7th Conference on Information-Theoretic Cryptography

ITC 2026, August 15–16, 2026, University of California, Santa Barbara, CA, USA

Edited by

Yevgeniy Dodis

LIPIcs – Vol. 385 – ITC 2026 www.dagstuhl.de/lipics

Editors

Yevgeniy Dodis
New York University, NY, USA
dodis@cs.nyu.edu

ACM Classification 2012
Mathematics of computing → Information theory; Theory of computation → Computational complexity and cryptography; Security and privacy → Cryptography

ISBN 978-3-95977-426-0

Published online and open access by
Schloss Dagstuhl – Leibniz-Zentrum für Informatik GmbH, Dagstuhl Publishing, Saarbrücken/Wadern, Germany. Online available at https://www.dagstuhl.de/dagpub/978-3-95977-426-0.

Publication date
August, 2026

Bibliographic information published by the Deutsche Nationalbibliothek
The Deutsche Nationalbibliothek lists all publications of this volume in the Deutsche Nationalbibliografie; detailed bibliographic data are available in the Internet at https://portal.dnb.de.

Digital Object Identifier: 10.4230/LIPIcs.ITC.2026.0

ISBN 978-3-95977-426-0 **ISSN 1868-8969** **https://www.dagstuhl.de/lipics**

LIPIcs – Leibniz International Proceedings in Informatics

LIPIcs is a series of high-quality conference proceedings across all fields in informatics. LIPIcs volumes are published according to the principle of Open Access, i.e., they are available online and free of charge.

ISSN 1868-8969

https://www.dagstuhl.de/lipics

Contents

Papers

7th Conference on Information-Theoretic Cryptography (ITC 2026).
Editor: Yevgeniy Dodis
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

Preface

The seventh Conference on Information-Theoretic Cryptography (ITC 2026) took place from August 15–16, 2026, at the University of California, Santa Barbara, USA. The general co-chairs were Prabhanjan Ananth, Daniel Collins and Eli Goldin, and the program chair was Yevgeniy Dodis. As in previous editions, the conference was held in cooperation with the International Association for Cryptologic Research (IACR).

In its seventh year, ITC continued its mission of uniting the cryptography and information theory communities, and advancing research in all aspects of information-theoretic techniques for cryptography and security.

Like last year, we received a total of 18 submissions, maintaining a high standard of quality. Following our tradition, we facilitated interactive and anonymous discussions with the authors to clarify technical issues. With the assistance of external reviewers, the program committee selected 12 papers for presentation (like last year). The proceedings contain the revised versions of these papers. The revisions were not reviewed, and the authors bear full responsibility for the content.

Continuing the successful format from previous years, the conference featured a publication track, a spotlight track and a highlights track. The publication track consisted of talks selected among the conference submissions by the program committee for publication and presentation. The spotlight track consisted of invited talks (not published in the proceedings) that highlight the most exciting recent advances in information-theoretic cryptography. The highlights track comprised ITC-related papers that have appeared in the last years including at venues like STOC 2026 and Eurocrypt 2026; these talks were mostly given by students and postdocs.

We are deeply grateful to everyone who contributed to the success of the 7th ITC conference. Our sincere thanks go out to the authors who submitted their papers. We extend our heartfelt thanks to the PC members and external reviewers for their dedicated efforts in providing thorough reviews, insightful discussions, and expert opinions. We are deeply indebted to the steering committee, particularly co-chairs Benny Applebaum and Elette Boyle, for their invaluable guidance. Special thanks are also due to the previous PC chairs for sharing their experience and providing answers to numerous questions. Thank you Prabhanjan Anath for your service as a general co-chair for another successful year. I would be remiss not to mention Daniel Collins and Eli Goldin whose significant efforts on various fronts behind the scenes made this conference possible. Lastly, we extend our gratitude to all the invited speakers, presenting authors, and participants who devoted their time and energy to ensuring the success of this conference.

Yevgeniy Dodis

7th Conference on Information-Theoretic Cryptography (ITC 2026).
Editor: Yevgeniy Dodis
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

Steering Committee

- Benny Applebaum (Co-Chair, Tel-Aviv University)
- Elette Boyle (Co-Chair, NTT Research and Reichman University)
- Ivan Damgård (Aarhus University)
- Yevgeniy Dodis (New York University)
- Yuval Ishai (Technion and AWS)
- Ueli Maurer (ETH Zurich)
- Kobbi Nissim (Georgetown)
- Krzysztof Pietrzak (IST Austria)
- Manoj Prabhakaran (IIT Bombay)
- Adam Smith (Boston University)
- Yael Tauman Kalai (MIT)
- Stefano Tessaro (University of Washington)
- Vinod Vaikuntanathan (MIT)
- Hoeteck Wee (NTT Research)
- Daniel Wichs (Northeastern University)
- Mary Wootters (Stanford)
- Chaoping Xing (Shanghai Jiao Tong University)
- Moti Yung (Google)

7th Conference on Information-Theoretic Cryptography (ITC 2026).
Editor: Yevgeniy Dodis

Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

Organization

General chairs

- Prabhanjan Ananth (University of California, Santa Barbara)
- Daniel Collins (New York University and Hebrew University)
- Eli Goldin (New York University)

Program chair

- Yevgeniy Dodis (New York University)

Program committee

- Adam Smith (Boston University)
- Bhavana Kanukurthi (Indian Institute of Science)
- Daniel Collins (New York University and Hebrew University)
- Daniel Tschudi (Concordium)
- Eli Goldin (New York University)
- Eylon Yogev (Bar-Ilan University)
- Hemanta K. Maji (Purdue University)
- Itai Dinur (Ben-Gurion University)
- Lior Rotem (Hebrew University)
- Maciej Obremski (National University of Signapore)
- Miryam Huang (University of Southern California)
- Nico Döttling (Helmholtz Center for Information Security (CISPA))
- Nicolas Resch (University of Amsterdam)
- Noah Golowich (Microsoft Research)
- Noam Mazor (New York University)
- Qipeng Liu (University of California, San Diego)
- Rahul Ilango (Institute for Advanced Study)
- Ran Cohen (Reichman University)
- Roman Langrehr (University of Waterloo)
- Siyao Gao (New York University - Shanghai)
- Tomoyuki Morimae (University of Kyoto)
- Vassilis Zikas (Georgia Institute of Technology)
- Yifan Song (Tsinghua University)
- Zhengzhong Jin (Northeastern University)

External reviewers

Akshima, Liyan Chen, Pouyan Forghani, Hanjun Li, George Lu, Mohammad Mahmoody, Hai Nguyen, Jaspal Singh, Er-cheng Tang, Yu Wei, Zhiye Xie.

7th Conference on Information-Theoretic Cryptography (ITC 2026).
Editor: Yevgeniy Dodis

Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

Tighter Bounds for the Oblivious Bit-Fixing Inner Product Extractor on Biased Seeds

Jack Doerner
University of Virginia, Charlottesville, VA, USA

Lawrence Roy
Aarhus University, Denmark

Abstract

The *Inner Product Extractor* (IPE) of Impagliazzo, Levin, and Luby (STOC'89) takes a seed $h \in \mathbb{F}^\gamma$ and a source $x \in \{0,1\}^\gamma$ for some $\gamma \in \mathbb{N}$ and produces $\langle h, x \rangle$ with error $\varepsilon = \mathsf{SD}((\langle \mathcal{H}, \mathcal{X} \rangle, \mathcal{H}), (\mathcal{Y}, \mathcal{H}))$ such that

$$\varepsilon \leq \frac{1}{2}\sqrt{|\mathbb{F}|^\gamma / 2^{H_\infty(\mathcal{H})}}\ \sqrt{|\mathbb{F}| / 2^{H_\infty(\mathcal{X})}}$$

where $\mathcal{Y}$ is the uniform distribution over $\mathbb{F}$, and $\mathcal{H}$ and $\mathcal{X}$ are the independent but possibly *non-uniform* distributions from which h and x are drawn, respectively. In other words, the IPE's error grows with the square root of seed bias, at most. This square root arises because prior works bound the squared error using the 2-universality of the IPE. The analysis requires an even power of the error, and the IPE is not 4-universal.

Motivated by applications to multiparty computation, we revisit the problem of the IPE with biased seeds and prove far tighter bounds on the influence of seed bias by bypassing universal hashing. We first prove an *Elevated* General Leftover Hash Lemma, which yields an n^{th} root bound for functions that are *almost* n-universal. Bounding number of inputs on which the IPE is not 4-universal yields $\varepsilon = \mathsf{SD}((\langle \mathcal{H}, \mathcal{W} \rangle, \mathcal{H}), (\mathcal{Y}, \mathcal{H}))$ where

$$\varepsilon \lesssim \frac{2.1}{2}\left(|\mathbb{F}|^\gamma / 2^{H_\infty(\mathcal{H})}\right)^{\frac{1}{4}} \sqrt{|\mathbb{F}| / 2^{H_\infty(\mathcal{W})}}$$

for any *oblivious bit-fixing* source $\mathcal{W}$ with $2^{0.585 H_\infty(\mathcal{W})} \leq |\mathbb{F}| \leq 2^{H_\infty(\mathcal{W})}$. Next, we use matroid theory to *directly* analyze the n-way multicollision probability of the IPE, yielding an asymptotic bound for any even n. For $n \geq 4$, $0 < \epsilon \leq 0.83/(n-2)$, and $|\mathbb{F}| \leq 2^{(1-\epsilon)\cdot H_\infty(\mathcal{W})}$, as $|\mathbb{F}| \to \infty$,

$$\varepsilon \leq \frac{(n-1)}{2}\left(|\mathbb{F}|^\gamma / 2^{H_\infty(\mathcal{H})}\right)^{\frac{1}{n}} \sqrt{2^{-\epsilon \cdot H_\infty(\mathcal{W})}}\ (1 + o(1)).$$

Computing a *concrete* version of this bound requires time exponential in n. We compute concrete $\{4, 6, 8\}^{\text{th}}$-root bounds and demonstrate that no one choice of n is optimal. Finally, we introduce a new class of *seed-adaptive* oblivious bit-fixing sources, extend our results to such sources, and use this extension to fix a bug that we identify in the proof of the oblivious linear evaluation protocol of Doerner et al. (SP'24).

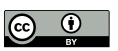
2012 ACM Subject Classification Security and privacy → Cryptography; Security and privacy → Information-theoretic techniques

Keywords and phrases Leftover hash lemma, Inner product extractor, Randomness extraction, Oblivious linear evaluation

Digital Object Identifier 10.4230/LIPIcs.ITC.2026.1

Related Version *Full Version*: `https://eprint.iacr.org/2026/654` [12]

Funding *Jack Doerner*: The work of Jack Doerner was supported in part by the Brown University Data Science Institute.
Lawrence Roy: The work of Lawrence Roy was supported by the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme under grant agreement number 101124977 (DECRYPSIS).



Acknowledgements The authors thank Daniel Wichs for a helpful discussion, and abhi shelat and Ethan Rathbun for the use of their computers. Part of this work was conducted while Lawrence Roy was visiting the Simons Institute for the Theory of Computing.

1 Introduction

A *randomness extractor* (often simply an *extractor*) is a function that produces a nearly uniformly-distributed output using a weak source of entropy. As such, they are a foundational primitive with a wide variety of applications in cryptography and connections to many other important concepts in theoretical computer science. *Seeded* extractors, first introduced by Nisan and Zuckerman [22], take a uniformly sampled seed in addition to their weak entropy source. *Strong* seeded extractors guarantee that the statistical distance between the uniform distribution and the distribution of the extractor's output (usually known as the *error* of the extractor) be bounded even given knowledge of the seed, which ensures that the seed can be reused, while the source is sampled afresh at each invocation. Intuitively, a seeded extractor can be thought of as (re)using a small amount of high-quality randomness to turn a large amount of low-quality randomness into an output that is close to uniform.

Perhaps the best known specific randomness extractor construction is the *Inner Product Extractor* (IPE) of Chor and Goldreich [2], which takes as its seed a vector h of γ elements of some field $\mathbb{F}$, and as its source a γ-bit vector x of sufficient min-entropy, and produces an output $y = \langle h, x \rangle$. It is a classic exercise in many introductory cryptography courses to prove that the error of the inner product extractor is bounded by the following equation:

$$\mathsf{SD}((\langle \mathcal{H}, \mathcal{X} \rangle, \mathcal{H}), (\mathcal{Y}, \mathcal{H})) \leq \sqrt{|\mathbb{F}|/2^{H_\infty(\mathcal{X})}} \tag{1}$$

where $\mathcal{X}$ is the source distribution, $\mathcal{Y}$ is the uniform distribution over the output domain $\mathbb{F}$, and $\mathcal{H}$ is the distribution from which h is drawn (i.e. the uniform distributions over $\mathbb{F}^\gamma$), and where $H_\infty(\cdot)$ is the min-entropy function and $\mathsf{SD}(\cdot, \cdot)$ is statistical distance. This proof typically proceeds by first observing that any two distinct inputs differ in at least one bit, and since the seed element corresponding to this bit is uniformly sampled and contributes to the output for only one of the two inputs, the probability of a collision on any two distinct inputs (over the choice of the seed) is $1/|\mathbb{F}|$. Functions with this collision probability on distinct inputs are said to be *2-Universal Hash Functions*, and we can apply the celebrated *Leftover Hash Lemma* (LHL) of Impagliazzo, Levin, and Luby [17] to show that the output of such a function on a uniform seed and a source from $\mathcal{X}$ has a statistical distance bounded as per Equation (1).

It is not always the case that a high-quality seed is available in practice, which motivates the study of *multi-source* extractors that take two or more weak entropy sources. If a multi-source extractor attains its statistical guarantee even under the disclosure of one or more of its sources, it is said to be *strong*.[1] Generalizing the LHL to account for biased seeds yields the implication that the IPE is a strong seed-biased extractor with

$$\mathsf{SD}((\langle \mathcal{G}, \mathcal{X} \rangle, \mathcal{G}), (\mathcal{Y}, \mathcal{G})) \leq \sqrt{|\mathbb{F}|^{\gamma+1}/2^{H_\infty(\mathcal{G})+H_\infty(\mathcal{X})}}$$

for any *arbitrarily biased* seed distribution $\mathcal{G}$ over $|\mathbb{F}|^\gamma$ that is independent of $\mathcal{X}$ [25, 3, 20].

[1] Hereinafter, we often refer to strong multi-source extractors as *seed-biased* extractors.

Why Study the IPE?

In the 40 years since the IPE was first introduced, extractors have advanced a great deal. In particular, a recent work of Doran and Fridman [13] proposes an extractor for oblivious bit-fixing sources (the same variety we consider) that outputs almost all of the entropy in its source, requires very little source entropy (relative to the total number of source bits), and achieves low error, all *without* the use of a seed, which eliminates the possibility that the seed can be biased, or that the bias of the source can depend upon the seed. Yet we claim that the IPE retains an important advantage that makes it worthy of continuing study: it can be computed by a depth-1 circuit over $\mathbb{F}$, and it is linear in both the source and seed. These properties make it ideal in the context of *multiparty computation* (MPC) based upon linear secret sharing over $\mathbb{F}$, where round count usually depends upon circuit depth and the IPE fits natively into the computation paradigm. Because the IPE also achieves negligible error, it is suitable for use in MPC protocols that compute cryptographic functions. To the best of our knowledge, the only other extractors that combine these properties are also based upon universal hashing.[2]

Why Focus on Oblivious Bit-Fixing Sources?

Each sample from an *oblivious bit-fixing* (OBF) distribution $\mathcal{W}$ over $\mathbb{1}^\gamma$ contains $H_\infty(\mathcal{W})$ uniform bits (at fixed locations) and $\gamma - H_\infty(\mathcal{W})$ bits that are constant. Oblivious bit-fixing sources naturally model a common situation in MPC protocol analysis: selective failure attacks (by which the adversary conditions protocol aborts on the values of specific secrets) can sometimes effectively fix secret random bits. This pattern appears in particular in a number of works that construct Oblivious Linear Evaluation (OLE) in the Oblivious Transfer (OT) hybrid model [19, 8, 9, 16, 10, 5]. This work focuses exclusively on oblivious bit-fixing sources. We do not know how to extend our results to sources with arbitrary bias,[3] and we consider such an extension to be an interesting open problem for follow-up work.

The Impact of Public Seeds

In the MPC context, it would be ideal to sample and fix a seed as a public *common random string*, and thereafter use the IPE as a quasi-seedless extractor. This would eliminate the need to use expensive seed-sampling protocols later on, which are likely to incur additional rounds in the best case, and may not be possible at all in the worst case [4]. Fixing the seed in advance introduces a new issue, however: bias in the source can now depend upon the value of the seed! This violates the source/seed independence upon which the generalized LHL (and the analyses of most two-source extractors) relies. In order to use the IPE in this way, we must bound the effects of source/seed dependence upon the IPE's error. We show in this work that in the context of OBF sources and the IPE, this kind of seed-dependent source bias can be modeled as (additional) bias in the seed alone: thus we can focus our attention on minimizing the impact of seed bias on the IPE's error.

[2] For example, universal hashing from polynomial evaluation achieves these properties *if* online secure computation of the powers of the seed can be avoided. See e.g. Degabriele et al. [6] and the citations therein.

[3] See Section 3.2 for a discussion of the barriers that prevent them from being extended in the same way as the LHL.

Why can We Hope for Improvement?

Both the classic LHL and the General LHL upper-bound the statistical distance of the IPE's output distribution with a square-root. This is because, within the proof of both lemmas, the *square* of the statistical distance is transformed into a statement about 2-way collision probability, and this is bounded using the 2-universality of the IPE. The choice of the exponent used is motivated by specific constraints: the n^{th} power of statistical distance can be transformed into a statement about n-way collision probability, and a bound on the latter would imply an n^{th}-root bound on statistical distance. Moreover, due to the use of Jensen's lemma, the bias term $|\mathbb{F}|^\gamma/2^{H_\infty(\mathcal{G})}$ appears *without* an exponent, which means that it appears under an n^{th}-root in the final bound, dramatically reducing its impact (and improving the performance of the extractor under seed bias). However, Jensen's inequality only applies to convex functions, which implies that we can only choose even values of n. The IPE is *not* 4-universal, and thus the classic proof breaks down for any $n > 2$. This leaves a pair of open questions:

> Can we bound the n-way collision probability of the IPE for $n \geq 4$?
>
> Is the impact of seed bias on the error of the IPE bounded by the n^{th} root of the bias, for any $n > 2$?

