{S} = \left\{\mathbf{x} | \mathbf{A}\mathbf{x} = \mathbf{y}, \|\mathbf{x}\| \leq \sigma\sqrt{\frac{m}{2}}\right\}$ and

$$|\psi'_{\mathbf{y}}\rangle = \left(\sum_{\mathbf{x}\in\mathcal{S}} \rho_{\frac{\sigma}{\sqrt{2}}}(\mathbf{x})\right)^{-\frac{1}{2}} \sum_{\mathbf{x}\in\mathcal{S}} \rho_\sigma(\mathbf{x})|\mathbf{x}\rangle$$

be a 'truncated' Gaussian coset state. Consider the following projectors

- $\mathit{\Pi}_0 = \sum_{\mathbf{x}\in\mathcal{S},\mathbf{x}\neq\mathbf{x}_1} |\mathbf{x}\rangle\langle\mathbf{x}|$ is a projector that projects onto all short preimages we want.
- $\mathit{\Pi}_1 = |\psi'_{\mathbf{y}}\rangle\langle\psi'_{\mathbf{y}}|$ is the approximate revocation projector. The trace distance between $\mathit{\Pi}_1$ and the actual revocation projector $|\psi_{\mathbf{y}}\rangle\langle\psi_{\mathbf{y}}|$ is negligible by Banaszczyk's tail bound [Ban93].

Suppose that $\mathbf{A}$ is a full-rank matrix, if $\mathrm{Tr}[|\psi_{\mathbf{y}}\rangle\langle\psi_{\mathbf{y}}|\rho_{\mathrm{R}}] = \varepsilon$ we have

$$\begin{aligned}\mathrm{Tr}[\mathit{\Pi}_0\rho_{\mathrm{R}}] &\geq \mathrm{Tr}[\mathit{\Pi}_1\mathit{\Pi}_0\rho_{\mathrm{R}}] \\ &\geq \mathrm{Tr}[\mathit{\Pi}_1\rho_{\mathrm{R}}] - \mathsf{negl}(\lambda) \\ &\geq \mathrm{Tr}[|\psi_{\mathbf{y}}\rangle\langle\psi_{\mathbf{y}}|\rho_{\mathrm{R}}] - \mathsf{negl}(\lambda) \\ &= \varepsilon - \mathsf{negl}(\lambda).\end{aligned}$$

where the second inequality follows from Lemma 3. Note that $\mathbf{A}$ is full-rank with $1 - \mathsf{negl}(\lambda)$ probability. Combine all arguments above, we proof this claim.

## 8 Applications

Combining our result with [APV23], we obtain constructions for

- Public-Key Encryption with Classical Key Revocation.
- Key-Revocable Fully Homomorphic Encryption.
- Revocable Pseudorandom Functions.

### 8.1 Public-Key Encryption with Classical Key Revocation

A public-key encryption with classical key-revocation is a public-key encryption such that whenever we want to perform key revocation:

- The lessee runs $\mathsf{Delete}$ on its quantum secret key $\rho_{\mathsf{SK}}$ and produce a classical certificate $\pi$.
- The lessor runs $\mathsf{Revoke}$ on input $\pi$ and output $\mathsf{Valid}$ if it is a valid certificate.

The security of such scheme captures the idea that if an adversary produces a certificate $\pi$ that passes the revocation then the remaining adversary cannot distinguish between a ciphertext of chosen message from a random ciphertext. In [APV23], they built a public-key encryption with classical key-revocation assuming the security of key-revocable Dual-Regev encryption. Combine with our result, we obtain the following theorem.

**Theorem 7.** *Assuming the polynomial hardness of* $\mathsf{LWE}$ *with sub-exponential modulus. The scheme* $\mathsf{CRevDual} = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Delete}, \mathsf{Revoke})$ *(Construction 2, [APV23]) is a secure public-key encryption with classical key-revocation (Definition 7.1, 7.2, [APV23]).*

## 8.2 Key-Revocable Fully Homomorphic Encryption

A key-revocable fully homomorphic encryption is a fully homomorphic encryption with quantum key revocation just like the key-revocable Dual-Regev Encryption. In [APV23], they built a key-revocable fully homomorphic encryption assuming the security of key-revocable Dual-Regev encryption. Meanwhile, this construction can be adapted to feature classical revocation via techniques used in public-key encryption with classical key-revocation mentioned above. Combine with our result, we obtain the following theorem.

**Theorem 8.** *Assuming the polynomial hardness of* $\mathsf{LWE}$ *and* $\mathsf{SIS}$ *with sub-exponential modulus. The scheme* $\mathsf{RevDualGSW} = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Eval}, \mathsf{Revoke})$ *(Construction 3, [APV23]) is a secure key-revocable fully homomorphic encryption (Definition 5.3, [APV23]). Meanwhile, this construction can be adapted to feature classical revocation via (Construction 2, [APV23]).*

## 8.3 Revocable Pseudorandom Functions

A key-revocable (or simply, revocable) pseudorandom function is a weak pseudorandom function with its evaluation key revocable. The $\mu$-security of such scheme captures the idea that if the revocation succeeds, the remaining adversary cannot distinguish between $\mu$ images $y_1 = \mathsf{PRF}(x_1), y_2 = \mathsf{PRF}(x_2), \cdots, y_\mu = \mathsf{PRF}(y_\mu)$ from $\mu$ random preimages $x_1, x_2, \cdots, x_\mu$ and uniform random values $y_1, y_2, \cdots, y_\mu$. Meanwhile, this construction can also be adapted to feature classical revocation. Combine with our result, we obtain the following theorem.

**Theorem 9.** *Assuming the polynomial hardness of* $\mathsf{LWE}$ *and* $\mathsf{SIS}$ *with sub-exponential modulus. The scheme* $(\mathsf{Gen}, \mathsf{PRF}, \mathsf{Eval}, \mathsf{Revoke})$ *(Construction 5, [APV23]) is a* poly*-secure revocable* $\mathsf{PRF}$ *scheme (Definition 9.2, 9.3, [APV23]). Meanwhile, this construction can be adapted to feature classical revocation via (Construction 2, [APV23]).*

**Acknowledgments.** We thank Jiahui Liu for patiently answering our questions about her work [CGJL23] and thank anonymous reviewers for their helpful comments.

Prabhanjan Ananth is supported by the National Science Foundation under Grant No. 2329938. This work was done while Zihan Hu was an intern at Shanghai Qi Zhi Institute.

## References

[Aar09] Aaronson, S.: Quantum copy-protection and quantum money. In: 2009 24th Annual IEEE Conference on Computational Complexity, pp. 229–242. IEEE (2009)

[Aar16] Aaronson, S.: The Complexity of Quantum States and Transformations: From Quantum Money to Black Holes (2016). arXiv:1607.05256 [quant-ph]

[AGKZ20] Amos, R., Georgiou, M., Kiayias, A., Zhandry, M.: One-shot signatures and applications to hybrid quantum/classical authentication. In: Proceedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing, pp. 255–268 (2020)

[AHH24] Ananth, P., Hu, Z., Huang, Z.: Quantum key- revocable dual-regev encryption, revisited. Cryptology ePrint Archive, Paper 2024/738 (2024). https://eprint.iacr.org/2024/738

[AKN+23] Agrawal, S., Kitagawa, F., Nishimaki, R., Yamada, S., Yamakawa, T.: Public key encryption with secure key leasing. arXiv preprint arXiv:2302.11663 (2023)

[ALL+21] Aaronson, S., Liu, J., Liu, Q., Zhandry, M., Zhang, R.: New approaches for quantum copy-protection. In: Malkin, T., Peikert, C. (eds.) CRYPTO 2021. LNCS, vol. 12825, pp. 526–555. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-84242-0_19

[APV23] Ananth, P., Poremba, A., Vaikuntanathan, V.: Revocable cryptography from learning with errors. In: Rothblum, G., Wee, H. (eds) TTCC 2023, Part IV. LNCS, vol. 14372, pp. 93–122. Springer, Cham (2023). ISBN 978-3-031-48623-4. https://doi.org/10.1007/978-3-031-48624-1_4

[Ban93] Banaszczyk, W.: New bounds in some transference theorems in the geometry of numbers. Mathematische Annalen **296**(4), 625–636 (1993). http://eudml.org/doc/165105

[BCM+21] Brakerski, Z., Christiano, P., Mahadev, U., Vazirani, U., Vidick, T.: A cryptographic test of quantumness and certifiable randomness from a single quantum device (2021). arXiv:1804.00640 [quant-ph]

[BDGM20] Brakerski, Z., Döttling, N., Garg, S., Malavolta, G.: Factoring and pairings are not necessary for IO: circularsecure LWE suffices. Cryptology ePrint Archive (2020)

[BGG+14] Boneh, D., et al.: Fully key-homomorphic encryption, arithmetic circuit ABE and compact garbled circuits. In: Nguyen, P.Q., Oswald, E. (eds.) EUROCRYPT 2014. LNCS, vol. 8441, pp. 533–556. Springer, Heidelberg (2014). https://doi.org/10.1007/978-3-642-55220-5_30

[BI20] Broadbent, A., Islam, R.: Quantum encryption with certified deletion. In: Pass, R., Pietrzak, K. (eds.) TCC 2020. LNCS, vol. 12552, pp. 92–122. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-64381-2_4

[BK22] Bartusek, J., Khurana, D.: Cryptography with certified deletion (2022). https://doi.org/10.48550/ARXIV.2207.01754, https://arxiv.org/abs/2207.01754

[BL20] Broadbent, A., Lord, S.: Uncloneable quantum encryption via oracles. In: Flammia, S.T. (ed.) 15th Conference on the Theory of Quantum Computation, Communication and Cryptography (TQC 2020), vol. 158. Leibniz International Proceedings in Informatics (LIPIcs). Dagstuhl, Germany: Schloss Dagstuhl– Leibniz-Zentrum für Informatik, pp. 4:1–4:22 (2020). https://doi.org/10.4230/LIPIcs.TQC.2020.4

[CGJL23] Chardouvelis, O., Goyal, V., Jain, A., Liu, J.: Quantum key leasing for PKE and FHE with a classical lessor. arXiv preprint arXiv:2310.14328 (2023)

[DGT+10] Dodis, Y., Goldwasser, S., Kalai, Y.T., Peikert, C., Vaikuntanathan, V.: Public-key encryption schemes with auxiliary inputs. In: Micciancio, D. (eds.) TCC 2010. LNCS, vol. 5978, pp. 361–381. Springer, Cham (2010). ISBN 978-3-642-11799-2. https://doi.org/10.1007/978-3-642-11799-2_22

[Die82] Dieks, D.G.B.J.:Communication by EPR devices. Phys. Lett. A **92**(6), 271–272 (1982)

[Gao15] Gao, J.: Quantum union bounds for sequential projective measurements. Phys. Rev. A **92**(5), 052331 (2015)

[GPV08] Gentry, C., Peikert, C., Vaikuntanathan, V.: Trapdoors for hard lattices and new cryptographic constructions. In: Proceedings of the Fortieth Annual ACM Symposium on Theory of Computing, pp. 197–206 (2008)

[HMNY21] Hiroka, T., Morimae, T., Nishimaki, R., Yamakawa, T.: Quantum encryption with certified deletion, revisited: public key, attribute-based, and classical communication (2021). arXiv: 2105.05393 [quant-ph]

[LMZ23] Liu, J., Montgomery, H., Zhandry, M.: Another round of breaking and making quantum money: how to not build it from lattices, and more. In: Hazay, C., Stam, M. (eds.) EUROCRYPT 2023. LNCS, vol. 14004, pp. 611–638. Springer, Cham (2023). https://doi.org/10.1007/978-3-031-30545-0_21

[Mah18] Mahadev, U.: Classical homomorphic encryption for quantum circuits. In: Thorup, M. (ed.) 59th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2018, Paris, France, 7–9 October 2018, pp. 332–338. IEEE Computer Society (2018). https://doi.org/10.1109/FOCS.2018.00039

[MW05] Marriott, C., Watrous, J.: Quantum Arthur-Merlin Games (2005). arXiv: cs/0506068 [cs.CC]

[Por22] Poremba, A.: Quantum proofs of deletion for learning with errors (2022). https://doi.org/10.48550/ARXIV.2203.01610

[Qua20] Quach, W.: UC-secure OT from LWE, revisited. In: Galdi, C., Kolesnikov, V. (eds.) SCN 2020. LNCS, vol. 12238, pp. 192–211. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-57990-6_10

[WZ82] Wootters, W.K., Zurek, W.H.: A single quantum cannot be cloned. Nature **299**(5886), 802–803 (1982)

[Zha20] Zhandry, M.: Schrödinger's pirate: how to trace a quantum decoder. Cryptology ePrint Archive, Paper 2020/1191. https://eprint.iacr.org/2020/1191.2020

# On Black-Box Separations of Quantum Digital Signatures from Pseudorandom States

Andrea Coladangelo[1] and Saachi Mutreja[2](✉)

[1] University of Washington, Seattle, USA
coladan@cs.washington.edu
[2] Columbia University, New York, USA
sm5540@columbia.edu

**Abstract.** It is well-known that digital signatures can be constructed from one-way functions in a black-box way. While one-way functions are essentially the minimal assumption in classical cryptography, this is not the case in the quantum setting. A variety of qualitatively weaker and *inherently quantum* assumptions (e.g. EFI pairs, one-way state generators, and pseudorandom states) are known to be sufficient for non-trivial quantum cryptography.

While it is known that commitments, zero-knowledge proofs, and even multiparty computation can be constructed from these assumptions, it has remained an open question whether the same is true for quantum digital signatures schemes (QDS). In this work, we show that there *does not* exist a black-box construction of a QDS scheme *with classical signatures* from pseudorandom states with linear, or greater, output length. Our result complements that of Morimae and Yamakawa (2022), who described a *one-time* secure QDS scheme with classical signatures, but left open the question of constructing a standard *multi-time* secure one.

## 1 Introduction

One of the foundational goals of cryptography is to study the *minimal assumptions* needed to construct cryptographic functionalities of interest. While the existence of one-way functions is generally considered to be the minimal assumption that is useful for cryptography, a recent line of work, initiated by Kretschmer [Kre21], has shown that this may not be the case *in a quantum world*. Since Kretschmer's result, the topic has seen a surge of interest, with many recent works constructing cryptography from assumptions that are potentially weaker than one-way functions [AQY22, MY22b, MY22a, AGQY23, ALY23, KT23].

These constructions are based on novel primitives whose security is formulated in terms of the hardness of an *inherently quantum* problem. The first example of such a primitive, a pseudorandom state (PRS), was proposed by Ji, Liu, and Song [JLS18a]. A PRS can be thought of as the quantum analogue of a pseudorandom generator (PRG), and it refers to an ensemble of

© International Association for Cryptologic Research 2025
E. Boyle and M. Mahmoody (Eds.): TCC 2024, LNCS 15366, pp. 289–317, 2025.
https://doi.org/10.1007/978-3-031-78020-2_10

efficiently preparable quantum states that are *computationally* indistinguishable from Haar random. Other more recent examples are EFI pairs [BCQ22], and one-way state generators [MY22a]. These inherently quantum primitives are sometimes collectively referred to as "*MicroCrypt*"[1]. They are especially interesting for two reasons. First, they are qualitatively weaker than one-way functions: while one-way functions imply all of them, Kretschmer showed that they are provably not sufficient to construct one-way functions when used in a black-box way [Kre21]. Second, these primitives are sufficient to construct many cryptographic primitives of interest, namely commitments, zero-knowledge proofs, symmetric-key encryption, and even oblivious transfer and multiparty computation [AQY22,MY22b,MY22a,AGQY23,ALY23,KT23]. These results are counterintuitive because in a classical world all of these primitives require, at the very least, one-way functions! The new constructions circumvent this requirement because some component of the construction involves quantum states, e.g. the communication in the case of commitments or oblivious transfer, and the ciphertext in the case of symmetric-key encryption. In light of this, what more can we hope to construct in MicroCrypt, and what is beyond reach? In the rest of this introduction, we focus on PRS, as they imply all other known primitives in MicroCrypt.

One of the most important primitives whose relationship to MicroCrypt is still elusive are digital signatures. While classical digital signatures imply one-way functions, and are thus beyond the reach of MicroCrypt, recent works have explored the possibility of constructing digital signature schemes where the public key is a quantum state. In particular, Morimae and Yamakawa [MY22a] construct a *one-time* secure digital signature scheme with quantum public keys from pseudo random states in a black-box way. One-time security means that the adversary is only allowed to make *one* query to the signing oracle, before attempting to produce a valid signature of a different message. Morimae and Yamakawa's construction is a quantum public-key version of the classical Lamport signature scheme. However, it is unclear how to extend their construction to satisfy the standard notion of *multi-time* security (where the adversary is allowed an arbitrary polynomial number of queries to the signing oracle). One of the main obstacles is that the public keys are quantum states (the classical approach involves signing public keys, and signing quantum states is known to be impossible in general [AGM21]). Morimae and Yamakawa thus leave open the question of whether there exists a black-box construction of a quantum digital signature scheme with multi-time security from a PRS. This is the central question that we focus on in this work.

*Does there exist a black-box construction of multi-time secure quantum digital signatures from PRS?*

[1] *MicroCrypt* is an addition to Impagliazzo's five worlds [Imp95]. This is a world in which one-way functions do not exist, but inherently quantum primitives, like PRS, exist, and thus non-trivial cryptography is possible. As far as we know, the term was coined by Tomoyuki Morimae.

On the one hand, the barrier in extending Morimae and Yamakawa's scheme seems fundamental, and coming up with entirely new schemes is always a difficult endeavour. On the other hand, Kretschmer's original separation of PRS from one-way functions [Kre21] is the only black-box separation involving MicroCrypt that we are aware of, and thus known techniques are fairly limited.

### 1.1 Our Results

We provide a partial answer to the above question on the negative side. Namely, we show the following.

**Theorem 1 (Informal).** *There is a quantum oracle $O$ relative to which:*

1. *PRS with linear, or greater, output length exist.*
2. *No digital signature scheme with a* quantum *public key, and classical secret key and signatures (and message length at least $2 \log \lambda$) exists.*

The oracle is similar to the one that Kretschmer used to separate PRS and one-way functions [Kre21]. Our analysis builds on a key technique introduced in the same work, but needs to circumvent several additional roadblocks that we discuss in the technical overview (Sect. 2). We believe that these roadblocks may not be unique to this setting, and that our proof ideas (described in Sect. 2.2) might find application elsewhere. As a corollary, our result implies the following.

**Corollary 1 (Informal).** *There does not exist a fully black-box construction of a digital signature scheme with a quantum public key (and classical secret key and signatures) from a PRS with linear, or greater, output length.*[2]

To the best of our knowledge, this is the first non-trivial black-box separation involving MicroCrypt beyond Kretschmer's original separation. We point out that our separation is in the same setting as Morimae and Yamakawa's positive result (PRS of linear output length, and classical secret key and signatures). Thus it directly answers, in the negative, their question of whether their approach could be extended to yield multi-time security.

How should one view a black-box separation? The vast majority of known cryptographic constructions are *fully black-box*. This means that:

(i) The construction of primitive $\mathcal{Q}$ from primitive $\mathcal{P}$ does not make use of the "code" of $\mathcal{P}$, but only uses $\mathcal{P}$ as a black-box.
(ii) There exists a black-box security reduction, i.e. given an adversary $A$ that breaks $\mathcal{Q}$, there exists an adversary $A'$ that breaks $\mathcal{P}$ by using $A$ as a black-box.

[2] In fact, our result is slightly stronger than this. We rule out a fully black-box construction where secret keys can also be quantum states, as long as the secret key generation algorithm does not make queries to the PRS generation algorithm. See Remark 2 for the details.

Showing that such a fully black-box reduction does not exist rules out the most natural class of constructions, and establishes that any attempt to construct $\mathcal{Q}$ from $\mathcal{P}$ *must* violate either (i) or (ii). From the point of view of "cryptographic complexity", the black-box separation establishes that primitive $\mathcal{Q}$ is, at the very least, not qualitatively weaker than primitive $\mathcal{P}$, and possibly stronger.

The approach of exhibiting an oracle separation as a means to prove the impossibility of a black-box construction was introduced in a seminal work of Impagliazzo and Rudich [IR89]. In Sect. 4, we include a formal discussion of the relationship between oracle separations and black-box constructions *in a quantum world*, i.e. a world in which oracles are unitary and constructions are quantum algorithms. Such a discussion, to the best of our knowledge, was missing despite recent works on the topic. The summary is that, when talking about a black-box construction of primitive $\mathcal{Q}$ from primitive $\mathcal{P}$ in a quantum world, one needs to be careful about defining the kind of "access to $\mathcal{P}$" that is available to the construction. One natural definition is that "access to $\mathcal{P}$" means having access to a "unitary implementation" of $\mathcal{P}$. However, a natural question is: is access to the "inverse" also available? Perhaps unsurprisingly, if one wants to rule out a black-box construction *with access to the inverse*, then one should exhibit a separation relative to a pair of oracles $(O, O^{-1})$.

We point out that our result (Corollary 1) only rules out a fully black-box construction *without* access to the inverse. This limitation is shared by Kretschmer's separation of PRS and one-way functions (this is not by coincidence, but it is rather because our result leverages some of techniques used there).

## 1.2 Open Questions

Our result comes short of a full answer to the general question of the relationship between digital signatures and PRS in two respects:

1. First, our result only rules out a black-box construction of digital signatures from PRS with long output (linear or greater). However, can digital signatures be constructed from PRS with short output (sublinear)? We should point out that, unlike for classical PRGs, for which the output can be stretched, and also (trivially) shrunk, the relationship between PRS with short and long output is still very much unclear. We do not know whether one implies the other (and if so in what direction), or whether they are incomparable. Recent work [ALY23] shows that PRS with short output can be used to construct primitives (e.g. QPRGs) that we do not know how to construct from PRS with long output. So it seems at least in principle possible that there could be a black-box construction of digital signatures from PRS with short output.
2. Second, our result only applies to digital signatures with a quantum public key, but with classical secret key and signatures. If we allow the latter to be quantum as well, then is there a construction? On the one hand, it is unclear to us how this relaxation may be helpful in realizing a construction. On the other, our current techniques to prove a separation run into a barrier in this setting, which we discuss in the technical overview, and in further detail in Sect. 6.4.

## 2 Technical Overview

We give a detailed informal overview of our result that PRS with linear, or greater, output length, cannot be used to construct, in a black-box way, a digital signature scheme with quantum public key, and classical secret key and signatures. For convenience, from here on, we simply refer to the latter type of scheme as a QDS.

To show this result, it is sufficient to construct an oracle (classical or quantum) relative to which PRS exist but QDS do not (see Sect. 4 for more details about why this is sufficient). Before explaining our approach, we give a (slightly informal) definition of a QDS scheme and its security.

A QDS scheme is specified by a tuple of algorithms ($SKGen, PKGen, Sign, Verify$) satisfying the following:

- $SKGen(1^\lambda) \rightarrow sk$: is a QPT algorithm that takes as input $1^\lambda$, and outputs a classical secret key $sk$.
- $PKGen(sk) \rightarrow |pk\rangle$: is a deterministic algorithm that takes as input a secret key $sk$, and outputs the quantum state $|pk\rangle$.[3] We additionally require $|pk\rangle$ to be fixed given $sk$, i.e. the algorithm $SKGen$ is deterministic (it consists of a fixed unitary quantum circuit acting on the input $sk$, and some auxiliary registers).[4]
- $Sign(sk,m) \rightarrow \sigma$: is a QPT algorithm that takes as input a secret key $sk$ and a classical message $m$, and outputs a classical signature $\sigma$.
- $Verify(|pk\rangle, m, \sigma) \rightarrow \mathsf{accept/reject}$: is a QPT algorithm that takes as input a public key $|pk\rangle$, a message $m$, and a candidate signature $\sigma$, and outputs $\mathsf{accept}$ or $\mathsf{reject}$.

We take messages and the secret key to be of length $\lambda$ for simplicity. In this work, we focus on standard *multi-time* security, defined in terms of the following "unforgeability" game between an adversary $\mathcal{A}$ and a challenger $\mathcal{C}$.

(i) $\mathcal{C}$ samples $sk \leftarrow SKGen(1^\lambda)$, and $\mathcal{A}$ receives polynomially many copies of $|pk\rangle = PKGen(sk)$.
(ii) $\mathcal{A}$ obtains from $\mathcal{C}$ the signatures of polynomially many messages of its choice.
(iii) $\mathcal{A}$ sends a pair $(m, \sigma)$ to $\mathcal{C}$, where $m$ is not among the previously signed messages.
(iv) $\mathcal{A}$ wins the game if $Verify(pk, m, \sigma)$ accepts.

The QDS scheme is *multi-time* secure if any quantum polynomial time adversary has negligible winning probability in this game.

[3] To clarify, $|pk\rangle$ is allowed to be an arbitrary pure state (not necessarily a standard basis state).

[4] We include this requirement so that the notion of "quantum" public key is a little more faithful to the spirit of a classical public key. This requirement ensures that the party in possession of the secret key can generate multiple copies of the corresponding public key. Note that for a completely classical digital signature scheme this requirement is without loss of generality, since any randomness used in the generation procedure can be included in the secret key.

*The Separating Oracle.* We are now ready to describe the oracle relative to which PRS exist, but QDS schemes of type (1) does not. As mentioned earlier, the oracle is similar to the one used by Kretschmer in [Kre21]. The oracle $O$ consists of a pair of oracles $(\mathcal{U}, \mathcal{Q})$, where $\mathcal{Q}$ is a classical oracle solving a fixed $\mathsf{EXP}$-complete problem, and $\mathcal{U}$ is a collection of Haar-random unitaries $\{\mathcal{U}_\ell\}_{\ell \in \mathbb{N}}$, where each $\mathcal{U}_\ell$ is an indexed list of $2^\ell$ Haar-random unitaries acting on $\ell$ qubits.

*Why is* $(\mathcal{U}, \mathcal{Q})$ *a natural choice of oracle?* First of all, relative to $\mathcal{U}$, there is a trivial construction of a PRS: on input a seed $k$, apply the unitary from $\mathcal{U}_{|k|}$ with index $k$ to the state $|0\rangle^{\otimes |k|}$. Importantly, this construction is still secure even in the presence of the second oracle $\mathcal{Q}$, provided $\mathcal{Q}$ is fixed independently of the sampled $\mathcal{U}$ (this is shown in Lemma 31 of [Kre21]).

The hard part of our result is constructing, for any QDS scheme relative to these oracles, an adversary $\mathcal{A}^{\mathcal{U},\mathcal{Q}}$ that breaks it. In this technical overview, as a warm-up, we start by considering the case of a QDS scheme with a *classical* public key (and classical signatures). In this case, a simple approach suffices: $\mathcal{A}$ uses $\mathcal{Q}$ to perform a "brute-force" search for a signature that passes the verification procedure (this "brute-force" search can be performed because $\mathcal{Q}$ solves an $\mathsf{EXP}$-complete problem). Since $\mathcal{Q}$ is a function that is fixed *before* $\mathcal{U}$ is sampled (informally speaking, $\mathcal{Q}$ does not have access to $\mathcal{U}$), the brute-force search approach has to be combined with a technique introduced by Kretschmer [Kre21] to simulate the queries that the verification procedure makes to $\mathcal{U}$. We then move on to the case of a QDS scheme with a *quantum* public key (and classical signatures). Here, there are several challenges that prevent us from using the same "brute-force" search approach. We describe these challenges, and an approach that overcomes them.

## 2.1 Warm Up: Oracle Separation Between PRS and QDS with *Classical* Public Key

Consider a QDS scheme $(SKGen^{\mathcal{U},\mathcal{Q}}, PKGen^{\mathcal{U},\mathcal{Q}}, Sign^{\mathcal{U},\mathcal{Q}}, Verify^{\mathcal{U},\mathcal{Q}})$ with a classical public key. Unless specified otherwise, we will always consider schemes with classical secret keys and signatures. Our task is to construct an adversary $\mathcal{A}^{\mathcal{U},\mathcal{Q}}$ that wins the multi-time unforgeability game with non-negligible probability.

*How does* $\mathcal{A}$ *Use* $\mathcal{Q}$*?* For concreteness, suppose signatures for some message $m$ are $\lambda$-bit strings. Let $pk$ be the public key. The simplest approach to finding a valid signature for message $m$, is to do a "brute-force search" over the space of $\lambda$-bit strings for a string that is accepted by *Verify*$(pk, \cdot)$. One needs to be a bit more careful though since *Verify* makes queries to both $\mathcal{U}$ and $\mathcal{Q}$. Ignoring $\mathcal{U}$ for a moment, since $\mathcal{Q}$ solves a fixed $\mathsf{EXP}$-complete problem, the brute-force search naively corresponds to an $\mathsf{EXP}^{\mathsf{EXP}}$ problem (which may thus be outside of $\mathsf{EXP}$). However, since *Verify* is a *QPT* algorithm (and thus can only make polynomial-size queries to $\mathcal{Q}$), this brute-force search actually corresponds to an EXP problem (and can thus be reduced to an instance of the EXP-complete problem solved by $\mathcal{Q}$)[5].

The more delicate issue is that, while the algorithm *Verify* has a succinct description, unfortunately the oracle $\mathcal{U}$ does not. Moreover, the oracle $\mathcal{U}$ is sampled *after* $\mathcal{Q}$ is fixed, and so even an inefficient description of $\mathcal{U}$ cannot be "hardcoded" into $\mathcal{Q}$. As anticipated, this issue can be resolved using a technique by Kretschmer [Kre21], which we describe below.[6] This approach still runs into several fundamental issues when the public key is *quantum*. We describe these issues, and how to overcome them, in Sect. 2.2, but for now we focus on the case of a *classical* public key.

*Remark 1.* For simplicity, in the rest of this section, we will often describe $\mathcal{Q}$ as an exponential-time algorithm. Formally, however, $\mathcal{Q}$ is a fixed function computing a fixed EXP-complete problem. So, when describing $\mathcal{Q}$ as an exponential-time algorithm what we formally mean is: we cast the underlying problem that the algorithm is solving as an EXP problem, and then reduce it to the particular EXP-complete problem computed by $\mathcal{Q}$.

*Simulating Queries to $\mathcal{U}$.* The technique relies crucially on the strong concentration property of the Haar measure. The corollary of this property that is most relevant here is the following (stated informally). Let $C$ be a quantum circuit that makes $poly(\lambda)$ queries to a Haar random unitary acting on $\lambda$ qubits. Then, with overwhelming probability over (independently) sampling two such Haar random unitaries $U$ and $U'$, the output distributions of the circuits $C^U$ and $C^{U'}$ (on, say, the $|0\rangle$ input) are within a small constant TV distance of each other. In fact, the concentration is strong enough to support a union bound over *all* standard basis inputs. So, with overwhelming probability over $U$ and $U'$, the output distributions of $C^U$ and $C^{U'}$ on *all* standard basis inputs, are within a small constant TV distance of each other.

In our setting, $C$ is the circuit $Verify(pk, m, \cdot)$ for some message $m$, which makes $T$ queries to the family $\mathcal{U}$. Thanks to the above concentration property, $\mathcal{Q}$ can now perform the brute-force search "without access to $\mathcal{U}$" by simply replacing the oracle calls of $Verify(pk, m, \cdot)$ with unitary $T$-designs. This will perfectly simulate $T$ queries to freshly sampled Haar random unitaries.

There is still one remaining subtlety. Recall that $\mathcal{U}$ is a *family* of unitaries $\{\mathcal{U}_\ell\}_{\ell \in \mathbb{N}}$, where each $\mathcal{U}_\ell$ is an indexed list of $2^\ell$ Haar-random unitaries acting on $\ell$ qubits. However, the concentration property only holds for Haar random unitaries acting on a *large enough number of qubits.* This issue can be circumvented because for smaller dimensions, up to $O(\log(\lambda))$ qubits, the unitaries can be "learnt" efficiently (in $O(poly(\lambda))$ queries) by performing process tomography.

[5] For more details, see the second footnote within Algorithm 2.

[6] As an alternative to this classical oracle $\mathcal{Q}$, one might also consider a quantum oracle $\mathcal{Q}$ that makes queries to $\mathcal{U}$. Given exponentially many queries to $\mathcal{U}$, such an oracle would allow the adversary to easily break the QDS scheme. However, such an oracle would also break the security of the PRS built using $\mathcal{U}$.

We can package Kretschmer's technique into one procedure, which we will denote as Sim-Haar. The latter procedure has two parameters $\eta$ and $\delta$. It has oracle access to $\mathcal{U}$, and takes as input the description of a quantum circuit $C$ (with a one-bit output) that makes queries to $\mathcal{U}$, and it outputs another quantum circuit $C'$ (with a one-bit output) that *does not* make queries to $\mathcal{U}$. The guarantee of Sim-Haar is that, with probability $1 - e^{-\eta}$ over $\mathcal{U}$ and the randomness of the procedure, it holds that, for a given $x$,

$$\left| \Pr[C^{\mathcal{U}}(x) = 1] - \Pr[C'(|x\rangle) = 1] \right| \leq \delta.$$

The runtime of Sim-Haar (which includes queries to $\mathcal{U}$) is $poly(|C|, T, \eta, 1/\delta)$, where $|C|$ is the size of $C$.

To put things together, the adversary $\mathcal{A}^{\mathcal{U},\mathcal{Q}}$ for the QDS scheme picks an arbitrary message $m$. It first obtains a circuit $Verify'$ by running $\mathsf{Sim\text{-}Haar}^{\mathcal{U}}$ on input $Verify(pk, m, \cdot)$ (with a sufficiently large $\eta$, and a small constant $\delta$). Then, it invokes $\mathcal{Q}$ to search for a signature $\sigma$ such that $Verify'(\sigma)$ accepts with probability greater than a sufficiently large constant, and finally it outputs $(m, \sigma)$.

## 2.2 Oracle Separation Between PRS and QDS with *Quantum* Public Key

There are several issues when trying to extend the previous 'brute force' attack to a QDS scheme with a quantum public key.

*The First Issue with a Quantum pk.* Let the public key be a quantum state $|pk\rangle$. The first issue is syntactical: since $\mathcal{Q}$ is a classical oracle, how does $\mathcal{A}$ describe the circuit $Verify(|pk\rangle, \cdot)$ to $\mathcal{Q}$? The natural way to fix this is to consider a *quantum* oracle $\mathcal{Q}$ (i.e. one that can take quantum inputs) that is still independent of $\mathcal{U}$. However, even then, $\mathcal{Q}$ would in general need exponentially many copies of $|pk\rangle$ in order to run a brute force search algorithm. This is because the public key state is potentially disturbed every time the circuit $Verify$ is run.[7]

*An Alternative Approach:* Note that we should have expected the previous approach to fail. This is because it did not make use of the adversary's ability to make queries to the signing oracle. Since there exists a black-box construction of a one-query secure QDS scheme from a PRS [MY22b], any adversary breaking the QDS scheme must necessarily make use of signing queries (and in fact polynomially many of them). We consider the following alternative approach.

– $\mathcal{A}$ makes polynomially many queries to the signing oracle, obtaining message-signature pairs $(m_i, \sigma_i)$.

[7] Note that attempts to "uncompute" the circuit and recover $|pk\rangle$ fail in general for several reasons. One of them is that correctness and soundness are not perfect, so negligible errors can add up to a noticeable quantity when performing a brute force search.

- $\mathcal{A}$ uses $\mathcal{Q}$ to find a set of secret keys that are "consistent" with all of the $(m_i, \sigma_i)$. We will refer to this set as $Consistent$. That is, $sk \in Consistent$ if, for all $(m_i, \sigma_i)$, $\Pr[Verify^{\mathcal{U},\mathcal{Q}}(PKGen^{\mathcal{U},\mathcal{Q}}(sk), m_i, \sigma_i) = \mathsf{accept}]$ is greater than some threshold, for example $\frac{9}{10}$. Once again, just like in Subsect. 2.1, $\mathcal{Q}$ is independent of $\mathcal{U}$, and so we need to first obtained a simulated version of the circuit
$Verify^{\mathcal{U},\mathcal{Q}}(PKGen^{\mathcal{U},\mathcal{Q}}(\cdot), \cdot, \cdot)$ using the Sim-Haar procedure described in Subsect. 2.1. Recall that the guarantee of Sim-Haar is that, with high probability over $\mathcal{U}$, on *all* classical inputs $sk, m, \sigma$ the simulated circuit's acceptance probability is close to the original.
For the rest of the section, whenever we refer to a circuit that originally made queries to $\mathcal{U}$, we will simply assume that we are utilizing a simulated version of that circuit obtained using Sim-Haar, and we will drop $\mathcal{U}$ from the notation. For ease of notation, since $\mathcal{Q}$ is fixed, we will also drop $\mathcal{Q}$ from the notation.
- $\mathcal{A}$ signs a fresh message using a uniformly random key from $Consistent$.

*The Main Challenge:* Observe that the set of consistent secret keys are, by definition, those $sk$ such that, for all $(m_i, \sigma_i)$, $\Pr[Verify(PKGen(sk), \cdot) = \mathsf{accept}] > 9/10$. By a suitable concentration bound (taking the number of queried message-signature pairs to be a large enough polynomial), we have that, with overwhelming probability over the $(m_i, \sigma_i)$, such $sk$ are also such that, for most $m$,

$$\Pr[Verify(PKGen(sk), m, \sigma) = \mathsf{accept}] \geq \Omega(1), \text{ where } \sigma \leftarrow Sign(sk^*, m). \quad (1)$$

In other words, the secret keys in $Consistent$ accept (with high probability) not only the queried message, signature pairs, but also *fresh* signatures signed using the true secret key $sk^*$.

Unfortunately, this is not quite the guarantee we are looking for! What we want is the reverse (i.e. we would like the roles of $sk$ and $sk^*$ to be swapped): an $sk$ such that, for most $m$,

$$\Pr[Verify(PKGen(sk^*), m, \sigma) = \mathsf{accept}] \geq \Omega(1), \text{ where } \sigma \leftarrow Sign(sk, m). \quad (2)$$

To make the issue concrete, consider an $sk$ for which $Verify(PKGen(sk), \cdot, \cdot)$ simply accepts everything with probability 1. Such an $sk$ would certainly be in the consistent set of secret keys, but it may not be capable of generating signatures that are accepted by the true secret key $sk^*$ (and the existence of such an $sk$ does not appear to contradict any property of a QDS scheme).

To summarize, so far we have identified a consistent set of secret keys that clearly contains $sk^*$, but is potentially exponentially large, and may contain secret keys that do not satisfy Eq. 2, i.e. they are not "good signers". So, how do we proceed?

*Finding a Sequence of Smaller and Smaller Subsets Containing $sk^*$.* We describe an iterative procedure that identifies smaller and smaller subsets of $Consistent$, which are *guaranteed to still contain* $sk^*$ (or else there is an easy way to find a signature accepted by $PKGen(sk^*)$). Eventually, these subsets only contain the true secret key $sk^*$.

Before describing the iterative procedure, we define the following terms:

- *"Good signer" secret keys*: Informally, a secret key $sk \in Consistent$ is a "good signer" if $\frac{9}{10}$ of the other secret keys in $Consistent$ accept signatures generated using $sk$ on a constant fraction of the message space, with constant probability. More precisely, $sk \in Consistent$ is a good signer iff the following is true: $|accept_{sk}| \geq \frac{9}{10} \cdot |Consistent|$, where $accept_{sk}$ is the set of all secret keys $sk' \neq sk$ such that at least $\frac{1}{8}$ fraction of the message space satisfies the following: $\Pr[Verify(PKGen(sk'), m, Sign(sk, m)] > \frac{1}{8}$. The exact constants in this and the next definition are not important.
- *"Stingy" secret keys*: Informally, a secret key $sk \in Consistent$ is "stingy" if it does not accept most signatures generated by most secret keys in $Consistent$. More precisely, $sk \in Consistent$ is stingy iff the following is true: $|friends_{sk}| \leq \frac{1}{2} \cdot |Consistent|$, where $friends_{sk}$ is the set of all secret keys $sk' \neq sk$ such that $sk \in accept_{sk'}$.

The two sets above can be defined analogously with respect to a set of secret keys $S$, which is not necessarily the set $Consistent$. In this case, we denote them as $GoodSigner_S$ and $Stingy_S$.
We are now ready to define the following sequence of nested subsets of $Consistent$. Let $S_0 = Consistent$. For $j \in [T]$, where $T$ is a large enough polynomial in $\lambda$, define

$$S_j = GoodSigner_{S_{j-1}} \cap Stingy_{S_{j-1}}.$$

**Observation 1:** The sets $S_j$ shrink in size very quickly. More precisely, one can show that $|S_j| \leq \frac{9}{10}|S_{j-1}|$ (or $|S_j| = 1$). This is a somewhat straightforward combinatorial argument based on the fact that the $GoodSigner$ and $Stingy$ sets impose conflicting restrictions on their elements.

**Observation 2:** The second crucial observation is that $sk^* \in GoodSigner_{S_j}$ for all $j$. This is for the following reason. By definition, the $S_j$ are all subsets of $Consistent$. Moreover, provided the set of queried message-signature pairs $(m_i, \sigma_i)$ is a sufficiently large polynomial, then, with overwhelming probability over the queried pairs (by a concentration bound), *all* secret keys $sk \in Consistent$ accept most signatures generated using the true secret key $sk^*$ (we argued this earlier too in Eq. (1)). In other words, with overwhelming probability, *all* $sk \in Consistent$ belong to $accept_{sk^*}$. We emphasize that this part of the proof crucially relies on the fact that the QDS adversary can make *polynomially* many signing queries (this is why our black-box separation does not contradict the one-query secure black-box construction in [MY22b]).

Observation 2 implies that, for all $j$, $sk^* \in S_j$ if and only if $sk^* \in Stingy_{S_{j-1}}$. There are two cases:

1. $sk^* \in Stingy_{S_j}$ for all $j$. Then it must be that $S_T = \{sk^*\}$ (because of the shrinking property of the $S_j$).

2. $sk^* \notin Stingy_{S_j}$ for some $j$. Then, note that, by definition of $Stingy_{S_j}$, this implies that $PKGen(sk^*)$ accepts signatures generated by a constant fraction of $sk$ in $S_j$, with a constant probability (for a constant fraction of messages). More formally, $sk^* \in accept_{sk}$ for a constant fraction of $sk$.

Returning to our adversary for the QDS scheme, $\mathcal{A}$ asks $\mathcal{Q}$ to do the following: compute the sets $S_j$ as defined above, and output *one* uniformly random element from each $S_j$ (note that this is an exponential-time computation, and so it can be "run" by $\mathcal{Q}$). $\mathcal{A}$ picks a uniformly random secret key from this (polynomial-size) set, and uses it to sign a uniformly random message. By the properties we proved above, the list of secret keys output by $\mathcal{Q}$ contains $sk^*$ (in case 1), or contains (in case 2), with constant probability, an $sk$ such that $sk^* \in accept_{sk}$. Therefore, $\mathcal{A}$ wins the unforgeability game with inverse-polynomial probability.

## 3 Preliminaries

### 3.1 Basic Notation

Throughout the paper, $[n]$ denotes the set of integers $\{1, 2, \ldots, n\}$. If $X$ is a probability distribution, we use $x \sim X$ to denote that $x$ is sampled according to $X$. A function $f$ is *negligible* if for every constant $c > 0$, $f(n) \leq \frac{1}{n^c}$ for all sufficiently large $n$. We use the abbreviation QPT for a quantum polynomial time algorithm. We use the notation $A^{(\cdot)}$ to refer to an algorithm (classical or quantum) that makes queries to an oracle.

### 3.2 Quantum Information

We use $TD(\rho, \sigma)$ to denote the trace distance between density matrices $\rho$ and $\sigma$. For a quantum channel $\mathcal{A}$, we let $\|A\|_\diamond$ denote its *diamond norm*. The diamond norm and trace distance satisfy the following relation:

*Fact 1.* [NC11] Let $\mathcal{A}$ and $\mathcal{B}$ be quantum channels and $\rho$ be a density matrix. Then,

$$TD(\mathcal{A}(\rho), \mathcal{B}(\rho)) \leq ||\mathcal{A} - \mathcal{B}||_\diamond$$

### 3.3 Haar Measure and Its Concentration

We use $\mathbb{U}(N)$ to denote the group of $N \times N$ unitary matrices, and $\mu_N$ to denote the Haar measure on $\mathbb{U}(N)$. Given a metric space $(\mathcal{M}, d)$ where $d$ denotes the metric on the set $\mathcal{M}$, a function $f : \mathcal{M} \to \mathbb{R}$ is $\mathcal{L}$-lipschitz if for all $x, y \in \mathcal{M}, |f(x) - f(y)| \leq \mathcal{L} \cdot d(x, y)$. The following inequality involving Lipschitz continuous functions captures the strong concentration of Haar measure.

**Theorem 2** **([Mec19]).** *Given $N_1, N_2, \ldots, N_k \in \mathbb{N}$, let $X = \mathbb{U}(N_1) \bigoplus \cdots \bigoplus \mathbb{U}(N_k)$ be the space of block diagonal unitary matrices with blocks of size $N_1, N_2, \ldots, N_k$. Let $\nu = \nu_1 \times \cdots \times \nu_k$ be the product of Haar measures on $X$. Suppose that $f : X \to \mathbb{R}$ is $L$-Lipshitz with respect to the Frobenius norm. Then for every $t > 0$,*

$$\Pr_{U \leftarrow \nu}[f(U) \geq \mathbb{E}_{V \leftarrow \nu}[f(V)] + t] \leq \exp(-\frac{(N-2)t^2}{24L^2}),$$

*where $N = \min N_1, \ldots, N_k$.*

**Lemma 1** **([Kre21]).** *Let $A^{(\cdot)}$ be a quantum algorithm that makes $T$ queries to an oracle, and let $|\psi\rangle$ be any input to $A^{(\cdot)}$. Define $f(U) = \Pr[A^U(|\psi\rangle) = 1]$. Then, $f$ is $2T$-Lipshitz with respect to the Frobenius norm.*

## 3.4 Quantum Pseudorandomness

The notion of pseudorandom quantum states was introduced in [JLS18a]. The following is a formal definition. We use $\sigma_d$ to denote the Haar measure on $d$-dimensional pure quantum states.

**Definition 1 (Pseudorandom Quantum State (PRS)).** *A Pseudorandom Quantum State (PRS) is a pair of QPT algorithms* $(\mathsf{GenKey}, \mathsf{GenState})$ *such that the following holds. There exists $n : \mathbb{N} \to \mathbb{N}$, and a family $\{\mathcal{K}_\lambda\}_{\lambda \in \mathbb{N}}$ of subsets of $\{0,1\}^*$ such that:*

- $\mathsf{GenKey}(1^\lambda) \to k$*: Takes as input a security parameter $\lambda$, and outputs a key $k \in \mathcal{K}_\lambda$.*
- $\mathsf{GenState}(k) \to |\mathrm{PRS}(k)\rangle$*: Takes as input a key $k \in \mathcal{K}_\lambda$, for some $\lambda$, and outputs an $n(\lambda)$-qubit state. We additionally require that the state on input $k$ be unique, and we denote this as $|\mathrm{PRS}(k)\rangle$.*

*Moreover, the following holds. For any (non-uniform) QPT quantum algorithm $A$, and any $m = poly$, there exists a negligible function* negl *such that, for all $\lambda \in \mathbb{N}$,*

$$\left| \Pr_{k \leftarrow \mathsf{GenKey}(1^\lambda)}[A\big(|\mathrm{PRS}(k)\rangle^{\otimes m(\lambda)}\big) = 1] - \Pr_{|\psi\rangle \leftarrow \sigma_{2^{n(\lambda)}}}[A\big(|\psi\rangle^{\otimes m(\lambda)}\big) = 1] \right| \leq \mathrm{negl}(n).$$

**Definition 2 (Pseudorandom unitary transformations (PRU) ([JLS18a])).** *Let $n : \mathbb{N} \to \mathbb{N}$. Let $\{\mathcal{U}_\lambda\}_{\lambda \in \mathbb{N}}$ be a family of unitaries where $\mathcal{U}_\lambda$ is a family of $n(\lambda)$-qubit unitaries $\{U_k\}_{k \in \{0,1\}^\lambda}$. We say that $\{\mathcal{U}_\lambda\}_{\lambda \in \mathbb{N}}$ is pseudorandom if the following conditions hold:*

1. *(Efficient computation) There is a QPT algorithm $G$ that implements $U_k$ on input $k$, meaning that for any $n$-qubit input $|\psi\rangle$, $G(k, |\psi\rangle) = U_k|\psi\rangle$.*
2. *(Computationally indistinguishable) For any QPT algorithm $A^{(\cdot)}$, there exists a negligible function* negl *such that, for all $\lambda$,*

$$\left| \Pr_{k \leftarrow \{0,1\}^\lambda}[A^{U_k}(1^\lambda) = 1] - \Pr_{U \leftarrow \mu_{2^n}}[A^U(1^\lambda) = 1] \right| \leq \mathrm{negl}(\lambda).$$

## 4 On Quantum Oracle Separations and Black-Box Constructions

In this section, we clarify what we mean by a "black-box construction" of primitive $\mathcal{Q}$ from primitive $\mathcal{P}$ when the primitives involve *quantum* algorithms (and possibly quantum state outputs). We also clarify the relationship between a *quantum* oracle separation of $\mathcal{P}$ and $\mathcal{Q}$ and the (im)possibility of a black-box construction of one from the other. To the best of our knowledge, while black-box separations in the quantum setting have been the topic of several recent works, a somewhat formal treatment of the terminology and basic framework is missing. This section also appears verbatim in the concurrent work [CCS24].

In the quantum setting, it is not immediately obvious what the correct notion of "black-box access" is. There are a few reasonable notions of what it means for a construction to have "black-box access" to another primitive. We focus on three variants: *unitary* access, *isometry* access, and access to *both the unitary and its inverse.*

The summary is that, similarly to the classical setting, a *quantum* oracle separation of primitives $\mathcal{P}$ and $\mathcal{Q}$ (i.e. a quantum oracle relative to which $\mathcal{P}$ exists but $\mathcal{Q}$ does not) implies the impossibility of a black-box construction of $\mathcal{Q}$ from $\mathcal{P}$, but with one caveat: the type of oracle separation corresponds directly to the type of black-box construction that is being ruled out. For example, the oracle separation needs to be "closed under giving access to the inverse of the oracle", i.e. the separation needs to hold relative to an oracle *and* its inverse. We start by introducing some terminology.

*Terminology.* A quantum channel is a CPTP (completely-positive-trace-preserving) map. The set of quantum channels captures all admissible "physical" processes in quantum information, and it can be thought of as the quantum analogue of the set of functions $f : \{0,1\}^* \to \{0,1\}^*$.

For the purpose of this section, a quantum channel is specified by a family of unitaries $\{U_n\}_{n\in\mathbb{N}}$ (where $U_n$ acts on an input register of size $n$, and a work register of some size $s(n)$). The quantum channel maps an input (mixed) state $\rho$ on $n$ qubits to the (mixed) state obtained as follows: apply $U_n(\cdot)U_n^\dagger$ to $\rho \otimes (|0\rangle\langle 0|)^{\otimes s(n)}$; measure a subset of the qubits; output a subset of the qubits (measured or unmeasured). We say that the family $\{U_n\}_{n\in\mathbb{N}}$ is a *unitary implementation* of the quantum channel. We say that the quantum channel is QPT if it possesses a unitary implementation $\{U_n\}_{n\in\mathbb{N}}$ that is additionally a uniform family of efficiently computable unitaries. In other words, the quantum channel is implemented by a QPT algorithm.

One can also consider the family of isometries $\{V_n\}_{n\in\mathbb{N}}$ where $V_n$ takes as input $n$ qubits, and acts like $U_n$, but with the work register fixed to $|0\rangle^{s(n)}$, i.e. $V_n : |\psi\rangle \mapsto U_n(|\psi\rangle|0\rangle^{\otimes s(n)})$. We refer to $\{V_n\}_{n\in\mathbb{N}}$ as the *isometry implementation* of the quantum channel.

We will also consider QPT algorithms with access to some oracle $\mathcal{O}$. In this case, the unitary (resp. isometry) implementation $\{U_n\}_{n\in\mathbb{N}}$ should be *efficiently computable given access to* $\mathcal{O}$.

Before diving into formal definitions, a bit informally, a *primitive* $\mathcal{P}$ can be thought of as a set of conditions on tuples of algorithms $(G_1, \dots, G_k)$. For example, for a digital signature scheme, a valid tuple of algorithms is a tuple (*Gen*, *Sign*, *Verify*) that satisfies "correctness" (honestly generated signatures are accepted by the verification procedure with overwhelming probability) and "security" (formalized via an unforgeability game). Equivalently, one can think of the tuple of algorithms $(G_1, \dots, G_k)$ as a *single* algorithm $G$ (with an additional control input).

A thorough treatment of black-box constructions and reductions in the classical setting can be found in [RTV04]. Our definitions are a quantum analog of those in [RTV04]. They follow the latter style whenever possible, and they depart from it whenever necessary.

**Definition 3.** *A* primitive *$\mathcal{P}$ is a pair $\mathcal{P} = (\mathcal{F}_\mathcal{P}, \mathcal{R}_\mathcal{P})$*[8] *where $\mathcal{F}_\mathcal{P}$ is a set of quantum channels, and $\mathcal{R}_\mathcal{P}$ is a relation over pairs $(G, A)$ of quantum channels, where $G \in \mathcal{F}_\mathcal{P}$.*

*A quantum channel $G$ is an* implementation *of $\mathcal{P}$ if $G \in \mathcal{F}_\mathcal{P}$. If $G$ is additionally a QPT channel, then we say that $G$ is an* efficient implementation *of $\mathcal{P}$ (in this case, we refer to $G$ interchangeably as a QPT channel or a QPT algorithm).*

*A quantum channel $A$ (usually referred to as the "adversary") $\mathcal{P}$-breaks $G \in \mathcal{F}_\mathcal{P}$ if $(G, A) \in \mathcal{R}_\mathcal{P}$. We say that $G$ is a* secure implementation *of $\mathcal{P}$ if $G$ is an implementation of $\mathcal{P}$ such that no QPT channel $\mathcal{P}$-breaks it. The primitive $\mathcal{P}$* exists *if there exists an efficient and secure implementation of $\mathcal{P}$.*

*Let $U$ be a unitary (resp. isometry) implementation of $G \in \mathcal{P}$. Then, we say that $U$ is a* unitary (resp. isometry) implementation *of $\mathcal{P}$. For ease of exposition, we also say that quantum channel $A$ $\mathcal{P}$-breaks $U$ to mean that $A$ $\mathcal{P}$-breaks $G$.*

Since we will discuss oracle separations, we give corresponding definitions *relative to an oracle*. Going forward, for ease of exposition, we often identify a quantum channel with the algorithm that implements it.

**Definition 4 (Implementations relative to an oracle).** *Let $\mathcal{O}$ be a unitary (resp. isometry) oracle. An* implementation *of primitive $\mathcal{P}$ relative to $\mathcal{O}$ is an oracle algorithm $G^{(\cdot)}$ such that $G^\mathcal{O} \in \mathcal{P}$*[9]*. We say the implementation is efficient if $G^{(\cdot)}$ is a QPT oracle algorithm.*

*Let $U$ be a unitary (resp. isometry) implementation of $G^\mathcal{O}$. Then, we say that $U$ is a* unitary (resp. isometry) implementation *of $\mathcal{P}$* relative to *$\mathcal{O}$.*

**Definition 5.** *We say that a primitive $\mathcal{P}$ exists relative to an oracle $\mathcal{O}$ if:*

*(i) There exists an efficient implementation $G^{(\cdot)}$ of $\mathcal{P}$ relative to $\mathcal{O}$, i.e. $G^\mathcal{O} \in \mathcal{P}$ (as in Definition 4).*

[8] Here $\mathcal{F}_\mathcal{P}$ should be thought of as capturing the "correctness" property of the primitive, while $\mathcal{R}_\mathcal{P}$ captures "security".

[9] We clarify that here $G^\mathcal{O}$ is only allowed to query the unitary $\mathcal{O}$, not its inverse. However, as will be the case later in the section, $\mathcal{O}$ itself could be of the form $\mathcal{O} = (W, W^{-1})$ for some unitary $W$.

*(ii) The security of $G^{\mathcal{O}}$ holds against all QPT adversaries that have access to $\mathcal{O}$. More precisely, for all QPT $A^{(\cdot)}$, $(G^{\mathcal{O}}, A^{\mathcal{O}}) \notin \mathcal{R}_{\mathcal{P}}$.*

There are various notions of black-box constructions and reductions (see, for example, [RTV04]). Here, we focus on (the quantum analog of) the notion of a *fully black-box construction*. We identify and define three analogs based on the type of black-box access available to the construction and the security reduction.

**Definition 6.** *A QPT algorithm $G^{(\cdot)}$ is a* fully black-box construction of $\mathcal{Q}$ from **unitary access** to $\mathcal{P}$ *if the following two conditions hold:*

1. *(*black-box construction with unitary access*) For every unitary implementation $U$ of $\mathcal{P}$, $G^U$ is an implementation of $\mathcal{Q}$.*
2. *(*black-box security reduction with unitary access*) There is a QPT algorithm $S^{(\cdot)}$ such that, for every unitary implementation $U$ of $\mathcal{P}$, every adversary $A$ that $\mathcal{Q}$-breaks $G^U$, and every unitary implementation $\tilde{A}$ of $A$, it holds that $S^{\tilde{A}}$ $\mathcal{P}$-breaks $U$.*

**Definition 7.** *A QPT algorithm $G^{(\cdot)}$ is a* fully black-box construction of $\mathcal{Q}$ from **isometry access** to $\mathcal{P}$ *if the following two conditions hold:*

1. *(*black-box construction with isometry access*) For every isometry implementation $V$ of $\mathcal{P}$, $G^V$ is an implementation of $\mathcal{Q}$.*
2. *(*black-box security reduction with isometry access*) There is a QPT algorithm $S^{(\cdot)}$ such that, for every isometry implementation $V$ of $\mathcal{P}$, every adversary $A$ that $\mathcal{Q}$-breaks $G^V$, and every isometry implementation $\tilde{A}$ of $A$, it holds that $S^{\tilde{A}}$ $\mathcal{P}$-breaks $V$.*

**Definition 8.** *A QPT algorithm $G^{(\cdot)}$ is a* fully black-box construction of $\mathcal{Q}$ from $\mathcal{P}$ **with access to the inverse** *if the following two conditions hold:*

1. *(*black-box construction with access to the inverse*) For every unitary implementation $U$ of $\mathcal{P}$, $G^{U,U^{-1}}$ is an implementation of $\mathcal{Q}$.*
2. *(*black-box security reduction with access to the inverse*) There is a QPT algorithm $S^{(\cdot)}$ such that, for every unitary implementation $U$ of $\mathcal{P}$, every adversary $A$ that $\mathcal{Q}$-breaks $G^{U,U^{-1}}$, and every unitary implementation $\tilde{A}$ of $A$, it holds that $S^{\tilde{A},\tilde{A}^{-1}}$ $\mathcal{P}$-breaks $U$[10].*

We now clarify the relationship between a *quantum* oracle separation of primitives $\mathcal{P}$ and $\mathcal{Q}$ and the (im)possibility of a black-box construction of one from the other.

The following is a quantum analog of a result by Impagliazzo and Rudich [IR89] (formalized in [RTV04] using the above terminology).

[10] One could define even more variants of "fully black-box constructions" by separating the type of access that $G$ has to the implementation of $\mathcal{P}$ from the type of access that $S$ has to $A$ (currently they are consistent in each of Definitions 6, 7, and 8). Here, we choose to limit ourselves to the these three definitions.

**Theorem 3.** *Suppose there exists a fully black-box construction of primitive $\mathcal{Q}$ from unitary (resp. isometry) access to primitive $\mathcal{P}$. Then, for every unitary (resp. isometry) $\mathcal{O}$, if $\mathcal{P}$ exists relative to $\mathcal{O}$, then $\mathcal{Q}$ also exists relative to $\mathcal{O}$.*

This implies that a unitary (resp. isometry) oracle separation (i.e. the existence of an oracle relative to which $\mathcal{P}$ exists but $\mathcal{Q}$ does not) suffices to rule out a fully black-box construction of $\mathcal{Q}$ from unitary (resp. isometry) access to $\mathcal{P}$.

*Proof (Proof of Theorem 3).* We write the proof for the case of unitary access to $\mathcal{P}$. The proof for the case of isometry access is analogous (replacing unitaries with isometries). Suppose there exists a fully black-box construction of $\mathcal{Q}$ from $\mathcal{P}$. Then, by definition, there exist QPT $G^{(\cdot)}$ and $S^{(\cdot)}$ such that:

1. (*black-box construction*) For every unitary implementation $U$ of $\mathcal{P}$, $G^U$ is an implementation of $\mathcal{Q}$.
2. (*black-box security reduction*) For every implementation $U$ of $\mathcal{P}$, every adversary $A$ that $\mathcal{Q}$-breaks $G^U$, and every unitary implementation $\tilde{A}$ of $A$, it holds that $S^{\tilde{A}}$ $\mathcal{P}$-breaks $U$.

Let $\mathcal{O}$ be a quantum oracle relative to which $\mathcal{P}$ exists. Since, by Definition 5, $\mathcal{P}$ has an *efficient* implementation relative to $\mathcal{O}$, there exists a uniform family of unitaries $U$ that is *efficiently computable* with access to $\mathcal{O}$, such that $U$ is a unitary implementation of $\mathcal{P}$. Moreover, $U$ (or rather the quantum channel that $U$ implements) is a secure implementation of $\mathcal{P}$ relative to $\mathcal{O}$.

We show that the following QPT oracle algorithm $\tilde{G}^{(\cdot)}$ is an efficient implementation of $\mathcal{Q}$ relative to $\mathcal{O}$, i.e. $\tilde{G}^{\mathcal{O}} \in \mathcal{Q}$. $\tilde{G}^{\mathcal{O}}$ runs as follows: implement $G^U$ by running $G$, and simulate each call to $U$ by making queries to $\mathcal{O}$. Note that $\tilde{G}^{(\cdot)}$ is QPT because $U$ is a uniform family of efficiently computable unitaries given access to $\mathcal{O}$. Since $\tilde{G}^{\mathcal{O}}$ is equivalent to $G^U$, and $G^U \in \mathcal{Q}$ (by property 1 above), then $\tilde{G}^{\mathcal{O}} \in \mathcal{Q}$.

We are left with showing that $\tilde{G}^{\mathcal{O}}$ is a secure implementation relative to $\mathcal{O}$, i.e. that there is no QPT adversary $A^{(\cdot)}$ such that $A^{\mathcal{O}}$ $\mathcal{Q}$-breaks $\tilde{G}^{\mathcal{O}}$. Suppose for a contradiction that there was a QPT adversary $A^{(\cdot)}$ such that $\mathcal{A}^{\mathcal{O}}$ $\mathcal{Q}$-breaks $\tilde{G}^{\mathcal{O}}$ (which is equivalent to $G^U$). Then, by property 2, $S^{A^{\mathcal{O}}}$ $\mathcal{P}$-breaks $U$. Note that adversary $S^{A^{\mathcal{O}}}$ can be implemented efficiently with oracle access to $\mathcal{O}$, because both $S^{(\cdot)}$ and $A^{(\cdot)}$ are QPT. Thus, this contradicts the security of $U$ relative to $\mathcal{O}$ (formally, of the quantum channel that $U$ implements).

Similarly, we state a version of Theorem 3 for fully black-box constructions with access to the inverse.

**Theorem 4.** *Suppose there exists a fully black-box construction of primitive $\mathcal{Q}$ from primitive $\mathcal{P}$ with access to the inverse. Then, for every unitary $\mathcal{O}$, if $\mathcal{P}$ exists relative to $(\mathcal{O}, \mathcal{O}^{-1})$, then $\mathcal{Q}$ also exists relative to the oracle $(\mathcal{O}, \mathcal{O}^{-1})$.*

*Proof.* The proof is analogous to the proof of Theorem 3. The only difference is that now $G^{(\cdot)}$ additionally makes queries to the inverse of the unitary implementation $U$ of $\mathcal{P}$. Since $U^{-1}$ can be implemented efficiently given access to $(\mathcal{O}, \mathcal{O}^{-1})$, we can now define an efficient implementation $\tilde{G}^{(\cdot)}$ of $\mathcal{P}$ relative to $(\mathcal{O}, \mathcal{O}^{-1})$. Proving that $\tilde{G}^{\mathcal{O},\mathcal{O}^{-1}}$ is a secure implementation of $\mathcal{P}$ relative to $(\mathcal{O}, \mathcal{O}^{-1})$ also proceeds analogously.

## 5 Quantum Public Key Digital Signatures

In this section, we define QDS schemes with *quantum* public keys (and classical secret key and signatures). Going forward, unless we specify otherwise, we use the term QDS to refer to this type of signature scheme.

**Definition 9.** *A Quantum Digital Signature scheme (QDS) is a tuple of algorithms* $(\mathit{SKGen}, \mathit{PKGen}, \mathit{Sign}, \mathit{Verify})$ *satisfying the following:*

- $\mathit{SKGen}(1^\lambda) \to \mathit{sk}$*: is a QPT algorithm that takes as input* $1^\lambda$*, and outputs a classical secret key* $\mathit{sk}$*. We assume that* $\mathit{sk}$ *has length* $\lambda$*.*
- $\mathit{PKGen}(\mathit{sk}) \to |\mathit{pk}\rangle$*: is a deterministic algorithm that takes as input a secret key* $\mathit{sk}$*, and outputs the quantum state* $|\mathit{pk}\rangle$*.*[11] *We additionally require* $|\mathit{pk}\rangle$ *to be fixed given* $\mathit{sk}$*, i.e. the algorithm SKGen is deterministic (it consists of a fixed unitary quantum circuit acting on the input* $\mathit{sk}$*, and some auxiliary registers).*[12]
- $\mathit{Sign}(\mathit{sk}, m) \to \sigma$*: is a QPT algorithm that takes as input a secret key* $\mathit{sk}$ *and a classical message* $m$ *from some message space (that may depend on the security parameter), and outputs a classical signature* $\sigma$*. For a security parameter* $\lambda$*, we denote by* $\mathcal{M}_\lambda$ *the corresponding message space.*
- $\mathit{Verify}(|\mathit{pk}\rangle, m, \sigma) \to \mathsf{accept}/\mathsf{reject}$*: is a QPT algorithm that takes as input a public key* $|\mathit{pk}\rangle$*, a message* $m$*, and a candidate signature* $\sigma$*, and outputs* accept *or* reject*.*

*We require the following "correctness" property. There exists a negligible function* negl *such that, for all* $\lambda \in \mathbb{N}$*, the following holds except with probability* $\mathrm{negl}(\lambda)$ *over sampling* $\mathit{sk} \leftarrow \mathit{SKGen}(1^\lambda)$*. For all* $m \in \mathcal{M}_\lambda$*,*

$$\Pr[Ver(\mathit{PKGen}(\mathit{sk}), \mathit{Sign}(\mathit{sk}, m)) = 1] \geq 1 - \mathrm{negl}(\lambda).$$

*If there is a function* $\ell : \mathbb{N} \to \mathbb{N}$*, such that, for all* $\lambda$*,* $\mathcal{M}_\lambda$ *is the set of strings of length* $\ell(\lambda)$*, then we say that the scheme is a QDS for messages of length* $\ell(\lambda)$*.*

For simplicity, we will consider QDS where each $\mathcal{M}_\lambda$ is the set of strings of a certain length.

[11] To clarify, $|pk\rangle$ is allowed to be an arbitrary pure state (not necessarily a standard basis state).

[12] We include this requirement so that the notion of "quantum" public key is a little more faithful to the spirit of a classical public key. This requirement ensures that the party in possession of the secret key can generate multiple copies of the corresponding public key. Note that for a completely classical digital signature scheme this requirement is without loss of generality, since any randomness used in the generation procedure can be included in the secret key.

*Multi-time Security.* The notion of security that we focus on in this work is the standard "multi-time" security. The latter allows the adversary to make an arbitrary polynomial number of queries to a signing oracle, before having to produce a valid signature of an "unqueried" message. Formally, multi-time security is defined in terms of the following security game between a challenger $\mathcal{C}$ and an adversary $\mathcal{A}$.

1. $\mathcal{C}$ runs $sk \leftarrow SKGen(1^\lambda)$.
2. $\mathcal{A}$ receives $|pk\rangle^{\otimes t(\lambda)}$ (for some polynomially-bounded function $t$ that depends on $\mathcal{A}$).
3. For each $i \in \{1, \dots t\}$, $\mathcal{A}$ sends a message $m_i$ to $\mathcal{C}$; $\mathcal{C}$ runs $\sigma_i \leftarrow Sign(m_i, sk)$, and sends $\sigma_i$ to $\mathcal{A}$.
4. $\mathcal{A}$ sends $(m, \sigma)$ to $\mathcal{C}$ such that $m \notin \{m_1, \dots m_t\}$.
5. $\mathcal{C}$ outputs 1 iff $Verify(|pk\rangle, m, \sigma)$ accepts.

Let $\mathsf{multi\text{-}time}(\lambda, \mathcal{A})$ be a random variable denoting the output of the game above.

**Definition 10 (multi-time security).** *A QDS scheme satisfies multi-time security if, for all QPT adversaries $\mathcal{A}$, there exists a negligible function* negl *such that*

$$\Pr[\mathit{multi\text{-}time}(\lambda, \mathcal{A}) = 1] = \mathrm{negl}(\lambda). \tag{3}$$

In this work, we consider QDS schemes relative to some oracle $O$. In this setting, the syntax and security definitions are identical to the ones given in this section, except that all of the algorithms, including the adversary, have access to $O$.

## 6 Oracle Separation of Quantum Digital Signatures and PRS

In this section, we describe an oracle relative to which PRS exist, and a QDS scheme that satisfies multi-time security (Definitions 9 and 10) does not. Most of this section is dedicated to proving the latter. For the rest of the section, unless we mention otherwise, any QDS scheme that we consider has a quantum public key and classical signatures. We also restrict our attention to multi-time security.

*The Separating Oracle.* The oracle is similar to the one used by Kretschmer in [Kre21]. The oracle $O$ consists of a pair of oracles $(\mathcal{U}, \mathcal{Q})$, where $\mathcal{Q}$ is a classical oracle solving a fixed $\mathsf{EXP}$-complete problem, and $\mathcal{U}$ is a collection of Haar-random unitaries $\{\mathcal{U}_\ell\}_{\ell \in \mathbb{N}}$, where each $\mathcal{U}_\ell$ is an indexed list of $2^\ell$ Haar-random unitaries acting on $\ell$ qubits.

In this section, we show the following two theorems.

**Theorem 5.** *With probability 1 over $\mathcal{U}$, there exists a family of PRUs relative to $(\mathcal{U}, \mathcal{Q})$.*

**Theorem 6.** *Let $\ell : \mathbb{N} \rightarrow \mathbb{N}$ be such that $\ell(\lambda) \geq 2 \cdot \log(\lambda)$ for all sufficiently large $\lambda$. Then, relative to $(\mathcal{U}, \mathcal{Q})$, there does not exist a QDS scheme for messages of length $\ell(\lambda)$ (where $\lambda$ is the security parameter).*

As mentioned previously, we can also show that, relative to $(\mathcal{U}, \mathcal{Q})$, QDS schemes for shorter message length do not exist under a slightly different unforgeability definition than Definition 10. In this definition, for the adversary to succeed, it suffices to produce a valid signature that has not been previously produced by the challenger (even if this is the signature of a previously queried message).

## 6.1 Existence of PRS Relative to the Oracle

Theorem 5 follows from the fact that an appropriate family of pseudorandom unitaries (PRU) exists relative to $O$. The latter was shown by Kretschmer[13] [Kre21].

The construction of the PRU family is very natural. For input length $\lambda$, the family is precisely $\mathcal{U}_\lambda$. This is a PRU family on $n(\lambda) = \lambda$ qubits. For the proof, we refer the reader to Theorem 32 in [Kre21]. For any $n(\lambda) \geq \lambda$, we can construct a corresponding PRU family $\{\mathcal{U}'_\lambda\}$ on $n(\lambda)$ qubits by taking $\mathcal{U}'_\lambda = \mathcal{U}_{n(\lambda)}$. Security follows analogously.

Now, a PRU family $\{\mathcal{U}_\lambda\}_{\lambda \in \mathbb{N}}$ on $n(\lambda)$ qubits immediately implies a PRS with output length $n(\lambda)$ as follows. Let $\mathcal{U}_\lambda = \{U_k\}_{k \in \{0,1\}^\lambda}$. Then, for $k \in \{0,1\}^\lambda$, one defines $|\mathrm{PRS}(k)\rangle = U_k|0\rangle$.

The rest of this section, is dedicated to proving Theorem 6.

## 6.2 Simulating Haar Random Unitaries

Before describing an adversary that, with query-bounded access to $(\mathcal{U}, \mathcal{O})$, breaks the security of any QDS scheme, we need to introduce a crucial procedure that allows an adversary to "simulate" certain kinds of interactions with the oracle $\mathcal{U}$, by only making a small number of queries to $\mathcal{U}$. This simulation procedure was introduced by Kretschmer [Kre21]. We give a high-level description first.

**Kretschmer's Simulation Procedure: a High-Level Description.** The key property of the Haar measure that makes the following simulation possible is its strong concentration, which we describe informally. Let $C$ be a fixed binary-output quantum circuit that makes $T$ queries to a Haar random unitary on $n$ qubits (i.e. on a Hilbert space of dimension $2^n$). Then, with high probability over sampling a pair of Haar random unitaries $U, U'$, the output distributions

[13] Kretschmer shows this for a slightly different oracle $(\mathcal{U}, \mathcal{Q})$, where $\mathcal{U}$ is the same, but $\mathcal{Q}$ is a PSPACE oracle, instead of an EXP oracle. The proof is analogous. The only step in the proof where this comes into play is in Lemma 31 from [Kre21] which essentially provides a lower bound on the number of queries to $\mathcal{U}$. Crucially this lower bound holds against any unbounded quantum algorithm.

of the circuits $C^U$ and $C^{U'}$ (say on the $|0\rangle$ input) are within small TV distance. Quantitatively, for any $\delta > 0$,

$$\Pr_{U,U' \leftarrow \mu_{2^n}} \left[ \left| \Pr[C^U(|0\rangle) = 1] - \Pr[C^{U'}(|0\rangle) = 1] \right| \leq \delta \right] \leq \exp\left( -\Omega\left( \frac{2^n \cdot \delta^2}{T^2} \right) \right). \tag{4}$$

For example, when $T = poly(n)$, and $\delta$ is as small $2^{-c \cdot n}$ for $c < \frac{1}{2}$, the upper bound is *doubly* exponentially small in $n$. Thus, the concentration is strong enough to support a union bound over *all* standard basis inputs. So, with overwhelming probability over $U$ and $U'$, the output distributions of $C^U$ and $C^{U'}$ on *all* standard basis inputs are within a small constant TV distance of each other.

How does this help an adversary for a QDS scheme? Suppose the adversary is trying to perform a brute-force search over inputs to a circuit $C$ (for example the $Verify$ circuit of a QDS scheme, in the hope of finding an accepting input). This search would normally require exponentially many queries to $\mathcal{U}$ (since there are exponentially many possible inputs to the circuit). The concentration property tells us that we can ignore the particular oracle $\mathcal{U}$, and instead replace queries to $\mathcal{U}$ with queries to a family of freshly sampled Haar unitaries (or $T$-designs) by paying only a small cost in TV distance.

There is only one issue. Recall that $\mathcal{U}$ is a *family* of unitaries $\{\mathcal{U}_\ell\}_{\ell \in \mathbb{N}}$, where each $\mathcal{U}_\ell$ is an indexed list of $2^\ell$ Haar-random unitaries acting on $\ell$ qubits. When the dimension is small, relative to the number of queries, e.g. the number of qubits is $\ell = o(\log T)$, the concentration property no longer holds (or rather the upper bound in (4) becomes trivial)! Fortunately, this issue can be circumvented because, for small enough dimension, the unitaries can be "learnt" to high precision efficiently by performing process tomography. More precisely, process tomography allows one to learn an arbitrary unitary on a space of dimension $d$ by making only $poly(d)$ queries. Thus, let $\lambda$ be a security parameter. Then, unitaries in $\mathcal{U}_\ell$, for $\ell = O(\log \lambda)$, can be learnt with only $poly(\lambda)$ queries. Moreover, notice that when $\ell = O(\log \lambda)$, there are only a total of $2^\ell = poly(\lambda)$ unitaries in $\mathcal{U}_\ell$, and so one can learn *all* of them with a total of only $poly(\lambda)$ time and queries. On the other hand, unitaries in $\mathcal{U}_\ell$ for $\ell \geq c \cdot \log \lambda$, for a large enough constant $c$, enjoy a strong enough concentration, and can thus be replaced with fresh Haar unitaries, or $T$-designs, at a small cost in TV distance.

To summarize, let $C$ be a quantum circuit on inputs of size $\lambda$. Suppose $C$ makes $poly(\lambda)$ queries to $\mathcal{U}$. With high probability over $\mathcal{U}$, the output distribution of $C$ on *all* standard basis inputs can be simulated to within a small constant in TV distance with only $poly(\lambda)$ queries to $\mathcal{U}$ by:

– Using $T$-designs to replace the oracle calls to $\mathcal{U}_\ell$, for large enough $\ell$ (such as $\ell \geq c \cdot \log \lambda$), for some large enough constant $c$). This perfectly simulates $T$ queries to a freshly sampled family $\mathcal{U}_\ell$, and, by the concentration property, this in turn approximates the original output distribution of each run to

within a small $\delta$, where we can take $\delta$ to be a small constant[14]. This step does not require any queries to $\mathcal{U}$.
- Efficiently learning all of the unitaries in $\mathcal{U}_\ell$, for all small enough $\ell$ (e.g. $\ell \leq \Theta(\log \lambda)$). This step requires $poly(\lambda)$ time and queries to $\mathcal{U}$.

**Kretschmer's Simulation Procedure: a Formal Description.** Before formally describing the simulation procedure, we introduce the necessary lemmas relating to process tomography and $t$-designs. For a unitary $U$, we let $U(\cdot)U^\dagger$ denote the quantum channel $\rho \rightarrow U\rho U^\dagger$.

*Process Tomography* We focus on process tomography for a unitary channel. In this setting, the algorithm is given black-box access to a unitary $Z$. After a number of queries to $Z$, the algorithm should output a classical description of a unitary $\tilde{Z}$ with the goal of minimizing the following quantity:

$$\mathbb{E}\left[\left\|Z(\cdot)Z^\dagger - \tilde{Z}(\cdot)\tilde{Z}^\dagger\right\|_\diamond\right].$$

In [HKOT23], the authors describe an algorithm with the following guarantees. This algorithm is used as a sub-routine in Kretschmer's simulation procedure.[15]

**Theorem 7 ([HKOT23]).** *There is a quantum algorithm that, given $\epsilon, \eta \in (0,1)$, as well as black-box access to an unknown $d$-dimensional unitary $Z \in \mathbb{U}(d)$, makes $O(\frac{d^2}{\epsilon}\log(1/\eta))$ queries to the black box and outputs a classical description of a unitary $\tilde{Z} \in \mathbb{U}(d)$ such that $\Pr\left[\left\|Z(\cdot)Z^\dagger - \tilde{Z}(\cdot)\tilde{Z}^\dagger\right\|_\diamond \leq 3\epsilon\right] \geq 1-\eta$. The gate complexity of this algorithm is $poly(d, 1/\epsilon, \log(1/\eta))$*

*Approximate T Designs.* An $\epsilon$-approximate quantum unitary $t$-design is a distribution over unitaries that "$\epsilon$-approximates" a Haar random unitary, when considering their action via a $t$-copy parallel repetition.

**Definition 11.** *(Approximate Unitary Design [BHH16]) Let $\epsilon \in [0,1], t \in \mathbb{N}$. A probability distribution $S$ over $\mathbb{U}(N)$ is an $\epsilon$-approximate unitary $t$-design if:*

$$(1-\epsilon)\mathbb{E}_{U \leftarrow \mu_N}[(U(.)U^\dagger)^{\otimes t}] \preceq \mathbb{E}_{U\sim S}[(U(.)U^\dagger)^{\otimes t}] \preceq (1-\epsilon)\mathbb{E}_{U \leftarrow \mu_N}[(U(.)U^\dagger)^{\otimes t}],$$

*where $B \preceq A$ means that $B - A$ is positive semidefinite.*

It is well-known that there are efficient constructions of such unitary $t$-designs.

[14] Note that we cannot take $\delta$ to be exponentially small in $\lambda$ here because $\ell$ could be as small as $c \cdot \log \lambda$.

[15] The original result of [HKOT23] is for $\eta = \frac{1}{3}$, but this can be boosted to any $\eta$ at a cost of factor of $\log(1/\eta)$ in the number of queries, and gate complexity (see Proposition 2.4 in [HKOT23]).

**Lemma 2.** **([BHH16]).** *There exists $m : \mathbb{N} \to \mathbb{N}$, such that the following holds. For each $n, t \in \mathbb{N}$, and $\epsilon > 0$, there is a poly$(n, t, \log(\frac{1}{\epsilon}))$-time classical algorithm $A$ that takes $m(n)$ bits of randomness as input, and outputs a description of a unitary quantum circuit on $n$ qubits such that the output distribution of $A$ is an $\epsilon$-approximate unitary $t$-design (over $\mathbb{U}(2^n)$).*

An $\epsilon$-approximate unitary $t$-design $S$ is said to be *phase-invariant* if, for any unitary $U$ in the support of $S$, $U$ and $\omega U$ are sampled with the same probability, where $\omega$ is the $(t+1)$-th root of unity. The following lemma says that replacing the Haar measure with a phase-invariant $\epsilon$-approximate unitary $t$-design is undetectable to any algorithm that makes only $t$ queries to the Haar random unitary.

**Lemma 3.** **([Kre21]).** *Let $S$ be a phase-invariant $\epsilon$ approximate unitary $t$-design over $\mathbb{U}(N)$, and let $D^{(\cdot)}$ be any $t$-query quantum algorithm. Then,*

$$(1-\epsilon) \Pr_{U \leftarrow \mu_N}[D^U = 1] \leq \Pr_{U \leftarrow S}[D^U = 1] \leq (1+\epsilon) \Pr_{U \leftarrow \mu_N}[D^U = 1]$$

*(where in the above, $D^{(\cdot)}$ is also allowed to make queries to controlled-$U$).*

We are now ready to describe Kretschmer's simulation procedure formally. For convenience, we denote the simulation procedure as Sim-Haar. Formally, Sim-Haar takes as input a description of a circuit $C^{(\cdot)}$ that makes queries to $\mathcal{U}$ (and possibly some other oracle $\mathcal{Q}$), as well as some other parameters that we will describe shortly. It outputs a circuit $C'$ that does not make queries to $\mathcal{U}$ (but possibly to $\mathcal{Q}$). Here is a formal description.

In the following lemma, $\mathcal{U}$ is as in Algorithm 1, and $\mathcal{Q}$ is any other fixed oracle.

**Lemma 4** **([Kre21]).** *Let $\eta \in \mathbb{N}$ and $\delta \in (0, 1/3)$. Let Sim-Haar$_{\eta,\delta}$ denote Algorithm 1 where the inputs $\eta$ and $\delta$ are fixed. Let $C^{(\cdot)}$ be a binary-output quantum circuit that uses space $s$ and makes $T$ queries to $(\mathcal{U}, \mathcal{Q})$. Then, Sim-Haar$^{\mathcal{U}}_{\eta,\delta}(C)$ runs in time poly$(\eta, s, T, \frac{1}{\delta})$, and, with probability at least $1 - 2e^{-\eta}$ over the randomness of Sim-Haar$_{\eta,\delta}$ and the sampling of $\mathcal{U}$, we have that, for all $x \in \{0,1\}^n$ (where $n$ is the length of inputs to $C$),*

$$\left| \Pr[C'^{\mathcal{Q}}(|x\rangle) = 1] - \Pr[C^{\mathcal{U},\mathcal{Q}}(|x\rangle) = 1] \right| \leq 3\delta + e^{-\frac{\eta}{2}}.$$

*Proof.* Recall that $d = \log(192\frac{1}{\delta^2}(\eta + s)T^2 + 2)$, and $|x\rangle$ is a computational basis state that $C^{\mathcal{U},\mathcal{Q}}$ takes as input. We define the following sequence of "hybrids". These are probability distributions, where the first is the output distribution of circuit $C^{\mathcal{U},\mathcal{Q}}(|x\rangle)$, and the last is the output distribution of circuit $C'^{\mathcal{Q}}(|x\rangle)$. We show that each two consecutive distributions are close. Let $x \in \{0,1\}^n$.

1. $H_1$: $C^{\mathcal{U},\mathcal{Q}}(|x\rangle)$.
2. $H_2$: $C_2(|x\rangle)$, where $C_2$ is identical to $C^{\mathcal{U},\mathcal{Q}}$ except that, for all $\ell \in [d+1, s]$, $\mathcal{U}_\ell$ is replaced by a freshly sampled family of $2^\ell$ Haar random unitaries.

**Algorithm 1.** Sim-Haar$^{(\cdot)}$

**Oracle access:** The algorithm has query access to an oracle $\mathcal{U} = \{\mathcal{U}_\ell\}_{\ell\in\mathbb{N}}$, where each $\mathcal{U}_\ell$ is a list of $2^\ell$ different $\ell$-qubit unitary transformations (and possibly to another oracle $\mathcal{Q}$).

**Input**: A quantum circuit $C^{(\cdot)}$ using space $s$ that makes $T$ queries to $\mathcal{U}$ (and possibly to another oracle $\mathcal{Q}$), $\eta \in \mathbb{N}$, and $\delta \in (0, 1/3)$.

We denote the unitaries in the list $\mathcal{U}_\ell$ as $\{U_{k\ell}\}_{k\in\{0,1\}^\ell}$. Let $d = \log(192\frac{1}{\delta^2}(\eta+s)\cdot T^2+2)$.

For $l \in [s]$:

- If $\ell \in [d]$, view the list $\mathcal{U}_\ell$ as a unitary on a larger space that includes a control register for the index $k$. Classically simulate $\mathcal{U}_\ell$ by running the process tomography algorithm from Theorem 6 on inputs $\epsilon = \frac{\delta}{T}$ and $\mu = \frac{1}{d}e^{-2(\eta+s)}$. This produces estimates $\tilde{\mathcal{U}_\ell}$ such that $\|\tilde{\mathcal{U}_\ell}(\cdot)\tilde{\mathcal{U}_\ell}^\dagger - \mathcal{U}_\ell(\cdot)\mathcal{U}_\ell{}^\dagger\|_\diamond \leq \frac{\delta}{T}$ with probability at least $1 - \frac{1}{d}e^{-2(\eta+s)}$. From Theorem 6, this can be done, for each $\ell$, with a number of queries that is $O\left(\frac{2^{2\ell}T}{\delta}\cdot\log(\frac{1}{e^{-\eta}})\right) \leq O\left(\frac{2^{2d}T}{\delta}\log(\frac{1}{e^{-\eta}})\right) \leq O(\frac{T^3\eta^2}{\delta^3})$ and running time $poly(\ell, T/\delta)$.
- Otherwise, if $\ell \in [d+1, s]$, do the following. Let $\epsilon = \frac{\delta}{s2^s}$. Let $m : \mathbb{N} \to \mathbb{N}$ be as in Lemma 2. Let $A$ be the "unitary design sampler" algorithm from Lemma 2 with parameters $\epsilon, n = \ell$, and $t = T$. Sample $f_\ell : \{0,1\}^\ell \to \{0,1\}^{m(l)}$ from a $2T$-wise independent family of functions. Set $\tilde{U}_{k\ell} = A(f_\ell(k))$.

**Output**: The quantum circuit $C'$ defined as follows. $C'$ is identical to $C$ except that, for each $\ell$ and $k \in \{0,1\}^\ell$, all queries to $\mathcal{U}_{k\ell}$ are replaced with a direct application of the unitary $\tilde{U}_{k\ell}$ defined earlier. Thus $C'$ makes no queries to $\mathcal{U}$ (but still makes queries to $\mathcal{Q}$, if $C$ does).

3. $H_3$: $C_3(|x\rangle)$, where $C_3$ is sampled as follows. Let $A$ be the algorithm from Lemma 2 that samples from a phase invariant $\epsilon$-approximate unitary $T$-design, where $\epsilon = \frac{\delta}{s2^s}$. For $\ell \in [d+1, s]$, sample a function $g_\ell : \{0,1\}^\ell \to \{0,1\}^{m(\ell)}$ uniformly at random. $C_3$ is identical to $C_2$ except that, for $k \in \{0,1\}^\ell$, we replace queries to $U_{k\ell}$ with queries to $A(g_\ell(k))$.
4. $H_4$: $C_4(|x\rangle)$, where $C_4$ is sampled in the same way as $C_3$ except that we replace $g_\ell$ (which was previously sampled uniformly at random) with $f_\ell : \{0,1\}^\ell \to \{0,1\}^{m(\ell)}$, sampled from a $2T$-wise independent function family.
5. $H_5$: $C_5(|x\rangle)$, where $C_5$ is sampled in the same way as $C_4$ except that, for $\ell \in [d]$, queries to $\mathcal{U}_\ell$ are replaced by queries to $\tilde{\mathcal{U}_\ell}$, obtained via the process tomography algorithm from Theorem 7 with parameters $\epsilon = \frac{\delta}{T}$, and $\mu = \frac{1}{d}e^{-2(\eta+s)}$. Note that this circuit is exactly $C'^Q(|x\rangle)$.

Let $f(\mathcal{U}) = \Pr[C^{\mathcal{U},\mathcal{Q}}(|x\rangle) = 1]$. Lemma 1 implies that this function is $2T$-Lipshitz. Invoking the strong concentration of the Haar measure in Theorem 2 with $t = \delta$ and $L = 2T$, we have that, for any standard basis input $|x\rangle$,

$$\Pr_{\mathcal{U},\mathcal{U}'}[|\Pr[C^{\mathcal{U},\mathcal{Q}}(|x\rangle) = 1] - \Pr[C^{\mathcal{U}',\mathcal{Q}}(|x\rangle) = 1]| \geq \delta] \leq \exp(-\frac{(2^d-2)\delta^2}{24(2T)^2}) \leq e^{-2(\eta+s)},$$

where the last inequality follows from the definition of $d$. By an averaging argument and a straightforward calculation, the latter implies that, in fact,

$$\Pr_{\mathcal{U}}\Big[\big|\Pr[C^{\mathcal{U},\mathcal{Q}}(|x\rangle)=1]-\Pr[C_2(|x\rangle)=1]\big|\geq\delta+e^{-\eta-s}\Big]\leq e^{-\eta-s}$$

(where the main difference from the previous expression is that the probability over $\mathcal{U}'$ has been absorbed inside $C_2$).