In this work, we answer both questions in the affirmative. We demonstrate that the square-root GLHL is not tight for the IPE on bit-fixing sources by introducing concrete n^{th}-root bounds for $n \in \{4, 6, 8\}$, which we derive via multiple techniques, and an asymptotic bound for any fixed even n as $H_\infty(\mathcal{W}), |\mathbb{F}| \to \infty$.

1.1 Our Contributions

1. We introduce a new class of oblivious bit-fixing sources in which the locations of the fixed bits are chosen *adaptively* by an adversary with knowledge of the seed, and show that the error due to this kind of source bias is upper-bounded by the error due to a certain magnitude of seed bias alone. This notation is similar to that of seedless extraction, in that it *non-constructively* implies a seedless extractor by taking the best seed (the seed with the least worst-case bias).
2. We bound the number of inputs on which the inner product extractor is not 4-universal, and use this to produce 4^{th}-root bound for the error of the IPE via a new generalization of the LHL that does not require the extractor to be a perfectly universal hash. This result is stated in full as Theorem 32 in Section 5, but its simplest and clearest statement is via the following:

   ▶ **Corollary 1** (of Theorem 32). *Let $H = \mathbb{F}_m^\gamma$ be the IPE family (Definition 13) from $W = \mathbb{1}^\gamma$ to $Y = \mathbb{F}_m$. If $\gamma \geq \xi \geq \log_2 m \geq 0.585\xi$, $m > 3$, $2^\xi \gg 4$, and $\eta > 0$, then for any arbitrary seed distribution $\mathcal{H}$ on H such that $H_\infty(\mathcal{H}) \geq \eta$ and any oblivious bit-fixing (Definition 8) source distribution $\mathcal{W}$ on W with $H_\infty(\mathcal{W}) \geq \xi$, we have*

$$\mathsf{SD}((\langle \mathcal{H}, \mathcal{W} \rangle, \mathcal{H}), (\mathcal{Y}, \mathcal{H})) \lesssim \frac{2.1}{2} \sqrt[4]{\frac{m^\gamma}{2^\eta}} \sqrt{\frac{m}{2^\xi}}$$

   where $\mathcal{Y}$ is the uniform distribution on Y. For any seed-adaptive *oblivious bit-fixing (Definition 10) distribution $\mathcal{W}'$ on W with $H_\infty(\mathcal{W}') \geq \xi$,*

$$\mathsf{SD}((\langle \mathcal{H}, \mathcal{W}' \rangle, \mathcal{H}), (\mathcal{Y}, \mathcal{H})) \lesssim \frac{2.1}{2} \sqrt[4]{\frac{m^\gamma \binom{\gamma}{\xi}}{2^\eta}} \sqrt{\frac{m}{2^\xi}}$$

3. Taking a different approach, we analyze the overall i-way collision probability of the inner product extractor using techniques from *matroid* theory. Specifically, we describe an equation with exponentially many terms that exactly computes the probability we need, and we show how this equation yields a concrete n^{th}-root bound on the error of the IPE for any n. Using a $2^{O(n^2)}$ algorithm, we compute concrete n^{th}-root bounds for the IPE at $n \in \{4, 6, 8\}$. The 4^{th}-root bound we derive via this method is actually both tighter and simpler to express than the one we derived manually.[4] On the other hand, the 8^{th}-root bound we derive via this method has 89 terms under the radical and takes nearly one page to print. Using these concrete bounds, we show that the optimal n depends on $|\mathbb{F}|$, γ, and $H_\infty(\mathcal{W})$. This result is reported in the full version of this paper [12].
4. In the full version of this paper [12], we asymptotically bound our concrete n^{th}-root error bound, as both the output length and the min-entropy of the source go to ∞. This result is stated in full as Theorem 38 in Section 6, but its simplest and clearest statement is via the following:

   ▶ **Corollary 2** (of Theorem 38). *Let $H = \mathbb{F}_m^\gamma$ be the IPE family (Definition 13) from $W = \mathbb{1}^\gamma$ to $Y = \mathbb{F}_m$. If $\gamma \geq \xi > 0$, $\eta > 0$, and $m \leq 2^{(1-\epsilon)\cdot\xi}$ for some $0 < \epsilon \leq 0.83/(n-2)$, then for any even constant integer $n \geq 4$, any arbitrary seed distribution $\mathcal{H}$ on H such that $H_\infty(\mathcal{H}) \geq \eta$, and any oblivious bit-fixing (Definition 8) source distribution $\mathcal{W}$ on W with $H_\infty(\mathcal{W}) \geq \xi$, as $m \to \infty$ we have*

$$\mathsf{SD}((\langle \mathcal{H}, \mathcal{W} \rangle, \mathcal{H}), (\mathcal{Y}, \mathcal{H})) \leq \frac{n-1}{2} \sqrt[n]{\frac{m^\gamma}{2^\eta}} \sqrt{2^{-\epsilon\cdot\xi}} \, (1 + o(1))$$

   where $\mathcal{Y}$ is the uniform distribution on Y. For any seed-adaptive *oblivious bit-fixing (Definition 10) distribution $\mathcal{W}'$ on W with $H_\infty(\mathcal{W}') \geq \xi$, as $m \to \infty$ we have*

$$\mathsf{SD}((\langle \mathcal{H}, \mathcal{W}' \rangle, \mathcal{H}), (\mathcal{Y}, \mathcal{H})) \leq \frac{n-1}{2} \sqrt[n]{\frac{m^\gamma \binom{\gamma}{\xi}}{2^\eta}} \sqrt{2^{-\epsilon\cdot\xi}} \, (1 + o(1))$$

5. We describe a bug in the security proof of a widely-deployed OLE protocol [8, 9, 10] that stems from treating a particular IPE source as if it were oblivious bit-fixing, when in actuality it is a *seed-adaptive* oblivious bit-fixing source. We compute the exact number of additional source bits required in order for this protocol to attain the security level originally claimed, under each of the bounds we have introduced. In some *deployed* parameter regimes, our bounds improve over the GLHL of Lee et al. [20] by a factor of nearly 150, in terms of the number of extra source bits.

1.2 Abbreviated Technical Overview

We set the stage in Section 3 by showing that reasoning about the performance of the IPE on seed-adaptive OBF sources and *arbitrarily* biased seeds can be reduced to the problem of reasoning about the performance of the IPE on uniform sources and seeds drawn uniformly from an arbitrary subset of the overall seed set. This simplifies the problem considerably.

As we previously discussed, within the proof of the original LHL, the *square* of the statistical distance is transformed into a statement about 2-way collision probability, and this is bounded using the 2-universality of the IPE. Most of the proof is compatible with any

[4] The value of our EGLHL-derived 4^{th}-root bound comes from the fact that we prove it without using a computer to enumerate exponentially-large sets.

even power n of statistical distance (which will be transformed into a statement about n-way collision probability), but the IPE is not n-universal for any even $n > 2$. We develop two distinct means to work around this limitation.

First, we observe that even if the IPE is not n-universal for some even n, there may be only a negligible number of inputs that witness this fact, and thus the IPE behaves as though it were n-universal with overwhelming probability over the coins of the source. Motivated by this observation, we develop a general n^{th}-root *Elevated General Hash Lemma* (EGLHL) for any hash function on uniform sources and seeds drawn uniformly from an arbitrary subset of the total seed space. Our lemma does not require n-universality, but instead requires a bound on j-way *distinct-input collision probability* (CPDI) of the hash function, for every $j \in [n]$. This result is restated and proven in Section 4.1:

▶ **Lemma 3** (Elevated General Leftover Hash Lemma). *Let H be a family of hash functions from X to Y. Let $G \subseteq H$ and let $\mathcal{G}$, $\mathcal{X}$, and $\mathcal{Y}$ denote the uniform distributions over G, X, and Y respectively. Let* CPDI *be the* distinct-input collision probability *of H (see Definition 16), and let $\varepsilon = \mathsf{SD}((\mathcal{G}(\mathcal{X}), \mathcal{G}), (\mathcal{Y}, \mathcal{G}))$. If $n \in \mathbb{N}^+$ is even, then*

$$\varepsilon \leq \frac{1}{2}\sqrt[n]{\frac{|H|}{|G|}\left(1 + \sum_{j=1}^{n} \mathsf{CPDI}(j) \sum_{i=j}^{n} \binom{n}{i} \left\{ {i \atop j} \right\} \frac{|X|^{\underline{j}} |Y|^{i-1}}{(-1)^{n-i} |X|^i}\right)}$$

For the IPE in particular, we can bound the j-way distinct-input collision probability using the *exceptional input fraction* (EIF) for sets of j inputs; that is, the fraction of all sets of j inputs that are witnesses to the fact that the IPE is not j-universal. In Section 5, we bound the EIF for every $j \leq 4$ by writing each possible set of inputs as the columns of a matrix, and then counting the possibilities for each row. We exhaustively categorize combinations of rows that imply j-universality; in particular, we need sufficiently-many linearly independent rows, and the fraction of possibilities that have this property is enlarged by the requirement that columns be distinct. Categorizing possible row combinations yields a bound on EIF, and by implication the 4^{th} root bound reported in Corollary 1. However, we are unable to devise a similar bound on EIF for sets of $j > 4$ inputs. We thus introduce a second approach that allows us to *exactly* compute n-way collision probability for $n > 4$.

Our second approach essentially generalizes the first, using tools from combinatorics. In particular, there is a matroid corresponding to the set of all possible $\{0, 1\}$-matrices with n distinct columns; this is the linear matroid of all vertices in an $(n-1)$-cube. Using this matroid, we can sample rows *independently*, and compute the probability that the corresponding matrix has a particular rank (which implies a particular collision probability). Properties related to sampling randomly from the matroid and computing ranks are all described by a polynomial invariant of the matroid called the Tutte polynomial (and the closely-related *coboundary* polynomial that can be derived from it), and we can compute the exact collision probability of the IPE by evaluating a particular derivative of this polynomial at a constant. Using our exact formula for collision probability, we give an n^{th}-root bound for the error of the IPE on uniform sources and seeds drawn uniformly from an arbitrary subset of the total seed space. This result is restated and proven in the full version [12]:

▶ **Theorem 4.** *Let $H = \mathbb{F}_m^{\xi}$ be an IPE family (Definition 13) from $X = \mathbb{1}^{\xi}$ to $Y = \mathbb{F}_m$. Let $G \subseteq H$ and let $\mathcal{G}$, $\mathcal{X}$, and $\mathcal{Y}$ denote the uniform distributions over G, X, and Y respectively. Let $\overline{\chi}_{M_{i-1}^{\square}}$ be the coboundary polynomial of the cube matroid (Definition 36), and let $\varepsilon = \mathsf{SD}((\langle \mathcal{G}, \mathcal{X} \rangle, \mathcal{G}), (\mathcal{Y}, \mathcal{G}))$. If $n \in \mathbb{N}^+$ is even, then*

$$\varepsilon \leq \frac{1}{2}\sqrt[n]{\frac{|H|}{|G|}\left(1 + \sum_{i=1}^{n} \binom{n}{i} \frac{(-1)^{n-i}}{2^{\xi \cdot (i-1)}} \left(\frac{d}{d\alpha}\right)^{\xi} \overline{\chi}_{M_{i-1}^{\square}}(m, e^{\alpha})\Big|_{\alpha=0}\right)}$$

Unfortunately, the most efficient algorithm that we are aware of for computing Tutte polynomials runs in $2^{O(n^2)}$ time. For small $n \leq 8$ this is feasible and we do it. The resulting polynomials and the error bounds that they imply include dozens of terms, but evaluating them on specific sets of parameters is efficient, once they are computed. We have included these error bounds in the full version [12]. Towards understanding the behavior of the IPE's error when n is larger, we perform an asymptotic analysis of the Tutte polynomial of the cube matroid. We begin by decomposing it into an exponential number of terms, each of which relates to a specific *flat* of the matroid,[5] and then we observe that the largest terms asymptotically are those corresponding flats that have the largest size relative to their rank. We classify the terms according to this property, with one class of maximal terms per rank, plus one class of non-maximal terms. We then bound each class individually; although our bounds are relatively course, they give us sufficient information to identify and concretely upper-bound the asymptotically dominant term in the overall error, and to asymptotically upper-bound the other terms. Thus we arrive at Corollary 2.

1.3 Applications to Malicious OLE-from-OT Protocols via Seed-Adaptive Oblivious Bit Fixing

Oblivious Transfer (OT) [23, 14] is a two-party protocol that allows a *receiver* to select one of two messages supplied by a *sender*. *Oblivious Linear Evaluation* (OLE) [21] is a natural algebraic generalization of OT, in which the sender supplies the coefficients of a line, and the receiver learns the evaluation of this line over some finite ring on a chosen secret point. We can view OT as producing secret shares of the product of boolean secrets, and OLE as producing secret shares of the product of secrets in an arbitary finite ring. Both OT and OLE have been the subject of intense study, and both have served as building blocks in a very large variety of applications.

It is natural to ask whether OLE can be derived from OT, and if so, how efficiently? One of the earliest and best known constructions answering this question was Gilboa's semi-honest multiplication protocol [15], which can perform a single OLE over $\mathbb{F}_m$ using $\mu = \log_2(m)$ OT instances. In the *malicious* setting, generic approaches can be applied to enhance the security of Gilboa's protocol [18], but these attain efficiency only under amortization. The question of an explicit, purely OT-based maliciously secure OLE protocol with concrete efficiency in the non-amortized setting remained open until it was implicitly closed by Keller, Orsini, and Scholl [19], whose protocol required a few additional rounds and a communication overhead factor of roughly 6 relative to Gilboa's baseline. Later, purely OT-based maliciously secure OLE protocols with overhead factors between 2 and 4 and no added rounds (in the random oracle model) were given by Doerner et al. [8, 9, 10]. The last of these remains the state of the art. Haitner et al. [16] proposed a protocol with even lower overhead, but it realizes a *relaxed* notion of OLE or else relies upon on the DDH assumption. Later, Doerner et al. [7] took a different approach based upon the Chinese remainder theorem which yields a purely OT-based OLE protocol with communication cost that is *subquadratic* in the modulus length. Their protocol is the first to outperform Gilboa's semi-honest baseline in the non-amortized setting, but in order to achieve malicious security it requires at least five rounds.

[5] A flat can be thought of as a collection of rows to which no additional rows can be added without increasing the rank, i.e., a linear subspace.

Fixing A Bug in the DKLs OLE Protocols

The DKLs OLE protocols [8, 9, 10] work by securely computing the IPE using OT instances. Specifically, OT is used to compute a sharing of $\alpha \cdot \langle h, x \rangle$, where $x \in \mathbb{1}^\gamma$ is the extractor source, supplied by the receiver, $\alpha \in \mathbb{F}_m$ is the sender's secret input to the multiplication, and $h \in \mathbb{F}_m^\gamma$ is uniformly random *public* extractor seed. This secret sharing is then converted into a secret sharing of $\alpha \cdot \beta$ for an arbitrary secret input $\beta \in \mathbb{F}_m$ supplied by the receiver.

Part of the security analysis of Doerner et al. deals with selective failure attacks, by which a malicious sender can cause the protocol to detectably abort if it guesses a particular subset of the bits of x correctly. If such an attack occurs, and the protocol *does not abort*, then the corrupt sender learns those bits, and they thus become fixed. The security proof relies upon the extractor's output being indistinguishable from uniform, conditioned on the fixing of these source bits. Specifically, Doerner et al. *exclude* the seed elements corresponding to the fixed bits from the analysis, and argue about the distribution of the inner product of the remaining source bits and seed elements.

It is exactly at this point that there is a bug in the versions of Doerner et al.'s proof dated December 2023 and earlier. Specifically, in their proof of Claim 6.10, they analyze the extractor's statistical distance from uniform using the classic LHL [17], which requires a uniform seed. However, the adversary can adaptively choose which bits to fix (and thus which seed elements to exclude) with full knowledge of the seed, which was sampled and published in advance. The non-excluded seed elements are therefore adversarially biased, and the classic LHL does not apply.

We can fix the bug in Doerner et al. by characterizing the seed bias induced by seed-adaptive oblivious bit fixing, and then analyzing the extractor under this bias. In Section 3 we prove that if c out of γ source bits are fixed, then the min-entropy of the seed is reduced by $\log_2(\binom{\gamma}{c})$ and the source min-entropy is reduced by c. This leads finally to a key question:

> How many *extra* source bits are needed to attain a particular level of security under seed-adaptive oblivious bit-fixing, relative to non-adaptive oblivious bit-fixing.

In fixing the bug, we incur a performance penalty that hinges on the answer to this question. Specifically, we must perform a number of additional OT instances proportionate to the number of extra source bits, which means that the cost of the patched protocol will grow linearly with this number. We answer the question and report the number of extras concretely for a variety of parameter regimes in the full version [12]. Doerner et al. [10] suggest to deploy their protocol with $|m| = 256$ and to set the statistical parameter $s = 80$. In this parameter regime, the techniques we introduce in this paper reduce the number of extra OT instances that we must add by a factor of 11, from 271 to 22. The improvement is even greater if $|m| = 384$ is used (as might be required when working with pairing curves [11], for example): in this case the number of extra OTs reduces from 295 to 2!

2 Preliminaries

Notation

In this document, we use $=$ for equality, $\equiv$ for congruence, and $\leftarrow$ for sampling. We typically use capital italic letters (e.g. X) to denote sets, capital caligraphic letters (e.g. $\mathcal{X}$) to denote distributions, and lower case italic letters (e.g. x) to denote matrices, vectors, or scalars. Set literals are denoted with $\{\ldots\}$, ordered tuple literals with $(\ldots)$, and $[a, b]$ generates the ordered tuple of integers from a to b inclusive (or from 1 to b if a is omitted). $|X|$ denotes

the size of X. Function names are typeset either in roman font for standard mathematical functions, or sans-serif font for functions that we define. Some mathematical functions (such as rank and dim) use a subscript to disambiguate the field or matroid relative to which they are defined. $\mathbb{N}^+$ denotes the set of all natural numbers greater than zero, $\mathbb{F}_m$ a prime field with order m, and $\mathbb{1} = \{0, 1\}$. The variable e always refers to Euler's number. $H_\infty(\cdot)$ denotes the min-entropy function and $\mathsf{SD}(\cdot, \cdot)$ denotes statistical distance. For additional notation related to matroids, we direct the reader to the full version [12].

2.1 Useful Combinatorial Notions

▶ **Definition 5** (The Falling Factorial Function)**.**

$$x^{\underline{n}} = \overbrace{x(x-1)\dots(x-n+1)}^{n \text{ factors}}$$

▶ **Definition 6** (Stirling Numbers of the Second Kind)**.** *The second-kind Stirling number* $\left\{ {i \atop j} \right\}$ *counts the number of ways to partition* i *elements into exactly* j *subsets. It is given by the following sum:*

$$\left\{ {i \atop j} \right\} = \frac{1}{j!} \sum_{k=0}^{j} (-1)^{j-k} \binom{j}{k} k^i$$

2.2 Randomness Extractors

A strong extractor is an efficient function which takes a random source and a random seed, and produces uniform-looking bits, even to a distinguisher that can see the seed.[6] The strongest notion we will consider takes input from an arbitrary distribution of sufficient min-entropy.

▶ **Definition 7** ((Strong) Randomness Extractors)**.** *Given* $\varepsilon > 0$*, a source distribution* $\mathcal{X}$ *over source set* X*, and a seed distribution* $\mathcal{H}$ *over seed set* H*, a* strong $(\mathcal{X}, \mathcal{H}, \varepsilon)$-extractor *is an efficient function* $\mathrm{Ext} : H \times X \to Y$*, usually written* $h(x)$ *for* $h \in H$[7] *and* $x \in X$*, such that* $\mathsf{SD}((\mathcal{H}(\mathcal{X}), \mathcal{H}), (\mathcal{Y}, \mathcal{H})) \leq \varepsilon$,[8] *where* $\mathcal{Y}$ *is the uniform distributions on the output set* Y*.*

Overloading notation, we say that an $(\xi, \mathcal{H}, \varepsilon)$-extractor *is an* $(\mathcal{X}, \mathcal{H}, \varepsilon)$*-extractor over every source* $\mathcal{X}$ *such that* $H_\infty(\mathcal{X}) \geq \xi$*.*

However, we will mostly stick to a weaker class of extractors, which are only guaranteed to work on oblivious bit-fixing sources.

▶ **Definition 8** (Oblivious Bit-Fixing (OBF) Sources)**.** *A* (ξ, γ)-OBF source *is a distribution* $\mathcal{W}$ *on* $W = \mathbb{1}^\gamma$ *wherein the individual bits are independently sampled, each bit is either uniform or constant, and there are exactly* ξ *uniform bits, in fixed locations. Equivalently,* $\mathcal{W}$ *is a fixed permutation applied to the uniform distribution on* $\mathbb{1}^\xi \times \{f\}$*, for some fixed bit vector* $f \in \mathbb{1}^{\gamma - \xi}$*.*

[6] *Weak* extractors provide guarantees only when the seed is hidden. We do not consider any weak extractors in this work, and therefore all extractors mentioned herein should be assumed to be strong.
[7] We often refer to H as an *extractor family* and think of $h \in H$ as a function $h : X \to Y$.
[8] The $\mathcal{H}$'s in this equation are the same random variable, not independent samples from $\mathcal{H}$.

We note that the literature contains a variation on bit-fixing sources wherein an adversary can choose the values of the fixed bits as a function of the uniform ones [1]. These are known as *non-oblivious* bit-fixing (NOBF) sources. We do not consider such sources in this work, and unless otherwise specified, all bit-fixing sources used in this work should be presumed to be oblivious.

▶ **Definition 9** (Oblivious Bit-Fixing (OBF) Extractors). *A* strong $(\xi, \mathcal{H}, \varepsilon)$-OBF extractor *is a strong* $(\mathcal{W}, \mathcal{H}, \varepsilon)$*-extractor per Definition 7 for every source* $\mathcal{W}$ *that is* (ξ, γ)-oblivious-bit-fixing *for some* $\gamma \geq \xi$*, per Definition 8.*

We also introduce a new variation on bit-fixing sources, in which the indices of the fixed bits are chosen adaptively by an adversary as a function of the seed.

▶ **Definition 10** (Seed-Adaptive OBF Sources). *A* (ξ, γ)-SAOBF source *is a distribution* $\mathcal{W}$ *on* $W = \mathbb{1}^\gamma$ *wherein the individual bits are independently sampled, each bit is either uniform or constant, and there are exactly* ξ *uniform bits, in locations chosen by the adversary as a function of the extractor's seed. Equivalently,* $\mathcal{W}$ *is a permutation that is* adversarially *derived from the seed and applied to the uniform distribution on* $\mathbb{1}^\xi \times \{f\}$*, for some fixed* $f \in \mathbb{1}^{\gamma-\xi}$*.*

▶ **Definition 11** (Seed-Adaptive OBF Extractors). *A* strong $(\xi, \mathcal{H}, \varepsilon)$-SAOBF extractor *is a strong* $(\mathcal{W}, \mathcal{H}, \varepsilon)$*-extractor per Definition 7 for every source* $\mathcal{W}$ *that is* (ξ, γ)-seed-adaptive-oblivious-bit-fixing *for some* $\gamma \geq \xi$*, per Definition 10.*

Extractors as defined above are often flexible enough to tolerate sources with significant unknown (and even adversarial) bias, so long as those sources have sufficient min-entropy. Allowing this same kind of flexibility over the seed distribution defines a biased-seed extractor; our main concern in this paper is analyzing extractors with biased seeds.

▶ **Definition 12** (Biased-Seed Extractors). *A* strong $(\mathcal{X}, \eta, \varepsilon)$-biased-seed extractor *is an strong* $(\mathcal{X}, \mathcal{H}, \varepsilon)$*-extractor for all seed distributions* $\mathcal{H}$ *such that* $H_\infty(\mathcal{H}) \geq \eta$*.*

The particular extractor we will analyze is the inner product extractor.

▶ **Definition 13** (Inner Product Extractor). *The* $\mathbb{F}_m$*-inner-product extractor has source set* $X = \mathbb{1}^\gamma$*, seed set* $H = \mathbb{F}_m^\gamma$*, and evaluation* $h(x) = \langle h, x \rangle$*.*

2.3 Universal Hashing and Exceptional Inputs

▶ **Definition 14** (Multicollision Probability). *For a family of hash functions* $H : X \to Y$ *and an integer* i*, let* CP *be a function that computes the* i*-way collision probability of a uniformly-sampled member of* H*. That is, if we let* $\mathcal{H}$ *and* $\mathcal{X}$ *be the uniform distributions over* H *and* X *respectively, then*

$$\mathsf{CP}(i) = \Pr_{\substack{h \leftarrow \mathcal{H} \\ (x_1, \ldots, x_i) \leftarrow \mathcal{X}^i}} \left[h(x_{i_1}) = h(x_{i_2}) \forall i_1, i_2 \right].$$

▶ **Definition 15** (Distinct Inputs). *For an input domain* X *and an integer* j *let* DI *be a function computing the* j*-pairwise-distinct input set, i.e.*

$$\mathsf{DI}(j) = \left\{ (x_1, \ldots, x_j) \in X^j : \nexists\, i_1, i_2 \text{ s.t. } i_1 \neq i_2 \wedge x_{i_1} = x_{i_2} \right\}.$$

We denote the uniform distribution over this set as $\mathcal{U}_{\mathsf{DI}(j)}$*. Notice,* $|\mathsf{DI}(j)| = |X|^{\underline{j}}$*.*

▶ **Definition 16** (Distinct-Input Multicollision Probability). *For a family of hash functions $H : X \to Y$ and an integer j, let* CPDI *be a function that computes the j-way collision probability of a uniformly-sampled member of H, conditioned on pairwise-distinct inputs. That is,*

$$\mathsf{CPDI}(j) = \Pr_{\substack{h \leftarrow \mathcal{H} \\ (x_1,\ldots,x_j) \leftarrow \mathcal{U}_{\mathsf{DI}(j)}}} \left[h(x_{i_1}) = h(x_{i_2}) \forall i_1, i_2\right].$$

For convenience we also assert that $\mathsf{CPDI}(1) = 1$ *for any X.*

▶ **Definition 17** (Universal Hash Function). *A hash function family $H : X \to Y$ is ℓ-universal if for every $(x_1, \ldots, x_\ell) \in \mathsf{DI}(\ell)$,*

$$\Pr_{h \leftarrow \mathcal{H}}\left[h(x_{i_1}) = h(x_{i_2}) \forall i_1, i_2\right] \leq |Y|^{1-\ell}.$$

Note that the above condition trivially implies $\mathsf{CPDI}(\ell) \leq |Y|^{1-\ell}$.

▶ **Definition 18** (Exceptional Inputs). *For a family of hash functions $H : X \to Y$ and integers j and k, let* EI *be a function computing the set of j-pairwise-distinct input tuples on which H is not k-universal. That is,*

$$\mathsf{EI}(j,k) = \left\{ \begin{array}{l} (x_1, \ldots, x_j) \in \mathsf{DI}(j) : \\ \quad \Pr_{h \in \mathcal{H}}\left[h(x_{i_1}) = h(x_{i_2}) \forall i_1, i_2\right] > |Y|^{1-k} \end{array} \right\}.$$

Abusing notation, we will often write $\mathsf{EI}(j) = \mathsf{EI}(j,j)$.

▶ **Definition 19** (Exceptional Input Fraction). *For a family of hash functions $H : X \to Y$ and integers j and k, let* EIF *be a function computing the* fraction *of j-pairwise-distinct input tuples on which H is not k-universal. That is,*

$$\mathsf{EIF}(j,k) = \frac{|\mathsf{EI}(j,k)|}{|\mathsf{DI}(j)|}.$$

Abusing notation, we will often write $\mathsf{EIF}(j) = \mathsf{EIF}(j,j)$.

▶ **Proposition 20.** *For any ℓ-universal hash function family, we have the following generic equalities and bounds on* EIF*:*

$$\begin{aligned} &\mathsf{EIF}(j,k) \leq \mathsf{EIF}(j,k+1) \\ &\mathsf{EIF}(j,k) = 0 && \text{if } k \leq \ell \\ &\mathsf{EIF}(j) = 0 && \text{if } j \leq \ell \end{aligned}$$

▶ **Proposition 21.** *For any ℓ-universal hash function family $H : X \to Y$, we have the following generic bounds on* CPDI*:*

$$\begin{aligned} \mathsf{CPDI}(j) &\leq |Y|^{1-j} + \sum_{k=\ell+1}^{j} \mathsf{EIF}(j,k)\left(|Y|^{2-k} - |Y|^{1-k}\right) && (2) \\ &\leq |Y|^{1-j} + \mathsf{EIF}(j)|Y|^{1-\ell} && (3) \\ \mathsf{CPDI}(j) &\leq \mathsf{CPDI}(j-1) \\ \mathsf{CPDI}(j) &\leq |Y|^{1-j} && \text{if } j \leq \ell \end{aligned}$$

▶ **Proposition 22.** *For any function family $H : X \to Y$, we have the following generic equalities for* CP*:*

$$\mathsf{CP}(i) = \sum_{y \in Y} \Pr_{\substack{x \leftarrow \mathcal{X} \\ h \leftarrow \mathcal{H}}} [h(x) = y]^i$$

$$\mathsf{CP}(0) = |Y|$$

$$\mathsf{CP}(1) = 1$$

$$\mathsf{CP}(i) = \sum_{j=1}^{i} \left\{ {i \atop j} \right\} \frac{|X|^{\underline{j}}}{|X|^i} \mathsf{CPDI}(j) \qquad \text{if } i > 0 \tag{4}$$

Proofs of Equations (2) and (4) are given in the full version [12]. The full version also contains additional preliminaries relating to matroids, Tutte polynomials, and coboundary polynomials, and a simple proof that the rank of a matrix is only sensitive to the field over which the entries of that matrix is defined when the order of that field is small relative to the length of the short side of the matrix.

3 Restricting the Problem

In this section, we will prove a few useful lemmas that allow us to restrict our attention to only *uniform* sources and *uniform-over-subset* seeds for the remainder of the document.

3.1 The Sufficiency of Uniform-Over-Subset Seed Distributions

Randomness extractors typically assume a uniform seed distribution and tolerate a biased source distribution with sufficient min-entropy. A standard lemma about randomness extractors [24, Lemma 6.10] asserts that in order to reason about all sources with a certain min-entropy, it is sufficient to reason about sources that are uniformly distributed over arbitrary correspondingly-sized subsets of the source space. We adapt this idea to arbitrarily-biased *seeds* using following lemma (proven in the full version [12]). Hereafter in this document, we will restrict our focus to seed distributions $\mathcal{G}$ that are uniform over subsets $G \subseteq H$.

▶ **Lemma 23** (Convex Combination of Uniform Seeds)**.** *Let ε be a concave function and let H be an extractor family. If H is a strong $(\mathcal{X}, \mathcal{G}, \varepsilon(|G|^{-1}))$-extractor per Definition 7 for every $G \subseteq H$ and $\mathcal{G}$ uniformly distributed over G, then then H is also a strong $(\mathcal{X}, \eta, \varepsilon(2^{-\eta}))$-*biased-seed *extractor for all η, per Definition 12.*

3.2 The (In)Sufficiency of Uniform Source Distributions

Lemma 23 showed that in order to reason about any arbitrily-biased seed distributions $\mathcal{H}$, we need only prove that an extractor is good on the set of all seed distributions $\mathcal{G}$ that are uniform over some $G \subseteq H$ where $|G| = 2^{H_\infty(\mathcal{H})}$. It would be tempting to use a similar line of reasoning to restrict our attention to uniform-over-subset source distributions, and claim a generalization to all sources with sufficient min-entropy, and indeed this is often done in other works (in particular, this technique is used in the classic LHL proof, and in Lee et al.'s proof of the GLHL [20]). Unfortunately, our analysis in the remaining sections of this paper does *not* permit this line of reasoning. We are able to show that IPE is a good extractor on the source distribution that is uniform over the entire source set X, but there is a specific point in each of our proofs where the chain of implication fails if the source distribution is uniform over only a *subset* of X:

- In Definition 19 and the proof of Lemma 31 we take $\mathsf{EIF}(j)$ to be the number of inputs on which the extractor is not j-universal, over the number of pairwise-distinct input tuples. If the source distribution is uniform over a subset of X (and has probability 0 outside that subset), then the concrete bound we calculate on $\mathsf{EIF}(4)$ is false, because we must assume all "bad" inputs to be within the subset, meaning that the denominator shrinks with the subset, but the numerator does not. This invalidates Theorem 32 for arbitrary non-uniform source distributions. Our analysis could potentially be adjusted to account for this, but we do not expect such an adjustment to produce useful bounds for signficantly biased distributions over X.
- An additional example relates to a proof that appears in the full version of this paper [12].

(Static) Oblivious Bit-Fixing Sources

Although we cannot generalize from our results for uniform sources to results for *arbitrary* source distributions, we *can* generalize to the more restricted classes of *oblivious bit-fixing* (OBF) sources given in Definition 8. We begin with the simpler case of bits that are fixed *independently* of the seed.[9]

▶ **Lemma 24** (Generalization to Oblivious Bit-Fixing Sources). *Let $\varepsilon, \eta \geq 0$ and let ξ and γ be integers such that $\gamma \geq \xi$. Let $H_\xi = \mathbb{F}_m^\xi$ be the IPE family (Definition 13) from $X = \mathbb{1}^\xi$ to $Y = \mathbb{F}_m$, and let $H_\gamma = \mathbb{F}_m^\gamma$ be the IPE family from $W = \mathbb{1}^\gamma$ to Y. If H_ξ is a strong $(\mathcal{X}, \eta - (\gamma - \xi)\log_2(m), \varepsilon)$-biased-seed extractor per Definition 12, where $\mathcal{X}$ is the uniform distribution over X, then H_γ is a strong (ξ, η, ε)-biased-seed-OBF extractor per Definitions 9 and 12.*

Proof. By Lemma 23, we can restrict our attention to the set of seed distributions $\mathcal{G}$ that are uniform over some $G \subseteq H$ such that $H_\infty(\mathcal{G}) \geq \eta$. Recall from Definition 8 that any (ξ, γ)-OBF source $\mathcal{W}$ on $W = \mathbb{1}^\gamma$ is a fixed permutation applied to the uniform distribution on $\mathbb{1}^\xi \times \{f\}$, for some fixed bit vector $f \in \mathbb{1}^{\gamma-\xi}$. In other words, every $w \in W$ can be partitioned bitwise into f and some $x \in X$, and sampling $w \leftarrow \mathcal{W}$ corresponds to sampling $x \leftarrow \mathcal{X}$ and interleaving the bits of x with the bits of f. We can also partition any $h \in H_\gamma$ into $h_x \in H_\xi$ (the elements of h corresponding to bits of w that are partitioned into x) and h_f (the elements of h corresponding to bits of w that are partitioned into f). It is easy to see that $\langle h, w\rangle = \langle h_x, x\rangle + \langle h_f, f\rangle$, where $\langle h_x, x\rangle$ is an instance of the inner-product extractor on the uniform source $\mathcal{X}$, and thus the statistical distance between $\langle h, w\rangle$ and the uniform distribution is no greater than the distance between $\langle h_x, x\rangle$ and the uniform distribution.