Now, $C_3$ replaces every unitary $U_{k\ell}$ for $\ell\in[d+1,s]$ and $k\in\{0,1\}^\ell$ with $A(g_\ell(k))$. Using Lemma 3, the total change in acceptance probability is

$$\sum_{\ell=d+1}^{s}\sum_{k\in\{0,1\}^\ell}\frac{\delta}{s2^s}\leq\delta.$$

Thus,

$$\big|\Pr[C_3(|x\rangle)=1]-\Pr[C_2(|x\rangle)=1]\big|\leq\delta.$$

From [Zha12], we know that $T$ queries to a random function are perfectly indistinguishable from queries to a $2T$-wise independent family of functions. Thus, we have

$$\Pr[C_4(|x\rangle)=1]=\Pr[C_3(|x\rangle)=1].$$

From Theorem 7, for each $\ell\in[d]$, with probability at least $1-\frac{1}{d}e^{-2(\eta+s)}$ (over the randomness of the process tomography algorithm), we have that $\|\tilde{\mathcal{U}}_\ell(\cdot)\tilde{\mathcal{U}}_\ell^\dagger-\mathcal{U}_\ell(\cdot)\mathcal{U}_\ell^\dagger\|_\diamond\leq\frac{\delta}{T}$. Thus, by a union bound, with probability $\geq 1-e^{-2(\eta+s)}$ (over the randomness of the process tomography), we have that, for all $\ell\in[d]$,

$$\|\tilde{\mathcal{U}}_\ell(\cdot)\tilde{\mathcal{U}}_\ell^\dagger-\mathcal{U}_\ell(\cdot)\mathcal{U}_\ell^\dagger\|_\diamond\leq\frac{\delta}{T}.$$

Since $C_5$ and $C_4$ only make $T$ queries to $\mathcal{U}$, it follows, by triangle inequalities and Fact 1, that

$$\big|\Pr[C_5(|x\rangle)=1]-\Pr[C_4(|x\rangle)=1]\big|\leq\delta.$$

Adding up differences in acceptance probabilities (and adding up the probability losses) we get that, with probability at least $1-(e^{-\eta-s}+e^{-2(\eta+s)})$ over the randomness of $\mathcal{U}$, and the randomness in the process tomography (i.e. the randomness of $\mathsf{Sim\text{-}Haar}_{\eta,\delta}$),

$$\Big|\Pr[C^{\mathcal{U},\mathcal{Q}}(|x\rangle)=1]-\Pr[C'^{\mathcal{Q}}(|x\rangle)=1]\Big|\leq 3\delta+e^{-\frac{\eta}{2}}.$$

Finally, taking a union bound over all standard basis inputs $|x\rangle$, we have that, with probability at least $1-2^s\cdot(e^{-\eta-s}+e^{-2(\eta+s)})\geq 1-2e^{-\eta}$ over the randomness of $\mathcal{U}$, and the randomness of $\mathsf{Sim\text{-}Haar}_{\eta,\delta}$), for all standard basis inputs $|x\rangle$,

$$\Big|\Pr[C^{\mathcal{U},\mathcal{Q}}(|x\rangle)=1]-\Pr[C'^{\mathcal{Q}}(|x\rangle)=1]\Big|\leq 3\delta+e^{-\frac{\eta}{2}},\tag{5}$$

as desired.

## 6.3 An Adversary Breaking Any QDS Scheme Relative to the Oracle

In this section, we prove Theorem 6. Concretely, we describe an adversary that, relative to $(\mathcal{U}, \mathcal{Q})$ (where $(\mathcal{U}, \mathcal{Q})$ is defined at the start of Sect. 6), breaks any QDS scheme for messages of length $\ell(\lambda)$ for any $\ell$ such that $\ell(\lambda) \geq c \cdot \log(\lambda)$ for large enough $\lambda$. We show, for example, that one can take $c = 2$ (although our analysis is not tight).

We describe our adversary in Sect. 6.3, and we provide the analysis in Sect. 6.3. For a more informal overview see the technical overview (Sects. 2.1 and 2.2).

**The Adversary.** Let $(SKGen^{\mathcal{U},\mathcal{Q}}, PKGen^{\mathcal{U},\mathcal{Q}}, Sign^{\mathcal{U},\mathcal{Q}}, Verify^{\mathcal{U},\mathcal{Q}})$ be a QDS scheme. We take the length of the secret key to be the security parameter $\lambda$.

$\mathcal{A}$ behaves as follows on input $1^\lambda$ (technically $\mathcal{A}$ also receives polynomially many copies of $|pk\rangle$, but it does not need them).

1. Let $t = 40\lambda$. $\mathcal{A}$ samples messages $m_1, \ldots, m_t \leftarrow \mathcal{M}_\lambda$, and queries the challenger at these messages. The messages are sampled uniformly at random (possibly with repetitions) subject to the condition that $\bigcup_{i \in [t]} \{m_i\} \neq \mathcal{M}$. Let $\sigma_1, \ldots, \sigma_t$ be the signatures returned by the challenger.
2. Define the circuit $VerPKGen^{(\cdot)}(\cdot, \cdot, \cdot)$ to be such that
$$VerPKGen^{\mathcal{U},\mathcal{Q}}(sk, m, \sigma) := Verify^{\mathcal{U},\mathcal{Q}}(PKGen(sk), m, \sigma).$$
$\mathcal{A}$ obtains $VerPKGen'^{(\cdot)}(\cdot, \cdot, \cdot) \leftarrow \mathsf{Sim\text{-}Haar}^{\mathcal{U},\mathcal{Q}}_{\lambda, \frac{1}{300} - e^{-\frac{\lambda}{2}}}(VerPKGen)$, where $VerPKGen'^{(\cdot)}(\cdot, \cdot, \cdot)$ is a circuit that makes queries to $\mathcal{Q}$ (but not to $\mathcal{U}$). Here, as earlier, the notation $\mathsf{Sim\text{-}Haar}_{\eta,\delta}$ refers to running algorithm $\mathsf{Sim\text{-}Haar}$ (from Algorithm 1) on the fixed inputs $\eta$ and $\delta$. Going forward, for ease of notation, we simply denote this circuit by $VerPKGen'$.
3. Define the circuit $VerPKGenSign^{(\cdot)}(\cdot, \cdot, \cdot)$ to be such that
$$VerPKGenSign^{\mathcal{U},\mathcal{Q}}(sk, m, sk') := Verify^{\mathcal{U},\mathcal{Q}}(PKGen(sk), m, Sign(sk', m)).$$
$\mathcal{A}$ obtains $VerPKGenSign'^{(\cdot)}(\cdot, \cdot, \cdot) \leftarrow \mathsf{Sim\text{-}Haar}^{\mathcal{U},\mathcal{Q}}_{\lambda, \frac{1}{300} - e^{-\frac{\lambda}{2}}}(VerPKGenSign)$. Note that $VerPKGenSign'^{(\cdot)}(\cdot, \cdot, \cdot)$ is a circuit that makes queries to $\mathcal{Q}$ (but not to $\mathcal{U}$). Going forward, for ease of notation, we simply denote this circuit by $VerPKGenSign'$.
4. At this point, $\mathcal{A}$ invokes $\mathcal{Q}$. For simplicity, we will describe $\mathcal{Q}$'s behaviour as a probabilistic exponential time algorithm. However, formally, $\mathcal{Q}$ is a *deterministic* function that, on input the instance of a fixed $\mathsf{EXP}$-complete search problem, returns the solution. So, formally,
   - $\mathcal{A}$ also provides the randomness as input to $\mathcal{Q}$ (and this is fine since the algorithm we describe only uses a polynomial length random string).

– $\mathcal{A}$ first computes a reduction from the search problem $P$ solved by the algorithm to the fixed EXP-complete problem, and maps the original input to the corresponding input according to the reduction.

From here on, we will not consider these two formalities.

$\mathcal{A}$ provides $\{m_i, \sigma_i\}_{i\in[t]}$, $VerPKGen'$, $VerPKGenSign'$ as input to $\mathcal{Q}$, which returns a set candidates as in Algorithm 2 below.

**Algorithm 2.**

```
Input: $\{m_i, \sigma_i\}_{i\in[t]}$, $VerPKGen'$, $VerPKGenSign'$.
1. Initialize Consistent $= \emptyset$. For $sk \in \{0,1\}^\lambda$:
   • If $\Pr[VerPKGen'(sk, m_i, \sigma_i) = 1] \geq 9/10$ for all $i \in [t]$, update Consistent $\leftarrow$
     Consistent $\cup \{sk\}$[18].
2. Let $S_1 =$ Consistent, and candidates $= \emptyset$. For $j \in [\lambda^2]$:
   • Initialize $stingy_j = \emptyset$. For $sk \in S_j$:
     ∗ Let $friends_{sk} = \emptyset$.
     ∗ For each $sk' \neq sk \in S_j$, do the following:
       · Count the number of $m \in \mathcal{M}$ such that
         $\Pr[VerPKGenSign'(sk, m, sk') = 1] > \frac{1}{10}$. If this is at least
         $\frac{1}{10} \cdot |\mathcal{M}|$, update $friends_{sk} \leftarrow friends_{sk} \cup \{sk'\}$.
     ∗ If $|friends_{sk}| \leq \frac{1}{2} \cdot |S_j|$, then $stingy_j \leftarrow stingy_j \cup \{sk\}$.
   • Initialize $goodSigner_j = \emptyset$. For $sk \in S_j$:
     ∗ Let $accept_{sk,j} = \emptyset$.
     ∗ For $sk' \neq sk \in S_j$, do the following:
       · Count the number of $m \in \mathcal{M}$ such that
         $\Pr[VerPKGenSign'(sk', m, sk) = 1] \geq \frac{1}{10}$ (note that the role of
         $sk$ and $sk'$ is flipped compared to the definition of $friends_{sk}$). If this
         is at least $\frac{1}{10} \cdot |\mathcal{M}|$, update $accept_{sk,j} \leftarrow accept_{sk,j} \cup \{sk'\}$.
     ∗ If $accept_{sk,j} = S_j \setminus \{sk\}$, update
         $$goodSigner_j \leftarrow goodSigner_j \cup \{sk\}\,.$$
   • Let $S_{j+1} = stingy_j \cap goodSigner_j$. If $S_{j+1} = \emptyset$, halt and output candidates.
     Otherwise, sample $sk \leftarrow S_{j+1}$. Update candidates $\leftarrow$ candidates $\cup \{sk\}$.
3. Output candidates.
```

[16] Recall that $VerPKGenSign'$ makes queries to $\mathcal{Q}$. Nonetheless, the problem of computing whether $\Pr[VerPKGenSign'(sk', m_i, \sigma_i) = 1] \geq \frac{9}{10}$ can still be cast as an EXP problem: this is the problem of computing whether the magnitude squared of a particular entry of a vector, obtained by performing (exponentially-sized) matrices-vector multiplications, is $\geq \frac{9}{10}$. The problem can be cast in this way because each query to $\mathcal{Q}$ that the algorithm makes is a multiplication by an (exponential-sized) unitary that corresponds to solving $\mathcal{Q}$'s EXP-complete problem (on inputs of a certain polynomial size). Note that computing whether an entry of the resulting vector is greater or equal to some *rational number* can be done deterministically.

5. Let $\mathcal{M}_{\text{queried}} = \bigcup_{i \in [t]} \{m_i\}$. $\mathcal{A}$ samples $sk \leftarrow \mathsf{candidates}$, $m \leftarrow \mathcal{M} \setminus \mathcal{M}_{\text{queried}}$, and runs
   $\sigma \leftarrow Sign^{\mathcal{U},\mathcal{Q}}(sk, m)$. $\mathcal{A}$ outputs $(m, \sigma)$.

**The Analysis.** Please see the full version of the paper for the analysis.

### 6.4 Why Our Attack Does Not Work for QDS with Quantum Secret Key And/Or Signatures

One of the main questions left open by this work is whether there exists a black-box construction from PRS of a QDS scheme with quantum public keys and *quantum* secret keys and/or signatures. Our attack from the previous subsection crucially only applies to QDS schemes with quantum public key but *classical* secret key and signatures. The main issue in extending this attack is a bit subtle. The issue is that in Lemma 4, which is based on the strong concentration of the Haar measure, the closeness guarantee of Lemma 4 holds for all *standard basis* inputs to the circuits. Crucially, it does not hold for all possible quantum state inputs. This is because, the proof of Lemma 4 relies on a union bound, over all standard basis inputs, to obtain Eq. (5). This gives a useful bound because the set of standard basis inputs is of size "only" $2^n$, where $n$ is the input-size of the circuit. However, it is unclear how to argue similarly when the union bound is over quantum states, since there are infinitely many of them. One could hope to define an $\epsilon$-net for the set of quantum states (for some sufficiently small $\epsilon$), and apply a union bound over the states in the $\epsilon$-net. Unfortunately, the number of states in the $\epsilon$-net is doubly exponential (even when $\epsilon$ is a constant)! This is too large for a union bound to provide a non-trivial bound. So it seems that the strong concentration of the Haar measure is not quite strong enough for the current simulation technique to be useful in this setting.

*Remark 2.* As mentioned earlier, our oracle separation does extend to a restricted kind of QDS scheme with *quantum* secret keys. Specifically, if the QDS scheme is such that *SKGen* outputs a quantum state, but does *not* query $\mathcal{U}$, then (a slight variation on) our attack still works. The point is that if *SKGen* does not query $\mathcal{U}$ at all, then the set of possible secret keys $|sk\rangle$ is "only" of size $2^{poly(\lambda)}$ (rather than being doubly exponential in an $\epsilon$-net): these are all of the post-measurement states that can result from running a poly-size quantum circuit and making a partial measurement on a subset of the qubits. Thus, a union bound is indeed still useful, and one can adjust the failure probability of $\mathsf{Sim\text{-}Haar}$ appropriately (while maintaining polynomial runtime) based on the size of the circuit for *SKGen*.

## References

[AGM21] Alagic, G., Gagliardoni, T., Majenz, C.: Can you sign a quantum state? Quantum **5**, 603 (2021)

[AGQY23] Ananth, P., Gulati, A., Qian, L., Yuen, H.: Pseudorandom (function-like) quantum state generators: new definitions and applications. In: Kiltz, E., Vaikuntanathan, V. (eds.) TCC 2022, Part I. LNCS, vol. 13747, pp. 237–265. Springer, Cham (2023). https://doi.org/10.1007/978-3-031-22318-1_9

[ALY23] Ananth, P., Lin, Y.-T., Yuen, H.: Pseudorandom strings from pseudorandom quantum states. arXiv preprint arXiv:2306.05613 (2023)

[AQY22] Ananth, P., Qian, L., Yuen, H.: Cryptography from pseudorandom quantum states. In: Dodis, Y., Shrimpton, T. (eds.) CRYPTO 2022, Part I. LNCS, vol. 13507, pp. 208–236. Springer, Cham (2022). https://doi.org/10.1007/978-3-031-15802-5_8

[BCQ22] Brakerski, Z., Canetti, R., Qian, L.: On the computational hardness needed for quantum cryptography. arXiv preprint arXiv:2209.04101 (2022)

[BHH16] Fernando Brandao, G.S.L., Harrow, A.W., Horodecki, M.: Local random quantum circuits are approximate polynomial-designs. Commun. Math. Phys. 397–434 (2016)

[CCS24] Chen, B., Coladangelo, A., Sattath, O.: The power of a single HAAR random state: constructing and separating quantum pseudorandomness (2024, to appear)

[HKOT23] Haah, J., Kothari, R., O'Donnell, R., Tang, E.: Query-optimal estimation of unitary channels in diamond distance. arxiv preprint arxiv:2302.14066 (2023)

[Imp95] Impagliazzo, R.: A personal view of average-case complexity. In: Tenth Annual IEEE Conference on Proceedings of Structure in Complexity Theory, pp. 134–147. IEEE (1995)

[IR89] Impagliazzo, R., Rudich, S.: Limits on the provable consequences of one-way permutations. In: Proceedings of the Twenty-First Annual ACM Symposium on Theory of Computing, pp. 44–61 (1989)

[JLS18a] Ji, Z., Liu, Y.-K., Song, F.: Pseudorandom quantum states. In: Shacham, H., Boldyreva, A. (eds.) CRYPTO 2018, Part III. LNCS, vol. 10993, pp. 126–152. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-96878-0_5

[Kre21] Kretschmer, W.: Quantum pseudorandomness and classical complexity. In: 16th Conference on the Theory of Quantum Computation, Communication and Cryptography (2021)

[KT23] Khurana, D., Tomer, K.: Commitments from quantum one-wayness. arXiv preprint arXiv:2310.11526 (2023)

[Mec19] Meckes, E.S.: The Random Matrix Theory of the Classical Compact Groups. Cambridge Tracts in Mathematics, pp. 6–9 (2019)

[MY22a] Morimae, T., Yamakawa, T.: One-wayness in quantum cryptography. arXiv preprint arXiv:2210.03394 (2022)

[MY22b] Morimae, T., Yamakawa, T.: Quantum commitments and signatures without one-way functions. In: Dodis, Y., Shrimpton, T. (eds.) CRYPTO 2022, Part I. LNCS, vol. 13507, pp. 269–295. Springer, Cham (2022). https://doi.org/10.1007/978-3-031-15802-5_10

[NC11] Nielsen, M.A., Chuang, I.L.: Quantum Computation and Quantum Information: 10th Anniversary Edition, 10th edn. Cambridge University Press, Cambridge (2011)

[RTV04] Reingold, O., Trevisan, L., Vadhan, S.: Notions of reducibility between cryptographic primitives. In: Naor, M. (ed.) TCC 2004. LNCS, vol. 2951, pp. 1–20. Springer, Heidelberg (2004). https://doi.org/10.1007/978-3-540-24638-1_1

[Zha12] Zhandry, M.: Secure identity-based encryption in the quantum random oracle model. In: Safavi-Naini, R., Canetti, R. (eds.) CRYPTO 2012. LNCS, vol. 7417, pp. 758–775. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-32009-5_44

# On the Black-Box Complexity of Private-Key Inner-Product Functional Encryption

Mohammad Hajiabadi[1], Roman Langrehr[2](✉), Adam O'Neill[3], and Mingyuan Wang[4]

[1] University of Waterloo, Waterloo, Canada
mdhajiabadi@uwaterloo.ca
[2] ETH Zürich, Zürich, Switzerland
roman.langrehr@inf.ethz.ch
[3] Manning CICS, UMass Amherst, Amherst, USA
adamo@cs.umass.edu
[4] NYU Shanghai, Shanghai, China
mingyuan.wang@nyu.edu

**Abstract.** We initiate the study of the black-box complexity of *private-key* functional encryption (FE). Of central importance in the private-key setting is the *inner-product* functionality, which is currently only known from assumptions that imply public-key encryption, such as Decisional Diffie-Hellman or Learning-with-Errors. As our main result, we rule out black-box constructions of private-key inner-product FE from random oracles. This implies a black-box separation between private-key inner-product FE from all symmetric-key primitives implied by random oracles (e.g., symmetric-key encryption and collision-resistant hash functions).

Proving lower bounds for private-key functional encryption schemes introduces challenges that were absent in prior works. In particular, the combinatorial techniques developed by prior works for proving black-box lower bounds are only useful in the public-key setting and predicate encryption settings, which all fail for the private-key FE case. Our work develops novel combinatorial techniques based on Fourier analysis to overcome these barriers. We expect these techniques to be widely useful in future research in this area.

**Keywords:** Black-box impossibility · Functional encryption

## 1 Introduction

### 1.1 Background and Main Question

A major goal in cryptography is to identify minimal assumptions sufficient for realizing cryptographic primitives. Functional encryption (FE) [19,30,34] is a vast generalization of standard encryption whereby secret key holders can decrypt a given ciphertext to various corresponding functions of its underlying plaintext. In particular, secret keys are associated with functions, and a secret key holder for a function $f$ can learn $f(x)$ from an encryption of $x$. Security notions for FE capture the intuitive idea that secret keys for functions $f_1, \ldots, f_w$ should reveal only what can already be learned from the outputs of these functions on the underlying plaintexts.

For functional encryption, if the number of corruptions is *a priori* bounded such that the size of the system parameters can depend on this bound, the minimal complexity required is well-understood. In particular, one can build public-key (resp., private-key) bounded-collusion FE for any function family from the minimal assumption that CPA-secure public-key (resp., private-key) encryption schemes exist [11,25,33].

However, things are unclear in the *unbounded collusion* case, where an adversary can obtain (a.k.a., corrupt) secret keys for many functions of her choosing. It is known that functional encryption with unbounded collusions for arbitrary polynomial-sized circuits implies Indistinguishability Obfuscation (iO) [10,15]. Therefore, it is unlikely that we can build unbounded functional encryptions from plain CPA-secure encryption schemes. However, the question remains if, for some less expressive families of functions, unbounded functional encryptions can be built from minimal assumptions. Thus, much research has been devoted to realizing and improving the efficiency of unbounded FE for specific restrictive functionalities such as identity-based encryption [17], attribute-based encryption [34], predicate encryption [28], inner-product FE [2], quadratic FE [12], and attribute-weighted sums [6].

*Private Key vs Public Key:* For most advanced encryption systems (e.g., key-dependent message (KDM) security [16,20], homomorphic encryption [23]) building private-key schemes appears to be as much challenging as their public-key counterparts, and sometimes even in a provable way [32]. For FE, the situation seems to be different. For example, consider identity-based encryption (IBE) [17], which corresponds to point functions defined as $F_{\mathsf{id}}(\mathsf{id}', m) = m$ if $\mathsf{id} = \mathsf{id}'$, and $F_{\mathsf{id}}(\mathsf{id}', m) = \perp$, otherwise.[1] IBE is so far only possible in a black-box way from pairings/LWE, and is known to be black-box impossible from trapdoor permutations (TDPs) [18] or generic groups [31,35,39]. On the other hand, FE for point functions in the private-key setting can be trivially built from pseudo-random functions (PRFs) [24] as follows. Let the master secret key $\mathsf{msk}$ be a PRF key, and let $k[\mathsf{id}'] := \mathsf{PRF}(\mathsf{msk}, \mathsf{id}')$ be a secret key for a point $\mathsf{id}'$. Define $\mathrm{Enc}(\mathsf{msk}, (\mathsf{id}', m))$ as $\mathrm{Enc}_{\mathsf{priv}}(k[\mathsf{id}'], m)$, where $k[\mathsf{id}'] = \mathsf{PRF}(\mathsf{msk}, \mathsf{id}')$, and where $\mathrm{Enc}_{\mathsf{priv}}$ is the encryption function of a CPA-secure private-key encryption scheme.

What makes the above private-key construction possible are two points: (a) that a master secret key can implicitly generate exponentially many private keys and (b) each ciphertext can be decrypted by *exactly one* secret key, the same identity. In particular, the above observation readily generalizes to

[1] The standard security notion for IBE allows the ciphertext is allowed to leak id. IBEs that also hide the identity are called *anonymous*.

building private-key FE for any function family $F$ under which for any plaintext $x$, for all but a polynomial number (in the security parameter $\kappa$) of keys $f \in F$, $f(x) = \bot$. For encrypting $x$, if $f_1, \ldots, f_{\mathsf{poly}(\kappa)}$ are the only functions for which $f_i(x) \neq \bot$ for $i \in [\mathsf{poly}(\kappa)]$, then encrypt $f_i(x)$ for every $i$ under $\mathsf{PRF}(\mathsf{msk}, f_i)$, the secret key for $f_i$. The size of the final ciphertext remains polynomial because at most a polynomial number of $f_i(x)$ values are encrypted.

Thus, unlike in the public-key setting, for which we have lower bounds on FE for certain function families (e.g., IBE) [18,29] and a better understanding of the hierarchy between different functionalities (e.g., separations between IBE and more expressive functionalities such as attribute-based and predicate encryption [26]), our understanding of FE in the private-key setting is lacking. In particular, the lower bounds in the public key setting fail in the private setting, due to the positive result above and due to the more technical reason that in the private-key setting, encryptions are made relative to master secret keys, capable of generating exponentially many private keys. But in the public-key setting, encryptions are made relative to master public keys, which are bound to encode at most polynomially many public keys. We will elaborate more on these later.

Motivated by the above discussion, a seemingly basic characterization of what FE families can be built from OWFs in the private-key setting is missing. Therefore, the research direction our work aims to make progress on is:

*For what function families $F$ is private-key FE for $F$ (im)possible from one-way functions?*

In this work, we take the first step in this research direction. In particular, we consider the inner product functionality [2], where for a modulus $q$ and dimension $n$, a function from the family is associated with $\mathbf{y} \in \mathbb{Z}_q^n$ and is defined as $f_{\mathbf{y}}(\mathbf{x}) = \langle \mathbf{x}, \mathbf{y} \rangle$ for all $\mathbf{x} \in \mathbb{Z}_q^n$. Inner-product is a simple and fundamental operation in both theory and practice, and there is a large body of work on this functionality and variants/extensions, *e.g.* [1–5,7–9,9,12,14,21,22]. This functionality is especially attractive in the *private-key setting* because it can be *function-hiding* [14], namely, the key for $f_{\mathbf{y}}$ can hide $\mathbf{y}$ (to the extent possible given the inner products).However, the only known constructions of private-key IPFE, even without function-hiding, are based on *algebraic* assumptions (*e.g.*, DDH/LWE), starting with the work of [2].[2]

While most positive results focus on IPFE for *restricted* inner-products (*i.e.*, where there is a polynomial number of possible inner-products recovered by decryption), we focus on the *unrestricted setting.* In particular, such a scheme is known from class groups [21]. The unrestricted setting is arguably more natural, as the restricted setting came about as a result of limitations of the proposed constructions, not any external desire for achieving it. This brings us to the main question of this work:

*Is private-key inner-product FE black-box possible from one-way functions?*

[2] Note [2] construct public-key schemes, which trivially imply private-key ones. Later works starting with [14] explicitly address the private-key setting.

We answer this question negatively in this work. The challenge we overcome in answering the above question is that all existing 'combinatorial' techniques employed in all the black-box impossibility results in the public-key setting (*e.g.*, for IBE [18,29] and ABE [26]) completely fail in the private-key setting (see Sects. 1.3 and 2 for further discussions.) Thus, our work departs significantly from prior works and develops new combinatorial and other proof techniques to answer the above question.

## 1.2 Our Results

We prove that building private-key FE for inner-product functions (IPFE) is black-box impossible from OWFs, or more generally from any assumptions that hold relative to a random oracle (RO)—*e.g.*, collision-resistant hash functions (CRHFs), KDM-secure private-key encryption. We stress that our result does *not* require function-hiding for IPFE and holds relative to seemingly minimal formulations of its security.

Technically, we prove our result by showing that private-key inner-product FE (IPFE) cannot be constructed in the information-theoretic random oracle model [27]. Central to our impossibility proofs is a combinatorial lemma such that if proved for a function family, then we will have a black-box impossibility for that function family from ROs. We show that the combinatorial lemma holds for the inner-product functionality using techniques from Fourier analysis and covering problems in linear subspaces. The characterization of this combinatorial lemma and the proof of the lemma for inner products are the main novelties of our work, and will hopefully pave the way for characterizing FE functionalities provably impossible from ROs.

The main motivation behind the above question is to initiate the study of the black-box complexity of private-key functional encryption. Moreover, our work will facilitate future efforts to understand the black-box complexity of variants of IPFE. For example, function-hiding IPFE is so far only known from pairings [14], and we have limited impossibility results for it from lattice assumptions [36,37]. However, we do not have any impossibility results for it in the generic-group model (GGM) without pairings. Some of the challenges that appear in ruling out function-hiding IPFE in the GGM also emerge in our setting, so we are hopeful that our work will also be useful for proving such an impossibility result.

## 1.3 Novelty, Comparison to Prior Work and Open Problems

As mentioned earlier, there are impossibility results for predicate encryption (PE) schemes in the public-key setting [18,26,29]. But all these results crucially make use of the public-key setting, and fail in the private-key setting (*c.f.*, the positive construction of private-key IBE from OWFs). Second, our results concern FE schemes that are of fine-grained access (*i.e.*, that each decryption reveals some partial information about the plaintext), whereas previous impossibility results concern only PE schemes, which are of the all-or-nothing nature.[3] Ruling out

[3] That is, decryption reveals either the entire plaintext or nothing about it.

fine-grained FE schemes present additional challenges, as explained below. For example, for all we know IPFE might be possible from CPA-secure encryption schemes because IPFE is not know to imply any PE schemes that are ruled out from CPA-secure schemes. For instance, we know how to build IPFE from black-box DDH [2], but we have black-box impossibility results for IBE from DDH [31,35,39]. This suggests that building IPFE might be 'easier' than IBE, or than other related PE primitives.

*Private-Key vs Public-Key.* Let us illustrate why at a technical level the PE impossibility results of [18,26,29] fail in the private-key setting. In the public-key version for IBE, encryptions are made under $\mathsf{mpk}$, which can encode at most polynomially-many base public keys (call this Property (*) below), while in the private-key version, encryptions are made under $\mathsf{msk}$, which can potentially encode exponentially-many secret keys. We mentioned this point before. The above public-key impossibility results crucially rely on (*), implying that at most polynomially-many public keys $\mathsf{pk}_1, \ldots, \mathsf{pk}_t$ can be embedded into $\mathsf{mpk}$. Specifically, if one corrupts $q \gg t$ identities and learns their secret keys, one has learned enough trapdoors associated with $\mathsf{pk}_i$ to be able to decrypt for an uncorrupted identity. But this intuition fails in the secret key setting as exemplified earlier.

The results of [26] shows that threshold predicate encryption is black-box impossible from IBE. This work gives an impossibility from IBE, a primitive for which the master secret key can generate exponentially-many trapdoors. But the main difference between [26] and ours is that in [26], too, they crucially rely on the public-key setting, used to argue that a master public key for the threshold FE can encode at most polynomially-many public-keys.

In light of the above, it is an open problem if, and to what extent, the PE impossibility results of [18,26,29] generalize to the secret-key setting. This tension between private-key and public-key concerns both the predicate encryption and our impossibilities for FE.

*Functional Encryption vs Predicate Encryption.* As mentioned earlier, the key difference between FE and PE is the partial decryption vs. full decryption natures of these primitives. Consequently, the security game of FE puts *more restrictions* on the set of keys that the adversary is allowed to corrupt during a successful attack. Hence, the 'combinatorial' techniques employed in the black-box impossibility result must be proven in a more stringent setting obeying such restrictions. We explain the difference between our combinatorial lemma and those of prior works in detail in Sect. 2.2.

*Open Problems.* The main problem left open by our work is to prove black-box impossibilities in the private-key setting for other FE functionalities. One concrete functionality that we were not able to handle is *fuzzy* FE [34]. Here plaintext and secret key vectors are all in $\mathbb{Z}_2^n$ and $F(\mathbf{x}, \mathbf{v}) = 1$ if the Hamming distance between $\mathbf{v}$ and $\mathbf{x}$ is at least $\eta n$ (for some fixed $\eta < 1$), and $F(\mathbf{x}, \mathbf{v}) = 0$, otherwise. Note that each decryption reveals some information about the plaintext $\mathbf{x}$ (because the decryption either outputs zero or one). Just like for our impossibility result, coming up with appropriate combinatorial lemmas and proving them will be the main challenges here.

It would also be interesting to see if and how the black-box impossibility results of [26] for PE extend to the private-key setting. In general, understanding the black-box complexity of PE primitives in the private-key setting is a worthwhile goal.

Finally, as we mentioned, it is an intriguing open problem if we can extend our results to rule out the existence of function-hiding private-key IPFE in the GGM without pairings. Similarly, it would be interesting to extend our results to rule out black-box constructions of *restricted* IPFE from OWFs.

## 2 Technical Overview

A private-key IPFE relative to a random oracle $O$ for vectors in $\mathbb{Z}_q^n$ is given by $\mathcal{E}^O := (\mathrm{KGen}^O, \mathrm{Enc}^O, \mathrm{Dec}^O)$, satisfying the following properties.[4] Let $w = w(\kappa)$ be the length of a master-secret key, where $\kappa$ is the security parameter. The algorithm $\mathrm{KGen}^O(\mathsf{msk}, \mathbf{v})$ outputs a vector secret key $\mathsf{sk}[\mathbf{v}]$. One can use $\mathsf{msk}$ to encrypt a plaintext vector $\mathbf{x}$ as $C \leftarrow \mathrm{Enc}^O(\mathsf{msk}, \mathbf{x})$. Finally, decrypting $C$ using $\mathsf{sk}[\mathbf{v}]$ as $\mathrm{Dec}^O(\mathsf{sk}[\mathbf{v}], C)$ returns $\langle \mathbf{v}, \mathbf{x} \rangle$. ——— We require the following weak notion of indistinguishability security (See Definition 2 for more details.) An adversary submits (non-adaptively, at once) $t$ vectors $\mathbf{v}_1, \ldots, \mathbf{v}_t$, where $t = t(\kappa)$ can be an arbitrarily-large polynomial, as well as a *challenge secret-key vector* $\mathbf{v}^*$. It is required that $\mathbf{v}^* \notin \mathsf{Span}(\mathbf{v}_1, \ldots, \mathbf{v}_t)$, where $\mathsf{Span}()$ denotes the linear span of the corresponding vectors. In response, the adversary receives the secret keys $\{\mathsf{sk}[\mathbf{v}_i]\}_{i \in [t]}$ for these $t$ vectors, as well as $t$ ciphertexts $C_1, \ldots, C_t$ of $t$ random plaintexts $\mathbf{x}_1, \ldots, \mathbf{x}_t$ sampled by the challenger, and $m_1, \ldots, m_t$, formed as follows:

- If the challenge bit $b = 0$, $m_i = \langle \mathbf{v}^*, \mathbf{x}_i \rangle$ for $i \in [t]$.
- If $b = 1$, $m_i \leftarrow \mathbb{Z}_q$ for all $i \in [t]$.

The adversary should be able to guess the value of $b$ only with a probability negligibly greater than 1/2. Note that the adversary is not given the underlying plaintexts $\mathbf{x}_1, \ldots, \mathbf{x}_t$.

*Breaking $\mathcal{E}^O$ relative to ROs.* Let $\mathcal{E}^O := (\mathrm{KGen}^O, \mathrm{Enc}^O, \mathrm{Dec}^O)$ be a candidate IPFE. Here we describe an adversary $\mathsf{Brk}^O$ that makes a polynomial number of queries to $O$, and then analyze its advantage. The attack is based on a polynomial $t(\kappa)$, instantiated later.

The adversary $\mathsf{Brk}^O$ chooses the challenge secret key vector $\mathbf{v}^*$ uniformly at random from $\mathbb{Z}_q^n$, chooses a random $(n-1)$-dimensional subspace $S$ subject to $\mathbf{v}^* \notin S$ and chooses $\mathbf{v}_1, \ldots, \mathbf{v}_t$ uniformly at random from $S$. Let

[4] There is also an additional $\mathrm{Setup}^O$ algorithm, that generates a master secret key $\mathsf{msk}$. But we can remove this algorithm because an $\mathsf{msk}$ can be a uniformly random string from an appropriate space $\{0,1\}^w$, for some $w = w(\kappa)$.

$(\{\mathsf{sk}[\mathbf{v}_i]\}_{i\in[t]}, \{C_i\}_{i\in[t]}, \{m_i\}_{i\in[t]})$ be the variables returned to $\mathsf{Brk}^O$, as per the description of the game above. Also, suppose $\mathbf{x}_i$ is the underlying plaintext vector for $C_i$. The adversary should determine if $m_i$'s are totally random values, or are the inner products of $\mathbf{x}_i$ with $\mathbf{v}^*$.

*Simple Case:* Enc *makes no $O$ queries.* To lay out our main techniques and to point out the challenges, let us make an overly simplifying assumption that Enc makes no queries. This is not a reasonable assumption, but we start with this assumption to describe our main techniques—later, we show how to remove this assumption using our combinatorial lemmas. Equipped with this assumption, we show how to design $\mathsf{Brk}$ to break the IPFE scheme by making a polynomial number of queries, and by making some other computation that takes exponential time but involves no queries. This will be sufficient for a black-box impossibility proof because ROs are OWFs against any adversary that can run in exponential time but which can make at most a polynomial number of queries. In particular, it shows that the adversary $\mathsf{Brk}$, which breaks the IPFE scheme, cannot be used as a black-box to break the one-wayness of the oracle $O$.

Let $\mathsf{S}_g$ be the set of the query-answer (Q-A) pairs made to generate the challenge secret key $\mathsf{sk}[\mathbf{v}^*] \leftarrow \mathrm{KGen}(\mathsf{msk}, \mathbf{v}^*)$ and $\rho$ the bit-length of the description of such a set for any vector secret key. Moreover, let $\mu$ be the length of a vector secret key. The adversary $\mathsf{Brk}^O$ attacks the scheme as follows.

(i) If there exists a secret key $\mathsf{sk} \in \{0,1\}^\mu$ and $\mathsf{S}_g \in \{0,1\}^\rho$ such that the following condition holds, return 0; otherwise, return 1:
   (a) $\mathrm{Dec}^{O'}(\mathsf{sk}, C_h) = m_h$ for all $h \in [t]$, where $O'$ is defined as follows. If the query appears in the set $\mathsf{S}_g$, respond to it accordingly, else, respond with a random value.

Note that $\mathsf{Brk}$ makes no queries to the oracle $O$ at all—simulating the query responses based on $\mathsf{S}_g$ and random values, without invoking $O$ itself. (The fact that $\mathsf{Brk}$ makes no queries is because of the above two simplifying assumptions.) Let us analyze the advantage of $\mathsf{Brk}$.

*Challenge Bit $b = 0$:* Suppose $\mathcal{E}$ is $(1-\alpha)$-correct, meaning that for any oracle $O$, and for any secret-key vector $\mathbf{v}$, the following holds. If we generate $\mathsf{msk}$, $\mathsf{sk}[\mathbf{v}]$ (a secret key for $\mathbf{v}$) and $C$ (a ciphertext for a random plaintext vector $\mathbf{x}$) all at random, the probability that $\langle \mathbf{v}, \mathbf{x}\rangle = \mathrm{Dec}^O(\mathsf{sk}[\mathbf{v}], C)$ is at least $1-\alpha$. (See Definition 1.) We show when $b = 0$, $\mathsf{Brk}$ outputs 0 with probability at least $(1-\alpha)$. To see this, we argue that Condition (i)a will hold with probability at least $(1-\alpha)$ when $\mathsf{sk}$ is set to $\mathsf{sk}[\mathbf{v}^*]$. The non-triviality of this lies in the fact that decryption is performed relative to $O'$, and not relative to $O$ itself. But since Enc makes no queries at all, and since we try all possible $\mathsf{S}_g$—the set of Q-A pairs made to build $\mathsf{sk}[\mathbf{v}^*] \leftarrow \mathrm{KGen}^O(\mathsf{msk}; r)$—had we run everything relative to $O'$ instead (under the same randomness), we would have gotten the same $\mathsf{sk}[\mathbf{v}^*]$ and $C$, and hence $\mathrm{Dec}^{O'}(\mathsf{sk}, C_i)$ should output $m_i$ with the same probability.

*Challenge Bit $b = 1$:* We argue that in this case $\mathsf{Brk}^O$ outputs 1 with all but negligible probability. Fix a secret key $\mathsf{sk}$ and a set $\mathsf{S}_g$. Since $b = 1$, all $m_i$'s are chosen at random, and so the probability that $\mathrm{Dec}^{O'}(\mathsf{sk}, C_h) = m_h$ for all $h \in [t]$ is at most $\frac{1}{q^t}$. By the union bound over all $\mathsf{sk}$, the probability that $\mathsf{Brk}$ mistakenly outputs 0 is at most $\frac{2^{\mu+\rho}}{q^t}$. By choosing $t$ large enough, this probability will become negligible.

$\mathrm{Enc}^O$ *making $O$ queries.* We now show how to lift the assumption, that $\mathrm{Enc}^O$ makes no $O$ queries. Let us re-run the previous description of $\mathsf{Brk}$ (Step (i)a) to see where it fails when *Enc* makes $O$ queries. Our analysis for $b = 0$ fails: because a query made during $\mathrm{Dec}^{O'}(\mathsf{sk}[\mathbf{v}^*], C_h)$ might be one that was asked before when generating $C_h \leftarrow \mathrm{Enc}^O(\mathsf{msk}, \mathbf{x}_h)$, and if $O'$ replies to it randomly, we will have an inconsistency (i.e., we cannot argue the simulated decryption $\mathrm{Dec}^{O'}(\mathsf{sk}[\mathbf{v}^*], C_h)$ outputs $m_h$ when $b = 0$). Letting $\mathsf{Q}_h$ be the set Q-A pairs made during the generation of $C_h \leftarrow \mathrm{Enc}^O(\mathsf{msk}, \mathbf{x}_i)$, we should somehow learn all those Q-A pairs in $\mathsf{Q}_h$ that also appear during $\mathrm{Dec}^O(\mathsf{sk}[\mathbf{v}^*], C_h)$. Fix the index $h$ below. Consider the following strategy for decrypting the $h$th ciphertext.

1. For all $i \in [t]$, run $\mathrm{Dec}^O(\mathsf{sk}[\mathbf{v}_i], C_h)$, and record all Q-A pairs in the set $\mathsf{S}_g$.
2. Perform Step (i) from before.

The intuition is that if $t$ is large enough and if a query is to appear during $\mathrm{Dec}^O(\mathsf{sk}[\mathbf{v}^*], C_h)$, then it should have also appeared during one of the $\mathrm{Dec}^O(\mathsf{sk}[\mathbf{v}_i], C_h)$ executions in Step 1, except with small probability. However, proving this leads to the following challenge: the vector $\mathbf{v}^*$ is sampled from the entire space $\mathbb{Z}_q^n$, while $\mathbf{v}_i$ vectors are sampled from an $(n-1)$-dimensional subspace, leading to $\mathsf{sk}[\mathbf{v}^*]$ having a different distribution from $\mathsf{sk}[\mathbf{v}_i]$. The above statement would have been easy to prove if all of $\mathbf{v}_i$'s were picked from the entire space $\mathbb{Z}_q^n$, but that is not the case here.

The above challenge is about a *covering* problem. Suppose $\ell$ queries are made during $C \leftarrow \mathrm{Enc}^O(\mathsf{msk}, \mathbf{x})$. (We replace $C_h$ with $C$ for better readability.) Let us number these queries as $1, \ldots, \ell$. Consider a function $F\colon \mathbb{Z}_q^n \to 2^{[\ell]}$, where $j \in F(\mathbf{y})$ if Query $j$ appears during the decryption of $\mathrm{Dec}^O(\mathsf{sk}[\mathbf{y}], C)$, where $\mathsf{sk}[\mathbf{y}] \leftarrow \mathrm{KGen}^O(\mathsf{msk}, \mathbf{y})$. Here $2^{[\ell]}$ denotes the set of subsets of $[\ell]$. We would like to prove that with high probability $F(\mathbf{v}^*) \subseteq \cup_i F(\mathbf{v}_i)$, where $\mathbf{v}^*$ is the challenge secret key vector, and $\mathbf{v}_i$'s are the vectors from the $(n-1)$-dimensional subspace, whose secret keys are given to $\mathsf{Brk}$.

We abstract out the above problem as a combinatorial lemma.

**Lemma 1 (Combinatorial Lemma).** *Let $n = n(\kappa) \geqslant 3$ be such that $\frac{1}{q^n}$ is negligible. Let $\ell = \ell(\kappa)$ be an arbitrary polynomial. Let $F : \mathbb{Z}_q^n \to 2^{[\ell]}$ be an arbitrary function, assigning a subset of $[\ell]$ to every vector. Then, for all large enough polynomial values of $t = t(\kappa)$, with overwhelming probability*

$$F(y^*) \subseteq \bigcup_{i=1}^{t} F(y_i), \tag{1}$$

*where $y^* \leftarrow \mathbb{Z}_q^n$, and $y_1, \ldots, y_t$ are sampled as follow: sample an $(n-1)$-dimensional subspace $V \subseteq \mathbb{Z}_q^n$ uniformly at random conditioned on $y^* \notin V$ and then sample $y_1, \ldots, y_t$ uniformly at random from $V$.*

### 2.1 Proof of the Combinatorial Lemma

In this overview we will focus on the case $q = 2$. In the main part of the paper, we will prove the same result for any prime $q$ with slightly looser bounds.

*A simpler problem.* Before describing how to prove this result, we focus on a related but simpler problem: We show that there is no polynomial $\ell$ that satisfies the above condition is violated in a *worst-case sense*. In other words, there exists no polynomial $\ell$ and $F : \mathbb{Z}_2^n \to 2^{[\ell]}$ such that for every $\mathbf{y}^*$ and every $(n-1)$-dimensional subspace $V$ with $\mathbf{y}^* \notin V$ we have

$$F(\mathbf{y}^*) \not\subseteq \bigcup_{\mathbf{y} \in V} F(\mathbf{y}). \tag{2}$$

We will refer in the following to the set $[\ell]$ as colors and to $F$ as a coloring of the vectors in $\mathbb{Z}_q^n$. A simple coloring strategy that satisfies Condition (2) is to set $\ell = 2^n - 1$ and assign each non-zero vector one individual color. Another simple strategy is to also use $\ell = 2^n - 1$ and pick for every $(n-1)$-dimensional subspace one new individual color and add it to the colorings of each vector *not* contained in the subspace.

We show that using $\ell \geqslant 2^n - 1$ colors is necessary to satisfy Condition (2) by a double-counting argument on the vectors $\mathbf{y}^*$ and $(n-1)$-dimensional subspaces $V$ with $\mathbf{y}^* \notin V$.

For a vector $\mathbf{y}^*$ and an $(n-1)$-dimensional subspace $V$ we say the color $\mathsf{cl}$ is *useful* for $(\mathbf{y}^*, V)$ if $\mathbf{y}^* \notin V$, $\mathsf{cl} \in F(\mathbf{y}^*)$ and $\mathsf{cl} \notin \bigcup_{\mathbf{y} \in V} F(\mathbf{y})$. Condition (2) says for every combination of $\mathbf{y}^*$ and $V$ we need at least one useful color.

In $\mathbb{Z}_2^n$, each $(n-1)$-dimensional subspace is uniquely defined by a non-zero vector; *i.e.*, the subspace that is orthogonal to the underlying non-zero vector. Thus, we have $2^n - 1$ different $(n-1)$-dimensional subspaces. Each subpsace $V$ has $2^{n-1}$ different vectors $\mathbf{y}^*$ not contained in $V$. Thus, the number of triples $(V, \mathbf{y}^*, \mathsf{cl})$ where $\mathsf{cl}$ is useful for $(\mathbf{y}^*, V)$ is at least $(2^n - 1)2^{n-1}$. We now use another way of counting to argue the number of such triples is at most $\ell 2^{n-1}$, implying $\ell \geqslant 2^n - 1$.

Fix a color $\mathsf{cl}$. We argue that the color $\mathsf{cl}$ is useful for at most $2^{n-1}$ different combinations $(\mathbf{y}^*, V)$, implying the number of such triples as above is at most $\ell 2^{n-1}$. Let $\mathcal{S}$ be the set of all the $(n-1)$-dimensional subspaces $V$ such that there exists at least one $\mathbf{y}^\star$ with $\mathsf{cl}$ being useful for $(\mathbf{y}^*, V)$. Then, no vector in $U := \bigcup_{V \in \mathcal{S}} V$ has the color $\mathsf{cl}$ and, in the worst case, every vector in $\mathbb{Z}_2^n \setminus U$ has the color $\mathsf{cl}$. Each of the subspaces $V \in \mathcal{S}$ can be described by one linear equation, i.e. for each $V \in \mathcal{S}$ we can pick a vector $\mathbf{x}_i$ such that $V = \{\mathbf{v} \mid \langle \mathbf{x}_i, \mathbf{v} \rangle = 0\}$, because $V$ is $(n-1)$-dimensional. When we have $t := |\mathcal{S}|$ subspaces, we get at least $d \geqslant \lceil \log_2(t) \rceil$ linear independent equations. Let $\mathbf{x}_1, \ldots, \mathbf{x}_d$ be the vectors representing these linear independent equations. In order for a vector $\mathbf{y}^*$ to be in $\mathbb{Z}_2^n \setminus U$, it is necessary that

$$\langle \mathbf{y}^*, \mathbf{x}_1 \rangle \neq 0 \wedge \cdots \wedge \langle \mathbf{y}^*, \mathbf{x}_d \rangle \neq 0.$$

Since we are working over $\mathbb{Z}_2$, this is equivalent to

$$\langle \mathbf{y}^*, \mathbf{x}_1 \rangle = 1 \wedge \cdots \wedge \langle \mathbf{y}^*, \mathbf{x}_d \rangle = 1.$$

This version shows us that $\mathbb{Z}_2^n \setminus U$ is contained in an $(n-d)$-dimensional affine subspace of $\mathbb{Z}_2^n$ and thus there can be at most $2^{n-d}$ vectors in $\mathbb{Z}_2^n \setminus U$. Thus the number of combinations for which a color can be useful is at most $t \cdot 2^{n-d} \leqslant 2^d \cdot 2^{n-d} = 2^n$. Since we need for each of the combination at least one useful color, we need at least

$$\ell \geqslant \frac{(2^n-1)2^{n-1}}{2^n} = \frac{(2^n-1)}{2} \text{ colors.}$$

In the main body we enhance this argument and observe that not every combination of $\mathbf{x}_1, \ldots, \mathbf{x}_t$ is allowed. Concretely, we show that they must be a sum-free set. This can be used to improve the bound on the dimension to $d \geqslant \lceil \log_2(t) \rceil + 1$, which then implies $\ell \geqslant 2^n - 1$.

We present a formal proof for the case $\mathbb{Z}_2$ in the full version. A formal proof for $\mathbb{Z}_q$ is given in Sect. 4.

*The Combinatorial Lemma over $\mathbb{Z}_2$.* Next, we sketch how to generalize the worst-case arguments to the average case, as required by Lemma 1. In this setting we are restricted to use $\ell = \mathsf{poly}(\kappa)$ different colors and we have to argue that we will hit with noticeable probability a vector $\mathbf{y}^*$ and an $(n-1)$-dimensional subspace $V \not\ni \mathbf{y}^*$ such that every color that appears in $\mathbf{y}^*$ also appears in many vectors in $V$. The latter condition will ensure that if we sample $\mathbf{y}_1, \ldots, \mathbf{y}_t$ from $V$ (for a large enough polynomial $t$), they will satisfy Condition (1) with high probability.

We assume here that every color in $F(\mathbf{y}^*)$ is used often in the whole space $\mathbb{Z}_2$ (concretely, in at least $2^{n-1}/\ell$ vectors). With noticeable probability (here: at least $1/2$), this is satisfied when picking $\mathbf{y}^*$ uniformly at random.

We then prove that a color that appears that often in the whole space must also appear often (concretely in $2^{n-3}/\ell$ vectors) in all but negligible many $(n-1)$-dimensional subspaces. The argument for this is a generalization of the simpler problem we described above and uses the Fourier transform for hypercubes.

We present a formal proof for the Combinatorial Lemma over $\mathbb{Z}_2$ in the full version.

*On Generalizing to any Prime Modulus $q$.* Generalizing our results to arbitrary prime moduli $q$ is non-trivial. This can be best seen when focusing on the simpler problem we described in the beginning. Recall that there we needed to count the number of vectors that are non-orthogonal to every vector in a fixed set of vectors. We used there that

$$\langle \mathbf{y}^*, \mathbf{x}_1 \rangle \neq 0 \wedge \cdots \wedge \langle \mathbf{y}^*, \mathbf{x}_d \rangle \neq 0$$

is equivalent to

$$\langle \mathbf{y}^*, \mathbf{x}_1 \rangle = 1 \wedge \cdots \wedge \langle \mathbf{y}^*, \mathbf{x}_d \rangle = 1$$

and thus the solutions are an $(n-d)$-dimensional subspace. Over $\mathbb{Z}_q$, it seems that this problem does not have a nice algebraic structure. Of course, we could use

$$\langle \mathbf{y}^*, \mathbf{x}_1 \rangle \in \mathbb{Z}_q^* \wedge \cdots \wedge \langle \mathbf{y}^*, \mathbf{x}_d \rangle \in \mathbb{Z}_q^* \tag{3}$$

instead. However, this makes the resulting bound worse by a factor of $(q-1)^d$ and the result is no longer useful. The reason the (almost) tight bound from before becomes here very loose is that many of the vectors satisfying Eq. (3) will in fact be orthogonal to one of the linear combinations of $\mathbf{x}_1, \ldots, \mathbf{x}_d$.[5] We prove results for $\mathbb{Z}_q$ that are similar to $\mathbb{Z}_2$, but use different techniques. The main technique is to use the Cauchy-Schwartz inequality to get a lower bound on the "overlap" of many $(n-1)$-dimensional subspaces, which then again allows us to argue that if many vectors are colored in the whole subspace $\mathbb{Z}_q^n$, in almost all $(n-1)$-dimensional subspaces many vectors are colored.

We present a formal proof for the Combinatorial Lemma over $\mathbb{Z}_q$ in Sect. 4.

## 2.2 Comparison with Prior Combinatorial Lemmas

Katz and Yerukhimovich [29] generalize the results of Boneh et al. [18] to rule out a broader class of PE primitives from trapdoor permutations. Here is a simplified version of the combinatorial lemma of [29, Lemma 1].

If there exists predicates $f_1, \ldots, f_q$ together with attributes $A_1, \ldots, A_q$ such that for all $i$: $f_i(A_i) = 1$ but $f_{i+1}(A_i) = \ldots = f_q(A_i) = 0$, then they show an impossibility. The idea is to corrupt all they keys for $(i+1, \ldots, q)$ and use the info to decrypt for $A_i$. And [29] shows that certain predicates (e.g., IBE, broadcast encryption) satisfy this property, using the Pigeonhole principle. The above property can be established also for zero inner-product encryption (where the predicate is satisfied iff the inner product is zero). Let $f_i = (id_i, 1)$ and let $A_i = (-1, id_i)$, allowing one to rule out public-key inner-product predicate encryption from PKE.

However, for FE, the combinatorial lemma becomes much more complicated, both in terms of its description and also establishing it for a functionality. The main reason is: under PE, only certain decryptions reveal information about the plaintext (an all-or-nothing property), whereas under FE, all decryptions do. For instance, extending the above combinatorics, established for zero inner-product encryption as above, to IPFE faces the following challenge: the vector $A_i$ will be the whole plaintext vector, and we must make sure for all $i$, the vector $f_i$ is not in the span of $(f_{i+1}, \ldots, f_q)$. This will limit how large $q$ can become, while being able to make $q$ arbitrarily large was crucial in the arguments of [29]. Thus, we need to come up with a more specialized combinatorial lemma. And we cannot establish the resulting combinatorial lemma using simple combinatorial techniques anymore (*e.g.*, the pigeonhole principle as in [18, 29]) and need more advanced tools.

[5] This also happens with $q = 2$ for sums of $\mathbf{x}_1, \ldots, \mathbf{x}_d$ with an even number of summands. However, this costs us there only a factor of 2 and we manage to get ride of this factor by using that $\mathbf{x}_1, \ldots, \mathbf{x}_t$ has to be sum-free.

## 3 Preliminaries

*Notation.* We use $\kappa$ to denote the security parameter. We use $[n] := \{1, \dots, n\}$ for $n \in \mathbb{N}$ and $2^S$ to denote the power set of $S$.

**Lemma 2.** *Let $X_1, \dots, X_{t+1}$ be independent, Bernoulli random variables, where $\Pr[X_i = 1] = p$, for all $i \leqslant t + 1$. Then*

$$\Pr[X_1 = 0 \wedge \cdots \wedge X_t = 0 \wedge X_{t+1} = 1] \leqslant \frac{1}{t} .$$

We give the definitions for private-key IPFE schemes relative to an oracle. As we mentioned, we consider *unrestricted* IPFE where there is no restriction on the number of possible inner-products recovered by decryption. We do not explicitly mention this point in the remainder of the paper.

A private-key IPFE scheme $\mathcal{E}^O = (\mathrm{KGen}^O, \mathrm{Enc}^O, \mathrm{Dec}^O)$ is given by three algorithms. We assume without loss of generality that a master secret key is chosen uniformly at random from $\{0,1\}^w$, for some $w := w(\kappa)$. Moreover, we assume $\mathbb{Z}_q$ is the underlying field.

- $\mathrm{KGen}^O(\mathsf{msk}, \mathbf{v})$: On input a master secret key $\mathsf{msk}$ and a vector $\mathbf{v}$, the key generation algorithm outputs a secret key $\mathsf{sk}[\mathbf{v}]$.
- $\mathrm{Enc}^O(\mathsf{msk}, \mathbf{x})$: On input a master secret key $\mathsf{msk}$ and a vector $\mathbf{x}$, the encryption algorithm outputs a ciphertext $C$.
- $\mathrm{Dec}^O(\mathsf{sk}[\mathbf{v}], C)$: On input a secret key $\mathsf{sk}[\mathbf{v}]$ and a ciphertext $C$, the decryption algorithm outputs $y \in \mathbb{Z}_q$.

We require the following properties.

**Definition 1 (IPFE Correctness).** *Fix an oracle $O$. We say an oracle-aided IPFE $(\mathrm{KGen}^O, \mathrm{Enc}^O, \mathrm{Dec}^O)$ is $\nu$-correct for dimension $n$ relative to $O$, if for any key vector $\mathbf{v} \in \mathbb{Z}_q^n$, the following experiment outputs one with probability at least $\nu$. Sample $\mathsf{msk}$ uniformly at random, $\mathsf{sk}[\mathbf{v}] \leftarrow \mathrm{KGen}^O(\mathsf{msk}, \mathbf{v})$, $\mathbf{x} \leftarrow \mathbb{Z}_q^n$ and $C \leftarrow \mathrm{Enc}^O(\mathsf{msk}, \mathbf{x})$. The experiment outputs one if $\langle \mathbf{v}, \mathbf{x} \rangle = \mathrm{Dec}^O(\mathsf{sk}[\mathbf{v}], C)$.*

Next, we give a definition for selective-security for a private-key IPFE below.

**Definition 2 (IPFE Security).** *We work with a selective-security definition for IPFE. Fix a dimension $n$. The adversary designates a challenge secret-key vector $\mathbf{v}^*$ (sent to the challenger) and the adversary can make two types of queries, as follows, but all the adversary's queries should be made non-adaptively at once. The challenger starts by sampling a master secret key $\mathsf{msk}$ and a challenge bit $b \leftarrow \{0,1\}$ uniformly at random.*

- *Key Queries: The adversary submits a vector $\mathbf{v}$ and receives a secret key for $v \leftarrow \mathrm{KGen}(\mathsf{msk}, \mathbf{v})$.*
- *Inner-Product Queries: Upon calling this oracle, the challenger samples $\mathbf{w} \leftarrow \mathbb{Z}_q^n$ and returns $(\mathrm{Enc}(\mathsf{msk}, \mathbf{w}), m^*)$ to the adversary, where $m^* = \langle \mathbf{w}, \mathbf{v}^* \rangle$ if $b = 0$, and $m^*$ is chosen freshly for each query if $b = 1$.*

*We say an adversary $\mathcal{A}$ is* admissible *if $\mathcal{A}$ makes all its queries non-adaptively at once, and if $\mathbf{v}^* \notin \mathsf{Span}(\mathbf{v}_1, \ldots, \mathbf{v}_t)$, where $\mathbf{v}_1, \ldots, \mathbf{v}_t$ are all the adversary's key queries. We say a private-key IPFE is selectively secure if any admissible PPT adversary $\mathcal{A}$ has at most $1/2 + negl(\kappa)$ advantage in guessing the value of $b$.*

Our IPFE definition is strictly weaker than standard ones [2], making our impossibility result stronger. Specifically, ours requires security only for random plaintext vectors, while the standard ones concern *all* vectors. Our definition is implied by the standard definitions via a simple hybrid argument. To see why it is strictly weaker, change a given scheme $\mathcal{E} = (\text{KGen}, \text{Enc}, \text{Dec})$ meeting the standard definitions into a scheme $\mathcal{E}' = (\text{KGen}', \text{Enc}', \text{Dec}')$ so that $\text{Enc}'(\mathsf{msk}, \mathbf{e}_1)$ outputs $\text{Enc}(\mathsf{msk}, \mathbf{e}_1) \| \mathbf{e}_1$, where $\mathbf{e}_1$ is the first unit vector, and $\text{Dec}'$ is defined accordingly. The rest of the scheme remains the same. The new scheme satisfies our notion but not the standard ones. Intuitively, it satisfies ours because we only need security with respect to random plaintext vectors.

## 4 The Combinatorial Problem over $\mathbb{Z}_q$

**Lemma 3.** *Fix $n = n(\kappa)$ and suppose $q^{-n} \in negl(\kappa)$ and $n \geqslant 3$. Let $\ell = \mathsf{poly}(\kappa)$, $q$ be a prime number and $F : \mathbb{Z}_q^n \to 2^{[\ell]}$. Fix a constant $c$. Then, there exists a polynomial $t = t(\kappa)$ such that with probability at least $1 - \kappa^{-c}$*

$$F(\mathbf{y}^*) \subseteq \bigcup_{i=1}^{t} F(\mathbf{y}_i),$$

*where $\mathbf{y}^* \leftarrow \mathbb{Z}_q^n$, and we sample a random $(n-1)$-dimensional subspace $V$ subject to $\mathbf{y}^* \notin V$ and we sample $\mathbf{y}_1, \ldots, \mathbf{y}_t$ all uniformly at random from $V$.*

The following theorem is the key tool in proving the above Lemma.

**Theorem 1.** *For any subset $S \subseteq \mathbb{Z}_q^n$ with $|S| = p \cdot q^n$, there exists at most $4q/p^2$ $(n-1)$-dimensional subspaces $h$ of $\mathbb{Z}_q^n$ with*

$$|S \cap h| \leqslant \frac{|S|}{2q} = \frac{p}{2} q^{n-1}.$$

*Proof (of Lemma 3 using Theorem 1).* Fix a mapping $F : \mathbb{Z}_q^n \to 2^{[\ell]}$. We will refer to the set $[\ell]$ as the set of colors and say that $F$ colors each vector of $\mathbb{Z}_q^n$.

For a color $\mathsf{cl}$, let $F^{-1}(\mathsf{cl}) = \{\mathbf{y} \mid \mathsf{cl} \in F(\mathbf{y})\}$. We say that a color $\mathsf{cl}$ is $p$-heavy if $|F^{-1}(\mathsf{cl})| \geqslant p \cdot q^n$. For $V \subseteq \mathbb{Z}_q^n$, we say that $\mathsf{cl}$ is $p$-heavy in $V$ if $|F^{-1}(\mathsf{cl}) \cap V| \geqslant p \cdot |V|$. We use the heaviness threshold

$$p = \frac{1}{2\ell\kappa^c}.$$

For uniformly random $\mathbf{y}^*$ the probability that $F(\mathbf{y}^*)$ contains a non-$p$-heavy color is less than

$$\ell \cdot \frac{1}{2\ell\kappa^c} = \frac{1}{2\kappa^c}.$$

The rest of the proof assumes all colors in $F(\mathbf{y}^*)$ are $p$-heavy.

Now, for a uniformly random subspace $V^*$ conditioned on $\mathbf{y}^* \notin V^*$ we can claim that, with overwhelming probability, all colors $\mathsf{cl} \in F(\mathbf{y}^*)$ are also $(p/2)$-heavy in $V^*$. This is because applying Theorem 1 with $S := F^{-1}(\mathsf{cl})$ shows that if $\mathsf{cl}$ is $p$-heavy in the entire space, there exist at most $\frac{4q}{p^2}$ $(n-1)$-dim subspaces where $\mathsf{cl}$ is not $p/2$-heavy in those subspaces. Here we are using the fact that an $(n-1)$-dim subspace has $q^{n-1}$ elements. Applying the union bound for all colors in $F(\mathbf{y}^*)$ shows that the number of $(n-1)$-dimensional subspaces $S$ where at least one color of $F(\mathbf{y}^*)$ is not $(p/2)$-heavy in $S$ ("bad subspaces") is at most

$$|F(\mathbf{y}^*)|\frac{4q}{p^2} \leqslant \frac{4\ell q}{p^2} = 16\ell^3 q\kappa^{2c}.$$

The number of $(n-1)$-dimensional subspaces containing $\mathbf{y}^*$ is $\frac{q^{n-1}-1}{q-1}$, because each such subspace can be identified with an $n-2$-dimensional subspace in the $n-1$-dimensional quotient space $(\mathbb{Z}_q^n)_{/\langle \mathbf{y}^* \rangle}$. Thus, the total number of $(n-1)$-dimensional subspaces not containing $\mathbf{y}^*$ is $\frac{q^n - q^{n-1}}{q-1}$. Hence, by sampling one of these subspaces uniformly at random, we hit a bad subspace with probability

$$\frac{16\ell^3 q\kappa^{2c}(q-1)}{q^n - q^{n-1}} = \frac{16\ell^3 \kappa^{2c}(q-1)}{q^{n-1} - q^{n-2}},$$

which is negligible in $\kappa$ since $n \geqslant 3$ and $q^{-n}$ is negligible.

We now analyze the probability of $F(\mathbf{y}^*) \subseteq \bigcup_{i=1}^{t} F(\mathbf{y}_i)$ when setting $t = \lceil 2\kappa/p \rceil$. In the following, the probability is taken over the choice of $\mathbf{y}^*, \mathbf{y}_1, \ldots, \mathbf{y}_t$.

$$\begin{aligned}\Pr[\forall \mathsf{cl} \in F(\mathbf{y}^*)\ \exists i \in [t] : \mathsf{cl} \in F(y_i)] &= 1 - \Pr[\exists \mathsf{cl} \in F(\mathbf{y}^*) \forall i \in [t] : \mathsf{cl} \notin F(y_i)] \\ &\geqslant 1 - \ell \max_{\mathsf{cl} \in F(\mathbf{y}^*)} \Pr[\forall i \in [t] : \mathsf{cl} \notin F(y_i)] \\ &\geqslant 1 - \ell \left(1 - \frac{p}{2}\right)^t \geqslant 1 - \ell e^{-t \cdot \frac{p}{2}} = 1 - \ell e^{-\kappa}.\end{aligned}$$

The last inequality follows from $1 - \frac{p}{2} \leqslant e^{-\frac{p}{2}}$ (the Bernoulli inequality) and taking both sides to the $t$-th power. $\square$

*Proof (of Theorem 1).* Bennett proved an existence-only version of Theorem 1 [13, Lemma 4.1]. The following proof follows Bennett's ideas.

Let $H$ be the set of $(n-1)$-dimensional subspaces $h$ of $\mathbb{Z}_q^n$ with

$$|S \cap h| \leqslant \frac{|S|}{2q}.$$

Fix a bijection $\phi : [q^n] \to \mathbb{Z}_q^n$ and define the vectors $\mathbf{u}, \mathbf{v} \in \mathbb{R}^{(q^n)}$ via

$$\mathbf{u}_i = \begin{cases} 1 & \text{if } \phi(i) \notin S \\ 0 & \text{otherwise} \end{cases} \quad \text{and} \quad \mathbf{v}_i = |\{h \in H \mid \phi(i) \in h\}|.$$

For a set $Y \subseteq \mathbb{Z}_q^n$ we use $\chi_Y$ to denote the characteristic function of $Y$.

We get

$$\langle \mathbf{u}, \mathbf{u} \rangle = \sum_{\mathbf{x} \notin S} 1^2 = q^n - |S| = q^n(1-p)$$

$$\langle \mathbf{v}, \mathbf{v} \rangle = \sum_{\mathbf{x} \in \mathbb{Z}_q^n} \left( \sum_{h \in H} \chi_h(\mathbf{x}) \right)^2 = \sum_{h_1, h_2 \in H} \sum_{\mathbf{x} \in \mathbb{Z}_q^n} \chi_{h_1}(\mathbf{x}) \chi_{h_2}(\mathbf{x}) \leqslant q^{n-1}|H| + q^{n-2}|H|^2.$$

In the last inequality we use that $\sum_{\mathbf{x} \in \mathbb{Z}_q^n} \chi_h(\mathbf{x}) \chi_h(\mathbf{x}) = q^{n-1}$ and for $h_1 \neq h_2$ we have $\sum_{\mathbf{x} \in \mathbb{Z}_q^n} \chi_{h_1}(\mathbf{x}) \chi_{h_2}(\mathbf{x}) \leqslant q^{n-2}$.

Each $h \in H$ has by definition $|S \cap h| \leqslant \frac{|S|}{2q}$ and thus $|(\mathbb{Z}_q^n \setminus S) \cap h| = q^{n-1} - |S \cap h| \geqslant q^{n-1} - \frac{|S|}{2q}$ which gives us

$$\langle \mathbf{u}, \mathbf{v} \rangle = \sum_{\mathbf{x} \notin S, h \in H} \chi_h(\mathbf{x}) \geqslant |H| \left( q^{n-1} - \frac{|S|}{2q} \right) = |H| q^{n-1} \left( 1 - \frac{p}{2} \right).$$

Applying the Cauchy-Schwartz inequality to the vectors $\mathbf{u}$ and $\mathbf{v}$ gives us $\langle \mathbf{u}, \mathbf{v} \rangle^2 \leqslant \langle \mathbf{u}, \mathbf{u} \rangle \cdot \langle \mathbf{v}, \mathbf{v} \rangle$. Plugging in the bounds from above leads to the following inequality

$$\begin{aligned}
& |H|^2 q^{2n-2} \left(1 - \frac{p}{2}\right)^2 \leqslant q^n(1-p)(q^{n-1}|H| + q^{n-2}|H|^2) \\
\iff \quad & \frac{(1-p)}{q^{n-2}\left(1 - \frac{p}{2}\right)^2} \geqslant \frac{|H|^2}{(q^{n-1}|H| + q^{n-2}|H|^2)} = \frac{|H|}{q^{n-2}(q + |H|)} \\
\iff \quad & \frac{(1-p)}{\left(1 - \frac{p}{2}\right)^2} \geqslant \frac{|H|}{(q + |H|)} \\
\iff \quad & 1 - \frac{(1-p)}{\left(1 - \frac{p}{2}\right)^2} \leqslant 1 - \frac{|H|}{(q + |H|)} = \frac{q}{(q + |H|)} \leqslant \frac{q}{|H|} \\
\iff \quad & |H| \leqslant \frac{q}{1 - \frac{(1-p)}{\left(1 - \frac{p}{2}\right)^2}} = \frac{q\left(1 - \frac{p}{2}\right)^2}{\left(1 - \frac{p}{2}\right)^2 - (1-p)} = \frac{4q\left(1 - \frac{p}{2}\right)^2}{p^2} \\
\iff \quad & |H| \leqslant \frac{4q}{p^2}\,.
\end{aligned}$$

□

## 5 Separating IPFE from OWFs

The definition below gives a procedure that allows one to sample a random vector $\mathbf{v}^*$ together with $t$ random vectors from a random $(n-1)$-dimensional subspace of $\mathbb{Z}_q^n$ which does not span $\mathbf{v}^*$.

**Definition 3 (Sampling Spanning Vectors).** *The procedure* $(\mathbf{v}_1, \ldots, \mathbf{v}_t, \mathbf{v}^*) \leftarrow \text{SubSpcSamp}(\mathbb{Z}_q^n, t)$ *works as follows. Sample a random vector* $\mathbf{v}^* \leftarrow \mathbb{Z}_q^n$*, and sample a random* $(n-1)$*-dimensional subspace* $\mathsf{S}$ *of* $\mathbb{Z}_q^n$ *subject to* $\mathbf{v}^* \notin S$*. Sample* $\mathbf{v}_1, \ldots, \mathbf{v}_t$ *uniformly at random from* $\mathsf{S}$*. The sampling procedure of* SubSpcSamp *can be performed in* $\mathsf{poly}(n, t, \log q)$ *time.*

*Description of the Attack.* Let $\mathcal{E}^O := (\text{KGen}^O, \text{Enc}^O, \text{Dec}^O)$ be a candidate IPFE construction for vectors in $\mathbb{Z}_q^n$. Assume without loss of generality that $\mathcal{E}^O$ is $(1 - \frac{1}{2^\kappa})$-correct.[6] Our goal is to remove $O$ queries from the decryption algorithm $\text{Dec}^O$, while impacting correctness and security only minimally. In order to do this, if one knows the set $Q_s$ of all the Q-A pairs asked during the generation of a secret key $\mathsf{sk}[\mathbf{v}]$ and the set $Q_e$ of all the Q-A pairs formed to generate a ciphertext $C$, then one can remove $O$ queries from $\text{Dec}(\mathsf{sk}[\mathbf{v}], C)$ as follows: if an issued query appears in $Q_s \cup Q_e$, reply to it accordingly; else, reply with a random response, without calling $O$. In fact, this argument still holds if we just know the subset $Q_e \cap Q_d$, where $Q_d$ is the set of Q-A pairs that appear during $\text{Dec}(\mathsf{sk}[\mathbf{v}], C)$.

The adversary will then decrypt all the ciphertexts obtained via the inner-product queries with all secret keys it received and store all Q-A pairs that appeared during this process in a set $\mathsf{L}$. The combinatorial Lemma from the previous section guarantees that with high probability $Q_e \cap Q_d \subseteq \mathsf{L}$, if the number of keys and ciphertexts is large enough.

Finally, the adversary makes a brute-force search over $\mathsf{sk}[\mathbf{v}^*]$ and the set $Q_s$ and check each candidate by decrypting all challenge ciphertexts without asking any queries.

*The Attack in Detail.*

**Attack 2** *Let* $\mathcal{E}^O := (\text{Setup}, \text{KGen}^O, \text{Enc}^O, \text{Dec}^O)$ *be an IPFE. We give a polynomial query adversary* $\mathsf{Brk}^O$ *which breaks the security of* $\mathcal{E}^O$*.*

*Parameters. The adversary's algorithm is based on integers* $t$ *and* $\eta$*, instantiated later. Also, let* $\mu := \mu(\kappa, n)$ *be the bit size of a secret key, generated by* $\text{KGen}^O(\mathsf{msk}, *)$ *and* $\rho := \rho(\kappa, n)$ *the bit size of a the Q-A pairs of all queries made during secret key generation.*[7]

*Phase 1: Corrupting Keys and Setting Up the Challenge.*

1. *Sample* $(\mathbf{v}_1, \ldots, \mathbf{v}_t, \mathbf{v}^*) \leftarrow \text{SubSpcSamp}(\mathbb{Z}_q^n, t)$*. The vector* $\mathbf{v}^*$ *will be the challenge secret-key vector.*
2. *For all* $i \in [t]$*, make a key query* $\mathbf{v}_i$ *to receive* $\mathsf{sk}[\mathbf{v}_i]$*.*
3. *For all* $i \in [\eta]$ *make an inner-product query to receive* $(C_i, m_i)$*, where recall that* $m_i \in \mathbb{Z}_q$*.*

[6] If the scheme is $1/2 + \frac{1}{\mathsf{poly}(\kappa)}$ correct, we can boost its correctness all the way up to $(1 - \frac{1}{2^\kappa})$ by encrypting the vector many times and taking the majority during decryption..

[7] The assumption that the length of a secret key and and the Q-A pairs is a fixed function of $\kappa$ and $n$ is without loss of generality.

*Phase 2: Learning Important Decryption Queries.*

1. *Let* $\mathsf{L} := \emptyset$. *For all* $i \in [t]$ *and all* $j \in [\eta]$, *execute* $\mathrm{Dec}^{O}(\mathsf{sk}[\mathbf{v}_i], C_j)$ *and add all the Q-A pairs to the set* $\mathsf{L}$.

*Phase 3: Leveraging the Set of Learned Q-A Pairs to Decrypt. In this phase,* $\mathsf{Brk}$ *uses the set* $\mathsf{L}$ *to break the security game. In this phase,* $\mathsf{Brk}$ *does not make any queries to* $O$.

1. *If there exists* $\mathsf{sk} \in \{0,1\}^{\mu}$ *and a set of Q-A pairs* $Q_s \in \{0,1\}^{\rho}$ *such that for all* $h \in [\eta]$, $\mathrm{Dec}^{O'}(\mathsf{sk}, C_i) = m_i$, *where* $O'$ *is a random oracle sampled uniformly at random on-the-fly subject to being consistent with* $\mathsf{L}$ *and* $Q_s$, *return* 0; *else, return 1.*

We now show how to set the parameters, and then discuss the effectiveness of the attack.

**Parameters 3 (Setting the parameters).** *Set the parameters as follows.*

- *Set* $\eta$ *such that* $\eta \geqslant \kappa + \mu + \rho$.
- *Let* $\ell$ *be the number of queries made by* $\mathrm{Enc}^{O}(\mathsf{msk}, \cdot)$; *i.e., the number of queries made to generate each of* $C_1, \ldots, C_\eta$ *in the attack above. Choose a constant* $c$ *such that* $\kappa^c \geqslant 2\eta\kappa$. *Choose* $t$ *based on* $\ell$ *and* $\kappa^{-c}$ *as per Lemma 3.*

**Lemma 4.** *Let* $\mathsf{Brk}^{O}$ *be as in Attack 2, and let* $b'$ *be the output of* $\mathsf{Brk}^{O}$. *Suppose* $n \geqslant 3$ *and* $q^n \in \omega(\mathsf{poly}(\kappa))$. *We have* $\Pr[b' = 0 \mid b = 0] \geqslant 1 - \frac{1}{\kappa}$.

*Proof.* First, recall that $\mathcal{E}^O$ has correctness $1 - \frac{1}{2^\kappa}$ (c.f. the footnote of Page 16). Let $\mathsf{sk}[\mathbf{v}^*]$ be the secret key for $\mathbf{v}^*$ relative to the real oracle $O$, namely $\mathsf{sk}[\mathbf{v}^*] \leftarrow \mathrm{KGen}^{O}(\mathsf{msk}, \mathbf{v}^*)$. By correctness of $\mathcal{E}^O$, for each $h \in [\eta]$, with probability at least $1 - \frac{1}{2^\kappa}$, $\mathrm{Dec}^{O}(\mathsf{sk}[\mathbf{v}^*], C_h) = m_h$. We claim that performing the decryption relative to $O'$ (as opposed to the real oracle $O$) does not impact the decryption result much. In particular, for any $h \in [\eta]$,

$$\Pr[\mathrm{Dec}^{O'}(\mathsf{sk}[\mathbf{v}^*], C_h) = m_h] \geqslant 1 - \frac{1}{2^\kappa} - \kappa^{-c}. \tag{4}$$

Thus, the probability that $\mathsf{Brk}$ mistakenly outputs one when $b = 0$ is at most $\eta(\frac{1}{2^\kappa} + \kappa^{-c})$. This is because $\mathsf{Brk}$ goes through all choices of vector secret keys and Q-A pairs, hitting $\mathsf{sk}[\mathbf{v}^*]$ and $Q_s$ at some point. The reason that the multiplicative factor $\eta$ appears is that we require for all $h \in [\eta]$, the decryption result be $m_h$. (Line 1 of Phase 3 of $\mathsf{Brk}$'s procedure.) Since by Parameters 3, $\kappa^c \geqslant 2\eta\kappa$

$$\Pr[b' = 1 \mid b = 0] \leqslant \eta\left(\frac{1}{2^\kappa} + \kappa^{-c}\right) \leqslant \eta\left(\frac{1}{2^\kappa} + \frac{1}{2\eta\kappa}\right) \leqslant \frac{1}{\kappa}, \tag{5}$$

for all large enough $\kappa$.

To argue about Eq. 4, fix $h \in [\eta]$. Let $\mathsf{S}_0$ and $\mathsf{S}_1$ be the set of Q-A pairs made during the generation of $\mathsf{sk}[\mathbf{v}^*]$ and during the generation of $C_h \leftarrow \mathrm{Enc}^{O}(\mathsf{msk}, \mathbf{x}_h)$. Define the event $\mathsf{Bad}$ as follows.

– $\mathsf{Bad}$: the event that a query in $\mathsf{S}_1 \setminus \mathsf{L}$ is asked during the decryption of $\mathrm{Dec}^O(\mathsf{sk}[\mathbf{v}^*], C_h)$, where recall that the set $\mathsf{L}$ is defined in Step 1 of Phase 2 of $\mathsf{Brk}$'s procedure.

If $Q_s = \mathsf{S}_0$ and $\overline{\mathsf{Bad}}$ holds, then the decryption execution of $\mathrm{Dec}^{O'}(\mathsf{sk}[\mathbf{v}^*], C_h)$ proceeds identically to that of $\mathrm{Dec}^O(\mathsf{sk}[\mathbf{v}^*], C_h)$. Thus, we show $\Pr[\mathsf{Bad}] \leqslant \kappa^{-c}$, which implies

$$\begin{aligned}\Pr[\mathrm{Dec}^{O'}(\mathsf{sk}[\mathbf{v}^*], C_h) = m_h] &\geqslant \Pr[\mathrm{Dec}^{O}(\mathsf{sk}[\mathbf{v}^*], C_h) = m_h] - \kappa^{-c} \\ &\geqslant 1 - 2^{-\kappa} - \kappa^{-c},\end{aligned}$$

as desired.

By Lemma 3 we obtain $\Pr[\mathsf{Bad}] \leqslant \kappa^{-c}$. To see this, set $\ell$ to be the number of queries in $\mathsf{S}_1$. (Parameters 3.) Name these queries as $1, \ldots, \ell$. Let $F\colon \mathbb{Z}_q^n \to 2^{[\ell]}$ be a function, where $F(\mathbf{y})$ contains those queries in $[\ell]$ that appear during the decryption of $\mathrm{Dec}^O(\mathsf{sk}[\mathbf{y}], C_h)$, where $\mathsf{sk}[\mathbf{y}] \leftarrow \mathrm{KGen}^O(\mathsf{msk}, \mathbf{y})$. Now by Lemma 3 the set $\mathsf{L}$ contains all the queries in $\cup_{i\in[t]} F(\mathbf{v}_i)$, except with probability at most $\frac{1}{\kappa^c}$. The proof is now complete. □

**Lemma 5.** *Let $\mathsf{Brk}^O$ be as in Attack 2, and let $b'$ be the output of $\mathsf{Brk}^O$. Then, $\Pr[b' = 1 \mid b = 1] \geqslant 1 - 2^{\mu+\rho-\eta}$. By setting the parameters as in Parameters 3 (specifically that $\eta \geqslant \kappa + \mu + \rho$), $\Pr[b' = 1 \mid b = 1] \geqslant 1 - 2^{\kappa}$.*

*Proof.* Fix a vector secret key $\mathsf{sk} \in \{0,1\}^\mu$ and a set of Q-A pairs $Q_s \in \{0,1\}^\rho$. For any $i \in [\eta]$, the probability that $\mathrm{Dec}^{O'}(\mathsf{sk}, C_i) = m_i$ is at most 1/2 because the righthand side is completely independent of the lefthand side (because $b = 1$; see Definition 2). Thus, the probability that for all $i \in [\eta]$, $\mathrm{Dec}^{O'}(\mathsf{sk}, C_i) = m_i$ is at most $\frac{1}{2^\eta}$. Doing a union bound over all vector secret keys $\mathsf{sk} \in \{0,1\}^\mu$ and all Q-A pairs $Q_s \in \{0,1\}^\rho$, the probability that $\mathsf{Brk}$ outputs zero is at most $2^{\mu+\rho-\eta}$. The proof is now complete. □

Putting together the above lemmas, we achieve the final impossibility result.

**Theorem 4.** *Suppose $n \geqslant 3$ and $q^n$ is super-polynomial in the security parameter. There exists no black-box construction of IPFE for dimension $n$ and modulus $q$ from any primitive that exists relative to a random oracle.*

The condition of $q^n$ being super-polynomial in $\kappa$ in the above theorem is necessary. For example, there exists trivial black-box IPFE constructions for $\mathbb{Z}_2^n$ from OWFs, where $n = \log \kappa$, as follows. Let the master secret key be a PRF key, and let the secret key for a vector be the PRF output for that vector. When encrypting a plaintext vector $\mathbf{x}$ under $\mathsf{msk}$, encrypt the result of $\langle \mathbf{x}, \mathbf{v}\rangle$ under the corresponding secret key for $\mathbf{v}$, for all $\mathbf{v}$. The size of the ciphertext remains polynomial.

**Acknowledgments.** We thank the anonymous reviewers for their helpful comments, in particular for pointing out a technical issue in an earlier draft of this work and suggesting how to resolve it.

Part of the work was done while Mingyuan Wang was at UC Berkeley. Mingyuan Wang is supported in part by an NYU startup fund.

## A The Combinatorial Problem over $\mathbb{Z}_2$

Proving the combinatorial problem over $\mathbb{Z}_2$ is significantly easier than over $\mathbb{Z}_q$ for any prime $q$. Thus, we present it separately for $\mathbb{Z}_2$.

### A.1 Existential Version

As a warm-up, we prove a weaker statement that just establishes the existence of vectors $\mathbf{y}^*, \mathbf{y}_1, \ldots \mathbf{y}_t$ with the desired properties.

**Lemma 6.** *Fix $n = n(\kappa) \in \omega(\log \kappa)$. For $\ell = \mathsf{poly}(\kappa)$ and any map $F : \mathbb{Z}_2^n \to 2^{[\ell]}$ there exists $t = \mathsf{poly}'(\kappa)$, $\mathbf{y}_1, \ldots, \mathbf{y}_t, \mathbf{y}^* \in \mathbb{Z}_2^n$ with*

$$\mathbf{y}^* \notin \mathsf{Span}\left(\mathbf{y}_1, \ldots, \mathbf{y}_t\right) \quad \textit{and} \quad F(\mathbf{y}^*) \subseteq \bigcup_{i=1}^{t} F(\mathbf{y}_i)$$

*when $\kappa$ is sufficiently large.*

The proof uses the following result of [38] on the maximal size of sum-free sets in finite groups. A subset $S$ of a finite group is sum free, if there exist no $a, b, c \in S$ with $a + b = c$.

**Lemma 7.** *[38] The maximal size of a sum-free subset $S \subseteq \mathbb{Z}_2^n, n \geqslant 1$ is $2^{n-1}$.*

Yap [38] proves a bound for arbitrary finite abelian groups. We use this to prove the following theorem, which is the existence-only analog of Thm. 6.

**Theorem 5.** *For every non-empty subset $S \subseteq \mathbb{Z}_2^n, n \geqslant 1$, there are at most $2^{n-1}/|S|$ subspaces $V \subseteq \mathbb{Z}_2^n$ of dimension $n-1$ with $V \cap S = \emptyset$.*

*Proof.* Let $t$ be the number of subspaces of dimension $(n-1)$ that do not contain any of the vectors in $S$. Each such subspace $V$ can be described by a vector $\mathbf{v}$ as follows: $V = \{\mathbf{v} \mid \langle \mathbf{x}, \mathbf{v} \rangle = 0\}$. So let $\mathbf{x}_1, \ldots, \mathbf{x}_t$ be the vectors describing all the subspaces $V$ with $V \cap S = \emptyset$. No subset of 3 of these vectors can be colinear: Assume there exists indices $i, j, k \in [t]$ with $\mathbf{x}_i + \mathbf{x}_j = \mathbf{x}_k$. Then for every vector $\mathbf{v} \in \mathbb{Z}_2^n$ we have $\langle \mathbf{x}_i, \mathbf{v} \rangle = 0$ or $\langle \mathbf{x}_j, \mathbf{v} \rangle = 0$ or $(\langle \mathbf{x}_i, \mathbf{v} \rangle = 1$ and $\langle \mathbf{x}_j, \mathbf{v} \rangle = 1)$. In the last case $\langle \mathbf{x}_k, \mathbf{v} \rangle = \langle (\mathbf{x}_i + \mathbf{x}_j), \mathbf{v} \rangle = 0$. This means every vector $\mathbf{v}$ is contained in the subspace associated to $\mathbf{x}_i$, $\mathbf{x}_j$ or $\mathbf{x}_k$ and thus $S$ would have to be empty. This is a contradiction.

Lemma 7 implies that the vectors $\mathbf{x}_1, \ldots, \mathbf{x}_t$ must contain at least $d := \lceil \log(t+1) \rceil$ linear independent vectors. Let this be without loss of generality $\mathbf{x}_1, \ldots, \mathbf{x}_d$. Clearly, every vector $\mathbf{v} \in S$ must satisfy $\langle \mathbf{x}_i, \mathbf{v} \rangle = 1$. At most $2^{n-d} \leqslant 2^{n-(\log(t)+1)} = 2^{n-1}/t$ can satisfy this equation. Thus $|S| \leqslant 2^{n-1}/t$ which can be rearranged to $t \leqslant 2^{n-1}/|S|$. □

The proof uses two facts that are specific to $\mathbb{Z}_2$:

1. The union of three subspaces, where each one is orthogonal to one of the three vectors $\mathbf{x}_i, \mathbf{x}_j, \mathbf{x}_k$ with $\mathbf{x}_i + \mathbf{x}_j = x_k$, is all of $\mathbb{Z}_2$.
2. For fixed vectors $\mathbf{x}_1, \ldots, \mathbf{x}_t$ the set

$$\{\mathbf{v} \mid \forall i \in [t] : \langle \mathbf{v}, \mathbf{x}_k \rangle \neq 0\}$$

   is an affine subspace of $\mathbb{Z}_2^n$.

In particular that the set is in the second point seems to be algebraically much more complicated over $\mathbb{Z}_q$.

We next give a proof of Lemma 6 using Thm. 5. This part can be easily generalized to $\mathbb{Z}_q$.

*Proof (Proof of Lemma 6).* First of all, note that the requirement that $t$ is polynomial is not necessary here: If there exists $\mathbf{y}^*$ and an $(n-1)$-dimensional subspace $V^*$ with $\mathbf{y}^* \notin V^*$ and $F(\mathbf{y}^*) \subseteq \bigcup_{i=1}^{t} F(\mathbf{y}_i)$, we can set $t := |F(\mathbf{y}^*)| \leqslant m$ and pick for each color $\mathsf{cl} \in F(\mathbf{y}^*)$ a vector $\mathbf{y}_i$.

We prove the Lemma with the double counting technique.

We count the number triples $(\mathsf{cl}, \mathbf{y}^*, V) \in [\ell] \times \mathbb{Z}_q^n \times 2^{\mathbb{Z}_q^n}$ such that

1. $\mathbf{y}^* \notin V$,
2. $V$ is an $(n-1)$-dimensional subspace,
3. $\mathsf{cl} \in F(\mathbf{y}^*)$, and $\mathsf{cl} \notin \bigcup_{\mathbf{y} \in V} F(\mathbf{y})$.

We say that a triple is valid, if it satisfies all of the above conditions. To satisfy the condition in the lemma, for every $\mathbf{y}^*$ and $V$ that satisfy 1 and 2, there must be at least one valid triple. We have $2^n - 1$ choices for a non-zero vector $\mathbf{y}^*$ and then $2^{n-1}$ choices for $V$ such that 1 and 2 are satisfied. Thus there must be at least $(2^n - 1) \cdot 2^{n-1}$ valid triples.

On the other hand, if a color is used for $s$ vectors, by Thm. 5 there are at most $2^{n-1}/s$ $(n-1)$-dimensional subspaces not containing any vector having this color. Thus, for a fixed color there can be at most $2^{n-1}$ contributions.

This leads to the following inequality

$$\ell 2^{n-1} \geqslant (2^n - 1) \cdot 2^{n-1} \quad \Longleftrightarrow \quad \ell \geqslant (2^n - 1).$$

Since $\ell$ grows only polynomial in $\kappa$ but $2^n$ grows exponential in $\kappa$, this inequality can not hold for sufficiently large $\kappa$. □

## A.2 Probabilistic Version

We now give the proof for the probabilistic version over $\mathbb{Z}_2^n$. The proof uses Fourier transforms, but the techniques do not seem to extend beyond $q > 2$.

To prove our impossibility result, we need to strengthen the theorem in two ways:

1. The vector $\mathbf{y}^*$ needs to be uniformly random.

2. The vectors $\mathbf{y}_1, \ldots, \mathbf{y}_t$ have to be efficiently sampleable *without* knowing $F$.

This is formalized by the following lemma:

**Lemma 8.** *Fix $n = n(\kappa) \in \omega(\log \kappa)$. Let $\ell = \mathsf{poly}(\kappa)$. Fix a constant $c$. Then, there exists a polynomial $t = t(\kappa)$ such that with probability at least $1 - n^{-c}$*

$$F(\mathbf{y}^*) \subseteq \bigcup_{i=1}^{t} F(\mathbf{y}_i),$$

*where $\mathbf{y}^* \leftarrow \mathbb{Z}_2^n$, and we sample a random $(n-1)$-dimensional subspace $V$ subject to $\mathbf{v}^* \notin S$ and we sample $\mathbf{y}_1, \ldots, \mathbf{y}_t$ all uniformly at random from $V$.*

This lemma can be proven as Lemma 3, but we can replace Thm. 1 (that is used for the proof) with the following version that has an easier proof and gives a slightly better bound at the cost of being specific for $\mathbb{Z}_2$.

**Theorem 6.** *For any subset $S \subseteq \mathbb{Z}_2^n$, there exist at most $\frac{2^{n+2}}{|S|}$ linear subspaces $V$ of dimension $n-1$ such that*

$$|S \cap V| \leqslant \frac{1}{2} \cdot \frac{|S|}{2}.$$

To prove Theorem 6 we recall the Fourier transform for Boolean hypercubes.