It remains only to reason about the distribution of h_x, which we will denote $\mathcal{G}_x$. In particular, the partitioning process maps *at least* one and *at most* $m^{\gamma-\xi}$ values $h \in H_\gamma$ onto each $h_x \in H_\xi$. It therefore follows that $H_\infty(\mathcal{G}_x) \geq H_\infty(\mathcal{G}) - (\gamma - \xi)\log_2(m)$. Since H_ξ is a $(\mathcal{X}, \mathcal{G}_x, \varepsilon)$-extractor by assumption, it follows that H_γ is a $(\mathcal{W}, \mathcal{G}, \varepsilon)$-extractor. This holds for any (ξ, γ)-OBF source $\mathcal{W}$, any biased seed $\mathcal{G}$ such that $H_\infty(\mathcal{G}) \geq \eta$, and any $\mathcal{G}_x$ induced by the pair. Thus we have the lemma. ◀

3.3 Seed-Adaptive Oblivious Bit-Fixing Sources

We now turn our attention to *seed-adaptive* OBF sources, as described in Definition 10. As above, we can partition the source bit vector w into a sub-vector f of fixed bits and a sub-vector x of uniformly distributed bits, and we can partition the corresponding elements

[9] Such an analysis was already performed by Doerner et al. [8, 9, 10]. Their analysis is correct for the non-seed-adaptive case, but as we discussed in Section 1.3, they erred in that their proof actually required a seed-adaptive analysis.

of h into h_f and h_x. When $\mathcal{W}$ is *seed-adaptive*, however, the latter partitioning induces (additional) bias into the seed. We quantify that bias by defining an augmented seed distribution $\mathcal{D}^{\mathsf{Adv}}(\mathcal{G}, c)$ as a function of the (pre-bit-fixing) distribution $\mathcal{G}$ and the number c of fixed bits, after which we lower bound the min-entropy of $\mathcal{D}^{\mathsf{Adv}}_{\mathcal{G},c}$ relative to the min-entropy of $\mathcal{G}$.

Algorithm 1 $\mathcal{D}^{\mathsf{Adv}}_{\mathcal{G},\xi}$.

$\mathcal{D}$ `samples` $h = (h_1, \ldots, h_\gamma) \leftarrow \mathcal{G}$
`The adversary produces` $I \leftarrow \mathsf{Adv}(h, \xi)$ `such that` $I \subset [\gamma]$ `and` $|I| = \gamma - \xi$
$\mathcal{D}$ `outputs` $h^* = (h_i)_{i \notin I}$

▶ **Lemma 25.** *For all integers $\gamma \geq \xi > 0$ and all (not necessarily efficient) adversaries* Adv*, and any distibution $\mathcal{G}$ on $\mathbb{F}_m^\gamma$*

$$H_\infty\left(\mathcal{D}^{\mathsf{Adv}}_{\mathcal{G},\xi}\right) \geq H_\infty(\mathcal{G}) - (\gamma - \xi)\log_2(m) - \log_2\binom{\gamma}{\xi}$$

Proof. The min-entropy $H_\infty(\mathcal{D}^{\mathsf{Adv}}_{\mathcal{G},\xi})$ is defined as the negative logarithm of the probability of the most likely event in the distribution $\mathcal{D}^{\mathsf{Adv}}_{\mathcal{G},\xi}$. The adversary's influence is restricted to deleting $(\gamma - \xi)$ entries from h. For any fixed output h^*, the number of pre-image vectors h from which $(\gamma - \xi)$ elements can be deleted to yield h^* is $\binom{\gamma}{\gamma-\xi} m^{\gamma-\xi}$. The probability of the most likely value of h^* is upper-bounded by the probability that one of its pre-images is sampled, which is $\binom{\gamma}{\gamma-\xi} m^{\gamma-\xi} / 2^{H_\infty(\mathcal{G})}$, and the lemma follows. ◀

We therefore have the following seed-adaptive counterpart to Lemma 24:

▶ **Lemma 26** (Generalization to SAOBF Sources)**.** *Let $\varepsilon, \eta \geq 0$ and let ξ and γ be integers such that $\gamma \geq \xi$. Let $H_\xi = \mathbb{F}_m^\xi$ be the IPE family (Definition 13) from $X = \mathbb{1}^\xi$ to $Y = \mathbb{F}_m$, and let $H_\gamma = \mathbb{F}_m^\gamma$ be the IPE family from $W = \mathbb{1}^\gamma$ to Y. If H_ξ is a strong $(\mathcal{X}, \eta - (\gamma - \xi)\log_2(m) - \log_2\binom{\gamma}{\xi}, \varepsilon)$-biased-seed extractor per Definition 12, where $\mathcal{X}$ is the uniform distribution over X, then H_γ is a strong (ξ, η, ε)-biased-seed-SAOBF extractor per Definitions 11 and 12.*

Proof Sketch. The proof is similar to the proof of Lemma 24, except that the distribution of h_x is $\mathcal{D}^{\mathsf{Adv}}_{\mathcal{G},\xi}$ instead of $\mathcal{G}_x$. ◀

Hereafter in this document, we will restrict our focus to the uniform source distribution $\mathcal{X}$ over $\mathbb{1}^\xi$.

4 The Elevated General Leftover Hash Lemma

In this section, we develop an *Elevated* General Leftover Hash Lemma for *Almost-n*-Universal Hash functions. By *elevated*, we mean that our hash function is derived via the n^{th} power of the norm (as opposed to the squared norm, in the case of the standard GLHL) and it therefore establishes an n-root bound on statistical distance. By *almost-n*-universal, we mean that it does not require the hash function on which it acts to be n-universal, but instead requires only that the Distinct-Input Collision Probability be bounded generally. This is crucial: it allows us to produce a tighter analysis of a hash function that is ℓ-universal and *almost but not quite* n-universal than we could using ℓ-universality alone. Apart from this bound on collision probability, we require little information about the hash function, and work over general, unstructured domains (H, G, X, Y). Our proof proceeds very similarly to the proof of the GLHL of Lee et al. [20] and to the original LHL proof of Impagliazzo, Levin, and Luby [17].

In Section 4.2, we use additional structural information about the IPE to specialize our lemma and replace the bound on distinct-input collision probability with a bound on the number of *exceptional* inputs on which the IPE is not n-universal[10]. Later, in Section 5, we finally bound the number of exceptional inputs for 4-universality, allowing us to calculate an explicit 4^{th}-root upper bound on statistical distance for the IPE.

4.1 A Family-Agnostic n^{th} GLHL

▶ **Lemma 3.** *Let H be a family of hash functions from X to Y. Let $G \subseteq H$ and let $\mathcal{G}$ and $\mathcal{X}$ denote the uniform distributions over G and X. Let* CPDI *be the distinct-input collision probability of H (Definition 16). If $n \in \mathbb{N}^+$ is even, then H is a strong $(\mathcal{X}, \mathcal{G}, \varepsilon)$-extractor per Definition 7, where*

$$\varepsilon \leq \frac{1}{2}\sqrt[n]{\frac{|H|}{|G|}\left(1+\sum_{j=1}^{n}\mathsf{CPDI}(j)\sum_{i=j}^{n}\binom{n}{i}\left\{{i \atop j}\right\}\frac{|X|^{\underline{j}}|Y|^{i-1}}{(-1)^{n-i}|X|^{i}}\right)}$$

Proof. For all even n we have:

$$\begin{aligned}
&2^n \cdot \mathsf{SD}((\langle\mathcal{G},\mathcal{X}\rangle,\mathcal{G}),(\mathcal{Y},\mathcal{G}))^n \\
&= \left(\sum_{h\in G}\sum_{y\in Y}\frac{1}{|G|}\left|\Pr_{x\leftarrow\mathcal{X}}[h(x)=y]-\frac{1}{|Y|}\right|\right)^n \\
&= |Y|^n\left(\frac{\sum_{h\in G}\sum_{y\in Y}\left|\Pr_{x\leftarrow\mathcal{X}}[h(x)=y]-\frac{1}{|Y|}\right|}{\sum_{h\in G}\sum_{y\in Y}1}\right)^n \\
&\leq |Y|^n\frac{\sum_{h\in G}\sum_{y\in Y}\left(\Pr_{x\leftarrow\mathcal{X}}[h(x)=y]-\frac{1}{|Y|}\right)^n}{\sum_{h\in G}\sum_{y\in Y}1} \qquad (5)\\
&= \frac{|Y|^{n-1}}{|G|}\sum_{h\in G}\sum_{y\in Y}\left(\Pr_{x\leftarrow\mathcal{X}}[h(x)=y]-\frac{1}{|Y|}\right)^n \\
&\leq \frac{|Y|^{n-1}}{|G|}\sum_{h\in H}\sum_{y\in Y}\left(\Pr_{x\leftarrow\mathcal{X}}[h(x)=y]-\frac{1}{|Y|}\right)^n \qquad (6)\\
&= \frac{|Y|^{n-1}}{|G|}\sum_{h\in H}\sum_{y\in Y}\sum_{i=0}^{n}\binom{n}{i}\Pr_{x\leftarrow\mathcal{X}}[h(x)=y]^i\frac{1}{(-|Y|)^{n-i}} \qquad (7)\\
&= \frac{|H|}{|G|}\sum_{i=0}^{n}\binom{n}{i}\sum_{y\in Y}\Pr_{\substack{x\leftarrow\mathcal{X}\\ h\leftarrow\mathcal{H}}}[h(x)=y]^i\frac{|Y|^{i-1}}{(-1)^{n-i}} \\
&= \frac{|H|}{|G|}\sum_{i=0}^{n}\binom{n}{i}\mathsf{CP}(i)\frac{|Y|^{i-1}}{(-1)^{n-i}} \qquad (8)\\
&= \frac{|H|}{|G|}\left(1+\sum_{i=1}^{n}\binom{n}{i}\sum_{j=1}^{i}\left\{{i \atop j}\right\}\frac{|X|^{\underline{j}}}{|X|^i}\mathsf{CPDI}(j)\frac{|Y|^{i-1}}{(-1)^{n-i}}\right)
\end{aligned}$$

[10] A hash function is not n universal on a tuple of n distinct inputs if it has a probability greater than $|Y|^{1-n}$ of producing an n-way collision on those *specific* inputs, over the choice of its seed. See also Definition 18.

$$= \frac{|H|}{|G|}\left(1 + \sum_{j=1}^{n} \mathsf{CPDI}(j) \sum_{i=j}^{n} \binom{n}{i} \left\{ {i \atop j} \right\} \frac{|X|^{\underline{j}} |Y|^{i-1}}{(-1)^{n-i}|X|^i}\right)$$

where

Equation (5) follows from Jensen's inequality and the fact that n is even
Equation (6) follows from the fact that $G \subseteq H$
Equation (7) follows from the binomial theorem
The lemma follows from taking the n^{th} root. ◀

4.2 Applying Lemma 3 to the IPE

▶ **Proposition 27.** *For the Inner Product Extractor we have* $\mathsf{CPDI}(j) \geq |Y|^{1-j}$.

Proof. For any tuple of inputs $(x_1, \ldots, x_j) \in \mathsf{DI}(j)$, the probability of a collision under $(x_1, \ldots, x_j)$ is exactly

$$\mathsf{CPDI}(j) = \sum_{y \in \mathbb{F}_m} \Pr_{h \leftarrow \mathbb{F}_m^{\xi}} \left[\begin{bmatrix} h^T & y \end{bmatrix} \begin{bmatrix} x_1 & \cdots & x_j \\ 1 & \cdots & 1 \end{bmatrix} = \vec{0} \right].$$

Since $\vec{0}$ is in the span of any matrix, the system of linear equations implicitly defined by matrix multiplication within the probability statement is always consistent. It has $\xi + 1$ unknowns (h, y) and a coefficient matrix with rank no greater than j (since this is the number of its columns), and thus by the Rouché-Capelli Theorem the number of solutions (each of which corresponds to a collision) is no less than $m^{\xi+1-j}$. These solutions are partitioned among the various fixed values of y over which the sum is taken, with a consistent denominator comprising the total number of possible assignments of h, which is m^{ξ}. Thus the final probability of a collision must be greater than $m^{\xi+1-j}/m^{\xi} = |Y|^{1-j}$ ◀

▶ **Lemma 28.** *The Inner Product Extractor is* 3*-universal if the source is at least 2 bits long.*

Proof. Per Proposition 20, we can equivalently prove that $\mathsf{EIF}(2) = \mathsf{EIF}(3) = 0$. The proof that $\mathsf{EIF}(2) = 0$ is traditional, we omit it. To see that $\mathsf{EIF}(3) = 0$, we need only a slight extension of the argument that $\mathsf{EIF}(2) = 0$. Observe that if we interpret any three inputs as the columns in a boolean matrix, then any row where the first two columns are distinct must have a value in the third column equal to the value in either the first or second. Without loss of generality, let it be the first. Because 3-universality quantifies only over input sets with three distinct inputs (i.e. three distinct columns in the corresponding matrix), there must exist another row in which the third column is *not* equal to the first, and using this row, we can repeat the argument for $\mathsf{EIF}(2) = 0$ to conclude that $\mathsf{EIF}(3) = 0$. ◀

▶ **Lemma 29.** *Let* H *be an IPE family (Definition 13) from* X *to* Y*. Let* $G \subseteq H$ *and let* $\mathcal{G}$ *and* $\mathcal{X}$ *denote the uniform distributions over* G *and* X*. If* $n \in \mathbb{N}^+$ *is even, then* H *is a strong* $(\mathcal{X}, \mathcal{G}, \varepsilon)$*-extractor per Definition 7, where*

$$\varepsilon \leq \frac{1}{2}\sqrt[n]{\frac{|H|}{|G|}\left(\begin{array}{l} 1 + \displaystyle\sum_{j=1}^{n}\sum_{i=j}^{n} \binom{n}{i}\left\{ {i \atop j} \right\} \frac{|X|^{\underline{j}}|Y|^{i-j}}{(-1)^{n-i}|X|^i} \\ \quad + \displaystyle\sum_{j=4}^{n}\sum_{k=4}^{j} \mathsf{EIF}(j,k) \sum_{i=\lceil j/2 \rceil}^{n/2} \binom{n}{2i}\left\{ {2i \atop j} \right\} \frac{|X|^{\underline{j}}|Y|^{2i-k+1}}{|X|^{2i}} \end{array}\right)}$$

Proof. Beginning from Lemma 3, we have

$$
\begin{aligned}
&2^n \cdot \mathsf{SD}((\langle \mathcal{G}, \mathcal{X} \rangle, \mathcal{G}), (\mathcal{Y}, \mathcal{G}))^n \\
&\leq \frac{|H|}{|G|} \left(1 + \sum_{j=1}^{n} \mathsf{CPDI}(j) \sum_{i=j}^{n} \binom{n}{i} \left\{ {i \atop j} \right\} \frac{|X|^{\underline{j}} |Y|^{i-1}}{(-1)^{n-i} |X|^i} \right) \\
&= \frac{|H|}{|G|} \left(1 + \sum_{j=1}^{n} \mathsf{CPDI}(j) \left(\begin{array}{l} \displaystyle\sum_{i=\lceil j/2 \rceil}^{n/2} \binom{n}{2i} \left\{ {2i \atop j} \right\} \frac{|X|^{\underline{j}} |Y|^{2i-1}}{|X|^{2i}} \\ \displaystyle - \sum_{i=\lfloor j/2 \rfloor}^{n/2} \binom{n}{2i+1} \left\{ {2i+1 \atop j} \right\} \frac{|X|^{\underline{j}} |Y|^{2i}}{|X|^{2i+1}} \end{array} \right) \right) \\
&\leq \frac{|H|}{|G|} \left(\begin{array}{l} \displaystyle 1 + \sum_{j=1}^{n} |Y|^{1-j} \sum_{i=\lceil j/2 \rceil}^{n/2} \binom{n}{2i} \left\{ {2i \atop j} \right\} \frac{|X|^{\underline{j}} |Y|^{2i-1}}{|X|^{2i}} \\ \displaystyle + \sum_{j=1}^{n} \sum_{k=\ell+1}^{j} \mathsf{EIF}(j,k) |Y|^{2-k} \sum_{i=\lceil j/2 \rceil}^{n/2} \binom{n}{2i} \left\{ {2i \atop j} \right\} \frac{|X|^{\underline{j}} |Y|^{2i-1}}{|X|^{2i}} \\ \displaystyle - \sum_{j=1}^{n} |Y|^{1-j} \sum_{i=\lfloor j/2 \rfloor}^{n/2} \binom{n}{2i+1} \left\{ {2i+1 \atop j} \right\} \frac{|X|^{\underline{j}} |Y|^{2i}}{|X|^{2i+1}} \end{array} \right)
\end{aligned}
$$

Where the last step follows from Proposition 27 and Equation (2) of Proposition 21, and the conclusion follows from Lemma 28 ◀

▶ **Lemma 30** (Simplified Version of Lemma 29)**.** *Let H be an IPE family (Definition 13) from X to Y. Let $G \subseteq H$ and let $\mathcal{G}$ and $\mathcal{X}$ denote the uniform distributions over G and X. If $n \in \mathbb{N}^+$ is even, then H is a strong $(\mathcal{X}, \mathcal{G}, \varepsilon)$-extractor per Definition 7, where*

$$
\varepsilon \leq \frac{1}{2} \sqrt[n]{\frac{|H|}{|G|} \left(\begin{array}{l} \displaystyle 1 + \sum_{j=1}^{n} \sum_{i=j}^{n} \binom{n}{i} \left\{ {i \atop j} \right\} \frac{|X|^{\underline{j}} |Y|^{i-j}}{(-1)^{n-i} |X|^i} \\ \displaystyle + \sum_{j=4}^{n} \mathsf{EIF}(j) \sum_{i=\lceil j/2 \rceil}^{n/2} \binom{n}{2i} \left\{ {2i \atop j} \right\} \frac{|X|^{\underline{j}} |Y|^{2i-3}}{|X|^{2i}} \end{array} \right)}
$$

Proof. The proof is identical to the proof for Lemma 29, except that we use Equation (3) from Proposition 21 instead of Equation (2). ◀

5 A Simple Bound for the IPE at $n = 4$

In this section we bound the number of exceptional inputs to 4-universality for the IPE using techniques similar to those employed by prior works to show that it is 2-universal. We then plug this bound into Lemma 29 to derive our first explicit 4^{th}-root bound on the statistical distance of the IPE.

5.1 Bounding $\mathsf{EIF}(4)$

▶ **Lemma 31.** *If $\xi \geq 5$ and $m > 3$, then the IPE family (Definition 13) from $X = \mathbb{1}^\xi$ to $Y = \mathbb{F}_m$ has $\mathsf{EIF}(4) \leq 15 \cdot 6^{\log_2|X|}/|X|^{\underline{4}}$*

Proof. Let $W(\xi, 4) = \mathbb{1}^{\xi \times 4}$ and let $\widetilde{W}(\xi, 4) \subset W(\xi, 4)$ be the set of all matrices in $W(\xi, 4)$ that have four pairwise-distinct columns. We can interpret any set of 4 inputs $(x_1, x_2, x_3, x_4) \in \mathsf{DI}(4)$ as a unique matrix $w \in \widetilde{W}(\xi, 4)$ and vice versa. We will divide our argument into two mutually-exclusive cases based upon the value w, and then bound the probability that each of these cases occurs when w is sampled.

1. w contains a row such that the value in one column of that row is distinct from the values in the others; i.e. there exists a row in w with parity 1. In this case, we can apply the argument that $\mathsf{EIF}(3) = 0$ (Lemma 28) to the columns that are identical to see that the hash function is 4-universal on w.
2. w does *not* contain a row such that the value in one column of that row is distinct from the values in the others; i.e. all rows in w have parity 0. Let us now consider the *sub*-case that there are five distinct rows with Hamming weight 2 (i.e. the values in any two columns of these rows are 0 and the values in the other two columns are 1). We first observe that among any five distinct rows with Hamming weight 2, four must be linearly independent.[11][12] Without loss of generality, let us number these four linearly independent rows 1 through 4. Let $(a_1, \ldots, a_\xi) = h$ be the individual elements of the seed of the IPE, and suppose that there exists some y such that

$$\langle a_{[4]}, w_{[4],k} \rangle + y'_k = y \qquad \text{and} \qquad y'_k = \langle a_{[5,\xi]}, w_{[5,\xi],k} \rangle$$

for $k \in [4]$. Let us fix any $a_{[5,\xi]}$. This in turn fixes y'_1, y'_2, y'_3, y'_4. For each of the m possible values of y, we have a system of four equations with four unknowns (i.e. $a_{[4]}$) and a coefficient matrix of rank four (i.e. $w_{[4],k}$). By the Rouché–Capelli theorem, such a system has at most one solution. That is, for every w, $a_{[5,\xi]}$, there is at most one assignment of $a_{[4]}$ that causes a four-way collision with output value y. Summing over the m possible values of y, and dividing by the m^4 possible assignments of $a_{[4]}$ gives us an upper bound on the overall collision probability of m^{-3}. In other words, the hash function is 4-universal in this sub-case.

Putting our cases together, we find that the set $\overline{W}(\xi, 4)$ of matrices on which the inner product extractor is *not* 4-universal is a subset of those whose rows assume the values of at most four distinct vectors of hamming weight 2, or the 0 or 1 vector. In other words $|\overline{W}(\xi, 4)| \leq \binom{6}{4} 6^\xi$. We have

$$\mathsf{EIF}(4) \leq \frac{|\overline{W}(\xi, 4) \cap \widetilde{W}(\xi, 4)|}{|\widetilde{W}(\xi, 4)|} \leq \frac{|\overline{W}(\xi, 4)|}{|\widetilde{W}(\xi, 4)|} \leq 15 \frac{6^\xi}{|X|^{\underline{4}}}$$ ◀

5.2 Putting it Together

Now we can derive a concrete upper bound on distinguishability.

[11] Up to permutations on rows, there are 6 boolean matrices of dimension $(5, 4)$ that have pairwise-distinct rows, each of hamming weight 2. All of them have rank 4 over the real numbers and therefore rank 4 over $\mathbb{F}_m$ for $m > 3$, and rank is invariant under permutation of rows.

[12] Most of the matrices with exactly four distinct rows of hamming weight 2 are also 4 universal, which means it's easy to make this analysis slightly tighter, but it's uglier and it doesn't appear to lead to a concrete improvement.

▶ **Theorem 32.** *Let $H = \mathbb{F}_m^{\xi}$ be the IPE family (Definition 13) from $X = \mathbb{1}^{\xi}$ to $Y = \mathbb{F}_m$. Let $G \subseteq H$ and let $\mathcal{G}$ and $\mathcal{X}$ denote the uniform distributions over G and X. If $|Y| > 3$, then H is a strong $(\mathcal{X}, \mathcal{G}, \varepsilon)$-extractor per Definition 7, where*

$$\varepsilon \leq \frac{1}{2}\sqrt[4]{\frac{|H|}{|G|}\left(\begin{array}{l}\left(\frac{|X|^{\underline{4}}}{|X|^4} - 4\frac{|X|^{\underline{3}}}{|X|^3} + 6\frac{|X|^{\underline{2}}}{|X|^2} - 3\right) \\ + \frac{|Y|}{|X|}\left(6\frac{|X|^{\underline{3}}}{|X|^3} - 12\frac{|X|^{\underline{2}}}{|X|^2} + 6\right) \\ + \frac{|Y|^2}{|X|^2}\left(7\frac{|X|^{\underline{2}}}{|X|^2} - 4\right) + \frac{|Y|^3}{|X|^3} + 15\frac{|Y|3^{\log_2|X|}}{|X|^3}\end{array}\right)}$$

Proof. Beginning from Lemma 30, we have

$$\begin{aligned}
&\mathsf{SD}((\langle\mathcal{G},\mathcal{X}\rangle,\mathcal{G}),(\mathcal{Y},\mathcal{G})) \\
&\leq \frac{1}{2}\sqrt[4]{\frac{|H|}{|G|}\left(\begin{array}{l}1 + \sum_{j=1}^{4}\sum_{i=j}^{4}\binom{4}{i}\left\{{i \atop j}\right\}\frac{|X|^{\underline{j}}|Y|^{i-j}}{(-1)^{4-i}|X|^i} \\ + \mathsf{EIF}(4)\binom{4}{4}\left\{{4 \atop 4}\right\}\frac{|X|^{\underline{4}}|Y|}{|X|^4}\end{array}\right)} \\
&\leq \frac{1}{2}\sqrt[4]{\frac{|H|}{|G|}\left(\begin{array}{l}1 + \sum_{j=1}^{4}\sum_{i=j}^{4}\binom{4}{i}\left\{{i \atop j}\right\}\frac{|X|^{\underline{j}}|Y|^{i-j}}{(-1)^{4-i}|X|^i} \\ + 15\frac{|Y|6^{\log_2|X|}}{|X|^4}\end{array}\right)} \qquad (9) \\
&= \frac{1}{2}\sqrt[4]{\frac{|H|}{|G|}\left(\begin{array}{l}1 + \sum_{i=1}^{4}\binom{4}{i}\frac{|X||Y|^{i-1}}{(-1)^{4-i}|X|^i} + \sum_{i=2}^{4}\binom{4}{i}\left\{{i \atop 2}\right\}\frac{|X|^{\underline{2}}|Y|^{i-2}}{(-1)^{4-i}|X|^i} \\ + \sum_{i=3}^{4}\binom{4}{i}\left\{{i \atop 3}\right\}\frac{|X|^{\underline{3}}|Y|^{i-3}}{(-1)^{4-i}|X|^i} + \frac{|X|^{\underline{4}}}{|X|^4} + 15\frac{|Y|6^{\log_2|X|}}{|X|^4}\end{array}\right)} \\
&= \frac{1}{2}\sqrt[4]{\frac{|H|}{|G|}\left(\begin{array}{l}-3 + 6\frac{|Y|}{|X|} - 4\frac{|Y|^2}{|X|^2} + \frac{|Y|^3}{|X|^3} + 6\frac{|X|^{\underline{2}}}{|X|^2} - 12\frac{|X|^{\underline{2}}|Y|}{|X|^3} \\ + 7\frac{|X|^{\underline{2}}|Y|^2}{|X|^4} - 4\frac{|X|^{\underline{3}}}{|X|^3} + 6\frac{|X|^{\underline{3}}|Y|}{|X|^4} + \frac{|X|^{\underline{4}}}{|X|^4} + 15\frac{|Y|6^{\log_2|X|}}{|X|^4}\end{array}\right)}
\end{aligned}$$

where Equation (9) follows from Lemma 31. ◀

We simplify the above theorem statement slightly by observing that for any constant c, $x^{\underline{c}}/x^c \approx 1$ when $x \gg c$.

▶ **Corollary 33.** *Let $H = \mathbb{F}_m^{\xi}$ be the IPE family (Definition 13) from $X = \mathbb{1}^{\xi}$ to $Y = \mathbb{F}_m$. Let $G \subseteq H$ and let $\mathcal{G}$ and $\mathcal{X}$ denote the uniform distributions over G and X. If $|Y| > 3$ and $|X| \gg 4$, then H is a strong $(\mathcal{X}, \mathcal{G}, \varepsilon)$-extractor per Definition 7, where*

$$\varepsilon \lesssim \frac{1}{2}\sqrt[4]{\frac{|H|}{|G|}\left(3\frac{|Y|^2}{|X|^2} + \frac{|Y|^3}{|X|^3} + 15\frac{|Y|3^{\log_2|X|}}{|X|^3}\right)}$$

Proof. Follows directly from Theorem 32. ◀

▶ **Corollary 34.** *Let $H = \mathbb{F}_m^\gamma$ be the IPE family (Definition 13) from $W = \mathbb{1}^\gamma$ to $Y = \mathbb{F}_m$. If $m > 3$, $\gamma \geq \xi$, $2^\xi \gg 4$, and $\eta > 0$, then H is a strong (ξ, η, ε)-biased-seed-OBF extractor per Definitions 9 and 12, where*

$$\varepsilon \lesssim \frac{1}{2}\sqrt[4]{\frac{m^\gamma}{2^\eta}\left(3\frac{m^2}{2^{2\xi}} + \frac{m^3}{2^{3\xi}} + 15\frac{m\cdot 3^\xi}{2^{3\xi}}\right)}$$

Furthermore, H is a strong $(\xi, \eta, \varepsilon')$-biased-seed-SAOBF extractor per Definitions 11 and 12, where

$$\varepsilon' \lesssim \frac{1}{2}\sqrt[4]{\frac{m^\gamma\binom{\gamma}{\xi}}{2^\eta}\left(3\frac{m^2}{2^{2\xi}} + \frac{m^3}{2^{3\xi}} + 15\frac{m\cdot 3^\xi}{2^{3\xi}}\right)}$$

Proof. We first observe that the function bounding ε in Corollary 33 is a concave function of $|G|^{-1}$, and thus we can apply Lemma 23. The corollary then follows from Lemmas 24 and 26. ◀

Finally, restricting the relationship between $|X|$ and $|Y|$ gives us the clean statement of the 4^{th}-root bound featured in the abstract.

▶ **Corollary 35.** *Let $H = \mathbb{F}_m^\xi$ be the IPE family (Definition 13) from $X = \mathbb{1}^\xi$ to $Y = \mathbb{F}_m$. Let $G \subseteq H$ and let $\mathcal{G}$ and $\mathcal{X}$ denote the uniform distributions over G and X. If $|X| \geq |Y| \geq |X|^{0.585}$, $|Y| > 3$, and $|X| \gg 4$, then H is a strong $(\mathcal{X}, \mathcal{G}, \varepsilon)$-extractor per Definition 7, where*

$$\varepsilon \lesssim \frac{2.1}{2}\sqrt[4]{\frac{|H|}{|G|}}\sqrt{\frac{|Y|}{|X|}}$$

Proof. Follows from Theorem 32 via the fact that $|Y|^3/|X|^3 \leq |Y|^2/|X|^2$ when $|X| \geq |Y| > 0$ and $|Y|3^{\log_2|X|}/|X|^3 \leq |Y|^2/|X|^2$ when $|Y| \geq |X|^{\log_2 3 - 1} > 0$. ◀

▶ **Corollary 1.** *Let $H = \mathbb{F}_m^\gamma$ be the IPE family (Definition 13) from $W = \mathbb{1}^\gamma$ to $Y = \mathbb{F}_m$. If $\gamma \geq \xi \geq \log_2 m \geq 0.585\xi$, $m > 3$, $2^\xi \gg 4$, and $\eta > 0$, then H is a strong (ξ, η, ε)-biased-seed-OBF extractor per Definitions 9 and 12, where*

$$\varepsilon \lesssim \frac{2.1}{2}\sqrt[4]{\frac{m^\gamma}{2^\eta}}\sqrt{\frac{m}{2^\xi}}$$

Furthermore, H is a strong $(\xi, \eta, \varepsilon')$-biased-seed-SAOBF extractor per Definitions 11 and 12, where

$$\varepsilon' \lesssim \frac{2.1}{2}\sqrt[4]{\frac{m^\gamma\binom{\gamma}{\xi}}{2^\eta}}\sqrt{\frac{m}{2^\xi}}$$

Proof. We first observe that the function bounding ε in Corollary 35 is a concave function of $|G|^{-1}$, and thus we can apply Lemma 23. The corollary then follows from Lemmas 24 and 26. ◀

6 Further Results from Matroid Theory in the Full Version

In the full version of this paper [12], we relate the *exact* multicollision probability of the IPE to the coboundary polynomial of the cube matroid.

▶ **Definition 36** (The Cube Matroid). *Let $M_n^{\square}$ be the matroid on ground set $\mathbb{1}^n$ with rank function $\mathrm{rank}_{M_n^{\square}}(A) = \dim(A)$. In other words, the rank function of $M_n^{\square}$ is defined identically to the rank function for matrices. Note that as with matrices, the rank function can be defined over any field, not necessarily $\mathbb{F}_2$. We leave the field over which rank is defined implicit unless it is relevant.*

▶ **Lemma 37.** *Let $H = \mathbb{F}_m^{\xi}$ be an IPE family (Definition 13) from $X = \mathbb{1}^{\xi}$ to $Y = \mathbb{F}_m$. Let $\overline{\chi}_{M_{i-1}^{\square}}$ be the coboundary polynomial of the cube matroid (Definition 36). The i-way multicollision probability of H for $i \in \mathbb{N}^+$ is exactly given by*

$$\mathsf{CP}(i) = \frac{m^{1-i}}{2^{\xi\cdot(i-1)}} \left(\frac{d}{d\alpha}\right)^{\xi} \overline{\chi}_{M_{i-1}^{\square}}(m, e^{\alpha})\Bigg|_{\alpha=0}$$

Then, we use the above result to compute a *second* version of the Elevated General Leftover Hash Lemma for the IPE.

▶ **Theorem 4.** *Let $H = \mathbb{F}_m^{\xi}$ be an IPE family (Definition 13) from $X = \mathbb{1}^{\xi}$ to $Y = \mathbb{F}_m$. Let $G \subseteq H$ and let $\mathcal{G}$ and $\mathcal{X}$ denote the uniform distributions over G and X. Let $\overline{\chi}_{M_{i-1}^{\square}}$ be the coboundary polynomial of the cube matroid (Definition 36). If $n \in \mathbb{N}^+$ is even, then H is a strong $(\mathcal{X}, \mathcal{G}, \varepsilon)$-extractor per Definition 7, where*

$$\varepsilon \leq \frac{1}{2}\sqrt[n]{\frac{|H|}{|G|}\left(1 + \sum_{i=1}^{n}\binom{n}{i}\frac{(-1)^{n-i}}{2^{\xi\cdot(i-1)}}\left(\frac{d}{d\alpha}\right)^{\xi}\overline{\chi}_{M_{i-1}^{\square}}(m, e^{\alpha})\Bigg|_{\alpha=0}\right)} \tag{10}$$

The first version of an IPE-specific EGLHL given in Lemma 29 was limited in its usefulness because it requires us to prove a concrete bound on $\mathsf{EIF}(j)$ for every $j \in [4, n]$, and we do not know how to prove such bounds for any $j > 4$. On the other hand, we know how to concretely compute the second version for any n, but only in $2^{O(n^2)}$ time. In the full version of this paper, we perform this exponential computation for $n \in \{4, 6, 8\}$ and report the concrete bounds that result. Additionally, we asymptotically upper-bound the right hand side of Equation (10). This yields the following theorem and its corollary:

▶ **Theorem 38.** *Let $H = \mathbb{F}_m^{\xi}, X = \mathbb{1}^{\xi}, Y = \mathbb{F}_m$, and $G \subseteq H$ be as in Theorem 4, let $\mathcal{G}$ and $\mathcal{X}$ be the uniform distributions over G and X, and assume that $m \leq 2^{(1-\epsilon)\cdot\xi}$ for some $0 < \epsilon < 1$. For any even constant $n \in \mathbb{N}^+$, as $m \to \infty$, H is a strong $(\mathcal{X}, \mathcal{G}, \varepsilon)$-extractor per Definition 7, where*

$$\varepsilon \leq \frac{1}{2}\sqrt[n]{\frac{|H|}{|G|}\left((n-1)!!\cdot 2^{-n\cdot\epsilon\cdot\xi/2} + O\left(2^{-(n/2+1)\cdot\epsilon\cdot\xi} + 2^{-\xi} + \left(\frac{3}{4}\right)^{\xi} 2^{-\epsilon\cdot\xi}\right)\right)}.$$

▶ **Corollary 2.** *Let $H = \mathbb{F}_m^{\gamma}$ be the IPE family (Definition 13) from $W = \mathbb{1}^{\gamma}$ to $Y = \mathbb{F}_m$. If $\gamma \geq \xi > 0$, $\eta > 0$, and $m \leq 2^{(1-\epsilon)\cdot\xi}$ for some $0 < \epsilon \leq 0.83/(n-2)$, then for any even constant integer $n \geq 4$, as $m \to \infty$, H is a strong (ξ, η, ε)-biased-seed-OBF extractor per Definitions 9 and 12, where*

$$\varepsilon \leq \frac{n-1}{2}\sqrt[n]{\frac{m^{\gamma}}{2^{\eta}}}\sqrt{2^{-\epsilon\cdot\xi}}\,(1 + o(1))$$

Furthermore, H is a strong $(\xi, \eta, \varepsilon')$-biased-seed-SAOBF extractor per Definitions 11 and 12, where

$$\varepsilon' \leq \frac{n-1}{2}\sqrt[n]{\frac{m^{\gamma}\binom{\gamma}{\xi}}{2^{\eta}}}\sqrt{2^{-\epsilon\cdot\xi}}\,(1 + o(1))$$

References

1 Michael Ben-Or and Nathan Linial. Collective coin flipping. *Adv. Comput. Res.*, 5:91–115, 1989.

2 Benny Chor and Oded Goldreich. Unbiased bits from sources of weak randomness and probabilistic communication complexity. In *26th Annual Symposium on Foundations of Computer Science (sfcs 1985)*, pages 429–442, 1985. doi:10.1109/SFCS.1985.62.