**Definition 4 (Fourier Transform).** *For any function $f : \mathbb{Z}_2^n \to \mathbb{R}$, its Fourier coefficient $\widehat{f}(\mathbf{u})$ for any $\mathbf{u} \in \mathbb{Z}_2^n$ is defined as*

$$\widehat{f}(\mathbf{u}) = \frac{1}{2^n} \cdot \sum_{\mathbf{x} \in \mathbb{Z}_2^n} (-1)^{\langle \mathbf{x}, \mathbf{u} \rangle} \cdot f(\mathbf{x}).$$

**Theorem 7 (Parseval's Identity)**

$$\sum_{\mathbf{x} \in \mathbb{Z}_2^n} f(\mathbf{x})^2 = 2^n \cdot \sum_{\mathbf{u} \in \mathbb{Z}_2^n} \widehat{f}(\mathbf{u})^2.$$

*Proof (of Theorem 7)*

$$\begin{aligned}
\sum_{\mathbf{u} \in \mathbb{Z}_2^n} \widehat{f}(\mathbf{u})^2 &= \sum_{\mathbf{u} \in \mathbb{Z}_2^n} \left( \frac{1}{2^n} \cdot \sum_{\mathbf{x} \in \mathbb{Z}_2^n} (-1)^{\langle x, \mathbf{u} \rangle} \cdot f(x) \right)^2 \\
&= \frac{1}{2^{2n}} \sum_{\mathbf{u}, \mathbf{x}, \mathbf{y} \in \mathbb{Z}_2^n} (-1)^{\langle \mathbf{x}, \mathbf{u} \rangle} \cdot (-1)^{\langle \mathbf{y}, \mathbf{u} \rangle} \cdot f(\mathbf{x}) \cdot f(\mathbf{y}) \\
&= \frac{1}{2^{2n}} \sum_{\mathbf{x}, \mathbf{y} \in \mathbb{Z}_2^n} f(\mathbf{x}) \cdot f(\mathbf{y}) \cdot \sum_{\mathbf{u} \in \mathbb{Z}_2^n} (-1)^{\langle \mathbf{x}+\mathbf{y}, \mathbf{u} \rangle} \\
&= \frac{1}{2^{2n}} \sum_{\mathbf{x} \in \mathbb{Z}_2^n} f(\mathbf{x})^2 \cdot 2^n = \frac{1}{2^n} \sum_{\mathbf{x} \in \mathbb{Z}_2^n} f(\mathbf{x})^2
\end{aligned}$$

□

We are now ready for the proof.

*Proof (of Theorem 6).* Let $f$ be the indicator function for $S$. We know

$$\sum_{\mathbf{x}\in\mathbb{Z}_2^n} f(x)^2 = |S|.$$

By Theorem 7,

$$\sum_{\mathbf{u}\in\mathbb{Z}_2^n} \widehat{f}(\mathbf{u})^2 = \frac{|S|}{2^n}.$$

Observe that,

$$\sum_{\mathbf{u}\neq 0^n} \widehat{f}(\mathbf{u})^2 = \frac{|S|}{2^n} - \left(\frac{|S|}{2^n}\right)^2 \leqslant \frac{|S|}{2^n}.$$

For every non-zero $\mathbf{u}$, let $V_{\mathbf{u}}$ denote the subspace orthogonal to $\mathbf{u}$. Observe that,

$$\widehat{f}(\mathbf{u}) = \frac{|S \cap V_{\mathbf{u}}| - |S \setminus V_{\mathbf{u}}|}{2^n} = \frac{2\cdot|S\cap V_{\mathbf{u}}| - |S|}{2^n}.$$

Hence,

$$|S \cap V_{\mathbf{u}}| \leqslant \frac{1}{2}\cdot\frac{|S|}{2} \implies \widehat{f}(\mathbf{u})^2 \geqslant \left(\frac{|S|}{2^{n+1}}\right)^2.$$

The number of such $\mathbf{u}$ can be upper bounded by

$$\frac{2^{n+2}}{|S|}.$$

□

## References

1. Abdalla, M., Benhamouda, F., Gay, R.: From single-input to multi-client inner-product functional encryption. In: Galbraith, S.D., Moriai, S. (eds.) Advances in Cryptology – ASIACRYPT 2019, Part III. Lecture Notes in Computer Science, vol. 11923, pp. 552–582. Springer, Cham, Switzerland, Kobe, Japan (Dec 8–12, 2019). https://doi.org/10.1007/978-3-030-34618-8_19
2. Abdalla, M., Bourse, F., De Caro, A., Pointcheval, D.: Simple functional encryption schemes for inner products. In: Katz, J. (ed.) PKC 2015. LNCS, vol. 9020, pp. 733–751. Springer, Heidelberg (2015). https://doi.org/10.1007/978-3-662-46447-2_33
3. Abdalla, M., Catalano, D., Fiore, D., Gay, R., Ursu, B.: Multi-input functional encryption for inner products: function-hiding realizations and constructions without pairings. In: Shacham, H., Boldyreva, A. (eds.) Advances in Cryptology – CRYPTO 2018: 38th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 19–23, 2018, Proceedings, Part I, pp. 597–627. Springer International Publishing, Cham (2018). https://doi.org/10.1007/978-3-319-96884-1_20

4. Abdalla, M., Catalano, D., Gay, R., Ursu, B.: Inner-product functional encryption with fine-grained access control. In: Moriai, S., Wang, H. (eds.) Advances in Cryptology – ASIACRYPT 2020: 26th International Conference on the Theory and Application of Cryptology and Information Security, Daejeon, South Korea, December 7–11, 2020, Proceedings, Part III, pp. 467–497. Springer International Publishing, Cham (2020). https://doi.org/10.1007/978-3-030-64840-4_16
5. Abdalla, M., Gay, R., Raykova, M., Wee, H.: Multi-input inner-product functional encryption from pairings. In: Coron, J.-S., Nielsen, J.B. (eds.) Advances in Cryptology – EUROCRYPT 2017: 36th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Paris, France, April 30 – May 4, 2017, Proceedings, Part I, pp. 601–626. Springer International Publishing, Cham (2017). https://doi.org/10.1007/978-3-319-56620-7_21
6. Abdalla, M., Gong, J., Wee, H.: Functional encryption for attribute-weighted sums from k-Lin. In: Micciancio, D., Ristenpart, T. (eds.) Advances in Cryptology – CRYPTO 2020: 40th Annual International Cryptology Conference, CRYPTO 2020, Santa Barbara, CA, USA, August 17–21, 2020, Proceedings, Part I, pp. 685–716. Springer International Publishing, Cham (2020). https://doi.org/10.1007/978-3-030-56784-2_23
7. Agrawal, S., Clear, M., Frieder, O., Garg, S., O'Neill, A., Thaler, J.: Ad hoc multi-input functional encryption. In: Vidick, T. (ed.) ITCS 2020: 11th Innovations in Theoretical Computer Science Conference. vol. 151, pp. 40:1–40:41. LIPIcs, Seattle, WA, USA (Jan 12–14, 2020). https://doi.org/10.4230/LIPIcs.ITCS.2020.40
8. Agrawal, S., Libert, B., Maitra, M., Titiu, R.: Adaptive simulation security for inner product functional encryption. In: Kiayias, A., Kohlweiss, M., Wallden, P., Zikas, V. (eds.) Public-Key Cryptography – PKC 2020: 23rd IACR International Conference on Practice and Theory of Public-Key Cryptography, Edinburgh, UK, May 4–7, 2020, Proceedings, Part I, pp. 34–64. Springer International Publishing, Cham (2020). https://doi.org/10.1007/978-3-030-45374-9_2
9. Agrawal, S., Libert, B., Stehlé, D.: Fully secure functional encryption for inner products, from standard assumptions. In: Robshaw, M., Katz, J. (eds.) Advances in Cryptology – CRYPTO 2016: 36th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 14-18, 2016, Proceedings, Part III, pp. 333–362. Springer Berlin Heidelberg, Berlin, Heidelberg (2016). https://doi.org/10.1007/978-3-662-53015-3_12
10. Ananth, P., Jain, A.: Indistinguishability obfuscation from compact functional encryption. In: Gennaro, R., Robshaw, M. (eds.) Advances in Cryptology – CRYPTO 2015: 35th Annual Cryptology Conference, Santa Barbara, CA, USA, August 16-20, 2015, Proceedings, Part I, pp. 308–326. Springer Berlin Heidelberg, Berlin, Heidelberg (2015). https://doi.org/10.1007/978-3-662-47989-6_15
11. Ananth, P., Vaikuntanathan, V.: Optimal bounded-collusion secure functional encryption. In: Hofheinz, D., Rosen, A. (eds.) Theory of Cryptography: 17th International Conference, TCC 2019, Nuremberg, Germany, December 1–5, 2019, Proceedings, Part I, pp. 174–198. Springer International Publishing, Cham (2019). https://doi.org/10.1007/978-3-030-36030-6_8
12. Baltico, C.E.Z., Catalano, D., Fiore, D., Gay, R.: Practical functional encryption for quadratic functions with applications to predicate encryption. In: Katz, J., Shacham, H. (eds.) Advances in Cryptology – CRYPTO 2017: 37th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 20–24, 2017, Proceedings, Part I, pp. 67–98. Springer International Publishing, Cham (2017). https://doi.org/10.1007/978-3-319-63688-7_3

13. Bennett, M.: Occurrence of right angles in vector spaces over finite fields. Eur. J. Comb. **70**, 155–163 (2018). https://doi.org/10.1016/j.ejc.2017.12.005
14. Bishop, A., Jain, A., Kowalczyk, L.: Function-hiding inner product encryption. In: Iwata, T., Cheon, J.H. (eds.) Advances in Cryptology – ASIACRYPT 2015: 21st International Conference on the Theory and Application of Cryptology and Information Security,Auckland, New Zealand, November 29 – December 3, 2015, Proceedings, Part I, pp. 470–491. Springer Berlin Heidelberg, Berlin, Heidelberg (2015). https://doi.org/10.1007/978-3-662-48797-6_20
15. Bitansky, N., Vaikuntanathan, V.: Indistinguishability obfuscation from functional encryption. In: Guruswami, V. (ed.) 56th Annual Symposium on Foundations of Computer Science. pp. 171–190. IEEE Computer Society Press, Berkeley, CA, USA (Oct 17–20, 2015). https://doi.org/10.1109/FOCS.2015.20
16. Black, J., Rogaway, P., Shrimpton, T.: Black-box analysis of the block-cipher-based hash-function constructions from PGV. In: Yung, M. (ed.) Advances in Cryptology – CRYPTO 2002. Lecture Notes in Computer Science, vol. 2442, pp. 320–335. Springer, Berlin, Heidelberg, Germany, Santa Barbara, CA, USA (Aug 18–22, 2002). https://doi.org/10.1007/3-540-45708-9_21
17. Boneh, D., Franklin, M.K.: Identity-based encryption from the Weil pairing. In: Kilian, J. (ed.) Advances in Cryptology – CRYPTO 2001. Lecture Notes in Computer Science, vol. 2139, pp. 213–229. Springer, Berlin, Heidelberg, Germany, Santa Barbara, CA, USA (Aug 19–23, 2001). https://doi.org/10.1007/3-540-44647-8_13
18. Boneh, D., Papakonstantinou, P.A., Rackoff, C., Vahlis, Y., Waters, B.: On the impossibility of basing identity based encryption on trapdoor permutations. In: 49th Annual Symposium on Foundations of Computer Science, pp. 283–292. IEEE Computer Society Press, Philadelphia, PA, USA (Oct 25–28, 2008). https://doi.org/10.1109/FOCS.2008.67
19. Boneh, D., Sahai, A., Waters, B.: Functional encryption: Definitions and challenges. In: Ishai, Y. (ed.) TCC 2011: 8th Theory of Cryptography Conference. Lecture Notes in Computer Science, vol. 6597, pp. 253–273. Springer, Berlin, Heidelberg, Germany, Providence, RI, USA (Mar 28–30, 2011). https://doi.org/10.1007/978-3-642-19571-6_16
20. Camenisch, J., Lysyanskaya, A.: An efficient system for non-transferable anonymous credentials with optional anonymity revocation. In: Pfitzmann, B. (ed.) Advances in Cryptology – EUROCRYPT 2001. Lecture Notes in Computer Science, vol. 2045, pp. 93–118. Springer, Berlin, Heidelberg, Germany, Innsbruck, Austria (May 6–10, 2001). https://doi.org/10.1007/3-540-44987-6_7
21. Castagnos, G., Laguillaumie, F., Tucker, I.: Practical fully secure unrestricted inner product functional encryption modulo p. In: Peyrin, T., Galbraith, S. (eds.) Advances in Cryptology – ASIACRYPT 2018, Part II. Lecture Notes in Computer Science, vol. 11273, pp. 733–764. Springer, Cham, Switzerland, Brisbane, Queensland, Australia (Dec 2–6, 2018). https://doi.org/10.1007/978-3-030-03329-3_25
22. Gay, R.: A new paradigm for public-key functional encryption for degree-2 polynomials. In: Kiayias, A., Kohlweiss, M., Wallden, P., Zikas, V. (eds.) PKC 2020: 23rd International Conference on Theory and Practice of Public Key Cryptography, Part I. Lecture Notes in Computer Science, vol. 12110, pp. 95–120. Springer, Cham, Switzerland, Edinburgh, UK (May 4–7, 2020). https://doi.org/10.1007/978-3-030-45374-9_4
23. Gentry, C.: Fully homomorphic encryption using ideal lattices. In: Mitzenmacher, M. (ed.) 41st Annual ACM Symposium on Theory of Computing. pp. 169–178.

ACM Press, Bethesda, MD, USA (May 31 – Jun 2, 2009). https://doi.org/10.1145/1536414.1536440
24. Goldreich, O., Goldwasser, S., Micali, S.: How to construct random functions (extended abstract). In: 25th Annual Symposium on Foundations of Computer Science. pp. 464–479. IEEE Computer Society Press, Singer Island, Florida (Oct 24–26, 1984). https://doi.org/10.1109/SFCS.1984.715949
25. Gorbunov, S., Vaikuntanathan, V., Wee, H.: Functional encryption with bounded collusions via multi-party computation. In: Safavi-Naini, R., Canetti, R. (eds.) Advances in Cryptology – CRYPTO 2012. Lecture Notes in Computer Science, vol. 7417, pp. 162–179. Springer, Berlin, Heidelberg, Germany, Santa Barbara, CA, USA (Aug 19–23, 2012). https://doi.org/10.1007/978-3-642-32009-5_11
26. Goyal, V., Kumar, V., Lokam, S.V., Mahmoody, M.: On black-box reductions between predicate encryption schemes. In: Cramer, R. (ed.) TCC 2012: 9th Theory of Cryptography Conference. Lecture Notes in Computer Science, vol. 7194, pp. 440–457. Springer, Berlin, Heidelberg, Germany, Taormina, Sicily, Italy (Mar 19–21, 2012). https://doi.org/10.1007/978-3-642-28914-9_25
27. Impagliazzo, R., Rudich, S.: Limits on the provable consequences of one-way permutations. In: 21st Annual ACM Symposium on Theory of Computing, pp. 44–61. ACM Press, Seattle, WA, USA (May 15–17, 1989). https://doi.org/10.1145/73007.73012
28. Katz, J., Sahai, A., Waters, B.: Predicate encryption supporting disjunctions, polynomial equations, and inner products. J. Cryptol. **26**(2), 191–224 (2013). https://doi.org/10.1007/s00145-012-9119-4
29. Katz, J., Yerukhimovich, A.: On black-box constructions of predicate encryption from trapdoor permutations. In: Matsui, M. (ed.) Advances in Cryptology – ASIACRYPT 2009. Lecture Notes in Computer Science, vol. 5912, pp. 197–213. Springer, Berlin, Heidelberg, Germany, Tokyo, Japan (Dec 6–10, 2009). https://doi.org/10.1007/978-3-642-10366-7_12
30. O'Neill, A.: Definitional issues in functional encryption. Cryptology ePrint Archive, Report 2010/556 (2010). https://eprint.iacr.org/2010/556
31. Papakonstantinou, P.A., Rackoff, C.W., Vahlis, Y.: How powerful are the DDH hard groups? Cryptology ePrint Archive, Report 2012/653 (2012). https://eprint.iacr.org/2012/653
32. Rothblum, R.: Homomorphic encryption: from private-key to public-key. In: Ishai, Y. (ed.) TCC 2011: 8th Theory of Cryptography Conference. Lecture Notes in Computer Science, vol. 6597, pp. 219–234. Springer, Berlin, Heidelberg, Germany, Providence, RI, USA (Mar 28–30, 2011). https://doi.org/10.1007/978-3-642-19571-6_14
33. Sahai, A., Seyalioglu, H.: Worry-free encryption: functional encryption with public keys. In: Al-Shaer, E., Keromytis, A.D., Shmatikov, V. (eds.) ACM CCS 2010: 17th Conference on Computer and Communications Security, pp. 463–472. ACM Press, Chicago, Illinois, USA (Oct 4–8, 2010). https://doi.org/10.1145/1866307.1866359
34. Sahai, A., Waters, B.R.: Fuzzy identity-based encryption. In: Cramer, R. (ed.) Advances in Cryptology – EUROCRYPT 2005. Lecture Notes in Computer Science, vol. 3494, pp. 457–473. Springer, Berlin, Heidelberg, Germany, Aarhus, Denmark (May 22–26, 2005). https://doi.org/10.1007/11426639_27
35. Schul-Ganz, G., Segev, G.: Generic-group identity-based encryption: A tight impossibility result. In: Tessaro, S. (ed.) 2nd Conference on Information-Theoretic Cryptography, ITC 2021, July 23-26, 2021, Virtual Conference. LIPIcs, vol. 199, pp. 26:1–26:23. Schloss Dagstuhl - Leibniz-Zentrum für Informatik (2021)

36. Tairi, E., Ünal, A.: Lower bounds for lattice-based compact functional encryption. In: Joye, M., Leander, G. (eds.) Advances in Cryptology – EUROCRYPT 2024, Part II. Lecture Notes in Computer Science, vol. 14652, pp. 249–279. Springer, Cham, Switzerland, Zurich, Switzerland (May 26–30, 2020). https://doi.org/10.1007/978-3-031-58723-8_9
37. Ünal, A.: Impossibility results for lattice-based functional encryption schemes. In: Canteaut, A., Ishai, Y. (eds.) Advances in Cryptology – EUROCRYPT 2020, Part I. Lecture Notes in Computer Science, vol. 12105, pp. 169–199. Springer, Cham, Switzerland, Zagreb, Croatia (May 10–14, 2020). https://doi.org/10.1007/978-3-030-45721-1_7
38. Yap, H.P.: Maximal sum-free sets of group elements. J. London Math. Society **s1-44**(1), 131–136 (1969). https://doi.org/10.1112/jlms/s1-44.1.131
39. Zhandry, M.: Augmented random oracles. In: Dodis, Y., Shrimpton, T. (eds.) Advances in Cryptology – CRYPTO 2022, Part III. Lecture Notes in Computer Science, vol. 13509, pp. 35–65. Springer, Cham, Switzerland, Santa Barbara, CA, USA (Aug 15–18, 2022). https://doi.org/10.1007/978-3-031-15982-4_2

## Authentication and Sequentiality

# Hide-and-Seek and the Non-resignability of the BUFF Transform

Jelle Don[1], Serge Fehr[1,2], Yu-Hsuan Huang[1](✉), Jyun-Jie Liao[3], and Patrick Struck[4]

[1] Centrum Wiskunde & Informatica (CWI), Amsterdam, The Netherlands
jelle.don@cwi.nl, serge.fehr@cwi.nl, yhh@cwi.nl
[2] Mathematical Institute, Leiden University, Leiden, The Netherlands
[3] Cornell University, Ithaca, NY, USA
jjliao@cs.cornell.edu
[4] University of Konstanz, Konstanz, Germany
patrick.struck@uni.kn

**Abstract.** The BUFF transform, due to Cremers et al. (S&P'21), is a generic transformation for digital signature scheme, with the purpose of obtaining additional security guarantees beyond unforgeability: exclusive ownership, message-bound signatures, and non-resignability. Non-resignability (which essentially challenges an adversary to re-sign an unknown message for which it only obtains the signature) turned out to be a delicate matter, as recently Don et al. (CRYPTO'24) showed that the initial definition is essentially unachievable; in particular, it is *not* achieved by the BUFF transform. This led to the introduction of new, weakened versions of non-resignability, which are (potentially) achievable. In particular, it was shown that a *salted* variant of the BUFF transform does achieves some weakened version of non-resignability. However, the salting requires additional randomness and leads to slightly larger signatures. Whether the *original* BUFF transform also achieves some meaningful notion of non-resignability remained a natural open question.

In this work, we answer this question in the affirmative. We show that the BUFF transform satisfies the (almost) strongest notions of non-resignability one can hope for, facing the known impossibility results. Our results cover both the statistical and the computational case, and both the classical and the quantum setting. At the core of our analysis lies a new security game for random oracles that we call *Hide-and-Seek*. While seemingly innocent at first glance, it turns out to be surprisingly challenging to rigorously analyze.

## 1 Introduction

**Digital Signatures and the BUFF Transform.** Digital signatures are at the very heart of modern cryptography. The gold standard security notion for digital signature schemes is (strong) unforgeability against chosen message attacks. However, in certain applications, additional security properties are desirable, or even necessary. For example, [17] showed that the *"Dynamically Recreatable Key"* protocol [18] is insecure if the signature scheme used in the protocol does not additionally offer some sort of non-malleability property that, informally, requires it to be hard to turn a signature for an unknown message into a signature for the same message but under a different public key (with a possibly known secret key), in other word, it should be hard to re-sign an unknown message. This property was named *non-resignability* in [17], and formally defined later in [10], along with two more properties: *exclusive ownership* and *message-bound signatures*. On top, [10] introduced a generic transformation, the *BUFF transform*, which can be applied to any signature scheme, and it was argued that the transformed signature scheme then satisfies these three additional properties (in the random oracle model). The transform is very simple: instead of signing the message $m$, a BUFF-transformed signature scheme signs the hash $H(\mathsf{pk}, m)$ of the public key and the message, and this hash value is also appended to the signature.

Motivated by the fact that the NIST call for additional post-quantum signatures [20] explicitly mentioned the above as "*additional desirable security properties*", several of the NIST post-quantum signature submissions have the BUFF transform built in, or mention the possibility of applying the BUFF transform to the proposed scheme.

**Recent Development.** Somewhat surprisingly given the apparently clear situation around the BUFF transform, the recent work [13] showed that the question of defining and achieving *non-resignability* is actually more subtle. Concretely, it was shown that non-resignability, as defined in [10], is almost unachievable as a matter of fact, both in the plain model and in the random oracle model.[1] In particular, it follows that the BUFF transform does *not* achieve non-resignability (as defined in [10]). The apparent contradiction to the positive claim from [10] comes from the fact that the proof in [10] relied on a non-malleability claim for the random oracle that was taken from [4], and which turned out to be false.

Towards showing a positive result, [13] introduced $\mathsf{NR}^{H,\perp}$, a weaker version of the original definition of non-resignability (in the ROM), and they showed that a *salted version* of the BUFF transform satisfies $\mathsf{NR}^{H,\perp}$. The situation is actually more complicated in that the non-resignability definition involves an entropy condition, of which one can consider a statistical or a computational variant. While the impossibility of [13] holds for both, the positive result on $\mathsf{NR}^{H,\perp}$ for the salted BUFF transform holds for the statistical variant only, and provably not for the computational variant.[2]

In reaction to the negative results from (an early version of) [13], the authors of [10] updated their paper to [11] by weakening their definition of

[1] There are hypothetical signature schemes to which the attack from [13] does not apply, though we are not aware of any natural scheme for which that is the case.

[2] We note that the statistic and the computational variants of $\mathsf{NR}^{H,\perp}$ are incomparable: in the computational case, the adversary is restricted in its computational power but is bound to a weaker entropy condition.

non-resignability and tried to argue that the (original) BUFF transform satisfies their weakened definition; however, their argument relies on an assumption that is shown to be false in [13].

Thus, the bottom line is that the following question has remained open:

*Does the BUFF transform satisfy some non-trivial notion(s) of non-resignability?*

**Our Results.** In this work, we answer the above question in the affirmative. Concretely, we introduce yet another variant of non-resignability, $\mathsf{sNR}^{H,\perp}$, and we show that the (original) BUFF transform satisfies $\mathsf{sNR}^{H,\perp}$, both in the statistical setting, where the entropy condition holds statistically and adversaries may be computationally unbounded, and in the computational setting, where the entropy condition holds computationally and adversaries have bounded computing power only.

In the statistical setting, $\mathsf{sNR}^{H,\perp}$ is strictly stronger than $\mathsf{NR}^{H,\perp}$; in the computational setting, the two notions are (probably) incomparable, yet $\mathsf{sNR}^{H,\perp}$ is strictly stronger than the notion considered in [11]. Therefore, given that [13] showed that the BUFF transform does not satisfy $\mathsf{NR}^{H,\perp}$ in the computational setting, our results appear to be the best we can hope for towards proving positive results on the non-resignability of the BUFF transform.

Our approach is inspired by the proof in [13] for the salted BUFF transform. Indeed, on the technical level, we can recycle and adjust some of the arguments, although we avoid the detour via some tailor-made non-malleability property for the random oracle. The crucial part of course is when [13] exploits the salt that originates from the salted BUFF transform, which we cannot do, given that we consider the original, unsalted variant. Instead, we capture the crucial, missing piece in the form of a particular, simple game in the random oracle model, which we call *Hide-and-Seek*, and we reduce the non-resignability property of the BUFF transform to the hardness of winning Hide-and-Seek. In essence, the game asks to find $x$ when given $H(x)$ and query-bounded access to $H$, where $x$ may be chosen arbitrarily *dependent* on $H$ subject to the condition that it is hard to guess when given access to $H$ only, i.e., without being given $H(x)$.

Despite its simplicity and harmless appearance, this game turns out to be surprisingly tricky to analyze. Thus, the technical core of this work is in analyzing Hide-and-Seek and showing that it is hard to win, both in the statistical and in the computational setting, and both in the classical and in the quantum ROM.

**Related Work.** The relevance of the BUFF security notions can be traced to attacks [3,17], which exploit the absence of additional security properties like exclusive ownership, message-bound signatures, and non-resignability. The former security notion (exclusive ownership) was first mentioned by Pornin and Stern [21] which can further be traced back to [6,19]. Along with defining exclusive ownership, Pornin and Stern also give three generic transformations that achieve exclusive ownership. The other security notions (message-bound signatures and non-resignability) were formalized in [10].

In very recent work, Aulbach et al. [2] analyzed the BUFF security of the schemes submitted to the recent NIST standardization process for post-quantum signature schemes [20], though considering an even weaker notion of non-resignability than $\mathsf{NR}^{H,\perp}$ (where there is no auxiliary information at all).

Also very recently, Düzlü et al. [14] reconsider the BUFF security notions for Falcon [22], exploiting the particular form of a Falcon signature, and they argue that all that is needed is to replace the hash $H(r, m)$ in a Falcon signature computation by $H(r, \mathsf{pk}, m)$; thus, the hash can be "recycled" (this was argued in [10] already), but also, it does *not* need to be appended to the signature as in the BUFF transform (in line with the lighter transform by Pornin and Stern [21]). Regarding non-resignability, they consider the variant from [11], which is weaker than $\mathsf{NR}^{H,\perp}$, but relax the HILL entropy requirement to a bound on the *computational unpredictability*, which makes the definition stronger in that aspect. Thus, strictly speaking, the considered variant is incomparable with the computational versions of $\mathsf{NR}^{H,\perp}$ and $\mathsf{sNR}^{H,\perp}$.

## 2 Preliminary

We start by briefly spelling out the notions of guessing probability and min-entropy, and recalling the random oracle model. Then, we introduce $\mathsf{sNR}^{H,\perp}$, the variant of non-resignability we consider in this paper. Finally, we recall the BUFF transform, as introduced in [10].

### 2.1 Guessing Probability and Min-Entropy

For a random variable $X$ over a finite set $\mathcal{X}$, the *guessing probability* and the *min-entropy* are respectively defined as

$$\mathsf{guess}(X) := \max_x \Pr[X = x] \qquad \text{and} \qquad \mathsf{H}_\infty(X) := -\log\big(\mathsf{guess}(X)\big)$$

where here and in the remainder, log is the binary logarithm. For random variables $X$ and $Y$ over respective finite sets $\mathcal{X}$ and $\mathcal{Y}$, the *conditional* guessing probability is defined as

$$\mathsf{guess}(X \mid Y) := \sum_y \Pr[Y = y] \max_x \Pr[X = x \mid Y = y]\,.$$

It is well known that $\mathsf{guess}(X \mid Y) = \max_f \Pr[X = f(Y)]$, where the maximization is over all (deterministic or randomized) functions $f : \mathcal{Y} \to \mathcal{X}$. In line with the unconditional case above, the *conditional* min-entropy is then given by $\mathsf{H}_\infty(X \mid Y) := -\log\big(\mathsf{guess}(X \mid Y)\big)$.

### 2.2 The Random-Oracle Model

Throughout, we consider the random oracle model (ROM) [5], i.e., we consider a uniformly randomly function $H \colon \mathcal{X} \to \mathcal{Y}$, where $\mathcal{X}$ and $\mathcal{Y}$ are suitably chosen, finite sets, and algorithms are (only) given *oracle* access to $H$. By default, we consider algorithms to be *classical* and thus make classical queries to $H$; however, we also consider the quantum setting, in which case we then explicitly refer to *quantum queries* and/or the *quantum* random oracle model (QROM) [7]. In some case, we also consider an algorithm that can make an unbounded number of queries to $H$, in which case it then is irrelevant if these are classical or quantum.

## 2.3 Non-resignability

Let $\mathcal{S} = (\mathsf{KGen}^H, \mathsf{Sign}^H, \mathsf{Vrfy}^H)$ be a signature scheme, where we make explicit that we consider schemes in the random oracle model, and thus key-generation, signing, and verification are given oracle access to $H$. As usual, we require $\mathsf{KGen}^H$, $\mathsf{Sign}^H$, and $\mathsf{Vrfy}^H$ to be PPT, and it is understood that $\mathsf{KGen}^H$ takes the unary representation of $\lambda$ as input, where $\lambda$ is the security parameter. By default, we denote the message space by $\mathcal{M}$ and the public-key and secret-key spaces by $\mathcal{PK}$ and $\mathcal{SK}$, respectively. Without loss of generality, we assume that the public key $\mathsf{pk}$ can be efficiently computed from its corresponding secret key $\mathsf{sk}$.

In this work, we consider a new variant of *non-resignability*, denoted $\mathsf{sNR}^{H,\perp}$. It is similar in spirit as $\mathsf{NR}^{H,\perp}$ introduced in [13]; in particular, a crucial aspect is that $\mathsf{aux}$ is not given access to $H$, but we additionally provide the adversary with the secret key $\mathsf{sk}$, and we adjust the entropy condition correspondingly (see below for a more detailed comparison). The security game is shown in Fig. 1. It is played by randomized oracle algorithms[3],

$$\mathcal{D}^H : \mathcal{SK} \to \mathcal{M} \qquad \text{and} \qquad \mathcal{A}^H : \mathcal{SK} \times \mathcal{SGN} \times \mathcal{AUX} \to \mathcal{PK} \times \mathcal{SGN}$$

given query access to $H$, referred to as *adversaries*, and a randomized algorithm $\mathsf{aux} : \mathcal{SK} \times \mathcal{M} \to \mathcal{AUX}$ with no access to $H$, referred to as *hint function.*[4]

```
$\mathsf{sNR}_{\mathcal{S}}^{H,\perp}(\mathcal{D}, \mathcal{A}, \mathsf{aux})$:
 1: $(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{KGen}^H$
 2: $m \leftarrow \mathcal{D}^H(\mathsf{sk})$
 3: $\sigma \leftarrow \mathsf{Sign}^H(\mathsf{sk}, m)$
 4: $(\mathsf{pk}', \sigma') \leftarrow \mathcal{A}^H(\mathsf{sk}, \sigma, \mathsf{aux}(m, \mathsf{sk}))$
 5: return $\mathsf{pk} \neq \mathsf{pk}' \wedge \mathsf{Vrfy}^H(\mathsf{pk}', m, \sigma') = 1$
```

**Fig. 1.** Our new variant of the non-resignability game $\mathsf{sNR}^{H,\perp}$.

While playing $\mathsf{sNR}^{H,\perp}$, we consider restricted ($\mathcal{S}$-dependent) classes of adversaries with a give bound $h$ on the entropy

$$\underset{\substack{(\mathsf{sk},\mathsf{pk}) \leftarrow \mathsf{KGen}^H \\ m \leftarrow \mathcal{D}^H(\mathsf{sk})}}{\mathsf{H}_\infty} (m \mid H, \mathsf{sk}, \mathsf{aux}(\mathsf{sk}, m)) \geq h\,. \tag{1}$$

[3] Here and in the remainder, we borrow from set notation to indicate the input and output space of (oracle) algorithms. In case of an algorithm that takes no input, we write the singleton set $\{\perp\}$ as domain.

[4] The hint function may be randomized, but we refer to it as a function for convenience.

For now we only consider the statistical variant, where we take an arbitrary but fixed security parameter for $\mathcal{S}$, where $\mathcal{D}$, $\mathcal{A}$ and $\mathsf{aux}$ may be computationally unbounded and we only limit their query complexity, and where the entropy requirement holds statistically, i.e., as in (1). The computational setting is handled later in Sect. 5; there, $\mathcal{D}, \mathcal{A}$ and $\mathsf{aux}$ are restricted to be (uniform or non-uniform) PPT algorithms, and the entropy requirement is expressed via HILL entropy (which causes some complications given that (1) conditions on the entire function table of $H$).

Informally, we say that a signature scheme $\mathcal{S} = (\mathsf{KGen}, \mathsf{Sign}, \mathsf{Vrfy})$ is *non-resignable* if for all $\mathcal{D}$, $\mathcal{A}$ and any hint function $\mathsf{aux}$ that satisfy the statistical entropy condition (1) for sufficiently large $h$, the probability of winning the $\mathsf{sNR}^{H,\perp}$ game, i.e.,

$$\mathbf{Adv}^{\mathsf{sNR}^{H,\perp}}(\mathcal{D}, \mathcal{A}, \mathsf{aux}) := \Pr\left[1 = \mathsf{sNR}^{H,\perp}_{\mathcal{S}}(\mathcal{D}, \mathcal{A}, \mathsf{aux})\right] ,$$

is small.

The recent developments have shown that formalizing non-resignability is a non-trivial task, and different weaker variants of the original (unachievable) version have been proposed. We quickly discuss here how $\mathsf{sNR}^{H,\perp}$ relates to those variants; namely, we show that is stronger than the versions proposed in [13] and [11].

*Comparison with Non-resignability from* [13]*.* The difference to $\mathsf{NR}^{H,\perp}$ as defined in [13] is that $\mathsf{sNR}^{H,\perp}$ provides the $\mathcal{D}$, $\mathcal{A}$ and the hint function $\mathsf{aux}$ with the secret key $\mathsf{sk}$, whereas $\mathsf{NR}^{H,\perp}$ only provides the public key $\mathsf{pk}$ (recall that we assume that $\mathsf{pk}$ can be computed from $\mathsf{sk}$). This of course gives more power to the adversary. The other difference lies in the entropy requirement: for $\mathsf{NR}^{H,\perp}$, the message is required to have high entropy conditioned on $\mathsf{pk}$ (and $\mathsf{aux}$) only, i.e.,

$$\mathsf{H}_\infty(m \mid H, \mathsf{pk}, \mathsf{aux}(\mathsf{pk}, m)) \geq \text{ h}$$

whereas $\mathsf{sNR}^{H,\perp}$ requires (1) to hold, which conditions on $\mathsf{sk}$ instead; this seems to be a stronger restriction, but we observe that for $m \leftarrow \mathcal{D}(\mathsf{pk})$, produced by a $\mathcal{D}$ that only gets the public key as input (as in $\mathsf{NR}^{H,\perp}$),

$$\mathsf{H}_\infty(m \mid H, \mathsf{pk}, \mathsf{aux}(\mathsf{pk}, m)) = \mathsf{H}_\infty(m \mid H, \mathsf{sk}, \mathsf{aux}(\mathsf{pk}, m))$$

since $\mathsf{sk} \to (H, \mathsf{pk}, \mathsf{aux}(\mathsf{pk}, m)) \to m$ forms a Markov chain then. This implies that any attack against $\mathsf{NR}^{H,\perp}$ can be cast as an attack against $\mathsf{sNR}^{H,\perp}$ with the same entropy bound, making the latter a stronger security notion.

*Comparison with Non-resignability from* [11]*.* We first note that [11] defines non-resignability only in the computational setting, so we compare it with the computational version of $\mathsf{sNR}^{H,\perp}$. While we have postponed the exact definition to Sect. 5, the high level reasoning can still be understood. First of all, in [11] the side information on $m$ (given by $\mathsf{aux}$ in our case) is required to be *computationally independent* of $m$, which is equivalent to allowing *no side information at all* when considering computationally bounded adversaries. Furthermore, in line with $\mathsf{sNR}^{H,\perp}$, the entropy condition (though phrased in terms of HILL entropy) is required to hold when conditioning on the secret key $\mathsf{sk}$. But on the other hand and in the spirit of $\mathsf{NR}^{H,\perp}$, the adversaries are only given $\mathsf{pk}$ as input, and not $\mathsf{sk}$. Altogether, this makes their notion weaker than our computational version of $\mathsf{sNR}^{H,\perp}$, which provided $\mathsf{sk}$ as input to the adversaries.

### 2.4 BUFF Transformation

The BUFF transform, as proposed in [10], transforms any signature scheme $\mathcal{S}$ into another signature scheme $\mathsf{BUFF}[\mathcal{S}, H]$. The transformation is described in Fig. 2; in essence, $\mathsf{BUFF}[\mathcal{S}, H]$ signs a message $m$ by signing the hash $H(\mathsf{pk}, m)$ and additionally appending this hash value to the signature.

```
$\mathsf{KGen}'^H$:
  1: $(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{KGen}^H$
  2: **return** $(\mathsf{sk}, \mathsf{pk})$

$\mathsf{Sign}'^H(\mathsf{sk}, m)$:
  1: $y := H(\mathsf{pk}, m)$
  2: $\sigma \leftarrow \mathsf{Sign}^H(\mathsf{sk}, y)$
  3: $\sigma' := (\sigma, y)$
  4: **return** $\sigma'$

$\mathsf{Vrfy}'^H(\mathsf{pk}, m, \sigma')$:
  1: $(\sigma, y) := \sigma'$
  2: **return** $\mathsf{Vrfy}^H(\mathsf{pk}, y, \sigma) \wedge$
  3:          $y = H(\mathsf{pk}, m)$
```

**Fig. 2.** The signature scheme $\mathsf{BUFF}[\mathcal{S}, H] = (\mathsf{KGen}'^H, \mathsf{Sign}'^H, \mathsf{Vrfy}'^H)$, obtained from applying the BUFF transform to $\mathcal{S} = (\mathsf{KGen}^H, \mathsf{Sign}^H, \mathsf{Vrfy}^H)$.

Here and in the remainder when considering the BUFF transform, we take it as understood that the random oracle $H\colon \mathcal{X} \to \mathcal{Y}$ has fitting domain and range, i.e., $\mathcal{X} \supseteq \mathcal{K} \times \mathcal{M}$ and $\mathcal{Y} \subseteq \mathcal{M}$, so that the BUFF transform is well defined.

## 3 Hide-and-Seek and the Non-resignability of BUFF

Our goal is to prove the non-resignability (in the sense of $\mathsf{sNR}^{H,\perp}$) of the BUFF transform, which signs a message $m$ by signing $H(\mathsf{pk}, m)$, with the hash value then appended to the signature. Clearly, for this non-resignability to hold, it must *necessarily* be hard to recover $m$ from $H(\mathsf{pk}, m)$. This hardness may look trivial at first glance, since $H$ is (typically) compressing, and modeled as a random oracle; however, it turns out to be not trivial at all. The reason is that in the $\mathsf{sNR}^{H,\perp}$ game, $m$ is produced arbitrarily and dependent on $H$, with the only promise being that $m$ is hard to guess from scratch (i.e., when $H(\mathsf{pk}, m)$ is not given).

In this section, we formally capture (a particular formulation of) this hardness via a game, which we call *Hide-and-Seek*, and we show that hardness of winning Hide-and-Seek is *sufficient* for proving the non-resignability of the BUFF transform. The main technical challenge then lies in proving that Hide-and-Seek is hard to win, which we do in Sect. 4.

Throughout the remainder, let $\mathcal{X}, \mathcal{Y}$, and $\mathcal{Z}$ be finite non-empty sets, and let $H\colon \mathcal{X} \to \mathcal{Y}$ be the random oracle.

### 3.1 The Hide-and-Seek Game

The Hide-and-Seek game is played by two adversaries $\mathcal{D}$ and $\mathcal{A}$: the (possibly query-unbounded) *hider* $\mathcal{D}^H\colon \{\perp\} \to \mathcal{X} \times \mathcal{Z}$, and the query-bounded *seeker* $\mathcal{A}^H\colon \mathcal{Y} \times \mathcal{Z} \to \mathcal{X}$ that is allowed to make at most $q$ queries to $H$. First, $\mathcal{D}^H$ chooses a challenge $x \in \mathcal{X}$ together with a hint $z \in \mathcal{Z}$ and "hides" $x$ as $H(x)$, and then $\mathcal{A}^H$ is supposed to find $x$ from $H(x)$ and $z$. The game is formally specified as follows:

```
$\mathsf{HnS}^H(\mathcal{D},\mathcal{A})$:
 1: $(x,z) \leftarrow \mathcal{D}^H$
 2: **return** $x = \mathcal{A}^H(H(x),z)$
```

In line with the entropy condition in $\mathsf{sNR}^{H,\perp}$, we require $x$ to be statistically hidden given $H$ and $z$. I.e., we require that

$$\mathsf{guess}(x \,|\, H,z) \leq \epsilon \tag{2}$$

for some small $\epsilon > 0$. Informally, we say that the random oracle $H$ satisfies the Hide-and-Seek property, or $\mathsf{HnS}^H$ for short, if for every such pair of $\mathcal{D},\mathcal{A}$ as above, the winning probability, given as

$$\mathbf{Adv}^{\mathsf{HnS}^H}(\mathcal{D},\mathcal{A}) := \Pr\big[1 = \mathsf{HnS}^H(\mathcal{D},\mathcal{A})\big] = \Pr_{(x,z)\leftarrow\mathcal{D}^H}\big[x = \mathcal{A}^H(H(x),z)\big]\,,$$

is small.

As mentioned above already, what is tricky about this game is that $x$ (and $z$) may depend arbitrarily on $H$, subject to the bound (2) on the guessing probability. Because of this, known results on inverting the random oracle do not apply, and it may not be fully clear whether we can actually expect it to be hard to win, i.e., that there is no sneaky way to win the game. We discuss this in more detail in Sect. 4, where we then analyze Hide-and-Seek and prove that it *is* hard to win after all.

### 3.2 Reducing $\mathsf{sNR}^{H,\perp}$ of BUFF to Hide-and-Seek

In the following statement, we reduce the $\mathsf{sNR}^{H,\perp}$ security of the BUFF transform $\mathsf{BUFF}[\mathcal{S},H]$ of a signature scheme $\mathcal{S} = (\mathsf{KGen}^H, \mathsf{Sign}^H, \mathsf{Vrfy}^H)$ to the hardness of winning the Hide-and-Seek game $\mathsf{HnS}^H$. In the lemma statement, the parameters $q_K$ and $q_S$ refer to (an upper bound on) the number of queries to $H$ that $\mathsf{KGen}^H$ and $\mathsf{Sign}^H$ perform.

**Lemma 1.** *Let $\mathcal{D}^H : \mathcal{SK} \to \mathcal{M}$ and $\mathcal{A}^H : \mathcal{SK} \times \mathcal{SGN} \times \mathcal{AUX} \to \mathcal{PK} \times \mathcal{SGN}$ be $\mathsf{sNR}^{H,\perp}$-adversaries against $\mathsf{BUFF}[\mathcal{S},H]$ for some $\mathsf{aux} : \mathcal{SK} \times \mathcal{M} \to \mathcal{AUX}$, making at most $q_D$ and $q_A$ queries to $H$, respectively. Then there exists a hider $\bar{\mathcal{D}} : \{\perp\} \to \mathcal{X} \times \mathcal{Z}$ and a seeker $\bar{\mathcal{A}} : \mathcal{Y} \times \mathcal{Z} \to \mathcal{X}$ with $\mathcal{Z} = \mathcal{SK} \times \mathcal{AUX}$, where $\bar{\mathcal{A}}$ makes at most $q_A + q_S$ queries to $H$, and such that*

$$\mathsf{H}_{\infty\atop (x,z)\leftarrow\bar{\mathcal{D}}^H}(x \mid H,z) = \mathsf{H}_{\infty\atop {(\mathsf{sk},\mathsf{pk})\leftarrow\mathsf{KGen}^H \atop m\leftarrow\mathcal{D}^H(\mathsf{sk})}}(m \mid H, \mathsf{sk}, \mathsf{aux}(\mathsf{sk},m)) \tag{3}$$

*and*

$$\mathbf{Adv}^{\mathsf{sNR}^{H,\perp}}_{\mathsf{BUFF}[\mathcal{S},H]}(\mathcal{D},\mathcal{A},\mathsf{aux}) \leq (q_A+q_S)\cdot \mathbf{Adv}^{\mathsf{HnS}^H}(\bar{\mathcal{D}},\bar{\mathcal{A}}) + q_K\epsilon + \frac{q_D+1}{|\mathcal{Y}|}\,, \quad (4)$$

*where* $\epsilon := 2^{-\mathsf{H}_\infty(x|H,z)}$*. In the case* $\mathcal{A}$ *makes quantum queries to* $H$*, then*

$$\mathbf{Adv}^{\mathsf{sNR}^{H,\perp}}_{\mathsf{BUFF}[\mathcal{S},H]}(\mathcal{D},\mathcal{A},\mathsf{aux}) \leq 2(q_A+q_S)\cdot \sqrt{\mathbf{Adv}^{\mathsf{HnS}^H}(\bar{\mathcal{D}},\bar{\mathcal{A}})} + q_K\epsilon + \frac{q_D+1}{|\mathcal{Y}|}\,, \quad (5)$$

*holds in place of* (4)*, and* $\bar{\mathcal{A}}$ *then makes quantum queries as well.*

*Furthermore, in the computational setting when considering a non-fixed security parameter and PPT algorithms* $\mathcal{D}$ *and* $\mathcal{A}$*, then* $\bar{\mathcal{D}}$ *and* $\bar{\mathcal{A}}$ *are PPT as well.*

The intuition behind the proof is as follows. Consider the $\mathsf{sNR}^{H,\perp}$ game. Due to the assumed hardness of Hide-and-Seek, $\mathcal{A}$ cannot recover $m$ from its input and thus makes no query to $H$ that has $m$ as suffix. But then it cannot gather any information on $H(\mathsf{pk}',m)$ for any $\mathsf{pk}'$, and thus it will not be able to output $y' = H(\mathsf{pk}',m)$ for any $\mathsf{pk}'$. Formally, we have to make sure that $\mathcal{A}$ gets no information on $y'$ via its input, which is controlled by $\mathsf{KGen}$ and $\mathcal{D}$, which may query $H$ on $H(\mathsf{pk}',m)$ for any $\mathsf{pk}'$. This is taken care of in our formal proof below.

*Proof.* In Fig. 3 we define a hybrid sequence reducing the $\mathsf{sNR}^{H,\perp}$ property of $\mathsf{BUFF}[\mathcal{S},H]$ to the $\mathsf{HnS}^H$ property of $H$. To start with, we note that the adversary $(\mathcal{D},\mathcal{B})$ playing $G_0$ is identical to $(\mathcal{D},\mathcal{A})$ playing $\mathsf{sNR}^{H,\perp}_{\mathsf{BUFF}[\mathcal{S},H]}$.
*The* $\mathsf{G}_0$ *to* $\mathsf{G}_1$ *hop.* The only difference between $\mathsf{G}_0$ and $\mathsf{G}_1$ is whether $\mathcal{B}$ is given oracle access to the original random oracle $H$, or the reprogrammed oracle $H[(\cdot,m)\mapsto\perp]$ and replies with $\perp$ to any query that has suffix $m$.

Consider the hider $\bar{\mathcal{D}}$ and seeker $\bar{\mathcal{A}}$, where $\bar{\mathcal{D}}$ samples $(\mathsf{sk},\mathsf{pk}) \leftarrow \mathsf{KGen}^H$ and $m \leftarrow \mathcal{D}^H(\mathsf{sk})$ and returns

$$x := (\mathsf{pk}, m) \qquad \text{and} \qquad z := (\mathsf{sk}, \mathsf{aux}(\mathsf{sk},m))\,,$$

and on input $H(x) = H(\mathsf{pk},m)$ and $z$, the seeker $\bar{\mathcal{A}}$ samples a random index $i \leftarrow [q_A+q_S]$, runs

$$\mathcal{B}^H(\mathsf{sk}, H(\mathsf{pk},m), \mathsf{aux}(\mathsf{sk},m)) = \mathcal{A}^H\big(\mathsf{sk}, \big(H(\mathsf{pk},m), \mathsf{Sign}^H(\mathsf{sk},y)\big), \mathsf{aux}(\mathsf{sk},m)\big)$$

internally, but then looks at / does a full measurement of the $i$-th query to obtain $(\mathsf{pk}^*_i, m^*_i)$, and returns $(\mathsf{pk}, m^*_i)$. It is clear by construction that $z \in \mathcal{SK}\times\mathcal{AUX}$, and (3) immediately follows from the fact that $\mathsf{pk}$ can be derived from $\mathsf{sk}$, and so

$$\mathsf{H}_\infty(x \mid H, z) = \mathsf{H}_\infty(\mathsf{pk}, m \mid H, \mathsf{sk}, \mathsf{aux}(\mathsf{sk},m)) = \mathsf{H}_\infty(m \mid H, \mathsf{sk}, \mathsf{aux}(\mathsf{sk},m)) \quad (6)$$

as claimed. It also follows from construction that $\bar{\mathcal{D}}$ and $\bar{\mathcal{A}}$ preserve the efficiency of $\mathcal{D}$ and $\mathcal{A}$. In terms of query complexity, $\bar{\mathcal{A}}$ makes at most $q_A+q_S$ queries to $H$.

```
$\mathsf{G}_0$:
  1: $(\mathsf{sk},\mathsf{pk}) \leftarrow \mathsf{KGen}^H$; $m \leftarrow \mathcal{D}^H(\mathsf{sk})$
  2: $(\mathsf{pk}',y') \leftarrow \mathcal{B}^H(\mathsf{sk},y,\mathsf{aux}(\mathsf{sk},m))$
  3: return $H(\mathsf{pk}',m)=y' \wedge \mathsf{pk}' \neq \mathsf{pk}$

$\mathcal{B}^H(\mathsf{sk},y,a)$:
  1: $\sigma \leftarrow \mathsf{Sign}^H(\mathsf{sk},y)$
  2: return $(\mathsf{pk}',y') \leftarrow \mathcal{A}^H(\mathsf{sk},(\sigma,y),a)$

$\mathsf{G}_1$:
  1: $(\mathsf{sk},\mathsf{pk}) \leftarrow \mathsf{KGen}^H$; $m \leftarrow \mathcal{D}^H(\mathsf{sk})$
  2: $(\mathsf{pk}',y') \leftarrow \mathcal{B}^{H[(\cdot,m)\mapsto\perp]}(\mathsf{sk},H(\mathsf{pk},m),\mathsf{aux}(\mathsf{sk},m))$
  3: return $H(\mathsf{pk}',m)=y' \wedge \mathsf{pk}' \neq \mathsf{pk}$

$\mathsf{G}_2$:
  1: $(\mathsf{sk},\mathsf{pk}) \leftarrow \mathsf{KGen}^H$; $m \leftarrow \mathcal{D}^H(\mathsf{sk})$
  2: abort if KGen queried $H(\cdot,m)$
  3: $(\mathsf{pk}',y') \leftarrow \mathcal{B}^{H[(\cdot,m)\mapsto\perp]}(\mathsf{sk},H(\mathsf{pk},m),\mathsf{aux}(\mathsf{sk},m))$
  4: return $H(\mathsf{pk}',m)=y' \wedge \mathsf{pk}' \neq \mathsf{pk}$

$\mathsf{G}_3^i$:
  1: $(\mathsf{sk},\mathsf{pk}) \leftarrow \mathsf{KGen}^H$; $m \leftarrow \mathcal{D}^H(\mathsf{sk})$
  2: abort if KGen queried $H(\cdot,m)$
  3: $(\mathsf{pk}',y') \leftarrow \mathcal{B}^{H[(\cdot,m)\mapsto\perp]}(\mathsf{sk},H(\mathsf{pk},m),\mathsf{aux}(\mathsf{sk},m))$
  4: return $H(\mathsf{pk}',m)=y' \wedge \mathsf{pk}' \neq \mathsf{pk} \wedge (\mathsf{pk}',m)=(\mathsf{pk}_i,m_i)$
  5: {where $(\mathsf{pk}_i,m_i)$ is $\mathcal{D}$'s $i$th query.}

$\mathsf{G}_4^i$:
  1: $(\mathsf{sk},\mathsf{pk}) \leftarrow \mathsf{KGen}^H$; $m \leftarrow \mathcal{D}^H(\mathsf{sk})$
  2: abort if KGen queried $H(\cdot,m_i)$
  3: $(\mathsf{pk}',y') \leftarrow \mathcal{B}^{H[(\cdot,m_i)\mapsto\perp]}(\mathsf{sk},H(\mathsf{pk},m_i),\mathsf{aux}(\mathsf{sk},m_i))$
  4: return $H(\mathsf{pk}_i,m_i)=y' \wedge \mathsf{pk}_i \neq \mathsf{pk}$
  5: {where $(\mathsf{pk}_i,m_i)$ is $\mathcal{D}$'s $i$th query.}
```

**Fig. 3.** Hybrid steps reducing $\mathsf{sNR}^{H,\perp}_{\mathsf{BUFF}[\mathcal{S},H]}$ to $\mathsf{HnS}^H$ when $\mathcal{D}$ is classical, i.e., (5), (4). In the derivations below we drop the parameter $k$ for notational convenience.

In the case where $\mathcal{A}$ makes classical queries, there is no difference in the two games when $\mathcal{B}$ makes no query to a point where the two oracles differ, and thus

$$\begin{aligned}\Pr[1\leftarrow\mathsf{G}_0] &\le \Pr[1\leftarrow\mathsf{G}_1] + \Pr\left[\exists\, i\in[q_A+q_S] \text{ s.t. } m_i^*=m\right]\\ &\le \Pr[1\leftarrow\mathsf{G}_1] + (q_A+q_S)\cdot \mathbf{Adv}^{\mathsf{HnS}^H}(\bar{\mathcal{D}},\bar{\mathcal{A}})\ .\end{aligned}$$

In the quantum case, the same kind of guarantee follows from the O2H lemma [1, Theorem 3], which gives us that

$$\begin{aligned}\Pr[1\leftarrow\mathsf{G}_0] &\le \Pr[1\leftarrow\mathsf{G}_1] + 2(q_A+q_S)\cdot\sqrt{\Pr[m_i^*=m]}\\ &\le \Pr[1\leftarrow\mathsf{G}_1] + 2(q_A+q_S)\cdot\sqrt{\mathbf{Adv}^{\mathsf{HnS}^H}(\bar{\mathcal{D}},\bar{\mathcal{A}})}\ .\end{aligned}$$

*The* $\mathsf{G}_1$ *to* $\mathsf{G}_2$ *hop.* The difference between $\mathsf{G}_1$ and $\mathsf{G}_2$ is that the latter aborts if

$\mathsf{KGen}^H$ ever makes a query of the form $(\cdot, m)$. Given that $m$ is produced given $(H, \mathsf{sk})$ but independent of $\mathsf{KGen}$'s $q_K$ queries conditioned on $(H, \mathsf{sk})$, and $m$ satisfies (1) for $h := \log(1/\epsilon)$, we have

$$\Pr[1 \leftarrow \mathsf{G}_1] \leq \Pr[1 \leftarrow \mathsf{G}_2] + \Pr[\mathsf{G}_2 \text{ abort}] \leq \Pr[1 \leftarrow \mathsf{G}_2] + q_K \epsilon \ .$$

*The* $\mathsf{G}_2$ *to* $\mathsf{G}_3^i$ *hop.* Assume without loss of generality that $\mathcal{D}$ never repeats its queries $(k_1, m_1), \ldots, (k_{q_D}, m_{q_D})$. Note that the queries of $\mathcal{A}$ in $\mathsf{G}_1$ are blocked at $(\cdot, m)$, and the game aborts if $\mathsf{KGen}$ ever queries $(\cdot, m)$. Since, conditioned on $\mathsf{KGen}$ not querying with $(\cdot, m)$, $k' \neq k$ and $(k', m) \neq (k_i, m_i)$ for all $i$, the output $H(k', m)$ is uniformly random and independent of $\mathcal{A}$'s input $(\mathsf{sk}, H(\mathsf{pk}, m), \mathsf{aux}(\mathsf{sk}, m))$ together with the oracle $H[(\cdot, m) \mapsto \perp]$ it has access to, we have

$$\begin{aligned}
\Pr[1 \leftarrow \mathsf{G}_2] &\leq \Pr\left[\begin{matrix} \exists i \in [q_D] \text{ s.t. } (k', m) = (k_i, m_i) \\ 1 \leftarrow \mathsf{G}_2 \end{matrix}\right] \\
&\quad + \Pr\left[ 1 \leftarrow \mathsf{G}_2 \,\middle|\, \begin{matrix} \mathsf{KGen} \text{ not querying } (\cdot, m) \\ (k', m) \neq (k_i, m_i)\ \forall i \in [q_D] \\ k' \neq k \end{matrix} \right] \\
&\leq \sum_{i \in [q_D]} \Pr\left[1 \leftarrow \mathsf{G}_3^i\right] + 1/|\mathcal{Y}| \ .
\end{aligned}$$

*The* $\mathsf{G}_3^i$ *to* $\mathsf{G}_4^i$ *hop.* Because of the extra condition $(\mathsf{pk}', m') = (\mathsf{pk}_i, m_i)$ in $\mathsf{G}_3^i$, replacing $\mathsf{pk}'$ with $\mathsf{pk}_i$ and $m'$ with $m_i$ as in $\mathsf{G}_4^i$, does not change the winning probability. We further drop the condition $(\mathsf{pk}', m') = (\mathsf{pk}_i, m_i)$, which does not decrease the winning probability.

Finally, it remains to upper bound the winning probability of $\mathsf{G}_4^i$ for each $i \in [q_D]$. By a lazy sampling argument, we note that conditioned on $\mathsf{KGen}$ not querying with $(\cdot, m_i)$ and $\mathsf{pk}_i \neq \mathsf{pk}$, the output $H(k_i, m_i)$ is uniform and independent of $(H[(\cdot, m_i) \mapsto \perp], k, H(k, m_i), \mathsf{aux}(m_i))$, and hence, $y'$ generated by $\mathcal{A}^{H[(\cdot, m_i) \mapsto \perp]}(k, H(k, m_i), \mathsf{aux}(m_i))$ is equal to $H(k_i, m_i)$ with probability at most $1/|\mathcal{Y}|$, i.e.

$$\Pr\left[1 \leftarrow \mathsf{G}_4^i\right] \leq 1/|\mathcal{Y}| \ ,$$

which concludes (4), (5). □

*Remark 1.* We point out that the claim on $\bar{\mathcal{D}}$ and $\bar{\mathcal{A}}$ be PPT if $\mathcal{D}$ and $\mathcal{A}$ are, fails to hold when aiming for a variant of Lemma 1 that considers $\mathsf{NR}^{H,\perp}$ instead of $\mathsf{sNR}^{H,\perp}$. The reason is that, on input $H(\mathsf{pk}, m)$ and $z$, the seeker $\bar{\mathcal{A}}$ needs to run $\mathcal{A}$ on *a signature of* $H(\mathsf{pk}, m)$, which it can do efficiently if given $\mathsf{sk}$ (which is part of $z$ here, exploiting that $\mathcal{D}$ is given $\mathsf{sk}$), but not if only given $\mathsf{pk}$. This is the reason why in the *computational* setting, treated in Sect. 5, our proof for showing that BUFF satisfies $\mathsf{sNR}^{H,\perp}$ does not carry over to $\mathsf{NR}^{H,\perp}$ (in line with the counter example given in [13]).

### 3.3 Main Result

By means of the above reduction to $\mathsf{HnS}$ and the analysis of $\mathsf{HnS}$ in the upcoming section, we obtain the following main result on the non-resignability of the BUFF transform $\mathsf{BUFF}[\mathcal{S}, H]$ of any signature scheme $\mathcal{S} = (\mathsf{KGen}^H, \mathsf{Sign}^H, \mathsf{Vrfy}^H)$. In the theorem statement, the parameters $q_K$ and $q_S$ refer to (an upper bound on) the number of queries to $H$ that $\mathsf{KGen}^H$ and $\mathsf{Sign}^H$ perform. The theorem is obtained via plugging in Theorem 2 and 3 into Lemma 1 with some simplification to the obtained upperbounds. For completeness, we spell out its proof in Appendix A.

**Theorem 1.** *Let $\mathcal{D}^H : \mathcal{SK} \to \mathcal{M}$ and $\mathcal{A}^H : \mathcal{SK} \times \mathcal{SGN} \times \mathcal{AUX} \to \mathcal{PK} \times \mathcal{SGN}$ be $\mathsf{sNR}^{H,\perp}$-adversaries against $\mathsf{BUFF}[\mathcal{S}, H]$ for some $\mathsf{aux} : \mathcal{SK} \times \mathcal{M} \to \mathcal{AUX}$, making at most $q_D$ and $q_A$ queries to $H$, respectively, where (1) is satisfied for $h$ such that $0 < \epsilon := 2^{-h} \leq \frac{1}{2}$. Then*

$$\mathbf{Adv}^{\mathsf{sNR}^{H,\perp}}_{\mathsf{BUFF}[\mathcal{S},H]}(\mathcal{D}, \mathcal{A}, \mathsf{aux}) \leq 8(q_A + q_S + 1)^2 \log\left(\frac{|\mathcal{SK}| \cdot |\mathcal{AUX}|}{\epsilon}\right)\epsilon + q_K\epsilon + \frac{q_D + 1}{|\mathcal{Y}|} ,$$

*and in the case $\mathcal{A}$ makes quantum queries to $H$, then*

$$\begin{aligned}&\mathbf{Adv}^{\mathsf{sNR}^{H,\perp}}_{\mathsf{BUFF}[\mathcal{S},H]}(\mathcal{D}, \mathcal{A}, \mathsf{aux}) \\ &\leq O\left(\sqrt{\left(\log \frac{|\mathcal{SK}| \cdot |\mathcal{AUX}|}{\epsilon} + q_A + q_S\right)(q_A + q_S)^3 \epsilon}\right) + q_K\epsilon + \frac{q_D + 1}{|\mathcal{Y}|} ,\end{aligned}$$

*where the asymptotic bound holds as $\min(1/\epsilon, q_A) \to \infty$, and the constants are absolute constants.*

*Remark 2.* In the case where $\mathcal{D}$ makes quantum queries to $H$, we expect a similar argument as in the proof of [13, Theorem 15] applies (resulting in adjusted bounds).

## 4 Analyzing Hide-and-Seek

As explained above, the technical core of proving the non-resignability property of the BUFF transform consists of analyzing the Hide-and-Seek game. Concretely, our goal is to show that the probability

$$\Pr\left[\, x = \mathcal{A}^H(H(x), z) \,\right]$$

is small, for any query-*un*bounded algorithm $\mathcal{D}^H$ that produces a pair $(x, z)$ such that $\mathsf{guess}(x \mid H, z) \leq \epsilon$ holds, and for any query-bounded algorithm $\mathcal{A}$.

Below in Sect. 4.2, we first consider the case of an $\mathcal{A}$ that makes classical queries to the random oracle $H$; later we also consider the case of quantum queries, which introduces additional challenges. We note that since $\mathcal{D}$ has unbounded query complexity, it is irrelevant whether those are classical or quantum; $\mathcal{D}$ may inspect the entire function table anyway.[5] We also emphasize that we do not restrict the computational complexity of $\mathcal{D}$ or $\mathcal{A}$.

Before jumping into the analysis though, we discuss the game a bit further, and in particular we look into the simple(r) variant where $x$ is uniformly random and independent of $H$, and $z$ is fixed.

## 4.1 Special Case: Uniform Challenges

What makes the game challenging to analyze is that the challenge $x$ (and the hint $z$) may be arbitrarily correlated with $H$, as long as $\mathsf{guess}(x \mid H, z) \leq \epsilon$. For instance, given a function $H : \mathcal{X} \to \mathcal{X}$, the hider $\mathcal{D}$ can pick a challenge $x$ that satisfies $H(x) = x$, and the seeker $\mathcal{A}$ can simply output $H(x)$. Although this is not a valid attack under the condition $\mathsf{guess}(x \mid H, z) \leq \epsilon$, because a random function $H : \mathcal{X} \to \mathcal{X}$ typically does not have many fixed points, this example suggests that one cannot argue that $H(x)$ reveals no information about $x$.

In the special case where $x$ is uniform and independent of $(H, z)$ and $z$ is fixed, it is straightforward to show that any $\mathcal{A}$ making at most $q$ classical queries to the random oracle $H$ satisfies

$$\Pr\big[x = \mathcal{A}^H(H(x), z)\big] \leq \frac{(q+1)}{|\mathcal{X}|} .$$

In addition, even if the hint $z$ can depend on $H$, tight bounds are known in the literature: the probability that a $q$-query seeker $\mathcal{A}$ succeeds is in the order of at most $q \log|\mathcal{Z}|/|\mathcal{X}|$ if $\mathcal{A}$ is classical [9,12], or of at most $q(q + \log|\mathcal{Z}|)/|\mathcal{X}|$ if $\mathcal{A}$ can make quantum queries [8].

However, in the general case, where the only guarantee about $x$ is that $\mathsf{guess}(x \mid H, z) \leq \epsilon$ for some $\epsilon < 1$, the strong bounds above do not apply. Nevertheless, in the remaining of this section, we will show how to reduce the tricky general case to the uniform-challenge case.

Inspired by [8], we will actually reduce the general case to the "multi-instance" case with uniform challenges and an independent hint. In particular, consider challenges $x_1^u, \ldots, x_k^u$ that are sampled uniformly and independently from $\mathcal{X}$, and a fixed hint $z^\circ \in \mathcal{Z}$ that does not depend on $x_1^u, \ldots, x_k^u$ and $H$. Then for any seeker that attempts to solve all $k$ challenges with the hint $z^\circ$, it is not hard to prove the following lemma. For completeness we give the proof in Appendix B.

**Lemma 2.** *For every oracle algorithm $\mathcal{A}^H : \mathcal{Y} \times \mathcal{Z} \to \mathcal{X}$ that makes at most $q$ classical queries to $H$,*

$$\Pr\left[\forall i \in [k]\colon x_i^u = \mathcal{A}^H(H(x_i^u), z^\circ)\right] \leq k! \frac{(q+1)^k}{|\mathcal{X}|^k} ,$$

*where $\mathcal{A}$ is independently re-executed for each $i$.*

[5] For the purpose of proving Theorem 1, it would be sufficient to restrict the seeker $\mathcal{D}$ to be query bounded as well; however, interestingly, we need the result for a query unbounded $\mathcal{D}$ for the computational case (see Sect. 5 and Remark 5).

The case where $\mathcal{A}$ can make quantum queries to $H$ is more involved, but has been studied in [8].

**Lemma 3 (Corollary of [8, Lemma 5.2]).** *For every oracle algorithm $\mathcal{A}^H : \mathcal{Y} \times \mathcal{Z} \to \mathcal{X}$ that makes at most $q$ quantum queries to $H$,*

$$\Pr\left[\forall i \in [k] \colon x_i^u = \mathcal{A}^H(H(x_i^u), z^\circ)\right] \leq O\left(\frac{kq + q^2}{|\mathcal{X}|}\right)^k \text{ as } \min(k, q, |\mathcal{X}|) \to \infty\,,$$

*where $\mathcal{A}$ is independently re-executed for each $i$, and the constants in the asymptotic bound are absolute constants.*

## 4.2 The Classical Case

The following provides a bound on the Hide-and-Seek property of the random oracle for a *classical* seeker $\mathcal{A}$.

**Theorem 2 (The RO satisfies $\mathsf{HnS}^H$, classically).** *Let $\mathcal{D} : \{\perp\} \to \mathcal{X} \times \mathcal{Z}$ and $\mathcal{A} : \mathcal{Y} \times \mathcal{Z} \to \mathcal{X}$ be $\mathsf{HnS}^H$-adversaries satisfying (2) for some $0 < \epsilon < 1$, where $\mathcal{A}$ makes $q$ classical queries to $H$. Then we have*

$$\mathbf{Adv}^{\mathsf{HnS}^H}(\mathcal{D}, \mathcal{A}) \leq 2(q+1)\big(\log|\mathcal{Z}| + \log(1/\epsilon) + 1\big)\epsilon + \epsilon\,.$$

Our strategy is to turn a successful $\mathsf{HnS}^H$ seeker $\mathcal{A}$ into a similarly successful guesser $\mathcal{G}$ that tries to guess $x$ from $H$ and $z$. Since such a successful guesser cannot exist by (2), no successful $\mathcal{A}$ can exist.

*Proof.* Given that $\mathcal{A}$ is classical here, we may assume it to be deterministic. For any fixed choices $H^\circ$ and $z^\circ$, we can thus define the set

$$S(H^\circ, z^\circ) := \{x^\circ \in \mathcal{X} \,|\, \mathcal{A}^{H^\circ}(H^\circ(x^\circ), z^\circ) = x^\circ\}$$

of all $x^\circ$ on which $\mathcal{A}$ succeeds.

Following the above strategy for proving the claimed statement, we consider the following guesser $\mathcal{G}$. On input $H$ and $z$, it samples and outputs a uniformly random $\hat{x} \in S(H, z)$ as guess for $x$ (with the convention that $\hat{x} = \perp$ in case $S$ is empty). We can then lower bound the success probability of $\mathcal{G}$ as follows, for any positive $T \in \mathbb{Z}$.

$$\begin{aligned}
\Pr[\hat{x} = x] &\geq \Pr[\hat{x} = x \wedge |S| \leq T] \\
&\geq \frac{1}{T}\Pr[\mathcal{A}^H(H(x), z) = x \wedge |S| \leq T] \\
&\geq \frac{1}{T}\big(\Pr[\mathcal{A}^H(H(x), z) = x] - \Pr[|S| > T]\big)\,,
\end{aligned}$$

where for the second inequality we exploit that for any fixed choices of $H, x$, and $z$, if $|S| \leq T$ then $\Pr[\hat{x} = x] = 1/|S| \geq 1/T$ whenever $x \in S$, i.e. $\mathcal{A}^H(H(x), z) = x$, and 0 otherwise, and so the inequality is obtained by averaging over the choices of $H$, $x$, and $z$. The last inequality is by union bound. Rearranging the terms, we thus have

$$\mathbf{Adv}^{\mathsf{HnS}^H}(\mathcal{D},\mathcal{A}) \leq T \cdot \Pr[\hat{x} = x] + \Pr[|S| > T] \leq T\epsilon + \Pr[|S| > T]\,. \tag{7}$$

In order to control $\Pr[|S| > T]$, we introduce

$$\sigma(H^\circ, z^\circ) \coloneqq \Pr\Big[x^u = \mathcal{A}^{H^\circ}(H^\circ(x^u), z^\circ)\Big] = \frac{|S(H^\circ, z^\circ)|}{|\mathcal{X}|} \tag{8}$$

where $x^u \leftarrow \mathcal{X}$, and we observe that, for any positive $k \in \mathbb{Z}$,

$$\sigma(H^\circ, z^\circ)^k = \Pr\Big[x_i^u = \mathcal{A}^{H^\circ}(H^\circ(x_i^u), z^\circ)\ \forall i \in [k]\Big]$$

where $x_1^u, \ldots, x_k^u \leftarrow \mathcal{X}$. What we are actually interested in is the average over the choice of $H$ and $z$. Towards this end, we note that

$$\begin{aligned}\mathbb{E}[\sigma(H,z)^k] &= \Pr\Big[x_i^u = \mathcal{A}^H(H(x_i^u), z)\ \forall i \in [k]\Big]\\ &= \sum_{z^\circ} \Pr\Big[z = z^\circ\ \wedge\ x_i^u = \mathcal{A}^H(H(x_i^u), z^\circ)\ \forall i \in [k]\Big]\\ &\leq \sum_{z^\circ} \Pr\Big[x_i^u = \mathcal{A}^H(H(x_i^u), z^\circ)\ \forall i \in [k]\Big]\\ &\leq |\mathcal{Z}| \cdot \frac{k!(q+1)^k}{|\mathcal{X}|^k}\,,\end{aligned} \tag{9}$$

where the last inequality is by Lemma 2. Thus

$$\mathbb{E}[|S(H,z)|^k] = |\mathcal{X}|^k \cdot \mathbb{E}[\sigma(H,z)^k] \leq |\mathcal{Z}| \cdot k!(q+1)^k\,,$$

and so by Markov's inequality,

$$\Pr\big[|S(H,z)| > 2k(q+1)\big] \leq \frac{\mathbb{E}\big[|S(H,z)|^k\big]}{\big(2k(q+1)\big)^k} \leq \frac{|\mathcal{Z}|}{2^k} \leq \epsilon$$

where the final inequality is achieved by choosing $k = \big\lceil \log|\mathcal{Z}| + \log(1/\epsilon)\big\rceil$. Thus, setting $T = 2k(q+1)$ and plugging into (7) we obtain that

$$\begin{aligned}\mathbf{Adv}^{\mathsf{HnS}^H}(\mathcal{D},\mathcal{A}) &\leq T\epsilon + \Pr[|S| > T]\\ &\leq 2(q+1)\big(\log|\mathcal{Z}| + \log(1/\epsilon) + 1\big)\epsilon + \epsilon\,.\end{aligned}$$

This proves the claim. □

### 4.3 A Bound for the Quantum Case

The following provides a bound on the Hide-and-Seek property of the random oracle for a *quantum* seeker $\mathcal{A}$.

**Theorem 3 (The RO satisfies $\mathsf{HnS}^H$, quantumly).** *Let $\mathcal{D} : \{\perp\} \to \mathcal{X} \times \mathcal{Z}$ and $\mathcal{A} : \mathcal{Y} \times \mathcal{Z} \to \mathcal{X}$ be $\mathsf{HnS}^H$-adversaries satisfying (2) for some $0 < \epsilon < 1$, where $\mathcal{A}$ makes $q$ quantum queries to $H$. Then we have*

$$\mathbf{Adv}^{\mathsf{HnS}^H}(\mathcal{D}, \mathcal{A}) \leq O\big((\log|\mathcal{Z}| + \log(1/\epsilon) + q)q\epsilon\big)$$

*as $\min(1/\epsilon, |\mathcal{Z}|, q) \to \infty$, where the constants in the asymptotic bound are absolute constants.*

The proof here follows very closely the proof for the classical case, except that we use Lemma 3 to bound the multi-instance game for a quantum algorithm. Furthermore, some additional changes are needed since we cannot assume $\mathcal{A}$ to be deterministic anymore.

*Proof.* Here, for any $H^\circ$ and $z^\circ$, we define the following "weighted set"

$$S^*(H^\circ, z^\circ) := \left\{ \big(x^\circ, w_{H^\circ,z^\circ}(x^\circ)\big) \,\middle|\, x^\circ \in \mathcal{X} \right\},$$

where each element $x^\circ$ comes with a weight, given by

$$w_{H^\circ,z^\circ}(x^\circ) := \Pr\big[x^\circ = \mathcal{A}^{H^\circ}(H^\circ(x^\circ), z^\circ)\big]\,.$$

The total weight of $S^*(H^\circ, z^\circ)$ is defined as $W(S^*(H^\circ, z^\circ)) := \sum_{x^\circ} w_{H^\circ,z^\circ}(x^\circ)$.

Here, we consider the guesser $\mathcal{G}$ that, on input $H$ and $z$, chooses its guess $\hat{x}$ by picking it from $\mathcal{X}$ according to the renormalized weights, i.e., according to the distribution

$$p_{H,z}(\hat{x}) := \frac{w_{H,z}(\hat{x})}{W(S^*(H,z))}\,.$$

We observe that this generalizes the approach in the previous section where $\mathcal{A}$ may assumed to be deterministic. All weights are then 0 or 1, giving rise to the set $S$ in the proof of Theorem 2 when keeping only the elements with weight 1, and the total weight of $S^*$ then matches up with $|S|$, and $\hat{x}$ is then uniformly random in $S$.

We proceed by following that approach, with obvious changes. Namely, first we note that

$$\begin{aligned}\Pr[\hat{x} = x] &\geq \Pr[\hat{x} = x \,\wedge\, W(S^*(H,z)) \leq T] \\ &\geq \frac{1}{T}\Pr[\mathcal{A}^H(H(x), z) = x \,\wedge\, W(S^*(H,z)) \leq T] \\ &\geq \frac{1}{T}\big(\Pr[\mathcal{A}^H(H(x), z) = x] - \Pr[W(S^*(H,z)) > T]\big)\,,\end{aligned}$$

where here, for the second inequality, we exploit that for any fixed choices of $H, x$ and $z$, if $W(S^*(H,z)) \leq T$ then $\Pr[\hat{x} = x] = p_{H,z}(x) \geq w_{H,z}(x)/T$, and so the inequality is obtained by averaging over these choices. Rearranging the terms, we have

$$\mathbf{Adv}^{\mathsf{HnS}^H}(\mathcal{D}, \mathcal{A}) \leq T\epsilon + \Pr[W(S^*(H,z)) > T]\,. \tag{10}$$

In order to control $\Pr[W(S^*(H,z)) > T]$, we introduce

$$\sigma(H^\circ, z^\circ) := \Pr\Big[x^u = \mathcal{A}^{H^\circ}(H^\circ(x^u), z^\circ)\Big] = \frac{W(S^*(H^\circ, z^\circ))}{|\mathcal{X}|} \tag{11}$$

where $x^u \leftarrow \mathcal{X}$. Recycling the line of reasoning in the previous section, we observe that, for any positive $k \in \mathbb{Z}$,

$$\sigma(H^\circ, z^\circ)^k = \Pr\Big[x_i^u = \mathcal{A}^{H^\circ}(H^\circ(x_i^u), z^\circ)\ \forall i \in [k]\Big]$$

where $x_1^u, \ldots, x_k^u \leftarrow \mathcal{X}$, and that

$$\begin{aligned}\mathbb{E}[\sigma(H,z)^k] &= \Pr\Big[x_i^u = \mathcal{A}^H(H(x_i^u), z)\ \forall i \in [k]\Big] \\ &= \sum_{z^\circ} \Pr\Big[z = z^\circ \wedge x_i^u = \mathcal{A}^H(H(x_i^u), z^\circ)\ \forall i \in [k]\Big] \\ &\leq \sum_{z^\circ} \Pr\Big[x_i^u = \mathcal{A}^H(H(x_i^u), z^\circ)\ \forall i \in [k]\Big] \\ &\leq |\mathcal{Z}| \cdot C^k \frac{(k+q)^k q^k}{|\mathcal{X}|^k}\,,\end{aligned} \tag{12}$$

for some absolute constant $C$, and $k, q, |\mathcal{X}|$ large enough, where the last inequality is now by Lemma 3, given that $\mathcal{A}$ is quantum. Thus

$$\mathbb{E}[W(S^*(H,z))^k] = |\mathcal{X}|^k \cdot \mathbb{E}[\sigma(H,z)^k] \leq |\mathcal{Z}| \cdot C^k (k+q)^k q^k\,,$$

and so by Markov inequality,

$$\Pr\big[W(S^*(H,z)) > 2C(k+q)q\big] \leq \frac{\mathbb{E}\big[W(S^*(H,z))^k\big]}{\big(2C(k+q)q\big)^k} \leq \frac{|\mathcal{Z}|}{2^k} \leq \epsilon$$

where the final inequality is achieved by choosing the minimum possible $k \geq \log|\mathcal{Z}| + \log(1/\epsilon)$. Thus, setting $T = 2C(k+q)q$ and plugging into (10) we obtain that

$$\begin{aligned}\mathbf{Adv}^{\mathsf{HnS}^H}(\mathcal{D}, \mathcal{A}) &\leq T\epsilon + \Pr[W(S^*(H,z)) > T] \\ &\leq O\big((\log|\mathcal{Z}| + \log(1/\epsilon) + q)q\epsilon\big)\,.\end{aligned}$$

This proves the claim. □

## 5 Non-resignability in the Computational Setting

Here, we want to extend our result on non-resignability of the BUFF transform to the computational setting, where $\mathcal{D}, \mathcal{A}$ and $\mathsf{aux}$ are polynomially bounded, and where the entropy requirement (1) holds computationally only; the latter is the reason why the computational case does not follow directly from the statistical case. In order to capture the entropy requirement (1) in the computational setting via HILL entropy, we need the notion of the HILL entropy *in the ROM*, as introduced in [13], which we briefly recall below.

Here and for the remainder of this section, we take it as understood that the domain and co-domain of $H : \mathcal{X} \to \mathcal{Y}$ may depend on the security parameter $\lambda$; for simplicity, we leave this dependency implicit. Moreover, we assume the co-domain of $H$ to be super-polynomially large, i.e., $|\mathcal{Y}| \geq \lambda^{\omega(1)}$. For simplicity, we restrict to asymptotic bounds below.

### 5.1 HILL Entropy in the ROM

The HILL entropy [15,16] is introduced as a computational analogue of min-entropy. For a pair of random variables $(X, Y)$, we say that $X$ has high conditional HILL entropy given $Y$, if there is another random variable $Z$, such that $(X, Y)$ and $(Z, Y)$ are computationally indistinguishable, and yet $Z$ has high min-entropy given $Y$.

However, expressing (1) naively using HILL entropy is problematic, since $H$, which is conditioned on, is too large for a computationally bounded distinguisher to even read. Because of this reason, [13] introduced the notion of HILL entropy *in the ROM*, where instead of conditioning on $H$, the distinguisher (that tries to distinguish $(X, Y)$ and $(Z, Y)$) is given bounded oracle access to $H$. We recall (the asymptotic version of) the formal definition.

**Definition 1.** *Let $(X_\lambda, Y_\lambda)$ be a pair of (possibly $H$-dependent) random variables for each $\lambda$. We say that $X = \{X_\lambda\}_\lambda$ has $k(\lambda)$ bits of* conditional HILL entropy *given $Y = \{Y_\lambda\}_\lambda$* in the ROM, *denoted by*

$$\mathsf{HILL}^H_\infty(X \mid Y) \geq k(\lambda) ,$$

*if for every $\lambda$ there exists a random variable $Z_\lambda$ with $\mathsf{H}_\infty(Z_\lambda \mid Y_\lambda, H) \geq k(\lambda)$, and so that $\{(X_\lambda, Y_\lambda)\}_\lambda$ and $\{(Z_\lambda, Y_\lambda)\}_\lambda$ are computationally indistinguishable for oracle algorithms.*

*Remark 3.* Following the standard definition, computationally indistinguishability holds for *non-uniform* PPT distinguishers; this then allows us to consider *non-uniform* PPT (oracle) algorithms $\mathcal{D}$ and $\mathcal{A}$ below. If instead we consider computationally indistinguishability for *uniform* PPT distinguishers only then below $\mathcal{D}$ and $\mathcal{A}$ need to be restricted to *uniform* PPT algorithms as well. Similarly, if we allow the distinguisher to be quantum, then $\mathcal{D}$ and $\mathcal{A}$ below may be quantum as well.

### 5.2 Achieving $\mathsf{sNR}^{H,\perp}$ in the Computational Setting

Here, we consider the computational variant of $\mathsf{sNR}^{H,\perp}$, where we restrict $\mathcal{D}^H, \mathcal{A}^H$ and $\mathsf{aux}$ to be PPT (oracle) algorithms. Furthermore, the entropy requirement (1) is replaced by

$$\underset{\substack{(\mathsf{sk},pk)\leftarrow\mathsf{KGen}^H\\ m\leftarrow\mathcal{D}^H(\mathsf{sk})}}{\mathsf{HILL}_\infty^H}(m \mid \mathsf{sk},\mathsf{aux}(\mathsf{sk},m)) \geq \omega(\log\lambda)\,, \tag{13}$$

and we then naturally demand that the game $\mathsf{sNR}^{H,\perp}$ can be won with negligible probability $\mathsf{negl}(\lambda)$ only.

*Remark 4.* Interestingly, and maybe somewhat surprisingly, in the computational setting $\mathsf{sNR}^{H,\perp}$ does not imply $\mathsf{NR}^{H,\perp}$, in contrast to the statistical setting, as explained in Sect. 2.3. Indeed, [13] showed that the BUFF transform does in general not satisfy $\mathsf{NR}^{H,\perp}$ in the computational setting, while below we show that it does satisfy $\mathsf{sNR}^{H,\perp}$. See Remark 1 for why our proof does not carry over to $\mathsf{NR}^{H,\perp}$. We suspect that the two notions are incomparable in the computational setting.

We get the following positive result on the computational $\mathsf{sNR}^{H,\perp}$ security of the BUFF transform $\mathsf{BUFF}[\mathcal{S},H]$.

**Theorem 4.** *Let $\mathcal{S} = (\mathsf{KGen},\mathsf{Sign}^H,\mathsf{Vrfy}^H)$ be a signature scheme in ROM, where* KGen *makes no query to $H$, and let $\mathsf{BUFF}[\mathcal{S},H]$ be the signature scheme obtained by applying the BUFF transform. Then for every PPT hint function* aux*, and for any PPT adversaries $\mathcal{D}^H,\mathcal{A}^H$ against $\mathsf{sNR}^{H,\perp}_{\mathsf{BUFF}[\mathcal{S},H]}$ that satisfy (13), we have*

$$\mathbf{Adv}^{\mathsf{sNR}^{H,\perp}}_{\mathsf{BUFF}[\mathcal{S},H]}(\mathcal{D},\mathcal{A},\mathsf{aux}) \leq \mathsf{negl}(\lambda)\,.$$

In spirit, we can recycle Lemma 1 to reduce the computational variant of $\mathsf{sNR}^{H,\perp}$ to the computational variant of Hide-and-Seek, and then we show in Lemma 4 that the latter is hard as well, which follows rather directly from the statistical hardness and the definition of the HILL entropy.

*Proof.* Take $(\bar{\mathcal{D}},\bar{\mathcal{A}})$ as in Lemma 1, for which

$$\mathbf{Adv}^{\mathsf{sNR}^{H,\perp}}_{\mathsf{BUFF}[\mathcal{S},H]} \leq \mathsf{poly}(\lambda)\cdot\mathbf{Adv}^{\mathsf{HnS}^H}(\bar{\mathcal{D}},\bar{\mathcal{A}}) + \mathsf{negl}(\lambda)\,,$$

where we exploit that the numbers of queries made by Sign, $\mathcal{D}$, and $\mathcal{A}$ are bounded by their (polynomial) running times, respectively, and that the additive term $q_K\epsilon$ in (5) vanishes due to the assumption that KGen makes no query to $H$. Hence it suffices to control the $\mathsf{HnS}^H$ advantage of $(\bar{\mathcal{D}},\bar{\mathcal{A}})$.

Towards this end, we first note that, by inspecting the construction of $\bar{\mathcal{D}}$ with $x = (\mathsf{pk},m)$ and $z = (\mathsf{sk},\mathsf{aux}(\mathsf{sk},m))$, the HILL entropy variant of (3) follows:

$$\underset{(x,z)\leftarrow\bar{\mathcal{D}}^H}{\mathsf{HILL}_\infty^H}(x \mid z) \geq k(\lambda) \iff \underset{\substack{(\mathsf{sk},pk)\leftarrow\mathsf{KGen}^H\\ m\leftarrow\mathcal{D}^H(\mathsf{sk})}}{\mathsf{HILL}_\infty^H}(m \mid \mathsf{sk},\mathsf{aux}(\mathsf{sk},m)) \geq k(\lambda)\,,$$

where the equivalence is due to the public key pk being *efficiently derivable* from its corresponding secret key sk, and so (6) also holds for the HILL entropy. Combining the above with (13), we obtain

$$\underset{(x,z)\leftarrow\bar{\mathcal{D}}^H}{\mathsf{HILL}_\infty^H}(x \mid z) \geq \omega(\log\lambda)\,.$$

Moreover, by Lemma 1, $\bar{\mathcal{D}}\colon \{\perp\} \to \mathcal{X} \times \mathcal{Z}$ with $\mathcal{Z} = \mathcal{SK} \times \mathcal{AUX}$. Hence

$$\log|\mathcal{Z}| = \log|\mathcal{SK}| + \log|\mathcal{AUX}| \leq \mathsf{poly}(\lambda)$$

due to both KGen and aux being poly-time. Finally, Lemma 1 ensures that $\bar{\mathcal{A}}$ is PPT whenever $\mathcal{A}$ is, which is satisfied by assumption. Thus, the assumptions for Lemma 4 below (the hardness of Hide-and-Seek in the computational setting) are all satisfied, and so

$$\mathbf{Adv}^{\mathsf{HnS}^H}(\bar{\mathcal{D}}, \bar{\mathcal{A}}) \leq \mathsf{negl}(\lambda) ,$$

which concludes the proof. □

The following provided the computational hardness of Hide-and-Seek.

**Lemma 4.** *Let $\mathcal{D}^H\colon \{\perp\} \to \mathcal{X} \times \mathcal{Z}$ and $\mathcal{A}^H : \mathcal{Y} \times \mathcal{Z} \to \mathcal{X}$ be adversaries against $\mathsf{HnS}^H$, with $\mathcal{A}$ being PPT, $\log|\mathcal{Z}| < \mathsf{poly}(\lambda)$, and*

$$\underset{(x,z)\leftarrow\mathcal{D}^H}{\mathsf{HILL}^H_\infty}(x \mid z) \geq \omega(\log\lambda) .$$

*Then $\mathbf{Adv}^{\mathsf{HnS}^H}(\mathcal{D}, \mathcal{A}) \leq \mathsf{negl}(\lambda)$.*

*Proof.* Let $(x, z) \leftarrow \mathcal{D}^H$. Via the entropy condition, there is an ($H$-dependent) random variable $x^* \in \mathcal{X}$ such that $\mathsf{guess}(x^* \mid H, z) \leq \mathsf{negl}(\lambda)$ and moreover $(x^*, z)$ and $(x, z)$ are computationally indistinguishable. Without loss of generality, we may assume $(x^*, z)$ is sampled via a (possibly unbounded) hider $\mathcal{D}^{*H}$. Now, inspect the displayed games $\mathsf{HnS}^H(\mathcal{D}, \mathcal{A})$ and $\mathsf{HnS}^H(\mathcal{D}^*, \mathcal{A})$ below.

```
$\mathsf{HnS}^H(\mathcal{D}, \mathcal{A})$
  1: $(x, z) \leftarrow \mathcal{D}^H$
  2: return $x = \mathcal{A}^H(H(x), z)$
```

```
$\mathsf{HnS}^H(\mathcal{D}^*, \mathcal{A})$:
  1: $(x^*, z) \leftarrow \mathcal{D}^{*H}$
  2: return $x^* = \mathcal{A}^H(H(x^*), z)$
```

By the computational indistinguishability, it follows that

$$|\mathbf{Adv}^{\mathsf{HnS}^H}(\mathcal{D}, \mathcal{A}) - \mathbf{Adv}^{\mathsf{HnS}^H}(\mathcal{D}^*, \mathcal{A})| \leq \mathsf{negl}(\lambda) .$$

Finally, we can apply Theorem 3 to the $\mathsf{HnS}^H$ adversaries $\mathcal{D}^*$ and $\mathcal{A}$, which satisfy the statistical entropy condition, and so we have $\mathbf{Adv}^{\mathsf{HnS}}(\mathcal{D}^*, \mathcal{A}) \leq \mathsf{negl}(\lambda)$. This concludes the proof. □

*Remark 5.* Interestingly, towards proving $\mathsf{sNR}^{H,\perp}$ of the BUFF transform in the *statistical* setting, as we did earlier in the paper, it would have been sufficient to show that the random oracle satisfies (the statistical variant of) HnS for a *query bounded* hider $\mathcal{D}$. However, for the above line of reasoning in the computational setting, it is essential that Theorem 2 holds for a query unbounded hider; indeed, above, $x^*$ may be arbitrarily dependent on $H$, and so might not be producible by a query bounded hider $\mathcal{D}^*$.