3 Benny Chor and Oded Goldreich. Unbiased bits from sources of weak randomness and probabilistic communication complexity. *SIAM Journal on Computing*, 17(2):230–261, 1988. doi:10.1137/0217015.

4 Richard Cleve. Limits on the security of coin flips when half the processors are faulty (extended abstract). In *Proceedings of the 18th Annual ACM Symposium on Theory of Computing (STOC)*, pages 364–369, 1986. doi:10.1145/12130.12168.

5 Ran Cohen, Jack Doerner, Yashvanth Kondi, and abhi shelat. Secure multiparty computation with identifiable abort via vindicating release. In *Advances in Cryptology - CRYPTO 2024 - 44th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 18-22, 2024, Proceedings, Part VIII*, 2024. doi:10.1007/978-3-031-68397-8_2.

6 Jean Paul Degabriele, Jan Gilcher, Jérôme Govinden, and Kenneth G. Paterson. Sok: Efficient design and implementation of polynomial hash functions over prime fields. In *IEEE Symposium on Security and Privacy, SP 2024, San Francisco, CA, USA, May 19-23, 2024*, pages 3128–3146. IEEE, 2024. doi:10.1109/SP54263.2024.00132.

7 Jack Doerner, Iftach Haitner, Yuval Ishai, and Nikolaos Makriyannis. From OT to OLE with subquadratic communication. In *Proceedings of the 2025 ACM SIGSAC Conference on Computer and Communications Security, CCS 2025, Taipei, Taiwan, October 13-17, 2025*, pages 2249–2263. ACM, 2025. doi:10.1145/3719027.3765225.

8 Jack Doerner, Yashvanth Kondi, Eysa Lee, and abhi shelat. Secure two-party threshold ECDSA from ECDSA assumptions. In *Proceedings of the 39th IEEE Symposium on Security and Privacy (S&P)*, 2018.

9 Jack Doerner, Yashvanth Kondi, Eysa Lee, and abhi shelat. Threshold ECDSA from ECDSA assumptions: The multiparty case. In *2019 IEEE Symposium on Security and Privacy, SP 2019, San Francisco, CA, USA, May 19-23, 2019*, pages 1051–1066. IEEE, 2019. doi:10.1109/SP.2019.00024.

10 Jack Doerner, Yashvanth Kondi, Eysa Lee, and abhi shelat. Threshold ECDSA in three rounds. In *Proceedings of the 45th IEEE Symposium on Security and Privacy (S&P)*, 2024.

11 Jack Doerner, Yashvanth Kondi, Eysa Lee, abhi shelat, and LaKyah Tyner. Threshold BBS+ signatures for distributed anonymous credential issuance. In *Proceedings of the 44th IEEE Symposium on Security and Privacy (S&P)*, 2023.

12 Jack Doerner and Lawrence Roy. Tighter bounds for the oblivious bit-fixing inner product extractor on biased seeds. Cryptology ePrint Archive, Paper 2026/654, 2026. URL: https://eprint.iacr.org/2026/654.

13 Dean Doron and Ori Fridman. Bit-Fixing Extractors for Almost-Logarithmic Entropy. In Alina Ene and Eshan Chattopadhyay, editors, *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2025)*, volume 353 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 33:1–33:19, Dagstuhl, Germany, 2025. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.APPROX/RANDOM.2025.33.

14 Shimon Even, Oded Goldreich, and Abraham Lempel. A randomized protocol for signing contracts. *Communications of the ACM*, 28(6):637–647, 1985. doi:10.1145/3812.3818.

15 Niv Gilboa. Two party RSA key generation. In *Annual International Cryptology Conference (CRYPTO)*, pages 116–129, 1999. doi:10.1007/3-540-48405-1_8.

16 Iftach Haitner, Nikolaos Makriyannis, Samuel Ranellucci, and Eliad Tsfadia. Highly efficient ot-based multiplication protocols. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques (EUROCRYPT)*, pages 180–209, 2022. doi:10.1007/978-3-031-06944-4_7.

17 Russell Impagliazzo, Leonid A. Levin, and Michael Luby. Pseudo-random generation from one-way functions (extended abstracts). In David S. Johnson, editor, *Proceedings of the 21st Annual ACM Symposium on Theory of Computing, May 14-17, 1989, Seattle, Washington, USA*, pages 12–24. ACM, 1989. doi:10.1145/73007.73009.

18 Yuval Ishai, Manoj Prabhakaran, and Amit Sahai. Founding cryptography on oblivious transfer - efficiently. In David A. Wagner, editor, *Advances in Cryptology - CRYPTO 2008, 28th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 17-21, 2008. Proceedings*, volume 5157 of *Lecture Notes in Computer Science*, pages 572–591. Springer, 2008. doi:10.1007/978-3-540-85174-5_32.

19 Marcel Keller, Emmanuela Orsini, and Peter Scholl. MASCOT: faster malicious arithmetic secure computation with oblivious transfer. In *ACM SIGSAC Conference on Computer and Communications Security (CCS)*, pages 830–842, 2016. doi:10.1145/2976749.2978357.

20 Chia-Jung Lee, Chi-Jen Lu, Shi-Chun Tsai, and Wen-Guey Tzeng. Extracting randomness from multiple independent sources. *IEEE Trans. Inf. Theory*, 51(6):2224–2227, 2005. doi:10.1109/TIT.2005.847746.

21 Moni Naor and Benny Pinkas. Oblivious polynomial evaluation. *SIAM Journal on Computing*, 35(5):1254–1281, 2006. doi:10.1137/S0097539704383633.

22 Noam Nisan and David Zuckerman. Randomness is linear in space. *J. Comput. Syst. Sci.*, 52(1):43–52, 1996. doi:10.1006/JCSS.1996.0004.

23 M. O. Rabin. How to exchange secrets by oblivious transfer. TR-81, Harvard, 1981.

24 Salil P. Vadhan. Pseudorandomness. *Found. Trends Theor. Comput. Sci.*, 7(1-3):1–336, 2012. doi:10.1561/0400000010.

25 Umesh Vazirani. Towards a strong communication complexity theory or generating quasi-random sequences from two communicating slightly-random sources. In *Proceedings of the Seventeenth Annual ACM Symposium on Theory of Computing*, STOC '85, pages 366–378, New York, NY, USA, 1985. Association for Computing Machinery. doi:10.1145/22145.22186.

Limits on the Power of Private Constrained PRFs

Mengda Bi
Tsinghua University, Beijing, China

Yaohua Ma
Tsinghua University, Beijing, China

Chenxin Dai
Tsinghua University, Beijing, China

Abstract

Private constrained PRFs are constrained PRFs where the constrained key hides information about the predicate circuit. Although there are many constructions and applications of PCPRF, its relationship to basic cryptographic primitives, such as one-way functions and public-key encryptions, has been unclear. For example, we don't know whether one-way functions imply PCPRFs for general predicates, nor do we know whether 1-key secure PCPRF for all polynomial-sized predicates imply public-key primitives such as public-key encryption and secret-key agreement.

In this work, we prove the black-box separation between a 1-key secure PCPRF for any predicate and a secret-key agreement, which is the first black-box separation result about PCPRF. Specifically, we prove that there exists an oracle relative to which 1-key secure PCPRFs exist while secret-key agreement does not. Our proof is based on the simulation-based technique proposed by Impagliazzo and Rudich (STOC 89). The main technical challenge in generalizing the simulation-based technique to PCPRF is the issue of *unfaithfulness* of Eve's simulation to the real world because our oracle is more complicated than a random oracle. We introduce a new technique which we call the "weighting" technique and show how to leverage it to circumvent the issue of unfaithfulness in the proof framework of Impagliazzo and Rudich.

2012 ACM Subject Classification Theory of computation → Cryptographic primitives

Keywords and phrases Black-box Separations, Private Constrained PRFs, Key Agreement

Digital Object Identifier 10.4230/LIPIcs.ITC.2026.2

Related Version *Full Version*: `https://eprint.iacr.org/2025/1196.pdf`

1 Introduction

1.1 Private Constrained Pseudorandom Functions

Constrained pseudorandom functions (CPRFs) are useful building blocks in cryptography [6, 21, 8]. Suppose $F : \mathcal{K} \times \mathcal{X} \to \mathcal{Y}$ is a pseudorandom function (PRF), F is called a CPRF if it additionally exhibits the following functionality: given a master key $k \in \mathcal{K}$, and a binary predicate $c : \mathcal{X} \to \{0, 1\}$, one can derive a constrained key $k[c]$. $k[c]$ can be used to evaluate $F(k, x)$ if $c(x) = 1$. However, if $c(x) = 0$, $k[c]$ should not reveal non-trivial information of $F(k, x)$.

Since the concept of CPRF was first proposed, there have been constructions of CPRF for various predicate classes under various assumptions. For example, CPRF for the simplest predicate, a point indicator function, also known as puncturable PRF, is shown to be constructable from one-way functions (OWFs) using the classical GGM tree-based PRF [18, 6, 8, 21]. Currently, the most expressive predicate family that can be constructed from OWF are the CNF predicates [13]. Other constructions require structural assumptions such as LWE [11], DDH [2], pairing [6, 5], or even stronger primitives such as indistinguishability obfuscation (iO) [7, 19, 5, 3, 13].

A CPRF can further be *private*[5], if the constrained key k_c reveals no information about the predicate c. A private CPRF is also abbreviated as PCPRF. The notion of privacy varies depending on how many constrained keys the adversary is allowed to obtain. Multi-key secure PCPRF for $\mathsf{P/poly}$ predicates is only known to be constructible from iO [5]. In fact, even 2-key secure PCPRF for a predicate class implies iO for the same circuit class [12]. To the best of our knowledge, without iO, multi-key privacy is only achievable for left-right predicates in the random oracle model from bilinear maps [6]. In contrast, PCPRF with 1-key privacy for all polynomial-sized predicates is known to be constructable from the LWE assumption [10].

Even with just 1-key privacy, PCPRF still has wide applications in cryptography because of its powerful predicate-hiding ability. Known applications include searchable encryption, watermarking, deniable encryption and others in [5], private information retrieval [27, 22], and 1-key secure private-key functional encryption [12]. However, even though PCPRFs seem to be powerful, we still do not know how to construct secret-key agreement – the fundamental primitive in Cryptomania – from 1-key secure PCPRFs. Therefore, it is natural to ask whether 1-key secure PCPRFs fall within Minicrypt or Cryptomania.

1.2 Black-Box separation

In cryptography, it is almost impossible to directly prove separation results in the form of "the existence of a certain primitive P is not sufficient to construct another primitive Q". Instead, great progress has been made in proving the so-called "black-box separations". A black-box separation refers to a cryptographic impossibility result showing that Q cannot be achieved using only black-box access to P. In other words, it demonstrates that there exists no efficient reduction from Q to P when the reduction is restricted to using P as a black-box, without any knowledge of its internal structure. This concept is fundamental in understanding the limitations of cryptographic primitives and provides insights into the relationship between different cryptographic primitives.

One of the most significant black-box separation results is the seminal work of Impagliazzo and Rudich [20]. It shows that OWF cannot be used to construct secret-key agreement (KA) in a black-box way, employing a relativizing technique in which an oracle is constructed such that OWFs exist but KA does not, relative to this oracle. Their result naturally establishes a black-box separation between the following two classes: (1) Minicrypt: comprising primitives that are implied by OWFs, such as pseudorandom generators and pseudorandom functions; (2) Cryptomania: comprising all primitives that imply KA in a black-box way, such as public key encryption and oblivious transfer.

Inspired by [20], researchers have developed black-box separation techniques to obtain various impossibility results, including [25, 28, 15, 16]. In this work, we mainly focus on the impossibility results concerning KA. Significant progress has been made in this field by many researchers: [4] improves the efficiency of the adversary in [20], and [9] simplifies the analysis for perfectly complete KA protocols.

1.3 Motivations and Questions

Although significant work has been devoted to the study of PCPRF, its relationship with OWFs and cryptographic primitives in Cryptomania remains unclear. Firstly, we do not know how to use OWFs to construct PCPRFs for more expressive predicate classes than

CNF (even providing only 1-key security). Secondly, we do not know how to use 1-key secure PCPRFs to construct any primitives in Cryptomania.[1]

Naturally, we aim to establish black-box separation results. Specifically, we propose the following questions:

- Open question 1: *Can we establish or refute black-box separation results between OWFs and more expressive PCPRFs?*
- Open question 2: *Can we establish or refute black-box separation results between 1-key secure PCPRFs and primitives in Cryptomania?*

1.4 Our Result

In this work, we prove the black-box separation between 1-key PCPRF for arbitrary predicate classes and KA, which gives a positive result regarding the second open question. Concretely, we prove the following theorem:

▶ **Theorem 1** (Informal). *There exists an oracle relative to which 1-key PCPRF exists while any KA protocol can be broken. This implies that it is impossible to construct KA using PCPRF in a black-box manner.*

Here, it is worth mentioning the predicate family of the PCPRF relative to our oracle. We use the setting of oracle-aided circuits proposed in [1] in which the circuits are only allowed to access a random oracle H but not the rest of the oracle. As stated in [1], this setting characterizes most of the non-black-boxes constructions such as punctured programming approach of Sahai and Waters [26] and its variants. This is further discussed in Section 2.4.

Our proof framework follows a similar approach to [20]. However, proving our result based on the techniques in [20] is not straightforward. The main challenge is explained in Section 2, where we introduce a new technique to overcome this issue.

2 Technical Overview

In this section, we first review the techniques from [20]. Then, we identify the main reason why the proof in [20] does not directly apply to our problem. Finally, we briefly outline our solution to this obstacle.

2.1 Techniques of [20]

In [20], it is proved that there exists an oracle relative to which a one-way function exists, but KA does not. The oracle consists of two parts: a random function f and a PSPACE-complete oracle. Clearly, f itself is one-way even when the PSPACE-complete oracle is present. Now we review how the attacker Eve in [20] breaks the KA protocol under such an oracle. Suppose Alice and Bob are the two parties of a KA protocol; Eve tries to find all the intersection queries – queries made by both Alice and Bob. It is shown in [20] that finding all the intersection queries suffices to break the KA protocol. During the attack, Eve maintains a list of query-answer pairs, recording the queries and oracle answers she has received so far. Eve's algorithm repeats the following two phases (which is called one "segment") polynomially many times after each message is sent.

[1] In [12] authors construct indistinguishability obfuscators for arbitrary circuits from 2-key secure PCPRF. Since we can construct public key encryption from iO and OWF [26], we already obtain a **non-black-box** construction of primitives in Cryptomania from 2-key secure PCPRF.

- **Simulation Phase**: Eve uniformly samples a simulated view of Alice that is consistent with the transcript and Eve's list (if Bob has just sent a message, she samples Bob's run instead) using the PSPACE-complete oracle. Any simulated query not on Eve's list receives a random answer. In general, such a random answer may differ from the actual one.
- **Update Phase**: For each query in the simulated run, Eve queries the actual oracle f and adds the query-answer pair to her list.

Suppose that Alice has just sent a message. We temporarily fix the view of Bob, the transcript and Eve's list, while only Alice's view is considered unknown. Let BPQ denote the set of "Bob's private queries", which are the queries made by Bob but not Eve. Clearly, the size of BPQ is bounded by the total number of queries made by Bob. We classify each simulated view of Alice into the following two types:

1. **Hit**: this simulated view contains at least 1 query in BPQ.
2. **Miss**: this simulated view does not contain any queries in BPQ.

By the above definition, each hit contains at least one query in BPQ. Then in the following update phase, these queries will be made by Eve. This means the size of BPQ decreases by at least 1 in this segment. Since the size of BPQ is bounded, the number of hits is also bounded. Thus, as long as Eve repeats the simulation sufficiently many times, most of the simulations will be misses.

The key to the proof in [20] is that, conditioned on a miss, the probability distribution of Alice's view in the simulation matches the distribution of her actual view. In other words, Eve's simulation of Alice's view must be faithful to the real one, given a "miss". Since the transcript and Bob's view is fixed, the query that will be made by Bob in the next round is also fixed. If Bob, in the next round, makes an intersection query with high probability, a significant portion of Alice's simulated view would already include this query. As a result, Eve has a high probability of identifying the intersection query. Therefore, even before a query becomes an intersection query, it is highly likely that Eve predicts it and adds it to her list.

2.2 Limitations of [20]

Now we consider how to apply the technique from [20] to our problem of separating 1-key secure PCPRF and KA. For now, we only consider predicates that are not allowed to access the oracle. In other words, they are just binary circuits in the common sense. As it turns out, this is already non-trivial.

The first step is to construct an oracle relative to which PCPRF exists. A natural approach is to define the oracle in the shape of a PCPRF. Let K, K', C be the spaces of master keys, constrained keys and predicates, respectively, and let X, Y be the spaces of inputs and outputs of the PRFs. We then define the oracle Γ to consist of the following components:

- $\mathsf{F} : K \times X \to Y$ serves as the PRFs for master keys;
- $\mathsf{f} : K' \times X \to Y$ serves as the PRFs for constrained keys;
- $\mathsf{Keygen} : K \times C \to K'$ serves as the constrained key generation algorithm (henceforth we denote $k[c] = \mathsf{Keygen}(k, c)$).
- PSPACE-complete oracle, as usual.

The above oracle construction actually implements a multiple-key secure PCPRF. Since 2-key secure PCPRF and OWF already implies KA, we cannot hope to prove that KA does not exist under such an oracle. In order to capture one-key security, a natural idea is to add

another weakening oracle R with the following functionality: given two different constrained keys $k[c_1]$ and $k[c_2]$ of the same master key k, $\mathsf{R}(k[c_1], k[c_2])$ will return k, c_1, c_2. In any other cases, R returns $\perp$. For the R oracle to be well-defined, every element $k' \in K'$ must correspond to **at most one** pair $(k, c) \in K \times C$ satisfying the relation $k' = k[c]$. In other words, the oracle Keygen should be injective.

The correctness of PCPRF states that $\mathsf{F}(k, x) = \mathsf{f}(k[c], x)$ for $c(x) = 1$. This correctness property naturally divides all possible F, f queries into many equivalent classes, and the answer for each equivalent class is sampled uniformly and independently from each other.

Now, we can see the first limitation of [20]. The major idea behind [20] is that: the only way that two parties achieve key agreement in the presence of Eve is essentially by making the same queries to the oracle. However, when we are considering a more sophisticated oracles like Γ instead of a simple random oracle, Alice and Bob can reach an agreement even if they do not make the same queries at all. Consider the following protocol:

- Alice samples $k \in K, c \in C, x \in X$ such that $c(x) = 1$. Then Alice queries $k' = k[c] = \mathsf{Keygen}(k, c)$ and sends k', x to Bob. The output of Alice is $\mathsf{F}(k, x)$.
- Bob receives k', x from Alice and outputs $\mathsf{f}(k', x)$.

The correctness of this key agreement protocol is guaranteed by the correctness of PCPRF: $\mathsf{F}(k, x) = \mathsf{f}(k[c], x)$. However, they make completely different queries.

The above issue is common especially when the internal dependencies exists among the answers of different oracle queries. A common solution is to consider the input elements separately, such as [23, 1]. Specifically, instead of intersection queries, we consider "intersection elements". We say an element in $K \cup K' \cup C \cup X$ appears in a party's view if this element is a part of the input or output of a query in this party's view. (Here, we do not consider elements in Y because they cannot appear as the input of a query.) Analogous to intersection queries, intersection elements are elements that appears in both Alice and Bob's views. Our attacker Eve's goal is to find all the intersection elements. In the above example, Alice's view contains k, c, x, k'; Bob's view contains k', x. Thus, k', x are both intersection elements. If Eve finds k', x, she can make the query $\mathsf{f}(k', x)$ and the answer would be the agreed key.

However, the above problem is not the major challenge we faced trying to apply the technique of [20] to our case. The main reason is that the distribution of Alice's view will depend on Bob's view, even conditioned on "miss". Since Bob's view is invisible to Eve, such dependency cannot be captured by Eve, which causes Eve's simulation to be unfaithful to the real Alice's view. In the following, we use 2 simple examples to demonstrate why the distribution of Alice's simulated view can be unfaithful to Alice's actual view even conditioned on "miss".

- **Example 1**: Consider the following 2-pass protocol. In the first pass, Bob randomly samples $k_B \in K$ and queries $\mathsf{F}(k_B, i) = y_B^i$ for $i = 1, 2, \ldots, n$ where n is a sufficiently large polynomial (such that $|K'|^{-1} > |Y|^{-n}$). No communication between Alice and Bob is made in this round. In the next pass, Alice randomly samples $k'_A \in K'$ and queries $\mathsf{f}(k'_A, i) = y_A^i$ for $i = 1, 2, \ldots, n$. Now we fix Bob's view, i.e. k_B and $y_B^i, i \in [n]$. Let E be the event that $y_A^i = y_B^i$ for all $i \in [n]$. In the real execution, as long as $k'_A = k_B[c]$ for some $c \in C$ such that $c(x) = 1$ for $x \in [n]$, we would have $y_A^i = y_B^i$ for all $i \in [n]$. The probability of Alice picking $k' = k[c]$ for such a c is at least $|K'|^{-1}$. Thus, the probability of E happening in the real execution is at least $|K'|^{-1}$. However, since Eve does not have Bob's view, when simulating Alice's view, each y_A^i are randomly sampled from Y. the probability that $y_A^i = y_B^i$ for all $i \in [n]$ in Eve's simulation is $|Y|^{-n}$. Thus, as long as n is sufficiently large, we would notice that the event E is more likely to happen in the real execution than in Eve's simulation, which means that Eve's simulation is unfaithful to Alice's actual view.

- **Example 2**: Consider the following 2-pass protocol. In the first pass, Bob randomly samples $k_B \in K, c_B \in C$ and queries $\mathsf{Keygen}(k_B, c_B) = k'_B$. In the second pass, Alice randomly samples $k_A \in K$ and $c_A \in C$ and queries $\mathsf{Keygen}(k_A, c_A) = k'_A$. Fix the view of Bob. Let E be the event that $k_A \neq k_B$ and $k'_A = k'_B$. Because Keygen is sampled from injective functions, the probability of E is 0 in the real execution. However, in the distribution of Eve's simulated view of Alice, event E has a non-zero probability if Eve does not know the relation $k'_B = k_B[c_B]$. This means that Eve's simulation is unfaithful to Alice's actual view.

Examples 1 and 2 show that the distribution of Alice's actual view and the distribution of Eve's simulated Alice's view might have very different probabilities on some specific "miss" views. In other words, the distribution of Eve's simulation is unfaithful to Alice's actual view even conditioned on "miss". Even with the weakening oracle R, these two examples can still happen. It should be noted here that these two examples are merely illustrations of the unfaithfulness issue. When we are considering arbitrary protocols, the situation might be more complicated.

To overcome this issue of unfaithfulness, we will modify the proof of [20]. In the proof of [20], the most important step is to prove that Eve's simulation is faithful to Alice's actual view conditioned on "miss", for any **fixed** tuple of Bob's view, Eve's view and the transcript so far. We denote such tuple as G. In our proof, we will analyze the statistical distance between Eve's simulation and Alice's actual view, conditioned on "miss". We show that the expectation of this statistical distance is negligible over a specific distribution of G, which will be enough under the framework of [20]. To do so, we introduce a new technique which we call the **weighting** technique. This is further discussed in the next section.

2.3 Our Approach

In this section we provide a high level overview of our approach. We first give a simplified version of our oracle.

2.3.1 Oracle description

Let K, K', C, X, Y be the space of master keys, constrained keys, predicates, the input and the output of the PCPRF, respectively. The size of these spaces are set as follows: $|K| = |C| = |X| = 2^\kappa, |K'| = 2^{\kappa^2}, |Y| = 2^{\kappa^3}$. Our oracle contains two layers. The first layer contains a random oracle $\mathsf{H} : Z = \{0,1\}^* \to \{0,1\}$ and a PSPACE-complete oracle. The elements in C are viewed as P.P.T.Ms that have access to the first layer. The second layer consists of the following components:

- $\mathsf{F} : K \times X \to Y$ is uniformly sampled from all functions.
- $\mathsf{Keygen} : K \times C \to K'$ is sampled from all injective functions. Henceforth we denote $k[c] = \mathsf{Keygen}(k, c)$.
- $\mathsf{f} : K' \times X \to Y$ is defined as follows: if $k' = k[c]$ and $c(x) = 1$, then $\mathsf{f}(k', x) = \mathsf{F}(k, x)$. Otherwise it is randomly sampled from Y.
- $\mathsf{R} : (K \cup K') \times K' \to C \cup (K \times C \times C) \cup \{\perp\}$ is defined as follows: if the input is $(k, k[c])$ for some $k \in K, c \in C$, then R returns c; if the input is $(k[c_1], k[c_2])$ for some $k \in K, c_1, c_2 \in C$ and $c_1 \neq c_2$, R returns (k, c_1, c_2); in any other cases, R returns $\perp$ as a sign of rejection.

We denote the whole oracle as $\Gamma = (\mathsf{H}, \mathsf{F}, \mathsf{Keygen}, \mathsf{f}, \mathsf{R})$. Here, we remark that, in our formal definition, the oracle is defined in a more complicated way: the second layer of Γ is sampled for each input length κ. For simplicity, we only consider a fixed input length in this overview.

2.3.2 The Weighting Technique

Before proceeding to our attacking algorithm, we first introduce the weighting technique that is used to address the unfaithfulness issue raised in the previous section. We start by redefining the concept of views by introducing artificial views that might contain collisions of Keygen. Generally speaking, a view is what can be seen by a party (or parties). It includes the randomness of the party, the transcript and the query answer pairs made by the party. Recall that the Keygen oracle is sampled from all injective functions, collisions will never happen in a view. However, we allow a view to contain collisions such as $k_1[c_1] = k_2[c_2] = k'$ where $k_1 \neq k_2$. (Here, by collisions, we do not specifically mean the collision of Keygen queries. R queries can also be a part of collisions. For example, the two query-answer pairs such as $\mathsf{Keygen}(k_1, c_1) = k', \mathsf{R}(k_2, k') = c_2$ also constitute a collision.) That being said, we do not wish a view to contain arbitrary collisions. We say a view V is "valid" if it satisfies the following 2 properties:

1. If two master keys k_1 and k_2 collide at k', that is $k_1[c_1] = k_2[c_2] = k'$ for some c_1, c_2, then k_1 and k_2 will never be a part of another collision in V.
2. There are no self-collisions, i.e. $k[c_1] = k[c_2]$ for $c_1 \neq c_2$.

To better understand the first property, we here give two counterexamples which violate it: (1), one master key simultaneously collides with two other master keys: $k_1[c_1] = k_2[c_2], k_1[c_1'] = k_3[c_3]$; (2), two master keys collide at two different constrained keys: $k_1[c_1] = k_2[c_2], k_1[c_1'] = k_2[c_2']$.

We then define the weight of a valid view. Given a valid view V and a specific instantiation H of the oracle H, define $w_H(V)$ as follows:

- If the H queries made in V are inconsistent with H, $w_H(V) = 0$.
- If the correctness of PCPRF (with H instantiated as H) is violated in V, $w_H(V) = 0$.
- Otherwise, define $w_H(V)$ as follows: each different constrained relation $k[c] = k'$ in V contributes a factor of $|K'|^{-1} = 2^{-\kappa^2}$; each different F or f queries contribute a factor of $|Y|^{-1} = 2^{-\kappa^3}$ However, those F or f queries that can be deduced to be the same from the correctness of PCPRF only count as one contribution. $w_H(V)$ is defined to be the product of all contributions.

Then, the weight of a view V is defined as $w(V) = \frac{1}{2^M} \sum_H w_H(V)$. Here, the summation is taken over all possible instantiations H. M is the size of the possible input space of H. Indeed, the input of H can be arbitrarily long. However, since all parties are P.P.T.Ms, they cannot make H queries with arbitrarily long inputs. Therefore we can view the input space of H as finite. 2^M is the number of possible instantiations of H, which means that $w(V)$ can be viewed as the average value of $w_H(V)$.

We now define two distributions $\mathcal{D}$ and $\mathcal{D}'$ over all valid views. Define $\mathcal{D}$ as the distribution of views generated by first randomly sampling the randomness of all three parties (Alice, Bob and Eve) and oracle Γ, then executing the protocol. $\mathcal{D}'$ is a distribution over all valid views. The probability of a valid view V in $\mathcal{D}'$ is proportional to $w(V)$. Then, we will prove that the statistical distance between $\mathcal{D}$ and $\mathcal{D}'$ is negligible, which is a key lemma in our analysis.

Our Attacking Algorithm

Now we consider how to break KA protocols relative to this oracle. Our attacker Eve is similar to [20]. She locally maintains a list of elements in $K \cup K' \cup C \cup X \cup Z$ and repeats the two phases (i.e. a segment) mentioned in Section 2.1 after each message is sent. Suppose Alice speaks in round $r-1$ and Bob speaks in round r. Then Eve's algorithm in round r will repeat the following two phases sufficiently many times:

- **Simulation Phase**: Eve samples a view of Alice up until the end of round $r-1$, denoted as Sim_A. Sim_A should be consistent with Eve's view and the transcript so far. Furthermore, Sim_A combined with Eve's view V_E (denoted as $\mathsf{Sim}_A \cup V_E$) should form a collision-free valid view. The probability that Sim_A is sampled is proportional to $w(\mathsf{Sim}_A \cup V_E)$.
- **Update Phase:** Eve will add all the elements in the sampled view Sim_A to her list. Then, Eve will try to find out all the information related to the elements in her list with the help of R.

We now briefly explain why our attacking algorithm addresses the unfaithfulness issue raised in Section 2.2. We will show that, for any fixed Bob and Eve's view V_B, V_E, if Sim_A is a "miss" simulation, that is, the intersection elements between Sim_A and V_B all appear in V_E, then the combined view $\mathsf{Sim}_A \cup V_E \cup V_B$ is a valid view which might contain collisions as discussed before. This is exactly why we allow a valid view to contain collisions. Furthermore, we will show that $w(\mathsf{Sim}_A \cup V_E)$ is proportional to $w(\mathsf{Sim}_A \cup V_E \cup V_B)$. Recall that $w(\mathsf{Sim}_A \cup V_E)$ is proportional to the probability that Sim_A is sampled by Eve. By the definition of $\mathcal{D}'$, $w(\mathsf{Sim}_A \cup V_E \cup V_B)$ is proportional to the probability of Alice's view being Sim_A in $\mathcal{D}'$ conditioned on V_B , V_E and the transcript so far. This means, Eve's simulation is actually faithful to Alice's view in $\mathcal{D}'$, conditioned on "miss". After that, we use the result $\mathcal{D} \approx \mathcal{D}'$ to show that Eve's simulation is also faithful to Alice's view in $\mathcal{D}$ with high probability. Then, we can apply the technique of [20] to our problem.

2.4 More on Related Work

Perfect Correctness vs. Imperfect Correctness

In this study, we examine key agreement (KA) protocols where Alice and Bob are allowed to output different keys, a characteristic referred to as **imperfect correctness**. In contrast, KA protocols with **perfect correctness** requires Alice and Bob to always output the same key. When trying to obtain black-box separation results between KA and other primitives, the proof is usually simpler if we restrict the KA protocols to be perfectly correct. For example, when considering the black box separation between OWFs and perfectly correct KA, there exists a proof [9, Section 4.1] much simpler than [20, 4]. There also exist impossibility results only for perfectly correct key-agreement, such as [1, 14].

However, in our case, the proof cannot be greatly simplified if we consider perfectly correct KA protocols. The reason is as follows: under our oracle, Eve's simulated view might not be possible in the real world, as shown in Example 2 in Section 2.2. In these impossible views, Alice and Bob might output different keys. Thus, perfect correctness does not help.

Black-Box Separations Involving Injective Primitives

The key-generation oracle Keygen is defined to be a random injective function. This raises the issue stated in Example 2 discussed in Section 2.2. Here, we acknowledge that similar challenges have appeared when dealing with injective primitives (e.g., injective OWFs [24], TDFs [17]), albeit with different technical details. However, their techniques are not sufficient for solving the problems we have met, because their problems are quite different from the problem we face.

Oracle-Aided Circuits

The predicate class in our work is defined as P.P.T.M.s with access to a lower level random oracle H and the PSPACE-complete oracle. We acknowledge that this is exactly the same as the oracle aided circuits proposed by [1]. This setting rules out constructions of KA in which the predicates might use the lower level primitives (such as OWF) in a non-black box way. We clarify that even in this setting, a 2-key secure PCPRF suffices to imply iO, and sequentially PKE via the punctured programming approach of Sahai and Waters [26]. However, this does not rule out constructions in which the code of the predicates might use a part of PCPRF itself in a non-black box way.

2.5 Paper Organization

The rest of our paper is organized as follows. In Section 3, we give a formal definition of PCPRF. In Section 4, we give a formal definition of our oracle Γ and prove that 1-key secure PCPRF exists under such an oracle Γ. In Section 5, we provide formal definitions of secret-key agreement protocols and views, then define the distribution $\mathcal{D}$. Section 6 is the foundation of our technique. In this section, we give the formal definition of valid views, the weights of valid views, and the distribution $\mathcal{D}'$. Then, we prove our key lemma $\mathcal{D} \approx \mathcal{D}'$. In Section 7, we present the algorithm of Eve that breaks any KA protocol, and our main black-box separation result.

3 Preliminaries

A non-negative function $\mu : \mathbb{N}^* \to \mathbb{R}^+ \cup \{0\}$ is called **negligible** if $\mu(n) = O\left(\frac{1}{n^c}\right)$ for any constant $c \in \mathbb{N}^*$, denoted by $\mu = \mathrm{negl}(n)$.

In this work, we use a notion stronger than negligible: a non-negative function $\mu : \mathbb{N}^* \to \mathbb{R}^+ \cup \{0\}$ is **super-negligible** if $\mu(n) = O\left(\exp\left(-n^{1+\epsilon}\right)\right)$ for some constant $\epsilon > 0$, denoted by $\mu = \mathrm{snegl}(n)$. In the proof, we will use the following fact: $\mathrm{snegl}(\log n) = \mathrm{negl}(n)$.