## 6 Conclusion

In the light of recent negative result on the notion of non-resignability in general, and the non-resignability of the BUFF transform in particular, we re-establish the non-resignability property for the original BUFF transform for the (almost) strongest notions of non-resignability that do not contradict any negative result. Our results cover both the statistical and the computational case, and both the classical and the quantum setting. This answers the pressing question left open in the recent works on the non-resignability of the BUFF transform.

One small gap that remains open from our work is to weaken the HILL entropy requirement in the computational setting to *computational unpredictability*, as considered in [14]. Having large HILL entropy implies computational unpredictability, but not the other way round. Thus, whether the BUFF transform satisfies the computational variant of $\mathsf{sNR}^{H,\perp}$ when the HILL entropy requirement is relaxed to computational unpredictability, remains open.

**Acknowledgments.** Yu-Hsuan Huang is supported by the Dutch Research Agenda (NWA) project HAPKIDO (Project No. NWA.1215.18.002), which is financed by the Dutch Research Council (NWO). Jyun-Jie Liao is supported by Eshan Chattopadhyay's NSF CAREER award 2045576. Patrick Struck acknowledges funding by the Hector Foundation II.

## A Proof of Theorem 1

*Proof.* In the classical case, combining Lemma 1 and Theorem 2, for $\mathcal{Z} = \mathcal{SK} \times \mathcal{AUX}$ and $q = q_A + q_S$ we obtain

$$\begin{aligned}\mathbf{Adv}^{\mathsf{sNR}^{H,\perp}}_{\mathsf{BUFF}[\mathcal{S},H]}(\mathcal{D},\mathcal{A},\mathsf{aux}) &\le q \cdot 4(q+1)(\log|\mathcal{Z}| + \log(1/\epsilon) + 1)\epsilon + q_K\epsilon + \frac{q_D+1}{|\mathcal{Y}|} \\ &\le 8(q+1)^2\big(\log|\mathcal{Z}| + \log(1/\epsilon)\big)\epsilon + q_K\epsilon + \frac{q_D+1}{|\mathcal{Y}|}\ ,\end{aligned}$$

where the second inequality exploits that $\log(1/\epsilon) \ge 1$. This concludes the classical bound.

Similarly, in the quantum case, combining Lemma 1 and Theorem 3, for $\mathcal{Z} = \mathcal{SK} \times \mathcal{AUX}$ and $q = q_A + q_S$ we obtain

$$\begin{aligned}\mathbf{Adv}^{\mathsf{sNR}^{H,\perp}}_{\mathsf{BUFF}[\mathcal{S},H]}(\mathcal{D},\mathcal{A},\mathsf{aux}) &\le 2q \cdot \sqrt{O\big((\log|\mathcal{Z}| + \log(1/\epsilon) + q)q\epsilon\big)} + q_K\epsilon + \frac{q_D+1}{|\mathcal{Y}|} \\ &\le O\Big(\sqrt{(\log|\mathcal{Z}| + \log(1/\epsilon) + q)q^3\epsilon}\Big) + q_K\epsilon + \frac{q_D+1}{|\mathcal{Y}|} \text{ as } \min(1/\epsilon, |\mathcal{Z}|, q) \to \infty,\end{aligned}$$

where the constants in the asymptotic bounds are absolute constants. Hence, there are absolute constants $n, C \ge 2$ such that

$$\mathbf{Adv}^{\mathsf{sNR}^{H,\perp}}_{\mathsf{BUFF}[\mathcal{S},H]}(\mathcal{D},\mathcal{A},\mathsf{aux}) \le C\sqrt{(\log|\mathcal{Z}| + \log(1/\epsilon) + q)q^3\epsilon} + q_K\epsilon + \frac{q_D+1}{|\mathcal{Y}|}\ ,$$

whenever $\min(1/\epsilon, |\mathcal{Z}|, q) \geq n$. In order to get a bound even when $|\mathcal{Z}| < n$, we increase $|\mathcal{AUX}|$ to $n|\mathcal{AUX}|$ without actually changing the algorithm $\mathsf{aux}$, and so get

$$\mathbf{Adv}^{\mathsf{sNR}^{H,\perp}}_{\mathsf{BUFF}[\mathcal{S},H]}(\mathcal{D}, \mathcal{A}, \mathsf{aux}) \leq C \cdot \sqrt{\left(\log \frac{|\mathcal{SK}| \cdot n|\mathcal{AUX}|}{\epsilon} + q\right) q^3\epsilon + q_K\epsilon + \frac{q_D+1}{|\mathcal{Y}|}}$$
$$\leq C\sqrt{2} \cdot \sqrt{\left(\log \frac{|\mathcal{SK}| \cdot |\mathcal{AUX}|}{\epsilon} + q\right) q^3\epsilon + q_K\epsilon + \frac{q_D+1}{|\mathcal{Y}|}}$$

whenever $\min(1/\epsilon, q) \geq n$, where the second inequality is via $q + \log n \leq q + n \leq 2q$. Finally, since $q \geq q_A$, the boundary condition of the above inequality can be relaxed to $\min(1/\epsilon, q_A) \geq n$. This concludes the proof. □

## B Proof of Lemma 2

*Proof.* First, we note that the input $z^\circ$ can be omitted, as it can be hardwired into $\mathcal{A}$.

For the case $k = 1$, consider $H'$ to be a fresh random oracle, independent of $H$. Then, the distributions of $\mathcal{A}^H(H(x^u))$ and $\mathcal{A}^{H'}(H(x^u))$ coincide, unless a query of $\mathcal{A}$ to $H$ happens to be a query on $x^u$, which happens with probability at most $\frac{q}{|\mathcal{X}|}$. Thus

$$\Pr\big[x^u = \mathcal{A}^H(H(x^u))\big] \leq \Pr\big[x^u = \mathcal{A}^{H'}(H(x^u))\big] + \frac{q}{|\mathcal{X}|} \leq \frac{q+1}{|\mathcal{X}|}\,.$$

For the case $k > 1$, instead of considering $\mathcal{A}^H(H(x^u_k))$, the run of $\mathcal{A}$ on the $k$-th instance, we consider a run of $\mathcal{A}^H_k(H(x^u_k), T_{k-1})$, specified as follows. $\mathcal{A}_k$ is given as additional input the collection $T_{k-1}$ of transcripts of the runs of $\mathcal{A}$ on the previous instances $x^u_1, \ldots, x^u_{k-1}$; this includes each instance $x^u_i$ and its hash $H(x^u_i)$, as well as all the hash queries and responses of these $k-1$ runs of $\mathcal{A}$. $\mathcal{A}_k$ then simply runs $\mathcal{A}$, but whenever $\mathcal{A}$ is about to query $H$ on an input that is contained in $T_{k-1}$, it reads out the hash from there, instead of querying $H$. $\mathcal{A}^H_k(H(x^u_k), T_{k-1})$ then obviously behaves identically to $\mathcal{A}^H(H(x^u_k))$. Furthermore, conditioned on any fixed $T_{k-1}$, the distributions of $\mathcal{A}^H_k(H(x^u_k), T_{k-1})$ and $\mathcal{A}^{H'}_k(H'(x^u_k), T_{k-1})$ coincide, where again $H'$ is a fresh random oracle, unless $x^u_k$ happens to be contained in $T_{k-1}$, which happens with probability $\frac{(k-1)(q+1)}{\mathcal{X}}$. Thus,

$$\begin{aligned}
&\Pr\big[x^u_k = \mathcal{A}^H(H(x^u_k)) \mid x^u_i = \mathcal{A}^H(H(x^u_i))\, \forall\, i < k\big] \\
&\quad = \Pr\big[x^u_k = \mathcal{A}^H_k(H(x^u_k), T_{k-1}) \,\big|\, x^u_i = \mathcal{A}^H(H(x^u_i))\, \forall\, i < k\big] \\
&\quad \leq \Pr\big[x^u_k = \mathcal{A}^{H'}_k(H'(x^u_k), T_{k-1}) \,\big|\, x^u_i = \mathcal{A}^H(H(x^u_i))\, \forall\, i < k\big] + (k-1)\frac{q+1}{|\mathcal{X}|} \\
&\quad \leq k\frac{q+1}{|\mathcal{X}|}
\end{aligned}$$

where the last inequality follows from the fact for any fixed choice of $T_{k-1}$, we are back to the case $k = 1$ due to the freshness of $H'$. Multiplying these probability gives the claimed bound. □

## References

1. Ambainis, A., Hamburg, M., Unruh, D.: Quantum security proofs using semi-classical oracles. In: Boldyreva, A., Micciancio, D. (eds.) CRYPTO 2019. LNCS, vol. 11693, pp. 269–295. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-26951-7_10
2. Aulbach, T., Düzlü, S., Meyer, M., Struck, P., Weishäupl, M.: Hash your keys before signing: BUFF security of the additional NIST PQC signatures. In: Saarinen, M.J., Smith-Tone, D. (eds.) PQCrypto 2024. LNCS, vol. 14772, pp. 301–335. Springer, Cham (2024). https://doi.org/10.1007/978-3-031-62746-0_13
3. Ayer, A.: Duplicate signature key selection attack in let's encrypt. https://www.agwa.name/blog/post/duplicate_signature_key_selection_attack_in_lets_encrypt (2015)
4. Baecher, P., Fischlin, M., Schröder, D.: Expedient non-malleability notions for hash functions. In: Kiayias, A. (ed.) CT-RSA 2011. LNCS, vol. 6558, pp. 268–283. Springer, Heidelberg (2011)
5. Bellare, M., Rogaway, P.: Random oracles are practical: a paradigm for designing efficient protocols. In: Denning, D.E., Pyle, R., Ganesan, R., Sandhu, R.S., Ashby, V. (eds.) ACM CCS 93, pp. 62–73. ACM Press (1993)
6. Blake-Wilson, S., Menezes, A.: Unknown key-share attacks on the station-to-station (STS) protocol. In: Imai, H., Zheng, Y. (eds.) PKC'99. LNCS, vol. 1560, pp. 154–170. Springer, Heidelberg (1999)
7. Boneh, D., Dagdelen, Ö., Fischlin, M., Lehmann, A., Schaffner, C., Zhandry, M.: Random oracles in a quantum world. In: Lee, D.H., Wang, X. (eds.) ASIACRYPT 2011. LNCS, vol. 7073, pp. 41–69. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-25385-0_3
8. Chung, K.-M., Guo, S., Liu, Q., Qian, L.: Tight quantum time-space tradeoffs for function inversion. In: 2020 IEEE 61st Annual Symposium on Foundations of Computer Science (FOCS), pp. 673–684 (2020)
9. Coretti, S., Dodis, Y., Guo, S., Steinberger, J.: Random oracles and non-uniformity. In: Nielsen, J.B., Rijmen, V. (eds.) EUROCRYPT 2018. LNCS, vol. 10820, pp. 227–258. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-78381-9_9
10. Cremers, C., Düzlü, S., Fiedler, R., Fischlin, M., Janson, C.: BUFFing signature schemes beyond unforgeability and the case of post-quantum signatures. In: 2021 IEEE Symposium on Security and Privacy, pp. 1696–1714. IEEE Computer Society Press (2021). Cryptology ePrint Archive version https://eprint.iacr.org/archive/2020/1525/20230116:141028 (Version 1.3)
11. Cremers, C., Düzlü, S., Fiedler, R., Fischlin, M., Janson, C.: BUFFing signature schemes beyond unforgeability and the case of post-quantum signatures, 2023. An updated version (Version 1.4.1) of [10], https://eprint.iacr.org/archive/2020/1525/20231023:114351
12. Dodis, Y., Guo, S., Katz, J.: Fixing cracks in the concrete: random oracles with auxiliary input, revisited. In: Coron, J.-S., Nielsen, J.B. (eds.) EUROCRYPT 2017. LNCS, vol. 10211, pp. 473–495. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-56614-6_16

13. Don, J., Fehr, S., Huang, Y.-H., Struck, P.: On the (in)security of the BUFF transform. In: Reyzin, L., Stebila, D. (eds.) CRYPTO 2024. LNCS, vol. 14920, pp. 246–275. Springer, Cham (2024). https://doi.org/10.1007/978-3-031-68376-3_8
14. Düzlü, S., Fiedler, R., Fischlin, M.: BUFFing FALCON without increasing the signature size. In: SAC 2024. LNCS (2024, to appear). Cryptology ePrint Archive version available at https://eprint.iacr.org/2024/710
15. Håstad, J., Impagliazzo, R., Levin, L.A., Luby, M.: A pseudorandom generator from any one-way function. SIAM J. Comput. **28**(4), 1364–1396 (1999)
16. Hsiao, C.-Y., Lu, C.-J., Reyzin, L.: Conditional computational entropy, or toward separating pseudoentropy from compressibility. In: Naor, M. (ed.) EUROCRYPT 2007. LNCS, vol. 4515, pp. 169–186. Springer, Heidelberg (2007). https://doi.org/10.1007/978-3-540-72540-4_10
17. Jackson, D., Cremers, C., Cohn-Gordon, K., Sasse, R.: Seems legit: automated analysis of subtle attacks on protocols that use signatures. In: Cavallaro, L., Kinder, J., Wang, X., Katz, J. (eds.) ACM CCS 2019, pp. 2165–2180. ACM Press (2019)
18. Kim, T.H.-J., Basescu, C., Jia, L., Lee, S.B., Hu, Y.-C., Perrig, A.: Lightweight source authentication and path validation. In: Proceedings of the 2014 ACM Conference on SIGCOMM, pp. 271–282 (2014)
19. Menezes, A., Smart, N.: Security of signature schemes in a multi-user setting. In Designs, Codes and Cryptography (2004)
20. National Institute of Standards and Technology. Call for additional digital signature schemes for the post-quantum cryptography standardization process (2022). https://csrc.nist.gov/csrc/media/Projects/pqc-dig-sig/documents/call-for-proposals-dig-sig-sept-2022.pdf
21. Pornin, T., Stern, J.P.: Digital signatures do not guarantee exclusive ownership. In: Ioannidis, J., Keromytis, A., Yung, M. (eds.) ACNS 2005. LNCS, vol. 3531, pp. 138–150. Springer, Heidelberg (2005). https://doi.org/10.1007/11496137_10
22. Prest, T., et al.: FALCON. Technical report, National Institute of Standards and Technology (2020). https://csrc.nist.gov/projects/post-quantum-cryptography/post-quantum-cryptography-standardization/round-3-submissions

# From One-Time to Two-Round Reusable Multi-signatures Without Nested Forking

Lior Rotem[1], Gil Segev[2(✉)], and Eylon Yogev[3]

[1] Computer Science Department, Stanford University, Stanford, USA
lrotem@cs.stanford.edu

[2] School of Computer Science and Engineering, Hebrew University of Jerusalem, Jerusalem, Israel
segev@cs.huji.ac.il

[3] Department of Computer Science, Bar-Ilan University, Ramat Gan, Israel
eylon.yogev@biu.ac.il

**Abstract.** Multi-signature schemes are gaining significant interest due to their blockchain applications. Of particular interest are two-round schemes in the plain public-key model that offer key aggregation, and whose security is based on the hardness of the DLOG problem. Unfortunately, despite substantial recent progress, the security proofs of the proposed schemes provide rather insufficient concrete guarantees (especially for 256-bit groups). This frustrating situation has so far been approached either by relying on the security of seemingly-stronger assumptions or by considering restricted classes of attackers (e.g., algebraic attackers, which are assumed to provide an algebraic justification of each group element that they produce).

We present a complementing approach by constructing multi-signature schemes that satisfy two relaxed notions of security, whose applicability nevertheless ranges from serving as drop-in replacements to enabling expressive smart contract validation procedures. Our first notion, *one-time unforgeability*, extends the analogous single-signer notion by considering attackers that obtain a single signature for some message and set of signers of their choice. We construct a *non-interactive* one-time scheme based on any ring-homomorphic one-way function, admitting efficient instantiations based on the DLOG and RSA assumptions. Aggregated verification keys and signatures consist of two group

L. Rotem—Supported by a research grant from Protocol Labs.
G. Segev—Supported by the Israel Science Foundation (Grant No. 1336/22) and by the European Union (ERC, FTRC, 101043243). Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Research Council. Neither the European Union nor the granting authority can be held responsible for them.
E. Yogev—Supported by the Israel Science Foundation (Grant No. 2302/22), European Research Union (ERC, CRYPTOPROOF, 101164375), and by an Alon Young Faculty Fellowship. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Research Council. Neither the European Union nor the granting authority can be held responsible for them.

elements and a single group element, respectively, and our security proof consists of a *single* application of the forking lemma (thus avoiding the substantial security loss exhibited by the proposed two-round schemes). Additionally, we demonstrate that our scheme naturally extends to a $t$-time scheme, where aggregated verification keys consist of $t+1$ group elements, while aggregated signatures still consist of a single group element.

Our second notion, *single-set unforgeability*, considers attackers that obtain *any polynomial number* of signatures but are restricted to a single set of signers of their choice. We transform any non-interactive one-time scheme into a two-round single-set scheme via a novel forking-free construction that extends the seminal Naor-Yung tree-based approach to the multi-signer setting. Aggregated verification keys are essentially identical to those of the underlying one-time scheme, and the length of aggregated signatures is determined by that of the underlying scheme while scaling linearly with the length of messages (noting that long messages can always be hashed using a collision-resistant function). Instantiated with our one-time scheme, we obtain aggregated verification keys and signatures whose lengths are completely independent of the number of signers.

## 1 Introduction

A multi-signature scheme [IN83, BN06] enables any set of signers, within a large and potentially permissionless system, to jointly produce a compact signature on a given message. Research on the design and analysis of multi-signature schemes has recently gained significant renewed interest, as such schemes were found particularly suitable for blockchain applications. These range from drop-in replacements for standard signatures (e.g., [BDN18, MPS+19]), to smart contracts with expressive multi-owner validation procedures (e.g., [BWG+21, Arg22, Sta23]).

**Two Breakthroughs: Plain PK Model and Key Aggregation.** The high-level of suitability exhibited by multi-signatures to blockchain applications follows mostly due to two major breakthroughs. First, Bellare and Neven [BN06] showed that multi-signature schemes can provide security in the plain public-key model, capturing a realistic and permissionless environment, while not compromising on practicality. Specifically, in this model, each signer locally produces their signing and verification keys, without engaging in an interactive key-generation process with other signers or with a registration authority, and without augmenting verification keys with proofs of knowledge that need to be individually verified by all other signers. Second, Boneh, Drijvers and Neven [BDN18] and Maxwell, Poelstra, Seurin and Wuille [MPS+19] showed that multi-signature schemes can support non-interactive aggregation of verification keys (in the plain public-key model). Therefore, once an aggregated verification key has been verified to correspond to a particular set of signers, any dependence on the number of signers during all future signature verifications may be completely eliminated. This essentially turns the verification of multi-signatures as practical as (and even fully compatible with) that of standard signatures.

Following up on earlier constructions (e.g., [OO91,LHL94,MOR01,Bol03, LOS+06,BGO+07,RY07] and the references therein), this significant progress has led to a host of recent multi-signature schemes [DEF+19,NRS+20,AB21, BD21,NRS21,BTT22,DOT+22,FSZ22,LK23,PW23,TZ23]. Of particular interest in the blockchain setting are two-round schemes in the plain public-key model whose security is based on the hardness of the discrete logarithm (DLOG) problem in prime-order groups, as such schemes may admit practical implementations over standard elliptic curves, such as Secp256k1 or Curve25519.

**Concrete Security Guarantees in 256-bit Groups?** As observed by Bellare and Dai [BD21] (and most recently also by Pan and Wagner [PW23]), despite the substantial recent efforts, the existing proofs that base the security of the proposed two-round schemes in the plain public-key model on the hardness of the DLOG problem provide rather insufficient concrete guarantees in 256-bit groups. As common for DLOG-based signatures, these proofs of security are based on the classic forking lemma [PS00,BN06], but instead of relying on a single application of the lemma, they rely on two nested applications – leading to a substantial loss in the provable concrete security.

At a high level, under the widely-accepted assumption that the success probability of any $t$-time algorithm in solving the DLOG problem in a group of order $p$ is at most $t^2/p$ [Sho97], proofs of security that rely on two nested applications of the forking lemma seem limited to bounding the success probability of $t$-time attackers with roughly $\left(t^2/p\right)^{1/4}$. For a 256-bit prime $p$, such a bound falls short of providing sufficient concrete security guarantees, especially when compared to the $\left(t^2/p\right)^{1/2}$ bound resulting from a single application of the forking lemma (as is the case, for example, with Schnorr signatures, and more generally with signature schemes obtained from identification protocols via the Fiat-Shamir transform [FS86,Sch91,AAB+02,KMP16]).[1]

This frustrating situation has so far been approached for DLOG-based schemes in the plain public-key model via two relaxations: Either relying on the security of a recently-introduced stronger variant of the DLOG assumption (the interactive XIDL assumption introduced by Bellare and Dai [BD21]), or by proving security with respect to restricted classes of attackers [AB21, BD21,NRS21,LK23] (most notably, algebraic attackers, which are assumed to provide an algebraic justification of each group element that they produce [FKL18,AHK20,BFL20,FPS20,MTT19,RS20]). On the one hand, these relaxations have indeed led to tighter concrete security bounds. On the other hand, however, the extent to which the concrete security bounds resulting from these relaxations capture the security of the relevant schemes relative to hardness of the DLOG problem, is naturally rather limited.

[1] For simplicity, in the above discussion we did not include additional factors that depend on the number of random-oracle queries and signing queries issued by attackers, but rather focused mainly on the dependence on the order $p$ of the group.

### 1.1 Our Contributions

We present a complementing approach for designing multi-signature schemes in the plain public-key model and obtaining a better understanding of their security: Instead of relying on recently-introduced assumptions or considering restricted classes of attackers via the algebraic group model, we construct multi-signature schemes that satisfy relaxed notions of security. Our approach relaxes the full-fledged notion of security for multi-signature schemes in the plain public-key model by restricting attackers' abilities to obtain signatures of their choice. This leads us to formalizing two notions of security, *one-time unforgeability* and *single-set unforgeability*, and to construct schemes that satisfy them. Although our notions are not as strong as the full-fledged one, their applicability nevertheless ranges from serving as drop-in replacements to enabling expressive smart contract validation procedures, as we discuss in Sect. 1.2.

**One-Time Multi-signatures.** Our notion of one-time unforgeability naturally extends the analogous single-signer notion to the multi-signer setting by considering attackers that obtain a single signature for some message and set of signers of their choice. We construct a multi-signature scheme satisfying this notion of security in the random-oracle model based on any ring-homomorphic one-way function, admitting instantiations based on the DLOG and RSA assumptions [CD98, CFG+15].[2]

Our scheme's aggregated verification keys consist of two group elements, and when compared to the known two-round multi-signature schemes that satisfy the full-fledged notion of security for such schemes based on the hardness of the DLOG problem without relying on the algebraic group model [BD21, NRS21, TZ23], our scheme offers:[3] (1) non-interactive signing, (2) aggregate signatures that consist of a single group element, and (3) security proof that consists of a single application of the forking lemma and thus avoids the substantial security loss resulting from two nested applications (without restricting adversaries to algebraic ones). In particular, when relying on the hardness of the DLOG problem, we recover the above-discussed $\left(t^2/p\right)^{1/2}$ bound, similarly to single-signer Schnorr signatures. Although here we focus mainly on the dependence on the order $p$ of the group, we note that our concrete bound includes additional terms that depend on the number of random-oracle queries issued by attackers. In addition, we demonstrate that our scheme naturally extends to a $t$-time scheme, where aggregated verification keys consist of $t+1$ group elements, while aggregated signatures still consist of a single group element.

**Single-Set Multi-signatures.** Our notion of single-set unforgeability considers attackers that obtain *any polynomial number* of signatures for messages of their choice, but are restricted to requesting all of these signatures with respect to

[2] We note that a notion of one-time unforgeability in the context of *aggregate* signatures was introduced by Boneh and Kim [BK20], as we discuss in Sect. 1.2.

[3] It is not clear how to compare the efficiency and concrete security guarantees of our one-time scheme to those of schemes that satisfy the full-fledged notion of security for multi-signature schemes. See Sect. 1.3 for more details.

a *single set* of signers. In this context, a single set of signers corresponds to a single vector of verification keys, which may be adversarially chosen based on the public parameters of the scheme and on the honestly-generated verification key that is attacked. As we discuss in Sect. 1.2, this notion already suffices for various applications of multi-signatures, such as validating blockchain transactions in a wide range of settings.

We show that any one-time multi-signature scheme with non-interactive signing can be transformed into a scheme satisfying our notion of single-set unforgeability, where the signing process of the resulting scheme consists of two rounds. Our transformation is obtained via a tree-based construction that relies on standard cryptographic tools, most notably on a non-interactive zero-knowledge proof system (we rely on these standard tools for overcoming the challenges that arise when extending the seminal Naor-Yung tree-based signature scheme [NY89] to the multi-signer setting).[4]

The length of the resulting scheme's verification keys and aggregated verification keys is independent of the number of signers, and the keys themselves are essentially identical to those of the underlying one-time scheme. The length of the resulting scheme's signatures and aggregated signatures is also independent of the number of signers, and determined by that of the underlying scheme while scaling linearly with the length of messages ($\ell\lambda$-bit signatures for $\ell$-bit messages, where $\lambda$ is the security parameter, as in the Naor-Yung transformation[5]). Instantiated with our one-time scheme, we obtain verification keys and signatures whose lengths are completely independent of the number of signers.

Our transformation demonstrates that at least for short messages there is no inherent and significant security loss when transforming one-time multi-signatures into single-set multi-signatures, and that our security loss essentially matches that of transforming single-user one-time signatures into re-usable ones.[6] Finally, We note that in the single-signer setting, tree-based signatures that utilize one-time signatures were initially mostly of foundational interest, whereas additional substantial efforts have demonstrated their practical applicability (see, for example, [BHH+15, AE18, BHK+19, HK22, KHR+22] and the references therein).

**One-Time Multi-signatures: Structure vs. Hardness.** Revisiting our one-time multi-signature scheme, it is quite noticeable that whereas one-time single-

[4] We emphasize that we rely on standard non-interactive zero-knowledge proofs, which can be realized based on well-studied falsifiable assumptions, and that we do not rely on succinct non-interactive arguments of knowledge (SNARKs) [Mic00, Gro10, GW11, BCI+13, BCS16]. In particular, we do not have any requirements regarding the length of the resulting proofs (they are not included in our signatures, and only play an intermediate role) and do not assume any form of proofs of knowledge.

[5] Without loss of generality, $\ell \leq \lambda$ as otherwise longer messages can first be hashed using a collision-resistant function.

[6] The work of Blazy, Kakvi, Kiltz, and Pan [BKK+15] presented a construction of re-usable signatures with a tight security reduction. However, their starting point was not a one-time signature scheme, but rather a Chameleon hash function, which is a significant more structured object.

signer signatures can be constructed based on any one-way function [Lam79], our multi-signer scheme is based on the more structured notion of a ring-homomorphic one-way function. Although such functions admit realizations based on standard number-theoretic assumptions [CD98, CFG+15], this raises the question of whether one-time multi-signatures require more structured forms of cryptographic hardness when compared to one-time single-signer signatures.

Addressing this fundamental question, we prove that one-time multi-signature schemes satisfying a natural property (which is satisfied by our one-time scheme) cannot be constructed in a fully black-box manner based on one-way functions. An interesting question for further research is whether this can be circumvented via a seemingly less-natural construction.

## 1.2 Overview of Our Approach

In this section we first briefly discuss the applicability of schemes that satisfy our notions of security. Then, we present a high-level overview of our constructions.

**The Applicability of One-Time and Single-Set Multi-signatures.** Our relaxed notions naturally serve as intermediate notions for obtaining a better understanding of the concrete security of full-fledged multi-signatures. At the same time, a direct application of one-time multi-signatures is for validating one-time transactions, such Bitcoin UTXOs [Nak09] (as described by Boneh and Kim [BK20] in the somewhat incomparable context of one-time aggregated signatures, which we discuss below). Specifically, a UTXO is spent after validating one or more signatures with respect to the verification keys committed in the UTXO. Once spent, it cannot be spent again, and the funds are transferred to a different UTXO. Thus, using a one-time multi-signature scheme for UTXOs can eliminate any dependence on the number of signers during verification, and to reduce both communication and storage cost. This comes at the cost of not using the same signing key for more than one UTXO, and this can be managed, for example, by deriving any number of one-time signing keys via a single master key for a pseudorandom function (thus, the one-time signing keys need not be stored, but can instead be reproduced upon demand).

A somewhat less direct application of one-time multi-signatures is for validating standard (i.e., reusable) transactions via account abstraction (e.g., [BWG+21, Arg22, Sta23]). At a high level, account abstraction (among its various features) enables smart contracts to offer arbitrary validation logic. Already in the single-signer setting, consider a smart contract whose storage includes a one-time verification key, and whenever the user provides a transaction they provide a signature with respect to the currently-stored verification key both on the provided input to the smart contract and on a newly-generated one-time verification key that the contract will store instead of its current one. This is motivated by the textbook path-based construction of signatures from one-time signatures [KL21], which is typically presented as a warm-up for the classic Naor-Yung tree-based construction [NY89]. Unlike the textbook construction, here the verification time and signature length do not scale with the number of previously-generated signatures, since each newly-generated verification key replaces its predecessor in the contract's storage.[7] Such a mechanism extends to the multi-signer setting in various ways using a one-time multi-signature scheme, where each transaction additionally updates the stored aggregated verification key. Here, the advantages of using a one-time multi-signature scheme with non-interactive signing and concrete security guarantees based on the hardness of the DLOG problem may be significant.

Finally, for our notion of single-set multi-signatures, where an adversary may observe any polynomial number of signatures for a single set of signers, account abstraction is again useful. Specifically, for any smart contract whose transactions require validating multiple signers, as long as each signer allocates a contract-specific signing key that is not used for any other purpose, then single-set security suffices (and there is no need for any key updates as with one-time multi-signatures). As discussed above, such contract-specific keys can be derived via a single master key, and thus do not have to be explicitly stored.

**Our One-Time Multi-signature Scheme.** The starting point of our one-time scheme is the Schnorr-based one-time signature scheme designed by Bellare and Shoup [BS08], which was extended by Boneh and Kim [BK20] to DLOG-based and lattice-based one-time *aggregate* signature schemes. Recall that aggregate signature schemes enable to aggregate signatures on *any* set of messages, whereas multi-signature schemes enable to aggregate signatures on the same message. As a result, the schemes of Boneh and Kim do not support aggregation of verification keys, and their verification time scales linearly with the number of signers. We use a different aggregation method that is tailored to aggregating signatures on the same message. Once such an aggregated key has been verified to correspond to a particular set of signers (e.g., in a preliminary phase as discussed above for blockchain transactions), this enables us to guarantee that the signing and verification operations are independent of the number of signers.

Our scheme is based on the abstract notion of a ring-homomorphic one-way function, introduced by Catalano, Fiore, Gennaro and Vamvourellis [CFG+15]. At a high level, we consider an efficiently-computable homomorphism $f : \mathcal{X} \to \mathcal{Y}$ for cyclic groups $\mathcal{X}$ and $\mathcal{Y}$ that allows computing linear operations "in the exponent" over a ring $\mathbb{K} = \mathbb{Z}_q$ for some prime $q$ (in the DLOG-based instantiation, the prime $q$ corresponds to the order of the cyclic groups[8], but in the RSA-based instantiation this is not the case – see Sect. 2.1 for a formal definition).

[7] It should be noted that, over time, such a mechanism may run into various synchronization issues since each signing key can be used only once. Such issues can be dealt with either by using a $t$-time multi-signature scheme, or by using a recovery mode (again, enabled by account abstraction) that allows key updates via a standard multi-signature scheme. Assuming that such issues are hopefully not-too-frequent, the overall efficiency of the validation procedure would be determined by that of the one-time scheme.

[8] In the DLOG-based instantiation, the homomorphism is simply the group exponentiation operation relative to a given generator.

Each signer in our scheme samples $x, r \leftarrow \mathcal{X}$, and sets $\mathsf{sk} = (x, r)$ and $\mathsf{vk} = (X, R) = (f(x), f(r))$ as their signing key and verification key, respectively. Then, a vector $\vec{\mathsf{vk}} = ((X_1, R_1), \ldots, (X_n, R_n))$ of verification keys, corresponding to $n$ signers, is aggregated by computing $\mathsf{aggvk} = (\mathsf{agg}X, \mathsf{agg}R) = (\prod_{i=1}^{n} X_i^{a_i}, \prod_{i=1}^{n} R_i^{a_i})$, where $(a_1, \ldots, a_n) = \mathsf{H}_1\left(\vec{\mathsf{vk}}\right)$ for a hash function $\mathsf{H}_1$. For any message $m$ and vector $\vec{\mathsf{vk}}$ of verification keys, each signer computes a signature $\sigma_i = r_i + \mathsf{H}_0(m, \mathsf{aggvk}) \cdot x_i$ for a hash function $\mathsf{H}_0$, and signatures are similarly aggregated by computing $\mathsf{agg}\sigma = \sum_{i=1}^{n} a_i \cdot \sigma_i$. In turn, this enables to verify an aggregate signature $\mathsf{agg}\sigma$ with respect to an aggregated verification key $\mathsf{aggvk} = (\mathsf{agg}X, \mathsf{agg}R)$ by checking whether $f(\mathsf{agg}\sigma) = (\mathsf{agg}X)^{\mathsf{H}_0(m, \mathsf{aggvk})} \cdot \mathsf{agg}R$.

Our security proof models the hash function $\mathsf{H}_0$ and $\mathsf{H}_1$ as random oracles, and reduces the task of breaking the one-time unforgeability of the scheme to that of breaking the one-wayness property of the ring-homomorphic function (see Definition 2.1). Specifically, given any attacker for our scheme, we construct an inverter that is given $X = f(x)$ for a randomly chosen $x$, and outputs a pair $(x', d)$ such that $f(x') = X^d$ and $d \neq 0$ (note that, in the DLOG-setting $d$ can always be efficiently inverted, and thus without loss of generality $d = 1$, but in the RSA-setting this is not the case). At a very high level, our proof programs the random oracle $\mathsf{H}_0$ for embedding the value $X$ in the honestly-generated verification key given as input to the attacker, while generating $R$ in a way that would be consistent with the response to the attacker's single signing query. Then, the proof relies on a single application of the forking lemma using the random oracle $\mathsf{H}_1$ for resampling the value of $a_i$ that corresponds to position of the honestly-generated verification key in the aggregated verification key with respect to which the attacker produces a forgery. Given two such forgeries, we are then able to efficiently produce $x'$ and $d$ as required. The proof naturally contains a variety of challenges for implementing this high-level idea, and we refer the reader to Sect. 4 for a complete and formal description.

**From One-Time to Single-Set Multi-signatures.** As mentioned above, our approach is inspired by the Naor-Yung transformation of a one-time signature scheme into a reusable one [NY89]. Recall that, in the Naor-Yung transformation, the signer implicitly holds a binary tree of exponential size, where each node of the tree is associated with a pair of one-time signing and verification keys. Specifically, for signing $\ell$-bit messages, the tree has $2^\ell$ leaves, where each leaf and each internal node $\alpha \in \{0, 1\}^{\leq \ell} \cup \{\varepsilon\}$ is associated with a pair $(\mathsf{sk}_\alpha, \mathsf{vk}_\alpha)$ of one-time keys.[9] The keys $\mathsf{sk}_\varepsilon$ and $\mathsf{vk}_\varepsilon$ corresponding to the root $\varepsilon$ serve as the signing key and verification key, respectively, and all other keys (and randomness that may be needed for using them) do not have to be explicitly generated or stored, but rather can be produced whenever needed using a pseudorandom function whose key is additionally included in the signing key.

For signing a message $m \in \{0, 1\}^\ell$, the signer first uses the signing key $\mathsf{sk}_m$ associated with the leaf correspond to the binary string $m$ for signing the message

[9] We denote by $\{0, 1\}^{\leq \ell}$ the set of all binary strings of length at most $\ell$, and by $\varepsilon$ the empty string.

$m$ itself. Then, it uses the signing keys positioned on the path connecting the root $\varepsilon$ to the leaf $m$ for certifying the path: For every $j \in \{0, \ldots, \ell-1\}$, the signer uses the key $\mathsf{sk}_{m|_j}$ associated with the internal node corresponding to the binary string $m|_j$ to sign the concatenation of the two verification keys $\mathsf{vk}_{m|_j 0}$ and $\mathsf{vk}_{m|_j 1}$ corresponding to its two children in the tree.[10] This makes sure that each signing key is used at most once, thus enabling to rely on the one-time security of the underlying scheme.

Equipped with the Naor-Yung transformation, let us attempt extending it to the multi-signer setting. As in the single-signer setting, suppose that each signer implicitly holds a tree, where each node $\alpha \in \{0,1\}^{\leq \ell} \cup \{\varepsilon\}$ is associated with a pair $(\mathsf{sk}_\alpha, \mathsf{vk}_\alpha)$ of keys for a one-time *multi-signature* scheme. The keys $\mathsf{sk}_\varepsilon$ and $\mathsf{vk}_\varepsilon$ corresponding to the root $\varepsilon$ again serve as the signing key and verification key, respectively, and all other keys are similarly generated using a pseudorandom function upon demand. Note that this structure enables to aggregate the root verification keys $\mathsf{vk}_\varepsilon^{(1)}, \ldots, \mathsf{vk}_\varepsilon^{(n)}$ of any $n$ signers by using the key-aggregation algorithm of the underlying one-time scheme. More generally, it enables to aggregate not only the root keys, but to implicitly define an *aggregated tree*. In the aggregated tree, each node $\alpha \in \{0,1\}^{\leq \ell} \cup \{\varepsilon\}$ is associated with an aggregated verification key $\mathsf{aggvk}_\alpha$ that is obtained by aggregating the one-time verification keys $\mathsf{vk}_\alpha^{(1)}, \ldots, \mathsf{vk}_\alpha^{(n)}$ associated with the node $\alpha$ in the $n$ individual trees.

This observation leads to the following elegant, yet insecure, two-round signing protocol. For signing a message $m$ with respect to signers with root verification keys $\mathsf{vk}_\varepsilon^{(1)}, \ldots, \mathsf{vk}_\varepsilon^{(n)}$, each signer first sends all other signers the $2\ell$ verification keys on the path leading from the signer's root to the leaf $m$. At this point, all signers know the verification keys $\mathsf{vk}_{m|_j b}^{(i)}$ for all $i \in [n]$, $j \in \{0, \ldots, \ell-1\}$ and $b \in \{0,1\}$. This enables each signer to compute the aggregated verification keys $\mathsf{aggvk}_{m|_j b}$ along the path from the root to the leaf $m$ in the aggregated tree. Now, for each level $j \in \{0, \ldots, \ell-1\}$, each signer uses their one-time signing key $\mathsf{sk}_{m|_j}^{(i)}$ to non-interactively compute a signature $\sigma_{m|_j}^{(i)}$ on the concatenation of the two aggregated verification keys $\mathsf{aggvk}_{m|_j 0}$ and $\mathsf{aggvk}_{m|_j 1}$ with respect to the signer set $\mathsf{vk}_{m|_j}^{(1)}, \ldots, \mathsf{vk}_{m|_j}^{(n)}$. Finally, each signer uses their one-time signing key $\mathsf{sk}_m^{(i)}$ to compute a signature $\sigma_m^{(i)}$ on the message $m$ with respect to the signer set $\mathsf{vk}_m^{(1)}, \ldots, \mathsf{vk}_m^{(n)}$. These signatures are then aggregated for each level $j \in \{1, \ldots, \ell\}$ to produce a signature that consists of $\ell$ aggregated one-time signatures. Note that both the length of the resulting signature and the time required to verify it scale linearly with the length of the message (as in the Naor-Yung scheme), but are completely independent of the number $n$ of signers.

At this point, we would like to argue that since each one-time signing key is used at most once, then we can rely on the security of the underlying one-time multi-signature scheme to claim that our tree-based scheme is secure against attackers issuing any polynomial number of signing queries. This argument fails,

[10] We denote by $m|_j$ the leftmost $j$ bits of a binary string $m$ (where $m|_0 = \varepsilon$), and we denote by $m|_j b$ the binary string obtained by concatenating the strings $m|_j$ and $b$.

however, if the attacker can request signatures with respect to more than one set of signers. The reason is that different sets of signers induce different aggregated trees. Consider an adversary that requests a signature from some signer $i$ with respect to a set $\mathcal{S}$, and then requests a signature from the same signer $i$ with respect to a different set $\mathcal{S}'$. The first signature includes a signature with respect to the one-time signing key $\mathsf{sk}_{\varepsilon}^{(i)}$ on information derived from the aggregated tree corresponding to $\mathcal{S}$, and the second signature includes a signature with respect to the same one-time signing key $\mathsf{sk}_{\varepsilon}^{(i)}$ on information derived from the aggregated tree corresponding to $\mathcal{S}'$. Since the underlying scheme is only guaranteed to be one-time secure, the security of the tree-based construction breaks down.

Still, one might hope that the construction is secure as long as the attacker issues signature queries with respect to a single set of signers, eliminating the issue we just described. This coincides with our single-set notion of security. However, this is not the case. This added restriction is insufficient because a root verification key may be used by malicious signers for different trees. That is, a malicious signer can still send different verification keys in the first round of two invocations of the signing protocol. This again results in two distinct aggregated trees (even though the two invocations of the signing protocol share the same set of signers).

We resolve this additional challenge by identifying signers not only with their root verification key, but also with a commitment containing a key for a pseudorandom function from which their entire tree is derived. Now, in the first round of the signing protocol, each signer sends all one-time verification keys on the path from their root to the respective leaf, while augmenting each such key with a non-interactive zero-knowledge proof asserting that it has been generated correctly. From a foundational standpoint, such proofs can be based on the existence of trapdoor functions[11] [FLS90]. For practical instantiations, these NIZK proofs can be based on one of the many recent efficient protocols[12]. Crucially, these proofs are not included as part of the resulting signature, but rather only serve as "proofs of semi-honest behavior" that enable each signer to continue to the second round of the signing protocol. This describes the high-level intuitive structure of our scheme, and we refer the reader to Sect. 5 for a complete and formal description.

## 1.3 Related Work

**DDH-Based Multi-signatures.** Whereas most of the work on multi-signatures in prime-order groups focused on DLOG-based schemes, several schemes were suggested also based on the DDH assumption (e.g. [LYG19, FH21, TSS+23, PW23]). As in the single-signer setting, DDH-based signatures may lead to tighter reductions. Although, when implemented in concrete groups,

[11] Technically speaking, these have to be certifiably injective and doubly-enhanced [BY96, Gol11, GR13, CL18].

[12] See [GS08, Gro16, BBB+18, BSBH+18, GWC19, CHM+20, XZS22, GLS+23] and the many references therein for a highly non-exhaustive list of examples.

such schemes typically do not offer the same efficiency guarantees as DLOG-based ones.

**DLOG-Based Two-Round Multi-signatures.** Existing DLOG-based two-round multi-signature schemes can be roughly divided into three categories: (1) schemes whose security is established in the algebraic group model [AB21,BD21, NRS21,LK23], (2) schemes whose security is established based on interactive variants of the DLOG problem (without relying on the algebraic group model) [BD21,NRS21], and (3) schemes whose security is established based on the standard DLOG problem (again, without relying on the algebraic group model) [BD21,NRS21,TZ23]. As discussed in Sect. 1.1, it is challenging to compare the efficiency and concrete security guarantees of our one-time scheme to those of the existing DLOG-based schemes, as these schemes satisfy the full-fledged notion of security for multi-signature schemes. If any comparison can be made, it would seem essential for focus on category 3, since the schemes in categories 1 and 2 seem to inherently avoid nested applications of the forking lemma.

Focusing on category 3, the schemes of Bellare and Dai [BD21], Nick, Ruffing and Seurin [NRS21], and Tessaro and Zhu [TZ23] rely on two nested applications of the forking lemma (whereas we rely on a single application), their verification keys consist of a single group element (whereas our verification keys consist of two group elements), and their signatures consist of two, two and three group elements, respectively (whereas our signatures consist of a single group element).

**Synchronized Multi-signatures.** A substantially different tree-based approach, both in terms of its goals and in terms of its structure, was recently presented by Fleischhacker, Simkin and Zhang [FSZ22]. They constructed a lattice-based multi-signature scheme in the *synchronized* model, where it is assumed that signers share a global notion of time, and the signing algorithm takes the current time step as an additional input. Most notably, it is additionally assumed that no signer signs more than one message per time step, and the goal is to aggregate signatures for the same message and *same time step*, *without knowing the set of signers in advance*. Thus, a signature may be aggregated together with those of any subset of other signers. As noted by Fleischhacker et al. such flexibility seems particularly useful in the blockchain setting for the task of confirming newly-generated blocks: Block validators may be synchronized by the number of the block that they sign, each validator does not sign more than one block in each time period, and validators do not know which of the other potential validators will actually participate.

In contrast to the synchronized model, we design our scheme in the plain public key model [BN06], where there is no global notion of time (or any other form of synchronization), and we aim at aggregating signatures not for a given time step but rather for a given set of signers that is specified during the signing process. In particular, our scheme guarantees that a signature may be aggregated only with a specific set of signers that is provided to the signing algorithm as input, while keeping the verification time independent of the number of signers.

From the technical perspective, as discussed above, our approach relies on trees of exponential size, which are never explicitly constructed in their entirety.

Each of the exponential number of leaves corresponds to a message-dedicated verification key for a one-time multi-signature scheme, and the path leading to each leaf is constructed upon demand using a pseudorandom function. In contrast, for supporting $T$ time periods, Fleischhacker et al. explicitly sample $T$ key pairs, and then compute a homomorphic Merkle tree with the corresponding $T$ verification keys as its leaves. As a result, their scheme seems limited to supporting only a polynomial number $T$ of time periods, and the efficiency of their key-generation algorithm scales linearly with $T$.

Finally, we note that since messages in their scheme are not signed with respect to a given set of signers (but rather a signature can be aggregated together with those of any subset of other signers), the notion of security required from their one-time primitive does not explicitly consider multiple signers. Specifically, Fleischhacker et al. introduce a notion of single-signer one-time key-homomorphic signatures, whereas we explicitly introduce a notion of one-time multi-signatures satisfying security guarantees tailored to the multi-signer setting, and our tree-based construction can rely on any multi-signature scheme that satisfies it.

### 1.4 Paper Organization

The remainder of this paper is organized as follows. First, in Sect. 2 we present main basic notions and the cryptographic primitives and tools that are used in this work. In Sect. 3 we formalize the notions of one-time and single-set security for multi-signature schemes. Then, in Sects. 4 and 5 we present our one-time and single-set multi-signature schemes, respectively, and state their security. Due to space limitations, we refer the reader to the full version of this work for some of our contributions (including formal proofs of security).

## 2 Preliminaries

In this section we present the main basic notions and the cryptographic primitives and tools that are used in this work. For an integer $n \in \mathbb{N}$ we denote by $[n]$ the set $\{1, \ldots, n\}$. For a distribution $X$ we denote by $x \leftarrow X$ the process of sampling a value $x$ from the distribution $X$. Similarly, for a set $\mathcal{X}$ we denote by $x \leftarrow \mathcal{X}$ the process of sampling a value $x$ from the uniform distribution over $\mathcal{X}$.

### 2.1 Ring-Homomorphic One-Way Functions

Our construction of a one-time multi-signature scheme relies on the notion of ring-homomorphic one-way functions, introduced by Catalano, Fiore, Gennaro and Vamvourellis [CFG+15]. This notion was presented by Catalano et al. as part of their framework for algebraic one-way functions, which is closely-related to the notion of group-homomorphic one-way functions introduced by Cramer and Damgård [CD98]. The notion considers function families $\mathsf{F} = (\mathsf{Setup}, \mathsf{Eval})$ where for any $\lambda \in \mathbb{N}$ and for any function index $\mathsf{Ind}$ produced by $\mathsf{F.Setup}(1^{\lambda})$ it holds that $\mathsf{F.Eval}(\mathsf{Ind}, \cdot) : \mathcal{X}_{\mathsf{Ind}} \to \mathcal{Y}_{\mathsf{Ind}}$ is an efficiently-computable homomorphism for cyclic groups $\mathcal{X}_{\mathsf{Ind}}$ and $\mathcal{Y}_{\mathsf{Ind}}$ that allows computing linear operations "in the exponent" over a ring $\mathbb{K}_{\mathsf{Ind}}$. For our purposes, we consider the specific case where $\mathbb{K}_{\mathsf{Ind}} = \mathbb{Z}_q$ for some prime $q = q(\mathsf{Ind})$, and note that already this case captures the known constructions based on the hardness of the discrete-logarithm problem (i.e., the exponentiation function in a cyclic group) and RSA problem (i.e., the RSA function) as we discuss below [CD98, CFG+15].

As formalize by Catalano et al. [CFG+15], such a function family $\mathsf{F}$ is *ring homomorphic* if there exists an efficient algorithm $\mathsf{Eval}'$ such that for any $\lambda \in \mathbb{N}$, $\mathsf{Ind}$ produced by $\mathsf{Setup}(1^\lambda)$, generators $h_1, \dots, h_m \in \mathcal{X}_{\mathsf{Ind}}$, vector of elements $\boldsymbol{W} = (W_1, \dots, W_\ell) \in \mathcal{X}_{\mathsf{Ind}}^\ell$ where $W_i = h_1^{\omega_i^{(1)}} \cdots h_m^{\omega_i^{(m)}} \cdot R_i$, for some $R_i \in \mathcal{X}_{\mathsf{Ind}}$ and some integers $\omega_i^{(j)} \in \mathbb{Z}$ (note that this decomposition may not be unique), and vector of integers $\boldsymbol{\alpha} \in \mathbb{Z}^\ell$, it holds that

$$\mathsf{Eval}'(\mathsf{Ind}, \boldsymbol{A}, \boldsymbol{W}, \boldsymbol{\Omega}, \boldsymbol{\alpha}) = h_1^{\langle \boldsymbol{\omega}^{(1)}, \boldsymbol{\alpha} \rangle} \cdots h_m^{\langle \boldsymbol{\omega}^{(m)}, \boldsymbol{\alpha} \rangle} \prod_{i=1}^{\ell} R_i^{\alpha_i}$$

where $\boldsymbol{A} = (A_1, \dots, A_m) \in \mathcal{Y}_{\mathsf{Ind}}^m$ is such that $A_i = F_{\mathsf{Ind}}(h_i)$, $\boldsymbol{\Omega} = \left(\omega_i^{(j)}\right)_{i,j} \in \mathbb{Z}^{\ell \times m}$, and each product $\langle \boldsymbol{\omega}^{(j)}, \boldsymbol{\alpha} \rangle$ in the exponent is computed over the ring $\mathbb{K}_{\mathsf{Ind}}$.

In terms of one-wayness, Catalano et al. formalized the following notion of *flexible one-wayness*, asking that given $X = \mathsf{F}_{\mathsf{Ind}}(x)$ for a uniformly distributed $x \in \mathcal{X}_{\mathsf{Ind}}$, it should be infeasible to output $x' \in \mathcal{X}_{\mathsf{Ind}}$ and $d \in \mathbb{K}_{\mathsf{Ind}}$ such that $\mathsf{F}_{\mathsf{Ind}}(x') = X^d$ and $d \neq 0_{\mathbb{K}_{\mathsf{Ind}}}$.

**Definition 2.1.** *A ring-homomorphic function family* $\mathsf{F} = (\mathsf{Setup}, \mathsf{Eval})$ *is* flexible one-way *if for every probabilistic polynomial-time algorithm $A$ there exists a negligible function $\nu(\cdot)$ such that*

$$\mathsf{Adv}_{\mathsf{F},A}^{\mathsf{hom\text{-}ow}}(\lambda) \overset{\mathsf{def}}{=} \Pr\left[\begin{array}{c} A(\mathsf{Ind}, X) = (x', d) \ s.t. \\ \mathsf{F}_{\mathsf{Ind}}(x') = X^d \ and \ d \neq 0_{\mathbb{K}_{\mathsf{Ind}}} \end{array}\right] \leq \nu(\lambda),$$

*where* $\mathsf{Ind} \leftarrow \mathsf{Setup}(1^\lambda)$, $x \leftarrow \mathcal{X}_{\mathsf{Ind}}$ *and* $X = \mathsf{F}_{\mathsf{Ind}}(x)$.

Note that if $\mathbb{K}_{\mathsf{Ind}} = \mathbb{Z}_q$, where $q$ is the order of $\mathcal{X}_\lambda$, then the above definition is equivalent to the standard notion of one-wayness. In particular, the hardness of computing discrete logarithms in cyclic groups is equivalent to the flexible one-wayness of the group exponentiation function in such groups. In addition, Catalano et al. [CFG+15] showed that the RSA function $x \to x^e \bmod N$ in the subgroup $\mathbb{QR}_N \subset \mathbb{Z}_N^*$ of quadratic residues (where $N$ is the product of two "safe primes" and thus $\mathbb{QR}_N$ is cyclic) is flexible one-way based on the RSA assumption. In this case, $\mathbb{K}_{\mathsf{Ind}} = \mathbb{Z}_e$ for any prime $e \geq 3$.

## 2.2 The Forking Lemma

The proof of security for our one-time multi-signature scheme relies on the "forking lemma" of Bellare and Neven [BN06] (following Pointcheval and Stern

[PS00]). Let $q \geq 1$ be an integer, and let $\mathcal{H}$, $\mathcal{X}$ and $\mathcal{Y}$ be a sets such that $|\mathcal{H}| \geq 2$. Let $A$ be a randomized algorithm that on input $(x, \vec{h}) \in \mathcal{X} \times \mathcal{H}^q$ returns either a pair $(i, y) \in [q] \times \mathcal{Y}$ or the dedicated symbol $\perp$. Let $F_A$ be an algorithm that takes inputs in $\mathcal{X}$ and returns either an output $(y, y') \in \mathcal{Y}^2$ or the dedicated symbol $\perp$, and is defined as follows:

1. Sample random coins $\rho \leftarrow \{0,1\}^*$ for $A$.
2. Sample $h_1, \ldots, h_q \leftarrow \mathcal{H}$ and compute $\mathsf{out}_1 = A(x, h_1, \ldots, h_q; \rho)$.
3. If $\mathsf{out}_1 = \perp$ then return $\perp$, and otherwise let $\mathsf{out}_1 = (i, y)$.
4. Sample $h'_i, \ldots, h'_q \leftarrow \mathcal{H}$ and compute $\mathsf{out}_2 = A(x, h_1, \ldots, h_{i-1}, h'_i, \ldots, h_q; \rho)$.
5. If $\mathsf{out}_2 = \perp$ then return $\perp$, and otherwise let $\mathsf{out}_2 = (i', y')$.
6. If $i' = i$ and $h_i \neq h'_i$ then return $(i, y, y')$, and otherwise return $\perp$.

The following lemma, due to Bellare and Neven [BN06], relates the probability that $F_A$ successfully provides an output (other than $\perp$) to the corresponding probability of $A$.

**Lemma 2.2** **([BN06]).** *For any algorithm $A$ and for any distribution $D$ over $\mathcal{X}$ it holds that*

$$\Pr_{x \leftarrow D}\left[F_A(x) \neq \perp\right] \geq \epsilon \cdot \left(\frac{\epsilon}{q} - \frac{1}{|\mathcal{H}|}\right),$$

*where*

$$\epsilon = \Pr_{\substack{x \leftarrow D \\ \vec{h} \leftarrow \mathcal{H}^q}}\left[A\left(x, \vec{h}\right) \neq \perp\right].$$

## 3 One-Time and Single-Set Security for Multi-signature Schemes

In this section we formalize our two notions of security for multi-signature schemes. These notions are obtained by relaxing the notion of security for multi-signature schemes, formalized by Bellare and Neven [BN06] and recently refined by Bellare and Dai [BD21], by restricting adversaries' abilities to obtain signatures of their choice. In what follows we briefly recall the syntax and correctness requirement of multi-signature schemes, and then formally present our notions of security in Sects. 3.1 and 3.2.

A multi-signature scheme is a six-tuple $\Pi = (\mathsf{Setup}, \mathsf{KG}, \mathsf{KAgg}, \mathsf{Sign}, \mathsf{SAgg}, \mathsf{Verify})$ of polynomial-time algorithms. The setup algorithm $\mathsf{Setup}$ receives as input the unary representation of the security parameter $\lambda \in \mathbb{N}$ and outputs public parameters $\mathsf{pp}$. The key-generation algorithm $\mathsf{KG}$ receives as input the public parameters $\mathsf{pp}$, and outputs a signing key $\mathsf{sk}$ and a verification key $\mathsf{vk}$. The key-aggregation algorithm $\mathsf{KAgg}$ is a deterministic algorithm that takes as input the public parameters $\mathsf{pp}$ and a vector of verification keys $\vec{\mathsf{vk}}$, and outputs an aggregated verification key $\mathsf{aggvk}$.[13] For schemes with non-interactive signing,

[13] For our framework we view collections of verification keys as vectors and not sets. Note that any set can be uniquely transformed into a vector by determining an order among its elements (e.g., lexicographic order).

the signing algorithm $\mathsf{Sign}$ receives as input the public parameters $\mathsf{pp}$, a signing key $\mathsf{sk}$, a vector $\vec{\mathsf{vk}}$ of verification keys, and a message $m$ that is taken from a message space $\mathcal{M}$, and outputs a signature $\sigma$. For schemes with interactive signing, the signing algorithm defines an interactive protocol by additionally receiving as input at each round the relevant party's internal state the communication produced by all other parties. The signature-aggregation algorithm $\mathsf{SAgg}$ is a deterministic algorithm that takes as input the public parameters $\mathsf{pp}$, a vector of verification keys $\vec{\mathsf{vk}}$, and a vector of signatures $\vec{\sigma}$, and outputs an aggregated signature $\sigma$. Finally, the verification algorithm $\mathsf{Verify}$ receives as input the public parameters $\mathsf{pp}$, an aggregated verification key $\mathsf{aggvk}$, a message $m$ and an aggregated signature $\sigma$, and outputs either 0 or 1.

In terms of correctness, we consider the following standard requirement, which we formalize for simplicity for schemes with non-interactive signing and then discuss its extension to schemes with interactive signing:

**Definition 3.1.** *A multi-signature scheme $\Pi = (\mathsf{Setup}, \mathsf{KG}, \mathsf{KAgg}, \mathsf{Sign}, \mathsf{SAgg}, \mathsf{Verify})$ with non-interactive signing over a message space $\mathcal{M} = \{\mathcal{M}_\lambda\}_{\lambda \in \mathbb{N}}$ is* perfectly correct *if for any polynomial $n = n(\cdot)$, security parameter $\lambda \in \mathbb{N}$, and message $m \in \mathcal{M}_\lambda$ it holds that*

$$\Pr\left[\mathsf{Verify}\left(\mathsf{pp}, \mathsf{KAgg}\left(\mathsf{pp}, \vec{\mathsf{vk}}\right), m, \mathsf{SAgg}\left(\mathsf{pp}, \vec{\mathsf{vk}}, \vec{\sigma}\right)\right) = 1\right] = 1$$

*for $\vec{\mathsf{vk}} = (\mathsf{vk}_1, \dots, \mathsf{vk}_n)$ and $\vec{\sigma} = (\sigma_1, \dots, \sigma_n)$, where the probability is taken over the choice of $\mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda)$, and over the choices of $(\mathsf{sk}_i, \mathsf{vk}_i) \leftarrow \mathsf{KG}(\mathsf{pp})$ and $\sigma_i \leftarrow \mathsf{Sign}\left(\mathsf{pp}, \mathsf{sk}_i, \vec{\mathsf{vk}}, m\right)$ for every $i \in [n]$.*

The above definition extends to schemes with interactive signing by letting $(\sigma_1, \dots, \sigma_n)$ denote the local output of each party in the interactive signing protocol, where each party $i \in [n]$ is provided with $\left(\mathsf{pp}, \mathsf{sk}_i, \vec{\mathsf{vk}}, m\right)$ as input. We refer the reader to the work of Bellare and Dai [BD21] for a formal treatment of the correctness requirement for schemes with interactive signing.

A standard relaxation of the above definition allows for a negligible error probability. Concretely, our single-set multi-signature scheme presented in Sect. 5 provides perfect correctness whenever the vector $\vec{\mathsf{vk}}$ consists of distinct verification keys. Since essentially any notion of security for signature schemes guarantees that collisions among honestly-generated verification keys may occur only with a negligible probability, this yields at most a negligible error probability.

## 3.1 One-Time Unforgeability

For presenting the notion of one-time unforgeability for multi-signature schemes, we focus on multi-signature schemes with non-interactive signing, and note that the following treatment naturally extends to schemes with interactive signing (our one-time multi-signature scheme in Sect. 4 has non-interactive signing).

This notion of security captures attacks in which an adversary may obtain a single signature of their choice. Specifically, we consider a security experiment in which the adversary first receives the public parameters $\mathsf{pp}$ of the scheme and an honestly-generated verification key $\mathsf{vk}$. Then, the adversary may issue a single signing query for some message $m$ with respect to some set of signers indicated via a vector $\vec{\mathsf{vk}} = (\mathsf{vk}_1, \ldots, \mathsf{vk}_n)$ of verification keys. Both the message $m$ and the vector $\vec{\mathsf{vk}}$ of verification keys may be arbitrarily chosen (in polynomial time) by the adversary based on the public parameters and the honest verification key. Next, the adversary obtains a signature $\sigma \leftarrow \mathsf{Sign}(\mathsf{pp}, \mathsf{sk}, \vec{\mathsf{vk}}, m)$ produced using the signing key $\mathsf{sk}$ corresponding to the honest verification key $\mathsf{vk}$ (note that if $\mathsf{vk} \notin \vec{\mathsf{vk}}$ then the signing algorithm may be defined to output $\perp$), and the adversary's goal is to output a non-trivial forgery $(\vec{\mathsf{vk}}^*, m^*, \mathsf{agg}\sigma^*)$. The non-triviality of the forgery is reflected in the fact that $\mathsf{vk} \in \vec{\mathsf{vk}}^*$ and $(\vec{\mathsf{vk}}^*, m^*) \neq (\vec{\mathsf{vk}}, m)$ (i.e., it could not have been directly obtained as the result of the signing query), and that the aggregated signature $\mathsf{agg}\sigma^*$ verifies correctly for the message $m^*$ with respect to the aggregate verification key corresponding to $\vec{\mathsf{vk}}^*$.

**Definition 3.2.** *Let $t = t(\lambda)$ and $\epsilon = \epsilon(\lambda)$ be functions of the security parameter $\lambda \in \mathbb{N}$. A non-interactive multi-signature scheme $\Pi =$ $(\mathsf{Setup}, \mathsf{KG}, \mathsf{KAgg}, \mathsf{Sign}, \mathsf{SAgg}, \mathsf{Verify})$ is* one-time $(t, \epsilon)$-unforgeable *if for any algorithm $A = (A_1, A_2)$ that runs in time at most $t$ it holds that*

$$\mathsf{Adv}^{\mathsf{1TimeMS}}_{\Pi,\mathcal{A}}(\lambda) \stackrel{\mathsf{def}}{=} \Pr\left[\mathsf{Exp}^{\mathsf{1TimeMS}}_{\Pi,A}(\lambda) = 1\right] \leq \epsilon(\lambda)$$

*for all sufficiently large $\lambda \in \mathbb{N}$, where the experiment $\mathsf{Exp}^{\mathsf{1TimeMS}}_{\Pi,A}(\lambda)$ is defined as follows:*

1. $\mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda)$.
2. $(\mathsf{sk}, \mathsf{vk}) \leftarrow \mathsf{KG}(\mathsf{pp})$.
3. $\left(\vec{\mathsf{vk}}, m, \mathsf{st}\right) \leftarrow A_1(1^\lambda, \mathsf{pp}, \mathsf{vk})$.
4. $\sigma \leftarrow \mathsf{Sign}\left(\mathsf{pp}, \mathsf{sk}, \vec{\mathsf{vk}}, m\right)$.
5. $\left(\vec{\mathsf{vk}}^*, m^*, \mathsf{agg}\sigma^*\right) \leftarrow A_2(\mathsf{st}, \sigma)$.
6. *If the following conditions are satisfied then output* $1$ *and otherwise output* $0$*:*
   (a) $\mathsf{vk} \in \vec{\mathsf{vk}}^*$.
   (b) $\left(\vec{\mathsf{vk}}^*, m^*\right) \neq \left(\vec{\mathsf{vk}}, m\right)$
   (c) $\mathsf{Verify}\left(\mathsf{pp}, \mathsf{KAgg}\left(\vec{\mathsf{vk}}^*\right), m^*, \mathsf{agg}\sigma^*\right) = 1$.

In some cases we omit the parameters $t$ and $\epsilon$ corresponding to the running time and success probability of adversaries, respectively, and consider polynomial-time adversaries with negligible success probabilities. In addition, when considering schemes whose security is analyzed in the random-oracle model [BR93], we augment all algorithms (including the adversary) with access to the random oracle, introduce an additional parameter $q_{\mathsf{H}}$ that upper bounds the number of direct random-oracle queries issued by the adversary, and consider all probabilities also over the randomness of the oracle.

## 3.2 Single-Set Unforgeability

Our notion of single-set unforgeability for multi-signature schemes considers adversaries that obtain any polynomial number of signatures of their choice but are restricted to requesting all of these signatures with respect to a single set of signers. In this context, a single set of signers corresponds to a single vector of verification keys, which may be adversarially chosen based on the public parameters of the scheme and on the honestly-generated verification key that the adversary is attacking.

Looking ahead, our construction of a multi-signature scheme that provides single-set security consists of a two-round signing protocol: Each party sends one message to all other parties, receives one message from all other parties, and then produces their output. Following Bellare and Dai [BD21], for formalizing the security of schemes with an interactive signing protocol, adversaries are provided access to a stateful signing oracle that enables to initiate sessions and to continue previously-initiated ones. Specifically, given a multi-signature scheme $\Pi$ with a two-round signing protocol $\mathsf{Sign} = (\mathsf{Sign}_1, \mathsf{Sign}_2)$, we denote by $\mathcal{O}_{\mathsf{Sign}}(\mathsf{pp}, \mathsf{sk}, \vec{\mathsf{vk}}, \cdot)$ the corresponding stateful signing oracle that is provided to the adversary, where $\mathsf{pp}$ denotes the public parameters, $\mathsf{sk}$ denotes the signing key corresponding to the honestly-generated verification key $\mathsf{vk}$ given as input to the adversary, and $\vec{\mathsf{vk}}$ denotes the vector of verification keys corresponding to the single set of signers chosen by the adversary. For this oracle, an adversary may issue two types of queries:

- Session-initiation queries: On query a message $m$, the oracle first assigns a unique session identifier $\mathsf{sid}$ (e.g., in an incremental manner) and computes $(\mathsf{msg}_1, \mathsf{st}) \leftarrow \mathsf{Sign}_1\left(\mathsf{pp}, \mathsf{sk}, \vec{\mathsf{vk}}, m\right)$. Then, it locally stores the pair $(\mathsf{sid}, \mathsf{st})$ and outputs $(\mathsf{sid}, \mathsf{msg}_1)$.
- Communication queries: On query a pair $(\mathsf{sid}, \mathsf{msg}_2)$, the oracle first retrieves the stored pair $(\mathsf{sid}, \mathsf{st})$, and then computes and outputs $\mathsf{out} \leftarrow \mathsf{Sign}_2(\mathsf{st}, \mathsf{msg}_2)$. If no stored pair exists for the session identifier $\mathsf{sid}$ then the oracle outputs $\perp$.

**Definition 3.3.** *Let $t = t(\lambda), q_{\mathsf{sign}} = q_{\mathsf{sign}}(\lambda)$, and $\epsilon = \epsilon(\lambda)$ be functions of the security parameter $\lambda \in \mathbb{N}$. A non-interactive multi-signature scheme $\Pi = (\mathsf{Setup}, \mathsf{KG}, \mathsf{KAgg}, \mathsf{Sign}, \mathsf{SAgg}, \mathsf{Verify})$ is* single-set $(t, q_{\mathsf{sign}}, \epsilon)$-unforgeable *if for any algorithm $A = (A_1, A_2)$ that runs in time at most $t$ and issues at most $q_{\mathsf{sign}}$ signing queries, it holds that*

$$\mathsf{Adv}^{\mathsf{SSetMS}}_{\Pi,A}(\lambda) \overset{\mathsf{def}}{=} \Pr\left[\mathsf{Exp}^{\mathsf{SSetMS}}_{\Pi,A}(\lambda) = 1\right] \leq \epsilon(\lambda)$$

*for all sufficiently large $\lambda \in \mathbb{N}$, where the experiment $\mathsf{Exp}^{\mathsf{SSetMS}}_{\Pi,A}(\lambda)$ is defined as follows:*

*1.* $\mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda)$.
*2.* $(\mathsf{sk}, \mathsf{vk}) \leftarrow \mathsf{KG}(\mathsf{pp})$.

*3.* $\left(\vec{\mathsf{vk}}, \mathsf{st}\right) \leftarrow A_1(1^\lambda, \mathsf{pp}, \mathsf{vk})$.

*4.* $\left(\vec{\mathsf{vk}}^*, m^*, \mathsf{agg}\sigma^*\right) \leftarrow A_2^{\mathcal{O}_{\mathsf{Sign}}(\mathsf{pp}, \mathsf{sk}, \vec{\mathsf{vk}}, \cdot)}(\mathsf{st})$.

*5. If the following conditions are satisfied then output* 1 *and otherwise output* 0*:*

*(a)* $\mathsf{vk} \in \vec{\mathsf{vk}}^*$.

*(b)* $\vec{\mathsf{vk}}^* \neq \vec{\mathsf{vk}}$ *or* $A_2$ *did not query* $\mathcal{O}_{\mathsf{Sign}}(\mathsf{pp}, \mathsf{sk}, \vec{\mathsf{vk}}, \cdot)$ *with* $m^*$.

*(c)* $\mathsf{Verify}\left(\mathsf{pp}, \mathsf{KAgg}\left(\vec{\mathsf{vk}}^*\right), m^*, \mathsf{agg}\sigma^*\right) = 1$.

As noted in Sect. 3.1, when considering schemes whose security is analyzed in the random-oracle model [BR93], we augment all algorithms (including the adversary) with access to the random oracle, introduce an additional parameter $q_{\mathsf{H}}$ that upper bounds the number of direct random-oracle queries issued by the adversary, and consider all probabilities also over the randomness of the oracle.

## 4 One-Time Multi-signatures via Ring Homomorphic One-Way Functions

In this section we describe our construction of a one-time multi-signature scheme and prove its security (in the full version of this work we also show that our construction naturally extends to a $t$-time multi-signature scheme without introducing any additional assumptions). Our construction is parameterized by the security parameter $\lambda \in \mathbb{N}$ and by an integer $n = n(\lambda)$ determining an upper bound on the size of supported signer sets. The construction relies on the following building blocks:

– A ring-homomorphic one-way function $\mathsf{F} = (\mathsf{F.Setup}, \mathsf{F.Eval})$. As formalized in Sect. 2.1 following Catalano et al. [CFG+15], recall that for any $\lambda \in \mathbb{N}$ and for any function index $\mathsf{Ind}$ produced by $\mathsf{F.Setup}(1^\lambda)$ it holds that $\mathsf{F.Eval}(\mathsf{Ind}, \cdot) : \mathcal{X}_{\mathsf{Ind}} \rightarrow \mathcal{Y}_{\mathsf{Ind}}$ is a homomorphism for cyclic groups $\mathcal{X}_{\mathsf{Ind}}$ and $\mathcal{Y}_{\mathsf{Ind}}$ that allows computing linear operations "in the exponent" over a ring $\mathbb{K}_{\mathsf{Ind}} = \mathbb{Z}_q$ for some prime $q = q(\mathsf{Ind}) \geq K(\lambda)$, where $q$ is at most the order of the cyclic groups. For simplifying our notation, for any function index $\mathsf{Ind}$ we let $\mathsf{F}_{\mathsf{Ind}}(\cdot) = \mathsf{F.Eval}(\mathsf{Ind}, \cdot)$.

– Hash functions $\mathsf{H}_0$ and $\mathsf{H}_1$ that will be modeled, for the security analysis, as random oracles. For any function index $\mathsf{Ind}$ of the ring-homomorphic function $\mathsf{F}$ we assume that $\mathsf{H}_0 : \mathcal{M}_\lambda \times \mathcal{Y}_{\mathsf{Ind}} \times \mathcal{Y}_{\mathsf{Ind}} \rightarrow \mathbb{K}_{\mathsf{Ind}}$ and $\mathsf{H}_1 : (\mathcal{Y}_{\mathsf{Ind}} \times \mathcal{Y}_{\mathsf{Ind}})^n \rightarrow \mathbb{K}_{\mathsf{Ind}}^n$, where $\mathcal{M} = \mathcal{M}_\lambda$ is the supported message space of the constructed multi-signature scheme (we can choose, for example, $\mathcal{M}_\lambda = \{0,1\}^\lambda$). In terms of input lengths, this means that we assume sufficiently long input lengths for $\mathsf{H}_0$ and $\mathsf{H}_1$. In terms of output lengths, recall that $\mathbb{K}_{\mathsf{Ind}} = \mathbb{Z}_q$ for some prime $q$, where $q$ is at most the order of the cyclic groups. In practice, this can be realized in a standard manner by producing sufficiently-long outputs using a cryptographic hash function, and then reducing the results modulo $q$ to obtain a statistically-small error.

In what follows we first describe our one-time scheme, denoted $\mathsf{1T}$, and then prove its correctness and security. For simplicity and for avoiding the introduction of additional notation, when describing the scheme and proving its security we assume that all signer sets are of size $n = n(\lambda)$, but we note that an upper bound on the size of supported signer sets would suffice. In addition, we note that, in practice our scheme does not essentially require any setup. Specifically, when realizing the ring-homomorphic one-way function via group exponentiation in a prime-order group [CD98, CFG+15], the function index $\mathsf{Ind}$ consists of the description of the group, which is typically fixed (e.g., standard 256-bit curves such as Secp256k1 or Curve25519).

**The scheme $\mathsf{1T} = (\mathsf{Setup}, \mathsf{KG}, \mathsf{KAgg}, \mathsf{Sign}, \mathsf{SAgg}, \mathsf{Verify})$**

**$\mathsf{Setup}(1^\lambda)$.** On input $1^\lambda$ the setup algorithm samples $\mathsf{Ind} \leftarrow \mathsf{F.Setup}(1^\lambda)$ and outputs $\mathsf{pp} = \mathsf{Ind}$.

**$\mathsf{KG}(\mathsf{pp})$.** On input $\mathsf{pp}$ as above, the key-generation algorithm samples $x, r \leftarrow \mathcal{X}_{\mathsf{Ind}}$, computes $X = \mathsf{F}_{\mathsf{Ind}}(x)$ and $R = \mathsf{F}_{\mathsf{Ind}}(r)$, and then outputs $(\mathsf{sk}, \mathsf{vk}) = ((x, r), (X, R))$.

**$\mathsf{KAgg}\left(\mathsf{pp}, \vec{\mathsf{vk}}\right)$.** On input $\mathsf{pp}$ as above and $\vec{\mathsf{vk}} = (\mathsf{vk}_1, \ldots, \mathsf{vk}_n)$, where $\mathsf{vk}_i = (X_i, R_i)$ for every $i \in [n]$, the key-aggregation algorithm computes

$$\begin{aligned}(a_1, \ldots, a_n) &= \mathsf{H}_1\left(\vec{\mathsf{vk}}\right) \in \mathbb{K}_{\mathsf{Ind}}^n \\ \mathsf{agg}X &= \prod_{i=1}^{n} X_i^{a_i} \in \mathcal{Y}_{\mathsf{Ind}} \\ \mathsf{agg}R &= \prod_{i=1}^{n} R_i^{a_i} \in \mathcal{Y}_{\mathsf{Ind}},\end{aligned}$$

and outputs $\mathsf{aggvk} = (\mathsf{agg}X, \mathsf{agg}R)$.

**$\mathsf{Sign}\left(\mathsf{pp}, \mathsf{sk}, \vec{\mathsf{vk}}, m\right)$.** On input $\mathsf{pp}$ as above, $\mathsf{sk} = (x, r)$, $\vec{\mathsf{vk}}$ and $m \in \mathcal{M}_\lambda$, the signing algorithm is defined as follows:

1. If $(\mathsf{F}_{\mathsf{Ind}}(x), \mathsf{F}_{\mathsf{Ind}}(r)) \notin \vec{\mathsf{vk}}$ then output $\perp$.
2. Otherwise, compute $\mathsf{aggvk} = \mathsf{KAgg}\left(\mathsf{pp}, \vec{\mathsf{vk}}\right)$, and output

$$\sigma = r + \mathsf{H}_0(m, \mathsf{aggvk}) \cdot x \in \mathcal{X}_{\mathsf{Ind}}.$$

**$\mathsf{SAgg}\left(\mathsf{pp}, \vec{\mathsf{vk}}, \vec{\sigma}\right)$.** On input $\mathsf{pp}$ as above, $\vec{\mathsf{vk}}$ and $\vec{\sigma} = (\sigma_1, \ldots, \sigma_n)$, the signature-aggregation algorithm computes $(a_1, \ldots, a_n) = \mathsf{H}_1\left(\vec{\mathsf{vk}}\right) \in \mathbb{K}_{\mathsf{Ind}}$ and outputs $\mathsf{agg}\sigma = \sum_{i=1}^{n} a_i \cdot \sigma_i \in \mathcal{X}_{\mathsf{Ind}}$.

**$\mathsf{Verify}(\mathsf{pp}, \mathsf{aggvk}, m, \mathsf{agg}\sigma)$.** On input $\mathsf{pp}$ as above, $\mathsf{aggvk} = (\mathsf{agg}X, \mathsf{agg}R)$, $m$ and $\mathsf{agg}\sigma$, if

$$\mathsf{F}_{\mathsf{Ind}}(\mathsf{agg}\sigma) = (\mathsf{agg}X)^{\mathsf{H}_0(m, \mathsf{aggvk})} \cdot \mathsf{agg}R$$

then the verification algorithm outputs 1 and otherwise it outputs 0.

**Correctness.** Fix any $\lambda \in \mathbb{N}$, any public parameters $\mathsf{pp} = \mathsf{Ind}$, and any vector of verification keys $\vec{\mathsf{vk}} = (\mathsf{vk}_1, \ldots, \mathsf{vk}_n)$. For every $i \in [n]$ let $\mathsf{vk}_i = (X_i, R_i)$, where $X_i = \mathsf{F}_{\mathsf{Ind}}(x_i)$ and $R_i = \mathsf{F}_{\mathsf{Ind}}(r_i)$. Then, for any message $m$ it holds that

$$\begin{aligned}
\mathsf{F}_{\mathsf{Ind}}(\mathsf{agg}\sigma) &= \mathsf{F}_{\mathsf{Ind}}\left(\sum_{i=1}^{n} a_i \cdot r_i + \mathsf{H}_0\left(m, \mathsf{aggvk}\right) \cdot \sum_{i=1}^{n} a_i \cdot x_i\right) \\
&= \left(\prod_{i=1}^{n} \mathsf{F}_{\mathsf{Ind}}(x_i)^{a_i}\right)^{\mathsf{H}_0(m, \mathsf{aggvk})} \cdot \left(\prod_{i=1}^{n} \mathsf{F}_{\mathsf{Ind}}(r_i)^{a_i}\right) \\
&= (\mathsf{agg}X)^{\mathsf{H}_0(m, \mathsf{aggvk})} \cdot \mathsf{agg}R \ ,
\end{aligned}$$

where $(a_1, \ldots, a_n) = \mathsf{H}_1\left(\vec{\mathsf{vk}}\right)$, and thus the scheme provides perfect correctness.

**Security.** The following theorem (which is proved in the full version of this work) establishes the security of our scheme based on that of the underling ring-homomorphic one-way function (recall Definition 2.1):

**Theorem 4.1.** *Let $A$ be a probabilistic polynomial-time algorithm that issues $q_{\mathsf{H}_0} = q_{\mathsf{H}_0}(\lambda)$ and $q_{\mathsf{H}_1} = q_{\mathsf{H}_1}(\lambda)$ queries to the oracles $\mathsf{H}_0$ and $\mathsf{H}_1$, respectively. Then, there exists a probabilistic polynomial-time algorithm $I$ such that for every $\lambda \in \mathbb{N}$ it holds that*

$$\mathsf{Adv}^{\mathsf{1TimeMS}}{}_{\mathsf{1T},A}(\lambda) \leq (q_{\mathsf{H}_0})^2 \cdot q_{\mathsf{H}_1} \cdot n \cdot \left(\sqrt{\mathsf{Adv}^{\mathsf{hom\text{-}ow}}_{\mathsf{F},I}(\lambda) + \frac{(q_{\mathsf{H}_0})^2 + (q_{\mathsf{H}_1})^2}{K(\lambda)}} + \frac{1}{K(\lambda)}\right)$$

## 5 From One-Time to Single-Set Multi-signatures

In this section we show that any one-time multi-signature scheme with non-interactive signing can be transformed into a multi-signature scheme that satisfies our notion of single-set unforgeability with a two-round signing protocol. Our construction relies on the following building blocks:

- A multi-signature scheme $\mathsf{1T} = (\mathsf{1T.Setup}, \mathsf{1T.KG}, \mathsf{1T.KAgg}, \mathsf{1T.Sign}, \mathsf{1T.SAgg}, \mathsf{1T.Verify})$ that is one-time unforgeable with non-interactive signing over a message space $\mathcal{M} = \{\mathcal{M}_\lambda\}_{\lambda \in \mathbb{N}}$. For simplicity we assume that $\mathcal{M}_\lambda = \{0,1\}^\ell$ for some polynomial $\ell = \ell(\lambda)$, and that $\ell$ is sufficiently large for enabling the scheme to sign, for example, the concatenation of two verification keys produced by its key-generation algorithm (otherwise, we can additionally rely on a collision-resistant hash function in the standard "hash-then-sign" manner). In addition, since the scheme has non-interactive signing, we assume without loss of generality that its signing algorithm is deterministic.
- A pseudorandom function $\mathsf{PRF} = (\mathsf{PRF.KG}, \mathsf{PRF.Eval})$, where for each $\lambda \in \mathbb{N}$ and for each key $\mathsf{k}$ produced by $\mathsf{PRF.KG}(1^\lambda)$ it holds that $\mathsf{PRF.Eval}(\mathsf{k}, \cdot) : \{0,1\}^{\leq \ell(\lambda)} \to \{0,1\}^{\ell'(\lambda)}$, and $\ell'(\lambda)$ is the length of the internal random string sampled by the key-generation algorithm $\mathsf{1T.KG}$.

- A collision-resistant hash function $\mathsf{CRH} = (\mathsf{CRH.KG}, \mathsf{CRH.Eval})$. As noted below, our usage of a collision-resistant hash function is in fact not required, and is done for providing a more direct proof of security.
- A non-interactive statistically-binding commitment $\mathsf{COM} = (\mathsf{COM.Setup}, \mathsf{COM.Commit}, \mathsf{COM.Verify})$.
- A non-interactive simulation-sound zero-knowledge proof system $\mathsf{ZK} = (\mathsf{ZK.Setup}, \mathsf{ZK.P}, \mathsf{ZK.V}, \mathsf{ZK.Sim}_1, \mathsf{ZK.Sim}_2)$ for the language $\mathcal{L} = \{\mathcal{L}_\lambda\}_{\lambda \in \mathbb{N}}$, where

$$\mathcal{L}_\lambda = \left\{ \begin{pmatrix} \mathsf{pp}_{\mathsf{1T}}, \mathsf{crs}_{\mathsf{com}}, \\ \mathsf{com}, x, \mathsf{vk} \end{pmatrix} : \begin{array}{c} \exists\, (\mathsf{decom}, \mathsf{k}_{\mathsf{PRF}}, \mathsf{sk}) \text{ s.t.} \\ \mathsf{COM.Verify}\,(\mathsf{crs}_{\mathsf{com}}, \mathsf{com}, \mathsf{decom}, \mathsf{k}_{\mathsf{PRF}}) = 1 \text{ and} \\ (\mathsf{sk}, \mathsf{vk}) = \mathsf{1T.KG}(\mathsf{pp}_{\mathsf{1T}}; \mathsf{PRF.Eval}(\mathsf{k}_{\mathsf{PRF}}, x)) \end{array} \right\}.$$

  Note that even if the security of the scheme $\mathsf{1T}$ is proved in the random-oracle model, then as long as the key-generation algorithm $\mathsf{1T.KG}$ does not access the random oracle (as with our scheme in Sect. 4), then $\mathcal{L}$ is an NP-language (assuming, of course, that $\mathsf{COM}$ is a standard-model commitment scheme, and that $\mathsf{PRF}$ is a standard-model pseudorandom function [Nao91, Gol01]). In this case, the single-set security of our construction is proved in the random-oracle model.

For presenting our scheme, we denote by $\varepsilon$ the empty string, we denote by $m|_i$ the leftmost $i$ bits of a binary string $m$ (where $m|_0 = \varepsilon$), and we denote by $m|_i b$ the binary string obtained by concatenating the strings $m|_i$ and $b$.

**The scheme $\Pi = (\mathsf{Setup}, \mathsf{KG}, \mathsf{KAgg}, \mathsf{Sign}, \mathsf{SAgg}, \mathsf{Verify})$**

**$\mathsf{Setup}(1^\lambda)$.** On input $1^\lambda$ the setup algorithm computes

$$\begin{aligned} \mathsf{pp}_{\mathsf{1T}} &\leftarrow \mathsf{1T.Setup}(1^\lambda) \\ \mathsf{crs}_{\mathsf{ZK}} &\leftarrow \mathsf{ZK.Setup}(1^\lambda) \\ \mathsf{crs}_{\mathsf{COM}} &\leftarrow \mathsf{COM.Setup}(1^\lambda) \\ \mathsf{k}_{\mathsf{CRH}} &\leftarrow \mathsf{CRH.KG}(1^\lambda), \end{aligned}$$

and returns $\mathsf{pp} = (\mathsf{pp}_{\mathsf{1T}}, \mathsf{crs}_{\mathsf{ZK}}, \mathsf{crs}_{\mathsf{COM}}, \mathsf{k}_{\mathsf{CRH}})$.

**$\mathsf{KG}(\mathsf{pp})$.** On input $\mathsf{pp}$ as above, the key-generation algorithm computes

$$\begin{aligned} (\mathsf{sk}_\varepsilon, \mathsf{vk}_\varepsilon) &\leftarrow \mathsf{1T.KG}(\mathsf{pp}_{\mathsf{1T}}) \\ \mathsf{k}_{\mathsf{PRF}} &\leftarrow \mathsf{PRF.KG}(1^\lambda) \\ (\mathsf{com}, \mathsf{decom}) &\leftarrow \mathsf{COM.Commit}\,(\mathsf{crs}_{\mathsf{COM}}, \mathsf{k}_{\mathsf{PRF}})\,, \end{aligned}$$

and returns $\mathsf{sk} = (\mathsf{sk}_\varepsilon, \mathsf{vk}_\varepsilon, \mathsf{k}_{\mathsf{PRF}}, \mathsf{com}, \mathsf{decom})$ and $\mathsf{vk} = (\mathsf{vk}_\varepsilon, \mathsf{com})$.

**$\mathsf{KAgg}\left(\mathsf{pp}, \vec{\mathsf{vk}}\right)$.** On input $\mathsf{pp}$ as above and

$$\vec{\mathsf{vk}} = \left(\left(\mathsf{vk}_\varepsilon^{(1)}, \mathsf{com}^{(1)}\right), \ldots, \left(\mathsf{vk}_\varepsilon^{(n)}, \mathsf{com}^{(n)}\right)\right)$$

the key-aggregation algorithm computes

$$\begin{aligned}\mathsf{aggvk}_{\varepsilon} &= \mathsf{1T.KAgg}\left(\mathsf{pp}_{\mathsf{1T}}, \left(\mathsf{vk}_{\varepsilon}^{(1)}, \ldots, \mathsf{vk}_{\varepsilon}^{(n)}\right)\right) \\ \mathsf{tag}_{\vec{\mathsf{vk}}} &= \mathsf{CRH.Eval}\left(\mathsf{k}_{\mathsf{CRH}}, \vec{\mathsf{vk}}\right),\end{aligned}$$

and returns $\mathsf{aggvk} = \left(\mathsf{aggvk}_{\varepsilon}, \mathsf{tag}_{\vec{\mathsf{vk}}}\right)$.

**Sign** $\left(\mathbf{pp}, \mathbf{sk}, \vec{\mathbf{vk}}, \boldsymbol{m}\right)$**.** On input $\mathsf{pp}$ as above, $\mathsf{sk} = (\mathsf{sk}_{\varepsilon}, \mathsf{vk}_{\varepsilon}, \mathsf{k}, \mathsf{com}, \mathsf{decom})$, $\vec{\mathsf{vk}}$ and $m \in \{0,1\}^{\ell}$, the signing algorithm proceeds as follows:

1. Let $\vec{\mathsf{vk}} = \left(\left(\mathsf{vk}_{\varepsilon}^{(1)}, \mathsf{com}^{(1)}\right), \ldots, \left(\mathsf{vk}_{\varepsilon}^{(n)}, \mathsf{com}^{(n)}\right)\right)$. If there is no index $i \in [n]$ for which $(\mathsf{vk}_{\varepsilon}, \mathsf{com}) = \left(\mathsf{vk}_{\varepsilon}^{(i)}, \mathsf{com}^{(i)}\right)$ or there is more than one such index, then abort. Otherwise, denote by $k^* \in [n]$ the unique such index.
2. For any $j \in \{0, \ldots, \ell-1\}$ and $b \in \{0,1\}$ compute

$$\begin{aligned}\left(\mathsf{sk}_{m|_j b}^{(k^*)}, \mathsf{vk}_{m|_j b}^{(k^*)}\right) &= \mathsf{1T.KG}(\mathsf{pp}_{\mathsf{1T}}; \mathsf{PRF.Eval}(\mathsf{k}, m|_j b)) \\ \pi_{m|_j b}^{(k^*)} &= \mathsf{ZK.P}\left(\mathsf{crs}_{\mathsf{ZK}}, \left(\mathsf{pp}_{\mathsf{1T}}, \mathsf{crs}_{\mathsf{com}}, \mathsf{com}, m|_j b, \mathsf{vk}_{m|_j b}^{(k^*)}\right), \right. \\ &\qquad \left. \left(\mathsf{decom}, \mathsf{k}, \mathsf{sk}_{m|_j b}^{(k^*)}\right)\right)\end{aligned}$$

   and send $\left\{\left(\mathsf{vk}_{m|_j b}^{(k^*)}, \pi_{m|_j b}^{(k^*)}\right)\right\}_{j \in \{0,\ldots,\ell-1\}, b \in \{0,1\}}$ to all other parties.
3. Upon receiving $\left\{\left(\mathsf{vk}_{m|_j b}^{(i)}, \pi_{m|_j b}^{(i)}\right)\right\}_{i \in [n]\setminus\{k^*\}, j \in \{0,\ldots,\ell-1\}, b \in \{0,1\}}$ from all other parties, if there exists $(i, j, b) \in ([n] \setminus \{k^*\}) \times \{0, \ldots, \ell-1\} \times \{0,1\}$ for which $\mathsf{vk}_{m|_j b}^{(i)} = \mathsf{vk}_{m|_j b}^{(k^*)}$ or

$$\mathsf{ZK.V}\left(\mathsf{crs}_{\mathsf{ZK}}, \left(\mathsf{pp}_{\mathsf{1T}}, \mathsf{crs}_{\mathsf{com}}, \mathsf{com}^{(i)}, m|_j b, \mathsf{vk}_{m|_j b}^{(i)}\right), \pi_{m|_j b}^{(i)}\right) = 0$$

   then abort. Otherwise, compute

$$\begin{aligned}\vec{\mathsf{vk}}_{m|_j b} &= \left(\mathsf{vk}_{m|_j b}^{(1)}, \ldots, \mathsf{vk}_{m|_j b}^{(n)}\right) \text{ for all } j \in \{0, \ldots, \ell-1\} \text{ and } b \in \{0,1\} \\ \mathsf{aggvk}_{m|_j b} &= \mathsf{1T.KAgg}\left(\mathsf{pp}_{\mathsf{1T}}, \vec{\mathsf{vk}}_{m|_j b}\right) \text{ for all } j \in \{0, \ldots, \ell-1\} \\ &\qquad\qquad\qquad\qquad \text{and } b \in \{0,1\} \\ \sigma_{m|_j}^{(k^*)} &= \mathsf{1T.Sign}\left(\mathsf{pp}_{\mathsf{1T}}, \mathsf{sk}_{m|_j}^{(k^*)}, \vec{\mathsf{vk}}_{m|_j}, \left(\mathsf{aggvk}_{m|_j 0}, \mathsf{aggvk}_{m|_j 1}\right)\right) \\ &\qquad\qquad\qquad\qquad\qquad \text{for all } j \in \{0, \ldots, \ell-1\} \\ \mathsf{tag}_{\vec{\mathsf{vk}}} &= \mathsf{CRH.Eval}\left(\mathsf{k}_{\mathsf{CRH}}, \vec{\mathsf{vk}}\right) \\ \sigma_m^{(k^*)} &= \mathsf{1T.Sign}\left(\mathsf{pp}_{\mathsf{1T}}, \mathsf{sk}_m^{(k^*)}, \vec{\mathsf{vk}}_m, \left(m, \mathsf{tag}_{\vec{\mathsf{vk}}}\right)\right)\end{aligned}$$

   and output

$$\sigma^{(k^*)} = \left(\left\{\left(\sigma_{m|_j}^{(k^*)}, \mathsf{vk}_{m|_j 0}^{(k^*)}, \mathsf{vk}_{m|_j 1}^{(k^*)}\right)\right\}_{j \in \{0,\ldots,\ell-1\}}, \sigma_m^{(k^*)}\right).$$

**$\mathsf{SAgg}\left(\mathsf{pp}, \vec{\mathsf{vk}}, \vec{\sigma}\right)$.** On input $\mathsf{pp}$ as above, $\vec{\mathsf{vk}}$ and $\vec{\sigma} = \left(\sigma^{(1)}, \ldots, \sigma^{(n)}\right)$ where

$$\sigma^{(i)} = \left(\left\{\left(\sigma^{(i)}_{m|_j}, \mathsf{vk}^{(i)}_{m|_j 0}, \mathsf{vk}^{(i)}_{m|_j 1}\right)\right\}_{j \in \{0,\ldots,\ell-1\}}, \sigma^{(i)}_m\right)$$

for every $i \in [n]$, the signature-aggregation algorithm computes

$$\begin{aligned}\mathsf{aggvk}_{m|_j b} &= \mathsf{1T.KAgg}\left(\mathsf{pp}_{\mathsf{1T}}, \left(\mathsf{vk}^{(1)}_{m|_j b}, \ldots, \mathsf{vk}^{(n)}_{m|_j b}\right)\right) \text{ for all } j \in \{0,\ldots,\ell-1\} \\ &\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \text{and } b \in \{0,1\} \\ \mathsf{agg}\sigma_{m|_j} &= \mathsf{1T.SAgg}\left(\mathsf{pp}_{\mathsf{1T}}, \vec{\mathsf{vk}}_{m|_j}, \left(\sigma^{(1)}_{m|_j}, \ldots, \sigma^{(n)}_{m|_j}\right)\right) \text{ for all } j \in \{0,\ldots,\ell\},\end{aligned}$$

and outputs

$$\mathsf{agg}\sigma = \left(\left\{\left(\mathsf{agg}\sigma_{m|_j}, \mathsf{aggvk}_{m|_j 0}, \mathsf{aggvk}_{m|_j 1}\right)\right\}_{j \in \{0,\ldots,\ell-1\}}, \mathsf{agg}\sigma_m\right).$$

**$\mathsf{Verify}(\mathsf{pp}, \mathsf{aggvk}, m, \mathsf{agg}\sigma)$.** On input $\mathsf{pp}$ as above, $\mathsf{aggvk} = \left(\mathsf{aggvk}_\varepsilon, \mathsf{tag}_{\vec{\mathsf{vk}}}\right)$, $m \in \{0,1\}^\ell$ and $\mathsf{agg}\sigma$, where

$$\mathsf{agg}\sigma = \left(\left\{\left(\mathsf{agg}\sigma_{m|_j}, \mathsf{aggvk}_{m|_j 0}, \mathsf{aggvk}_{m|_j 1}\right)\right\}_{j \in \{0,\ldots,\ell-1\}}, \mathsf{agg}\sigma_m\right),$$

the verification algorithm outputs 1 if and only if the following two requirements are satisfied:

1. $\mathsf{1T.Verify}\left(\mathsf{pp}_{\mathsf{1T}}, \mathsf{aggvk}_{m|_j}, \left(\mathsf{aggvk}_{m|_j 0}, \mathsf{aggvk}_{m|_j 1}\right), \mathsf{agg}\sigma_{m|_j}\right) = 1$ for every $j \in \{0,\ldots,\ell-1\}$.
2. $\mathsf{1T.Verify}\left(\mathsf{pp}_{\mathsf{1T}}, \mathsf{aggvk}_m, \left(m, \mathsf{tag}_{\vec{\mathsf{vk}}}\right), \mathsf{agg}\sigma_m\right) = 1$.

The following theorem (which is proved in the full version of this work) captures the security of the scheme $\Pi$ based on that on its underlying building blocks. Here, we note that our usage of a collision-resistant hash function for computing the tags $\mathsf{tag}_{\vec{\mathsf{vk}}} = \mathsf{CRH.Eval}\left(\mathsf{k}_{\mathsf{CRH}}, \vec{\mathsf{vk}}\right)$ is in fact not required, and is done for providing a more direct proof of security. Specifically, we could have instead used $\mathsf{tag}_{\vec{\mathsf{vk}}} = \mathsf{aggvk}_\varepsilon$, and rely on the fact that the key aggregation of any one-time multi-signature scheme is collision resistant (see the full version for more details).

**Theorem 5.1.** *Let $\epsilon_{\mathsf{binding}} = \epsilon_{\mathsf{binding}}(\lambda)$, $q_{\mathsf{Sign}} = q_{\mathsf{Sign}}(\lambda)$ and $\ell = \ell(\lambda)$ be functions of the security parameter $\lambda \in \mathbb{N}$, and let $A$ be a probabilistic polynomial-time algorithm that issues $q_{\mathsf{Sign}}$ signing queries for $\ell$-bit messages. Then, assuming that $\mathsf{COM}$ is $\epsilon_{\mathsf{binding}}$-statistically binding, there exist probabilistic polynomial-time algorithms $B_1, \ldots, B_6$ such that for every $\lambda \in \mathbb{N}$ it holds that*

$$\begin{aligned}\mathsf{Adv}^{\mathsf{SSetMS}}_{\Pi,A}(\lambda) \leq\ & \epsilon_{\mathsf{binding}}(\lambda) + \mathsf{Adv}^{\mathsf{zk}}_{\mathsf{ZK},B_1}(\lambda) + \mathsf{Adv}^{\mathsf{hiding}}_{\mathsf{COM},B_2}(\lambda) \\ & + \mathsf{Adv}^{\mathsf{prf}}_{\mathsf{PRF},B_3}(\lambda) + \mathsf{Adv}^{\mathsf{ss}}_{\mathsf{ZK},B_4}(\lambda) + \mathsf{Adv}^{\mathsf{crh}}_{\mathsf{CRH},B_5}(\lambda) \\ & + (2 \cdot \ell \cdot q_{\mathsf{Sign}} + 1) \cdot \mathsf{Adv}^{\mathsf{1TimeMS}}_{\mathsf{1T},B_6}(\lambda).\end{aligned}$$

## References

[AAB+02] Abdalla, M., An, J.H., Bellare, M., Namprempre, C.: From identification to signatures via the fiat-shamir transform: minimizing assumptions for security and forward-security. In: Knudsen, L.R. (ed.) EUROCRYPT 2002. LNCS, vol. 2332, pp. 418–433. Springer, Heidelberg (2002). https://doi.org/10.1007/3-540-46035-7_28

[AB21] Kılınç Alper, H., Burdges, J.: Two-round trip Schnorr multi-signatures via delinearized witnesses. In: Malkin, T., Peikert, C. (eds.) CRYPTO 2021. LNCS, vol. 12825, pp. 157–188. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-84242-0_7

[AE18] Aumasson, J.-P., Endignoux, G.: Improving stateless hash-based signatures. In: Smart, N.P. (ed.) CT-RSA 2018. LNCS, vol. 10808, pp. 219–242. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-76953-0_12

[AHK20] Agrikola, T., Hofheinz, D., Kastner, J.: On instantiating the algebraic group model from falsifiable assumptions. In: Canteaut, A., Ishai, Y. (eds.) EUROCRYPT 2020. LNCS, vol. 12106, pp. 96–126. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-45724-2_4

[Arg22] Argent. Part I: WTF is account abstraction (2022). https://www.argent.xyz/blog/wtf-is-account-abstraction/

[BBB+18] Bünz, B., Bootle, J., Boneh, D., Poelstra, A., Wuille, P., Maxwell, G.: Bulletproofs: short proofs for confidential transactions and more. In: IEEE Symposium on Security and Privacy, pp. 315–334 (2018)

[BCI+13] Bitansky, N., Chiesa, A., Ishai, Y., Paneth, O., Ostrovsky, R.: Succinct non-interactive arguments via linear interactive proofs. In: Sahai, A. (ed.) TCC 2013. LNCS, vol. 7785, pp. 315–333. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-36594-2_18

[BCS16] Ben-Sasson, E., Chiesa, A., Spooner, N.: Interactive oracle proofs. In: Hirt, M., Smith, A. (eds.) TCC 2016. LNCS, vol. 9986, pp. 31–60. Springer, Heidelberg (2016). https://doi.org/10.1007/978-3-662-53644-5_2

[BD21] Bellare, M., Dai, W.: Chain reductions for multi-signatures and the HBMS scheme. In: Tibouchi, M., Wang, H. (eds.) ASIACRYPT 2021. LNCS, vol. 13093, pp. 650–678. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-92068-5_22

[BDN18] Boneh, D., Drijvers, M., Neven, G.: Compact multi-signatures for smaller blockchains. In: Peyrin, T., Galbraith, S. (eds.) ASIACRYPT 2018. LNCS, vol. 11273, pp. 435–464. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-03329-3_15

[BFL20] Bauer, B., Fuchsbauer, G., Loss, J.: A classification of computational assumptions in the algebraic group model. In: Micciancio, D., Ristenpart, T. (eds.) CRYPTO 2020. LNCS, vol. 12171, pp. 121–151. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-56880-1_5

[BFM88] Blum, M., Feldman, P., Micali, S.: Non-interactive zero-knowledge and its applications. In: Proceedings of the 20th Annual ACM Symposium on Theory of Computing, pp. 103–112 (1988)

[BGO+07] Boldyreva, A., Gentry, C., O'Neill, A., Yum, D.H.: Ordered multisignatures and identity-based sequential aggregate signatures, with applications to secure routing. In: Proceedings of the ACM Conference on Computer and Communications Security, pp. 276–285 (2007)

[BHH+15] Bernstein, D.J., et al.: SPHINCS: practical stateless hash-based signatures. In: Oswald, E., Fischlin, M. (eds.) EUROCRYPT 2015. LNCS, vol. 9056, pp. 368–397. Springer, Heidelberg (2015). https://doi.org/10.1007/978-3-662-46800-5_15

[BHK+19] Bernstein, D.J., Hülsing, A., Kölbl, S., Niederhagen, R., Rijneveld, J., Schwabe, P.: The SPHINCS$^+$ signature framework. : Proceedings of the ACM Conference on Computer and Communications Security (CCS), pp. 2129–2146. ACM (2019)

[BK20] Boneh, D., Kim, S.: One-time and interactive aggregate signatures from lattices (2020). https://crypto.stanford.edu/~skim13/agg_ots.pdf

[BKK+15] Blazy, O., Kakvi, S.A., Kiltz, E., Pan, J.: Tightly-secure signatures from chameleon hash functions. In: Katz, J. (ed.) PKC 2015. LNCS, vol. 9020, pp. 256–279. Springer, Heidelberg (2015). https://doi.org/10.1007/978-3-662-46447-2_12

[BN06] Bellare, M., Neven, G.: Multi-signatures in the plain public-key model and a general forking lemma. In: Proceedings of the ACM Conference on Computer and Communications Security, pp. 390–399 (2006)

[Bol03] Boldyreva, A.: Threshold signatures, multisignatures and blind signatures based on the gap-Diffie-Hellman-group signature scheme. In: Desmedt, Y.G. (ed.) PKC 2003. LNCS, vol. 2567, pp. 31–46. Springer, Heidelberg (2003). https://doi.org/10.1007/3-540-36288-6_3

[BR93] Bellare, M., Rogaway, P.: Random oracles are practical: a paradigm for designing efficient protocols. In: Proceedings of the 1st ACM Conference on Computer and Communications Security, pp. 62–73 (1993)

[BS08] Bellare, M., Shoup, S.: Two-tier signatures from the Fiat-Shamir transform, with applications to strongly unforgeable and one-time signatures. IET Inf. Secur. **2**(2), 47–63 (2008)

[BSBH+18] Ben-Sasson, E., Bentov, I., Horesh, Y., Riabzev, M.: Scalable, transparent, and post-quantum secure computational integrity. Cryptology ePrint Archive, Paper 2018/046 (2018)

[BSM+91] Blum, M., Santis, A.D., Micali, S., Persiano, G.: Non-interactive zero-knowledge. SIAM J. Comput. **20**(6), 1084–1118 (1991)

[BTT22] Boschini, C., Takahashi, A., Tibouchi, M.: MuSig-L: lattice-based multi-signature with single-round online phase. In: Dodis, Y., Shrimpton, T. (eds.) CRYPTO 2022. LNCS, vol. 13508, pp. 276–305. Springer, Cham (2022). https://doi.org/10.1007/978-3-031-15979-4_10

[BWG+21] Buterin, V., et al.: EIP-4337: Account abstraction using Alt mempool [DRAFT]. Ethereum Improvement Proposals, no. 4337 (2021). https://eips.ethereum.org/EIPS/eip-4337)

[BY96] Bellare, M., Yung, M.: Certifying permutations: noninteractive zero-knowledge based on any trapdoor permutation. J. Cryptol. **9**(3), 149–166 (1996)

[CD98] Cramer, R., Damgård, I.: Zero-knowledge proofs for finite field arithmetic, or: can zero-knowledge be for free? In: Krawczyk, H. (ed.) CRYPTO 1998. LNCS, vol. 1462, pp. 424–441. Springer, Heidelberg (1998). https://doi.org/10.1007/BFb0055745

[CFG+15] Catalano, D., Fiore, D., Gennaro, R., Vamvourellis, K.: Algebraic (trapdoor) one-way functions: constructions and applications. Theor. Comput. Sci. **592**, 143–165 (2015)

[CHM+20] Chiesa, A., Hu, Y., Maller, M., Mishra, P., Vesely, N., Ward, N.: Marlin: preprocessing zkSNARKs with universal and updatable SRS. In: Canteaut, A., Ishai, Y. (eds.) EUROCRYPT 2020. LNCS, vol. 12105, pp. 738–768. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-45721-1_26

[CL18] Canetti, R., Lichtenberg, A.: Certifying trapdoor permutations, revisited. In: Proceedings of the 16th Theory of Cryptography Conference, pp. 476–506 (2018)

[DEF+19] Drijvers, M., et al.: On the security of two-round multi-signatures. In: IEEE Symposium on Security and Privacy, pp. 1084–1101 (2019)

[DOT+22] Damgård, I., Orlandi, C., Takahashi, A., Tibouchi, M.: Two-round $n$-out-of-$n$ and multi-signatures and trapdoor commitment from lattices. J. Cryptol. **35**(2), 14 (2022)

[FH21] Fukumitsu, M., Hasegawa, S.: A tightly secure DDH-based multisignature with public-key aggregation. Int. J. Network. Comput. **11**(2), 319–337 (2021)

[FKL18] Fuchsbauer, G., Kiltz, E., Loss, J.: The algebraic group model and its applications. In: Shacham, H., Boldyreva, A. (eds.) CRYPTO 2018. LNCS, vol. 10992, pp. 33–62. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-96881-0_2

[FLS90] Feige, U., Lapidot, D., Shamir, A.: Multiple non-interactive zero knowledge proofs based on a single random string. In: Proceedings of the 31st Annual IEEE Symposium on Foundations of Computer Science, pp. 308–317 (1990)

[FPS20] Fuchsbauer, G., Plouviez, A., Seurin, Y.: Blind Schnorr signatures and signed ElGamal encryption in the algebraic group model. In: Canteaut, A., Ishai, Y. (eds.) EUROCRYPT 2020. LNCS, vol. 12106, pp. 63–95. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-45724-2_3

[FS86] Fiat, A., Shamir, A.: How to prove yourself: practical solutions to identification and signature problems. In: Odlyzko, A.M. (ed.) CRYPTO 1986. LNCS, vol. 263, pp. 186–194. Springer, Heidelberg (1987). https://doi.org/10.1007/3-540-47721-7_12

[FSZ22] Fleischhacker, N., Simkin, M., Zhang, Z.: Squirrel: efficient synchronized multi-signatures from lattices. In: Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security (CCS), pp. 1109–1123 (2022)

[GLS+04] Gennaro, R., Leigh, D., Sundaram, R., Yerazunis, W.: Batching Schnorr identification scheme with applications to privacy-preserving authorization and low-bandwidth communication devices. In: Lee, P.J. (ed.) ASIACRYPT 2004. LNCS, vol. 3329, pp. 276–292. Springer, Heidelberg (2004). https://doi.org/10.1007/978-3-540-30539-2_20

[GLS+23] Golovnev, A., Lee, J., Setty, S., Thaler, J., Wahby, R.S.: Brakedown: linear-time and field-agnostic SNARKs for R1CS. In: Handschuh, H., Lysyanskaya, A. (eds.) CRYPTO 2023. LNCS, vol. 14082, pp. 193–226. Springer, Cham (2023). https://doi.org/10.1007/978-3-031-38545-2_7

[Gol01] Goldreich, O.: Foundations of Cryptography – Volume 1: Basic Techniques. Cambridge University Press, Cambridge (2001)

[Gol11] Goldreich, O.: In a world of P=BPP. In: Goldreich, O. (ed.) Studies in Complexity and Cryptography. Miscellanea on the Interplay between Randomness and Computation. LNCS, vol. 6650, pp. 191–232. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-22670-0_20

[GOS06] Groth, J., Ostrovsky, R., Sahai, A.: Non-interactive zaps and new techniques for NIZK. In: Dwork, C. (ed.) CRYPTO 2006. LNCS, vol. 4117, pp. 97–111. Springer, Heidelberg (2006). https://doi.org/10.1007/11818175_6

[GR13] Goldreich, O., Rothblum, R.D.: Enhancements of trapdoor permutations. J. Cryptol. **26**(3), 484–512 (2013)

[Gro06] Groth, J.: Simulation-sound NIZK proofs for a practical language and constant size group signatures. In: Lai, X., Chen, K. (eds.) ASIACRYPT 2006. LNCS, vol. 4284, pp. 444–459. Springer, Heidelberg (2006). https://doi.org/10.1007/11935230_29

[Gro10] Groth, J.: Short non-interactive zero-knowledge proofs. In: Abe, M. (ed.) ASIACRYPT 2010. LNCS, vol. 6477, pp. 341–358. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-17373-8_20

[Gro16] Groth, J.: On the size of pairing-based non-interactive arguments. In: Fischlin, M., Coron, J.-S. (eds.) EUROCRYPT 2016. LNCS, vol. 9666, pp. 305–326. Springer, Heidelberg (2016). https://doi.org/10.1007/978-3-662-49896-5_11

[GS08] Groth, J., Sahai, A.: Efficient non-interactive proof systems for bilinear groups. In: Smart, N. (ed.) EUROCRYPT 2008. LNCS, vol. 4965, pp. 415–432. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-78967-3_24

[GW11] Gentry, C., Wichs, D.: Separating succinct non-interactive arguments from all falsifiable assumptions. In: Proceedings of the 43rd Annual ACM Symposium on Theory of Computing, pp. 99–108 (2011)

[GWC19] Gabizon, A., Williamson, Z.J., Ciobotaru, O.: PLONK: permutations over Lagrange-bases for oecumenical noninteractive arguments of knowledge. Cryptology ePrint Archive, Paper 2019/953 (2019)

[HK22] Hülsing, A., Kudinov, M.A.: Recovering the tight security proof of SPHINCS$^+$. Cryptology ePrint Archive, Paper 2022/346 (2022)

[IN83] Itakura, K., Nakamura, K.: A public-key cryptosystem suitable for digital multisignatures. NEC Res. Dev. **71**, 1–8 (1983)

[IR89] Impagliazzo, R., Rudich, S.: Limits on the provable consequences of one-way permutations. In: Proceedings of the 21st Annual ACM Symposium on Theory of Computing, pp. 44–61 (1989)

[KHR+22] Kudinov, M.A., Hülsing, A., Ronen, E., Yogev, E.: SPHINCS+C: compressing SPHINCS+ with (almost) no cost. Cryptology ePrint Archive, Paper 2022/778 (2022)

[KL21] Katz, J., Lindell, Y.: Introduction to Modern Cryptography, 3rd edn. CRC Press, Boca Raton (2021)

[KMP16] Kiltz, E., Masny, D., Pan, J.: Optimal security proofs for signatures from identification schemes. In: Robshaw, M., Katz, J. (eds.) CRYPTO 2016. LNCS, vol. 9815, pp. 33–61. Springer, Heidelberg (2016). https://doi.org/10.1007/978-3-662-53008-5_2

[Lam79] Lamport, L.: Constructing digital signatures from a one way function. Technical report SRI-CSL-98, SRI International Computer Science Laboratory (1979)

[LHL94] Li, C.-M., Hwang, T., Lee, N.-Y.: Threshold-multisignature schemes where suspected forgery implies traceability of adversarial shareholders. In: De Santis, A. (ed.) EUROCRYPT 1994. LNCS, vol. 950, pp. 194–204. Springer, Heidelberg (1995). https://doi.org/10.1007/BFb0053435

[Lin06] Lindell, Y.: A simpler construction of CCA2-secure public-key encryption under general assumptions. J. Cryptol. **19**(3), 359–377 (2006)

[LK23] Lee, K., Kim, H.: Two-round multi-signatures from Okamoto signatures. Mathematics **11**(14) (2023)

[LOS+06] Lu, S., Ostrovsky, R., Sahai, A., Shacham, H., Waters, B.: Sequential aggregate signatures and multisignatures without random oracles. In: Vaudenay, S. (ed.) EUROCRYPT 2006. LNCS, vol. 4004, pp. 465–485. Springer, Heidelberg (2006). https://doi.org/10.1007/11761679_28

[LYG19] Le, D., Yang, G., Ghorbani, A.A.: A new multisignature scheme with public key aggregation for blockchain. In: Proceedings of the 17th International Conference on Privacy, Security and Trust, pp. 1–7 (2019)

[Mic00] Micali, S.: Computationally sound proofs. SIAM J. Comput. **30**(4), 1253–1298 (2000)

[MOR01] Micali, S., Ohta, K., Reyzin, L.: Accountable-subgroup multisignatures: extended abstract. In: Proceedings of the 8th ACM Conference on Computer and Communications Security (CCS), pp. 245–254 (2001)

[MPS+19] Maxwell, G., Poelstra, A., Seurin, Y., Wuille, P.: Simple Schnorr multi-signatures with applications to Bitcoin. Des. Codes Crypt. **87**(9), 2139–2164 (2019). https://doi.org/10.1007/s10623-019-00608-x

[MTT19] Mizuide, T., Takayasu, A., Takagi, T.: Tight reductions for Diffie-Hellman variants in the algebraic group model. In: Matsui, M. (ed.) CT-RSA 2019. LNCS, vol. 11405, pp. 169–188. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-12612-4_9

[Nak09] Nakamoto, S.: Bitcoin: a peer-to-peer electronic cash system (2009). http://www.bitcoin.org/bitcoin.pdf

[Nao91] Naor, M.: Bit commitment using pseudorandomness. J. Cryptol. **4**(2), 151–158 (1991). https://doi.org/10.1007/BF00196774

[NRS+20] Nick, J., Ruffing, T., Seurin, Y., Wuille, P.: MuSig-DN: Schnorr multi-signatures with verifiably deterministic nonces. In: ACM SIGSAC Conference on Computer and Communications Security (CCS), pp. 1717–1731 (2020)

[NRS21] Nick, J., Ruffing, T., Seurin, Y.: MuSig2: simple two-round Schnorr multi-signatures. In: Malkin, T., Peikert, C. (eds.) CRYPTO 2021. LNCS, vol. 12825, pp. 189–221. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-84242-0_8

[NY89] Naor, M., Yung, M.: Universal one-way hash functions and their cryptographic applications. In: Proceedings of the 21st Annual ACM Symposium on Theory of Computing, pp. 33–43 (1989)

[OO91] Ohta, K., Okamoto, T.: A digital multisignature scheme based on the fiat-Shamir scheme. In: Imai, H., Rivest, R.L., Matsumoto, T. (eds.) ASIACRYPT 1991. LNCS, vol. 739, pp. 139–148. Springer, Heidelberg (1993). https://doi.org/10.1007/3-540-57332-1_11

[PS00] Pointcheval, D., Stern, J.: Security arguments for digital signatures and blind signatures. J. Cryptol. **13**, 361–396 (2000)

[PW23] Pan, J., Wagner, B.: Chopsticks: fork-free two-round multi-signatures from non-interactive assumptions. In: Hazay, C., Stam, M. (eds.) EUROCRYPT 2023. LNCS, vol. 14008, pp. 597–627. Springer, Cham (2023). https://doi.org/10.1007/978-3-031-30589-4_21

[RS20] Rotem, L., Segev, G.: Algebraic distinguishers: from discrete logarithms to decisional Uber assumptions. In: Pass, R., Pietrzak, K. (eds.) TCC 2020. LNCS, vol. 12552, pp. 366–389. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-64381-2_13

[RY07] Ristenpart, T., Yilek, S.: The power of proofs-of-possession: securing multiparty signatures against rogue-key attacks. In: Naor, M. (ed.) EUROCRYPT 2007. LNCS, vol. 4515, pp. 228–245. Springer, Heidelberg (2007). https://doi.org/10.1007/978-3-540-72540-4_13

[Sah99] Sahai, A.: Non-malleable non-interactive zero knowledge and adaptive chosen-ciphertext security. In: Proceedings of the 40th Annual IEEE Symposium on Foundations of Computer Science, pp. 543–553 (1999)

[Sch91] Schnorr, C.P.: Efficient signature generation by smart cards. J. Cryptol. **4**(3), 161–174 (1991). https://doi.org/10.1007/BF00196725

[SCO+01] De Santis, A., Di Crescenzo, G., Ostrovsky, R., Persiano, G., Sahai, A.: Robust non-interactive zero knowledge. In: Kilian, J. (ed.) CRYPTO 2001. LNCS, vol. 2139, pp. 566–598. Springer, Heidelberg (2001). https://doi.org/10.1007/3-540-44647-8_33

[Sho97] Shoup, V.: Lower bounds for discrete logarithms and related problems. In: Fumy, W. (ed.) EUROCRYPT 1997. LNCS, vol. 1233, pp. 256–266. Springer, Heidelberg (1997). https://doi.org/10.1007/3-540-69053-0_18

[Sim98] Simon, D.R.: Finding collisions on a one-way street: can secure hash functions be based on general assumptions? In: Nyberg, K. (ed.) EUROCRYPT 1998. LNCS, vol. 1403, pp. 334–345. Springer, Heidelberg (1998). https://doi.org/10.1007/BFb0054137

[Sta23] StarkWare. Account abstraction: improving security and user experience for mainstream crypto adoption (2023). https://medium.com/starkware/account-abstraction-improving-security-and-user-experience-for-mainstream-crypto-adoption-eb57cb09023

[TSS+23] Takemure, K., Sakai, Y., Santoso, B., Hanaoka, G., Ohta, K.: More efficient two-round multi-signature scheme with provably secure parameters. Cryptology ePrint Archive, Paper 2023/155 (2023)

[TZ23] Tessaro, S., Zhu, C.: Threshold and multi-signature schemes from linear hash functions. In: Hazay, C., Stam, M. (eds.) EUROCRYPT 2023. LNCS, vol. 14008, pp. 628–658. Springer, Cham (2023). https://doi.org/10.1007/978-3-031-30589-4_22

[XZS22] Xie, T., Zhang, Y., Song, D.: Orion: zero knowledge proof with linear prover time. In: Dodis, Y., Shrimpton, T. (eds.) CRYPTO 2022. LNCS, vol. 13510, pp. 299–328. Springer, Cham (2022). https://doi.org/10.1007/978-3-031-15985-5_11

# Composability in Watermarking Schemes

Jiahui Liu[1](✉) and Mark Zhandry[2]

[1] Massachusetts Institute of Technology, Cambridge, USA
jiahuiliu@csail.mit.edu
[2] NTT Research, Sunnyvale, USA
mark.zhandry@ntt-research.com

**Abstract.** Software watermarking allows for embedding a mark into a piece of code, such that any attempt to remove the mark will render the code useless. Provably secure watermarking schemes currently seems limited to programs computing various cryptographic operations, such as evaluating pseudorandom functions (PRFs), signing messages, or decrypting ciphertexts (the latter often going by the name "traitor tracing"). Moreover, each of these watermarking schemes has an ad-hoc construction of its own.

We observe, however, that many cryptographic objects are used as building blocks in larger protocols. We ask: just as we can compose building blocks to obtain larger protocols, can we compose watermarking schemes for the building blocks to obtain watermarking schemes for the larger protocols? We give an affirmative answer to this question, by precisely formulating a set of requirements that allow for composing watermarking schemes. We use our formulation to derive a number of applications.

## 1 Introduction

Watermarking is an old idea, which aims to embed a mark in some object, such that any attempt to remove the mark destroys the object. In software watermarking, this means embedding a mark into program code, such that any attempt to remove the code will make the code useless. Such watermarking aims to deter piracy by identifying the source of pirated software. Recently, software watermarking has become an active area of research within cryptography, with numerous positive results for watermarking cryptographic functionalities, such as trapdoor functions [Nis13], pseudorandom functions [CHN+16,KW17,GKWW21], decryption [CFNP00] (under the name "traitor tracing"), and more [GKM+19].

In this work, we initiate the study of *composing* watermarked functionalities. That is, if a cryptographic primitive $A$ (or perhaps several primitives) is used to build a primitive $B$, can we use a watermarking scheme for $A$ to realize a watermarking scheme for $B$? Our aim is to show when such a composition is possible, based on properties of the construction and security proof for $B$ using $A$.

E. Boyle and M. Mahmoody (Eds.): TCC 2024, LNCS 15366, pp. 400–430, 2025.
https://doi.org/10.1007/978-3-031-78020-2_14

## 1.1 Motivation

Abstractions are central to cryptography, as they allow for decomposing various tasks into smaller building blocks, which can then be instantiated independently. The literature is full of results that show how to generically realize one abstraction assuming solutions to one or more input abstractions.

Given that cryptographic primitives are often composed, and that most watermarking schemes with provable security are for cryptographic primitives, an interesting question is whether watermarking schemes can be composed. To the best of our knowledge, this question has not been previously asked. In contrast, existing watermarking schemes are each developed "from scratch." Even if the underlying techniques are similar, watermarking schemes for different primitives must go through separate constructions and security proofs. This can be a time-consuming process.

*What Does it Mean to Watermark a PRF?* As a running example throughout this introduction, we will use pseudorandom functions (PRFs), which are one of the main workhorses in symmetric key cryptography, and are used as a central component in many higher-level protocols. In [CHN+16], the authors show how to watermark the evaluation procedure for a certain class of PRFs. We point out, however, that PRFs are typically not considered a cryptographic end goal, but rather a tool used to build other cryptographic notions. So *what, then, is the utility of watermarking a PRF?*

Another fundamental question is the following: for important reasons that we will not get into here, the watermarking guarantee proved by [CHN+16] (and all subsequent work on watermarking PRFs) is weaker than one may expect. Namely, they show that it is impossible to remove the mark without causing the program to fail on *random* inputs. Certain PRFs called "weak PRFs" are only guaranteed secure when the adversary sees evaluations on random inputs, and so for this reason may authors (e.g. [GKWW21, MW22, KN22]) refer to such a scheme as watermarking "weak PRFs." Weak PRFs can be used as building blocks for many applications, though not as many as ordinary PRFs. In the context of using PRFs as a building block, what is implication of watermarking a *weak* PRF?

*Composition of Watermarking Schemes.* Our thesis is that watermarking, at least in many cases, should be defined and executed in such a way as to be composable, allowing watermarking schemes for building blocks to generically imply watermarking schemes for higher-level protocols. Watermarking a weak PRF, for example, should generically enable watermarking for many applications of weak PRFs, such as CPA-secure symmetric encryption. Naturally, a more ambitious goal is: watermarking a message-authentication code scheme or a digital signatures scheme, when composed with a watermarkable PRF scheme, should lead to a watermarkable CCA-secure SKE scheme.

*Not all Compositions Support Watermarking.* Certainly, not all composition results from cryptography can be used to compose watermarking schemes. For example, pseudorandom functions can be built from any pseudorandom generator (PRG) [GGM86]. But this does not seem to yield a viable path toward constructing a watermarking scheme for PRFs. After all, what would it even mean to watermarking a PRG, given that the evaluation algorithm for the PRG is public?

### 1.2 Overview of Our Results

*Defining Watermarking Security.* Most cryptographic primitives are defined by an interactive game between an adversary $A$ and challenger. Given a security game, we can translate the game into a watermarking definition, as follows. The attacker gets a watermarked secret key, and then tries to produce a program $A$. We say that $A$ is "good" if it can win the security game with non-negligible advantage. We then require the existence of a tracing algorithm, which can extract the mark from any good program $A$ produced by the adversary. In the case of watermarking public key decryption functionalities, our notion corresponds exactly to existing notions of traitor tracing, since the security game is non-interactive. However, for other primitives, our definition is potentially stronger than existing notions: existing notions only ask for mark extraction for non-interactive programs $A$ that can evaluate some function, whereas we must extract from any $A$ which wins a security experiment, which is potentially interactive. The strengthened definitions will be crucial for our composition theorem, which we now describe.

*Composition Theorem.* Our first result is a composition theorem, which gives conditions under which a construction of a target primitive $P$ from input primitives $P_1, \cdots, P_k$ can be turned into a compiler for watermarking schemes.

**Theorem 1 (Main Theorem (Informal)).** *If the construction of a target primitive $P$ from input primitives $P_1, \cdots, P_k$ satisfies some given conditions and the watermarking schemes for $P_1, \cdots, P_k$ satisfy the above watermarking security definition, then we can compose the construction to watermarking schemes for $P_1, \cdots, P_k$, black-boxly into a watermarking scheme for $P$ that satisfies the above watermarking definition.*

Our conditions apply to a large class of constructions. They are, very roughly, as follows:

- The construction of $P$ from $P_1, \cdots, P_k$ is black-box. Moreover, the secret key $\mathsf{sk}$ for $P$ is $\mathsf{sk} = (\mathsf{sk}_1, \cdots, \mathsf{sk}_k)$ where $\mathsf{sk}_i$ is the secret key for $P_i$.
- The security proof for $P$ turns an adversary $A$ for $P$ into adversaries $A_1, \cdots, A_k$ for $P_1, \cdots, P_k$, respectively, such that if $A$ has non-negligible advantage, so does *at least* one of the $P_i$. Typical proofs in cryptography utilizing hybrid proofs will usually have this form.
- Moreover, we require a property of the security proof, which we call a "watermarking-compatible reduction". This is a rather technical definition, but roughly we allow the security games for $P, P_1, \cdots, P_k$ to consist of two

phases, where the adversary's winning condition is only dependent on the second phase. We require that the reduction respects the phases, in the sense that the second stages of $A_1, \cdots, A_k$ only depend on the second stage of $A$.
- Depending on the construction and reduction, *not all* input primitives $P_1, \cdots, P_k$ have to be watermarkable to give a watermarking construction for the target primitive $P$. That is, we can compose the "plain" constructions of these input primitives with the watermarkable versions of the other input primitives to give a watermarkable construction for $P$.
- If all input primitives (that need to be watermarkable) have collusion-resistant watermarking security, the resulting watermarkable $P$ also satisfies collusion-resistant security. If all input primitives can be traced using only a public key, the same is true of $P$.

Many reductions in the literature are watermarking-compatible. For example, consider constructing CPA-secure symmetric encryption from weak PRFs. Here, the first stage for CPA-security consists of all queries occurring prior to the challenge query, and the second stage consists of the challenge query and all subsequent queries. The winning condition does not depend on any first stage CPA queries (or second stage). The first and second stages for the weak PRF are just two rounds of queries to the weak PRF oracle, and here again the win condition does not depend on the actually queries. Thus, this reduction is watermarking compatible.

A non-example would be many proofs involving signatures as the target primitive $P$. The issue is that, the winning condition for a signature scheme security game is that the adversary produces a "new" signature on a message that was not seen in a previous query. But checking this win condition requires knowing all the queries. So if there is a first phase where the attacker can make signature queries, then the win condition is not solely dependent on the second stage. If we let the winning condition depend only on the second stage, the adversary can trivially win by querying a signature in the first stage and give it to the second stage adversary as the final output.

*Applications.* We demonstrate that many well-known cryptographic constructions have watermarking-compatible reductions, and therefore we can compose the watermarkable constructions of the input primitives to obtain watermarkable constructions of the target primitive. Within the scope of this work, we give the following examples:

- Two most simple examples are:
  - Watermarkable CPA-secure secret-key encryption scheme from watermarkable weak PRF
  - Watermarkable CCA2-secure secret-key encryption scheme from watermarkable weak PRF and watermarkable MACs.
- Some more advanced examples are:
  - Watermarkable CCA2-secure public-key encryption from watermarkable selectively secure identity based encryption and strong one-time signatures, which can in turn be based on LWE. Here, the strong one-time

signature we need is a "plain scheme" which does not need to be watermarkable.
- Watermarkable CCA2-secure PKE from watermarkable CPA-secure PKE and NIZK (without watermarking), which can in turn be based on LWE too.
- Watermarkable weak pseudorandom permutation from watermarkable weak PRF.
- Watermarkable CCA2-secure hybrid encryption scheme from watermarkable CCA2-secure PKE and CCA2-secure SKE (without watermarking).

Additionally, we show that all the input primitives (weak PRF, CPA-secure PKE, selective IBE, signatures, etc.) in the above examples have constructions that satisfy our security definition for watermarking. We can obtain these constructions by using or modifying the existing watermarking schemes in [GKM+19,GKWW21,MW22]. Therefore, the above composed schemes all have concrete constructions.

We also briefly discuss how the functional encryption construction in [GKP+13] is also watermarking compatible. We cannot possibly elaborate all the concrete examples of watermarking compatible reductions within the scope of this work, but given our generic framework of composition, one can easily verify whether a construction is watermarking compatible by looking into its security proof.

## 1.3 Other Related Work

[Nis20] presents a general framework for constructing watermarking schemes for any primitive which admits a certain "all-but-one" reduction. The work and ours focus on different aspects of watermarking: theirs is focused on constructing watermarking schemes in the first place, whereas our composition theorem shows how to combine watermarking schemes. We also note that the framework in [Nis20] seems very much tied to the collusion-free setting, whereas ours is much more general, and can accommodate collusions if the input tracing mechanisms are collusion resistant.

## 1.4 Technical Overview

*Watermarking.* We first briefly recall the definition of watermarking a cryptographic primitive: in a watermakrable cryptographic scheme, apart from the usual evaluation algorithms (such as key generation, encrypt, decrypt or sign and verify), it additionally has a Mark algorithm and an Extract algorithm (as well as the corresponding marking and extraction keys). The Mark algorithm allows one to embed a mark into the secret key used to evaluate the cryptographic functionality; the Extract key allows one to extract a mark from an allegedly marked key. The watermarking security, usually referred to as "unremovability", states that given a marked secret key with an adversarially requested mark $\tau$, the adversary should not be able to produce a circuit that has the same functionality as the secret key such that the above mark $\tau$ cannot be extracted from this adversarial circuit.

As explored in this work and some previous works ([GKM+19, GKWW21]), one important definitional aspect in the above security lies in what we mean by the "adversary's output circuit has the same functionality as the original secret key". We will elaborate in the following sections of the overview.

*Watermarking-Compatible Reduction.* Before we go into how we compose watermarking schemes, we expand more on the type of reductions that allow us to build a watermarking composition upon. We call these reductions watermarking-compatible. We will then give a concrete example to help comprehension.

Consider a black-box construction of a target primitive $P$ from input primitives $P_1, \cdots, P_k$ and consider a reduction algorithm $\mathcal{B}$ from security of $P$ to the security of $P_i$: we let both the adversary $\mathcal{A}$ and the reduction $\mathcal{B}$ be divided into two stages. In the first stage, stage-1 adversary $\mathcal{A}_1$, for the security game of $P$, receives some public parameters from stage-1 $\mathcal{B}_1$ and makes some queries; $\mathcal{B}_1$ answers these queries by making queries to the oracle provided by the challenger of the security game for $P_i$.

Entering the second stage, as one can expect, $\mathcal{A}_1$ can give an arbitrary state it to stage-2 adversary $\mathcal{A}_2$. However, what $\mathcal{B}_1$ can give to the second stage reduction $\mathcal{B}_2$ is more restricted: $\mathcal{B}_1$ can give all the public parameters to the second stage but *none of the queries* made by $\mathcal{A}_1$, to $\mathcal{B}_2$. $\mathcal{B}_2$ continues to simulate the query stage to answer $\mathcal{A}$'s queries. In particular, $\mathcal{B}$ records $\mathcal{A}_2$'s queries. Note that the challenge phase of the security game where $\mathcal{A}_2$ (and resp. $\mathcal{B}$) receives its challenge from the challenger always happens in stage 2[1].

In the final output phase, $\mathcal{B}_2$'s answer to the challenger in game $P_i$ will be dependent on (some of) the following pieces of information: challenger's challenge input to $\mathcal{B}_2$, $\mathcal{A}_2$'s queries made during stage 2, $\mathcal{B}_2$'s randomness used to prepare the challenge input for $\mathcal{A}_2$ and $\mathcal{A}_2$'s final output.

We give some intuition on why we need $B$ to be "oblivious" about $\mathcal{A}$'s queries in stage 1 of the above reduction. Looking forward, in the actual watermarking (unremovability) security game, we can replace "answering $\mathcal{A}$'s queries in stage 1" with a giving out a watermarked secret key to $\mathcal{A}$, where $\mathcal{B}$ will have no idea what inputs $\mathcal{A}$ has evaluated on using the key. We therefore model the reduction as the above to capture this scenario.

*Example: CCA2-Secure Secret-Key Encryption.* To make the above abstract description concrete, we take an example of the reduction from CCA2-security to weak PRF and MAC.

We briefly recall the textbook construction: the scheme's secret key consists of the PRF's secret key $\mathsf{sk}_1$ and MAC's secret key $\mathsf{sk}_2$. The encryption algorithm, on input message $m$, computes ciphertext $\mathsf{ct} = (r, \mathsf{ct}' = \mathsf{PRF.Eval}(\mathsf{sk}_1, r) \oplus m, \mathsf{sig})$,

1 But as we will see in some concrete examples, $\mathcal{A}$ can commit to some "challenge messages" in stage 1, which will be taken into stage-2 as an "auxiliary input" and later used by the challenger to prepare the challenge.

where $\mathsf{sig} \leftarrow \mathsf{MAC.Sign}(\mathsf{sk}_2, (r, \mathsf{ct}'))$. The decryption algorithm will first verify the signature $\mathsf{sig}$ on $(r, \mathsf{ct}')$, if yes continue to decrypt using $\mathsf{sk}_1$, else abort.

It's not hard to see how the CCA2 security game can fit into the format of a two-stage game we have depicted above: stage 1 consists simply of letting the adversary making queries to the encryption and decryption oracles. Entering stage 2, the adversary $\mathcal{A}$ is allowed to make more queries and then submits the challenge plaintexts; then $\mathcal{A}$ receives challenge ciphertexts from the challenger. $\mathcal{A}$ continues to make more admissible queries and finally outputs its answer.

By viewing the security game in two stages, we will recall the security proof for CCA2 security. The proof can go through a case-by-case analysis:

1. In case 1, in stage 2 of the CCA2 security game, there is a decryption query from $\mathcal{A}$ of the form $\mathsf{ct} = (\mathsf{ct}_0, \sigma)$, such that it has never been the output of an encryption query. Also, the decryption oracle did not output $\perp$ on this query.
2. In case 2, there is no such decryption query in stage 2.

In the first case, we can do a reduction to unforgeability of $\mathsf{MAC}$:

- $\mathcal{A}_{\mathsf{MAC}}$ samples its own $\mathsf{PRF}$ secret key $\mathsf{sk}_1$. For stage 1: For any encryption query from stage-1 adversary, $\mathcal{A}^1$, stage-1 reduction $\mathcal{A}^1_{\mathsf{MAC}}$ will simulate the response as follows: it will compute the $\mathsf{ct}_0$ part of ciphertext and then query the signing oracle $\mathsf{MAC.Sign}(\mathsf{sk}_2, \cdot)$ in the security game $G_{\mathsf{MAC}}$. For any decryption query, $\mathcal{A}^1_{\mathsf{MAC}}$ will query the verification oracle $\mathsf{MAC.Verify}(\mathsf{sk}_2, \cdot)$ first and decrypt those whose response is not $\perp$.
- After entering stage 2, for any encryption or decryption query from $\mathcal{A}^2$, $\mathcal{A}^2_{\mathsf{MAC}}$ will still simulate the response as the above. Recall that $\mathcal{A}^2_{\mathsf{MAC}}$ will not get to see any queries made in stage 1 and $\mathcal{A}^2_{\mathsf{MAC}}$ will record all queries from $\mathcal{A}_2$.
- In the challenge phase, $\mathcal{A}^2$ sends in challenge messages $(m_0, m_1)$; $\mathcal{A}^2_{\mathsf{MAC}}$ flips a coin $b \leftarrow \{0,1\}$, prepares and sends the signed encryption $\mathsf{ct}^*$ of $m_b$ to $\mathcal{A}^2$. $\mathcal{A}^2_{\mathsf{MAC}}$ continues to simulate the encryption and decryption oracles for $\mathcal{A}^2$, on queries $\mathsf{ct} \neq \mathsf{ct}^*$.
- In the end, $\mathcal{A}^2_{\mathsf{MAC}}$ will look up $\mathcal{A}^2$'s queries: find a decryption query $\mathsf{ct} = (\mathsf{ct}', \sigma)$ such that it has never been the output of an encryption query and the decryption oracle did not output $\perp$ on this query. $\mathcal{A}^2_{\mathsf{MAC}}$ output $(\mathsf{ct}', \sigma)$ as its forgery.

In the second case, we consider a reduction $\mathcal{A}_{\mathsf{PRF}}$ to the weak pseudorandomness of the PRF. In the following reduction, for he sake of simplicity, let's consider a variant of the weak PRF game: the adversary $\mathcal{A}_{\mathsf{PRF}}$ is given adaptive query access to an oracle $F : \{0,1\}^n \to \{0,1\}^m$ that computes a PRF function; there will be a challenge phase where a challenge input $r^*$ is sampled at random; $\mathcal{A}_{\mathsf{PRF}}$ will receive one of $(r^*, F(r^*))$ or $(r^*, y \leftarrow \{0,1\}^m)$ at random and try to tell which one it receives[2].

- In stage 1: $\mathcal{A}_{\mathsf{PRF}}$ samples its own $\mathsf{MAC}$ key $\mathsf{sk}_2$. For any encryption query from $\mathcal{A}^1$, $\mathcal{A}^1_{\mathsf{PRF}}$ will simulate the response as follows: it will query the PRF

[2] Watermarkable weak PRF with this type of security can be constructed in [GKM+19].

oracle $F(\cdot) := \mathsf{PRF.Eval}(\mathsf{sk}_1, \cdot)$ on a fresh $r$ of its own choice. After obtaining $(r, F(r))$, $\mathcal{A}_{\mathsf{PRF}}$ signs the message $(r, F(r) \oplus m)$ using $\mathsf{sk}_2$ and sends the entire ciphertext to $\mathcal{A}$.
Similarly, for decryption queries, $\mathcal{A}_{\mathsf{PRF}}$ first verifies the signature and then queries $F(\cdot)$ oracle to decrypt.

– After entering stage 2, for any encryption query from $\mathcal{A}^2$, $\mathcal{A}^2_{\mathsf{PRF}}$ will still simulate the response as the above. Recall that $\mathcal{A}^2_{\mathsf{PRF}}$ will not get to see any queries made in stage 1.
– In the challenge phase, $\mathcal{A}^2_{\mathsf{PRF}}$ will receive a challenge input-output pair $(r^*, y^*)$ from the weak PRF challenger: $r^* \leftarrow \{0,1\}^\ell, y^* \in \{0,1\}^\ell$ where $y^*$ is either $\mathsf{PRF.Eval}(\mathsf{sk}_1, r^*)$ or uniformly random.
– $\mathcal{A}^2$ sends in challenge messages, $(m_0, m_1)$; $\mathcal{A}^2_{\mathsf{PRF}}$ flips a coin $b \leftarrow \{0,1\}$ and sends $\mathsf{ct} = (r^*, y^* \oplus m_b, \sigma^*)$ to $\mathcal{A}^2$.
– If $\mathcal{A}^2$ guesses the correct $b$, then $\mathcal{A}^2_{\mathsf{PRF}}$ output 0, for "$y^*$ is $\mathsf{PRF.Eval}(\mathsf{sk}_1, r^*)$", else $\mathcal{A}^2_{\mathsf{PRF}}$ output 1, for "$y^*$ is uniformly random".

Note that in the above reductions, both $\mathcal{A}_{\mathsf{PRF}}$ and $\mathcal{A}_{\mathsf{MAC}}$ do not need the stage-1 queries from $\mathcal{A}$ to help them win their own games.

*Definition and Composition Framework of Watermarking.* Now we relate the above reduction to the watermarking goal: in our watermarking (i.e. unremovability) security game for primitive $P$, the watermarking adversary $\mathcal{A}$ acts like a "stage-1" adversary in the original security game for the underlying primitive P. Then $\mathcal{A}$ produces a program $C$ —-this signifies the end of "stage 1" of the security game.

A notable feature of our extraction algorithm is the following. The adversarial program $C$ produced by $\mathcal{A}$ will be treated as a stage-2 adversary by the extraction algorithm: the extraction algorithm will try to extract a watermark by having only black-box access to $C$. Since $C$ is supposed to function like a stage-2 adversary that wins the security game of $P$, the extraction procedure ensures $C$ operates properly by simulating the stage-2 security game for $C$.

Finally, $\mathcal{A}$ wins if $C$ is a "good" program in winning the corresponding security game for $P$, but no valid watermarks can be extracted from $C$.

Suppose a target primitive $P$ can be built from input primitives $P_1, \cdots, P_k$ via watermarking-compatible reductions, and each watermarking scheme for $P_i$ satisfies our definition, then we can compose the watermarkable versions of $P_1, \cdots, P_k$ to give a watermarkable $P$.

The composition construction works roughly as follows:

1. A watermarkable $P$'s key generation algorithm is similar to the unwatermarkable (plain) construction of $P$: by generating all keys of $P_1, \cdots, P_k$ and concatenate them, for secret/public keys respectively. The extra step is that now we also concatenate the marking keys and extraction keys generated from the watermarkble $P_1, \cdots, P_k$.
2. How $P$ evaluates is exactly the same as in the plain construction of $P$ form $P_1, \cdots, P_k$: by running the $P_1, \cdots, P_k$ algorithms as black-box subroutines.

3. To watermark $P$, we simply watermark the secret keys of the $P_1, \cdots, P_k$ respectively. The watermarked key for $P$ is then just the concatenation of the watermarked keys for $P_1, \cdots, P_k$. Since the construction is black-box and assumes the key is just a tuple of keys for $P_1, \cdots, P_k$, the evaluation with watermarked keys is still the same, running the evaluation algorithm with black-box from subroutines $P_1, \cdots, P_k$, except now with the marked keys respectively. We obtain correctness (functionality-preserving property), following from the correctness of the plain construction and functionality preserving properties of watermarkable $P_1, \cdots, P_k$.
4. In order to extract a mark, we turn any pirated algorithm $A$ for $P$ into pirated algorithm $A_1, \cdots, A_k$ for $P_1, \cdots, P_k$. We do this by applying the security reduction for $P$ to the pirated algorithm $A$, and let $A_1, \cdots, A_k$ be the adversaries produced by the reduction. We then interpret $A_1, \cdots, A_k$ as pirated programs for $P_1, \cdots, P_k$, and attempt to extract the mark from each of them. Note that any mark extraction algorithm is supposed to use only the input-output behavior of the pirate program.
   The guarantee of the reduction is that one of these pirated programs $A_i$ must actually be a "good" program in the security game for the corresponding primitive $P_i$, which means that mark extraction must succeed for that $A_i$. The security proof therefore shows that we are guaranteed to extract some mark[3].

Formalizing this composition takes some care, as we need to ensure that the mark extraction algorithm has the ability to actually transform $A$ into each of the $A_i$. There are a couple issues with getting this to work:

– In a reduction, it is typically assumed that, when the reduction is attacking $P_i$, it has access to the keys for $P_j, j \neq i$. This access is often used to simulate the view of $A$ when constructing $A_i$. In our watermarking construction, we therefore need to ensure that the tracing algorithm has this information.
– Security proofs often work via hybrid arguments that change various terms. Importantly, the hybrid arguments get to simulate the *entire* game. For us, however, we only get to simulate the second stage of the game; the first stage has actually already been fixed by the time the adversary produces its pirated program $A$. So we have to ensure that the tracing algorithm can carry out the reduction to create a program $A_i$, even though it cannot simulate the entire security experiment from the very beginning.

Our definition and proof take care of these issues and more, to give a composition theorem that applies any time our criteria are met.

Note that the restriction to games where the winning condition is independent of the first stage seems necessary. This is because the adversary actually

[3] The Extract algorithm assumes the (input) adversarial circuit to be possibly stateful and interactive, but does not require any input circuit to be stateful and interactive. For example, an honestly generated watermarked circuit is not stateful or interactive in our construction.

gets in its possession a watermarked key, which lets it compute the various cryptographic functions of the key. For example, in the case of digital signatures, the adversary can actually compute signatures for itself. In the case of encryption, the adversary can encrypt and decrypt messages. Moreover, there is no way to track what the adversary is doing, since it is all happening internally to the adversary rather than being explicitly queried (suppose for example, that the adversary signs a few messages, but has its entire state encrypted under a fully homomorphic encryption (FHE) scheme so that the signature and message being signed are all "under the hood" of the FHE scheme). We model this ability by having a first stage of the security experiment where the adversary can freely query the functionalities, but then do not have the win condition depend on those queries.

*Watermarking Composition Example: CCA2-Secure SKE.* Now we demonstrate how the above abstract watermarking composition works by using the concrete example of a CCA2-secure SKE built from weak PRF and MAC again.

Consider having input constructions as watermarkable weak PRF and watermarkable MAC which satisfies our syntax and unremovability security requirements, a watermarking CCA-secure scheme consists of several algorithms: $\mathsf{WMSetup}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Mark}, \mathsf{Extract}$. The $\mathsf{WMSetup}$ generates the secret key, marking key and extraction key by concatenating the corresponding keys from the PRF and MAC. The marking algorithm is on input $\mathsf{sk} = (\mathsf{sk}_{\mathsf{PRF}}, \mathsf{sk}_{\mathsf{MAC}})$ and watermarking message $\tau$, output $\tilde{\mathsf{sk}} = (\tilde{\mathsf{sk}}_{\mathsf{PRF}} \leftarrow \mathsf{PRF.Mark}(\mathsf{sk}_{\mathsf{PRF}}, \tau), \tilde{\mathsf{sk}}_{\mathsf{MAC}} \leftarrow \mathsf{PRF.Mark}(\mathsf{sk}_{\mathsf{MAC}}, \tau))$.

Finally, the $\mathsf{Extract}$ algorithm is slightly more complex: on input extraction key $\mathsf{xk} = (\mathsf{xk}_{\mathsf{MAC}}, \mathsf{xk}_{\mathsf{PRF}})$ and a circuit $C$, do *both* of the following:

1. Create a circuit $C_{\mathsf{PRF}}$:
   - Simulate the stage-2 CCA2-security for $C$ (as described in our previous paragraph on watermarking-compatible reduction from CCA2 security to PRF). Note that just as a real reduction $C_{\mathsf{PRF}}$ is only hardcoded with the MAC extraction key $\mathsf{xk}_{\mathsf{MAC}}$ (unless the underlying watermarkable PRF scheme has a public extraction key), which enables it to simulate the signing and verification of MAC; to compute the part that involves the PRF evaluation oracle, $C_{\mathsf{PRF}}$ need to make external queries to some "challenger" to get answers.

     How $C_{\mathsf{PRF}}$ uses $C$ to get its final output is exactly the same as in the watermarking-compatible reduction from CC SKE to weak PRF we described.

   After $C_{\mathsf{PRF}}$ is created, the extraction algorithm uses the watermarkable $\mathsf{PRF}$'s extraction algorithm to extract a mark from $C_{\mathsf{PRF}}$: $\tau/\bot \leftarrow \mathsf{PRF.Extract}(\mathsf{xk}_{\mathsf{PRF}}, C_{\mathsf{PRF}})$. Note that the "external queries" made by the circuit will be answered now by the $\mathsf{PRF.Extract}(\mathsf{xk}_{\mathsf{PRF}}, \cdot)$ algorithm, because it treats $C_{\mathsf{PRF}}$ as a stage-2 adversary in the weak pseudorandomness security game and uses the extraction key $\mathsf{xk}_{\mathsf{PRF}}$ to simulate the game.
2. Create a circuit $C_{\mathsf{MAC}}$:

  - Simulate the stage-2 CCA2-security for $C$ (as described in our previous paragraph on watermarking-compatible reduction from CCA2 security to MAC). Note that just as a real reduction $C_{\mathsf{MAC}}$ is only hard-coded with the PRF extraction key $\mathsf{xk}_{\mathsf{PRF}}$, which enables it to simulate the oracles of PRF; to compute the part that involves the MAC signing/verifying oracles, $C_{\mathsf{MAC}}$ need to make external queries to some "challenger" to get answers.

    How $C_{\mathsf{MAC}}$ uses $C$ to get its final output is exactly the same as in the watermarking-compatible reduction from CCA SKE to MAC we described.

After $C_{\mathsf{MAC}}$ is created, the extraction algorithm will use the watermarkable $\mathsf{MAC}$'s extraction algorithm to extract a mark from $C_{\mathsf{MAC}}$: $\tau/\bot \leftarrow \mathsf{MAC.Extract}(\mathsf{xk}_{\mathsf{MAC}}, C_{\mathsf{PRF}})$. Note that the "external queries" made by the circuit will be answered by the algorithm $\mathsf{MAC.Extract}(\mathsf{xk}_{\mathsf{MAC}}, \cdot)$ algorithm, because it treats $C_{\mathsf{MAC}}$ as a stage-2 adversary in the MAC security game and uses the extraction key $\mathsf{xk}_{\mathsf{MAC}}$ to simulate the game.

We say that the adversary $\mathcal{A}$ wins the watermarkable CCA2-secure SKE's unremovability game if no (previously queried) watermark can be extracted from either of the above circuits $C_{\mathsf{PRF}}, C_{\mathsf{MAC}}$ created during the extraction algorithm and $C$ is a "good" program in winning the CCA2 security game. The intuition for the security proof is relatively straightforward: if $C$ is a "good" program for winning CCA2-security game, then at least one of $C_{\mathsf{PRF}}$ and $C_{\mathsf{MAC}}$ must be a "good" program for winning its corresponding security game pf weak PRF or MAC, by the property of the reduction.

For instance, imagine a reduction algorithm $\mathcal{B}_{\mathsf{PRF}}$ to the unremovability of watermarkable $\mathsf{PRF}$: $\mathcal{B}_{\mathsf{PRF}}$ will simulate the marking query stage for $\mathcal{A}$ by making marking queries to the marking oracle of the weak PRF challenger, along with the $\mathsf{MAC}$ key it sampled on its own. Note that the evaluation queries to the $\mathsf{PRF}$ evaluation oracle can now be replaced by having access to a marked $\mathsf{PRF}$ key. After $\mathcal{A}$ outputs circuit $C$, $\mathcal{B}_{\mathsf{PRF}}$ simply creates the same circuit $C'_{\mathsf{PRF}}$ as the $C_{\mathsf{PRF}}$ created in the above extraction algorithm. If the $C_{\mathsf{PRF}}$ circuit created in the extraction is "good" at winning the weak pseudorandomness game, then so is $C'_{\mathsf{PRF}}$. Since we cannot extract a watermark from $C_{\mathsf{PRF}}$, neither can we extract from $C'_{\mathsf{PRF}}$ because they have the same input-output behavior when using the same $C$ as its subroutine black-boxly. The same argument applies to the reduction to unremovability of watermarkable MAC.

*More Advanced Watermarking Constructions with "Unwatermarkable" Building Blocks.* The watermarking reduction in the above example of CCA2-secure SKE is relatively straightforward to acquire. Some similar construction examples include a watermarkable weak PRP from watermarkable weak PRF (see full version for details).

However, there exist various constructions that look markedly distinct from the above example of CCA2-secue SKE and may be much more involved.

More importantly, some constructions have building blocks which are cryptographic primitives that don't make sense to watermark, for instance, a non-interactive zero knowledge proof scheme. Nevertheless, we show that a large class of them can be watermarking-compatible and in many scenarios, we simply do *not* need to watermark all the building blocks.

We briefly discuss two examples on a high level:

- **CCA-secure PKE from IBE and One-Time Signatures** The first example is the CCA2-secure PKE from selectively secure IBE and one-time signatures by [BCHK07].
  In the original scheme, the encryption samples a fresh signature signing/verification key pair $(\mathsf{sk}, \mathsf{vk})$, compute an IBE ciphertext by using the $\mathsf{vk}$ as the identity, then signs this ciphertext using signing key $\mathsf{sk}$, and finally outputs the IBE ciphertext, the signature and $\mathsf{vk}$.
  To decrypt, one first verifies the signature under $\mathsf{vk}$, if valid then derives an identity-embedded decryption key from the IBE master secret key using $\mathsf{vk}$ as the identity and decrypt the ciphertext.
  Note that in this scheme, the keys for one-time signatures are only sampled "online" upon every encryption algorithm and how to watermark such keys is not well-defined. However, as it turns out—we don't have to watermark the signing key in our composed construction. A watermarkable CCA2-secure PKE using the above construction only has to watermark the IBE secret key. The security proof would show that a successful unremovability adversary can help us either break the unremovability security of the watermark IBE scheme or break the strong one-time unforgeability security of the signature scheme. In other words, leveraging the "plain" security of the one-time signature scheme suffices for the composed watermarking scheme.
  In more detail, we can show the following: the pirated algorithm $C$ produced by an adversary is a "good" program for breaking the CCA-security and meanwhile no watermark can be extracted from $C$. However, it does not help us win the selective IBE CPA-security game. Then, simply following the hybrid argument in the security analysis of the CCA-secure PKE itself, we observe that $C$ must be breaking the security of the one-time signature scheme. Thereby, we can use $C$ black-boxly to create a reduction for the one-time signature scheme.
- **CCA-secure PKE from CPA-secure PKE and NIZK** A second example is the [NY90] CCA-secure PKE scheme built from CPA-secure PKE and a non-interactive zero-knowledge proof scheme (NIZK). The encryption algorithm encrypts a message twice under two different public keys and uses a NIZK proof to prove that they encrypt the same message.
  The watermarkable version of the above construction does not watermark the NIZK scheme, but only the two decryption keys of the PKE scheme. How the NIZK scheme is used in the watermarking scheme is less obvious to see: intuitively, the extraction algorithm will try to create two circuits that break the underlying CPA security game of the PKE, with the corresponding keys. If we look carefully into the proof for the [NY90] scheme, to make this reduction

go through, one has to first work in a hybrid game where the real NIZK proof in the challenge ciphertext is replaced with a simulated proof. Our extraction algorithm thereby uses such a simulated proof when interacting with the input circuit $C$. The security of NIZK helps us say that if $C$ is "good" in the original CCA-security game, so will $C$ be good in the game simulated by the extraction algorithm.

To summarize, we only have to watermark the secret keys of input primitives $P_i$ which have their keys generated in the main Key Generation algorithm of the target primitive $P$ and have their secret keys used during the evaluation algorithm of $P$. If an input primitive has its keys sampled "freshly" during every invocation of $P$'s evaluation algorithms or sampled in key generation, but not used in $P$'s evaluation algorithm (e.g. only used in the security proof instead), then we don't need a watermarkable version of $P_i$ to build a watermarkable $P$. Leveraging $P_i$'s original security property suffices.

More advanced examples include the functional encryption from attribute-based encryption, FHE and garbled circuits in [GKP+13], where we only have to watermark the ABE scheme; a hybrid CCA-secure encryption scheme from CCA-secure PKE and CCA secure SKE, where we only have to watermark the PKE scheme. We refer the readers to the construction and discussions of these examples in the full version.

## 2 Definitions: General Cryptographic Primitive and Watermarking-Compatible Constructions

### 2.1 General Cryptographic Primitive

In this section, we present syntax and definitions for a general cryptographic primitive. The notations and definitions formalized in this section will assist the demonstration of our generic watermarking framework.

*General Cryptographic Primitive Syntax.* A cryptographic primitive $P = (\mathsf{KeyGen}, \mathsf{SecEval}, \mathsf{PubEval})$ consists of the following algorithms:

- $\mathsf{KeyGen}(1^\lambda) \to (\mathsf{sk}, \mathsf{pk})$: is a (randomized) algorithm that takes a security parameter $\lambda$ and interacts with an adversary $\mathcal{A}$: $(\mathsf{sk}, \mathsf{pk}) \leftarrow (\mathcal{A} \Leftrightarrow \mathsf{KeyGen}(1^\lambda))$ where $\mathsf{sk}$ is some secret information unknown to $\mathcal{A}$, and $\mathcal{A}$ can get some public information $\mathsf{pk}$ from the interaction.
- $\mathsf{SecEval}(\mathsf{sk}, \mathsf{pk}, x \in \mathcal{X}) \to y \in \mathcal{Y}_s$ : a secret-evaluation algorithm that takes in the secret information $\mathsf{sk}$, public information $\mathsf{pk}$ and some input $x$ from input space $\mathcal{X}$, and output a value $y \in \mathcal{Y}_s$.
- $\mathsf{PubEval}(\mathsf{pk}, x \in \mathcal{X}) \to y \in \mathcal{Y}_p$ : a public-evaluation algorithm that takes in the public information $\mathsf{pk}$ and some input $x$ from input space $\mathcal{X}$, and output a value $y \in \mathcal{Y}_p$.

More generally, the $\mathsf{KeyGen}$ algorithm can generate several secret keys $\mathsf{sk}_1, \cdots, \mathsf{sk}_\ell$ for some polynomial $\ell$. There will be $\ell$ different secret algorithms $\{\mathsf{SecEval}_i(\mathsf{sk}_i, \mathsf{pk}, x \in \mathcal{X}_{s,i})\}_{i\in[\ell]}$ and (some constant) $m$ different public algorithms $\{\mathsf{PubEval}_j(\mathsf{pk}, x \in \mathcal{X}_{p,j})\}_{j\in[m]}$. For example, an identity-based encryption scheme can have two decryption algorithms, one using the master secret key, the other with a secret key embedded with an identity.

However, without loss of generality, we will make two simplifications:

- All algorithms have their input space padded to be the same length as $\mathcal{X}$;
- We view all secret algorithms $\{\mathsf{SecEval}_i(\mathsf{sk}_i, \mathsf{pk}, x \in \mathcal{X}_{s,i})\}_{i\in[\ell]}$ as one algorithm $\mathsf{SecEval}(\mathsf{sk}, \mathsf{pk}, \cdot)$ that will take in an index $i \in [\ell]$ to decide which mode to use, similarly for public algorithms. But we will assume such an index specification to be implicit and omit the use of indices in the algorithm to avoid an overflow of letters.

Occasionally, we denote all the algorithms $(\{\mathsf{SecEval}_i(\mathsf{sk}_i, \mathsf{pk}, x \in \mathcal{X}_{s,i})\}_{i\in[\ell]}, \{\mathsf{PubEval}_j(\mathsf{pk}, x \in \mathcal{X}_{p,j})\}_{j\in[m]})$ as a combined functionality hardcoded with the corresponding keys $\mathsf{Eval}(\mathsf{pk}, \mathsf{sk})$. We call it $\mathsf{Eval}$ for short.

*Correctness for Predicate $F_R$* Before going into the correctness property, we first define a notion important to our generic definition:

**Definition 1 (Predicate).** *A predicate $F(C, x, z_1, \cdots, z_k, r)$ is a binary outcome function that runs a program $C$ on a some input $x$ to get output $y$, and outputs 0/1 depending on whether $(x, y, z_1, \cdots, z_k, r) \in R$ for some binary relation defined by $R$. The randomness of input $x$, program $C$ both depend on randomness $r$. . $z_1, , z_k$ are auxiliary inputs that specify the relation.*

A correctness property of a primitive $P$ with respect to predicate $F_R$ says that

**Definition 2 (Correctness for Predicate $F_R$).** *$P$ satisfies correctness if there exists some function $\epsilon = \epsilon(\lambda) \in [0, 1]$ so that for all $\lambda \in \mathbb{N}, x \in \mathcal{X}$[4]:*

$$\Pr_{r \leftarrow D_r, (\mathsf{sk},\mathsf{pk}) \leftarrow \mathsf{KeyGen}(1^\lambda)} [F_R(\mathsf{Eval}, \mathsf{pk}, \mathsf{sk}, x, r) = 1] \geq 1 - \epsilon.$$

*The randomness used in checking the predicate is sampled from a distribution $D_r$. We can simply take $D_r$ to be the uniform distribution, which can be mapped to any distribution we need when computing the predicate.*

*All $\epsilon$ within the scope of this work is negligible in $\lambda$.*

*Remark 1.* To further explain the above correctness property, we consider the following (abstract) example. Given $\mathsf{Eval} = (\{\mathsf{SecEval}_i(\mathsf{sk}_i, \mathsf{pk}, x \in \mathcal{X})\}_{i\in[\ell]}, \{\mathsf{PubEval}_j(\mathsf{pk}, x \in \mathcal{X}\}_{j\in[m]})$ , $F_R$ samples input $x \in \mathcal{X}$ using the first part of $r$; then it runs algorithm (supposing randomized, using part 2 of string

[4] For a cryptographic primitive $P$, there can be many different correct properties, each defined with respect to a different predicate.

$r$) $\mathsf{SecEval}_1(\mathsf{sk}_1, \mathsf{pk}, x)$ to give some outcome $y_1$; then runs (supposing deterministic) $\mathsf{PubEval}_1(\mathsf{pk}_1, y_1)$ to give outcome $y_2$; check if $(x, y_1, y_2) \in R$; output 1 if yes, 0 otherwise.

A concrete example is a public key encryption scheme: the predicate is encrypting a message $x$ using $r$ and then decrypting it to check if one can recover the original message.

*Game-Based Security.* A security property of the cryptographic primitive $P$ is described by a interactive procedure $G_P$ between a challenger and adversary $\mathcal{A}$.

$G_P$ will involve $\mathsf{KeyGen}, \mathsf{SecEval}, \mathsf{PubEval}$ as its subroutines. $G_P$ outputs a bit 1 if the game is not aborted and a certain condition has been met at the end of the game; else it outputs 0.

The security of a primitive $P$ with respect to $G_P$ says: For all $\lambda \in \mathbb{N}$, there exists some function $\eta = \eta(\lambda)$ such that for all *admissible* $\mathcal{A}$ , there exists a function $\mathsf{negl}(\lambda)$:

$$\Pr[G_P(1^\lambda, \mathcal{A}) = 1] \leq \eta + \mathsf{negl}(\lambda)$$

where $\eta$ is the trivial probability for any admissible $\mathcal{A}$ to make $G_P$ output 1 and the probability is over the randomness used in $G_P$.

*2-Stage Game-Based Security.* To be compatible with the context of watermarking, we will view the game $G_P$ as a 2-stage security game. 2-stage game-based security is a central notion that connects a prmitive's "plain security' to its watermarking security.

The $\mathsf{KeyGen}$ procedure and a first part of the interactions between the challenger and $\mathcal{A}$ are taken out of the original game and executed as a first stage $G_P^1$. Then, $G_P^2$ denotes the rest of the game and will take in parameters $(\mathsf{sk}, \mathsf{pk})$ generated in stage 1 as well as some auxiliary input.

**Definition 3 (2-Stage Game-Based Security.).** *A security property of the cryptographic primitive $P$ is described by a stage-1 (possibly interactive) key generation procedure a challenger and adversary $\mathcal{A}_1$ followed by a fixed-parameter game $G_P^2$ between a challenger and adversary $\mathcal{A}_2$.*

*We denote $G_P^1(1^\lambda, \mathcal{A}_1)$ as the first stage adversary $\mathcal{A}_1$ interacting a challenger which runs the* $\mathsf{KeyGen}(1^\lambda)$, $\mathsf{SecEval}, \mathsf{PubEval}$ *algorithm; together they output a key pair* $(\mathsf{sk}, \mathsf{pk})$ *and some auxiliary parameters* $\mathsf{aux}$ *which will be later used in the game $G_P$. $\mathcal{A}_1$ only gets* $\mathsf{pk}, \mathsf{aux}$ *but may make arbitrary polynomial number of admissible queries to oracles provided by the challenger during the interaction.*

*The stage-2 game challenger $G_P^2$ is parametrized by inputs* $\mathsf{sk}, \mathsf{pk}, \mathsf{aux}$ *generated during stage 1 and will involve* $\mathsf{SecEval}, \mathsf{PubEval}$ *as its subroutines. Stage-2 adversary $\mathcal{A}_2$ gets an arbitrary polynomial size state* $\mathsf{st}$ *from stage 1 $\mathcal{A}_1$. $G_P^2$ outputs a bit 1 if the game is not aborted and a certain condition has been met at the end of the game; else it outputs 0.*

*The security of a primitive $P$ with respect to $G_P$ says: there exists some function $\eta = \eta(\lambda)$ such that for all* admissible $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ *and any non-negligible $\gamma = \gamma(\lambda)$, there exists a function* $\mathsf{negl}(\lambda)$ *for all $\lambda \in \mathbb{N}$:*

$$\Pr[\mathcal{A}_2(\mathsf{st}) \text{ is } \gamma\text{-good in } G_P^2(\mathsf{sk}, \mathsf{pk}, \mathsf{aux}, \cdot) : \{(\mathsf{sk}, \mathsf{pk}), \mathsf{aux}, \mathsf{st}\} \leftarrow G_P^1(\mathcal{A}_1, 1^\lambda)] \leq \mathsf{negl}(\lambda)$$

*where $\mathcal{A}_2$ is said to be $\gamma$-good if:*

$$\Pr[G_P^2(\mathsf{sk}, \mathsf{pk}, \mathsf{aux}, \mathcal{A}_2) = 1] \geq \eta + \gamma$$

*where $\eta$ is the trivial probability for any admissible $\mathcal{A}$ to make $G_P$ output 1 and the probability is over the randomness used in* $\mathsf{KeyGen}$ *and by $G_P$.*

*Remark 2.* While some readers may find the introduction of parameter $\gamma$ in the above 2-stage game confounding, we would like to make a note that the above two definitions are essentially equivalent.

An alternative way of stating the 2-stage game security would be: there exists some function $\eta = \eta(\lambda)$ such that for all *admissible* $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$, there exists two negligible functions $\mathsf{negl}_1(\lambda), \mathsf{negl}_2(\lambda)$ such that for all $\lambda \in \mathbb{N}$:

$$\begin{aligned}&\Pr[\Pr[G_P^2(\mathsf{sk}, \mathsf{pk}, \mathsf{aux}, \mathcal{A}_2(\mathsf{st})) = 1]\\ &\quad \leq \eta + \mathsf{negl}_1(\lambda) : \{(\mathsf{sk}, \mathsf{pk}), \mathsf{aux}, \mathsf{st}\} \leftarrow G_P^1(\mathcal{A}_1, 1^\lambda)] \geq 1 - \mathsf{negl}_2(\lambda)\end{aligned}$$

*Remark 3 (Division into a 2-stage Game).* How we divide a security game into two stages depends on the application and suppose that the game $G_P$ has different stages by its definition, the way we divide stage-1 and 2 is usually not the same as in the original definition of $G_P$. We will see examples later.

In most settings, stage-1 game $G_P^1$ only involves running the key generation $\mathsf{KeyGen}$ and letting $\mathcal{A}_1$ make some queries. In a few special settings, $\mathcal{A}_1$ needs to commit to some challenge messages which will be put into $\mathsf{aux}$ and be part of the input to stage 2. Examples include the challenge attribute/identity in an attribute/identity-based encryption, because the queries made in the first stage need to go through the "admissibility" check that depends on $\mathcal{A}$'s choice of challenge messages.

*Remark 4.* In the rest of this work, it is usually clear from the context which stage of $G_P$ we are refering to because $G_P^2$ will take in paramters $\mathsf{sk}, \mathsf{pk}, \mathsf{aux}$ while the entire game $G_P$ takes in only security parameter $1^\lambda$. Plus $G_P^1$ is seldomly used explicitly in our language. Occasionally we will omit the superscript and slightly abuse the notation to denote $G_P^2$ as $G_P(\mathsf{sk}, \mathsf{pk}, \mathsf{aux}, \cdot)$

For our convenience in later notations, we also give the following simple definition:

**Definition 4 (Stage-2 Game View).** *The stage-2 $G_P^2$ can also output a view* $\mathsf{view}(\mathsf{sk}, \mathsf{pk}, \mathsf{aux}, \mathcal{A}_2)$ *that is the transcript of interaction of $\mathcal{A}_2$ with the challenger in $G_P^2$, including the final output.*

*Remark 5.* We make a few notes on the scope of security games we consider in this work:

1. We mainly focus on game-based (and falsifiable) security notions within the scope of this work. We need the challenger in the security to be efficient for the watermarking construction to be efficient.
2. All the admissible adversary need to be PPT. We do not consider security models other than polynomially bounded in time/circuit size (such as bounded storage).

## 2.2 Watermarking-Compatible Construction of Cryptographic Primitive

In this section, we characterize what type of black-box cryptographic constructions are watermarking compatible. As we will see in later sections, watermarking compatible constructions allow us to give a watermarking scheme for the target primitive constructed, black-boxly from the watermarking schemes of the building blocks.

*Outline and Intuition.* Let use denote $P$ as the target, or outcome primitive built in a construction. Let $P_i, i \in [k]$ denote each underlying primitive we use to build $P$.

Intuitively, one expects to watermark all underlying building blocks to ensure watermarking security of the target primitive $P$. As discussed in our technical overview section "More advanced watermarking constructions with unwatermarked building blocks", many constructions possess building blocks of primitives that do *not* need to be watermarkable to ensure the watermarkability of the final target primitive.

Here, we discuss the matter in more details: we need the partitioning for primitives in $\{P_i\}_{i\in[k]}$ into $S$ and $\bar{S}$. These primitives in these two sets will play different roles when we construct the watermarking scheme for the target primitive $P$: the primitives in set $S$ need to be watermarkable and those in set $\bar{S}$ do not need to be watermarkable. The reasoning is that the primitives in set $S$ will have their secret keys generated during the Key Generation of the target primitive $P$, but primitives in set $\bar{S}$ will only have their secret keys generated freshly during every run of $\mathsf{SecEval}$ or $\mathsf{PubEval}$ algorithm of the target primitive $P$.

A simple example is CCA2-secure PKE scheme based on IBE and one-time signatures, which we discussed in the technical overview. During the encryption algorithm, one generates fresh one-time signature keys and the message is encrypted with the one-time signature's verification key as the identity. Therefore, the one-time signature in the above construction belongs to set $\bar{S}$ because its keys are only generated during the encryption algorithm.

In the watermarking construction for $P$, the $\mathsf{KeyGen}, \mathsf{PubEval}, \mathsf{SecEval}$ algorithms will follow from the plain construction for $P$ from primitives in set $S$ and $\bar{S}$. Therefore, we cannot watermark the keys for the primitives in set $\bar{S}$ because their keys are not generated when we give out the watermarked key (which only contains keys in set $S$) to a user. We therefore distinguish these two sets in our presentation for watermarking-compatible construction in this section and watermarkable implementation in Sect. 3.2.

Moreover, looking forward: we will observe that we indeed do *not* need watermarking security (i.e. unremovability) for the primitives in set $\bar{S}$, to achieve watermarking security for the target primitive P. We only need to rely on their "plain security" (e.g. unforgeability for a signature scheme, IND-CPA-security for an encryption scheme).

*Notations.* Now we give formalization for the above outline.

Suppose a cryptographic primitive $P$ is constructed black-boxly from primitives $P_1, P_2, \cdots, P_k$ in the following way, where $P_i, P_j, i \neq j$ are allowed to be the same primitive.

- Let $\mathcal{S}$ be some fixed subset of $[k]$ defined in the construction. $\mathcal{S}$ specifies two ways of using primitive $P_i$. The major difference is: for primitives $P_i, i \in \mathcal{S}$, the algorithm $\mathsf{KeyGen}(1^\lambda)$ will compute $P_i.\mathsf{KeyGen}(1^\lambda)$ and the secret keys $P_i.\mathsf{sk}$ generated will be used in the secret evaluation algorithm $\mathsf{wSecEval}$.
  Without loss of generality, we let the first $|\mathcal{S}|$ number of $P_i$'s be those corresponding to the set $\mathcal{S}$.
- For primitives $P_i, i \notin \mathcal{S}$, the algorithm $\mathsf{KeyGen}(1^\lambda)$ will not compute $P_i.\mathsf{KeyGen}(1^\lambda)$. They will either (1) have their keys generated freshly upon every run of $\mathsf{SecEval}$ or $\mathsf{PubEval}$ (2) will have a reference string (which can be viewed as a public key) generated during $\mathsf{KeyGen}(1^\lambda)$, but have no secret keys or their secret keys will not be used in the $\mathsf{SecEval}$ algorithm of the target primitive $P$.
- For the sake of formality, we make a further division of the set $\bar{S}$ into $T_S, T_P, T_K$. They perform slightly different functionalities in the construction.
  - Let $\mathcal{T}_S \in [k]$ denote the set of indices $i$ where $P_i$'s keys will be generated during the secret evaluation algorithm $\mathsf{SecEval}$;
  - Let $\mathcal{T}_P \in [k]$ denote the set of indices $i$ where $P_i$'s keys will be generated during the public evaluaton algorithm $\mathsf{PubEval}$;
  - Let $\mathcal{T}_K$ denote the set of indices $i$ where $P_i$ have no secret keys/secret keys are not used in $\mathsf{SecEval}$ and will have a public reference string (which can be viewed as a public key) generated during $\mathsf{KeyGen}(1^\lambda)$.[5].

*Watermarking-Compatible Construction Syntax*

$\mathsf{KeyGen}(1^\lambda) \to (\mathsf{sk}, \mathsf{pk})$:

[5] As we will see, primitives in $\mathcal{T}_k$ will have their secret keys (called trapdoors $\mathsf{td}$ here) used only in the security proofs.

Looking forward: the sets $\mathcal{T}_S, \mathcal{T}_P$ will play the same role in the watermarking reduction, and the set $\mathcal{T}_K$ will incur a slightly different argument but will essentially play the same role as the other primitives in set $\bar{S}$. That is, only their plain security is required to achieve the watermarking security of the target primitive $P$.

1. compute $(\mathsf{sk}_i, \mathsf{pk}_i) \leftarrow P_i.\mathsf{KeyGen}(1^\lambda)$ for all $i \in \mathcal{S}$; compute $(\mathsf{pk}_i, \mathsf{td}_i) \leftarrow P_i.\mathsf{KeyGen}(1^\lambda)$ for all $i \in \mathcal{T}_K$.
2. output $\mathsf{sk} = (\{\mathsf{sk}_i\}_{i \in \mathcal{S}}); \mathsf{pk} = (\{\mathsf{pk}_i\}_{i \in \mathcal{S} \cup \mathcal{T}_K})$.

$\mathsf{SecEval}(\mathsf{sk}, \mathsf{pk}, x \in \mathcal{X}) \to y \in \mathcal{Y}_s$ : is an algorithm that
1. uses $P_i.\mathsf{SecEval}(\mathsf{sk}_i, \mathsf{pk}_i, \cdot), i \in \mathcal{S}$ as subroutines.
2. uses $P_i.\mathsf{PubEval}(\mathsf{pk}_i, \cdot), i \in \mathcal{S} \cup \mathcal{T}_K$ as subroutines.
3. computes $(\mathsf{sk}_j, \mathsf{pk}_j) \leftarrow P_j.\mathsf{KeyGen}(1^\lambda)$ for some $j \in \mathcal{T}_S$ (can include these $\mathsf{pk}_j$ generated as part of the output).

$\mathsf{PubEval}(\mathsf{pk}, x \in \mathcal{X}) \to y \in \mathcal{Y}_p$ : is an algorithm that
1. uses $P_i.\mathsf{PubEval}(\mathsf{pk}_i, \cdot), i \in \mathcal{S} \cup \mathcal{T}_K$ as subroutines.
2. computes $(\mathsf{sk}_j, \mathsf{pk}_j) \leftarrow P_j.\mathsf{KeyGen}(1^\lambda)$ for all $j \in \mathcal{T}_P$ (may include these $\{\mathsf{pk}_j\}_j$ generated as part of the output).

We say the construction is watermarking compatible if the above construction of $P$ satisfies:

1. **Correctness of Construction:** the above construction of $P$ satisfies a correctness property defined by predicate $F_R$. Moreover, the proof of this correctness property follows from correctness properties of $P_1, \cdots, P_k$ for predicates $F_{R_i}$ respectively.
2. **Security of Construction:** The above construction satisfies security defined with game $G_P$. The security can be proved from the security properties of $P_1, \cdots, P_k$ with respect to some game $G_{P_1}, \cdots, G_{P_k}$
3. **Watermarking Compatible Reduction:** The above security proof is of the following format, which we call *Watermarking-Compatible Reduction*, shown below.

Among the above 3 properties, the first two are natural properties that come with any black-box cryptographic constructions with provable security and correctness. We will focus on discussing property 3.

*Watermarking Compatible Reductions.* On a high-level, a watermarking-compatible reduction is essentially a natural security reduction, except with the following feature: we view both the security game $G_P$ for the target security primitive $P$ and the security game $G_{P_i}$ for the input primitive $P_i$ as two-stage games defined in Definition 3. The supposed adversary $\mathcal{A}$ for $G_P$ will just be any usual PPT adversary. But we restrict the reduction in stage 2 to be "oblivious" about the queries made by $\mathcal{A}$ in stage 1: that is, it cannot pass on any queries made by $\mathcal{A}$ to the stage 2 reduction but we should nevertheless make the reduction go through successfully.

We divide our watermarking compatible reductions into two types. Type 1 reduction captures the reduction from breaking the security of $P$ to breaking the security of a primitive $P_i, i \in \mathcal{S}$, i.e. the primitives that have their secret keys generated in the $\mathsf{KeyGen}$ algorithm and used in the $\mathsf{SecEval}$ algorithm of $P$.

Type 2 reduction accordingly captures the reduction from breaking the security of $P$ to breaking the security of a primitive $P_i, i \in \bar{\mathcal{S}}$.

The main difference is in how they are used in proving the unremovability security in the composed watermarking scheme. In the actual watermarking unremovability, type 1 reduction is supposed to be oblivious about the queries made by the adversary and will eventually lead to breaking the unremovability of the underlying primitive; type 2 reduction operates similarly but will eventually lead to breaking the plain security of the underlying primitive.

*Watermarking-Compatible Reduction Type 1.* First we consider reductions for primitives $P_i$ for $i \in \mathcal{S}$.

1. Consider a reduction to $P_i$ for the $j$-th $P_i \in \mathcal{S}$; let $\mathcal{A}_i = (\mathcal{A}_i^1, \mathcal{A}_i^2)$ be the two-stage adversary for the 2-stage game of primitive $P_i$ defined in Definition 3.
2. $\mathcal{A}_i$ receives public key $\mathsf{pk}_i$
3. $\mathcal{A}_i$ prepares $(\mathsf{sk}_j, \mathsf{pk}_j) \leftarrow \mathsf{KeyGen}_j(1^\lambda)$ for all $j \neq i, j \in \mathcal{S}$ and $\mathcal{A}_i$ sends all $(\{\mathsf{pk}_j\}_{j \in \mathcal{S}})$ to $\mathcal{A}_1$.
4. $\mathcal{A}_i$ simulates the security game $G_P$ for $P$ with $\mathcal{A}$, while being the adversary in game $G_{P_i}$.
   – In stage 1, $\mathcal{A}_i^1$ provides the oracles needed for game $G_P^1$ and answer $\mathcal{A}^1$'s queries as follows:
     • For any (admissible) queries in the game $G_P^1$, if $\mathsf{sk}_i$ is required to compute the answer for the query, $\mathcal{A}_i$ can use the oracles provided in game $G_{P_i}^1$. To answer the entire query, $\mathcal{A}_i$ may finish the rest of the computation using $\{\mathsf{sk}_j\}_{j \neq i}$.
     • If only $\{\mathsf{sk}_j\}_{j \neq i}$ are needed, $\mathcal{A}_i$ answers the queries by itself.
   – During the interaction of stage-1 game, $\mathcal{A}^1$ and $\mathcal{A}_i^1$ generates an auxiliary information $\mathsf{aux}$ which is public to both of them.
   – Upon entering stage 2, $\mathcal{A}^1$ generates an arbitrarily polynomial-size state $\mathsf{st}$ and gives it to $\mathcal{A}^2$.
   – $\mathcal{A}_i$ also enters its second stage $\mathcal{A}_i^2$. $\mathcal{A}_i^2$ also receives all of $(\{\mathsf{sk}_j\}_{j \neq i}, \{\mathsf{pk}\}_{i \in \mathcal{S}}, \mathsf{aux})$ from But $\mathcal{A}_i^2$ does *not* obtain any of $\mathcal{A}^1$'s queries in stage 1.
   – $\mathcal{A}_i^2$ simulates the stage-2 game $G_P^2$ for $\mathcal{A}^2$, using the above strategy as $\mathcal{A}_i^1$ uses. Note that the oracle operations done by $\mathcal{A}_i^2$ and the conditions on admissible queries *cannot* be dependent on of $\mathcal{A}^1$'s queries in stage 1, because $\mathcal{A}_i^2$ cannot see these queries.
   – $\mathcal{A}_i^2$ also records all of $\mathcal{A}^2$'s queries.
   – In the challenge phase of $G_{P_i}^2$ $\mathcal{A}_i^2$ receives a challenge input $\mathsf{inp}_i$ from the challenger.
   – In $G_P$'s challenge phase, $\mathcal{A}_i^2$ samples some randomness $r$ and prepares a challenge input $\mathsf{inp}$ for $\mathcal{A}$ using $r$ and $\mathsf{inp}_i$. $\mathcal{A}_i^2$ sends $\mathsf{inp}_i$ to $\mathcal{A}_2$.[6]
   – $\mathcal{A}_i^2$ continues to simulate the oracles needed in $G_P^2$ game for $\mathcal{A}^2$ if there are further query stages.

[6] Note that $G_P$'s challenge phase may be before, after or concurrent with $G_{P_i}$'s challenge phase depending on the reduction. Similarly, the challenge $\mathcal{A}^2$ receives may be dependent on $\mathsf{inp}$.

5. $\mathcal{A}_i$ computes an efficiently computable function $f_i(\mathsf{out}, r, \mathcal{Q}, \mathsf{inp}_i)$ on input of $\mathcal{A}^2$'s final answer $\mathsf{out}$, the randomness used to prepare $\mathcal{A}^2$'s challenge input, $\mathcal{A}^2$'s queries $\mathcal{Q}$ and $A_i$'s challenge input $\mathsf{inp}_i$, and gives it to the challenger of primitive $P_i$.

*Remark 6.* In the actual watermarking unremovability reduction, $\mathcal{A}_i^1$ will receive a watermarked secret key $\tilde{\mathsf{sk}}_i$ and simulate the queries required using the oracles in game $G_{P_i}^1$ with $\tilde{\mathsf{sk}}_i$ instead.

More generically, we can also model the queries made by $\mathcal{A}^1$ in stage 1, related to the keys in the set $\mathcal{S}$, $\{\mathsf{sk}_i\}_{i \in \mathcal{S}}$ as some arbitrary polynomial size (admissible) leakage on these keys. In the plain security reduction, such leakage correspond to $\mathcal{A}^1$'s adaptive queries made to the oracles provided. In the watermarking unremovability game, it corresponds to admissible marking queries where we give out marked secret keys.

*Watermarking-Compatible Reduction Type 2.* Watermarking-compatible reduction type 2 operates essentially the same as the above reduction for $P_i$ but for some $i \notin \mathcal{S}$.

The main difference is that in the actual watermarking unremovability security game, in order to simulate oracle queries for $G_P^1$, the stage 1 reduction $\mathcal{A}_i^1$ will actually need oracles in the game $G_{P_i}^1$. Naturally, this is because the primitives in set $\bar{\mathcal{S}}$ are not watermarked in the watermarking composition and the corresponding reduction will not get a watermarked key from the challenger, but only the oracles provided in the plain security game of $G_{P_i}$. Even though this also means that the reduction $A_i^1$ gets to observe $\mathcal{A}^1$'s queries (using the oracles of the game $G_{P_i}$) in the actual watermarking security game, and can make use of them in stage 2, $A_i^1$ does not need to make any of such queries to the challenger since the keys of any primitive in set $\bar{\mathcal{S}}$ will only be sampled online by $A_i^1$ itself. Meanwhile, $\mathcal{A}_i^2$ should still not inherit any queries related to the "watermarked" primitives in set $\mathcal{S}$ since in the real watermarking unremovability game, $\mathcal{A}_i^2$ is not supposed to see them.

1. Consider doing reduction to $P_i$ for some $P_i, i \notin \mathcal{S}$; let $\mathcal{A}_i = (\mathcal{A}_i^1, \mathcal{A}_i^2)$ be the two-stage adversary for primitive $P_i$, where the stages are defined by Definition 3.
2. $\mathcal{A}_i$ receives public key $\mathsf{pk}_i$ from the challenger.
3. $\mathcal{A}_i$ prepares $(\mathsf{sk}_\ell, \mathsf{pk}_\ell) \leftarrow P_\ell.\mathsf{KeyGen}(1^\lambda)$ for all $\ell \in \mathcal{S}$ $\mathcal{A}_i$ sends all $(\{\mathsf{pk}_\ell)$ to $\mathcal{A}_1$.
4. $\mathcal{A}_i$ simulates the security game $G_P$ for $P$ with $\mathcal{A}$:
   - In stage-1 game $G_P^1$, for any (admissible) queries if $\mathsf{sk}_\ell, \ell \in \mathcal{S}$ is required to compute the answer for the query, $\mathcal{A}_i$ can answer the query since it possesses the keys $\{\mathsf{sk}_\ell\}_{\ell \in \mathcal{S}}$.
     The secret keys of primitives $\{P_j\}_{j \notin \mathcal{S}}$ are all sampled freshly upon every run of $\mathsf{wSecEval}(\mathsf{sk}, \mathsf{pk}, \cdot)$ (or sampled freshly in $\mathsf{wPubEval}(\mathsf{pk}, \cdot)$), which $\mathcal{A}_i^1$ can do it on its own.
   - During the interaction of stage-1 game, $\mathcal{A}^1$ and $\mathcal{A}_i^1$ generates an auxiliary information $\mathsf{aux}$ which is public to both of them.

- Entering stage 2, $\mathcal{A}^1$ generates an arbitrarily polynomial-size state $\mathsf{st}$ and gives it to $\mathcal{A}^2$.

  $\mathcal{A}_i$ also enters its second stage $\mathcal{A}_i^2$. $\mathcal{A}_i^2$ also receives all of $(\{\mathsf{sk}_\ell\}_{\ell\in\mathcal{S}}, \{\mathsf{pk}_\ell\}_{\ell\in\mathcal{S}}, \mathsf{aux})$ but does *not* obtain any of $\mathcal{A}^1$'s queries involving the use of $\{\mathsf{sk}_\ell\}_{\ell\notin\mathcal{S}}$.
- $\mathcal{A}_i^2$ continues to provide the oracles needed for game $G_P^2$ and answer $\mathcal{A}^2(\mathsf{st})$'s queries using the above strategy as $A_i^1$ uses.
- $\mathcal{A}_i^2$ also records all of $\mathcal{A}^2$'s queries.
- In the challenge phase of $G_{P_i}$, $\mathcal{A}_i^2$ receives a challenge input $\mathsf{inp}_i$ from the challenger.
- In $G_P^2$'s challenge phase, $\mathcal{A}_i^2$ samples some randomness $r$ and prepares a challenge input $\mathsf{inp}$ for $\mathcal{A}$ using $r$ and $\mathsf{inp}_i$. $\mathcal{A}_i^2$ sends $\mathsf{inp}_i$ to $\mathcal{A}_2$.
- $\mathcal{A}_i^2$ continues to simulate the oracles needed in $G_P^2$ game for $\mathcal{A}^2$ if there are further query stages

5. $\mathcal{A}_i$ computes an efficiently computable function $f_i(\mathsf{out}, r, \mathcal{Q}, \mathsf{inp}_i)$ on input of $\mathcal{A}^2$'s final answer $\mathsf{out}$, its queries $\mathcal{Q}$ and $A_i$'s challenge input $\mathsf{inp}_i$ and secrret radomness $r$, and gives it to the challenger of primitive $P_i$.

*Properties of Watermarking-Compatible Reductions.* We further give some properties of watermarking-compatible reduction. These properties come with any natural black-box security roof with hybrid argument and reductions. We present them here for the sake of convenience and use them as facts later.

**Fact 1 (Reduction Property 1).** A watermarking-compatible reduction guarantees that: if there exists some $\mathcal{A}$ such that the advantage of $\mathcal{A}^2$ winning the game $G_P$ is non-negligible, i.e. $\Pr[G_P(1^\lambda, A) = 1] \geq \eta + \gamma$ where $\eta$ is the trivial winning probability and $\gamma$ is non-negligible in $\lambda$, then there exists some $i \in [k]$ such that $\mathcal{A}_i$, using the above reduction strategy, wins $G_{P_i}$ with some non-negligible advantage $\gamma_i$.

*Remark 7.* The above property follows naturally from any hybrid argument of proof.

If using the language of the 2-stage security game: If there exists some adversary $\mathcal{A} = (\mathcal{A}^1, \mathcal{A}^2)$ such that for some non-negligible $\epsilon$ and some non-negligible $\gamma$, we have:

$$\Pr\left[\Pr[G_P^2(\mathsf{sk}, \mathsf{pk}, \mathsf{aux}, \mathcal{A}_2(\mathsf{st})) = 1] \geq \eta + \gamma : \{(\mathsf{sk}, \mathsf{pk}), \mathsf{aux}, \mathsf{st}\} \leftarrow G_P^1(\mathcal{A}_1, 1^\lambda)\right] \geq \epsilon$$

Then there exists some $i \in [k]$ such that $\mathcal{A}_i = (\mathcal{A}_i^1, \mathcal{A}_i^2)$, using the above reduction strategy, such that for some non-negligible $\gamma_i, \epsilon_i$, we have:

$$\Pr\left[\Pr[G_{P_i}^2(\mathsf{sk}_i, \mathsf{pk}_i, \mathsf{aux}, \mathcal{A}_i^2(\mathsf{st})) = 1] \geq \eta + \gamma_i : \{(\mathsf{sk}_i, \mathsf{pk}_i), \mathsf{aux}, \mathsf{st}\} \leftarrow G_{P_i}^1(\mathcal{A}_i^1, 1^\lambda)\right] \geq \epsilon_i.$$

**Fact 2 (Reduction Property 2).** For all $i \in \mathcal{S}$, once give the secret key $P_i.\mathsf{sk}$ in the clear, there exists a PPT algorithm $T_i$ that wins the security game $G_P$ with probability 1.

Additionally, such a $T_i$ can be used black-boxly by the watermarking-compatible reduction algorithm $\mathcal{A}_i$ to win the security game $G_{P_i}$ with noticeable probability.

*Reduction Function.* Recall that in the end of the reduction, $\mathcal{A}_i$ computes an efficiently computable function $f_i(\mathsf{out}, r, \mathcal{Q}, \mathsf{inp}_i)$ on input of $\mathcal{A}^2$'s final answer $\mathsf{out}$, the randomness used to prepare $\mathcal{A}^2$'s challenge input, $\mathcal{A}^2$'s queries $\mathcal{Q}$ and $A_i$'s challenge input $\mathsf{inp}_i$, and gives it to the challenger of primitive $P_i$.

For convenience, we will refer to the function $f_i$ used by the reduction as *reduction function.*

*Remark 8 (Reduction Function).* We refer to the above function $f_i$ used by the reduction as *reduction function.*

*Remark 9 (Special Primitives in $\mathcal{T}_K$).* The role of the primitives in the set $\mathcal{T}_K$ may be confusing to the readers at this point. We make some further explanation. As we go into later sections and go into examples (see full version for details ), its role will become clear.

More specifically, $\mathcal{T}_k$ only contains the simulation-based primitives $P_\ell$ where

1. there exists an efficient simulator algorithm such that the output of $P_\ell.\mathsf{PubEval}(P_\ell.\mathsf{pk}, \cdot)$ is indistinguishable from the output of the simulator $\mathsf{Sim}(P_\ell.\mathsf{td}, \cdot)$. Their secret key (trapdoor $\mathsf{td}$) will only come up in the security proof for $P$.
2. In the security proof for $P$, the rest of the hybrid arguments and reduction happen in a hybrid world where we have already invoked the above security for $P_\ell$.

The one example primitive in the set $\mathcal{T}_K$ we use within this work is a NIZK scheme with a trapdoor. Its "keys" (the common reference string $\mathsf{CRS}$, and trapdoor $\mathsf{td}$) are generated in the main key generation algorithm, but the trapdoor is not used in any of $\mathsf{SecEval}, \mathsf{PubEval}$, and only in the security proof.

## 3 Watermarking Composition Framework

### 3.1 Definition: Watermarkable Implementation of a Cryptographic Primitive

In this section, we will define what we call a "watermarkable implementation" of a cryptographic primitive $P$. We distinguish it from the usual naming ("watermarkable P") because of some differences in syntax and definition. But we will sometimes use watermarkable P in our work for convenience. Please note that all "watermarkable P" used in the technical part of this work refers to a watermarkable implementation of a cryptographic primitive $P$ defined below.

On a high-level, a watermarkable implementation of $P$ differs from most existing watermarking definition in two aspects:

1. Unremovability security: A pirate circuit produced by the adversary in the unremovability security game is considered to function successfully as long as it can break the security game $G_P$.
2. Extraction algorithm syntax: the extraction key used to extract a watermark is able to simulate the game $G_P$ for the underlying "plain security" of $P$.

In more detail, a watermarkable implementation of a cryptographic primitive has the following syntax and properties.

*Watermarkable Primitive Syntax* A watermarkable primitive $\mathsf{WP}$ for a cryptographic primitive $P$ with a security game $G_P$ consists of the following algorithms:

$\mathsf{WMSetup}(1^\lambda) \rightarrow (\mathsf{sk}, \mathsf{pk}, \mathsf{mk}, \mathsf{xk})$: on security parameter, outputs a secret key $\mathsf{sk}$, public key $\mathsf{pk}$, marking key $\mathsf{mk}$, extraction key $\mathsf{xk}$.

$\mathsf{wSecEval}(\mathsf{sk}, \mathsf{pk}, x)$: takes in the secret information $\mathsf{sk}$, public information $\mathsf{pk}$ and some input $x$ from input space $\mathcal{X}$, and output a value $y \in \mathcal{Y}_s$.

$\mathsf{wPubEval}(\mathsf{pk}, x \in \mathcal{X})$: takes in the public information $\mathsf{pk}$ and some input $x$ from input space $\mathcal{X}$, and output a value $y \in \mathcal{Y}_p$.

$\mathsf{Mark}(\mathsf{mk}, \mathsf{sk}, \tau \in \mathcal{M}_{\mathsf{Mark}}) \rightarrow \mathsf{sk}_\tau$: takes in marking key $\mathsf{mk}$, a secret key $\mathsf{sk}$ and a message $\tau \in \mathcal{M}_{\mathsf{Mark}}$; output a marked key $\mathsf{sk}_\tau$[7].

$\mathsf{Extract}(\mathsf{xk}, \mathsf{pk}, \mathsf{aux}, C) \rightarrow \tau \in \mathcal{M}_{\mathsf{Mark}}/\vec{\tau} \in \mathcal{M}_{\mathsf{Mark}}^k/\perp$: on input extraction key $\mathsf{sk}$, public key $\mathsf{pk}$, auxiliary information $\mathsf{aux}$ and circuit $C$, outputs a mark message $\tau \in \mathcal{M}_\tau$ or a tuple of marked messages $\vec{\tau} \in \mathcal{M}_\tau^k$ where $k$ is a constant/small polynomial parameter related to the concrete watermarking construction.

*Remark 10.* The $\mathsf{Extract}$ algorithm is allowed to output a tuple of marks when working on adversarial programs, when the watermarkable implementation is one that comes from composing underlying watermarking implementations. More details to be discussed later in Sect. 3.2.

*Remark 11.* The extraction algorithm takes in the public key $\mathsf{pk}$ (which is not a limitation because it's public) and an auxiliary input $\mathsf{aux}$. As we will see in the concrete examples, depending on the security game of the primitives we watermark, we may need $\mathsf{Extract}$ to take in some $\mathsf{aux}$ or may not.

When it is clear from the context that there is no public key or auxiliary information for $\mathsf{Extract}$ to take, we will omit them from the input parameters for notation cleanness.

*Remark 12.* There can be several different (constant number of) $\mathsf{wSecEval}$ and $\mathsf{wPubEval}$. As aforementioned, we view all secret algorithms as one algorithm $\mathsf{SecEval}(\mathsf{sk}, \mathsf{pk}, \cdot)$ that will take in an index $i \in [\ell]$ to decide which mode to use, similarly for public algorithms. We thus also use $\mathsf{sk}$ to denote $(\{\mathsf{sk}_i\}_{i\in[\ell]}$.

Let us denote $\mathsf{Eval} = (\mathsf{wSecEval}, \mathsf{wPubEval})$.

*Correctness.* The construction should satisfy "unmarked" correctness for unmarked keys: a watermarkable implementation of $P$ is said to be correct with

[7] In defintions in the watermarking literature, the output of $\mathsf{Mark}$ is the program $\mathsf{wSecEval}(\mathsf{sk}_\tau, \mathsf{pk}, \cdot)$. Since conventionally by Kerkhoff's principle, the evaluation algorithm itself is public, giving out only the marked secret key is equivalent to giving out the program, we give out the key for convenience that come up later. There are watermarking schemes where the evaluation algorithm may be different when running on a marked key; in that case we can consider $\mathsf{wSecEval}$ to have two modes, one for unmarked keys one for marked keys.

respect to predicate $F_R$ if there exists a negligible function $\mathsf{negl}(\cdot)$ such that for all $\lambda \in \mathbb{N}, x \in \mathcal{X}$:

$$\Pr[F_R(\mathsf{Eval}, \mathsf{pk}, \mathsf{sk}, x, r) = 1 : (\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{WMSetup}(1^\lambda), r \leftarrow D_r] \geq 1 - \mathsf{negl}(\lambda).$$

where the probability is taken over randomness used in $\mathsf{WMSetup}$ and $D_r$. Recall that randomness $r$ is used in checking whether the predicate $F_R(\mathsf{Eval}, \mathsf{pk}, \mathsf{sk}, x, r)$ is satisfied Definition 1 and $D_r$ can be simply taken to be the uniform distribution.

*Functionality Preserving vs. Exact Functionality Preserving.* We present both definitions of functionality-preserving for the sake of comprehensiveness because they have both appeared in watermarking literatures. Before [GKM+19], the watermarking literature mainly used exact functionality preserving only. While it is not particularly vital to the contributions in this work, we would like to mention that exact functionality preserving is a stronger notion. For some of the underlying building blocks we use, only constructions under the relaxed functionality-preserving definition are known, for example the watermarkable signature scheme in [GKM+19].

*Functionality Preserving.* The functionality-preserving property says: if for a predicate $F_R$, the underlying primitive $P$ satisfies he correctness in Definition 2, then there exists a negligible function $\mathsf{negl}(\lambda)$ ssuch that for all $\lambda \in \mathbb{N}, x \in \mathcal{X}$:

$$\Pr\left[F_R(\mathsf{Eval}, \mathsf{pk}, \tilde{\mathsf{sk}}, x, r) = 1 : \begin{array}{c}(\mathsf{sk}, \mathsf{pk}, \mathsf{xk}, \mathsf{mk}) \leftarrow \mathsf{WMSetup}(1^\lambda) \\ \tilde{\mathsf{sk}} \leftarrow \mathsf{Mark}(\mathsf{mk}, \mathsf{sk}, \tau), r \leftarrow D_r\end{array}\right] \geq 1 - \mathsf{negl}(\lambda).$$

*Exact Functionality Preserving.* The exact functionality preserving is a stronger property that: there exists a negligible function $\mathsf{negl}(\lambda)$ such that for all $\lambda \in \mathbb{N}, x \in \mathcal{X}, \tau \in \mathcal{M}_\tau$:

$$\Pr\left[\mathsf{wSecEval}(\tilde{\mathsf{sk}}, \mathsf{pk}, x) = \mathsf{SecEval}(\mathsf{sk}, \mathsf{pk}, x) : \begin{array}{c}(\mathsf{sk}, \mathsf{pk}, \mathsf{xk}, \mathsf{mk}) \leftarrow \mathsf{WMSetup}(1^\lambda) \\ \tilde{\mathsf{sk}} \leftarrow \mathsf{Mark}(\mathsf{mk}, \mathsf{sk}, \tau)\end{array}\right] \geq 1 - \mathsf{negl}(\lambda). \quad (1)$$

*Correctness of Extraction* There exists a publicly known efficient algorithm $T$ and a negligible function $\mathsf{negl}(\lambda)$ such that for all $\lambda \in \mathbb{N}$, for all $\tau \in \mathcal{M}_\tau$:

$$\Pr\left[\mathsf{Extract}(\mathsf{xk}, T(\mathsf{sk}_\tau, \cdot)) = \tau : \begin{array}{c}(\mathsf{sk}, \mathsf{pk}, \mathsf{xk}, \mathsf{mk}) \leftarrow \mathsf{WMSetup}(1^\lambda) \\ \tilde{\mathsf{sk}} \leftarrow \mathsf{Mark}(\mathsf{mk}, \mathsf{sk}, \tau)\end{array}\right] \geq 1 - \mathsf{negl}(\lambda)$$

*Security with Security Game $G_P$.* The watermarkable primitive $\mathsf{WP}$ satisfies the same security defined by the security game $G_P$ for $P$, except that the subroutines used in $G_P$, $\mathsf{KeyGen}, \mathsf{SecEval}, \mathsf{PubEval}$ are replaced with $\mathsf{WMSetup}, \mathsf{wSecEval}, \mathsf{wPubEval}$ respctively. The $\mathsf{Extract}, \mathsf{Mark}$ algorithms and keys are ignored in the context of these games.

*Watermarking Security Compatible with a Security Game.* Now we present the unremovability security of the watermarking implementation for primitive P.

Informally, the security guarantees that: any PPT adversary $\mathcal{A}$, when given the watermarked key(s), generates a pirate circuit $C^*$; if $C^*$ can win the security game $G_P$ with some non-negligible advantage, then we must be able to extract a (previously queried) watermark from $C^*$.

**Definition 5 ($\gamma$-Unremovability with Game $G_P$).** *We say a watermarkable implementation of $P$ is $\gamma$-unremovable if:*

*For every stateful $\gamma$-unremovable admissible PPT adversary $\mathcal{A}$, there exists a negligible function $\mathsf{negl}(\cdot)$ such that for all $\lambda \in \mathbb{N}$, the following holds:*

$$\Pr\left[\mathsf{Extract}(\mathsf{xk}, \mathsf{pk}, \mathsf{aux}, C^*) \notin \mathcal{Q} : \begin{array}{c} (\mathsf{sk}, \mathsf{pk}, \mathsf{mk}, \mathsf{xk}) \leftarrow \mathsf{WMSetup}(1^\lambda) \\ (\mathsf{aux}, C^*) \leftarrow \mathcal{A}^{\mathsf{Mark}(\mathsf{mk},\mathsf{sk},\cdot)}(1^\lambda, \mathsf{pk}) \end{array}\right] \leq \mathsf{negl}(\lambda)$$

*$\mathcal{Q}$ denotes the set of f marks queried by $\mathcal{A}$ to the marking oracle $\mathsf{Mark}(\mathsf{mk}, \mathsf{sk}, \cdot)$, $G_P^1$ is the stage 1 of a security game $G_P$, as defined in Definition 3.*

*We call a PPT adversary $\mathcal{A}$ as $\gamma$-unremovable admissible if $\mathcal{A}$'s output $C^*$ is an admissible adversary in the* stage-2 *security game $G_P^2$ and is $\gamma$-good, where $\gamma$-good is defined as:*

$$\Pr[G_P^2(\mathsf{sk}, \mathsf{pk}, \mathsf{aux}, C^*) = 1] \geq \eta + \gamma$$

*Here, $\eta$ is the trivial success probability for any admissible adversary in $G_P$.*

*The randomness over testing whether $C^*$ is $\gamma$-good is the randomness used answering any oracle queries from $C^*$ and preparing the challenge for $C^*$, in the stage-2 security game $G_P^2$.*

*To match our syntax, we denote $\mathsf{Extract}(\mathsf{xk}, \mathsf{pk}, \mathsf{aux}, C^*) \notin \mathcal{Q}$ to mean that:*

1. *If $\mathsf{Extract}(\mathsf{xk}, \mathsf{pk}, \mathsf{aux}, C^*)$ outputs a single mark $\tau \in \mathcal{M}_\tau / \tau = \perp$, it is considered to be not in the query set $\mathcal{Q}$ if and only if $\tau \notin \mathcal{Q}$.*
2. *If $\mathsf{Extract}(\mathsf{xk}, \mathsf{pk}, \mathsf{aux}, C^*)$ outputs a tuple of marks $\vec{\tau} = (\tau_1, \cdots, \tau_k)$, where each $\tau_i \in \mathcal{M}_\tau / \tau_i = \perp$, it is considered not in the query set $\mathcal{Q}$ if and only if for all $i$, $\tau_i \notin \mathcal{Q}$.*

For more discussions on the above definition, we refer to the full version of the paper.

**Definition 6 (Strong Unremovability).** *We say a watermarking implementation of $P$ satisfies strong unremovability if it satisfies $\gamma$-unremovability for any non-negligible $\gamma$.*

We leave more remark on the unremovability definition to the full version.

*Security Game Simulation Property of the Extraction Key* We additionally require the extraction key $\mathsf{Extract}$ to be able to simulate the (stage-2) security game $G_P^2$ for the primitive $P$ to be watermarked.

Consider the security game $G_P$ for the underlying primitive $P$: $G_P = (G_P^1, G_P^2)$ is an interactive game between a challenger and admissible $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$. Recall that in the stage-2 Definition 3 $G_P^2$ can output a view $\mathsf{view}(\mathsf{sk}, \mathsf{pk}, \mathsf{aux}, \mathcal{A}_2)$ (Definition 4).

**Definition 7 (Extraction key simulation property).** *The extraction key simulation property of the extraction key says that: given* $(\mathsf{sk}, \mathsf{pk}, \mathsf{aux}) \leftarrow G_P^1(\mathcal{A}_1, 1^\lambda)$ *where we use* $\mathsf{WMSetup}(1^\lambda)$ *to obtain* $(\mathsf{sk}, \mathsf{pk}, \mathsf{xk})$ *in* $G_P^1$*, there exists a PPT simulator* $\mathsf{Sim}(\mathsf{xk}, \mathsf{pk}, \mathsf{aux}, \mathcal{A}_2)$ *( where* $\mathsf{Sim}$ *interacts with* $\mathcal{A}_2$ *black-boxly) that outputs a simulated view* $\mathsf{view}_{\mathsf{Sim}}(\mathsf{xk}, \mathsf{pk}, \mathsf{aux}, \mathcal{A}_2)$*, such that for any* $\lambda \in \mathbb{N}$*, for any admissible* $\mathcal{A}$ *in* $G_P$*, the distributions* $\mathsf{view}_{G_P^2} = \{\mathsf{view}(\mathsf{sk}, \mathsf{pk}, \mathsf{aux}, \mathcal{A}_2)\}$ *and* $\mathsf{view}_{\mathsf{Sim}} = \{\mathsf{view}_{\mathsf{Sim}}(\mathsf{xk}, \mathsf{pk}, \mathsf{aux}, \mathcal{A}_2)\}$ *are perfectly/statistically/computationally indistinguishable.*

*Extraction Syntax: Simulation of Security Game* $G_P$ *inside* Extract. Following the above property, we require that the Extract algorithm in a watermarkable implementation of a primitive $P$ follows a specific format: it simulates the stage-2 security game $G_P^2$ for any input circuit, while trying to extract a mark.

```
Extract Algorithm

On input (xk, pk, aux) and a program $C$:
 – Run an algorithm $E$ where $E$ uses the following subroutine (for possibly
   poly($\lambda$)) many times):
     • Use (xk, pk, aux) to simulate the stage-2 game $G_P^2(\mathsf{sk}, \mathsf{pk}, \mathsf{aux}, C)$ for
       primitive $P$, running $C$ black-boxly by treating $C$ as the stage-2
       adversary $\mathcal{A}_2$ defined in Definition 3.
     • If $C$ is non-interactive (i.e. $C$ does not make any queries or respond to
       interaction in the challenge phase of $G_P^2$), then $E$ samples challenge
       inputs on its own and runs $C$ on them.
 – $E$ can take any of $C$'s outputs, including $C$'s queries made during the
   above simulated game, as inputs to compute the extraction algorithm's
   final output.
 – Output a watermark $\tau \in \mathcal{M}_\tau$ or $\perp$
```

*Examples of Extraction Algorithm that Simulates a Security Game.* It is not hard to create an extraction algorithm with the above syntax and simulation capability. In fact, some existing works have already built watermarking schemes that have Extract algorithm with such a format.

- To enable the extraction key to simulate the security game, a naive solution in the private extraction case is to simply let xk contain the secret key sk.
  One example is the extraction algorithm in a watermarkable signature scheme (see the full version ). To answer the signature queries for the pirate program, the extraction key is the signing key.
- In some other scenarios, we don't need the secret key to simulate the security game and one can even have public extraction.
  For example, a CPA secure PKE scheme. Another example is when we can sample from the input-output space of the evaluation oracles without having the actual key: in the weak PRF setting (with non-adaptive queries), we can simply answer its queries by sampling. In [GKWW21], one can use indistinguishability obfuscation to build such a sampler, so that we have watermarkable weak PRF with public extraction.

See the full version for more discussions on the extractability of watermarks, simulation property of the extraction key and additional definitions such as meaningfulness and collusion resistance.

### 3.2 Watermarking Composition: Target Primitive from Input Primitives

In this section, we show that if primtivive $P$ is built from $P_1, \cdots, P_k$ where the security can be shown via a watermarking-compatible reduction, we can construct a watermarkable implementation of $P$, named $\mathsf{WP}$ to satisfy definition in 3.1, from existing watermarkable implementations of $P_1, \cdots, P_k$ called $\mathsf{WP}_1, \mathsf{WP}_2, \cdots, \mathsf{WP}_k$, satisfying definition in 3.1. We will still refer to $P$ as the target primitive and $P_1, \cdots, P_k$ as input primitives.

*Outline and Intuition.* On a high level, the watermarking scheme of the target primitive $P$ simply follows the construction of plain $P$ from the plain underlying building blocks in terms of evaluation algorithms $\mathsf{SecEval}, \mathsf{PubEval}$. Correctness and functionality-preserving compose in a relatively natural way.

To mark a key of $P$, the marking algorithm concatenates the marked keys of all underlying $P_i$ that need to be marked. To extract a mark, we attempt to run the extraction algorithm of all underlying $P_i$ on the input circuit. If any valid mark is extracted, then the circuit is considered marked.

In particular, the extraction algorithm will treat the circuit as an adversary in the stage-2 security game of $P$ and turns it black-boxly into a reduction for the security game of each underlying $P_i$, one by one and then run the underlying extraction algorithm of $P_i$ on it.

In order to remove a mark from a $P$'s key, the adversary $\mathcal{A}$ must remove all marks from each $P_i$'s key. Meanwhile, the pirate program still needs to win the security game of $P$, then we must be able to use to break the unremovability of at least one underlying $P_i$. In more generic scenarios, the pirate program made by $\mathcal{A}$ may not break any unremovability security of watermarkable building blocks, but get around the task of removing marks by breaking the security of some unwatermarked building blocks. By similar means, we can use the pirate program to build our reduction to the security of these unwatermarked building blocks. By the properties of the watermarking-compatible reduction (Sect. 2.2) that $P$'s construction satisfies, the above analysis is exhaustive.

*Construction of* $\mathsf{WP}$. Similar to the description of construction in Sect. 2.2, we recall the following notations:

- Let $\mathcal{S} \subset [k]$ be a fixed set used in $P$'s construction from $P_1, \cdots, P_k$, where the primitives $P_i$ with $i \in \mathcal{S}$'s keys will be generated in the $\mathsf{KeyGen}$ algorithm of $P$. Without loss of generality, we let the first $|\mathcal{S}|$ number of $P_i$'s be those corresponding to the set $\mathcal{S}$.
- Let $\mathcal{T}_S \in [k]$ denote the set of indices $i$ where $P_i$'s keys will be generated during $\mathsf{SecEval}$; Let $\mathcal{T}_P \in [k]$ denote the set of indices $i$ where $P_i$'s keys will

be generated during PubEval. Let $\mathcal{T}_K$ denote the set of indices $i$ where $P_i$'s secret key (we call trapdoor td) will be generated during $\mathsf{KeyGen}(1^\lambda)$ but will not be used in $P$'s algorithms.

– Without loss of generality, we sometimes assume a numbering on the primitives so that the first $|\mathcal{S}|$ primitives are in the set $\mathcal{S}$.

```
$\mathsf{WMSetup}(1^\lambda) \to (\mathsf{sk}, \mathsf{pk}, \mathsf{xk}, \mathsf{mk})$ :
    1. compute $(\mathsf{sk}_i, \mathsf{pk}_i, \mathsf{xk}_i, \mathsf{mk}_i) \leftarrow \mathsf{WP}_i.\mathsf{WMSetup}(1^\lambda)$ for all $i \in \mathcal{S}$.
    2. compute $(\mathsf{pk}_\ell, \mathsf{td}_\ell) \leftarrow P_\ell.\mathsf{KeyGen}(1^\lambda)$ for all $\ell \in \mathcal{T}_K$;
    3. output $\mathsf{sk} = (\mathsf{sk}_{j_1}, \cdots, \mathsf{sk}_{|\mathcal{S}|})$; $\mathsf{pk} = (\mathsf{pk}_1, \cdots, \mathsf{pk}_{|\mathcal{S}|)}, \{\mathsf{pk}_\ell\}_{\ell \in \mathcal{T}_k}))$;
       $\mathsf{xk} = (\mathsf{xk}_1, \cdots, \mathsf{xk}_{|\mathcal{S}|}, \{\mathsf{td}_\ell\}_{\ell \in \mathcal{T}_k})$; $\mathsf{mk} = (\mathsf{mk}_1, \cdots, \mathsf{mk}_{|\mathcal{S}|})$
$\mathsf{wSecEval}(\mathsf{sk}, \mathsf{pk}, x)$:
    1. parse input $\mathsf{sk} = (\mathsf{sk}_1, \cdots, \mathsf{sk}_{|\mathcal{S}|})$; $\mathsf{pk} = (\mathsf{pk}_1, \cdots, \mathsf{pk}_{|\mathcal{S}|}, \{\mathsf{pk}_\ell\}_{\ell \in \mathcal{T}_k}))$.
    2. $\mathsf{wSecEval}(\mathsf{sk}, \mathsf{pk}, )$ is the same algorithm as $P.\mathsf{SecEval}(\mathsf{sk}, \mathsf{pk}, \cdot)$ in the construction of $P$ from $P_1, \cdots, P_k$, except that $P_i.\mathsf{SecEval}(\mathsf{sk}_i, \cdot)$ is replaced with $\mathsf{WP}_i.\mathsf{wSecEval}(\mathsf{sk}_i, \cdot)$ for $i \in \mathcal{S}$. Overall, $\mathsf{wSecEval}(\mathsf{sk}, \mathsf{pk}, )$ is an algorithm that:
       (a) uses $\mathsf{WP}_i.\mathsf{wSecEval}(\mathsf{sk}_i, \mathsf{pk}_i, \cdot), i \in \mathcal{S}$ as subroutines.
       (b) uses $\mathsf{WP}_i.\mathsf{wPubEval}(\mathsf{pk}_i, \cdot), i \in \mathcal{S} \cup \mathcal{T}_k$ as subroutines.
       (c) computes $(\mathsf{sk}_j, \mathsf{pk}_j) \leftarrow P_j.\mathsf{KeyGen}(1^\lambda)$ for some $j \in \mathcal{T}_S$ ( and may include these $\mathsf{pk}_j$ generated as part of the output).
$\mathsf{wPubEval}(\mathsf{pk}, x)$
    1. parse input $\mathsf{pk} = (\mathsf{pk}_1, \cdots, \mathsf{pk}_{|\mathcal{S}|}, \{\mathsf{pk}_\ell\}_{\ell \in \mathcal{T}_k})$.
    2. $\mathsf{wPubEval}(\mathsf{pk}, )$ is the same algorithm as $P.\mathsf{PubEval}(\mathsf{pk}, \cdot)$ in the construction of $P$ from $P_1, \cdots, P_k$, except that $P_i.\mathsf{PubEval}(\mathsf{pk}_i, \cdot)$ is replaced with $\mathsf{WP}_i.\mathsf{wPubEval}(\mathsf{pk}_i, \cdot)$ for $i \in \mathcal{S} \cup \mathcal{T}_k$. Overall, $\mathsf{wPubEval}(\mathsf{sk}, \mathsf{pk}, \cdot)$ is an algorithm that:
       (a) uses $\mathsf{WP}_i.\mathsf{wPubEval}(\mathsf{pk}_i, \cdot), i \in \mathcal{S}$ as subroutines.
       (b) uses $\mathsf{WP}_i.\mathsf{wPubEval}(\mathsf{pk}_i, \cdot), i \in \mathcal{S}$ as subroutines.
       (c) computes $(\mathsf{sk}_j, \mathsf{pk}_j) \leftarrow P_j.\mathsf{KeyGen}(1^\lambda)$ for some $j \in \mathcal{T}_P$ ( and may include these $\mathsf{pk}_j$ generated as part of the output).
$\mathsf{Mark}(\mathsf{mk}, \mathsf{sk}, \tau \in \mathcal{M}_{\mathsf{Mark}}) \to \mathsf{sk}_\tau$
    1. parse $\mathsf{mk} = (\mathsf{mk}_1, \cdots, \mathsf{mk}_{|\mathcal{S}|})$; $\mathsf{sk} = (\mathsf{sk}_1, \cdots, \mathsf{sk}_{|\mathcal{S}|})$; $\mathsf{pk} = (\mathsf{pk}_1, \cdots, \mathsf{pk}_{|\mathcal{S}|}, \{\mathsf{pk}_\ell\}_{\ell \in \mathcal{T}_k})$.
    2. Compute $\mathsf{sk}_{i,\tau} \leftarrow \mathsf{WP}_i.\mathsf{Mark}(\mathsf{mk}_i, \mathsf{sk}_i, \tau)$ for all $i \in \mathcal{S}$.
    3. output $\tilde{\mathsf{sk}} = (\mathsf{sk}_{1,\tau}, \cdots, \mathsf{sk}_{|\mathcal{S}|,\tau})$
$\mathsf{Extract}(\mathsf{xk}, \mathsf{pk}, \mathsf{aux}, C) \to \tau \in \mathcal{M}_{\mathsf{Mark}} / \vec{\tau} \in \mathcal{M}_{\mathsf{Mark}}^{|\mathcal{S}|} / \bot$ :
    1. parse $\mathsf{xk} = (\mathsf{xk}_1, \cdots, \mathsf{xk}_{|\mathcal{S}|}, \{\mathsf{td}_\ell\}_{\ell \in \mathcal{T}_K})$; $\mathsf{aux} = (\mathsf{aux}_1, \cdots, \mathsf{aux}_{|\mathcal{S}|})$; $\mathsf{pk} = (\mathsf{pk}_1, \cdots, \mathsf{pk}_{|\mathcal{S}|}, \{\mathsf{pk}_\ell\}_{\ell \in \mathcal{T}_k})$.
    2. Initialize an empty set $\vec{\tau}$.
    3. For each $i \in \mathcal{S}$:
       (a) prepare the following circuit $C_i$ using black-box access to $C$.
          i. $C_i$ uses $(\{\mathsf{xk}_j\}_{j \in \mathcal{S}, j \neq i}, \{\mathsf{td}_\ell\}_{\ell \in \mathcal{T}_K})$ and external queries to simulate security game $G_P^2(\mathsf{sk}, \mathsf{pk}, \mathsf{aux}, C)$ for $C$[8]:
```

[8] $C_i$ can also have $\mathsf{xk}_i$ if the watermarkable implementation $\mathsf{WP}_i$ has a public extraction key. In this case, $C_i$ also does not need to make external queries.

   - For any (admissible) queries in the stage-2 game $G_P^2$ $(\mathsf{sk}, \mathsf{pk}, \mathsf{aux}, C)$, if the oracles provided in security game $G_{P_i}(\mathsf{sk}_i, \mathsf{pk}_i, \mathsf{aux}_i, \cdot)$ is required to compute the answer for the query, $C_i$ can will make a query to an external challenger. To answer the entire query, $C_i$ may finish the rest of the computation using $(\{\mathsf{xk}_j\}_{j \neq i}, \{\mathsf{td}_\ell\}_{\ell \in \mathcal{T}_k})$.
   - If only $\{\mathsf{xk}_j\}_{j \neq i}$ are needed to answer a query, $C_i$ answers it by itself.

ii. $C_i$ records queries from $C$ into a set $\mathcal{Q}_C$.

iii. $C_i$ receives its challenge input $\mathsf{inp}_i$ from the interaction with an external challenger, samples a random string $r$, and prepares a challenge input $\mathsf{inp}$ for $C$ using $\mathsf{inp}_i$ and $r$.

iv. If there is a query phase after the challenge phase, $C_i$ continues to simulate the oracles required using $\{\mathsf{xk}_j\}_{j \neq i}$ and external queries to a challenger.

v. After $C$ makes its final output $\mathsf{out}$, $C_i$ computes the function $f_i(\mathsf{out}, r, \mathcal{Q}_c, \mathsf{inp})$ where $f_i$ is the reduction function (Remark 8) used in watermarking-compatible reduction of $P$ to $P_i$. Output the result of this computation.

(b) compute $\tau_i/\perp \leftarrow \mathsf{WP}_i.\mathsf{Extract}(\mathsf{xk}_i, C_i)$.

(c) add $\tau_i$(or $\perp$) to the tuple $\vec{\tau}$, and go to step 2 with $i := i + 1$.

4. Output
   - $\vec{\tau} = (\tau_1, \cdots, \tau_{|\mathcal{S}|})$ if $\exists i, j$ where $\tau_i \neq \tau_j$;
   - else if $\tau_i = \tau_j = \tau$(or $\perp$) for all $i, j$, output $\tau$ (or $\perp$ resp.).

We refer the proof of correctness, functionality-preserving, watermarking security and more discussions for the full version.

## References

[BCHK07] Boneh, D., Canetti, R., Halevi, S., Katz, J.: Chosenciphertext security from identity-based encryption. SIAM J. Comput. **36**(5), 1301–1328 (2007)

[CFNP00] Chor, B., Fiat, A., Naor, M., Pinkas, B.: Tracing traitors. IEEE Trans. Inform. Theory **46**(3), 893–910 (2000). https://doi.org/10.1109/18.841169

[CHN+16] Cohen, A., Holmgren, J., Nishimaki, R., Vaikuntanathan, V., Wichs, D.: Watermarking cryptographic capabilities. In: Proceedings of the Forty-Eighth Annual ACM Symposium on Theory of Computing, STOC 2016, pp. 1115–1127. Association for Computing Machinery, Cambridge, MA, USA (2016). https://doi.org/10.1145/2897518.2897651, ISBN: 9781450341325

[GGM86] Goldreich, O., Goldwasser, S., Micali, S.: How to construct random functions. J. ACM **33**(4), 792–807 (1986). https://doi.org/10.1145/6490.6503, ISSN: 0004-5411

[GKM+19] Goyal, R., Kim, S., Manohar, N., Waters, B., Wu, D.J.: Watermarking Public-Key Cryptographic Primitives. In: Boldyreva, A., Micciancio, D. (eds.) CRYPTO 2019. LNCS, vol. 11694, pp. 367–398. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-26954-8_12

[GKP+13] Goldwasser, S., Kalai, Y., Popa, R.A., Vaikuntanathan, V., Zeldovich, N.: Reusable garbled circuits and succinct functional encryption. In: Proceedings of the Forty-fifth Annual ACM Symposium on Theory of Computing, pp. 555–564 (2013)

[GKWW21] Goyal, R., Kim, S., Waters, B., Wu, D.J.: Beyond software watermarking: traitor-tracing for pseudorandom functions. In: Tibouchi, M., Wang, H. (eds.) ASIACRYPT 2021. LNCS, vol. 13092, pp. 250–280. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-92078-4_9

[KN22] Kitagawa, F., Nishimaki, R.: Watermarking PRFs against quantum adversaries. In: Dunkelman, O., Dziembowski, S. (eds.) Advances in Cryptology - EUROCRYPT 2022, pp. 488–518. Springer International Publishing, Cham (2022). https://doi.org/10.1007/978-3-031-07082-2_18

[KW17] Kim, S., Wu, D.J.: Watermarking cryptographic functionalities from standard lattice assumptions. In: Katz, J., Shacham, H. (eds.) CRYPTO 2017. LNCS, vol. 10401, pp. 503–536. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-63688-7_17

[MW22] Maitra, S., David, J.W.: Traceable PRFs: full collusion resistance and active security. In: Hanaoka, G., Shikata, J., Watanabe, Y. (eds.) Public-Key Cryptography - PKC 2022, pp. 439–469. Springer International Publishing, Cham (2022). https://doi.org/10.1007/978-3-030-97121-2_16

[Nis13] Nishimaki, R.: How to watermark cryptographic functions. In: Johansson, T., Nguyen, P.Q. (eds.) EUROCRYPT 2013. LNCS, vol. 7881, pp. 111–125. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-38348-9_7

[Nis20] Nishimaki, R.: Equipping public-key cryptographic primitives with watermarking (or: a hole is to watermark). In: Pass, R., Pietrzak, K. (eds.) TCC 2020. LNCS, vol. 12550, pp. 179–209. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-64375-1_7

[NY90] Nao, M., Yung, M.: Public-key cryptosystems provably secure against chosen ciphertext attacks. In: Proceedings of the Twenty-second Annual ACM symposium on Theory of Computing, pp. 427–437 (1990)

# Space-Lock Puzzles and Verifiable Space-Hard Functions from Root-Finding in Sparse Polynomials

Nico Döttling[1], Jesko Dujmovic[1,2](✉), and Antoine Joux[1]

[1] CISPA Helmholtz Center for Information Security, Saarbrücken, Germany
jesko.dujmovic@cispa.de
[2] Saarland University, Saarbrücken, Germany

**Abstract.** Timed cryptography has initiated a paradigm shift in the design of cryptographic protocols: Using timed cryptography we can realize tasks *fairly*, which is provably out of range of standard cryptographic concepts. To a certain degree, the success of timed cryptography is rooted in the existence of efficient protocols based on the *sequential squaring assumption*.

In this work, we consider space analogues of timed cryptographic primitives, which we refer to as *space-hard* primitives. Roughly speaking, these notions require honest protocol parties to invest a certain amount of space and provide security against space constrained adversaries. While inefficient generic constructions of timed-primitives from strong assumptions such as indistinguishability obfuscation can be adapted to the space-hard setting, we currently lack concrete and versatile algebraically structured assumptions for space-hard cryptography.

In this work, we initiate the study of space-hard primitives from concrete algebraic assumptions relating to the problem of root-finding of sparse polynomials. Our motivation to study this problem is a candidate construction of VDFs by Boneh et al. (CRYPTO 2018) which are based on the hardness of inverting permutation polynomials. Somewhat anticlimactically, our first contribution is a full break of this candidate. However, we then revise this hardness assumption by dropping the permutation requirement and considering arbitrary sparse high degree polynomials. We argue that this type of assumption is much better suited for space-hardness rather than timed cryptography. We then proceed to construct both space-lock puzzles and verifiable space-hard functions from this assumption.

## 1 Introduction

*Timed Cryptography.* Traditionally, in public key cryptography [DH76], the ability to decrypt ciphertexts which have been generated with a public key $pk$ is tied to the possession of a secret key $sk$ corresponding to $pk$. Likewise, generation of signatures with respect to a verification key $vk$ is tied to the possession of corresponding signing key. Timed cryptography [RSW96] adds a twist to this

© International Association for Cryptologic Research 2025
E. Boyle and M. Mahmoody (Eds.): TCC 2024, LNCS 15366, pp. 431–459, 2025.
https://doi.org/10.1007/978-3-031-78020-2_15

rigid paradigm: Rather than the possession of a secret key, *investing time* facilitates the decryption of a ciphertext or generation of signature. In other words, time-lock encryption allows to *encrypt to the future.*

This enables new applications both in theory and practice: Timed commitments [BN00] facilitate e.g. fair exchange and fair coin-toss in the two party setting, notions which have been shown to be beyond reach of standard cryptographic notions [Cle86]; Likewise, from a more practical angle time-lock puzzles play a crucial role in the design of public randomness beacons, a crucial component in the design of distributed ledgers (see e.g. [KWJ23]).

*Verfiable Delay Functions.* Boneh et al. [BBBF18] introduced the notion of *verifiable delay functions* (VDFs), which can be loosely thought of as the timed analogue of digital signatures: Computing a VDF output together with a certificate of its validity takes a long time $T$, whereas verification of a certificate can be performed rapidly, that is in time $\mathsf{poly}(\lambda, \log(T))$. VDFs are likewise powerful tools in the construction of randomness beacons and consensus protocols, as they e.g. facilitate techniques such as *self-selection* [CM19] and proofs of replication [ABBK16].

Boneh et al. [BBBF18] provide both generic and concrete constructions of VDFs. The generic constructions are obtained by combining specific sequential functions, such as iterated hashing, with incrementally verifiable computation [Val08, BCCT13].

Alas, when it comes to concrete assumptions, VDFs in particular and timed cryptography in general rest on a rather narrow foundation; most candidates of time-lock puzzles and verifiable delay functions are tied to the sequential squaring assumption in groups of unknown order. Bitansky et al. [BGJ+16] showed that by relying on indistinguishability obfuscation, timed primitives can be realized assuming the minimal assumption that inherently sequential problems exist. As this construction relies on very heavy theoretical tools, its appeal is currently limited to the domain of pure theory. Another significant candidate are verifiable delay functions from isogenies [DMPS19]. Proofs of sequential [MMV13, CP18, DLM19] can be seen as a more lightweight alternative to VDFs and are achievable from potentially weaker assumptions. However, PoSW lack a uniqueness property, which makes them unsuitable for many of the more advanced applications of VDFs.

The concrete VDF candidates given in [BBBF18] constitute a notable exception from the sequential squaring blueprint. These candidates are based on a novel family of hardness assumptions relating to the inversion of rational functions of high degree.

*Space-Hard Cryptography.* Conceptually, there is nothing intrinsically special about the computational resource of *time*. Hence, a natural conceptual next step is to consider more general computational resources. In fact, there is a growing body of works investigating the notion of *memory or space-hard functions* [Per09, AS15, AB16, ACP+17, BP17, ABB22, AGP24].

In this work, we are concerned with both *space-lock puzzles*, the space-analogue of time-lock puzzles, and verifiable space-hard functions, the analogue of VDFs.

Syntactically, we define a space-lock puzzle to consist of two algorithm $\mathsf{Gen}$ and $\mathsf{Solve}$. $\mathsf{Gen}$ takes as input a space parameter $S$ and a message $m$ and outputs a puzzle $\mathsf{p}$, whereas $\mathsf{Solve}$ takes a puzzle $\mathsf{p}$ and outputs a message $m$. In terms of efficiency, we require that $\mathsf{Gen}$ runs in time and space $\mathsf{poly}(\lambda, \log(S))$, whereas $\mathsf{Solve}$ runs in space $S$. In terms of security, we require that any algorithm running in time $\mathsf{poly}(\lambda, S)$ having access to space of size at most $S^{1-\epsilon}$ has at most negligible advantage guessing an encrypted bit.

A verifiable space-hard function syntactically consists of two algorithms $\mathsf{Eval}$ and $\mathsf{Verify}$ (potentially along with a setup algorithm producing public parameters). $\mathsf{Eval}$ takes a space parameter $S$ and a value $x$ and outputs a value $y$ and a certificate $\pi$, whereas $\mathsf{Verify}$ takes inputs $x$, $y$ and a certificate $\pi$ and outputs either accept or reject. In terms of efficiency, we require that $\mathsf{Eval}$ runs in space $S$, whereas $\mathsf{Verify}$ runs in time and space $\mathsf{poly}(\lambda, \log(S))$. In terms of security, we require *computational uniqueness* and *space-hardness*. Computational uniqueness requires that no algorithm running in time and space $\mathsf{poly}(\lambda, S)$ can produce a verifying tuple $x, y', \pi'$ with $y' \neq y$, where $(y, \pi) = \mathsf{Eval}(S, x)$. Space-hardness requires that no algorithm running in time $\mathsf{poly}(\lambda, S)$ and space $S^{1-\epsilon}$ finds $y$ with non-negligible probability.

In terms of assumptions and constructions, the design-space of space-hard cryptography is comparatively much less explored than that of timed cryptography. In terms of generic constructions, a closer look at the time-lock puzzle construction given in [BGJ+16] reveals that this construction can be adapted to space-lock puzzles, i.e. we can construct a space-lock puzzles assuming iO and, additionally, the minimal assumption that inherently space-hard computations exist.

However critically, there are currently no algebraically structured candidates for efficient space-lock puzzles.

## 1.1 Our Results

In this work, we take a first step in studying efficient space-hard primitives from algebraic assumptions relating to the solvability of sparse univariate polynomials of large degree. The contributions of our work are two-fold:

1. As a first contribution, we provide an efficient attack against the specific proposal of the "Inverting Injective Rational Maps" Assumption of Boneh et al. [BBBF18]. While we do not break the assumption in its most general form, we demonstrate a full break on their suggested candidate, which indicates that the assumption stands on brittle ground. A major challenge towards instantiating the general assumption is finding suitable families of injective rational maps, and [BBBF18] suggested a family of rational functions of (large) degree $d$ constructed by [GM97]. In [BBBF18] it was conjectured that this family cannot be inverted in time and space $\mathsf{poly}(\log(d))$ (i.e. by circuits of size

poly(log($d$))). We provide and implement an algorithm which inverts these rational functions in time and space poly(log($d$)), thus falsifying the main candidate instantiation of the Inverting Injective Rational Maps assumption. We remark that this VDF was a *weak* VDF to begin with, i.e. algorithms that run in time poly(log($d$)) and space $O(d^c)$ for some $c \geq 1$ were known and discussed in [BBBF18]. The main innovation of this part of our work is that our attack runs in *both* time and space poly(log($d$)).

2. In Sect. 4 we introduce and discuss a new algebraically structured space-hardness assumption which we refer to as the *sparse root finding* (SRF) assumption. Building on this, in Sect. 5 we provide a construction of spacelock puzzles from the SRF assumption, whereas in Sect. 6 we construct a verifiable space-hard function from the SRF assumption.

## 1.2 Our Techniques

*Invertibility of Guralnick Müller Polynomials.* As mentioned above, Boneh et al. [BBBF18] provided a concrete candidate for a VDF based on Guralnick-Müller permutation polynomials [GM97]. These are defined via rational functions $f_{\mu,q}$ over a finite field $\mathbb{F}_{p^m}$ and parametrized by an element $\mu \in \mathbb{F}_{p^m}$ and a (large) degree parameter $q = p^r$ (for some $r < m$). Both $\mu$ and $q$ need to obey some additional constraints to ensure that the function is a permutation. The function $f_{\mu,q}(X)$ is then given by

$$f_{\mu,q}(X) = \frac{(X^q - \mu X - \mu)(X^q - \mu X + \mu)^q + ((X^q - \mu X + \mu)^2 + 4\mu^2 X)^{(q+1)/2}}{2X^q}.$$

Notice that this function is neither linear nor affine, consequently at a first glance one may reasonably conjecture that it takes space proportional to $q$ to invert it on random inputs. In fact, [BBBF18] provide a survey of cryptanalytic techniques to invert rational functions and argue why these techniques fail for the case of Guralnick-Müller polynomials. This includes inversion of *extremely* sparse polynomials, linear algebraic attacks, as well as attacks against so-called *exceptional polynomials*, which remain permutations when considered as rational functions over the extension field $\mathbb{F}_{p^{m'}}$ *for infinitely many choices of the degree* $m'$. Boneh et al. [BBBF18] conjecture that inverting a function $f_{\mu,q}$ for a randomly chosen $\mu \in \mathbb{F}_{p^m}$ (under some constraints) takes time polynomial in the degree parameter $q$.

Yet, our first contribution in this work is a full break of this assumption. Interestingly, we draw the mathematical tools for this attack from the original work of Guralnick and Müller [GM97]. We observe the following: While the function $f_{\mu,q}$ itself is not affine, the problem of inverting $f_{\mu,q}$ on a target $t \in \mathbb{F}_{p^m}$ can be *embedded into* a linear system of higher degree. Specifically, [GM97] provides us with the following property of $f_{\mu,q}$: If $\theta$ is $q-1$-st root of $\mu$ in the algebraic closure of $\mathbb{F}_{p^m}$, then there exist efficiently computable coefficients $A_0, B_0, B_1, B_2$ (depending on $a$ and $t$) in an extension of $\mathbb{F}_{p^m}$ such that

$$\prod_{i \in \mathbb{F}_q} (f_{\mu,q}(X + i\theta) - t) = X^{q^3} + B_2 X^{q^2} + B_1 X^q + B_0 X + A_0. \qquad (1)$$

Consequently, any solution $\xi \in \mathbb{F}_{p^m}$ of $f_{\mu,q}(\xi) = t$ is also a solution to the right-hand side of Eq. (1), i.e. such a solution satisfies

$$\xi^{q^3} + B_2\xi^{q^2} + B_1\xi^q + B_0\xi + A_0 = 0. \tag{2}$$

Now observe that Eq. (2) is in fact a linear equation system (as exponentiation with $q$ is a Frobenius action). Hence we can efficiently compute a solution space using standard linear algebra techniques. In Theorem 2 in Sect. 3 we show that, except with negligible probability over the choice of $t$, the system (2) possesses a unique solution, hence our attack succeeds. Furthermore, our experiments in MAGMA (see Sect. A) demonstrate that this attack is indeed practical.

*Space-Hardness from Inverting Sparse Polynomials.* Guralnick-Müller polynomials are one specific instance of the more general problem of finding roots of sparse polynomials[1], a problem which we will refer to as Sparse Root Finding (SRF).

As [BBBF18] note, their root-finding-based candidate achieves only a mild form of sequentiality to begin with. In fact, a moderate polynomial increase in parallel computation power will enable a solver to find roots significantly faster. On the other hand, however, the space-hardness of these problems seems to be much more robust, as all known algorithms for this type of problem consume a large amount of space, in fact an amount of memory that scales linearly with the degree of the polynomial.

This is the starting point for the constructive results in this work. In a nutshell, we consider the problem of inverting sparse, high degree polynomials but drop the requirement that the polynomial needs to act as a permutation. Hence, the resulting problem carries significantly less structure than e.g. inverting Guralnick-Müller polynomials and does not provide an obvious angle for cryptanalysis.

More importantly, by basing our constructions on the problem of root-finding for general sparse polynomials, we can achieve a win-win scenario:

– If our assumptions hold, we obtain practically efficient candidates for space-hard cryptography
– While we do not provide a worst-to-average case reduction, refuting our assumptions would constitute a considerable advance in the algorithmic state-of-the art of polynomial factorization algorithms, as it is a long open problem to design polynomial factorization algorithms which leverage sparsity (in a non-extreme parameter regime).

*Space-Lock Puzzle.* Building Space-Lock Puzzles from the assumption that SRF is space-hard in sparse high-degree polynomials is fairly straight forward. To generate a puzzle for a random message $m$, generate a random sparse polynomial $f(X)$ with high degree and a random constant coefficient. Now, we know that

[1] More generally, we can consider this as finding roots of structured polynomials which possess a compact representation and can be evaluated quickly.

$f(X) - f(m)$ has a root at $m$ and can be output as a space-lock puzzle for $m$. There are two minor problems with this construction. First, we might want to create a puzzle for a non-random message, which we can resolve by using hybrid encryption. Second, there might be polynomials $f(X) - f(m)$ with multiple roots. We can fix this problem by padding the message and checking for the correct padding after solving the puzzle.

*Verifiable Space-Hard Functions.* We start by discussing our construction of verifiable space-hard functions from SRF. As we let go of the permutation requirement of the polynomials, we need to work harder to make this function verifiable. Our technical tool to achieve this is a novel and efficient special-purpose proof system for certifying the greatest common divisor between the polynomial $f(X)$ and $X^p - X$. This is sufficient, as given this gcd one can quickly and space-efficiently find the roots of $f(X)$.

For the purpose of this outline, assume that we have cheap way to prove equations over high-degree and possibly dense polynomials. We will later explain how to carry out these checks. We make use of the fact that the greatest common divisor between a polynomial $f(X)$ and $X^p - X$ is constant degree with high probability over the choice of a random sparse polynomial $f(X)$. Note that this gcd allows us to immediately compute the roots of $f(X)$ in $\mathbb{F}_p$. Our proof system establishes that some constant degree polynomial $g(X)$ is the gcd of $f(X)$ and $X^p - X$ in two phases. In the first phase, the prover computes $f'(X) = X^p - X \mod f(X)$ via square and multiply. Each step of this computation is defined by a simple polynomial equation $(X^{2^n} \mod f(X)) \cdot ((X^{2^n} \mod f(X)) = (X^{2^{n+1}} \mod f(X)) + h(X)f(X)$ for some polynomial $h(X)$.

In the second phase, we use that $gcd(X^p - X, f(X)) = gcd(f'(X), f(X))$ and compute $g(X) = gcd(f'(X), f(X))$ together with its Bézout coefficients $a(X)$, $b(X)$ via the extended Euclidean algorithm. The greatest common divisor is unique, up to normalization, hence we require the prover to normalize this polynomial. The verifier can easily check this property by checking that the leading coefficient is 1. Bézouts identity guarantees that for all $\bar{a}(X)$, $\bar{b}(X)$ we have $\bar{a}(X)f'(X) + \bar{b}(X)f(X)$ is a multiple of $gcd(f'(X), f(X))$. Further, the verifier can check whether $g(X) = a(X)f'(X)+b(X)f(X)$ is a divisor, by making sure that $f(X) \mod g(X) = 0$ and $X^p - X \mod g(X) = 0$. Now we have verified that $g(X)$ is a divisor of $f(X)$ and $f'(X)$ and that $g(X)$ is a multiple of the their greatest common divisor, therefore, it is a greatest common divisor.

So far we have skipped over the issue of how we can verify polynomial equations, when the polynomials have representations that are bigger than the verifier's space. Polynomial commitments such as [KZG10] would be the perfect tool for this but its common reference string scales with the degree of the polynomials, which we want to avoid. Instead we commit to evaluation of these polynomials at specific points, which in fact defines a Reed-Solomon code. We then use interactive oracle proofs of proximity to establish that the commitments are indeed close to a codeword of the corresponding Reed-Solomon code. To check the equations we pick a few random positions of the codeword and check whether the equations hold at these positions. We can then use the the Schwartz-Zippel lemma to argue soundness of the polynomial equations.

## 1.3 Open Problems

We consider it to be an interesting question to investigate whether it is possible to intrinsically and flexibly tie the resources of space and time in a puzzle or verifiable time-space hard function. That is, is it possible to force the puzzle solver to spend $S$ space for $T$ time? Here $S$ and $T$ are adjustable parameters. This concept may be most closely captured by the concept of sustained space complexity [ABP18].

We believe any solution to this problem that goes beyond taking a sequential function that has a scalable domain and generically applying incrementally verifiable computation to it might be of big interest. An example for such a function is sequential squaring over a modulus that scales with the space parameter. More specifically the function could be on input $x \in [2^\lambda]$ compute $h((N^s - x)^{2^T} \bmod N^s)$ where $h$ is some compressing function to reduce the size of the result.

## 1.4 Related Work

There are many works in memory restricted cryptography such as memory-hard functions and various forms of proof of space. Memory-hard functions are functions that are only computable with a large amount of memory accesses. The measure that many works use is called cumulative memory complexity. These functions are used to reduce the effectiveness of building application specific integrated circuits (ASICs) or field programmable gate arrays (FPGA) for brute force attacks because these excell at computation and do not have a faster way of accessing memory than off-the-shelf CPUs. Memory-hard functions are used in password hashing, proof of work, and other applications where the goal is to make computation expensive. The first memory-hard function was proposed by Percival in 2009 [Per09]. So far, all memory hard functions [Per09, AS15, AB16, ACP+17] use graphs with special structure and iteration of a function to force anyone trying to evaluate the function to do a lot of memory accesses.

Our functions are hard in a slightly different way. We are trying to increase the amount of maximum storage that is necessary to compute the function, which we call space hardness. A space-lock puzzle closely resemble trapdoor memory-hard function [AGP24], asymmetrically memory-hard functions [BP17] and memory-hard puzzles [ABB22], but under the notion of memory hardness.

We introduce space-hard functions and their verifiable counter part. Verifiable space-hard functions are a space-analogue of verifiable delay functions. We heavily deviate from the design space of memory-hard functions as most constructions are based on the random oracle model. Therefore, using incrementally verifiable computation to verify their evaluation requires proving statements over random oracle, which is concretely inefficient and conceptually unsatisfying. Indeed, [ABFG14] show how to verify that the function used much memory, but not that the output is correct. [DFKP15] further extend the notion to proof of space, where the prover executes a memory-hard function and then regularly gets queried to prove that he maintains a large amount of the computation in his memory. For an excellent overview on the topic, we refer to [RD16], as they detail the different notions and their relations.

Our verifiable space-hard function follows a similar design principle as the weak verifiable-delay function suggested by [BBBF18]. They suggest that inverting a fast to evaluate high degree permutation polynomial requires a lot of sequential computation. We instead conjecture and use the space-hardness of the same computation. Because we, however, want to move away from permutation polynomials to less structured polynomials our constructions require a special purpose proof system.

Indeed, verifiable space-hard functions can be though of as a space-analogue of verifiable delay functions [LW17,BBBF18,Wes19,Pie19,HHK+22,HHKK23] and space-lock puzzles as a space-analogue of time-lock puzzles [RSW96].

## 2 Preliminaries

### 2.1 Notation

We use the Landau notation to describe the asymptotic behavior of functions. We write $f(x) = O(g(x))$ if there exists a constant $c$ such that $|f(x)| \leq c|g(x)|$ for all $x$ larger than some constant $x_0$. Further we write $f(x) = o(g(x))$ if $\lim_{x\to\infty} f(x)/g(x) = 0$. We use the notation $[n]$ to denote the set $\{1, \ldots, n\}$. We use the notation $\mathsf{poly}$ to denote a polynomial. With $\mathsf{negl}$ we denote a negligible function, which is a function that is asymptotically smaller than the inverse of any polynomial. We use the notation $\lambda$ to denote the security parameter.

### 2.2 Finite Fields

For a prime-power $q = p^k$ we denote the finite field of size $q$ by $\mathbb{F}_q$. We call $p$ the characteristic of $\mathbb{F}_q$. Recall a few basic facts about finite fields. For a field $\mathbb{F}_q$ of characteristic the polynomial functions $x \mapsto x^{p^i}$ are called *Frobenius automorphisms* and it holds that $x^q = x^{p^k} = x$ for all $x \in \mathbb{F}_q$. Hence, the $q$ roots of the polynomial $X^q - X$ are precisely all the elements in $\mathbb{F}_q$. Consequently, if $x$ is in an extension field of $\mathbb{F}_q$ and $x^q - x = 0$, then it must hold that $x \in \mathbb{F}_q$. Likewise, the $q-1$ roots of the polynomial $X^{q-1} - 1$ are exactly all non-zero elements in $\mathbb{F}_q$.

### 2.3 Polynomials

We call the variable for polynomials $X$ and a polynomial is usually denoted like this $f(X)$ or in explicit form, e.g. $X + X^2$. We call a value $x$ a root of a polynomial $f(X)$ if $f(x) = 0$. The degree of a polynomial $f(X)$ is the highest power of $X$ that appears in $f(X)$. A polynomial is called monic if the coefficient of the highest power of $X$ is 1.

Recall that a univariate polynomial $f(X)$ is square-free if and only if it holds that $\gcd(f(X), f'(X)) = 1$, where $f'(X)$ is the formal derivative of $f(X)$ in $X$.

**Lemma 1 (Polynomial Identity Lemma).** *[Sch80, Zip79, DL78] Let $f(X)$ be a polynomial of degree $d$ over a field $\mathbb{F}_q$. Let $S \subseteq \mathbb{F}_q$ be a set of size $s$. Then for a random $x \leftarrow_\$ S$ we have $f(x) = 0$ with probability at most $d/s$.*

**Lemma 2 (Bézout's Identity for Polynomials).** *Let $f(X)$ and $g(X)$ of degree $d_1$ and $d_2$ be two polynomials with greatest common divisor $d(X)$. Then there exist polynomials $a(X)$ and $b(X)$ such that $a(X)f(X)+b(X)g(X) = d(X)$ and $\deg(a(X)) < d_2$ and $\deg(b(X)) < d_1$. Moreover, for all $\bar{a}(X)$, $\bar{b}(X)$ then polynomials of the form $\bar{a}(X)f(X)+\bar{b}(X)g(X)$ are exactly the multiples of $d(X)$.*

**Lemma 3 (Vandermonde Matrix Invertible).** *Over any field $\mathbb{F}$ the Vandermonde matrix*

$$\begin{pmatrix} 1 & x_1 & x_1^2 & \cdots & x_1^{n-1} \\ 1 & x_2 & x_2^2 & \cdots & x_2^{n-1} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_n & x_n^2 & \cdots & x_n^{n-1} \end{pmatrix}$$

*is invertible if the $x_i$ are distinct.*

## 2.4 Codes

A code $C$ is a set of codewords. A code is called linear if it is a vector space over a finite field $\mathbb{F}_q$. The distance of a code is the minimum Hamming distance between any two distinct codewords. The distance of a code is denoted by $\delta$. A code is called an $[n, k, \delta]$ code if it has length $n$, dimension $k$, distance $\delta$, and relative distance $\delta/n$. A Reed-Solomon code is a linear code that is defined by evaluating a polynomial at a set of points. The code is defined by a polynomial $f(X)$ of degree $k-1$ and a set of points $x_1, \ldots, x_n$. The codewords are then the evaluations of $f(X)$ at the points $x_1, \ldots, x_n$. The distance of a Reed-Solomon code is $n - k + 1$. We call the Reed-Solomon code evaluated at a set of points $\mathcal{L}$ with polynomials of degree $d$ $\mathsf{RS}[\mathcal{L}, d]$. The field this code is over will be clear from the context.

## 2.5 Combinatorics

$H$ is family of strongly universal$_t$ hash functions [WC81] that map from $X$ to $Y$ if for any distict $x_1, \ldots, x_t \in X$, and any possibly non-distinct $y_1, \ldots, y_t \in Y$, we have that

$$\mathsf{Pr}_{h \leftarrow_\$ H}[h(x_1) = y_1, \ldots, h(x_t) = y_t] = |Y|^{-t}.$$

We detail the first two Bonferroni inequalities.

**Lemma 4 (Bonferroni Inequalities).** *[Bon36] Let $A_1, \ldots, A_n$ be events. We have $\sum_{i \in [n]} \mathsf{Pr}[A_i] - \sum_{i \in [n], j \in [i-1]} \mathsf{Pr}[A_i \cap A_j] \leq \mathsf{Pr}[\cup_{i \in [n]} A_i] \leq \sum_{i \in [n]} \mathsf{Pr}[A_i]$.*

## 2.6 IOP

An interactive oracle proof (IOP) [BCS16] combines the powers of an interactive proof and a probabilistically checkable proof. It is a proof system where a prover and a verifier interactively exchange messages. The prover sends long strings that the verifier does not have to read entirely. The verifier only sends random challenges to the prover and checks the consistency of the responses. Formally:

**Definition 1 (IOP).** *An IOP with soundness error $\beta$ for a language $L$ with witness relation $R$ is defined by a tuple of algorithms $\mathsf{IOP} = (\mathsf{P}, \mathsf{V})$ defined with the following two properties:*

**Completeness** *For all $(x, w) \in R$:*

$$\Pr\left[\mathsf{V}^{\pi_1,\ldots,\pi_t}(x, \rho_1, \ldots, \rho_t) = 1 \;\middle|\; \begin{array}{c} \textit{Sample } \rho_1, \ldots, \rho_t \textit{ uniformly at random} \\ \pi_1 \leftarrow \mathsf{P}(x, w) \\ \pi_2 \leftarrow \mathsf{P}(x, w, \rho_1) \\ \vdots \\ \pi_t \leftarrow \mathsf{P}(x, w, \rho_1, \ldots, \rho_{t-1}) \end{array}\right]$$

*is 1.*

**Soundness** *For all $x \notin L$ and all unbounded provers $\mathsf{P}^*$:*

$$\Pr\left[\mathsf{V}^{\pi_1,\ldots,\pi_t}(x, \rho_1, \ldots, \rho_t) = 1 \;\middle|\; \begin{array}{c} \textit{Sample } \rho_1, \ldots, \rho_t \textit{ uniformly at random} \\ \pi_1 \leftarrow \mathsf{P}^*(x) \\ \pi_2 \leftarrow \mathsf{P}^*(x, \rho_1) \\ \vdots \\ \pi_t \leftarrow \mathsf{P}^*(x, w, \rho_1, \ldots, \rho_{t-1}) \end{array}\right]$$

*is $\leq \beta$*

**IOPP.** An interactive oracle proof of proximity (IOPP) is an IOP related notion where the prover is given a word of some code to which the verifier has query access. The prover then wants to convince the verifier that the word is close to a codeword of some code $C$. Formally:

**Definition 2 (IOPP).** *An IOPP for a code $C$ with distance $\delta$ and soundness error $\beta$ is defined by a tuple of algorithms $\mathsf{IOPP} = (\mathsf{P}, \mathsf{V})$ defined with the following two properties:*

**Completeness** *For all $c \in C$:*

$$\Pr\left[\mathsf{V}^{c,\pi_1,\ldots,\pi_t}(w, \rho_1, \ldots, \rho_t) = 1 \;\middle|\; \begin{array}{c} \textit{Sample } \rho_1, \ldots, \rho_t \textit{ uniformly at random} \\ \pi_1 \leftarrow \mathsf{P}(c) \\ \pi_2 \leftarrow \mathsf{P}(c, \rho_1) \\ \vdots \\ \pi_t \leftarrow \mathsf{P}(c, \rho_1, \ldots, \rho_{t-1}) \end{array}\right]$$

*is 1.*

**Soundness.** *For all $c$ is at distance $\geq \delta$ from $C$ and all unbounded provers* $\mathsf{P}^*$*:*

$$\Pr\left[\mathsf{V}^{c,\pi_1,\ldots,\pi_t}(w,\rho_1,\ldots,\rho_t)=1 \;\middle|\; \begin{array}{c} \textit{Sample } \rho_1,\ldots,\rho_t \textit{ uniformly at random} \\ \pi_1 \leftarrow \mathsf{P}^*(w) \\ \pi_2 \leftarrow \mathsf{P}^*(w,\rho_1) \\ \vdots \\ \pi_t \leftarrow \mathsf{P}^*(w,\rho_1,\ldots,\rho_{t-1}) \end{array}\right]$$

*is* $\leq \beta$*.*

There exist an IOPPs for Reed-Solomon codes [BBHR18, ACY23, ACFY24].

**Combiner.** Beyond IOPPs, we actually use the batch IOPPs as defined in Arnon et al. [ACFY24]. Specifically, we use a procedure $\mathsf{Combine}$ with the following properties. It takes as input a natural number $d^*$, some random field element $r$, $m$ functions $f_1,\ldots,f_m$ and corresponding degrees $d_1,\ldots,d_m$. If $f_i$ is a univariate polynomial of degree $d_i$ and $d_i \leq d^*$ for all $i \in [m]$ then $\mathsf{Combine}(d^*, r, (f_i, d_i)_{i\in[m]})$ is a degree $d^*$ univariate polynomial. Further if for some distance parameter $\delta$, rate $\rho$, and error $\mathsf{err}$ we have if

$$\Pr_{r\leftarrow_\$\mathbb{F}}[\varDelta(\mathsf{Combine}(d^*, r, (f_i)_{i\in[m]}), \mathsf{RS}[\mathcal{L}, d^*]) \leq \delta] > \mathsf{err}$$

then there exists $S \subset \mathcal{L}$ with $|S| \geq (1-\rho)\cdot|\mathcal{L}|$, and for all $i \in [m]$ exists a $u \in \mathsf{RS}[\mathcal{L}, d_i]$ with $f_i(S) = u(S)$. For guidance on how to choose the parameters we refer to [ACFY24].

## 3 Inverting the Guralnick–Müller Permutation Polynomial

In 1997, Guralnick and Müller [GM97] introduced a family of permutation polynomials that were later proposed as a candidate for building a verifiable delay function [BBBF18]. In this proposal it is crucial that inverting the polynomial is not easy. Guralnick–Müller polynomials do not seem as easy to invert as permutation polynomials from other families.

In this Section, we show – using the properties established by Guralnick and Müller – that, on the contrary, they can be easily inverted. As a consequence, they should not be used as a building block for verifiable delay functions.

We start by recalling the definition of the Guralnick–Müller polynomials and some of their relevant properties. We use the original notations of [GM97] rather the notations from [BBBF18].

### 3.1 Notations and Known Facts

Let $K$ be a finite field of characteristic $p$. A polynomial in $K[X]$ is called *exceptional* if it acts as a permutation on infinitely many finite extensions of $K$.

**Theorem 1 ((Part of) Theorem 1.4 of [GM97]).** *Let $K$ be a finite field of characteristic $p$. Let $q$ be a power of $p$. Given $\mu \in K$, set:*

$$a_\mu(X) = X^{2q} - 2\,\mu X^{q+1} + 2\,\mu X^q + \mu^2 X^2 + 2\,\mu^2 X + \mu^2.$$

*Define:*

$$f_\mu(X) = \frac{a_\mu(X)^{(q+1)/2} + (X^q - \mu X + \mu)^q (X^q - \mu X - \mu)}{2\,X^q}.$$

*Then $f_\mu$ is exceptional over $K$ provided that $\mu$ is not a $(q-1)$-st power in $K\,\mathbb{F}_q$.*

*Remark 1.* At first glance, $f_\mu(X)$ is defined as a rational fraction. However, its numerator is divisible by $X^q$, so $f_\mu(X)$ is a polynomial of degree $q^2$.

We use the following property of $f_\mu$.

**Proposition 1 (Proposition 3.4 of [GM97]).** *Let $t$ be a variable and set $Y = X^q - \mu X$ and $\delta = (t^2 - 4\,\mu^{2q+1})^{(q-1)/2}$. Let $\theta$ denote a $(q-1)$-st root of $\mu$ in $\bar{K}$. Then*

$$\prod_{i \in \mathbb{F}_q} (f(X + i\theta) - t) = H(Y),$$

*where $H(Y) = Y^{q^2} + A_2 Y^q + A_1 Y + A_0$. The coefficients $A_0, A_1, A_2 \in K$ are given by*

$$\begin{aligned} A_2 &= \frac{t^q - \delta t}{2\,\mu^q}, \\ A_1 &= -\mu^q \delta, \\ A_0 &= -\frac{t^q + (t - 2\,\mu^{q+1})\delta}{2} + \mu^{q^2}. \end{aligned}$$

## 3.2 Inversion in the Context of VDFs

In the setting of verifiable delay functions, the authors of [BBBF18] consider the use of the Guralnick–Müller on a fixed field where it acts as a permutation. More precisely, they take $q = p^r$ and consider the family $f_\mu$ on $\mathbb{F}_{p^n}$. Without loss of generality we may assume that $r$ and $n$ are co-prime. If they are not let $g$ denote their greatest common divisor. Then both $q$ and $p^n$ are powers of $p^g$, with coprime exponents. In the sequel, we change $r$ and $n$ and consider the following case:

$$q_0 = p^g, \quad q = q_0^r, \quad \text{and } Q = q_o^n.$$

Assume now that we want to solve the equation $f_\mu(x) = t$ for some given target $t$ in $\mathbb{F}_Q$ and some solution $x$ also in $\mathbb{F}_Q$. As a direct consequence of Proposition 1, $x$ has to be a root of:

$$H(X^q - \mu X) = X^{q^3} + B_2\,X^{q^2} + B_1\,X^q + B_0\,X + A_0, \tag{3}$$

with:

$$B_2 = A_2 - \mu^{q^2}, \quad B_1 = A_1 - \mu^q A_2, \quad \text{and } B_0 = -\mu A_1.$$

Since Eq. (3) is affine, it is easy to solve using standard techniques. More precisely, we use the action of Frobenius to amplify this equation in an affine system of $n$ equations in $n$ variables. First, we define each variable $X_i$ as $X^{q_0^i}$. Since we look for a solution in $\mathbb{F}_Q$, we have the constraint $X_n = X_0$ and similarly $X_i = X_{i \bmod n}$ for any $i$. Let $d$ be an integer modulo $n$, by raising Eq. (3) to the power $q_0^d$ we obtain (by linearity of Frobenius) that:

$$X_{d+3r} + B_2^{q_0^d} X_{d+2r} + B_1^{q_0^d} X_{d+r} + B_0^{q_0^d} X_d = -A_0^{q_0^d}.$$

Taken these $n$ equations together, we obtain a linear system and can recover the desired value $x$ as the value of $X_0$ in the solution.

## 3.3 Efficiency of the Algorithm

The hardness assumption in [BBBF18] assumes that inverting the Guralnick–Müller polynomial costs at least $q^2$ operations in $\mathbb{F}_Q$. Our algorithm falsifies this assumption. Indeed, its most time consuming part is the resolution of the linear system. Using Gaussian elimination, it can be achieved in $n^3$ arithmetic operation over $\mathbb{F}_Q$. Thus, its bit complexity is $\tilde{O}(n^4 \log q_0)$.

Note that a more careful implementation can instead solve a linear system of the same dimension over $\mathbb{F}_{q_0}$, thus reducing the bit complexity to $\tilde{O}(n^3 \log q_0)$.

It would be tempting to consider the use of faster linear algebra in this algorithm. However, this is not really useful, since there exists a faster algorithm that uses only $O(n)$ arithmetic operations in $\mathbb{F}_Q$ to invert the Guralnick–Müller polynomial.

## 3.4 Correctness of the Algorithm

In order to establish the correctness of our algorithm, we will need the following additional lemma which describes the result of the $\mathbb{F}_Q$-Frobenius action on $\theta$, which is a $q-1$-st root of $\mu$.

**Lemma 5.** *Let $\mu \in K$ be as in Theorem 1 and let $\theta$ be a $q-1$-st root of $\mu$. Let $|\mathbb{F}_Q| = |\mathbb{F}_q \cdot K| = q^n$. Then it holds that*

$$\theta^{q^n} = \mu^{\frac{q^n-1}{q-1}} \theta.$$

*Furthermore, given that $\theta \notin \mathbb{F}_q \cdot K$, it holds that $1 \neq \mu^{\frac{q^n-1}{q-1}} \in \mathbb{F}_q$.*

*Proof.* First recall that $\frac{q^n-1}{q-1} = \sum_{i=0}^{n-1} q^i$ is indeed an integer. Furthermore, since $\theta$ is a $q-1$-st root of $\mu$ it holds that

$$\theta^q = \theta^{q-1} \cdot \theta = \mu \cdot \theta.$$

Hence, by iterating this identity we obtain via induction that

$$\begin{aligned}\theta^{q^n} &= (\theta^q)^{q^{n-1}} \\ &= (\mu \cdot \theta)^{q^{n-1}} \\ &= \mu^{q^{n-1}} \cdot \theta^{q^{n-1}} \\ &= \mu^{q^{n-1}} \cdot \mu^{\frac{q^{n-1}-1}{q-1}} \cdot \theta \\ &= \mu^{q^{n-1}} \cdot \mu^{\sum_{i=0}^{n-2} q^i} \cdot \theta \\ &= \mu^{\sum_{i=0}^{n-1} q^i} \cdot \theta \\ &= \mu^{\frac{q^n-1}{q-1}} \cdot \theta,\end{aligned}$$

where the inductive step happens from the third to the fourth equality. For the second part of the statement, observe that as $\mu \in K \subseteq \mathbb{F}_q \cdot K$ it holds that

$$(\mu^{\frac{q^n-1}{q-1}})^{q-1} = \mu^{q^n-1} = 1,$$

i.e. $\mu^{\frac{q^n-1}{q-1}} \in \mathbb{F}_q$. Finally note that $\mu^{\frac{q^n-1}{q-1}} = 1$ would imply that

$$\theta^{q^n} - \theta = 0,$$

i.e. $\theta \in \mathbb{F}_q \cdot K$. But this immediately contradicts the assumption that $\theta \notin \mathbb{F}_q \cdot K$.

The following theorem establishes that our algorithm always returns the correct solution given that the coefficient $A_1 = -\mu^q \delta$ is non-zero. Note that $\delta = (t^2 - 4\,\mu^{2q+1})^{(q-1)/2}$ is non-zero whenever $t \neq \pm 2\,(\sqrt{\mu})^{2q+1}$. If $\sqrt{\mu} \notin K$ this event never happens, otherwise if $\sqrt{\mu} \in K$ the event that $t = \pm 2\,(\sqrt{\mu})^{2q+1}$ happens with negligible probability $2/|K|$, given that $t$ is distributed uniformly on $K$. Note further that in the [BBBF18] scheme the input $t$ is chosen uniformly random.

**Theorem 2.** *Let $H(Y) = Y^{q^2} + A_2 Y^q + A_1 Y + A_0$ as in Proposition 1. Given that the coefficient $A_1$ of $H(Y)$ is non-zero, the equation $H(X^q - \mu X) = 0$ has a unique solution in $K$.*

*Proof.* First off, note that since $f$ is permutation on $K$, by Proposition 1 the system $H(X^q - \mu X) = 0$ has at least one solution. Observe first that given that $A_1 \neq 0$, the polynomial $H(X^q - \mu X)$ is square free, i.e. over its algebraic closure it splits into distinct linear factors. This holds as its formal derivative is $-\mu A_1$, which is non-zero given that $A_1 \neq 0$.

Assume towards contradiction that $H(X^q - \mu X) = 0$ has two distinct solutions $x \neq x'$ in $K$. Thus, by Proposition 1 there exist $i, i' \in \mathbb{F}_q$ such that $x + i\theta$ and $x' + i'\theta$ are both roots of $f(X) - t$, i.e.

$$\begin{aligned} f(x + i\theta) - t &= 0 \\ f(x' + i'\theta) - t &= 0.\end{aligned}$$

Observe that not both $i$ and $i'$ can be 0, as otherwise $x$ and $x'$ would be two distinct roots of $f(X) - t$ in $K$, which contradicts the permutation property of $f$. Without loss of generality, assume that $i \neq 0$ and set $z = x + i\theta$. We claim that $x + i\mu^{\frac{q^n-1}{q-1}}\theta$ is also a root of $f(X) - t$, as

$$\begin{aligned} 0 &= (f(z) - t)^{q^n} \\ &= f((x + i\theta)^{q^n}) - t \\ &= f(x + i\theta^{q^n}) - t \\ &= f(x + i\mu^{\frac{q^n-1}{q-1}}\theta) - t. \end{aligned}$$

Here the second equality follows as $f(X)-t$ is in $K[X]$, the third equality follows as $x, i \in \mathbb{F}_q \cdot K$, and the last equality follows by the first item of Lemma 5.

Now set $i^* = i \cdot (\mu^{\frac{q^n-1}{q-1}} - 1)$ which is a non-zero element of $\mathbb{F}_q$ by Lemma 5. We claim that $z$ is also a root of $f(X + i^*\theta)$, as

$$f(z + i^*\theta) = f(x + i\theta + i \cdot (\mu^{\frac{q^n-1}{q-1}} - 1)\theta) = f(x + i\mu^{\frac{q^n-1}{q-1}}\theta) = 0.$$

Consequently, $z$ is a zero of both $f(X) - t$ and $f(X + i^*\theta) - t$, and therefore $X - z$ divides both $f(X) - t$ and $f(X + i^*\theta) - t$, and therefore, as $i^* \neq 0$ it holds that $(X - z)^2$ divides $\prod_{j \in \mathbb{F}_q}(f(X + j\theta) - t) = H(X^q - \mu X)$ (by Proposition 1). But this contradicts the square-freeness of $H(X^q - \mu X)$, which concludes the proof.

### 3.5 Implementation

We provide Magma code for the attack in Appendix A.

## 4 Space-Hardness of Root-Finding

We conjecture that root-finding for polynomials over a big finite field requires a lot of space. As far as we are aware of, all root finding algorithms [Ber70, CZ81] [VZGS92, Sho93, KS95, KU11, GNU16] in a finite field $\mathbb{F}_p$ for large $p$ and comparatively smaller degree $d$, start by computing $X^p - X \mod f(X)$. For a discussion of recent results, see [GNU16]. In general, this polynomial $X^p - X \mod f(X)$ is a dense polynomial of degree $d - 1$, whose representation requires $d$ elements in $\mathbb{F}_p$. For this reason, we conjecture a minimal space of $d$ for any algorithm with a runtime $o(p)$. Proving this conjecture wrong would greatly advance the state of the art concerning polynomial factorization.

**Assumption 1 (Sparse Root-Finding (SRF)).** *We define the space-hardness of finding a root in a polynomial from distribution $\mathcal{D}_{\lambda,S}$ as follows: Root-Finding is hard with a gap $\varepsilon < 1$ if there exists a polynomial $\tilde{S}(\cdot)$ such that* *for all polynomials $S(\cdot) \geq \tilde{S}(\cdot)$ and PPT adversaries $\{\mathcal{A}_\lambda\}_{\lambda \in \mathbb{N}}$ with space bound $S^\varepsilon(\lambda)$ there exists a negligible function* negl *such that for all $\lambda \in \mathbb{N}$:*

$$\Pr\left[f(x^*) = 0 \,\middle|\, \begin{array}{l} f(X) \leftarrow \mathcal{D}_{\lambda,S(\lambda)} \\ x^* \leftarrow \mathcal{A}_\lambda(f(X)) \end{array}\right] = \mathsf{negl}(\lambda)$$

We also define a space-hardness assumption for the problem of computing the greatest common divisor of a polynomial and $X^p - X$.

**Assumption 2 (Sparse GCD Computation).** *We define the space-hardness of gcd computation from distribution $\mathcal{D}_{\lambda,S}$ as follows: gcd computation is hard with a gap $\varepsilon < 1$ if there exists a polynomial $\tilde{S}(\cdot)$ such that for all polynomials $S(\cdot) \geq \tilde{S}(\cdot)$ and PPT adversaries $\{\mathcal{A}_\lambda\}_{\lambda \in \mathbb{N}}$ with space bound $S^\varepsilon(\lambda)$ there exists a negligible function* negl *such that for all $\lambda \in \mathbb{N}$:*

$$\Pr\left[g(X) = gcd(f(X), X^p - X) \,\middle|\, \begin{array}{c} f(X) \leftarrow \mathcal{D}_{\lambda,S(\lambda)} \\ g(X) \leftarrow \mathcal{A}_\lambda(f(X)) \end{array}\right] = \mathsf{negl}(\lambda)$$

**Lemma 6.** *If root-finding is hard with gap $\varepsilon < 1$ then gcd computation is hard with gap $\varepsilon < 1$.*

*Proof.* Given an adversary $\mathcal{A}$ that breaks Assumption 2 we construct an adversary $\mathcal{A}'$ that breaks Assumption 1.

$\mathcal{A}'(f(X))$:
- Let $g(X) \leftarrow \mathcal{A}(f(X))$ where $g(X)$ is an $n$-degree polynomial.
- Factor $g(X)$ into degree 1 polynomials $X - h_1, \ldots, X - h_n$ using the Cantor-Zassenhaus algorithm.
- Return $h_1$.

The factors of $g(X)$ can only be of degree 1 because $X^p - X$ only factors of degree 1. The Cantor-Zassenhaus [CZ81] algorithm has space-complexity $O(n \log p)$ [Sho93] and runs in $\mathsf{poly}(n, \log p)$.

We require these assumptions for two different but related applications, which we will detail in later chapters. One is a space lock puzzle and the other a verifiable space-hard function. For the space lock puzzle we only really require that the polynomials by the distributions are fast to evaluate and that root-finding is space-hard.

In general, we will stick with prime order fields because they tend to have less structure, which might protect them against structural attacks. We will also try to impose as little structure as possible on the polynomials. In order to make proofs over the these fields better we choose FFT-friendly fields.

A natural candidate is the distribution of random sparse polynomials. We make sure that the lowest two monomials are random for better estimation of number of roots. More formally, we define the distribution $\mathcal{D}_{\lambda,S}$ as follows:

**Definition 3 (Random Sparse Polynomial (with Uniform Constant and Linear Coefficient)).** *Pick a prime $p \in \Omega(2^\lambda)$, degree $d \in \Omega(S)$, and* *number of non-zero monomials $k \in \Omega(\lambda)$. Operations happen over $\mathbb{F}_p$. Output the following univariate (the variable is $X$) polynomial*

$$a_0 + a_1X + \sum_{i\in[k-2]} a_iX^{e_i} + X^d$$

*For uniformly random $a_i \leftarrow_\$ \mathbb{F}_q$ and $e_i \leftarrow_\$ [d-1]$.*

We prove these polynomials define a family of strongly universal$_2$ hash functions [WC81].

**Lemma 7.** *For any polynomial $h(X)$, any distict $x_1,\ldots,x_t \in \mathbb{F}_p$, and any possibly non-distinct $y_1,\ldots,y_t \in \mathbb{F}_p$, we have that*

$$\mathsf{Pr}_{a_0,\ldots,a_{t-1}\leftarrow_\$\mathbb{F}_p}[h(x_1)+l(x_1)=y_1,\ldots,h(x_t)+l(x_t)=y_t] = p^{-t}$$

*where $l(X) = \sum_{i\in[t]} a_{i-1}X^{i-1}$.*

*Proof.* The statment $h(x_1)+l(x_1)=y_1,\ldots,h(x_t)+l(x_t)=y_t$ is equivalent to the this linear system of equations:

$$\begin{pmatrix}1 & x_1 & \ldots & x_1^{t-1} & h(x_1)\\ \vdots & \vdots & \vdots & \vdots & \vdots\\ 1 & x_t & \ldots & x_t^{t-1} & h(x_t)\end{pmatrix}\begin{pmatrix}a_0\\ \vdots\\ a_{t-1}\\ 1\end{pmatrix} = \begin{pmatrix}y_1\\ \vdots\\ y_t\end{pmatrix} \qquad \Leftrightarrow$$

$$\begin{pmatrix}1 & x_1 & \ldots & x_1^{t-1}\\ \vdots & \vdots & \vdots & \vdots\\ 1 & x_t & \ldots & x_t^{t-1}\end{pmatrix}\begin{pmatrix}a_0\\ \vdots\\ a_{t-1}\end{pmatrix} = \begin{pmatrix}y_1 - x_1^d\\ \vdots\\ y_k - x_k^d\end{pmatrix}$$

The matrix is a Vandermonde matrix, which is invertible. Therefore, multiplication by it is a bijection. Because $a_0,\ldots,a_{t-1}$ are uniformly random, the probability of the system of equations to hold is $p^{-t}$.

Via a inclusion-exclusion argument, we can upper bound the probability of the polynomial having no roots.

**Lemma 8.** *For any polynomial $h(X)$ and uniform $a_0, a_1 \leftarrow_\$ \mathbb{F}_p$ we have $a_0 + a_1X + h(X)$ no roots with probability $\leq 1/2$.*

*Proof.* We have

$$\begin{aligned}&\mathsf{Pr}_{a_0,a_1}[\vee_{x\in\mathbb{F}_p}(a_0+a_1x+h(x)=0)]\\ \geq&\sum_{x\in\mathbb{F}_p}\mathsf{Pr}_{a_0,a_1}[a_0+a_1x+h(x)=0] \qquad (4)\\ &-\sum_{x_1<x_2\in\mathbb{F}_p}\mathsf{Pr}_{a_0,a_1}[\wedge_{x\in\{x_1,x_2\}}(a_0+a_1x+h(x)=0)]\\ =&1-\sum_{x_1<x_2\in\mathbb{F}_p}1/p^2 \qquad (5)\\ \geq&1/2\end{aligned}$$

Inequality 4 follows from a Bonferroni inequality 4 and equality 5 follows from Lemma 7.

We need the following basic fact.

**Lemma 9.** *The size of the image of a polynomial $h(X)$ corresponds to the number of $a_0 \in \mathbb{F}_p$ for which the polynomial $h(X) - a_0$ has a root in $\mathbb{F}_p$.*

*Proof.* If there exists an $x \in \mathbb{F}_p$ such that $h(x) = a_0$ then $h(X) - a_0$ has a root in $\mathbb{F}_p$ and vice versa.

**Lemma 10.** *For any polynomial $h(X)$ and uniform $a_1 \leftarrow_\$ \mathbb{F}_p$ it holds with probability $> 1 - 1/\sqrt{2}$ that $h(X) + a_1 X$ has a image of size $> p(1 - 1/\sqrt{2})$.*

*Proof.* Fix a polynomial $h(X)$. By Lemma 8 we know that the number of $a_0, a_1 \in \mathbb{F}_p$ for which the polynomial $f(X) = a_0 + a_1 X + h(X)$ has no root is at most $p^2/2$. Therefore, by a pidgeon-hole argument there are $< p/\sqrt{2}$ choices of for $a_1$ such that there exist $> p/\sqrt{2}$ choices of $a_0$ such that $a_0 + a_1 X + h(X)$ has no root. This means that there are $> p - p/\sqrt{2}$ choices for $a_1$ such that $\leq p/\sqrt{2}$ choices for $a_0$ such that $a_0 + a_1 X + h(X)$ has no root. Therefore, there are $> p - p/\sqrt{2}$ choices for $a_1$ such that $> p - p/\sqrt{2}$ choices for $a_0$ such that $a_0 + a_1 X + h(X)$ has a root. The statement follows by Lemma 9.

**Lemma 11.** *Fix a polynomial $h(X)$. The statistical distance between $(a_1, h(x) + a_1 x)$ and $(a_1, y)$ where $a_1, x$ is uniformly random over $\mathbb{F}_p$ and $y$ is uniformly random from the image of $h(X) + a_1 X$ is $> 1.5 - 2/\sqrt{2}$.*

*Proof.* By Lemma 10 $h(X) + a_1 X$ has a image of size $> p(1 - 1/\sqrt{2})$ with probability $1 - 1/\sqrt{2}$. For the rest of this analysis we assume to be in this case.

Picking a random $x$ and evaluating $h(X) + a_1 X$ on it is the same as sampling the output uniformly random from the multiset image of $h(X) + a_1 X$ (the multiset where each element $y$ has the multiplicity of the number of elements such that the element evaluates to $y$). This multiset has size $p$.

We now convert the multiset image to set by enumerating the multiplicities of each element and call this set $\mathcal{M}$. E.g. a multiset $\{8, 8, 8, 13, 55, 55\}$ would turn into a set of tuples $\{(8, 1), (8, 2), (8, 3), (13, 1), (55, 1), (55, 2)\}$. We do the same thing to the image of $h(X) + a_1 X$ and call it $\mathcal{D}$. However because it is a set it only ever has multiplicity one. So we would turn the set $\{8, 13, 55\}$ into $\{(8, 1), (13, 1), (55, 1)\}$. By the definition of these sets $\mathcal{D} \subseteq \mathcal{M}$. Because $\mathcal{M}$ is of size $p$ and $\mathcal{D}$ is of size $> p(1 - 1/\sqrt{2})$ sampling a random element from $\mathcal{M}$ will be in $\mathcal{D}$ with probability $> 1 - 1/\sqrt{2}$. Therefore, for a $1 - 1/\sqrt{2}$ fraction of random choices in sampling $x$ at random and then evaluating $h(x) + a_1 x$ behaves exactly as sampling uniformly random from the image. Thus, the statistical distance is $> (1 - 1/\sqrt{2})^2 = 1.5 - 2/\sqrt{2}$.

For our verifiable space-hard functions, we also want to have tight bound on the number of roots of these polynomials. A natural candidate for this is a distribution over permutation polynomials. In previous chapters we showed how to efficiently invert the specific set of Guralnick-Müller permutation polynomials [GM97], which [BBBF18] suggested as a time-lock puzzle. As we leveraged the specific structure of these polynomials for our attack, we believe it to be prudent to stay away from permutation polynomials.

If we instead use random sparse polynomials in our verfiable space-hard functions, we would want to have a good bound on the number of roots of these polynomials. The work of [Kel16] conjectures that the number of roots in a random sparse polynomial is $O(k \log p)$, which would be good enough for as this also implies that the probability of sampling a polynomial without roots is not overwhelming.

To have a lower probability of sampling a polynomial without roots that we can even prove, we suggest the distribution of polynomials which are the sum of a dense low-degree polynomial and a single high-degree monomial. We define the distribution $\mathcal{D}_{\lambda,S}$ as follows:

**Definition 4 (Low-Degree Dense).** *Pick a prime $p \in \Omega(2^\lambda)$, degree $d \in \Omega(S)$, and number of non-zero monomials $k$ such that $k \log k - 2k \geq \lambda$. Operations happen over $\mathbb{F}_p$.*

$$a_0 + \sum_{i \in [k-1]} a_i X^i + X^d$$

*For uniformly random $a_i \leftarrow_\$ \mathbb{F}_q$.*

**Lemma 12.** *For a polynomial $f(X)$ sampled from distribution $\mathcal{D}_{\lambda,S}$ we have $f(X)$ has at least $k$ (as in Definition 4) roots with probability $\leq 2^{-\lambda}$.*

*Proof.* A polynomial having $k$ roots equivalent to the statement that there exists a set of $\lambda$ distinct points $x_1, \ldots, x_k \in \mathbb{F}_p$ such that the polynomial evaluates to zero at these points. There are $\binom{p}{k}$ much sets. For each of those sets, the probability that the polynomial evaluates to zero at these points is $p^{-k}$ by Lemma 7. Therefore, by union bound the probability that the polynomial evaluates to zero at any set of $k$ distinct points is $\leq \frac{\binom{p}{k}}{p^k} \leq \frac{p(p-1)\cdots(p-k+1)}{k! \cdot p^k} \leq \frac{1}{k!} \leq 2^{-\lambda}$. The last inequality follows from Stirling's approximation and our choice of $k$ relative to $\lambda$.

## 5 Space-Lock Puzzle from SRF

We define space-lock puzzles analogously to time-lock puzzles but the resource we restrict is not sequential time but space.

**Definition 5 (Space-Lock Puzzle).** *A space-lock puzzle (SLP) with message space $\{0,1\}^n$ is a tuple of three algorithms* $\mathsf{SLP} = (\mathsf{Setup}, \mathsf{Gen}, \mathsf{Solve})$ *defined as follows:*

$\mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda, S)$: *The setup algorithm* $\mathsf{Setup}$ *takes as input a security parameter $1^\lambda$ and a space bound $S$ and outputs public parameters* $\mathsf{pp}$.

$\mathsf{p} \leftarrow \mathsf{Gen}(\mathsf{pp}, \mathsf{m})$*: The puzzle generation algorithm* Gen *takes as input public parameters* pp *and a message* m *and outputs a puzzle* p.

$\mathsf{m} \leftarrow \mathsf{Solve}(\mathsf{pp}, \mathsf{p})$*: The solving algorithm* Solve *takes as input a puzzle* p *and outputs a message* m.

**Statistical Correctness***:* $\mathsf{SLP} = (\mathsf{Setup}, \mathsf{Gen}, \mathsf{Solve})$ *is statistically correct if for all polynomials* $S(\cdot)$ *there exists a negligible function* negl *s.t. for all* $n, \lambda \in \mathbb{N}$ *and* $\mathsf{m} \in \{0,1\}^n$ *:*

$$\Pr\left[\mathsf{m} \neq \mathsf{Solve}(\mathsf{pp}, \mathsf{p}) \;\middle|\; \begin{array}{c} \mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda, S(\lambda)) \\ \mathsf{p} \leftarrow \mathsf{Gen}(\mathsf{pp}, \mathsf{m}) \end{array}\right] \leq \mathsf{negl}(\lambda)$$

**Efficiency***: There exists a polynomial* poly *such that for all* $n, \lambda, S \in \mathbb{N}$*, and* $\mathsf{m} \in \{0,1\}^n$ *the runtime (and therefore space usage) of* $\mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda, S)$ *and* $\mathsf{p} \leftarrow \mathsf{Gen}(\mathsf{pp}, \mathsf{m})$ *is* $\leq \mathsf{poly}(\lambda, n, \log S)$.

**Security***:* SLP *is secure with gap* $\varepsilon < 1$ *if there exists a polynomial* $\tilde{S}(\cdot)$ *such that all polynomials* $S(\cdot) \geq \tilde{S}(\cdot)$ *and PPT adversaries* $\{\mathcal{A}_\lambda\}_{\lambda \in \mathbb{N}}$ *with space bound* $S^\varepsilon(\lambda)$ *there exists a negligible function* negl *s.t. for all* $n, \lambda \in \mathbb{N}$*:*

$$\Pr\left[b = \mathcal{A}_\lambda(\mathsf{st}, \mathsf{p}) \;\middle|\; \begin{array}{c} \mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda, S(\lambda)) \\ (\mathsf{m}_0, \mathsf{m}_1, \mathsf{st}) \leftarrow \mathcal{A}_\lambda(\mathsf{pp}) \\ b \leftarrow_\$ \{0,1\} \\ \mathsf{p} \leftarrow \mathsf{Gen}(\mathsf{pp}, m_b) \end{array}\right] \leq 1/2 + \mathsf{negl}(\lambda)$$

We present the first space-lock puzzle based on the space hardness of finding roots of polynomials.

**Construction 1 (Space-Lock Puzzle).** *Let* $\mathcal{D}_{\lambda,S}$ *be a distribution of polynomial that are fast to evaluate and where finding roots is space-hard. Further,* $\mathsf{H} : \mathbb{F}_q \to \{0,1\}^{\lambda+n}$*, where* $n$ *is the size of the message space then the following defines a space-lock puzzle.*

```
Setup(1^λ, S): Return (λ, S).
Gen(pp, m):
    – Sample a degree $S$ polynomial $f(X) \in \mathcal{D}_{\lambda,S}$.
    – Sample a uniformly random element $z \leftarrow_\$ \mathbb{F}_q$.
    – Let $y = f(z)$.
    – Return $(f(X), y, \mathsf{H}(z) \oplus (0^\lambda || m))$.
Solve(p = (f(x), y, c)):
    – Compute $Z$, the set of roots of the polynomial $f(X) - y$.
    – For $z^* \in Z$:
        • Compute $\tilde{m} \leftarrow \mathsf{H}(z^*) \oplus c$.
        • If the first $\lambda$ bits of $\tilde{m}$ are all 0 return the rest of $\tilde{m}$
```

**Theorem 3.** *Construction 1 is a space-lock puzzle under the assumption that the SRF assumption 1 holds for polynomials with uniform linear coefficient.*

*Proof.* The proof follows from lemmas 13 to 15.

**Lemma 13 (Statistical Correctness).** *Construction 1 is statistically correct.*

*Proof.* Because $f(X) - y$ has degree $S$ it has at most $S$ roots. We know that $f(z) = y$, therefore, $z$ is a root of $f(X) - y$. Because $\mathsf{H}$ is a random oracle we have that for all $z^* \in Z \setminus \{z\}$ the probability that the first $\lambda$ many bits of $\tilde{m}$ are $0^\lambda$ is $2^{-\lambda}$. Therefore, $\mathsf{Solve}$ outputs $m$ with all but probability $\leq S(\lambda) \cdot 2^{-\lambda}$, which is negligible in $\lambda$.

**Lemma 14 (Efficiency).** *Construction 1 is efficient.*

*Proof.* The properties of $\mathcal{D}_{\lambda,S}$ tell us that evaluating $f(X)$ on $z$ can be done in $\mathsf{poly}(\lambda, \log S)$. It follows that $\mathsf{Gen}$ runs in time and space $\mathsf{poly}(\lambda, \log S)$.

**Lemma 15 (Security).** *Construction 1 is secure under the SRF assumption 1 for polynomials with uniform linear coefficient.*

*Proof.* To break security of the Construction 1 the adversary has to compute $z$ given $f(X)$ and $y$ otherwise he has no way of computing $\mathsf{H}(z)$.

$y$ is computed by evaluating a $f(X)$ at a uniform position. By Lemma 12 this is at constant $c$ statistical distance from sampling $y$ uniformly at random from the image of $f(X)$. This is the same as sampling $f(X)$ and $y$ uniformly at random under the condition that $f(X) - y$ has a root. Lemma 8 tells us that polynomial with uniform constant coefficient have no root with probability $\leq 1/2$. Therefore, any adversary that breaks the security of Construction 1 with probability $\epsilon$ breaks the SRF assumption with probability $c\epsilon/2$.

## 6 Verifiable Space-Hard Function from SRF

The definition of verifiable space hard function is similar to the definition of verifiable delay function but instead of a sequential time bound we have a space bound.

**Definition 6 (Verifiable Space-Hard Function).** *A verifiable space-hard function (VSHF) with domain space $\mathbb{X}$, codomain $\mathbb{Y}$, and proof space $\Pi$ has the following algorithms:*

$\mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda, S)$*: The setup algorithm* $\mathsf{Setup}$ *takes as input a security parameter $\lambda$ and a space bound $S$ and outputs public parameters* $\mathsf{pp}$.

$(y, \pi) \leftarrow \mathsf{Eval}(\mathsf{pp}, x)$*: The evaluation deterministic algorithm* $\mathsf{Eval}$ *takes as input public parameters* $\mathsf{pp}$ *and outputs a function output $y$ and a proof $\pi \in \Pi$.*

$b \leftarrow \mathsf{Verify}(\mathsf{pp}, x, y, \pi)$*: The verification algorithm* $\mathsf{Verify}$ *takes as input public parameters* $\mathsf{pp}$*, a function input $x \in \mathbb{X}$, a function output $y$ and a proof $\pi \in \Pi$ and outputs a bit $b$.*

*it has the following properties:*

**Correctness***:* $\mathsf{VSHF} = (\mathsf{Setup}, \mathsf{Eval}, \mathsf{Verify})$ *is correct if for all $\lambda, S \in \mathbb{N}$ and $x \in \mathbb{X}$ we have* $\mathsf{Verify}(\mathsf{pp}, x, y, \pi) = 1$ *for* $\mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda, S)$ *and* $(y, \pi) \leftarrow \mathsf{Eval}(\mathsf{pp}, x)$.

**Space Hardness**: $\mathsf{VSHF} = (\mathsf{Setup}, \mathsf{Eval}, \mathsf{Verify})$ *is sound with entropy $E$ and a gap $\varepsilon < 1$ if there exists a polynomial $\tilde{S}(\cdot)$ such that for all polynomials $S(\cdot) \geq \tilde{S}(\cdot)$ and PPT adversaries $\{\mathcal{A}_\lambda\}_{\lambda \in \mathbb{N}}$ with space bound $S^\varepsilon(\lambda)$ there exists a negligible function* $\mathsf{negl}$ *such that for all $\lambda \in \mathbb{N}$, $x \in \mathbb{X}$:*

$$\Pr\left[\mathsf{Eval}(\mathsf{pp}, x) = (y, \pi) \,\middle|\, \begin{array}{c}\mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda, S(\lambda)) \\ (x, y) \leftarrow \mathcal{A}_\lambda(\mathsf{pp})\end{array}\right] \leq 2^{-E} + \mathsf{negl}(\lambda)$$

**Computational Uniqueness**: $\mathsf{VSHF} = (\mathsf{Setup}, \mathsf{Eval}, \mathsf{Verify})$ *is computationally unique if for all PPT adversaries $\mathcal{A}$ there exists a negligible function* $\mathsf{negl}$ *s.t. for all $\lambda \in \mathbb{N}$ and $x \in \mathbb{X}$ :*

$$\Pr\left[\mathsf{Verify}(\mathsf{pp}, x, y^*, \pi^*) = 1 \wedge y^* \neq y \,\middle|\, \begin{array}{c}\mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda, S(\lambda)) \\ (x, y^*, \pi^*) \leftarrow \mathcal{A}_\lambda(\mathsf{pp}) \\ (y, \pi) \leftarrow \mathsf{Eval}(\mathsf{pp}, x)\end{array}\right] \leq \mathsf{negl}(\lambda)$$

**Efficiency**: *There exists a polynomial* $\mathsf{poly}$ *such that for all $\lambda, S \in \mathbb{N}$, $\pi \in \Pi$, and $x \in \mathbb{X}$ the runtime (and therefore space usage) of* $\mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda, S)$ *and* $\mathsf{Verify}(\mathsf{pp}, x, y, \pi)$ *is* $\leq \mathsf{poly}(\lambda, \log S)$.

We now show how to construct a verifiable space-hard function. We follow the same basic idea as the weak verifiable delay function of [BBBF18] but we require more care because we do not rely on permutation polynomials.

We present the function as an IOP, but the proof can be made non-interactive using the Fiat-Shamir heuristic and Merkle trees [BCS16].

**Construction 2 (Verifiable Space-Hard Function).** *Let* $(\mathsf{P}, \mathsf{V})$ *be an IOPP for Reed-Solomon codes that evaluates the polynomials at points $\mathcal{L}$ and has a negligible soundness error. It proves a string $x$ is at constant relative distance from a Reed-Solomon codeword (degree $d$ polynomials).*

*We also use the combine procedure form Sect. 2.6.*

*Our verifiable space-hard function does not require any setup.*

```
Eval($S, x$):
    – Let $f(X) = \mathsf{H}(x)$.
    – Let the degree of $f(X)$ be $d$.
    – Let $(p_i)_{i \in \{0\} \cup [\lfloor \log p \rfloor]}$ be the binary decomposition of $p$.
    – Let $m_0(X) = X$ and $v_0(X) = X^{p_0}$.
    – For $i \in [\lfloor \log p \rfloor]$:
        • Compute $m_i(X) = m_{i-1}(X) \cdot m_{i-1}(X) \mod f(X)$.
        • Compute $e_i(X) = (m_{i-1}(X) \cdot m_{i-1}(X) - m_i(X))/f(X)$
        • Compute $v_i(X) = v_{i-1}(X) \cdot m_i^{p_i}(X) \mod f(X)$.
        • Compute $w_i(X) = (i_{i-1}(X) \cdot m_i^{p_i}(X)(X))/f(X)$
        • Send $m_i(X)$, $e_i(X)$, $v_i(X)$, and $w_i(X)$ evaluated at $\mathcal{L}$ to the verifier.
    – Compute $g(X)$, the gcd of $v_{\lfloor \log p \rfloor}(X) - X$ and $f(X)$ together with their
      Bézout coefficients $a(X), b(X)$.
    – Send $a(X)$ and $b(X)$ evaluated at $\mathcal{L}$ to the verifier.
    – Pick a uniformly random field element $r \leftarrow_\$ \mathbb{F}_p$.
    – Run $c(X) \leftarrow \mathsf{Combine}(d,\ r,\ (m_i(X), d-1)_{i\in\{0\}\cup[\lfloor\log p\rfloor]},\ (e_i(X), d-2)_{i\in\{0\}\cup[\lfloor\log p\rfloor]},\ (v_i(X), d-1)_{i\in\{0\}\cup[\lfloor\log p\rfloor]},\ (w_i(X), d-2)_{i\in\{0\}\cup[\lfloor\log p\rfloor]},\ (a(X), d-1),\ (b(X), d-2))$.
    – Send $c(X)$ evaluated at $\mathcal{L}$ to the verifier.
    – Run $\mathsf{P}$ to prove that $c(X)$ is at a constant relative distance from a polynomial of degree $d$.
    – Return function output $y = \frac{g(X)}{\text{leading coefficient of } g(X)}$.
Verify$(x, y = g(X))$:
    – Let $f(X) = \mathsf{H}(x)$.
    – If $g(X)$ is not monic return 0.
    – Run $\mathsf{V}$ to verify that $c(X)$ is at a constant relative distance from a polynomial of degree $d$.
    – Let $(p_i)_{i\in\{0\}\cup[\lfloor\log p\rfloor]}$ be the binary decomposition of $p$.
    – Sample a random set $R \subset \mathcal{L}$ of size $\lambda$.
    – For $r \in R$:
        • Read $m_0(r)$ and $v_0(r)$ from the prover string.
        • If $m_0(r) \neq r$ return 0.
        • If $v_0(r) \neq r^{p_0}$ return 0.
    – For $i \in [\lfloor\log p\rfloor]$:
        • For $r \in R$:
            ∗ Read $(m_{i-1},\ m_i(r),\ e_i(r),\ v_{i-1}(r),\ v_i(r),\ w_i(r))$ from the prover string.
            ∗ If $m_{i-1}(r)^2 \neq m_i(r) + e_i(r)\cdot f(r)$ return 0.
            ∗ If $v_{i-1}(r)\cdot m_i(r)^{p_i} \neq v_i(r) + w_i(r)\cdot f(r)$ return 0.
    – For $r \in R$:
        • If $a(r)\cdot(v_{\lfloor\log p\rfloor}(r) - r) + b(r)\cdot f(r) \neq g(r)$ return 0.
    – If $f(X) \mod g(X) \neq 0$ or $X^p - X \mod g(X) \neq 0$ return 0.
    – Return 1.
```

*Remark 2.* Note, the above function does not have high output entropy because $f(X)$ does not have any roots with $\leq 1/2$ probability. This is required by many applications and can easily be fixed by repetition.

**Theorem 4.** *Construction 2 is a verifiable space-hard function.*

*Proof.* Follows from Lemmas lemmas 16 to 18 and Corollary 1.

**Lemma 16.** *[Correctness] Construction 2 is correct.*

*Proof.* Because the prover divides $g(X)$ by its leading coefficient it outputs a monic polynomial. All the checks made by $\mathsf{V}$ pass, which follows from the correctness of $(\mathsf{P}, \mathsf{V})$. For $i \in \{0\} \cup [\lfloor\log p\rfloor]$ it holds that $m_i(X) = X^{2^i} \mod f(X)$. We also have for $i \in [\lfloor\log p\rfloor]$ it holds $m_{i-1}(X)^2 = m_i(X) + f(X)e_i(X)$. Similarly, for binary decomposition $(p_i)_{i\in\{0\}\cup[\lfloor\log p\rfloor]}$ of $p$ we have

$$v_i(X) = \prod_{i\in\{\}\cup[\lfloor\log p\rfloor]} X^{2^i p_i} \mod f(X)$$

and

$$w_i(X) = \prod_{i \in \{\} \cup [\lfloor \log p \rfloor]} X^{2^i p_i} / v_i(X).$$

Therefore, evaluating all these polynomial at the point $r$ still makes these equations hold.

By definition of a common divisor $g(X)$ divides $v_{\lfloor \log p \rfloor}(X) - X$ and $f(X)$. Because $v_{\lfloor \log p \rfloor}(X) - X = X^p - X \mod f(X)$ we get if $g(X)$ divides $v_{\lfloor \log p \rfloor}(X) - X$ and $f(X)$ it also divides $X^p - X$. The Bézout coefficients have the property that $a(X) \cdot (v_{\lfloor \log p \rfloor}(X) - X) + b(X) \cdot f(X) = g(X)$. Therefore, each check passes if $\pi$ is honestly generated.

**Lemma 17.** *[Efficiency] Construction 2 is efficient.*

*Proof.* We require from $\mathcal{D}_{\lambda,S}$ that with all but negligible probability $f(X)$ has a polynomial number of roots. For polynomials where the $\lambda$ low order monomials are dense (see Definition 4) this follows from Lemma 12. Therefore, the degree of $g(X)$ is polynomial in $\lambda$. Reading the prover string for each polynomial at $\leq 2\lambda$ many locations is in $\mathsf{poly}(\lambda)$. If we have an IOPP for Reed-Solomon codes where the verifier runs in time $\mathsf{poly}(\lambda, \log S)$, then the verifier of the entire protocol runs in time $\mathsf{poly}(\lambda, \log S)$.

**Lemma 18.** *[Computational Uniqueness] The Construction 2 is computationally unique.*

*Proof.* Assume there exists an $S \subset \mathcal{L}$ with $|S| \geq (1 - \delta) \cdot |\mathcal{L}|$ such that

- for all $(m_i(X))_{i \in \{0\} \cup [\lfloor \log p \rfloor]}$, there exists a $\hat{m}_i(X) \in \mathsf{RS}[\mathcal{L}, d-1]$ with $m_i(S) = \hat{m}_i(S)$,
- for all $(e_i(X))_{i \in \{0\} \cup [\lfloor \log p \rfloor]}$, there exists a $\hat{e}_i(X) \in \mathsf{RS}[\mathcal{L}, d-2]$ with $e_i(S) = \hat{e}_i(S)$,
- for all $(v_i(X))_{i \in \{0\} \cup [\lfloor \log p \rfloor]}$, there exists a $\hat{v}_i(X) \in \mathsf{RS}[\mathcal{L}, d-1]$ with $v_i(S) = \hat{v}_i(S)$,
- for all $(w_i(X))_{i \in \{0\} \cup [\lfloor \log p \rfloor]}$, there exists a $\hat{w}_i(X) \in \mathsf{RS}[\mathcal{L}, d-2]$ with $w_i(S) = \hat{w}_i(S)$,
- there exist $\hat{a}(X) \in \mathsf{RS}[\mathcal{L}, d-1]$ and $\hat{b}(X) \in \mathsf{RS}[\mathcal{L}, d-2]$ with $a(S) = \hat{a}(S)$ and $b(S) = \hat{b}(S)$.

By Hoeffding's inequality we have that $\Pr(||S \cap R| - \lambda(1-\delta)| \geq \lambda(1-\delta)/2) \leq 2\exp(-2\lambda((1-\delta)/2)^2)$. Since $\delta$ is constant, we have that with all but negligible probability $|S \cap R| \geq \lambda(1-\delta)/2$. Each polynomial equation we check is at most of degree $2d$. Therefore, we get via polynomial identity lemma 1 if a polynomial equation does not hold then evaluating the polynomial at a random point on $S$ and then checking holds with probability $2d/|S|$, which, again, is constant. Because we run this test $|S \cap R|$ many times we detect it with all but negligible probability.

From these equations follows that $\hat{v}_{\lfloor \log p \rfloor}(X) - X = X^p - X \mod f(X)$. Therefore, $gcd(\hat{v}_{\lfloor \log p \rfloor}(X) - X, f(X)) = gcd(X^p - X, f(X))$. By Bézout's identity for all $a'(X)$, $b'(X)$ we have $a'(X) \cdot (\hat{v}_{\lfloor \log p \rfloor}(X) - X) + b'(X) \cdot f(X)$ is a multiple of $gcd(\hat{v}_{\lfloor \log p \rfloor}(X), f(X))$. So, the check $a(X) \cdot (\hat{v}_{\lfloor \log p \rfloor}(X) - X) + b(X) \cdot f(X) = g(X)$ verifies that $g(X)$ is a multiple of $gcd(X^p - X, f(X))$. The checks $f(X) \mod g(X) = 0$ and $X^p - X \mod g(X) = 0$ verify that $g(X)$ is a divisor of $f(X)$ and $X^p - X$. Therefore, $g(X)$ is a greatest common divisor of $f(X)$ and $X^p - X$. The gcd is unique up to multiplication by a field element, which is why we require $g(X)$ to be monic.

If our initial assumption does not hold, then by the soundness of Combine, detailed in Sect. 2.6, we get that $c(X)$ is far from $\mathsf{RS}[\mathcal{L}, d]$. By the soundness of the IOPP that checks that $c(X)$ is close to the code this can only happen with negligible probability.

*Remark 3.* With access to a extractable polynomial commitment scheme the above construction can be made much simpler by replacing the IOPPs with the polynomial commitment. Then we only need to check the polynomials at one location instead of $\lambda$ many locations.

**Corollary 1 (Space Hardness).** *Construction 2 is space-hard.*

*Proof.* Under Assumption 2 for polynomials that are dense in the low degrees 4 space-hardness follows directly from computational uniqueness.

**Acknowledgements.** Nico Döttling and Jesko Dujmovic: Funded by the European Union (ERC, LACONIC, 101041207). Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Research Council. Neither the European Union nor the granting authority can be held responsible for them.

## A Implemented Attack

It follows an implementation of the attack we describe in Sect. 3. The code is written in Magma.

```
//Parameters
p:=101;
s:=p^3;
n:=11;
q:=p^n;

GFq<a>:=GF(q);
GFpol<X>:=PolynomialRing(GFq);

//Definition of the polynomial
function PermEval(mu,x)
  a:=x^(2*s)-2*mu*x^(s+1)+2*mu*x^s+mu^2*x^2+2*mu^2*x+mu^2;
  return (a^((s+1) div 2)+(x^s-mu*x+mu)^s*(x^s-mu*x-mu))/(2*x^s);
end function;
```

```
//Generating the polynomial
//Checking mu is correct (this part is slow).
//Indeed, this is not even part of the attack;
//this defines the function we are trying to invert.
//Can be skipped as mu is likely to be correct.
notfound:=true;
while notfound do
  mu:=Random(GFq);
  if #Roots(X^(s−1)−mu) eq 0 then
    notfound:=false;
  end if;
end while;
input:=Random(GFq);
target:=PermEval(mu,input);

//Attack starts here
delta:=(target^2−4∗mu^(2∗s+1))^((s−1) div 2);
A2:=(target^s−delta∗target)/(2∗mu^s);
A1:=−mu^s∗delta;
A0:=−target^s/2−((target−2∗mu^(s+1))/2)∗delta+mu^(s^2);
B2:=A2−mu^(s^2);
B1:=A1−A2∗mu^s;
B0:=−mu∗A1;
M:=ZeroMatrix(GFq,n,n);
for i:=0 to n−1 do
    M[i+1,i+1]:=B0^(s^i);
    M[i+1,(i+1) mod n+1]:=B1^(s^i);
    M[i+1,(i+2) mod n+1]:=B2^(s^i);
    M[i+1,(i+3) mod n+1]:=1;
end for;
Q:=[];
for i:=0 to n−1 do
  Append(~Q,−A0^(s^i));
end for;
V:=Vector(Q);
M:=Transpose(M);
S:=Solution(M,V);

if (S[1] eq input) then
  print "Correct input recovered!";
end if;
```

## References

AB16. Alwen, J., Blocki, J.: Efficiently computing data-independent memory-hard functions. In: Robshaw, M., Katz, J. (eds.) CRYPTO 2016. LNCS, vol. 9815, pp. 241–271. Springer, Heidelberg (2016). https://doi.org/10.1007/978-3-662-53008-5_9

ABB22. Ameri, M.H., Block, A.R., Blocki, J.: Memory-hard puzzles in the standard model with applications to memory-hard functions and resource-bounded locally decodable codes. In: Galdi, C., Jarecki, S. (eds.) Security and Cryptography for Networks. SCN 2022. LNCS, vol. 13409. Springer, Cham (2022). https://doi.org/10.1007/978-3-031-14791-3_3

ABBK16. Armknecht, F., Barman, L., Bohli, J.-M., Karame, G.O.: Mirror: enabling proofs of data replication and retrievability in the cloud. In: Holz, T., Savage, S. (eds.) USENIX Security 2016, Austin, TX, USA, pp. 1051–1068, 10–12 August. USENIX Association (2016)

ABFG14. Ateniese, G., Bonacina, I., Faonio, A., Galesi, N.: Proofs of space: when space is of the essence. In: Abdalla, M., De Prisco, R. (eds.) SCN 2014. LNCS, vol. 8642, pp. 538–557. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-10879-7_31

ABP18. Alwen, J., Blocki, J., Pietrzak, K.: Sustained space complexity. In: Nielsen, J.B., Rijmen, V. (eds.) EUROCRYPT 2018. LNCS, vol. 10821, pp. 99–130. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-78375-8_4

ACFY24. Arnon, G., Chiesa, A., Fenzi, G., Yogev, E.: Reed-solomon proximity testing with fewer queries. Cryptology ePrint Archive, Stir (2024)

ACP+17. Alwen, J., Chen, B., Pietrzak, K., Reyzin, L., Tessaro, S.: `Scrypt` is maximally memory-hard. In: Coron, J.-S., Nielsen, J.B. (eds.) EUROCRYPT 2017. LNCS, vol. 10212, pp. 33–62. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-56617-7_2

ACY23. Arnon, G., Chiesa, A., Yogev, E.: Iops with inverse polynomial soundness error. In: 2023 IEEE 64th Annual Symposium on Foundations of Computer Science (FOCS), pp. 752–761. IEEE (2023)

AGP24. Auerbach, B., Günther, C.U., Pietrzak, K.: Trapdoor memory-hard functions. In: Annual International Conference on the Theory and Applications of Cryptographic Techniques, pp. 315–344. Springer (2024). https://doi.org/10.1007/978-3-031-58734-4_11

AS15. Alwen, J., Serbinenko, V.: High parallel complexity graphs and memory-hard functions. In: Servedio, R.A., Rubinfeld, R., (eds.) 47th ACM STOC, pp. 595–603, Portland, OR, USA, 14–17 June. ACM Press (2015)

BBBF18. Boneh, D., Bonneau, J., Bünz, B., Fisch, B.: Verifiable delay functions. In: Shacham, H., Boldyreva, A. (eds.) CRYPTO 2018. LNCS, vol. 10991, pp. 757–788. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-96884-1_25

BBHR18. Ben-Sasson, E., Bentov, I., Horesh, Y., Riabzev, M.: Fast reed-solomon interactive oracle proofs of proximity. In: Chatzigiannakis, I., Kaklamanis, C., Marx, D., Sannella, D., (eds.) ICALP 2018, Prague, Czech Republic, vol. 107. LIPIcs, pp. 14:1–14:17, 9–13 July. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik (2018)

BCCT13. Bitansky, N., Canetti, R., Chiesa, A., Tromer, E.: Recursive composition and bootstrapping for SNARKS and proof-carrying data. In: Boneh, D., Roughgarden, T., Feigenbaum, J. (eds.) 45th ACM STOC, Palo Alto, CA, USA, pp. 111–120, 1–4 June. ACM Press (2013)

BCS16. Ben-Sasson, E., Chiesa, A., Spooner, N.: Interactive oracle proofs. In: Hirt, M., Smith, A. (eds.) TCC 2016. LNCS, vol. 9986, pp. 31–60. Springer, Heidelberg (2016). https://doi.org/10.1007/978-3-662-53644-5_2

Ber70. Berlekamp, E.R.: Factoring polynomials over large finite fields. Math. Comput. **24**(111), 713–735 (1970)

BGJ+16. Bitansky, N., Goldwasser, S., Jain, A., Paneth, O., Vaikuntanathan, V., Waters, B.: Time-lock puzzles from randomized encodings. In: Sudan, M (ed.) ITCS 2016, Cambridge, MA, USA, pp. 345–356, 14–16 January. ACM (2016)

BN00. Boneh, D., Naor, M.: Timed commitments. In: Bellare, M. (ed.) CRYPTO 2000. LNCS, vol. 1880, pp. 236–254. Springer, Heidelberg (2000). https://doi.org/10.1007/3-540-44598-6_15

Bon36. Bonferroni, C.: Teoria statistica delle classi e calcolo delle probabilita. Pubblicazioni del R Istituto Superiore di Scienze Economiche e Commericiali di Firenze **8**, 3–62 (1936)

BP17. Biryukov, A., Perrin, L.: Symmetrically and asymmetrically hard cryptography. In: Takagi, T., Peyrin, T. (eds.) ASIACRYPT 2017. LNCS, vol. 10626, pp. 417–445. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-70700-6_15

Cle86. Cleve, R.: Limits on the security of coin flips when half the processors are faulty (extended abstract). In: 18th ACM STOC, Berkeley, CA, USA, pp. 364–369, 28–30 May. ACM Press (1986)

CM19. Chen, J., Micali, S.: Algorand: a secure and efficient distributed ledger. Theoret. Comput. Sci. **777**, 155–183 (2019)

CP18. Cohen, B., Pietrzak, K.: Simple proofs of sequential work. In: Nielsen, J.B., Rijmen, V. (eds.) EUROCRYPT 2018. LNCS, vol. 10821, pp. 451–467. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-78375-8_15

CZ81. Cantor, D.G., Zassenhaus, H.: A new algorithm for factoring polynomials over finite fields. Math. Comput. **36**(154), 587–592 (1981)

DFKP15. Dziembowski, S., Faust, S., Kolmogorov, V., Pietrzak, K.: Proofs of space. In: Gennaro, R., Robshaw, M. (eds.) CRYPTO 2015. LNCS, vol. 9216, pp. 585–605. Springer, Heidelberg (2015). https://doi.org/10.1007/978-3-662-48000-7_29

DH76. Diffie, W., Hellman, M.E.: New directions in cryptography. IEEE Trans. Inf. Theory **22**(6), 644–654 (1976)

DL78. DeMillo, R.A., Lipton, R.J.: A probabilistic remark on algebraic program testing. Inf. Process. Lett. **7**(4), 193–195 (1978)

DLM19. Döttling, N., Lai, R.W.F., Malavolta, G.: Incremental proofs of sequential work. In: Ishai, Y., Rijmen, V. (eds.) EUROCRYPT 2019. LNCS, vol. 11477, pp. 292–323. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-17656-3_11

DMPS19. De Feo, L., Masson, S., Petit, C., Sanso, A.: Verifiable delay functions from supersingular isogenies and pairings. In: Galbraith, S.D., Moriai, S. (eds.) ASIACRYPT 2019. LNCS, vol. 11921, pp. 248–277. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-34578-5_10

GM97. Guralnick, R.M., Müller, P.: Exceptional polynomials of affine type. J. Algebra **194**(2), 429–454 (1997)

GNU16. Guo, Z., Narayanan, A.K., Umans, C.: Algebraic problems equivalent to beating exponent 3/2 for polynomial factorization over finite fields. arXiv preprint arXiv:1606.04592 (2016)

HHK+22. Hoffmann, C., Hubáček, P., Kamath, C., Klein, K., Pietrzak, K.: Practical statistically-sound proofs of exponentiation in any group. In: Dodis, Y., Shrimpton, T. (eds.) Advances in Cryptology – CRYPTO 2022. CRYPTO 2022. Lecture Notes in Computer Science, vol 13508. Springer, Cham (2022). https://doi.org/10.1007/978-3-031-15979-4_13

HHKK23. Hoffmann, C., Hubáček, P., Kamath, C., Krňák, T.: (verifiable) delay functions from lucas sequences. Cryptology ePrint Archive (2023)

Kel16. Kelley, Z.: Roots of sparse polynomials over a finite field. LMS J. Comput. Math. **19**((A)), 196–204 (2016)

KS95. Kaltofen, E., Shoup, V.: Subquadratic-time factoring of polynomials over finite fields. In: Proceedings of the twenty-seventh annual ACM symposium on Theory of computing, pp. 398–406 (1995)

KU11. Kedlaya, K.S., Umans, C.: Fast polynomial factorization and modular composition. SIAM J. Comput. **40**(6), 1767–1802 (2011)

KWJ23. Kavousi, A., Wang, Z., Jovanovic, P.: Sok: Public randomness. Cryptology ePrint Archive, Paper 2023/1121 (2023). https://eprint.iacr.org/2023/1121

KZG10. Kate, A., Zaverucha, G.M., Goldberg, I.: Constant-size commitments to polynomials and their applications. In: Abe, M. (ed.) ASIACRYPT 2010. LNCS, vol. 6477, pp. 177–194. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-17373-8_11

LW17. Lenstra, A.K., Wesolowski, B.: Trustworthy public randomness with sloth, unicorn, and trx. Inter. J. Appli. Cryptography **3**(4), 330–343 (2017)

MMV13. Mahmoody, M., Moran, T., Vadhan, S.P.: Publicly verifiable proofs of sequential work. In: Kleinberg, R.D. (ed.) ITCS 2013, Berkeley, CA, USA, 9–12 January, pp. 373–388. ACM (2013)

Per09. Percival, C.: Stronger key derivation via sequential memory-hard functions (2009)

Pie19. Pietrzak, K.: Simple verifiable delay functions. In: Blum, A. (ed.) ITCS 2019, San Diego, CA, USA, vol. 124, pp. 60:1–60:15, 10–12 January. LIPIcs (2019)

RD16. Ren, L., Devadas, S.: Proof of space from stacked expanders. In: Hirt, M., Smith, A. (eds.) TCC 2016. LNCS, vol. 9985, pp. 262–285. Springer, Heidelberg (2016). https://doi.org/10.1007/978-3-662-53641-4_11

RSW96. Rivest, R.L., Shamir, A., Wagner, D.A.: Time-lock puzzles and timed-release crypto. Technical Report (1996)

Sch80. Schwartz, J.T.: Fast probabilistic algorithms for verification of polynomial identities. J. ACM (JACM) **27**(4), 701–717 (1980)

Sho93. Shoup, V.: Factoring polynomials over finite fields: asymptotic complexity vs. reality. In: Proceedings IMACS Symposium, Lille, France. Citeseer (1993)

Val08. Valiant, P.: Incrementally verifiable computation or proofs of knowledge imply time/space efficiency. In: Canetti, R. (ed.) TCC 2008. LNCS, vol. 4948, pp. 1–18. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-78524-8_1

VZGS92. Von Zur Gathen, J., Shoup, V.: Computing frobenius maps and factoring polynomials. In: Proceedings of the Twenty-fourth Annual ACM Symposium on Theory of Computing, pp. 97–105 (1992)

WC81. Wegman, M.N., Carter, J.L.: New hash functions and their use in authentication and set equality. J. Comput. Syst. Sci. **22**(3), 265–279 (1981)

Wes19. Wesolowski, B.: Efficient verifiable delay functions. In: Ishai, Y., Rijmen, V. (eds.) EUROCRYPT 2019. LNCS, vol. 11478, pp. 379–407. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-17659-4_13

Zip79. Zippel, R.: Probabilistic algorithms for sparse polynomials. In: Ng, E.W. (ed.) Symbolic and Algebraic Computation. LNCS, vol. 72, pp. 216–226. Springer, Heidelberg (1979). https://doi.org/10.1007/3-540-09519-5_73

# Black-Box Timed Commitments from Time-Lock Puzzles

Hamza Abusalah and Gennaro Avitabile(✉)

IMDEA Software Institute, Madrid, Spain
{hamza.abusalah,gennaro.avitabile}@imdea.org

**Abstract.** A Timed Commitment (TC) with time parameter $t$ is hiding for time at most $t$, that is, commitments can be force-opened by any third party within time $t$. In addition to various cryptographic assumptions, the security of all known TC schemes relies on the sequentiality assumption of repeated squarings in hidden-order groups. The repeated squaring assumption is therefore a security bottleneck.

In this work, we give a black-box construction of TCs from any time-lock puzzle (TLP) by additionally relying on one-way permutations and collision-resistant hashing.

Currently, TLPs are known from (a) the specific repeated squaring assumption, (b) the general (necessary) assumption on the *existence of worst-case non-parallelizing languages* and indistinguishability obfuscation, and (c) any iteratively sequential function and the hardness of the circular small-secret LWE problem. The latter admits a plausibly post-quantum secure instantiation.

Hence, thanks to the generality of our transform, we get i) the first TC whose *timed* security is based on the existence of non-parallelizing languages and ii) the first TC that is plausibly post-quantum secure.

We first define *quasi publicly-verifiable* TLPs (QPV-TLPs) and construct them from any standard TLP in a black-box manner without relying on any additional assumptions. Then, we devise a black-box commit-and-prove system to transform any QPV-TLPs into a TC.

## 1 Introduction

Time-lock puzzles (TLPs) introduced by Rivest, Shamir, and Wagner [RSW96] allow one to commit to a message by generating a message-dependent puzzle that ensures the message remains hidden for a certain time. In particular, a TLP consists of a puzzle generation algorithm $\mathsf{Gen}$ and a solving algorithm $\mathsf{Solve}$. On input a message $m$ and a time parameter $t$, $\mathsf{Gen}$ generates a puzzle $z$. On input an honestly-generated puzzle $z$, $\mathsf{Solve}$ retrieves the message $m$. A TLP must satisfy (1) *correctness:* $\mathsf{Solve}$ always recovers the correct message from honestly-generated puzzles (2) *hardness:* the message remains hidden for any massively parallel adversary running in time much less than $t$ and (3) *efficiency:* while $\mathsf{Solve}$ runs in time $t$, $\mathsf{Gen}$ must run in time $\mathsf{polylog}(t)$.

In [RSW96], a very efficient construction whose security is based on the repeated squaring assumption is proposed. This assumption states that given a

© International Association for Cryptologic Research 2025
E. Boyle and M. Mahmoody (Eds.): TCC 2024, LNCS 15366, pp. 460–493, 2025.
https://doi.org/10.1007/978-3-031-78020-2_16

group element $g$, an integer $t$, and an RSA modulus $N$, it is hard to compute $g^{2^t} \mod N$. The puzzle of [RSW96] works as follows: $\mathsf{Gen}(t, m)$ outputs $z = (m + g^{2^t} \mod N, g, N)$, while $\mathsf{Solve}$ computes $g^{2^t}$ to extract $m$. We stress that while correctness of TLPs guarantees solvability for *honestly generated* puzzles, it guarantees nothing for maliciously crafted puzzles. Furthermore, a TLP *per se* does not come with an algorithm that decides whether a TLP is honestly generated or not. For example, given the puzzle of [RSW96], one can retrieve $m$ via $t$ repeated squarings modular $N$, but cannot efficiently verify whether $N$ is an RSA modulus or not.

A standard commitment scheme is a 2-phase interactive protocol between a sender and a receiver such that at the end of the *commit phase*, the sender is bound to a message $m$ (binding) and the receiver learns nothing about $m$ (hiding). In the *open phase*, the sender reveals $m$ to the receiver, which either accepts or rejects. Boneh and Naor [BN00] introduced timed commitments (TCs) which add a time dimension to standard commitments and overcome the limitations of TLPs. In a TC with time parameter $t$, the *commit phase* can be *force-opened* in *sequential* time $t$ without executing the efficient open phase. This is useful in scenarios where the sender refuses to run the open phase. As a result, the message $m$ is only guaranteed to remain hidden for time at most $t$.

A TC must satisfy (standard) binding, *$t$-hiding*, *publicly-verifiable forced openings*, and *well-formedness*. *$t$-hiding* requires that a massively parallel (adversarial) receiver running in time much less than $t$ learns nothing about the committed message. *Well-formedness* requires that if the commit phase terminates successfully, the receiver is guaranteed that the commitment can be forced-open in time $t$ via a force-open algorithm. *Publicly-verifiable forced openings* guarantee that the force-open algorithm, in addition to $m$, outputs a proof $\pi$ that allows anyone in possession of the transcript of the commit phase to *quickly* verify that $m$ is the committed message. A TC must also be efficient, meaning that the commit phase must take time $\mathsf{polylog}(t)$.

Similar to TLPs hardness, $t$-hiding of TCs guarantees that the secrecy of $m$ is preserved for a certain amount of time $t$. The main difference between TLPs and TCs is that, unlike TCs, TLPs fall short of providing public verifiability and well-formedness. That is, TLPs do not provide any means of verifying the correctness of solutions without re-solving the puzzle, and furthermore, puzzles are not guaranteed to be solvable in time $t$ when maliciously generated.

*Known TCs and their Limitations.* Various constructions of TCs [BN00, KLX20, TCLM21, CJ23] have been proposed. Some of them feature useful additional properties such as non-interactive commit phase and CCA security [KLX20, TCLM21, CJ23], or homomorphic properties [TCLM21, CJ23]. Later, we discuss these constructions in the related work section. Nevertheless, they all follow a similar blueprint. To achieve $t$-hiding, all these constructions rely, in a non-black box way, on the repeated-squaring-based puzzle of [RSW96]. They ensure well-formedness by proving in zero knowledge (ZK), with either a proof system for NP or with one tailored to the specific language of interest, that the TLP was computed correctly. Moreover, to achieve public verifiability, they crucially rely on repeated squaring. For example, a common technique to ensure public verifiability [TCLM21, CJ23, FKPS21, BDD+21, BDD+23] is to exploit proofs of exponentiation (e.g., [Pie19, Wes19]) which allow to efficiently check that a value $y$ is equal to $g^{2^t}$ modulo $N$ without executing $t$ squarings.

Consequently, *all known TC constructions are based on the single repeated squaring assumption*. This limitation is significant as it is known that repeated squaring is not necessary to obtain TLPs. Indeed, Bitansky et al. constructed TLPs based on the existence of worst-case non-parallelizing languages and succinct randomized encodings [BGJ+16]. Non-parallelizing languages are necessary for timed-based primitives, such as TLPs and TCs, and are implied by the repeated squaring assumption [BGJ+16, JMRR21].

## 1.1 Our Contributions

The state of affairs in which all known TCs rely on the single sufficient-but-not-necessary assumption of repeated squarings in unknown-order groups is rather unsatisfactory and creates a single point of failure. In this work, we diverge from all known designs of TCs and give a *black-box* transformation of *any* TLP into a TC. By black-box we mean that the TLP (and any other underlying cryptographic primitive) is used only as an oracle. There has been a significant amount of research dedicated to obtaining black-box constructions for various cryptographic primitives [IKLP06, CDMW09, PW09, LP12, GLOV12, Kiy14, HV16, KOS18, Kiy20, COS22]. The benefit of the black-box approach is immediate: basing TCs on TLPs in a black box manner widens the class of known constructions of TCs and allows translating advances in TLPs to TCs. Furthermore, black-box constructions have the advantage that their complexity does not depend on the complexity of the implementation of the underlying primitives[1].

More specifically, we make the following contributions:

- We provide a formal definition of timed commitments. Previous works either gave a rather high-level description of such properties [BN00], or were tailored to the non-interactive setting [KLX20, CJ23].
- Along the way, we define and construct quasi publicly verifiable time-lock puzzles (QPV-TLPs), a novel tool we believe could be of independent interest. Roughly, a QPV-TLP is a TLP where the receiver, if the puzzle is well-formed, is able to provide a convincing proof of correctness of the solution which can be quickly verified. We show how to lift any TLP to a QPV-TLP with black-box use of the TLP itself and without relying on any additional assumptions.
- Relying on QPV-TLPs, we construct a black-box TC assuming the existence of TLPs, collision-resistant hash functions, and one-way permutations. Our construction has a five-round commit phase and a non-interactive open phase.

[1] To better understand this aspect consider a TC which uses a generic (non-black-box) ZK protocol to prove that the circuit computing a TLP was correctly executed. The complexity of this TC would crucially depend on the number of gates of such circuit.

Additionally binding, well-formedness, and public verifiability of forced openings hold w.r.t. unbounded adversaries. Our construction does not require any setup.

- By weakening the well-formedness guarantee to computational, we can get a TC assuming the existence of TLPs and one-way functions. Since TLPs imply one-way functions [BGJ+16], this shows that TLPs imply TCs. Thus, instantiating the TLP with the one of [BGJ+16], we get the first TC whose timed security relies on the weakest complexity assumption of the existence of worst-case non-parallelizing languages. This comes at the cost of the strong cryptographic assumption of indistinguishability obfuscation [BGI+01].
- Using the recent TLP of [AMZ24] as a building block in our transform, we get the first post-quantum secure TC: [AMZ24] constructs TLPs[2] whose security relies on any iteratively sequential function $f$ and the hardness of the circular small-secret LWE problem. By instantiating $f$ based on [LM23], their TLP is plausibly post-quantum secure.

## 1.2 Technical Overview

A straw-man approach to generically construct a TC from a TLP is the following. To achieve well-formedness, the sender proves with a generic ZK proof system that the TLP is correctly computed. To get public verifiability, the receiver uses a succinct non-interactive argument (SNARG) to prove that it solved the TLP correctly. SNARGs succinctness guarantees that verifying the proof is much faster than solving the puzzle. Apart from being non-black-box, this construction has other shortcomings. Indeed, known SNARGs (for P) require setups (e.g., [CJJ22,HJKS22,CGJ+23,Kiy23]) and the only general assumption they are known from is iO [SW14,WW24]. Additionally, SNARGs are only computationally sound, thus constructing a TC following this approach will inherently give computational public verifiability, unlike our construction that achieves it against unbounded adversaries. We propose a different approach that guarantees public verifiability by lifting the TLP to a QPV-TLP, and proves well-formedness in a black-box way.

*Publicly Verifiable Time-Lock Puzzles.* Publicly verifiable time-lock puzzles (PV-TLPs), introduced by Freitag et al. [FKPS21], are an intermediate notion between TLPs and TCs. Unlike TCs, PV-TLPs are not guaranteed to be well-formed. However, after having solved the puzzle, whether the puzzle has a solution $m$ or not, the solver produces a proof allowing anyone to efficiently verify that $m$ is the correct solution or that the puzzle is malformed. PV-TLPs are a good candidate building block to construct TCs as the public verifiability of the TC essentially follows from that of the underlying PV-TLP. Unfortunately,

[2] In the TLP constructions of [AMZ24], if a one-time public-coin setup is allowed, the TLP generation runs in time $\mathsf{polylog}(t)$, otherwise it runs in time $\sqrt{t}$. The (in)efficiency of puzzle generation of the TLP translates to the (in)efficiency of the commit phase of our TC construction.

all known PV-TLPs [FKPS21,BDD+21,BDD+23] are based on the repeated squaring assumption and random oracles.

*Quasi Publicly Verifiable Time-Lock Puzzles.* We notice that the property of providing a proof that is convincing even for invalid puzzles is not necessary to construct a TC. Indeed, to satisfy well-formedness the TC has to prove that the underlying TLPs are correctly generated, meaning that the receiver will not have to solve invalid puzzles. Therefore, we introduce the notion of quasi publicly verifiable time-lock puzzles (QPV-TLP), where the receiver is only guaranteed to produce a convincing proof when the puzzle is well-formed. Remarkably, this simple modification allows us to rely exclusively on the existance of TLPs. We point out that QPV-TLPs are different from *one-sided*[3] PV-TLPs, which are also only known from repeated squaring [FKPS21]. Even though they both provide sound proofs only for well-formed puzzles, one-sided PV-TLPs are additionally required to output accepting proofs for malformed puzzles while still remaining sound w.r.t. honest puzzles. We remark that our QPV-TLP is perfectly sound while PV-TLPs are only computationally sound [FKPS21,BDD+21,BDD+23].

*Our QPV-TLP.* Our QPV-TLP is very straightforward. Given any TLP, to generate a puzzle for a message $m$ and time $t$, we compute $z = (z_0, z_1)$ where $z_0 := \mathsf{Gen}(t, m; r_0)$ and $z_1 := \mathsf{Gen}(t, r_0; r_1)$ with random coins $r_0, r_1$. To solve $z$, we solve $z_0$ and $z_1$ in parallel using $\mathsf{Solve}$, and output $m$ as the message, and $r_0$ as the proof. To verify the correctness of a claimed message, it suffices to check that $z_0 = \mathsf{Gen}(t, m; r_0)$. If a puzzle is correctly generated, the solver is always able obtain the message and convincing proof. Additionally, our QPV-TLP is perfectly sound. This follows from the fact that TLPs are injective, and thus $z_0$ cannot belong to the support of both $\mathsf{Gen}(t, m_0)$ and $\mathsf{Gen}(t, m_1)$ with $m_0 \neq m_1$.

*TLPs as Over-Extractable Commitments.* As pointed out in [LPS17], TLPs can be related to the notion of *extractable* commitments [PW09] with *over-extraction*. A commitment scheme is said to be extractable if there exists an efficient extractor that, having black-box access to a malicious sender that successfully terminates the commit phase, extracts the committed value. An extractable commitment suffers from *over-extraction* if the extractor may output an arbitrary value when the commitment is invalid.[4] TLPs can be seen as over-extractable commitments where extraction is carried out in straight-line by brute-forcing the puzzle, but there is no guarantee on the extracted value if the puzzle is malformed.

Goyal et al. [GLOV12] constructed *weakly extractable* commitments which are extractable in a weaker sense meaning that the extraction can fail with probability 1/2 but without over-extraction. To commit to a message $m$, the

[3] One-sided PV-TLPs are a weaker primitive than PV-TLPs. They are defined and constructed in [FKPS21]. They use a one-sided PV-TLP and a random oracle to get a PV-TLP.

[4] We say that a commitment invalid if it does not have any valid decommitment. Otherwise, the commitment is valid.

sender computes a 2-out-of-2 secret sharing of $m$ and commits to the shares separately using a statistically binding commitment. Then, the receiver challenges the sender to open one of the two commitments. The commit phase terminates successfully if and only if the sender correctly decommits the challenged commitment. To extract the committed value, it suffices to rewind the sender so that it decommits both commitments and to then reconstruct the message starting from the revealed shares. If the sender does not provide a valid decommitment, the extractor outputs $\perp$, denoting that the extraction was not successful.

As a starting point to get well-formedness, we can apply the above idea by replacing the statistically binding commitment with a TLP. When challenged to open a puzzle, the sender provides the message and randomness used to generate the puzzle. The receiver then checks that on input the message and the randomness, the TLP generation algorithm $\mathsf{Gen}$ gives the same puzzle. To extract the committed value, it suffices to solve the unopened puzzle. Unlike [GLOV12], our extraction strategy still suffers from over-extraction. Indeed, our extraction proceeds in straight-line and, since there is no efficient way to decide whether a TLP is malformed, it could be possible to extract a garbage value. Nonetheless, the probability of over-extracting is now at most 1/2.

*A Black-Box Proof of Well-Formedness.* A natural idea to amplify the success probability of the extraction is parallel repetition. The committer samples $\lambda$ different 2-out-of-2 secret sharings of $m$ and commits to them in $2\lambda$ TLPs. Then, the receiver asks the sender to open one of the two commitments for each of the $\lambda$ repetitions. However, a cheating sender could use a different message in different repetitions, making a successful commit phase meaningless. We address this issue by proving, using a black-box commit-and-prove system, that the TLPs across the different repetitions all commit to the same message $m$.

In a black-box commit-and-prove system, a sender commits to a message $m$ so that later it can prove a predicate $\phi$ over the committed $m$ in ZK with black-box use of cryptographic primitives. Constant-round black-box commit-and-prove systems [GLOV12, KOS18, Kiy20, COS22] can be constructed using the powerful MPC-in-the-head paradigm introduced by Ishai et al. [IKOS07]. Let us first briefly describe how to construct a black-box commit-and-prove system with constant soundness error using a 3-party 2-private MPC protocol (e.g., [GMW87]). The sender first commits to the shares of a 3-out-of-3 secret sharing of $m := m_1 \oplus m_2 \oplus m_3$. Then, to prove that the committed message satisfies a predicate $\phi$, the sender runs in its head the MPC protocol computing the functionality $\phi'(m_1, m_2, m_3) := \phi(m_1 \oplus m_2 \oplus m_3)$ and commits to the resulting MPC views. Then, the receiver asks the sender to decommit to the shares $m_i, m_j$ and to the views $\mathsf{view}_i, \mathsf{view}_j$ for random $i, j$ with $i \neq j$. The receiver checks that (1) the decomittment information are correct, (2) $\mathsf{view}_i$ and $\mathsf{view}_j$ are consistent with each other and the output is 1 in both views, and (3) for $\delta \in \{i, j\}$, $m_\delta$ is the input of party $\delta$ in $\mathsf{view}_\delta$. Constant soundness follows from the binding of the commitment and the perfect correctness of the MPC, while ZK follows from the perfect 2-privacy of the MPC.

To get our TC, we start from the above protocol and modify it as follows: (1) to allow the receiver to force-open the commitment, we commit to the shares of the message with a TLP, (2) we repeat in parallel the above black-box commit-and-prove and define $\phi$ as the predicate that ensures that the *same* message is committed across *all* repetitions. To define the predicate, we use a clever technique proposed by Khurana et al. [KOS18]. Instead of committing to $m$, the sender commits to $m||r$, where $r$ is some random value. The receiver replies with a random value $\alpha$, and the sender sends $\gamma := r\alpha + m$ to the receiver. We set $\phi(m||r)$ as the predicate checking that $\gamma$ is computed correctly. By the Schwartz-Zippel lemma, if $m||r \neq m'||r'$ then $r\alpha + m \neq r'\alpha + m'$ with overwhelming probability. Therefore, the fact that $\gamma$ is a global value guarantees consistency of the message committed across the repetitions.

The resulting TC consists of five rounds, the first two of which are dedicated to defining $\phi$ and committing to (the shares of) $m||r$, while the last three consist of the black-box commit-and-prove showing that $\phi(m||r) = 1$ in all repetitions.

Intuitively, well-formedness follows from the fact that an accepting commit phase indicates that the vast majority of the unopened TLPs across the repetitions are well-formed, and that they all open to the same message. Thus, when solving all the unopened puzzles, the receiver would retrieve the committed message in many repetitions. Additionally, the use of a QPV-TLP instead of a regular TLP immediately gives public verifiability of forced openings. The resulting TC achieves $t$-hiding only in the presence of an honest receiver. Intuitively, this is because the underlying black-box commit-and-prove is only honest-verifier ZK. Nonetheless, we can get full-fledged $t$-hiding with a minor modification to the protocol following the approach of Goldreich and Kahan [GK96] to lift an honest-verifier ZK proof to a ZK proof. Namely, we make the receiver commit, with a two-round statistically hiding commitment, to all of its challenges in the second round and to reveal them in the fourth round. Notice that this modification does not increase the number of rounds.

*Public Verifiability Flavors and Efficiency of Forced Openings.* Our TC achieves a very strong flavor of public verifiability. Namely, given an arbitrary transcript of the commit phase, which can even be generated by a *computationally unbounded* malicious sender without interacting with a receiver, we require that it is infeasible to provide two valid force-open proofs for different messages. As a result, the receiver needs to solve all the puzzles during the force-open phase.

However, we can formulate a weaker but meaningful notion that enables a much more efficient force-open procedure. In particular, instead of an arbitrarily generated transcript of the commit phase, we consider a commitment from an accepting commit phase that the malicious sender run with an honest receiver. The well-formedness proof contained in the transcript guarantees that the vast majority of the repetitions contain the same message and, more importantly, the value $\gamma$ can be used to check whether the message extracted from a repetition is consistent with the ones contained in the other (unsolved) puzzles. Thus, as soon as the receiver finds a repetition where the extracted message is consistent with $\gamma$, it will be sure that $m$ is committed in the vast majority of repetitions. Therefore, the force-open procedure can output $m_i||r_i$ together with the proof of the QPV-TLP solved from the $i$-th repetition. Due to the perfect correctness of the MPC, it is easy to show that, with overwhelming probability, $\lambda - O(\log^2(\lambda))$ repetitions are correctly computed. Thus, the probability that the receiver samples more than $\log(\lambda)$ repetitions such that $\gamma \neq \alpha r_i + m_i$ or the unsolved QPV-TLP is malformed is negligible.

The above observation is also made in [KOS18], where the extractor of their black-box commit-and-prove stops rewinding the prover as soon as it can extract a message that is consistent with $\gamma$ from one of the repetitions. Interestingly, this feature of the extraction strategy used in the security proofs of [KOS18] can be exploited to improve the concrete efficiency of our TC. Indeed, in our case the extraction happens in straight-line and it is part of the actual protocol.

*TLPs Imply TCs.* In this work, we have focused on getting a TC whose security guarantees (except $t$-hiding) hold against unbounded adversaries. Our TC relies on TLPs, collision-resistant hashing, and one-way permutations.

However, by weakening the well-formedness guarantee to computational, it is possible to get a $t$-hiding TC without relying on statistically hiding commitments. For example, we can use the technique shown in Sect. 4.1.2 of [CLP20] (which is inspired by [Lin13]) to lift honest-verifier ZK arguments to ZK. The idea is to replace the challenge sent by the receiver in the fourth round of our honest-receiver construction by a coin tossing protocol. The result of such coin tossing is used as the receiver's challenge to finalize the remaining execution. The coin tossing protocol is based on the extractable commitment of Sect. 4 of [PW09] which allows the simulator to extract the receiver's share and bias the coin tossing to its advantage. In this coin tossing, the receiver first commits to a random string using the extractable commitment, the sender replies with its random share, and the receiver opens the commitment. The final challenge is computed by xoring the two shares.

The extractable commitment of [PW09] is a black-box construction from statistically binding commitments. By instantiating the statistically binding commitments in the resulting TC as a two-round protocol from one-way functions, we get a TC whose only assumption is the existence of TLPs, since TLPs imply one-way functions [BGJ+16]. As a result, by instantiating the TLP with the one of [BGJ+16] we get the first TC from general assumptions. Namely, the weakest timed assumption of the existence of worst-case non-parallelizing languages, in addition to indistinguishability obfuscation.

This alternative transformation to lift honest-verifier ZK to ZK leads to a constant increase of the number of rounds in the resulting TC. For the sake of clarity, since the main focus of our work is not to minimize the number of rounds, we have here illustrated a simple approach to base TCs solely on TLPs. Nonetheless, it is conceivable that ZK delayed-input black-box commit-and-prove [KOS18, COS22] could be applied in our setting as well, allowing to have TCs solely based on TLPs in five rounds. Indeed, in delayed-input ZK protocols the predicate to be proved can be decided even in the last round of the protocol. Therefore, the ZK delayed-input black-box commit-and-prove can start right away, in parallel with the exchange of messages used to decide the predicate in our TC. The constructions of [KOS18,COS22] are ZK arguments with four rounds, which become five when the underlying statistically binding commitments are instantiated from one-way functions instead of one-way permutations.

*A Remark on Strongly Extractable Commitments.* Strongly extractable commitments (sExtCom) are extractable commitments (ExtCom) that do not suffer from over-extraction [Kiy14]. In an sExtCom, if the extractor outputs $\perp$ it means that the commitment is invalid. We stress that in an sExtCom, a successful commit phase does not guarantee that the commitment is valid, which is required by TCs. Therefore, even though there exist black-box techniques to get an sExtCom from an ExtCom (e.g., [LP12,Kiy14]), they generally do not give a TC when the ExtCom is replaced with a TLP.

## 1.3 Related Work

*Timed Commitments.* The first (interactive) timed commitment was proposed by Boneh and Naor [BN00]. Their TC relies on a less standard assumption, stronger than repeated squaring, called the generalized Blum-Blum-Shub assumption. Katz et al. [KLX20] introduced and constructed non-interactive TCs. Their TC is also non-malleable under a CCA-like notion. Their TC relies on repeated squaring and uses generic non-interactive zero knowledge (NIZK) to prove well-formedness. It requires a trusted setup and it is inefficient as committing takes time $t$. Additionally, it does not provide public verifiability of forced openings. Thyagarajan et al. [TCLM21] constructed the first CCA-secure non-interactive TC with transparent setup. Their construction is based on the repeated squaring assumption in class groups and is also linearly homomorphic. Its security is proven in the random oracle model. Chvojka and Jager [CJ23] proposed several CCA-secure non-interactive TCs, which all require a trusted setup. They provided constructions with either linear or multiplicative homomorphisms. All their constructions rely on the repeated squaring assumption.

*Publicly Verifiable Time-Lock Puzzles.* Freitag et al. [FKPS21] introduced publicly verifiable time-lock puzzles (PV-TLPs). Their PV-TLP is non-malleable[5] and it is based on *strong trapdoor* VDFs which are only known from repeated squaring [Pie19]. Additionally, they require the (non-programmable auxiliary input) random oracle model and a mild form of setup. Baum et al. [BDD+21,BDD+23] formalize the security of timed primitives in the UC framework [Can01]. They construct PV-TLPs based on the (programmable) random oracle model and strong trapdoor VDFs. They prove that their UC notion can only be achieved in the (programmable) random oracle model.

[5] We refer the reader to [FKPS21] for a comparison between their non-malleability notions and the CCA-like notion of [KLX20].

## 2 Preliminaries

*Notation.* We use $\mathbb{N}$ to denote the set of natural numbers. Let $n \in \mathbb{N}$, we denote the set $\{1, \ldots, n\}$ as $[n]$. For a finite set $S$, we let $s \leftarrow S$ denote the sampling of $s$ uniformly at random from $S$. Let $A$ be a probabilistic algorithm, by $y \leftarrow A(x)$ we denote that $y$ is the output of $A$ on input $x$, while by $y = A(x; r)$ we denote that $y$ is the output of $A$ when it is run with input $x$ and random coins $r$. We model multi-stage algorithms $\mathsf{A} := (\mathsf{A}_0, \mathsf{A}_1, \ldots, \mathsf{A}_k)$ as *stateful* algorithms, i.e., $\mathsf{A}_i$ receives the *state* of $\mathsf{A}_{i-1}$. However, when the state contains data of interest that we wish to highlight, we give it explicitly and keep the rest of the state implicit. For a tuple of interactive randomized algorithms $(S, R)$, let

- $(y_s, y_r) \leftarrow \langle S(x_s) \leftrightarrow R(x_r)\rangle$ denote the interactive protocol between $S$ and $R$ on *private* inputs $x_s$ and $x_r$, respectively. Their respective *private* outputs are $y_s$ and $y_r$.
- $y_r \leftarrow \mathsf{out}_R(\langle S(x_s) \leftrightarrow R(x_r)\rangle)$ denote the *private* output of $R$ in the protocol.
- $\mathsf{trans}(\langle S(x_s) \leftrightarrow R(x_r)\rangle)$ denote the transcript of the protocol, i.e., all the messages $S$ and $R$ exchanged during the protocol.
- $\mathsf{view}_R(\langle S(x_s) \leftrightarrow R(x_r)\rangle)$ denote the view of $R$ in the protocol, i.e., all inputs, incoming and outgoing messages, and random coins of $R$ during the protocol.

### 2.1 Timed Commitments

A timed commitment (TC) is an interactive protocol between a sender $\mathsf{S}$ and a receiver $\mathsf{R}$ defined as follows. We define some specific properties of TCs. *Soundness* guarantees that after a valid commit phase the force-open algorithm will give the committed message $m$ and a convincing proof for $m$, *public verifiability* guarantees that, for any arbitrary commitment, it is unfeasible to provide two accepting force-open proofs w.r.t. different messages, and *t-hiding* guarantees that a massively parallel malicious receiver running in time less than $t$ learns nothing about the committed message. We formalize the TC definition of [BN00] and give different flavors of some of their notions.

**Definition 1.** *A tuple of PT algorithms* $(\mathsf{S}, \mathsf{R}, \mathsf{FOpen}, \mathsf{FVerify})$ *where* $\mathsf{S} = (\mathsf{S_c}, \mathsf{S_o})$ *and* $\mathsf{R} = (\mathsf{R_c}, \mathsf{R_o})$ *are the (stateful)* sender *and (stateful)* receiver *respectively, is an* (interactive) timed commitment scheme *for message and commitment spaces* $\mathcal{M} = (\mathcal{M}_\lambda)_{\lambda \in \mathbb{N}}, \mathcal{C} = (\mathcal{C}_\lambda)_{\lambda \in \mathbb{N}}$ *such that* $\perp \notin \mathcal{M}_\lambda \cup \mathcal{C}_\lambda$ *if the following holds:*

1. *Commit Phase:* $(\mathsf{dec}, \mathsf{com}) \leftarrow \langle \mathsf{S_c}(1^\lambda, t, m) \leftrightarrow \mathsf{R_c}(1^\lambda, t)\rangle$*: On common input a security* $\lambda$ *and a time parameter* $t$*, the receiver* $\mathsf{R_c}$ *runs an interactive protocol with the sender* $\mathsf{S_c}$*, which has a message* $m \in \mathcal{M}_\lambda$ *as additional input.* $\mathsf{R_c}$ *outputs a commitment* $\mathsf{com} \in \mathcal{C}_\lambda \cup \{\perp\}$ *and* $\mathsf{S_c}$ *outputs a decommitment* $\mathsf{dec}$*. (If* $\mathsf{com} \neq \perp$*, the commit phase is valid, otherwise, it is invalid.)*
2. *Open Phase:* $m' \leftarrow \mathsf{out}_{\mathsf{R_o}}(\langle \mathsf{S_o}(\mathsf{dec}) \leftrightarrow \mathsf{R_o}(\mathsf{com})\rangle)$*: The sender* $\mathsf{S_o}$ *on input a decommitment* $\mathsf{dec}$*, and the receiver* $\mathsf{R_o}$ *on input a commitment* $\mathsf{com}$*, run an interactive protocol which results in* $\mathsf{R_o}$ *outputting a message* $m' \in \mathcal{M}_\lambda \cup \{\perp\}$*. (*$m' \neq \perp$ *indicates a valid open phase, otherwise, it is invalid.)*

3. $\underline{(m, \pi) \leftarrow \mathsf{FOpen}(1^\lambda, 1^t, \mathsf{com})}$: *On input a security parameter* $\lambda$, *a time parameter* $t$, *and a commitment* $\mathsf{com}$, *it outputs a message/proof pair* $(m, \pi)$.
4. $\underline{b := \mathsf{FVerify}(1^\lambda, t, \mathsf{com}, m, \pi)}$: *On input a security parameter* $\lambda$, *a time parameter* $t$, *a commitment* $\mathsf{com}$, *a message* $m$, *and a proof* $\pi$, $\mathsf{FVerify}$ *accepts* $(b = 1)$ *or rejects* $(b = 0)$.

*We require correctness of honest and forced openings, soundness, public verifiability, t-hiding, and statistical binding:*

– *Correctness of honest openings: For every* $\lambda, t \in \mathbb{N}$ *and* $m \in \mathcal{M}_\lambda$,

$$\Pr\left[m' = m \;\middle|\; \begin{array}{l}(\mathsf{dec}, \mathsf{com}) \leftarrow \langle \mathsf{S_c}(1^\lambda, t, m) \leftrightarrow \mathsf{R_c}(1^\lambda, t)\rangle \\ m' \leftarrow \mathsf{out}_{\mathsf{R_o}}(\langle \mathsf{S_o}(\mathsf{dec}) \leftrightarrow \mathsf{R_o}(\mathsf{com})\rangle)\end{array}\right] = 1 \ .$$

– *Correctness of force openings: For every* $\lambda, t \in \mathbb{N}$ *and* $m \in \mathcal{M}_\lambda$,

$$\Pr\left[\begin{array}{l}\mathsf{FVerify}(1^\lambda, t, \mathsf{com}, m', \pi) = 1 \wedge \\ m' = m\end{array} \;\middle|\; \begin{array}{l}(\mathsf{dec}, \mathsf{com}) \leftarrow \langle \mathsf{S_c}(1^\lambda, t, m) \leftrightarrow \mathsf{R_c}(1^\lambda, t)\rangle \\ (m', \pi) \leftarrow \mathsf{FOpen}(1^\lambda, 1^t, \mathsf{com})\end{array}\right] = 1 \ .$$

– *Efficiency:* $\mathsf{FOpen}$ *runs in time* $t \cdot \mathsf{poly}(\lambda)$, *while* $\mathsf{FVerify}$ *as well as* $\mathsf{S_c}, \mathsf{S_o}, \mathsf{R_c}, \mathsf{R_o}$ *run in time* $\mathsf{poly}(\log t, \lambda)$.

– *Soundness: For every* $\lambda, t \in \mathbb{N}$ *and every* unbounded *malicious (stateful) sender* $\tilde{\mathsf{S}} := (\tilde{\mathsf{S}}_\mathsf{c}, \tilde{\mathsf{S}}_\mathsf{o})$, *there exists a negligible function* $\mathsf{negl}(\cdot)$ *such that*

$$\Pr\left[\begin{array}{l}\mathsf{com} \neq \bot \wedge \\ (\mathsf{FVerify}(1^\lambda, t, \mathsf{com}, m, \pi) = 0 \\ \vee (m' \neq \bot \wedge m' \neq m))\end{array} \;\middle|\; \begin{array}{l}(\tilde{\mathsf{dec}}, \mathsf{com}) \leftarrow \langle \tilde{\mathsf{S}}_\mathsf{c} \leftrightarrow \mathsf{R_c}(1^\lambda, t)\rangle \\ m' \leftarrow \mathsf{out}_{\mathsf{R_o}}(\langle \tilde{\mathsf{S}}_\mathsf{o}(\tilde{\mathsf{dec}}) \leftrightarrow \mathsf{R_o}(\mathsf{com})\rangle) \\ (m, \pi) \leftarrow \mathsf{FOpen}(1^\lambda, 1^t, \mathsf{com})\end{array}\right] \leq \mathsf{negl}(\lambda) \ .$$

– *Public verifiability: For every* $\lambda, t \in \mathbb{N}$ *and every* unbounded *adversary* $\mathsf{A}$, *there exists a negligible function* $\mathsf{negl}(\cdot)$ *such that*

$$\Pr\left[\begin{array}{l}\mathsf{FVerify}(1^\lambda, t, \mathsf{com}, m_0, \pi_0) = 1 \wedge \\ \mathsf{FVerify}(1^\lambda, t, \mathsf{com}, m_1, \pi_1) = 1 \wedge \\ m_0 \neq m_1\end{array} \;\middle|\; (\mathsf{com}, m_0, \pi_0, m_1, \pi_1) \leftarrow \mathsf{A}\right] \leq \mathsf{negl}(\lambda) \ .$$

– *Statistical binding: For every* $\lambda, t \in \mathbb{N}$ *and every* unbounded *malicious (stateful) sender* $\tilde{\mathsf{S}} := (\tilde{\mathsf{S}}_\mathsf{c}, \tilde{\mathsf{S}}_\mathsf{o}, \tilde{\mathsf{S}}'_\mathsf{o})$, *there exists a negligible function* $\mathsf{negl}(\cdot)$ *such that*

$$\Pr\left[\begin{array}{l}\mathsf{com} \neq \bot \wedge \\ m, m' \neq \bot \wedge \\ m \neq m'\end{array} \;\middle|\; \begin{array}{l}(\tilde{\mathsf{dec}}, \mathsf{com}) \leftarrow \langle \tilde{\mathsf{S}}_\mathsf{c} \leftrightarrow \mathsf{R_c}(1^\lambda, t)\rangle \\ m \leftarrow \mathsf{out}_{\mathsf{R_o}}(\langle \tilde{\mathsf{S}}_\mathsf{o}(\tilde{\mathsf{dec}}) \leftrightarrow \mathsf{R_o}(\mathsf{com})\rangle) \\ m' \leftarrow \mathsf{out}_{\mathsf{R_o}}(\langle \tilde{\mathsf{S}}'_\mathsf{o}(\tilde{\mathsf{dec}}) \leftrightarrow \mathsf{R_o}(\mathsf{com})\rangle)\end{array}\right] \leq \mathsf{negl}(\lambda) \ .$$

– *t-hiding: A timed commitment is t-hiding with gap* $\epsilon < 1$ *if there exists a polynomial* $\tilde{t}(\cdot)$, *such that for every polynomial* $t(\cdot) \geq \tilde{t}(\cdot)$ *and every polynomial-size distinguisher* $\mathsf{D} := (\mathsf{D}_\lambda)_{\lambda \in \mathbb{N}}$ *of* $\mathsf{depth}(\mathsf{D}_\lambda) \leq t(\lambda)^\epsilon$, *there exists a negligible function* $\mathsf{negl}(\cdot)$ *such that for every* $\lambda \in \mathbb{N}$ *and* $m_0, m_1 \in \mathcal{M}_\lambda$,

$$\Pr\left[b' = b \;\middle|\; b \leftarrow \{0,1\}; b' \leftarrow \mathsf{out}_{\mathsf{D}_\lambda}(\langle \mathsf{S_c}(1^\lambda, t, m_b) \leftrightarrow \mathsf{D}_\lambda\rangle)\right] \leq \frac{1}{2} + \mathsf{negl}(\lambda) \ .$$

We also define a weaker version of $t$-hiding called honest-receiver $t$-hiding, where the receiver is honest but curious, that is, it follows the protocol's specification but tries to distinguish the committed value. We define honest-receiver hiding for public-coin protocols, where the messages sent from the receiver to the sender are sampled uniformly at random.

**Definition 2 (Honest-Receiver $t$-Hiding).** *A timed commitment is* honest-receiver *$t$-hiding with gap $\epsilon < 1$ if there exists a polynomial $\tilde{t}(\cdot)$, such that for every polynomial $t(\cdot) \geq \tilde{t}(\cdot)$, there exists a polynomial-size simulator* $\mathsf{Sim} := (\mathsf{Sim}_\lambda)_{\lambda\in\mathbb{N}}$ *of* $\mathsf{depth}(\mathsf{Sim}_\lambda) < t(\lambda)^\epsilon$ *such that for every polynomial-size distinguisher* $\mathsf{D} := (\mathsf{D}_\lambda)_{\lambda\in\mathbb{N}}$ *of* $\mathsf{depth}(\mathsf{D}_\lambda) < t(\lambda)^\epsilon$*, there exists a negligible function* $\mathsf{negl}(\cdot)$ *such that for every $\lambda \in \mathbb{N}$ and message $m \in \mathcal{M}_\lambda$,*

$$\left|\Pr\left[\mathsf{D}_\lambda(\mathsf{trans}(\langle \mathsf{S_c}(1^\lambda, t, m) \leftrightarrow \mathsf{R_c}(1^\lambda, t)\rangle)) = 1\right] - \Pr[\mathsf{D}_\lambda(\mathsf{Sim}_\lambda) = 1]\right| \leq \mathsf{negl}(\lambda).$$

Finally, we define *weak public verifiability*. In the *weak public verifiability* game, the commitment is not directly provided by the adversary but it is the result of a commit phase run with an *honest receiver*. Notice that this weak property basically gives the same guarantees of (regular) public verifiability when commitment transcripts come from a trusted source.[6]

**Definition 3 (Weak Public Verifiability).** *A timed commitment is* weakly publicly verifiable *if for every $\lambda, t \in \mathbb{N}$ and every* unbounded *malicious (stateful) sender* $\tilde{\mathsf{S}} := (\tilde{\mathsf{S}}_\mathsf{c}, \tilde{\mathsf{S}}_\mathsf{fo})$*, there exists a negligible function* $\mathsf{negl}(\cdot)$ *such that,*

$$\Pr\left[\begin{array}{l|l} \mathsf{FVerify}(1^\lambda, t, \mathsf{com}, m_0, \pi_0) = 1\ \wedge & (\tilde{\mathsf{dec}}, \mathsf{com}) \leftarrow \langle \tilde{\mathsf{S}}_\mathsf{c} \leftrightarrow \mathsf{R_c}(1^\lambda, t)\rangle \\ \mathsf{FVerify}(1^\lambda, t, \mathsf{com}, m_1, \pi_1) = 1\ \wedge & (\mathsf{com}, m_0, \pi_0, m_1, \pi_1) \leftarrow \tilde{\mathsf{S}}_\mathsf{fo}(\tilde{\mathsf{dec}}) \\ \mathsf{com} \neq \perp\ \wedge\ m_0 \neq m_1 & \end{array}\right] \leq \mathsf{negl}(\lambda).$$

## 2.2 Time-Lock Puzzles

We recall the definition of time-lock puzzle given in [BGJ+16].

**Definition 4 (Time-Lock Puzzles [BGJ+16]).** *A time-lock puzzle* $\mathsf{TLP}$ *is a pair of algorithms* $(\mathsf{Gen}, \mathsf{Solve})$ *that works as follows:*

- $z \leftarrow \mathsf{Gen}(1^\lambda, t, s)$ *is a probabilistic algorithm that takes as input a security parameter $\lambda \in \mathbb{N}$, a difficulty parameter $t \in \mathbb{N}$, and a solution $s \in \{0,1\}^\lambda$, and outputs a puzzle $z$.*
- $s := \mathsf{Solve}(1^\lambda, 1^t, z)$ *is a deterministic algorithm that takes as input the security parameter $\lambda$, the difficulty parameter $t$, and a puzzle $z$. It outputs a solution $s$.*

*We require correctness and hardness.*

[6] To reduce trust one might require that, for each round, each party signs its outgoing messages, together with all the messages (both incoming and outgoing) exchanged so far. In this setting, the digital signatures act as a certificate to verify that the transcript actually comes from a protocol run by two specific parties. Another technique to reduce trust is to use decentralized TLS oracles [ZMM+20].

– *Correctness: For every security parameter $\lambda$, difficulty parameter $t$, and solution $s \in \{0,1\}^\lambda$:*

$$\Pr\left[\mathsf{Solve}(1^\lambda, 1^t, z) = s \;\middle|\; z \leftarrow \mathsf{Gen}(1^\lambda, t, s)\right] = 1$$

– *Efficiency:* $\mathsf{Gen}$ *runs in time* $\mathsf{poly}(\log t, \lambda)$ *while* $\mathsf{Solve}$ *runs in time* $t \cdot \mathsf{poly}(\lambda)$.
– *Hardness: A time-lock puzzle* $\mathsf{TLP} = (\mathsf{Gen}, \mathsf{Solve})$ *is hard with gap* $\epsilon < 1$ *if there exists a polynomial* $\tilde{t}$*, such that for every polynomial* $t(\cdot) \geq \tilde{t}(\cdot)$*, polynomial-size* $\mathsf{A} := (\mathsf{A})_{\lambda \in \mathbb{N}}$ *of* $\mathsf{depth}(\mathsf{A}) \leq t(\lambda)^\epsilon$*, there exists a negligible function* $\mathsf{negl}(\cdot)$*, such that for every* $\lambda \in \mathbb{N}$*, and every pair of solutions* $s_0, s_1 \in \{0,1\}^\lambda$*,*

$$\Pr\left[b \leftarrow \mathsf{A}(z) \;\middle|\; b \leftarrow \{0,1\}; z \leftarrow \mathsf{Gen}(1^\lambda, t, s_b)\right] \leq \frac{1}{2} + \mathsf{negl}(\lambda).$$

## 2.3 Secure Multi-party Computation

We give a definition of secure multi-party computation protocols based on the ones given by Ishai et al. [IKOS07]. We consider MPC protocols $\Pi_\phi$ that realize any deterministic $N$-party functionality $\phi$ whose output is a bit (received by all parties), and we assume that every party implicitly takes the functionality $\phi$ to be computed as input. We define the concept of consistent views as well as the properties that $\Pi_\phi$ has to satisfy below.

**Definition 5 (Consistent Views).** *Let $\Pi_\phi$ be an $N$-party MPC protocol for functionality $\phi$. We say* $\mathsf{view}_i$ *and* $\mathsf{view}_j$ *with $i, j \in [N]$ s.t. $i \neq j$ are* consistent *if the outgoing messages that can be subsumed from* $\mathsf{view}_i$ *are identical to the incoming messages contained in* $\mathsf{view}_j$ *and vice versa.*

**Definition 6 (Perfect Correctness).** *Let $\Pi_\phi$ be an $N$-party MPC protocol for functionality $\phi$. We say $\Pi_\phi$ has* perfect correctness *if for all private inputs to the parties $(x_1, \ldots, x_N)$, the probability that the output of a party in an honest execution of $\Pi_\phi$ is different from $\phi(x_1, \ldots, x_N)$ is* 0.

**Definition 7 (2-Privacy).** *Let $\Pi_\phi$ be an $N$-party MPC protocol for functionality $\phi$. We say $\Pi_\phi$ has* perfect 2-privacy *if there exists a simulator* $\mathsf{Sim}_{\mathsf{MPC}}$ *such that for any input of the parties $(x_1, \ldots, x_N)$ and corrupted (semi-honest) parties $i, j \in [N]$,* $\mathsf{Sim}_{\mathsf{MPC}}((i,j), (x_i, x_j), \phi(x_1, \ldots, x_n))$ *is identically distributed to the joint view* $(\mathsf{view}_i, \mathsf{view}_j)$.

## 3 Our Constructions

In Sect. 3.1, we define *quasi publicly verifiable* TLPs (QPV-TLPs) and we give a black-box construction from any standard TLP in Sect. 3.2. In Sect. 3.3, we give a black-box construction of TCs with honest-receiver $t$-hiding from any QPV-TLPs. In Sect. 3.4, we lift such construction to full $t$-hiding. This establishes our main result: a black-box timed commitment from any standard TLP. Finally, in Sect. 3.5 we show a version of our TCs that features a more efficient force-open.

## 3.1 Quasi Publicly Verifiable TLPs

We define the concept of quasi publicly verifiable time-lock puzzle as a time-lock puzzle satisfying some additional properties.

**Definition 8.** *A quasi publicly verifiable time-lock puzzle is a tuple of algorithms* $(\mathsf{Gen}, \mathsf{Solve}, \mathsf{Verify})$ *that works as follows:*

- $z \leftarrow \mathsf{Gen}(1^\lambda, t, s)$ *is a probabilistic algorithm that on input a security parameter* $\lambda \in \mathbb{N}$*, a difficulty parameter* $t \in \mathbb{N}$ *and a solution* $s \in \{0,1\}^\lambda$*, outputs a puzzle* $z$*.*
- $(s, \pi) \leftarrow \mathsf{Solve}(1^\lambda, 1^t, z)$ *is a probabilistic algorithm that takes as input the security parameter* $\lambda$*, the difficulty parameter* $t$*, and a puzzle* $z$*. It outputs a solution* $s$ *and a proof* $\pi$*.*
- $b := \mathsf{Verify}(1^\lambda, t, z, s, \pi)$ *is a deterministic algorithm that on input the security parameter* $\lambda$*, the difficulty parameter* $t$*, a puzzle* $z$*, a solution* $s$*, and a proof* $\pi$*, outputs a bit* $b$ *indicating acceptance* $(b = 1)$ *or rejection* $(b = 0)$*.*

*We require the same correctness (considering only the solution* $s$ *output by* $\mathsf{Solve}$*), the same hardness, and same efficiency properties of a regular TLP. Additionally, we require efficient verification, completeness, and perfect soundness.*

- *Efficient verification:* $\mathsf{Verify}$ *runs in time* $\mathsf{poly}(\log t, \lambda)$*.*
- *Completeness: For every security parameter* $\lambda$*, difficulty parameter* $t$*, and* $s \in \{0,1\}^\lambda$*,*

$$\Pr\left[\mathsf{Verify}(1^\lambda, t, z, \mathsf{Solve}(1^\lambda, 1^t, z)) = 1 \;\middle|\; z \leftarrow \mathsf{Gen}(1^\lambda, t, s)\right] = 1.$$

- *Perfect soundness: For all* $\lambda$*, difficulty parameter* $t$*,* $\nexists\, s_0, s_1 \in \{0,1\}^\lambda$*,* $\pi_0, \pi_1$*, and* $z$ *such that* $s_0 \neq s_1 \wedge \mathsf{Verify}(1^\lambda, t, z, s_0, \pi_0) = 1 \wedge \mathsf{Verify}(1^\lambda, t, z, s_1, \pi_1) = 1$*.*

## 3.2 Quasi Publicly Verifiable TLP from Any TLP

In this section, we show how to build a quasi publicly verifiable time-lock puzzle $\mathsf{TLP}^{\mathsf{qpv}} = (\mathsf{Gen}, \mathsf{Solve}, \mathsf{Verify})$ from any time-lock puzzle $\mathsf{TLP} = (\mathsf{TLP.Gen}, \mathsf{TLP.Solve})$. The construction is given in Fig. 1.

**Theorem 1.** *If* $(\mathsf{TLP.Gen}, \mathsf{TLP.Solve})$ *is a time-lock puzzle, then* $(\mathsf{Gen}, \mathsf{Solve}, \mathsf{Verify})$ *of Fig. 1 is a quasi publicly verifiable time-lock puzzle.*

*Remark 1.* An alternative construction would be to replace $z_0, z_1$ in Fig. 1 by $z_0 \leftarrow \mathsf{TLP.Gen}(1^\lambda, t, s || r_1)$ and $z_1 := \mathsf{TLP.Gen}(1^\lambda, t, s; r_1)$. This has the disadvantage that the solution space of $z_0$ is twice as large as that of $z_1$. The advantage, however, is that one needs to *only solve one puzzle* $(z_0)$ and recompute $z_1$.

*Proof (of Theorem 1).* Due to lack of space, we defer the proof to the full version. □

- $z := (z_0, z_1) \leftarrow \mathsf{Gen}(1^\lambda, t, s)$: On input a security parameter $\lambda$, a difficulty parameter $t$, and a solution $s \in \{0,1\}^\lambda$, do:
  - $r \leftarrow \{0,1\}^\lambda$
  - $z_0 := \mathsf{TLP.Gen}(1^\lambda, t, s; r)$
  - $z_1 \leftarrow \mathsf{TLP.Gen}(1^\lambda, t, r)$.
- $(s, \pi) := \mathsf{Solve}(1^\lambda, 1^t, z)$: On input a puzzle $z = (z_0, z_1)$, do
  - $s := \mathsf{TLP.Solve}(1^\lambda, 1^t, z_0)$
  - $\pi := \mathsf{TLP.Solve}(1^\lambda, 1^t, z_1)$.
- $b := \mathsf{Verify}(1^\lambda, t, z, s, \pi)$: On input a puzzle $z = (z_0, z_1)$, a solution $s$, and a proof $\pi$, set $b = 1$ if and only if $z_0 = \mathsf{TLP.Gen}(1^\lambda, t, s; \pi)$.

**Fig. 1.** A black-box construction of QPV-TLP from any (standard) TLP.

## 3.3 Timed Commitment with Honest-Receiver *t*-Hiding

In this section, we give a compiler that starting from a QPV-TLP, a non-interactive statistically binding commitment scheme $\mathsf{SBCom} = (\mathsf{Com}, \mathsf{Dec})$, and a secure MPC protocol $\Pi_\phi$ for a functionality $\phi$ defined below, gives a timed commitment with honest-receiver $t$-hiding. $\Pi_\phi$ computes the following boolean functionality

$$\phi(\alpha, \gamma, x_1, x_2, x_3) := 1 \text{ iff } \gamma = \alpha r + m \text{ where } m||r := x_1 \oplus x_2 \oplus x_3 \wedge |m| = |r| \quad (1)$$

where $\phi, \alpha, \gamma$ are known to all parties and $x_i$ is the private input of party $i$.

All our underlying primitives are used in a black-box way. Furthermore, MPC protocols are information-theoretic constructions, and black-box non-interactive statistically binding commitments can be realized from any one-way permutation [Gol01]. We assume that the messages to be committed to belong to a field $\mathbb{F} \subseteq \{0,1\}^\lambda$. The commit and open phases and the $\mathsf{FOpen}$ and $\mathsf{FVerify}$ algorithms are depicted in Fig. 2 and Fig. 3 respectively.

**Theorem 2.** *Let* $\mathsf{TLP}^{\mathsf{qpv}} = (\mathsf{Gen}, \mathsf{Solve}, \mathsf{Verify})$ *be a quasi publicly verifiable TLP (Def. 8),* $\mathsf{SBCom} = (\mathsf{Com}, \mathsf{Dec})$ *a non-interactive statistically binding and computationally hiding commitment scheme, and* $\Pi_\phi$ *a 3-party secure MPC (as defined in Sect. 2.3) for* $\phi$ *defined in (1), then* $\mathsf{HRTC} = (\mathsf{S} = (\mathsf{S_c}, \mathsf{S_o}), \mathsf{R} = (\mathsf{R_c}, \mathsf{R_o}), \mathsf{FOpen}, \mathsf{FVerify})$ *defined in Fig. 2 and Fig. 3 is an* honest-receiver $t$-hiding *timed commitment for message space* $\mathcal{M}_\lambda = \mathbb{F} \subseteq \{0,1\}^\lambda$.

We prove Theorem 2 by proving several lemmas, one for each of the properties required in Definition 1. Correctness of honest openings and correctness of force openings are easily verified by inspection.

**Lemma 1.** $\mathsf{HRTC}$ *is efficient if* $\mathsf{TLP}^{\mathsf{qpv}}$ *is a QPV-TLP.*

*Proof (of Lemma 1).* First, $\mathsf{FOpen}$ runs in time $t \cdot \mathsf{poly}(\lambda)$: its running time is dominated by $\lambda$ parallel invocations of $\mathsf{TLP}^{\mathsf{qpv}}.\mathsf{Solve}$, each of which runs in time $t \cdot \mathsf{poly}(\lambda)$. Second, $\mathsf{FVerify}$ runs in time $\mathsf{poly}(\log t, \lambda)$: its running time is

**Commit Phase**: $(\mathsf{dec}, \mathsf{com}) \leftarrow \langle \mathsf{S_c}(1^\lambda, t, m) \leftrightarrow \mathsf{R_c}(1^\lambda, t)\rangle$

**Step 1:** $\mathsf{S_c}$ does the following:

1. Sample randomness $r \leftarrow \mathbb{F} \setminus \{0\}$
2. $\forall i \in [\lambda]$, sample $v_{i,1}, v_{i,2} \leftarrow \{0,1\}^{2\lambda}$ and compute $v_{i,3} = v_{i,1} \oplus v_{i,2} \oplus (m||r)$
3. $\forall i \in [\lambda], j \in [3]$, sample randomness $\beta_{i,j}$ and compute $z_{i,j} := \mathsf{TLP}^{\mathsf{qpv}}.\mathsf{Gen}(1^\lambda, t, v_{i,j}; \beta_{i,j})$
4. **Send** $(z_{i,j})_{i\in[\lambda], j\in[3]}$ to $\mathsf{R_c}$.

**Step 2:** $\mathsf{R_c}$ samples $\alpha \leftarrow \mathbb{F} \setminus \{0\}$ and **sends** it to $\mathsf{S_c}$.

**Step 3:** $\mathsf{S_c}$ does the following:

1. Compute $\gamma := r\alpha + m$
2. $\forall i \in [\lambda], j \in [3]$:
   (a) Run $\Pi_\phi$ in the head, with public inputs $\alpha$ and $\gamma$, and $v_{i,j}$ as the private input to party $j$ in the $i$-th execution of $\Pi_\phi$. Let $\mathsf{view}_{i,1}, \mathsf{view}_{i,2}, \mathsf{view}_{i,3}$ be the views of the $i$-th execution
   (b) Compute $(c_{i,j}, d_{i,j}) \leftarrow \mathsf{Com}(1^\lambda, \mathsf{view}_{i,j})$
3. **Send** $(c_{i,j})_{i\in[\lambda], j\in[3]}$ and $\gamma$ to $\mathsf{R_c}$.

**Step 4:** $\forall i \in [\lambda]$, $\mathsf{R_c}$ samples random $a_i, b_i \in [3]$ with $a_i \neq b_i$ and **sends** $(a_i, b_i)_{i\in[\lambda]}$ to $\mathsf{S_c}$.

**Step 5:** $\mathsf{S_c}$ **sends** $(v_{i,a_i}, \mathsf{view}_{i,a_i}, \beta_{i,a_i}, d_{i,a_i}, v_{i,b_i}, \mathsf{view}_{i,b_i}, \beta_{i,b_i}, d_{i,b_i})_{i\in[\lambda]}$ to $\mathsf{R_c}$.

**Output of the commit phase:** $\mathsf{S_c}$ and $\mathsf{R_c}$ output $\mathsf{dec}$ and $\mathsf{com}$ where

- $\mathsf{S_c}$ sets $\mathsf{dec} = (v_{i,k}, \beta_{i,k})_{i\in[\lambda], k\in[3]\setminus\{a_i,b_i\}}$.
- $\mathsf{R_c}$ performs the following checks $\forall i \in [\lambda], \delta \in \{a_i, b_i\}$:
  1. $\mathsf{Dec}(1^\lambda, c_{i,\delta}, \mathsf{view}_{i,\delta}, d_{i,\delta}) = 1$
  2. $\mathsf{TLP}^{\mathsf{qpv}}.\mathsf{Gen}(1^\lambda, t, v_{i,\delta}; \beta_{i,\delta}) = z_{i,\delta}$
  3. $\mathsf{view}_{i,a_i}$ and $\mathsf{view}_{i,b_i}$ are consistent
  4. $(v_{i,\delta}, 1)$ is the input/output pair of $P_\delta$ defined by $\mathsf{view}_{i,\delta}$.

  If the checks are successful, $\mathsf{R_c}$ sets

$$\mathsf{com} = (z_{i,1}, z_{i,2}, z_{i,3}, \alpha, \gamma, a_i, b_i, v_{i,a_i}, \beta_{i,a_i}, v_{i,b_i}, \beta_{i,b_i})_{i\in[\lambda]} \tag{2}$$

  otherwise, $\mathsf{com} = \perp$.

**Open Phase**: $m \leftarrow \mathsf{out}_{\mathsf{R_o}}(\langle \mathsf{S_o}(\mathsf{dec}) \leftrightarrow \mathsf{R_o}(\mathsf{com})\rangle)$

**Decommitment:** $\mathsf{S_o}$ **sends** $\mathsf{dec}$ to $\mathsf{R_o}$.

**Output of the open phase:** If for more than $\lambda/2$ repetitions[a] $i \in [\lambda]$, there exists a *fixed* $m' \in \mathbb{F}$ s.t. $\mathsf{TLP}^{\mathsf{qpv}}.\mathsf{Gen}(1^\lambda, t, v_{i,k}; \beta_{i,k}) = z_{i,k}$ where $k \in [3] \setminus \{a_i, b_i\}$ and $v_{i,a_i} \oplus v_{i,b_i} \oplus v_{i,k} = m'||r_i$ for any $r_i \in \{0,1\}^\lambda$, then $\mathsf{R_o}$ outputs $m = m'$. Otherwise, $m = \perp$.

[a] Allowing up to $\lambda/2$ inconsistent positions in the open phase ensures that a (potentially malicious) accepting commit phase corresponds to a valid commitment, i.e., there exists a valid decommitment for it.

**Fig. 2.** Commit and open phases of $\mathsf{HRTC}$.

```
$(m, \pi) \leftarrow \mathsf{FOpen}(1^\lambda, 1^t, \mathsf{com})$: Perform the following steps:
    1. Initialize the set $I = \emptyset$ and parse $\mathsf{com}$ as in (2)
    2. In parallel, $\forall i \in [\lambda]$ and $k \in [3] \setminus \{a_i, b_i\}$, run $(v_{i,k}, \pi_{i,k}) \leftarrow \mathsf{TLP}^{\mathsf{qpv}}.\mathsf{Solve}(1^\lambda, 1^t, z_{i,k})$. If $\mathsf{TLP}^{\mathsf{qpv}}.\mathsf{Verify}(1^\lambda, t, z_{i,k}, v_{i,k}, \pi_{i,k}) = 1$, then $I = I \cup \{i\}$.
    3. If there exists $J \subseteq I$ s.t. $|J| > \lambda/2$ and $\forall j \in J$ it holds that $v_{j,a_j} \oplus v_{j,b_j} \oplus v_{j,k} = m'||r_j$ where $k \in [3] \setminus \{a_j, b_j\}$, $r_j \in \{0,1\}^\lambda$ arbitrary and $m' \in \mathbb{F}$ fixed, then set $m = m'$ and $\pi := (J, (v_{j,k}, \pi_{j,k}))_{j \in J, k \in [3] \setminus \{a_j, b_j\}}$; otherwise return $m = \pi = \perp$.

$b := \mathsf{FVerify}(1^\lambda, t, \mathsf{com}, m, \pi)$: If either $m = \perp$ or $\pi = \perp$ return 0. Otherwise, parse $\mathsf{com}$ as in (2) and $\pi := (J, (v_j^*, \pi_j^*)_{j \in J})$ and set $b = 1$ iff $|J| > \lambda/2$ and $\forall j \in J$:
    1. $z_{j,a_j} = \mathsf{TLP}^{\mathsf{qpv}}.\mathsf{Gen}(1^\lambda, t, v_{j,a_j}; \beta_{j,a_j})$
    2. $z_{j,b_j} = \mathsf{TLP}^{\mathsf{qpv}}.\mathsf{Gen}(1^\lambda, t, v_{j,b_j}; \beta_{j,b_j})$
    3. $\mathsf{TLP}^{\mathsf{qpv}}.\mathsf{Verify}(1^\lambda, t, z_{j,k}, v_j^*, \pi_j^*) = 1$ for $k \in [3] \setminus \{a_j, b_j\}$
    4. $v_{j,a_j} \oplus v_{j,b_j} \oplus v_j^* = m||r_j$ for any $r_j \in \{0,1\}^\lambda$.
```

**Fig. 3.** FOpen and FVerify algorithms of HRTC.

dominated by $O(\lambda)$ invocations of $\mathsf{TLP}^{\mathsf{qpv}}.\mathsf{Gen}$ and $\mathsf{TLP}^{\mathsf{qpv}}.\mathsf{Verify}$, each of which runs in time $\mathsf{poly}(\log t, \lambda)$. Finally, $\mathsf{S_c}, \mathsf{S_o}, \mathsf{R_c}, \mathsf{R_o}$ all run in time $\mathsf{poly}(\log t, \lambda)$ as all the primitives they invoke either take time $\mathsf{poly}(\lambda)$ (e.g. SBCom and MPC in the head) or $\mathsf{poly}(\log t, \lambda)$ (e.g. $\mathsf{TLP}^{\mathsf{qpv}}.\mathsf{Gen}$ and $\mathsf{TLP}^{\mathsf{qpv}}.\mathsf{Verify}$). □

**Lemma 2.** HRTC *is sound if* $\mathsf{TLP}^{\mathsf{qpv}}$ *is a QPV-TLP,* SBCom *is a non-interactive statistically biding commitment, and* $\Pi_\phi$ *is a secure* 3*-party MPC.*

*Proof (of Lemma 2).* For simplicity, in the following proof, we treat both SBCom and $\mathsf{TLP}^{\mathsf{qpv}}$ as perfectly binding. This means that once the sender commits to a view (with SBCom) or a share (with $\mathsf{TLP}^{\mathsf{qpv}}$), it is impossible for the sender to later decommit to a different value than the one initially committed to. While $\mathsf{TLP}^{\mathsf{qpv}}$ is indeed perfectly binding (a proof of this is given in Lemma 4), SBCom is only statistically binding. However, the statistical binding property of SBCom ensures that each commitment defines a string such that, except with negligible probability, only that specific string can be decommitted to later. Therefore, the soundness error of our construction when using statistically binding commitments is at most negligibly larger than that of using perfectly binding commitments (i.e., this difference accounts for the negligible probability that the sender breaks the binding of one of the commitments).

Let us call a repetition $i \in [\lambda]$ bad if $\phi(\alpha, \gamma, v_{i,1}, v_{i,2}, v_{i,3}) = 0$ or $\exists u \in [3]$ s.t. the time-lock puzzle $z_{i,u}$ is malformed. We argue that conditioned on $\mathsf{com} \neq \perp$, the probability that a repetition is bad at most $\frac{1}{3}$. First, observe that the receiver detects a malformed puzzle with probability at least $\frac{1}{3}$. Indeed, the receiver would ask the sender to provide $v_{i,u}, \beta_{i,u}$ s.t. $z_{i,u} = \mathsf{TLP}^{\mathsf{qpv}}.\mathsf{Gen}(1^\lambda, t, v_{i,u}; \beta_{i,u})$ with probability $\frac{1}{3}$ (i.e., see the second verification check performed by $\mathsf{R_c}$ when deciding the output of the commit phase in Fig. 2).

Second, if $\phi(\alpha, \gamma, v_{i,1}, v_{i,2}, v_{i,3}) = 0$, then the receiver will also detect it with probability at least $\frac{1}{3}$. Indeed, if within a repetition $\phi(\alpha, \gamma, v_{i,1}, v_{i,2}, v_{i,3}) = 0$, by the perfect correctness of $\Pi_\phi$, for all choices of $v_{i,1}, v_{i,2}, v_{i,3}$, the outputs of all 3 parties in any honest execution of $\Pi_\phi$ must be 0. Therefore, considering the inputs $(v_{i,j})_{j\in[3]}$ and views $(\mathsf{view}_{i,j})_{j\in[3]}$ committed by the sender in Step 1 and Step 3 of the commit phase respectively, either in all the views the output is 0, or there exists two views which are malformed (i.e., they are either inconsistent with each other or have different inputs w.r.t. the ones committed in Step 1). In the former case, the receiver would always reject, while in the latter it would reject with probability at least $\frac{1}{3}$ (i.e., see the third and fourth verification checks performed by $\mathsf{R_c}$ when deciding the output of the commit phase in Fig. 2). Therefore, the probability that, given that $\mathsf{com} \neq \perp$, a repetition $i \in [\lambda]$ is $\mathsf{bad}$ is at most $\frac{1}{3}$.

Let us now consider the probability that, conditioned on $\mathsf{com} \neq \perp$, more than $\log^2(\lambda)$ repetitions are $\mathsf{bad}$. Such probability is at most $(\frac{1}{3})^{\log^2(\lambda)} < (\frac{1}{2})^{\log^2(\lambda)} = (\frac{1}{\lambda})^{\log(\lambda)}$, which is negligible.

Let $S \subseteq [\lambda]$ be the set containing the indices of the repetitions that are not $\mathsf{bad}$. For every $i \in S$ where $v_{i,1} \oplus v_{i,2} \oplus v_{i,3} = m_i || r_i$ with $|m_i| = |r_i|$, it holds that $\gamma = \alpha r_i + m_i$. By the Schwartz-Zippel lemma, with overwhelming probability over the choice of $\alpha$, there must exist $(m, r)$ such that $m_i = m$ and $r_i = r$ for all $i \in S$. Recall that $\forall i \in S$ and $k \in [3]$ s.t. $a_i, b_i \neq k$ we have that $z_{i,k}$ is well-formed. The correctness and the completeness of $\mathsf{TLP^{qpv}}$ guarantee that whenever a puzzle is correctly generated, the solution/proof pair output by $\mathsf{TLP^{qpv}.Solve}$ will be accepting and will contain the committed value. Thus, since $|S| \geq \lambda - \log^2(\lambda)$ except with negligible probability, the force open phase and the open phase will agree on the same message $m$ on more than $\frac{\lambda}{2}$ positions (i.e., $m' = m$) and $\mathsf{FVerify}$ will output 1 on input the message/proof pair returned by $\mathsf{FOpen}$. $\square$

**Lemma 3.** $\mathsf{HRTC}$ *is publicly verifiable if* $\mathsf{TLP^{qpv}}$ *is a QPV-TLP.*

*Proof (of Lemma 3).* Assume $\mathsf{A}$ outputs $\mathsf{com} = (z_{i,1}, z_{i,2}, z_{i,3}, \alpha, \gamma, a_i, b_i, v_{i,a_i}, \beta_{i,a_i}, v_{i,b_i}, \beta_{i,b_i})_{i\in[\lambda]}$ and two proof/message pairs $(m_0, \pi_0 = (v^0_{i,k}, \pi^0_{i,k})_{i\in[\lambda]})$ and $(m_1, \pi_1 = (v^1_{i,k}, \pi^1_{i,k})_{i\in[\lambda]})$ with $m_0 \neq m_1$ and that are accepting w.r.t. $\mathsf{com}$. This means that for more than $\lambda/2$ repetitions $v_{i,a_i} \oplus v_{i,b_i} \oplus v^0_{i,k} = m_0 || r^0_i$ and $v_{i,a_i} \oplus v_{i,b_i} \oplus v^1_{i,k} = m_1 || r^1_i$. Thus, there must be at least one repetition $i \in [\lambda]$ s.t. $v^0_{i,k} \neq v^1_{i,k}$ and $\mathsf{TLP^{qpv}.Verify}(1^\lambda, t, z_{i,k}, v^0_{i,k}, \pi^0_{i,k}) = 1$ and $\mathsf{TLP^{qpv}.Verify}(1^\lambda, t, z_{i,k}, v^1_{i,k}, \pi^1_{i,k}) = 1$, which contradicts the perfect soundness of $\mathsf{TLP^{qpv}}$. $\square$

**Lemma 4.** $\mathsf{HRTC}$ *is statistically binding if* $\mathsf{TLP^{qpv}}$ *is a QPV-TLP.*

*Proof (of Lemma 4).* Assume that the malicious sender provides two valid decommitments $\mathsf{dec}_0 = (v^0_{i,k}, \beta^0_{i,k})_{i\in[\lambda]}$ and $\mathsf{dec}_1 = (v^1_{i,k}, \beta^1_{i,k})_{i\in[\lambda]}$ to $m_0$ and $m_1$, respectively, such that $m_0 \neq m_1$, for the same $\mathsf{com} = (z_{i,1}, z_{i,2}, z_{i,3}, \alpha, \gamma, a_i, b_i, v_{i,a_i}, \beta_{i,a_i}, v_{i,b_i}, \beta_{i,b_i})_{i\in[\lambda]}$. This means that for more than $\lambda/2$ repetitions $v_{i,a_i} \oplus v_{i,b_i} \oplus v^0_{i,k} = m_0 || r^0_i$ and $v_{i,a_i} \oplus v_{i,b_i} \oplus v^1_{i,k} = m_1 || r^1_i$. Thus, there exists at least one repetition $i \in [\lambda]$ such that $v^0_{i,k} \neq v^1_{i,k}$ and $z_{i,k} = \mathsf{TLP}^{\mathsf{qpv}}.\mathsf{Gen}(1^\lambda, t, v^0_{i,k}; \beta^0_{i,k})$ and $z_{i,k} = \mathsf{TLP}^{\mathsf{qpv}}.\mathsf{Gen}(1^\lambda, t, v^1_{i,k}; \beta^1_{i,k})$, which contradicts the correctness of $\mathsf{TLP}^{\mathsf{qpv}}$. Indeed, the correctness property of TLPs guarantees that they are injective. For the reader's convenience we report again here the proof of this fact, as we have already shown in the proof of perfect soundness of $\mathsf{TLP}^{\mathsf{qpv}}$ (see Sect. 3.2). Any puzzle $z$, even if maliciously generated, belongs to the support of $\mathsf{TLP}^{\mathsf{qpv}}.\mathsf{Gen}(1^\lambda, t, s)$ for at most one solution $s$. Assume, for the sake of contradiction, that there exists a $z$ belonging to the support of both $\mathsf{TLP}^{\mathsf{qpv}}.\mathsf{Gen}(1^\lambda, t, s_0)$ and $\mathsf{TLP}^{\mathsf{qpv}}.\mathsf{Gen}(1^\lambda, t, s_1)$ with $s_0 \neq s_1$. Let $s = \mathsf{TLP}^{\mathsf{qpv}}.\mathsf{Solve}(1^\lambda, 1^t, z)$. If $s \neq s_0$, this contradicts the correctness of $\mathsf{TLP}^{\mathsf{qpv}}.\mathsf{Solve}$ regarding puzzles in the support of $\mathsf{TLP}^{\mathsf{qpv}}.\mathsf{Gen}(1^\lambda, t, s_0)$. If $s = s_0$, it contradicts the correctness of $\mathsf{TLP}^{\mathsf{qpv}}.\mathsf{Solve}$ regarding puzzles in the support of $\mathsf{TLP}^{\mathsf{qpv}}.\mathsf{Gen}(1^\lambda, t, s_1)$. Hence, for any puzzle $z$, there exists at most one solution $s$, and if a solution $s$ exists, then $s = \mathsf{TLP}^{\mathsf{qpv}}.\mathsf{Solve}(1^\lambda, 1^t, z)$. □

**Lemma 5.** HRTC *is honest-receiver t-hiding if* SBCom *is a non-interactive computationally hiding commitment,* $\mathsf{TLP}^{\mathsf{qpv}}$ *is a QPV-TLP, and* $\Pi_\phi$ *is a secure 3-party MPC.*

*Proof (of Lemma 5).* The simulator Sim proceeds as follows:

1. Sample random $\alpha, \gamma \leftarrow \mathbb{F} \setminus \{0\}$
2. $\forall i \in [\lambda]$ sample random $a_i, b_i \leftarrow [3]$ with $a_i \neq b_i$
3. $\forall i \in [\lambda], j \in \{a_i, b_i\}, k \in [3] \setminus \{a_i, b_i\}$ sample $v_{i,j} \leftarrow \{0,1\}^{2\lambda}$ and set $v_{i,k}$ to garbage, say $v_{i,k} = 0^{2\lambda}$
4. $\forall (i,j) \in [\lambda] \times [3]$ sample randomness $\beta_{i,j}$ and compute $z_{i,j} := \mathsf{TLP}^{\mathsf{qpv}}.\mathsf{Gen}(1^\lambda, t, v_{i,j}; \beta_{i,j})$
5. $\forall i \in [\lambda], k \in [3] \setminus \{a_i, b_i\}$, compute $(\mathsf{view}_{i,a_i}, \mathsf{view}_{i,b_i}) \leftarrow \mathsf{Sim}_{\mathsf{MPC}}((a_i, b_i), (v_{i,a_i}, v_{i,b_i}), 1)$ and set $\mathsf{view}_{i,k} := 0^\ell$ where $\ell$ is the size of a view
6. $\forall (i,j) \in [\lambda] \times [3]$ compute $(c_{i,j}, d_{i,j}) \leftarrow \mathsf{Com}(1^\lambda, \mathsf{view}_{i,j})$
7. Output the protocol's transcript $(z_{i,1}, z_{i,2}, z_{i,3}, \alpha, c_{i,1}, c_{i,2}, c_{i,3}, \gamma, a_i, b_i, v_{i,a_i}, \mathsf{view}_{i,a_i}, \beta_{i,a_i}, d_{i,a_i}, v_{i,b_i}, \mathsf{view}_{i,b_i}, \beta_{i,b_i}, d_{i,b_i})_{i \in [\lambda]}$.

Notice the running time of Sim is essentially independent of $t$. We prove indistinguishability from a regular transcript via a series of indistinguishable hybrids.

$\mathcal{H}_0$ : This is identical to the sender in the commit phase of Fig. 2 except that, instead of receiving $\alpha, (a_i, b_i)_{i \in [\lambda]}$ from the receiver, the sender *itself* samples samples random $\alpha \leftarrow \mathbb{F} \setminus \{0\}$ and for all $i \in [\lambda]$, $a_i, b_i \leftarrow [3]$ with $a_i \neq b_i$. Formally, $\mathcal{H}_0$ is defined as follows:

1. Sample random $r, \alpha \leftarrow \mathbb{F} \setminus \{0\}$ and set $\gamma := r\alpha + m$
2. $\forall i \in [\lambda]$ sample random $a_i, b_i \in [3]$ with $a_i \neq b_i$
3. $\forall i \in [\lambda]$ sample $v_{i,1}, v_{i,2} \leftarrow \{0,1\}^{2\lambda}$ and compute $v_{i,3} = v_{i,1} \oplus v_{i,2} \oplus (m||r)$
4. $\forall i \in [\lambda], j \in [3]$, sample randomness $\beta_{i,j}$ and compute $z_{i,j} := \mathsf{TLP}^{\mathsf{qpv}}.\mathsf{Gen}(1^\lambda, t, v_{i,j}; \beta_{i,j})$

5. $\forall i \in [\lambda], j \in [3]$, run $\Pi_\phi$ in the head, with public inputs $\alpha$ and $\gamma$, and $v_{i,j}$ as the private input to party $j$ in the $i$-th execution of $\Pi_\phi$. Let $\mathsf{view}_{i,1}, \mathsf{view}_{i,2}, \mathsf{view}_{i,3}$ be the views of the $i$-th execution
6. $\forall i \in [\lambda], j \in [3]$, compute $(c_{i,j}, d_{i,j}) \leftarrow \mathsf{Com}(1^\lambda, \mathsf{view}_{i,j})$
7. Output the protocol's transcript: $(z_{i,j}, \alpha, c_{i,j}, \gamma, a_i, b_i, v_{i,a_i}, \mathsf{view}_{i,a_i}, \beta_{i,a_i}, d_{i,a_i}, v_{i,b_i}, \mathsf{view}_{i,b_i}, \beta_{i,b_i}, d_{i,b_i})_{i\in[\lambda], j\in[3]}$

$\mathcal{H}_1$ : This is identical to $\mathcal{H}_0$ except that for all $i \in [\lambda]$ and $j \in \{a_i, b_i\}$, the shares $v_{i,j}$ are randomly sampled before drawing the remaining share depending on $m||r$. In more details, Step 3. of $\mathcal{H}_0$ is modified as follows. The remaining steps remain unchanged. Formally, $\mathcal{H}_1$ is defined as follows:

⋮

3. $\forall i \in [\lambda], j \in \{a_i, b_i\}, k \in [3] \setminus \{a_i, b_i\}$, sample $v_{i,j} \leftarrow \{0,1\}^{2\lambda}$ and set $v_{i,k} = v_{i,a_i} \oplus v_{i,b_i} \oplus (m||r)$

⋮

$\mathcal{H}_2^i$ : This hybrid is parameterized by $i \in \{0, \dots, \lambda\}$. We define $\mathcal{H}_2^0 := \mathcal{H}_1$. For $i \geq 1$ $\mathcal{H}_2^i$ be identical to $\mathcal{H}_2^{i-1}$ except that in the $i$-th repetition of the unopened party in Step 6. of $\mathcal{H}_1$, instead of committing to $\mathsf{view}_{i,k}$ where $k \in [3] \setminus \{a_i, b_i\}$, commit to a garbage value, say 0. Formally, $\forall i \in [\lambda]$, $\mathcal{H}_2^i$ is defined as follows:

⋮

6. $\forall j \in [\lambda]$ and $k \in [3] \setminus \{a_j, b_j\}$ :
   - $(c_{j,a_j}, d_{j,a_j}) \leftarrow \mathsf{Com}(1^\lambda, \mathsf{view}_{j,a_j})$
   - $(c_{j,b_j}, d_{j,b_j}) \leftarrow \mathsf{Com}(1^\lambda, \mathsf{view}_{j,b_j})$
   - If $j \leq i$:$(c_{j,k}, d_{j,k}) \leftarrow \mathsf{Com}(1^\lambda, 0^\ell)$ where $\ell := |\mathsf{view}_{j,a_j}| = |\mathsf{view}_{j,b_j}|$, i.e., views are padded and are of size $\ell$.
   - If $j > i$: $(c_{j,k}, d_{j,k}) \leftarrow \mathsf{Com}(1^\lambda, \mathsf{view}_{j,k})$.

⋮

$\mathcal{H}_3^i$ : This hybrid is parameterized by $i \in \{0, \dots, \lambda\}$ and we define $\mathcal{H}_3^0 := \mathcal{H}_2$. For $i \geq 1$, $\mathcal{H}_3^i$ is identical to $\mathcal{H}_3^{i-1}$ except that in $i$-th repetition where $k \in [3] \setminus \{a_i, b_i\}$, instead of committing with $\mathsf{TLP}^{\mathsf{qpv}}$ to the input $v_{i,k}$ of the unopened party in Step 4. of $\mathcal{H}_2^\lambda$, we commit to a garbage value, say 0. Formally, $\forall i \in [\lambda]$, $\mathcal{H}_3^i$ is defined as follows:

⋮

4. $\forall j \in [\lambda]$ and $k \in [3] \setminus \{a_j, b_j\}$ :
   - $z_{j,a_j} := \mathsf{TLP}^{\mathsf{qpv}}.\mathsf{Gen}(1^\lambda, t, v_{j,a_j}; \beta_{j,a_j})$
   - $z_{j,b_j} := \mathsf{TLP}^{\mathsf{qpv}}.\mathsf{Gen}(1^\lambda, t, v_{j,b_j}; \beta_{j,b_j})$
   - If $j \leq i$: $z_{j,k} := \mathsf{TLP}^{\mathsf{qpv}}.\mathsf{Gen}(1^\lambda, t, 0^{2\lambda}; \beta_{j,k})$
   - If $j > i$: $z_{j,k} := \mathsf{TLP}^{\mathsf{qpv}}.\mathsf{Gen}(1^\lambda, t, v_{j,k}; \beta_{j,k})$.

```
                    ⋮
```

$\mathcal{H}_4$ : $\mathcal{H}_4$ is identical to $\mathcal{H}_3^\lambda$ except that in Step 5. of $\mathcal{H}_3^\lambda$ for all $i \in [\lambda]$ we compute the opened views as $(\mathsf{view}_{i,a_i}, \mathsf{view}_{i,b_i}) \leftarrow \mathsf{Sim}_{\mathsf{MPC}}((a_i, b_i), (v_{i,a_i}, v_{i,b_i}), 1)$. Formally, $\mathcal{H}_4$ is defined as follows:

```
                    ⋮
5. $\forall i \in [\lambda] : (\mathsf{view}_{i,a_i}, \mathsf{view}_{i,b_i}) \leftarrow \mathsf{Sim}_{\mathsf{MPC}}((a_i, b_i), (v_{i,a_i}, v_{i,b_i}), 1)$
                    ⋮
```

$\mathcal{H}_5$ : $\mathcal{H}_5$ is identical to $\mathcal{H}_4$ except that $\gamma$ in Step 1. is sampled uniformly at random. Formally, $\mathcal{H}_5$ is defined as:

```
1. Sample random $\alpha, \gamma \leftarrow \mathbb{F} \setminus \{0\}$
                    ⋮
```

Notice that $\mathcal{H}_0$ coincides with the distribution of the transcript of an honest commit phase and $\mathcal{H}_5$ is identical to $\mathsf{Sim}$. We now introduce the following claims.

**Claim 1.** *$\mathcal{H}_1$ is perfectly indistinguishable from $\mathcal{H}_0$.*

**Claim 2.** *If $\mathsf{SBCom}$ is computationally hiding, then for every $i \in [\lambda]$, $\mathcal{H}_2^i$ is computationally indistinguishable from $\mathcal{H}_2^{i-1}$.*

**Claim 3.** *If $\mathsf{TLP}^{\mathsf{qpv}}$ is hard, then for every $i \in [\lambda]$, $\mathcal{H}_3^i$ and $\mathcal{H}_3^{i-1}$ are indistinguishable for every poly-size distinguisher with depth upper-bounded by $t^\epsilon$.*

**Claim 4.** *If $\Pi_\phi$ is perfectly 2-private, then $\mathcal{H}_4$ is perfectly indistinguishable from $\mathcal{H}_3^\lambda$.*

**Claim 5.** *$\mathcal{H}_5$ is perfectly indistinguishable from $\mathcal{H}_4$.*

*Proof (of Claim 1).* It follows from the fact that for any $x \in \{0,1\}^{2\lambda}$ and all $i, j, k \in [3]$ s.t. $i \neq j \neq k$ where $v_i, v_j \leftarrow \{0,1\}^{2\lambda}$ and $v_k = v_i \oplus v_j \oplus x$, any pair $(v_a, v_b)$ with $a, b \in [3]$ s.t. $a \neq b$ is uniformly distributed in $\{0,1\}^{2\lambda} \times \{0,1\}^{2\lambda}$. □

*Proof (of Claim 2).* Assume there exists a poly-sized distinguisher $\mathsf{D}$ which is able to distinguish between $\mathcal{H}_2^i$ and $\mathcal{H}_2^{i-1}$ with non-negligible probability, we can use $\mathsf{D}$ to build an adversary $\mathsf{A}$ that wins the hiding game of $\mathsf{SBCom}$ with the same probability. $\mathsf{A}$ plays in the hiding game with $m_0 = \mathsf{view}_{i,k}$ and $m_1 = 0^\ell$ and gets back from the challenger a commitment $c$. $\mathsf{A}$ constructs the transcript regularly except that for the $i$-th repetition it sets $c_{i,k} = c$. It then forwards the transcript to $\mathsf{D}$ and outputs whatever $\mathsf{D}$ outputs. Notice that if $b = 0$ in the hiding game, then $\mathsf{A}$ perfectly simulates $\mathcal{H}_2^{i-1}$, and if $b = 1$ then $\mathsf{A}$ perfectly simulates $\mathcal{H}_2^i$. Therefore, $\mathsf{A}$ wins the hiding game with the same probability with which $\mathsf{D}$ distinguishes between $\mathcal{H}_2^i$ and $\mathcal{H}_2^{i-1}$. □

*Proof (of Claim 3).* Assume there exists a poly-sized distinguisher $\mathsf{D}$ whose depth is bounded by $t^\epsilon$ which is able to distinguish between $\mathcal{H}_3^i$ and $\mathcal{H}_3^{i-1}$ with non-negligible probability, we can use $\mathsf{D}$ to build an adversary $\mathsf{A}$ that breaks the hardness of $\mathsf{TLP}^{\mathsf{qpv}}$ with the same probability. $\mathsf{A}$ plays in the hardness game with $s_0 = v_{i,k}$ and $s_1 = 0^{2\lambda}$ and gets back from the challenger a puzzle $z$. $\mathsf{A}$ constructs the transcript regularly except that for the $i$-th repetition it sets $z_{i,k} = z$. It then forwards the transcript to $\mathsf{D}$ and outputs whatever $\mathsf{D}$ outputs. Notice that if $b = 0$ in the hardness game, then $\mathsf{A}$ perfectly simulates $\mathcal{H}_3^{i-1}$, and if $b = 1$ then $\mathsf{A}$ perfectly simulates $\mathcal{H}_3^i$. Notice that constructing the transcript keeps the depth of $\mathsf{A}$ bounded since the MPC protocol, the commitment algorithm, and the puzzle generation algorithm all run in polynomial time essentially independent of $t$. Therefore, $\mathsf{A}$ wins the hardness game with the same probability with which $\mathsf{D}$ distinguishes between $\mathcal{H}_3^i$ and $\mathcal{H}_3^{i-1}$. □

*Proof (of Claim 4).* It directly follows from the perfect indistinguishability of a pair of real views and a pair of simulated views of $\Pi_\phi$. □

*Proof (of Claim 5).* It follows from the fact that for any $m \in \mathbb{F}$ and random $\alpha, r \in \mathbb{F} \setminus \{0\}$ the value $\gamma = r\alpha + m$ is uniformly distributed in $\mathbb{F} \setminus \{0\}$. □

The proof is concluded by observing that it simply follows from the proofs of Claims 1 -5, since all of the hybrids are indistinguishable by a poly-size distinguisher whose depth is bounded by $t^\epsilon$. □

## 3.4 $t$-Hiding Timed Commitment

To lift our *honest-receiver* $t$-hiding timed commitment to a $t$-hiding timed commitment we adopt the approach used by Goldreich and Kahan [GK96] to get a constant-round ZK proof from a constant-round honest-verifier ZK proof. Basically, we have the receiver commit, with a two-round statistically-hiding commitment ($\mathsf{SHCom}$), to all of its random challenges at the beginning of the protocol, and then open these commitments in the subsequent rounds. Two-round statistically-hiding commitments can be constructed from collision-resistant hashing [HM96]. We describe the commit phase of our $t$-hiding TC in Fig. 4.

The verification of the commit phase and its output, of the open phase, and the $\mathsf{FOpen}, \mathsf{FVerify}$ algorithms are identical to the ones described in Sect. 3.3.

**Theorem 3.** *Let* $\mathsf{TLP}^{\mathsf{qpv}} = (\mathsf{Gen}, \mathsf{Solve}, \mathsf{Verify})$ *be a QPV-TLP (Def. 8),* $\mathsf{SBCom} = (\mathsf{Com}, \mathsf{Dec})$ *a non-interactive statistically binding and computationally hiding commitment scheme,* $\mathsf{SHCom}$ *a two-round statistically hiding and computationally binding commitment scheme, and* $\Pi_\phi$ *a 3-party secure MPC for functionality* $\phi$ *defined in* (1)*, then* $\mathsf{TC} = (\mathsf{S} = (\mathsf{S_c}, \mathsf{S_o}), \mathsf{R} = (\mathsf{R_c}, \mathsf{R_o}), \mathsf{FOpen}, \mathsf{FVerify})$*, where* $\mathsf{S}$ *and* $\mathsf{R}$ *are defined in Fig. 4 and* $\mathsf{FOpen}, \mathsf{FVerify}$ *are defined Fig. 3, is a (fully fledged)* $t$-hiding *timed commitment scheme (Def. 1).*

We only prove $t$-hiding and soundness, as the proofs of public verifiability and binding are identical to the construction in Sect. 3.3 – see Lemmas 3 and 4.

**Commit Phase**: $(\mathsf{dec}, \mathsf{com}) \leftarrow \langle \mathsf{S_c}(1^\lambda, t, m) \leftrightarrow \mathsf{R_c}(1^\lambda, t)\rangle$

**Step 1:** $\mathsf{S_c}$ does the following:
1. Sample randomness $r \leftarrow \mathbb{F} \setminus \{0\}$
2. $\forall i \in [\lambda]$:
   (a) Sample $v_{i,1}, v_{i,2} \leftarrow \{0,1\}^{2\lambda}$ and compute $v_{i,3} = v_{i,1} \oplus v_{i,2} \oplus (m||r)$
   (b) $\forall j \in [3]$, sample randomness $\beta_{i,j}$ and compute $z_{i,j} := \mathsf{TLP^{qpv}.Gen}(1^\lambda, t, v_{i,j}; \beta_{i,j})$
   (c) Compute the first message $\sigma_i[1]$ of the two-round $\mathsf{SHCom}$
3. **Send** $(z_{i,j})_{i\in[\lambda], j\in[3]}, (\sigma_i[1])_{i\in[\lambda]}$ to $\mathsf{R_c}$.

**Step 2:** $\mathsf{R_c}$ does the following:
1. $\forall i \in [\lambda]$,
   (a) Sample random $a_i, b_i \in [3]$ with $a_i \neq b_i$
   (b) Commit using $\mathsf{SHCom}$ to $(a_i, b_i)$ by computing the second message $\sigma_i[2]$ of the two-round $\mathsf{SHCom}$. ($\sigma_i = (\sigma_i[1], \sigma_i[2])$ commits to $(a_i, b_i)$.)
2. Sample random $\alpha \leftarrow \mathbb{F} \setminus \{0\}$
3. **Send** $\alpha, (\sigma_i[2])_{i\in[\lambda]}$ to $\mathsf{S_c}$

**Step 3:** $\mathsf{S_c}$ does the following:
1. Compute $\gamma = r\alpha + m$.
2. $\forall i \in [\lambda], j \in [3]$:
   (a) Run $\Pi_\phi$ in the head, with public input $\alpha$ and $\gamma$, and $v_{i,j}$ as the private input to party $j$ in the $i$-th execution of $\Pi_\phi$. Let $\mathsf{view}_{i,1}, \mathsf{view}_{i,2}, \mathsf{view}_{i,3}$ be the views of the $i$-th execution
   (b) Compute $(c_{i,j}, d_{i,j}) \leftarrow \mathsf{Com}(1^\lambda, \mathsf{view}_{i,j})$
3. **Send** $(c_{i,j})_{i\in[\lambda], j\in[3]}$ and $\gamma$ to $\mathsf{R_c}$.

**Step 4:** $\forall i \in [\lambda]$, $\mathsf{R_c}$ computes decommitment $y_i$ to $(a_i, b_i)$ w.r.t. $\sigma_i$ and **sends** $(y_i)_{i\in[\lambda]}$ to $\mathsf{S_c}$.

**Step 5:** If $\forall i \in [\lambda]$, $y_i$ is a valid decommitment to $(a_i, b_i)$ w.r.t. $\sigma_i$, $\mathsf{S_c}$ **sends** $(v_{i,a_i}, \mathsf{view}_{i,a_i}, \beta_{i,a_i}, d_{i,a_i}, v_{i,b_i}, \mathsf{view}_{i,b_i}, \beta_{i,b_i}, d_{i,b_i})_{i\in[\lambda]}$ to $\mathsf{R_c}$. Otherwise, $\mathsf{S_c}$ **aborts**.

The output of the commit phase, i.e., $(\mathsf{dec}, \mathsf{com})$, as well as the open phase are identical to those in Fig. 2.

**Fig. 4.** Commit phase of $\mathsf{TC}$.

**Lemma 6.** $\mathsf{TC}$ *is $t$-hiding if* $\mathsf{SBCom}$ *is a non-interactive computationally hiding commitment scheme,* $\mathsf{SHCom}$ *is a two-round computationally binding commitment scheme,* $\mathsf{TLP^{qpv}}$ *is a quasi publicly verifiable time-lock puzzle, and $\Pi_\phi$ is a 3-party secure MPC for functionality $\phi$ defined in* (1)*.*

*Proof (of Lemma 6).* Our proof strategy involves constructing a bounded-depth simulator $\mathsf{Sim}$ that, with black-box access to a malicious (bounded-depth) receiver $\mathsf{R_c^*}$, is able to simulate its view. Such simulated view is indistinguishable from the one coming from a commit phase with an honest sender for every possible message $m \in \mathcal{M}_\lambda$. Let $\approx$ denote that two distributions are indistinguishable by a polynomial-size distinguisher whose depth is bounded by $t^\epsilon$. We then observe that for all $m_0, m_1 \in \mathcal{M}_\lambda$: $\mathsf{view}_{\mathsf{R_c^*}}(\langle \mathsf{S_c}(1^\lambda, t, m_0) \leftrightarrow \mathsf{R_c^*}(1^\lambda, t)\rangle) \approx$

$\mathsf{Sim}^{\mathsf{R}_\mathsf{c}^*} \approx \mathsf{view}_{\mathsf{R}_\mathsf{c}^*}(\langle \mathsf{S}_\mathsf{c}(1^\lambda, t, m_1) \leftrightarrow \mathsf{R}_\mathsf{c}^*(1^\lambda, t)\rangle)$. Therefore, $\mathsf{view}_{\mathsf{R}_\mathsf{c}^*}(\langle \mathsf{S}_\mathsf{c}(1^\lambda, t, m_0) \leftrightarrow \mathsf{R}_\mathsf{c}^*(1^\lambda)\rangle) \approx \mathsf{view}_{\mathsf{R}_\mathsf{c}^*}(\langle \mathsf{S}_\mathsf{c}(1^\lambda, t, m_1) \leftrightarrow \mathsf{R}_\mathsf{c}^*(1^\lambda)\rangle)$, i.e., the existence of this simulator implies $t$-hiding. We first describe a *simplified* simulator $\mathsf{Sim}$:

1. $\forall i \in [\lambda]$, compute the first message $\sigma_i[1]$ of $\mathsf{SHCom}$; $\forall i \in [\lambda], j \in [3]$, compute $z_{i,j} := \mathsf{TLP}^{\mathsf{qpv}}.\mathsf{Gen}(1^\lambda, t, 0^{2\lambda}; \beta_{i,j})$ for random $\beta_{i,j}$; **send** $(\sigma_i[1])_{i\in[\lambda]}$ and $(z_{i,j})_{i\in[\lambda],j\in[3]}$ to $\mathsf{R}_\mathsf{c}^*$.
2. **receive** $\forall i \in [\lambda]$ from $\mathsf{R}_\mathsf{c}^*$ its statistically hiding commitment $\sigma_i[2]$ to $(a_i, b_i)$, and the value $\alpha$.
3. $\forall i \in [\lambda], j \in [3]$, compute $(c_{i,j}, d_{i,j}) \leftarrow \mathsf{Com}(1^\lambda, 0^\ell)$ ; sample a random $\gamma \leftarrow \mathbb{F} \setminus \{0\}$; **send** $(c_{i,j})_{i\in[\lambda],j\in[3]}$ and $\gamma$ to $\mathsf{R}_\mathsf{c}^*$.
4. **receive** $\forall i \in [\lambda]$ from $\mathsf{R}_\mathsf{c}^*$ the decommitments $y_i$ w.r.t. $\sigma_i$ to the challenge indices $(a_i, b_i)$; **abort** if any of the decommitments is not valid.
5. **Rewind Phase:** repeatedly rewind $\mathsf{R}_\mathsf{c}^*$ back to Step 3, until $\mathsf{R}_\mathsf{c}^*$ decommits to $(a'_i, b'_i)_{i\in[\lambda]} = (a_i, b_i)_{i\in[\lambda]}$. In particular, instead of Steps 3 and 4, do the following:
   (a) $\forall i \in [\lambda]$, compute $(\mathsf{view}_{i,a_i}, \mathsf{view}_{i,b_i}) \leftarrow \mathsf{Sim}_{\mathsf{MPC}}((a_i, b_i), (v_{i,a_i}, v_{i,b_i}), 1)$ and $\mathsf{view}_{i,k} := 0^\ell$ for $k \in [3] \setminus \{a_i, b_i\}$; $\forall i \in [\lambda], j \in [3]$, compute $(c_{i,j}, d_{i,j}) \leftarrow \mathsf{Com}(1^\lambda, \mathsf{view}_{i,j})$; sample a random $\gamma \leftarrow \mathbb{F}\setminus\{0\}$; **send** $\gamma$ and $(c_{i,j})_{i\in[\lambda],j\in[3]}$ to $\mathsf{R}_\mathsf{c}$.
   (b) **receive** $\forall i \in [\lambda]$ from $\mathsf{R}_\mathsf{c}^*$ the decommitments $y'_i$ w.r.t. $\sigma_i$ to the challenge indices $(a'_i, b'_i)$; if any of the decommitments $y'_i$ is not valid, execute again Step (5a) (with fresh randomness).
   (c) if $\exists i \in [\lambda]$ s.t. $(a_i, b_i) \neq (a'_i, b'_i)$, output $\mathsf{ambiguous}$. Otherwise, exit rewind phase and proceed to Step 6.
6. **send** $(v_{i,a_i}, \mathsf{view}_{i,a_i}, \beta_{i,a_i}, d_{i,a_i}, v_{i,b_i}, \mathsf{view}_{i,b_i}, \beta_{i,b_i}, d_{i,b_i})_{i\in[\lambda]}$ to $\mathsf{R}_\mathsf{c}^*$.

Akin to what happens in the the protocol by Goldreich and Kahan [GK96], the simple simulation strategy of $\mathsf{Sim}$ suffers from the problem that $\mathsf{Sim}$ may not terminate in expected polynomial time. The issue stems from the fact that $\mathsf{SBCom}$ is only computationally hiding. Let $p_0$ be the probability with which $\mathsf{R}_\mathsf{c}^*$ correctly decommits $(\sigma_i)_{i\in\lambda}$ when it receives commitments $(c_{i,j})_{i\in\lambda,j\in[3]}$ to $0^{2\lambda}$ (i.e., Step 3). Similarly, let $p_1$ be probability with which $\mathsf{R}_\mathsf{c}^*$ correctly decommits $(\sigma_i)_{i\in\lambda}$ when (some of the) $(c_{i,j})_{i\in\lambda,j\in[3]}$ are commitments to the output of $\mathsf{Sim}_{\mathsf{MPC}}$ as in Step 5 of $\mathsf{Sim}$. Although $|p_0 - p_1|$ is negligible due to the computational hiding of $\mathsf{SBCom}$, this (negligible) difference in the behaviour of $\mathsf{R}_\mathsf{c}^*$ may cause $\mathsf{Sim}$ to run in exponential time.

To solve this issue, we can use the same technique of [GK96] to modify the simple simulator above to ensure that it does not run for too long. This technique involves first estimating, via a polynomial number of rewinds, the value of $p_0$ and using such value to limit the total the number of rewinds. Since this technique identically applies to our setting, we omit its description and refer the reader to [GK96,Lin16] for more details. For the rest of the proof we can ignore such subtlety and refer to the simplified simulator above. Indeed, the modifications introduced by the technique of [GK96] do not involve the simulation strategy itself, but they only take care of the running time of the simulator.

It remains to show that the output of $\mathsf{Sim}$ and the view of $\mathsf{R}_\mathsf{c}^*$ in an interaction with honest $\mathsf{S}_\mathsf{c}$ are indistinguishable by a polynomial-size distinguisher whose depth is bounded by $t^\epsilon$. We show this via a sequence of indistinguishable hybrids. Let $\mathcal{H}_0$ be the real world interaction between $\mathsf{S}_\mathsf{c}$ and $\mathsf{R}_\mathsf{c}^*$ (i.e., Fig. 4).

$\mathcal{H}_1$ : $\mathcal{H}_1$ is identical to $\mathcal{H}_0$ except that $\mathcal{H}_1$ behaves like $\mathsf{Sim}$ in the sense that it runs the rewind phase in the same way as $\mathsf{Sim}$, and outputs $\mathsf{ambiguous}$ under the same conditions. However, $\mathcal{H}_1$ is provided with $m$, and it commits (via $\mathsf{TLP}^{\mathsf{qpv}}$) to an honest secret sharing of $m||r$ and commits (via $\mathsf{Com}$) to honest views from the MPC in the head. Formally, $\mathcal{H}_1$ is defined as follows:

1. sample randomness $r \leftarrow \mathbb{F} \setminus \{0\}$; $\forall i \in [\lambda]$, compute the first message $\sigma_i[1]$ of $\mathsf{SHCom}$, sample $v_{i,1}, v_{i,2} \leftarrow \{0,1\}^{2\lambda}$ and compute $v_{i,3} = v_{i,1} \oplus v_{i,2} \oplus (m||r)$; $\forall i \in [\lambda], j \in [3]$, sample random $\beta_{i,j}$ and compute $z_{i,j} := \mathsf{TLP}^{\mathsf{qpv}}.\mathsf{Gen}(1^\lambda, t, v_{i,j}; \beta_{i,j})$ ; **send** $(\sigma_i[1])_{i\in[\lambda]}$ and $(z_{i,j})_{i\in[\lambda],j\in[3]}$ to $\mathsf{R}_\mathsf{c}^*$.
2. **receive** $\forall i \in [\lambda]$ from $\mathsf{R}_\mathsf{c}^*$ its statistically hiding commitment $\sigma_i[2]$ to $(a_i, b_i)$, and the value $\alpha$.
3. $\forall i \in [\lambda]$, run $\Pi_\phi$ in the head, with public inputs $\alpha$ and $\gamma$, and $v_{i,j}$ as the private input to party $j$ in the $i$-th execution of $\Pi_\phi$. Let $\mathsf{view}_{i,1}, \mathsf{view}_{i,2}, \mathsf{view}_{i,3}$ be the views of the $i$-th execution;
4. $\forall i \in [\lambda], j \in [3]$, compute $(c_{i,j}, d_{i,j}) \leftarrow \mathsf{Com}(1^\lambda, \mathsf{view}_{i,j})$; **send** $(c_{i,j})_{i\in[\lambda],j\in[3]}$ and $\gamma := r\alpha + m$ to $\mathsf{R}_\mathsf{c}^*$.
5. **receive** $\forall i \in [\lambda]$ from $\mathsf{R}_\mathsf{c}^*$ the decommitments $y_i$ w.r.t. $\sigma_i$ to the challenge indices $(a_i, b_i)$; **abort** if any of the decommitments is not valid.
6. **Rewind Phase:** repeatedly rewinds $\mathsf{R}_\mathsf{c}^*$ back to Step 3.:
   (a) $\forall i \in [\lambda]$, run $\Pi_\phi$ in the head, with public inputs $\alpha$ and $\gamma$, and $v_{i,j}$ as the private input to party $j$ in the $i$-th execution of $\Pi_\phi$. Let $\mathsf{view}_{i,1}, \mathsf{view}_{i,2}, \mathsf{view}_{i,3}$ be the views of the $i$-th execution
   (b) $\forall i \in [\lambda], j \in [3]$, compute $(c_{i,j}, d_{i,j}) \leftarrow \mathsf{Com}(1^\lambda, \mathsf{view}_{i,j})$
   i. **send** $(c_{i,j})_{i\in[\lambda],j\in[3]}$ and $\gamma := r\alpha + m$ to $\mathsf{R}_\mathsf{c}^*$.
   (c) **receive** $\forall i \in [\lambda]$ from $\mathsf{R}_\mathsf{c}^*$ the decommitments $y_i'$ w.r.t. $\sigma_i$ to the challenge indices $(a_i', b_i')$; if any of the decommitments $y_i'$ is not valid, execute again Step 6. (with fresh randomness).
   (d) If $\exists i \in [\lambda]$ s.t. $(a_i, b_i) \neq (a_i', b_i')$, output $\mathsf{ambiguous}$. Otherwise, exit the rewind phase and proceed Step 7..
7. **send** $(v_{i,a_i}, \mathsf{view}_{i,a_i}, \beta_{i,a_i}, d_{i,a_i}, v_{i,b_i}, \mathsf{view}_{i,b_i}, \beta_{i,b_i}, d_{i,b_i})_{i\in[\lambda]}$ to $\mathsf{R}_\mathsf{c}^*$.

$\mathcal{H}_2^i$ : This hybrid is parameterized by $i \in \{0, \ldots, \lambda\}$. We define $\mathcal{H}_2^0 := \mathcal{H}_1$ and let $\mathcal{H}_2^i$ be identical to $\mathcal{H}_2^{i-1}$ except that in Step 6., $\mathcal{H}_2^i$ computes $(c_{i,\delta}, d_{i,\delta}) \leftarrow \mathsf{Com}(1^\lambda, \mathsf{view}_{i,\delta})$ and $(c_{i,k}, d_{i,k}) \leftarrow \mathsf{Com}(1^\lambda, 0^\ell)$ for $\delta \in \{a_i, b_i\}$ and $k \in [3]$ s.t. $a_i, b_i \neq k$. Formally, $\mathcal{H}_2^i$ is defined as follows:

```
⋮
6. $\forall j \in [\lambda]$ and $k \in [3] \setminus \{a_j, b_j\}$:
   • $(c_{j,a_j}, d_{j,a_j}) \leftarrow \mathsf{Com}(1^\lambda, \mathsf{view}_{j,a_j})$
   • $(c_{j,b_j}, d_{j,b_j}) \leftarrow \mathsf{Com}(1^\lambda, \mathsf{view}_{j,b_j})$
   • If $j \le i$:$(c_{j,k}, d_{j,k}) \leftarrow \mathsf{Com}(1^\lambda, 0^\ell)$
   • If $j > i$: $(c_{j,k}, d_{j,k}) \leftarrow \mathsf{Com}(1^\lambda, \mathsf{view}_{j,k})$.
⋮
```

$\mathcal{H}_3$ : This is identical to $\mathcal{H}_2^\lambda$ except that in Step 6., the MPC views are simulated. Formally, $\mathcal{H}_3$ is defined as follows:

```
⋮
6. $\forall i \in [\lambda]$, compute
   $(\mathsf{view}_{i,a_i}, \mathsf{view}_{i,b_i}) \leftarrow \mathsf{Sim}_{\mathsf{MPC}}((a_i, b_i), (v_{i,a_i}, v_{i,b_i}), 1)$ and $\mathsf{view}_{i,k} := 0^\ell$
   for $k \in [3] \setminus \{a_i, b_i\}$
⋮
```

$\mathcal{H}_4^i$ : This hybrid is parameterized by $i \in \{0, \ldots, 3\lambda\}$. We define $\mathcal{H}_4^0 := \mathcal{H}_3$. For $i \ge 1$, $\mathcal{H}_4^i$ is identical to $\mathcal{H}_4^{i-1}$ except that, instead of committing to a share of $m||r$, all the puzzles up until the $i$-th of the $3\lambda$ puzzles sent in Step 1. commit to $0^{2\lambda}$. $\mathcal{H}_4^i$ is formally defined as follows:

```
1. sample randomness $r \leftarrow \mathbb{F} \setminus \{0\}$; $\forall \ell \in [\lambda]$, compute the first message
   $\sigma_\ell[1]$ of $\mathsf{SHCom}$ and sample $v_{\ell,1}, v_{\ell,2} \leftarrow \{0,1\}^{2\lambda}$ and compute $v_{\ell,3} =$
   $v_{\ell,1} \oplus v_{\ell,2} \oplus (m||r)$.
   $\forall k \in [3\lambda]$:
   • $u = \lceil \frac{k}{3} \rceil$, $\delta = (k-1) \mod 3 + 1$, sample randomness $\beta_{u,\delta}$
   • if $k \le i$, $z_{u,\delta} = \mathsf{TLP}^{\mathsf{qpv}}.\mathsf{Gen}(1^\lambda, t, 0^{2\lambda}; \beta_{u,\delta})$
   • else $z_{u,\delta} = \mathsf{TLP}^{\mathsf{qpv}}.\mathsf{Gen}(1^\lambda, t, v_{u,\delta}; \beta_{u,\delta})$
   **send** $(\sigma_u[1])_{u \in [\lambda]}$ and $(z_{u,\delta})_{u \in [\lambda], \delta \in [3]}$ to $\mathsf{R}_\mathsf{c}^*$.
⋮
```

$\mathcal{H}_5^i$ : This hybrid is parameterized by $i \in \{0, \ldots, 3\lambda\}$. We define $\mathcal{H}_5^0 := \mathcal{H}_3$. For $i \ge 1$, $\mathcal{H}_5^i$ is identical to $\mathcal{H}_5^{i-1}$ except that , instead of committing to an MPC view, all the commitments up until the $i$-th of the $3\lambda$ commitments sent in Step 4. commit to $0^{2\lambda}$. $\mathcal{H}_5^i$ is formally defined as follows:

```
⋮
4. $\forall k \in [3\lambda]$:
   • $u = \lceil \frac{k}{3} \rceil$, $\delta = (k-1) \mod 3 + 1$
   • if $k \le i$, $(c_{u,\delta}, d_{u,\delta}) \leftarrow \mathsf{Com}(1^\lambda, 0^\ell)$;
   • else $(c_{u,\delta}, d_{u,\delta}) \leftarrow \mathsf{Com}(1^\lambda, \mathsf{view}_{u,\delta})$
   **send** $(c_{u,\delta})_{u \in [\lambda], \delta \in [3]}$ and $\gamma := r\alpha + m$ to $\mathsf{R}_\mathsf{c}^*$.
```

$\vdots$

$\mathcal{H}_6$ This hybrid is identical to $\mathcal{H}_5^{3\lambda}$, but $\gamma$ is sampled uniformly at random.

Notice that $\mathcal{H}_6$ is identical to $\mathsf{Sim}$. We now introduce the following claims.

**Claim 6.** *If* $\mathsf{SHCom}$ *is computationally binding, then* $\mathcal{H}_1$ *is statistically indistinguishable from* $\mathcal{H}_0$*.*

**Claim 7.** *If* $\mathsf{SBCom}$ *is computationally hiding, then for every* $i \in [\lambda]$*,* $\mathcal{H}_2^i$ *is computationally indistinguishable from* $\mathcal{H}_2^{i-1}$*.*

**Claim 8.** *If* $\Pi_\phi$ *is perfectly 2-private, then* $\mathcal{H}_3$ *is identically distributed to* $\mathcal{H}_2^\lambda$*.*

**Claim 9.** *If* $\mathsf{TLP}^{\mathsf{qpv}}$ *is hard, then for every* $i \in [3\lambda]$*,* $\mathcal{H}_4^i$ *and* $\mathcal{H}_4^{i-1}$ *are indistinguishable for every poly-size distinguisher with depth upper-bounded by* $t^\epsilon$*.*

**Claim 10.** *If* $\mathsf{SBCom}$ *is computationally hiding, then for every* $i \in [3\lambda]$*,* $\mathcal{H}_5^i$ *is computationally indistinguishable from* $\mathcal{H}_5^{i-1}$*.*

**Claim 11.** $\mathcal{H}_6$ *is perfectly indistinguishable from* $\mathcal{H}_5^{3\lambda}$*.*

*Proof (of Claim 6).* Conditioned on not outputting $\mathsf{ambiguous}$, the output distribution of $\mathcal{H}_1$ is identical to $\mathcal{H}_0$. It remains to show that $\mathcal{H}_1$ outputs $\mathsf{ambiguous}$ only with negligible probability. Assuming that there exists an infinite series of inputs that makes $\mathcal{H}_1$ output $\mathsf{ambiguous}$ with non-negligible probability, one can easily construct an adversary $\mathsf{A}$ for the binding of $\mathsf{SHCom}$. $\mathsf{A}$ runs $\mathcal{H}_1$ on such an input and looks for $i \in \lambda$ s.t.$(a_i, b_i) \neq (a'_i, b'_i)$ and both pairs have a valid decommitment sent by $\mathsf{R_c}^*$ when interacting with $\mathcal{H}_1$. $\mathsf{A}$ simply re-uses such decommitments in the binding game. The only subtlety is that $\mathcal{H}_1$ runs in expected polynomial time, whereas $\mathsf{A}$ must run in strict polynomial time. Nevertheless, this issue can be addressed by simply truncating $\mathcal{H}_1$ to twice its expected running time. By Markov's inequality, this adjustment decreases the attack's success probability against the binding of $\mathsf{SHCom}$ of at most 1/2, which remains non-negligible. □

*Proof (of Claim 9).* If there exists a polynomial-size distinguisher $\mathsf{D}_\lambda$, whose depth is bounded by $t^\epsilon$, distinguishing with noticeable probability $\mathcal{H}_4^i$ from $\mathcal{H}_4^{i-1}$, then we can use $\mathsf{D}$ to build a bounded-depth adversary $\mathsf{A}$ against the hardness property of $\mathsf{TLP}^{\mathsf{qpv}}$. $\mathsf{A}$ runs the instructions of $\mathcal{H}_4^{i-1}$ with one change. That is, when $\mathsf{A}$ has to compute the $i$-th puzzle at Step 1. it plays in the hardness game of $\mathsf{TLP}^{\mathsf{qpv}}$ by sending $s_0$ equal to a share of $m||r$ and $s_1 = 0^{2\lambda}$ and uses the puzzle $z$ that it gets back from the challenger as the $i$-th puzzle of the simulation. When the simulation concludes, then $\mathsf{A}$ invokes $\mathsf{D}$ on the output generated by the simulator, and outputs whatever $\mathsf{D}$ outputs. Notice that when $b = 0$ in the hardness game, $\mathsf{A}$ perfectly simulates $\mathcal{H}_4^{i-1}$, otherwise it perfectly simulates $\mathcal{H}_4^i$. Therefore, $\mathsf{A}$ wins the hardness game with the same probability with which $\mathsf{D}$ distinguishes between $\mathcal{H}_4^i$ and $\mathcal{H}_4^{i-1}$. The only subtlety is that $\mathcal{H}_4^{i-1}$ runs in expected polynomial time essentially independent of $t$, whereas $\mathsf{A}$ must run in strict polynomial time essentially independent of $t$. Nevertheless, this issue can be addressed by simply truncating the running time of $\mathcal{H}_4^{i-1}$. □

The proof of Claims 7 and 10 are analogous to the one of 2, with the only difference that we have to truncate the running time of $\mathcal{H}_2^{i-1}$. The proofs of Claims 8 and 11 are identical to the ones of 4 and 5 respectively.

The proof is concluded by observing that it simply follows from the proofs of Claims 6 -11, since all of the hybrids are indistinguishable by a poly-size distinguisher whose depth is bounded by $t^{\epsilon}$. □

**Lemma 7.** $\mathsf{HRTC}$ *is sound if* $\mathsf{TLP}^{\mathsf{qpv}}$ *is a quasi publicly verifiable TLP,* $\mathsf{SBCom}$ *is a non-interactive statistically biding commitment scheme,* $\mathsf{SHCom}$ *is a two-round statistically hiding commitment scheme, and* $\Pi_{\phi}$ *is a* 3*-party secure MPC.*

*Proof (of Lemma 7).* The only difference with Lemma 2 is the presence of statistically hiding commitments going from the receiver to the malicious sender. Since such commitments statistically convey no information about the receiver's challenges, soundness is argued similarly. □

## 3.5 A More Efficient Force-Open

By carefully modifying $\mathsf{FOpen}$ and $\mathsf{FVerify}$, we can get a much more efficient force-open phase of our TC constructions (Sects. 3.3 and 3.4). In the modified TC scheme, commitments can be force opened by solving $\log(\lambda)$, instead of $\lambda$, puzzles. The intuition is that after a valid commit phase, the value $\gamma$ can be used to check if, after having solved the puzzle associated with a certain repetition $i \in [\lambda]$, the recovered message is the right one. By right we mean that the sender did not cheat in the $i$-th repetition and thus the recovered message coincides with the one committed in (the vast majority of) the other repetitions.

In a nutshell, $\mathsf{FOpen}$ force-opens random repetitions until it finds one where $\gamma$ is consistent with the recovered message $m$ and the proof $\pi$ given in output by $\mathsf{TLP}^{\mathsf{qpv}}.\mathsf{Solve}$ is accepting. Since, with overwhelming probability, there are at most $\log^2(\lambda)$ bad repetitions (i.e., where $\gamma$ is not consistent with $m$ or $\pi$ is not accepting) after a valid commit phase (see the proof of Lemma 2), the probability of only finding bad repetitions with more than $\log(\lambda)$ tries is negligible. Whenever $\mathsf{FOpen}$ finds a good repetition, it outputs the repetition's index $i$ and the recovered message $m$, along with the share and the proof given in output by $\mathsf{TLP}^{\mathsf{qpv}}.\mathsf{Solve}$. Then, $\mathsf{FVerify}$ just verifies that: (1) $m$ coincides with the one reconstructed from the shares of the $i$-th repetition, (2) $m$ is consistent with $\gamma$, and (3) the proof is accepting w.r.t. the unopened puzzle of the $i$-th repetition.

However, this efficiency improvement comes at the price of only satisfying *weak* public verifiability. Recall that in the weak public verifiability game (Def. 3), the commitment is not directly provided by the adversary but it is the result of a commit phase run with the honest receiver. Intuitively, we can only get weak public verifiability with this modification because the commitment (transcript) $\mathsf{com}$ could be simulated by an adversary not interacting with a receiver at all. Therefore, every repetition could possibly contain a different message while still being consistent w.r.t. $\gamma$. The modified $\mathsf{FOpen}$ and $\mathsf{FVerify}$ are in Fig. 5.

```
$(m, \pi) \leftarrow \mathsf{FOpen}(1^\lambda, 1^t, \mathsf{com})$: Perform the following steps:
    1. Set $I := [\lambda]$ and parse $\mathsf{com}$ as in (2).
    2. Sample random $i \leftarrow I$. Let $k \in [3]$ be s.t. $a_i, b_i \neq k$ and compute
       $(v_{i,k}, \pi_{i,k}) \leftarrow \mathsf{TLP}^{\mathsf{qpv}}.\mathsf{Solve}(1^\lambda, 1^t, z_{i,k})$
    3. Check if $v = v_{i,a_i} \oplus v_{i,b_i} \oplus v_{i,k}$ can be parsed as $m||r$ so that $\gamma = \alpha r + m$
    4. Check if $\mathsf{TLP}^{\mathsf{qpv}}.\mathsf{Verify}(1^\lambda, t, z_{i,k}, v_{i,k}, \pi_{i,k}) = 1$
    5. If all the checks are successful output $m$ and proof $\pi = (v_{i,k}, \pi_{i,k}, i, k)$.
       Otherwise, update $I = I \setminus \{i\}$. If $|I| \geq \lambda - \log(\lambda)$, go back to Step 2,
       otherwise output $(\perp, \perp)$.
$b := \mathsf{FVerify}(1^\lambda, t, \mathsf{com}, m, \pi)$: Set $b = 1$ iff all the following checks are successful:
    1. Parse $\mathsf{com}$ as in (2) and $\pi$ as $(v_{i,k}, \pi_{i,k}, i, k)$
    2. Check that $\mathsf{TLP}^{\mathsf{qpv}}.\mathsf{Verify}(1^\lambda, t, z_{i,k}, v_{i,k}, \pi_{i,k}) = 1$
    3. Parse $v_{i,a_i} \oplus v_{i,b_i} \oplus v_{i,k}$ as $m||r$ with $|m| = |r|$ and check that $\gamma = \alpha r + m$.
```

**Fig. 5.** Modified $\mathsf{FOpen}$ and $\mathsf{FVerify}$ algorithms.

**Theorem 4.** *Let* $\mathsf{TLP}^{\mathsf{qpv}} = (\mathsf{Gen}, \mathsf{Solve}, \mathsf{Verify})$ *be a quasi publicly verifiable TLP (Def. 8),* $\mathsf{SBCom} = (\mathsf{Com}, \mathsf{Dec})$ *a non-interactive statistically binding and computationally hiding commitment scheme,* $\mathsf{SHCom}$ *a two-round statistically hiding and computationally binding commitment scheme, and* $\Pi_\phi$ *a* 3*-party secure MPC for functionality* $\phi$ *defined in* (1)*, then the scheme* $\mathsf{HRTC}$ *from Sect. 3.3 (respectively* $\mathsf{TC}$ *from Sect. 3.4) where the algorithms* $\mathsf{FOpen}$ *and* $\mathsf{FVerify}$ *are modified as reported in Fig. 5 is honest-verifier (respectively* $t$*-hiding) timed commitment scheme with weak public verifiability.*

To prove the above theorem, we only need to argue soundness and weak public verifiability of the modified schemes. Indeed, we do not need to prove (honest-receiver) $t$-hiding and binding again as these property only involve the commit and open phases, which are left unaltered.

**Lemma 8.** $\mathsf{C} \in \{\mathsf{HRTC}, \mathsf{TC}\}$ *where the algorithms* $\mathsf{FOpen}$ *and* $\mathsf{FVerify}$ *are modified as reported in Fig. 5 is sound if* $\mathsf{TLP}^{\mathsf{qpv}}$ *is a QPV-TLP,* $\mathsf{SBCom}$ *is a non-interactive statistically biding commitment scheme,* $\Pi_\phi$ *is a* 3*-party secure MPC.*

*Proof (of Lemma 8).* Let us call a repetition $i \in [\lambda]$ $\mathsf{bad}$ if $\phi(\alpha, \gamma, v_{i,1}, v_{i,2}, v_{i,3}) = 0$ or $\exists\ u \in [3]$ such that the TLP $z_{i,u}$ is malformed. Recall that conditioned on $\mathsf{com} \neq \perp$, the probability that a repetition is $\mathsf{bad}$ at most $\frac{1}{3}$ (see the proof of Lemma 2). As a result, if the commit phase is successful, i.e., $\mathsf{com} \neq \perp$, the probability that at least $\log^2(\lambda)$ repetitions are $\mathsf{bad}$ is at most $(\frac{1}{3})^{\log^2(\lambda)} < (\frac{1}{2})^{\log^2(\lambda)} = (\frac{1}{\lambda})^{\log(\lambda)}$, which is negligible. Therefore, the probability that all $\log(\lambda)$ samples of $\mathsf{FOpen}$ are $\mathsf{bad}$ is at most $(\frac{\log^2(\lambda)}{\lambda})^{\log(\lambda)}$, which is negligible[7].

Let $S \subseteq [\lambda]$ be the set containing the indices of the repetitions that are not $\mathsf{bad}$, from the above discussion it follows that $|S| \geq \lambda - \log^2(\lambda)$ except with

[7] Notice that eliminating a bad repetition from the set of repetitions that may be forced-open only further reduces the probability of encountering a bad repetition. Indeed, $\log^2(\lambda) < \lambda$ and $\frac{n}{k} > \frac{n-i}{k-i}$ for any $n, k > 1, n < k$, and $i \geq 1$.

negligible probability. For every $i \in S$ where $v_{i,1} \oplus v_{i,2} \oplus v_{i,3} = m_i || r_i$ for $|m_i| = r_i$, it holds that $\gamma = \alpha r_i + m_i$. By the Schwartz-Zippel lemma, with overwhelming probability over the choice of $\alpha$, there must exist $(m, r)$ such that $m_i = m$ and $r_i = r$ for all $i \in S$. Recall that $\forall i \in S$ and $k \in [3]$ s.t. $a_i, b_i \neq k$ we have that $z_{i,k}$ is well-formed. The correctness and the completeness of $\mathsf{TLP}^{\mathsf{qpv}}$ guarantee that whenever a puzzle is correctly generated, the solution/proof pair output by $\mathsf{TLP}^{\mathsf{qpv}}.\mathsf{Solve}$ will be accepting and will contain the committed value. Recall that a successful honest open phase outputs the same message $m'$ in more than $\lambda/2$ repetitions. Thus, since $|S| \geq \lambda - \log^2(\lambda)$, except with negligible probability, $m'$ coincides with the message $m$ output by $\mathsf{FOpen}$ (recall that a repetition can only be opened in one way since $\mathsf{TLP}^{\mathsf{qpv}}$ is perfectly binding). Additionally, $\mathsf{FVerify}$ outputs 1 on input the message/proof pair returned by $\mathsf{FOpen}$. $\square$

**Lemma 9.** $\mathsf{C} \in \{\mathsf{HRTC}, \mathsf{TC}\}$ *where the algorithms* $\mathsf{FOpen}$ *and* $\mathsf{FVerify}$ *are modified as reported in Fig. 5 is weak publicly verifiable if* $\mathsf{TLP}^{\mathsf{qpv}}$ *is a QPV-TLP.*

*Proof (of Lemma 9).* Assume $\mathsf{A}$ outputs two accepting proof/message pairs $(m, \pi = (v_{i,k}, \pi_{i,k}, i, k))$, and $(m', \pi' = (v_{i',k'}, \pi_{i',k'}, i', k'))$ with $m \neq m'$ for the same $\mathsf{com} = (z_{i,1}, z_{i,2}, z_{i,3}, \alpha, \gamma, a_i, b_i, v_{i,a_i}, \beta_{i,a_i}, v_{i,b_i}, \beta_{i,b_i})_{i \in [\lambda]}$. If $i = i'$, then $m = m'$ since, due to the perfect soundness of $\mathsf{TLP}^{\mathsf{qpv}}$, there exists only one possible share $v_{i,k}$ for which an accepting proof $\pi_{i,k}$ can be provided for the puzzle $z_{i,k}$. Together with $v_{i,a_i}, v_{i,b_i}$, the share $v_{i,k}$ fixes the message $m$. Let us assume $i \neq i'$. As already discussed above, due to the perfect soundness of $\mathsf{TLP}^{\mathsf{qpv}}$, within each repetition $\ell \in [\lambda]$ there exists only a possible share $v_{\ell,k}$ for which an accepting proof $\pi_{\ell,k}$ can be provided for the puzzle $z_{\ell,k}$. Hence, the malicious sender is bound to a single $v_{\ell,k}$, which results in a single $v_\ell = v_{\ell,a_\ell} \oplus v_{\ell,b_\ell} \oplus v_{\ell,k}$ within each repetition. Thus at the end of the commit phase, the sender is bound to a single $v_\ell$ in each repetition. By the Schwartz-Zippel lemma we know that, with overwhelming probability over the choices of $\alpha$, for every $v_\ell = m_\ell || r_\ell$ such that $\gamma = \alpha r_\ell + m_\ell$ it must be that all $m_\ell$ are such that $m_\ell = m$. Hence, with overwhelming probability, the two force opening proofs provided by the adversary will point to two repetition indices $i$ and $i'$ which contain $v_i = m || r$ and $v_{i'} = m' || r'$ such that $m = m'$. $\square$

**Acknowledgements.** We thank Peter Chvojka and Ivan Visconti for insightful comments and discussions. We thank the anonymous reviewers of TCC 2024 for their feedback. This result is part of projects that have received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation program under project PICOCRYPT (grant agreement No. 101001283), and from the Spanish Government under projects PRODIGY (TED2021-132464B-I00) and ESPADA (PID2022-142290OB-I00). The last two projects are co-funded by European Union EIE, and NextGenerationEU/PRTR funds.

## References

AMZ24. Agrawalr, S., Malavolta, G., Zhang, T.: Time-lock puzzles from lattices. In: Reyzin, L., Stebila, D. (eds.) Advances in Cryptology – CRYPTO 2024. CRYPTO 2024. LNCS, vol. 14922. Springer, Cham (2024). https://doi.org/10.1007/978-3-031-68382-4_13

BDD+21. Baum, C., David, B., Dowsley, R., Nielsen, J.B., Oechsner, S.: TARDIS: a foundation of time-lock puzzles in UC. In: Canteaut, A., Standaert, F.-X. (eds.) EUROCRYPT 2021. LNCS, vol. 12698, pp. 429–459. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-77883-5_15

BDD+23. Carsten Baum, Bernardo David, Rafael Dowsley, Ravi Kishore, Jesper Buus Nielsen, and Sabine Oechsner. CRAFT: Composable randomness beacons and output-independent abort MPC from time. In Alexandra Boldyreva and Vladimir Kolesnikov, editors, *PKC 2023, Part I*, volume 13940 of *LNCS*, pages 439–470, Atlanta, GA, USA, May 7–10, 2023. Springer, Heidelberg, Germany

BGI+01. Barak, B., et al.: On the (Im)possibility of obfuscating programs. In: Kilian, J. (ed.) CRYPTO 2001. LNCS, vol. 2139, pp. 1–18. Springer, Heidelberg (2001). https://doi.org/10.1007/3-540-44647-8_1

BGJ+16. Bitansky, N., Goldwasser, S., Jain, A., Paneth, O., Vaikuntanathan, V., Waters, B.: Time-lock puzzles from randomized encodings. In: Sudan, M. (eds.) ITCS 2016, 14–16 January, pp. 345–356. ACM, Cambridge, MA, USA (2016)

BN00. Boneh, D., Naor, M.: Timed commitments. In: Bellare, M. (ed.) CRYPTO 2000. LNCS, vol. 1880, pp. 236–254. Springer, Heidelberg (2000). https://doi.org/10.1007/3-540-44598-6_15

Can01. Canetti, R.: Universally composable security: a new paradigm for cryptographic protocols. In: 42nd FOCS, 14–17 October, pp. 136–145. IEEE Computer Society Press, Las Vegas, NV, USA (2001)

CDMW09. Choi, S.G., Dachman-Soled, D., Malkin, T., Wee, H.: Simple, black-box constructions of adaptively secure protocols. In: Reingold, O. (ed.) TCC 2009. LNCS, vol. 5444, pp. 387–402. Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-642-00457-5_23

CGJ+23. Choudhuri, A.R., Garg, S., Jain, A., Jin, Z., Zhang, J.: Correlation intractability and SNARGs from sub-exponential DDH. In: Handschuh, H., Lysyanskaya, A. (eds) Advances in Cryptology – CRYPTO 2023. CRYPTO 2023. LNCS, vol. 14084. Springer, Cham (2023). https://doi.org/10.1007/978-3-031-38551-3_20

CJ23. Chvojka, P., Jager, T.: Simple, fast, efficient, and tightly-secure non-malleable non-interactive timed commitments. In: Boldyreva, A., Kolesnikov, V. (eds.) Public-Key Cryptography – PKC 2023. PKC 2023. LNCS, vol. 13940. Springer, Cham (2023). https://doi.org/10.1007/978-3-031-31368-4_18

CJJ22. Choudhuri, A.R., Jain, A., Jin, Z.: SNARGs for $\mathcal{P}$ from LWE. In: 62nd FOCS, 7–10 February, Denver, CO, USA, pp. 68–79. IEEE Computer Society Press (2022)

CLP20. Chatterjee, R., Liang, X., Pandey, O.: Improved black-box constructions of composable secure computation. Cryptology ePrint Archive, Report 2020/494 (2020). https://eprint.iacr.org/2020/494

COS22. Ciampi, M., Orsini, E., Siniscalchi, L. Four-round black-box non-malleable schemes from one-way permutations. In: Kiltz, E., Vaikuntanathan, V. (eds.) Theory of Cryptography. TCC 2022. LNCS, vol. 13748. Springer, Cham (2022). https://doi.org/10.1007/978-3-031-22365-5_11

FKPS21. Freitag, C., Komargodski, I., Pass, R., Sirkin, N.: Non-malleable time-lock puzzles and applications. In: Nissim, K., Waters, B. (eds.) TCC 2021. LNCS, vol. 13044, pp. 447–479. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-90456-2_15

GK96. Goldreich, O., Kahan, A.: How to construct constant-round zero-knowledge proof systems for NP. J. Cryptol. **9**(3), 167–189 (1996). https://doi.org/10.1007/BF00208001

GLOV12. Goyal, V., Lee, C.-K., Ostrovsky, R., Visconti, I.: Constructing non-malleable commitments: a black-box approach. In: 53rd FOCS, 20–23 October, pp. 51–60, New Brunswick, NJ, USA. IEEE Computer Society Press (2012)

GMW87. Goldreich, O., Micali, S., Wigderson, A.: How to play any mental game or A completeness theorem for protocols with honest majority. In: Aho, A. (ed.) 19th ACM STOC, 25–27 May, pp. 218–229. ACM Press, New York City (1987)

Gol01. Goldreich, O.: The Foundations of Cryptography - Volume 1: Basic Techniques. Cambridge University Press (2001)

HJKS22. Hulett, J., Jawale, R., Khurana, D., Srinivasan, A.: SNARGs for P from Sub-exponential DDH and QR. In: Dunkelman, O., Dziembowski, S. (eds.) Advances in Cryptology – EUROCRYPT 2022. EUROCRYPT 2022. LNCS, vol. 13276. Springer, Cham (2022). https://doi.org/10.1007/978-3-031-07085-3_18

HM96. Halevi, S., Micali, S.: Practical and provably-secure commitment schemes from collision-free hashing. In: Koblitz, N. (ed.) CRYPTO 1996. LNCS, vol. 1109, pp. 201–215. Springer, Heidelberg (1996). https://doi.org/10.1007/3-540-68697-5_16

HV16. Hazay, C., Venkitasubramaniam, M.: On the power of secure two-party computation. In: Robshaw, M., Katz, J. (eds.) CRYPTO 2016. LNCS, vol. 9815, pp. 397–429. Springer, Heidelberg (2016). https://doi.org/10.1007/978-3-662-53008-5_14

IKLP06. Ishai, Y., Kushilevitz, E., Lindell, Y., Petrank, F.: Black-box constructions for secure computation. In: Kleinberg, J.M (eds.) 38th ACM STOC, 21–23 May, Seattle, WA, USA, pp. 99–108. ACM Press (2006)

IKOS07. Ishai, Y., Kushilevitz, E., Ostrovsky, R., Sahai, A.: Zero-knowledge from secure multiparty computation. In: Johnson, D.S., Feige, U. (eds.) 39th ACM STOC, 11–13 June, San Diego, CA, USA, pp. 21–30. ACM Press (2007)

JMRR21. Jaques, S., Montgomery, H., Rosie, R., Roy, A.: Time-release cryptography from minimal circuit assumptions. In: Adhikari, A., Küsters, R., Preneel, B. (eds.) INDOCRYPT 2021. LNCS, vol. 13143, pp. 584–606. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-92518-5_26

Kiy14. Kiyoshima, S.: Round-efficient black-box construction of composable multi-party computation. In: Garay, J.A., Gennaro, R. (eds.) CRYPTO 2014. LNCS, vol. 8617, pp. 351–368. Springer, Heidelberg (2014). https://doi.org/10.1007/978-3-662-44381-1_20

Kiy20. Kiyoshima, S.: Round-optimal black-box commit-and-prove with succinct communication. In: Micciancio, D., Ristenpart, T. (eds.) CRYPTO 2020. LNCS, vol. 12171, pp. 533–561. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-56880-1_19

Kiy23. Kiyoshima, S.: Holographic SNARGs for P and Batch-NP from (polynomially hard) learning with errors. In: Rothblum, G., Wee, H. (eds.) Theory of Cryptography. TCC 2023. LNCS, vol. 14371. Springer, Cham (2023). https://doi.org/10.1007/978-3-031-48621-0_12

KLX20. Katz, J., Loss, J., Xu, J.: On the security of time-lock puzzles and timed commitments. In: Pass, R., Pietrzak, K. (eds.) TCC 2020. LNCS, vol. 12552, pp. 390–413. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-64381-2_14

KOS18. Khurana, D., Ostrovsky, R., Srinivasan, A.: Round optimal black-box "Commit-and-Prove". In: Beimel, A., Dziembowski, S. (eds.) TCC 2018. LNCS, vol. 11239, pp. 286–313. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-03807-6_11

Lin13. Lindell, Y.: A note on constant-round zero-knowledge proofs of knowledge. J. Cryptol. **26**(4), 638–654 (2013)

Lin16. Lindell, Y.: How to simulate it - a tutorial on the simulation proof technique. Cryptology ePrint Archive, Paper 2016/046 (2016). https://eprint.iacr.org/2016/046

LM23. Lai, R.W.F., Malavolta, G.: Lattice-Based Timed Cryptography. In: Handschuh, H., Lysyanskaya, A. (eds.) Advances in Cryptology – CRYPTO 2023. CRYPTO 2023. LNCS, vol. 14085. Springer, Cham (2023). https://doi.org/10.1007/978-3-031-38554-4_25

LP12. Lin, H., Pass, R.: Black-Box constructions of composable protocols without set-up. In: Safavi-Naini, R., Canetti, R. (eds.) CRYPTO 2012. LNCS, vol. 7417, pp. 461–478. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-32009-5_27

LPS17. Lin, H., Pass, R., Soni, P.: Two-round and non-interactive concurrent non-malleable commitments from time-lock puzzles. In Umans, C (ed.) 58th FOCS, 15–17 October, Berkeley, CA, USA, pp. 576–587. IEEE Computer Society Press (2017)

Pie19. Pietrzak, K.: Simple verifiable delay functions. In: Blum, A (ed.) ITCS 2019, 10–12 January, San Diego, CA, USA, vol. 124, pp. 60:1–60:15. LIPIcs (2019)

PW09. Pass, R., Wee, H.: Black-box constructions of two-party protocols from one-way functions. In: Reingold, O. (ed.) TCC 2009. LNCS, vol. 5444, pp. 403–418. Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-642-00457-5_24

RSW96. Rivest, R.L., Shamir, A., Wagner, D.A.: Time-lock puzzles and timed-release crypto. Technical report, USA (1996)

SW14. Sahai, A., Waters, B.: How to use indistinguishability obfuscation: deniable encryption, and more. In: Shmoys, D.B (ed.) 46th ACM STOC, 31 May–3 June, New York, NY, USA, pp. 475–484. ACM Press (2014)

TCLM21. Thyagarajan, D.A.K., Castagnos, G., Laguillaumie, F., Malavolta, G.: Efficient CCA timed commitments in class groups. In: Vigna. G., Shi, E (eds.) ACM CCS 2021, pp. 2663–2684, Virtual Event, Republic of Korea, 15–19 November. ACM Press (2021)

Wes19. Wesolowski, B.: Efficient verifiable delay functions. In: Ishai, Y., Rijmen, V. (eds.) EUROCRYPT 2019. LNCS, vol. 11478, pp. 379–407. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-17659-4_13

WW24. Waters, B., Wu, D.J.: Adaptively-sound succinct arguments for NP from indistinguishability obfuscation. In: Mohar, B., Shinkar, I., O'Donnell, R. (eds.) Proceedings of the 56th Annual ACM Symposium on Theory of Computing, STOC 2024, 24-28 June, Vancouver, BC, Canada, pp. 387–398. ACM (2024)

ZMM+20. Zhang, F., Maram, D., Malvai, H., Goldfeder, S., Juels, A.: DECO: liberating web data using decentralized oracles for TLS. In: Ligatti, J., Ou, X., Katz, J., Vigna, G. (eds.) ACM CCS 2020, 9–13 November, pp. 1919–1938, Virtual Event, USA. ACM Press (2020)

# Author Index

**A**
Abusalah, Hamza 460
Ananth, Prabhanjan 257
Avitabile, Gennaro 460

**B**
Brakerski, Zvika 91

**C**
Çakan, Alper 225
Champion, Jeffrey 156
Coladangelo, Andrea 289

**D**
Don, Jelle 347
Döttling, Nico 431
Dujmovic, Jesko 431

**F**
Fehr, Serge 347

**G**
Goyal, Rishab 3
Goyal, Vipul 193, 225

**H**
Hajiabadi, Mohammad 318
Hanaoka, Goichiro 124
Hu, Zihan 257
Huang, Yu-Hsuan 347
Huang, Zikuan 257

**J**
Joux, Antoine 431

**K**
Katsumata, Shuichi 124
Kimura, Kei 124

**L**
Langrehr, Roman 318
Liao, Jyun-Jie 347
Liu, Jiahui 400

**M**
Malavolta, Giulio 193
Medina, Stav 91
Mutreja, Saachi 289

**O**
O'Neill, Adam 318

**P**
Pointcheval, David 31

**R**
Raizes, Justin 193
Rotem, Lior 371

**S**
Schädlich, Robert 31
Segev, Gil 371
Struck, Patrick 347

**T**
Takemure, Kaoru 124

**W**
Wang, Mingyuan 318
Waters, Brent 65
Wichs, Daniel 65
Wu, David J. 156

**Y**
Yadugiri, Saikumar 3
Yamada, Shota 124
Yogev, Eylon 371

**Z**
Zhandry, Mark 400