We define two notations for simplifying our proof: $a(n) \approx b(n)$ means that $\left|\frac{a(n)}{b(n)} - 1\right| = \mathrm{negl}(n)$, and notation $a(n) \approx_s b(n)$ means that $\left|\frac{a(n)}{b(n)} - 1\right| = \mathrm{snegl}(n)$.

Given a probability distribution $\mathcal{S}$ over the set S, denote $\sup(\mathcal{S})$ as the support set of $\mathcal{S}$. That is, $\sup(\mathcal{S})$ is the set of elements in S that have non-zero probability in $\mathcal{S}$.

3.1 Statistical Distance

▶ **Definition 2.** *Given two (discrete) probability distributions $\mathcal{S}_1$ and $\mathcal{S}_2$ over the set S, the statistical distance between them is defined as*

$$\Delta(\mathcal{S}_1, \mathcal{S}_2) = \frac{1}{2} \sum_{s \in S} \left| \Pr_{s' \leftarrow \mathcal{S}_1}[s' = s] - \Pr_{s' \leftarrow \mathcal{S}_2}[s' = s] \right|.$$

The statistical distance satisfies triangular inequality, i.e., for any distributions $\mathcal{S}_1, \mathcal{S}_2, \mathcal{S}_3$ over S, we have $\Delta(\mathcal{S}_1, \mathcal{S}_3) \leq \Delta(\mathcal{S}_1, \mathcal{S}_2) + \Delta(\mathcal{S}_2, \mathcal{S}_3)$. Another important fact is that statistical distance does not increase when applying a (possibly randomized) function f, i.e. $\Delta(\mathcal{S}_1, \mathcal{S}_2) \geq \Delta(f(\mathcal{S}_1), f(\mathcal{S}_2))$.

3.2 Private Constrained PRF

In this work, we use a simulation-based definition of 1-key oracle-aided private constrained PRF (PCPRF), our notion is similar to [12] except that both adversary and simulator have access to an oracle $\mathcal{O}$. Since the definition of 1-key security of PCPRF is straightforward to grasp, but a rigorous formulation would be laborious, we therefore omit it here and refer the interested readers to the full version.

4 PCPRF Relative to Oracle Γ

4.1 Oracle Design

Our oracle consists of the following parts:

- A PSPACE-complete oracle.
- $\mathsf{H} : Z = \{0,1\}^* \to \{0,1\}$ is a random oracle, i.e., for arbitrary binary string z, $\mathsf{H}(z)$ is sampled from uniform distribution over $\{0,1\}$.
- For every $\kappa \in \mathbb{N}^*$, we denote PCPRF oracle of length κ as Γ_κ defined below:
  - $K_\kappa = X_\kappa = C_\kappa = \{0,1\}^\kappa, K'_\kappa = \{0,1\}^{\kappa^2}, Y_\kappa = \{0,1\}^{\kappa^3}$. Here elements of C_κ are viewed as arbitrary $\mathsf{P/poly}$ predicates with input length κ. They are allowed to query H and the PSPACE-complete oracle. However, they are only allowed to query H for at most κ times.
  - $\Gamma_\kappa = (\mathsf{F}_\kappa, \mathsf{f}_\kappa, \mathsf{Keygen}_\kappa, \mathsf{R}_\kappa)$ are four oracles sampled by the following way:
    1. $\mathsf{F}_\kappa : K_\kappa \times X_\kappa \to Y_\kappa$ is uniformly sampled from all functions.
    2. $\mathsf{Keygen}_\kappa : K_\kappa \times C_\kappa \to K'_\kappa$ is sampled from all **injective** functions. Denote $k[c] = \mathsf{Keygen}(k, c)$.
    3. $\mathsf{f}_\kappa : K'_\kappa \times X_\kappa \to Y_\kappa$ is defined as follows: given $k' \in K'_\kappa, x \in X_\kappa$, if there exists $k \in K_\kappa, c \in C_\kappa$ s.t. $k' = k[c]$ and $c(x) = 1$, then $\mathsf{f}_\kappa(k', x) = \mathsf{F}_\kappa(k, x)$. The other locations of f_κ are sampled uniformly from Y_κ.
    4. $\mathsf{R}_\kappa : (K_\kappa \cup K'_\kappa) \times K'_\kappa \to C_\kappa \cup (K_\kappa \times C_\kappa \times C_\kappa) \cup \{\bot\}$ is defined as follows: if the input is (k, k') where $k \in K'_\kappa$ and $k' = k[c]$, then $\mathsf{R}_\kappa(k, k') = c$. If the input is (k'_1, k'_2) where $k'_1, k'_2 \in K'_\kappa$ s.t. there exists $k \in K_\kappa$, $c_1, c_2 \in C_\kappa, c_1 \neq c_2$, $k'_1 = k[c_1], k'_2 = k[c_2]$, then $\mathsf{R}_\kappa(k'_1, k'_2) = (k, c_1, c_2)$. In any other cases, R_κ will return $\bot$ as a sign of rejection.
- We define oracle Γ to be the concatenation of Γ_κ for every length, H and the PSPACE-complete oracle, i.e., $\Gamma = (\text{PSPACE-complete oracle}, \mathsf{H}, \{\Gamma_\kappa\}_{\kappa \in \mathbb{N}^*})$, also define K, X, C, K', Y to be the concatenation of all corresponding fields.

Here we remark that the elements of the sets we defined before are different elements, even though they might have the same binary representation. For example, suppose $z \in Z$ is of length κ, then z can only be used as input of H queries. We cannot use z as the input of a Γ_κ query. Similarly, an element $k \in K_\kappa$ can only be used as the first input of F_κ or R_κ queries; an element $x \in X_\kappa$ can only be used as the second input of F_κ or f_κ queries, etc.

In the following, we will often omit the index κ of the oracles in Γ_κ. For example, we might simply write $\mathsf{F}(k, x)$ instead of $\mathsf{F}_\kappa(k, x)$. This does not introduce ambiguity because κ is fully determined by the length of the input.

4.2 Construction of PCPRF

Now we propose our 1-key secure PCPRF construction:

- $\mathbf{Gen}(1^\kappa)$: randomly sample a master key $\mathsf{MSK} \xleftarrow{R} K_\kappa$.
- $\mathbf{Constrain}(\mathsf{MSK}, c)$: answer $\mathsf{Keygen}(\mathsf{MSK}, c)$.
- $\mathbf{Eval}(k, x)$: if k is a master key, i.e., $k \in K_\kappa$, return $\mathsf{F}(k, x)$; otherwise k is a constrain key, i.e., $k \in K'_\kappa$, return $\mathsf{f}(k, x)$.

The perfect correctness of our scheme follows directly the functionality of oracle Γ: whenever $c(x) = 1$, we have

$$\mathbf{Eval}(\mathsf{MSK}, x) = \mathsf{F}(\mathsf{MSK}, x) = \mathsf{f}(\mathsf{Keygen}(\mathsf{MSK}, c), x) = \mathbf{Eval}(\mathsf{CK}, x).$$

Since the oracle Γ os defined in the shape of PCPRFs, the proof of security naturally follows from the definition. Therefore we omit it here and refer the interested readers to the full version. Note that the recovery oracle R does not help in breaking the 1-key privacy because R takes two different constrained keys to generate an output that is not $\perp$.

5 Additional Preliminaries

5.1 Secret-Key Agreement Protocol

5.1.1 Definition

A secret-key agreement protocol consists of two (oracle-aided) P.P.T.M. called Alice and Bob. In addition, they have a common communication tape that both parties can read and write. A run of the protocol is as follows: Alice and Bob both start with input 1^λ, where λ is the security parameter; Alice and Bob run, communicating via the common tape; at last, Alice and Bob both output a string of length l, where $l = l(\lambda)$ is a polynomial over λ. If these two strings are the same, Alice and Bob are said to agree. The entire history of the writes to the communication tape is called the transcript. $\alpha(\lambda)$ will denote the probability that Alice and Bob agree on the same secret with security parameter λ.

We say (oracle-aided) P.P.T.M. Eve breaks a secret-key agreement protocol, if Eve, given only the transcript, can guess the secret with probability $1/\mathrm{poly}(\lambda)$ conditioned on Alice and Bob agree on the same secret. A protocol is secure if no P.P.T.M Eve can break it.

5.1.2 Normal Form

In this section, we will convert arbitrary KA protocol into a protocol satisfying certain conditions, which we call the "normal form". There are many steps in this conversion. In each step, the protocol gains a certain restriction that will facilitate our analysis. The conversion does not compromise the security of the KA protocol and only causes polynomial blow-up in running time.

Restriction 1: Avoid querying Γ_κ with small κ

Let λ be the security parameter of the key agreement protocol. The first step in our conversion is to eliminate any Γ_κ queries with $\kappa \leq \log \lambda$.

We first assume that all three parties have a preparation phase at the beginning of the protocol. In this phase, they make sufficient queries to H so that for every pair of (c, x) where $c \in C_\kappa, x \in X_\kappa, \kappa \leq \log \lambda$, the queries made in the preparation phase is enough to evaluate $c(x)$. For each $\kappa \leq \log \lambda$, $|C_\kappa| = |X_\kappa| = 2^\kappa \leq \lambda$. This means that there are only polynomially many such pairs. For each pair (c, x), to evaluate $c(x)$, at most $\log \lambda$ queries to H need to be made. Thus each party only need to make polynomially many queries to H in the preparation phase. Clearly, adding the preparation phase will not compromise the security of the KA protocol.

Then, we assume that each party also recovers the whole Keygen_κ oracle in the preparation phase. Since the input of Keygen_κ is in the set $|K_\kappa| \times |C_\kappa|$ which has size $2^{2\kappa} \leq \lambda^2$, this step only takes polynomial time. Once Keygen_κ is known to every party, there is no need to query

R_κ in Γ_κ, and all F_κ and f_κ queries are naturally divided into many equivalent classes by the correctness of PCPRF. The queries in each equivalent class have the same answer, and the answer for each equivalent class is individually and independently sampled from Y_κ. We can simulate the F_κ and f_κ oracle by a part of H that is never used in the original protocol. Thus, we may assume that F_κ and f_κ are never queried. As a result, only Keygen_κ is queried in Γ_κ.

Now, instead of letting each party recover Keygen_κ, we let Alice generate a "fake" Keygen_κ oracle and send it to Bob. Whenever a party needs to query Keygen_κ, they will instead use the fake Keygen_κ. As a result, each party will never access Γ_κ.

Restriction 2: Maintain the "completeness" with F, f, Keygen Queries

We first formally define what is the meaning of the "completeness" of a view:

▶ **Definition 3.** *We say a view is* ***complete*** *if for every κ and every pair of $c \in C_\kappa, x \in X_\kappa$ that appears in this view, the H queries made in this view are enough to evaluate $c(x)$.*

We are now ready to describe restriction 2 of the normal form. This restriction ensures that, if a party's view is complete before a $\mathsf{F}, \mathsf{f}, \mathsf{Keygen}$ query, then the view is also complete after this query. Specifically, suppose the next query of a party is $\mathsf{F}(k_0, x_0)$, $\mathsf{f}(k_0', x_0)$ or $\mathsf{Keygen}(k_0, c_0)$. Before making this query, this party will make sure that

- If the next query is $\mathsf{F}_\kappa(k_0, x_0)$ or $\mathsf{f}_\kappa(k_0', x_0)$, for every $c \in C_\kappa$ that has appeared in this party's view, enough H queries must be made in this view to evaluate $c(x_0)$.
- If the next query is $\mathsf{Keygen}_\kappa(k_0, c_0)$, for every $x \in X_\kappa$ that has appeared in this party's view, enough H queries must be made in this view to evaluate $c_0(x)$.

Restriction 3: Repair the completeness after R queries

Suppose a party just made an R_κ query and the answer is not $\perp$. Then, the answer must contain 1 or 2 predicates. Denote them as c_1 (and c_2). This party will first make sure that, for every $x \in X_\kappa$ that has appeared in this view, enough H queries are made to evaluate $c_1(x)$ (and $c_2(x)$).

Note that restriction 2 of the normal form ensures that $\mathsf{F}, \mathsf{f}, \mathsf{Keygen}$ queries cannot break the completeness of a party's view. Clearly, H queries cannot break the completeness of a party's view. Thus, only R queries can break the completeness of a party's view. Indeed, if an R query yields an answer other than $\perp$, then the answer might include 1 or 2 predicates. Denote them as c_1 (and c_2). If c_1 (and c_2) did not appear in this party's view, the completeness of a party's view might be temporarily broken. Restriction 3 ensures that the completeness of this party's view is fixed (after a number of H queries.)

Restriction 4: Find hidden constrained relations among the keys

Suppose the next query of a party is $\mathsf{F}(k_0, x_0)$, $\mathsf{f}(k_0', x_0)$ or $\mathsf{Keygen}(k_0, c_0)$, this party will first make sure that there are no hidden constrained relations among the master keys and constrained keys that have appeared in this view (along with k_0 or k_0'). Specifically, for each $k \in K_\kappa$ and $k' \in K_\kappa'$ that have appeared in this view (along with k_0 or k_0'), it must be clear in this view whether k' is a constrained key generated from k; for each $k_1', k_2' \in K_\kappa'$ appeared in this view (along with k_0'), it must be clear in this view whether they are the constrained keys generated from the same master key k. All of this can be achieved by the recovery oracle R.

The Normal Form

Let $\kappa_{\max}(\lambda)$ be the maximum κ such that Alice or Bob might query Γ_κ. Clearly, $\kappa_{\max}$ is also a polynomial over λ. We assume that the protocol has $n = n(\lambda)$ rounds where $n(\lambda)$ is a polynomial over λ. Alice speaks in odd rounds and Bob speaks in even rounds. Suppose r is even, then in round r:

- Bob first does his own computation. Then he makes an arbitrary query. This query is later referred as the first query of round r. This query can be an H query or Γ_κ query for some $\kappa \geq \log \lambda$. If this query is not an H query, then the input contains two elements. We make the restriction that at least one of the two input elements has already appeared in Bob's view. In other words, only one can be new between the two input elements. Denote the new input element as $q_{r,1}$. (If neither input elements are new or the first query is an H query, let $q_{r,1}$ be the first input element.)
- After the first query, Bob will make $2n\kappa_{\max}$ H queries. Let $q_{r,j}, (2 \leq j \leq 2n\kappa_{\max} + 1)$ be the input of the jth query. Especially, if the first query of round r is an R query and the answer is not $\perp$, then the following $2n\kappa_{\max}$ H queries must be first used to make sure that the completeness of the view still holds as requested by restriction 3. ($2n\kappa_{\max}$ H queries are enough to do this: R query introduces at most 2 new predicates; there are at most n elements in X in Bob's view because at most 1 new elements in X appears in Bob's view in each round; to evaluate $c(x)$ where both c, x appeared in this view, it takes at most $\kappa_{\max}$ H queries. Thus $2n\kappa_{\max}$ H queries are enough.)
- Bob then sends one message Conv^r to Alice. Denote $M^r = (\mathsf{Conv}^1, \mathsf{Conv}^2, \ldots, \mathsf{Conv}^r)$ be the messages sent so far.

For odd r, we only need to change the role of Alice and Bob.

Moreover, before Alice and Bob output their secrets s_A, s_B, we add 2 rounds at the last of the protocol. In these two rounds, the first query of Alice and Bob is $\mathsf{H}(s_A)$ and $\mathsf{H}(s_B)$. These two rounds ensure that, if Alice and Bob agree on the same secret s, both of them would have made the query $\mathsf{H}(s)$.

As we have discussed before, $\mathsf{H}, \mathsf{Keygen}, \mathsf{F}, \mathsf{f}$ queries will not break the completeness of a party's view. Only R queries might temporarily break the completeness. If the completeness of a party's view is broken by an R query, the following $2n\kappa_{\max}$ H queries make sure that the completeness is fixed. Therefore, each party's view is complete at the end of each round.

5.2 Views

Denote V to be the *view* of all three parties at the end of the secret-key agreement protocol. It consists of all three parties' randomness r_A, r_B, r_E, all messages M^n, as well as all the queries and corresponding answers made so far by all three parties to Γ. For now, the oracle behavior in a view can be arbitrary, as long as the same query yields the same answer. We will define what is a valid view in the next section.

Given a view V, if we only consider a part of it, we get a partial view. Let r be a round number, denote $V_A^{r,j}$ as the partial view of Alice in V up to and include the jth query of round r. This does not include the message sent by Alice in round r. Denote V_A^r as the partial view of Alice in V up to the end of round r which includes the message sent by Alice at round r. Same for $V_B^{r,j}$ and V_B^r. The notation is slightly different for Eve, $V_E^{r,i}$ denotes the partial view of Eve in V up to and include segment i of round r. Here, we only need to know that Eve's algorithm in round r includes many segments. In each segment, Eve might make multiple queries. Especially, $V_E^{r,0}$ denotes the view of Eve in V right before Eve's algorithm starts in round r.

Given a (possibly partial) view V, let $P(V)$ be the set of oracle query-answer pairs that appear in V. We also denote $Q(V)$ as the elements of $Z \cup X \cup C \cup K \cup K'$ that appears in this view. Note that we have excluded the elements in Y, because they can only appear as an output of a query.

Conversely, given several partial views, if their common parts are the same, we can combine them together to form another (possibly partial) view. We slightly abuse the notion $\cup$ to denote the combination of partial views. For example, let U, V are two views and M^r are the same in these two views, then we denote $U_A^r \cup V_B^r \cup V_E^{r,i}$ to be the view obtained by replacing the partial view of Alice in V^r to U_A^r.

In this work, we focus on a probability distribution over views:

▶ **Definition 4.** *Define $\mathcal{D}$ as the distribution of views generated by first randomly sampling all parties' randomness and oracle Γ defined in Section 4, then execute the protocol.*

6 Valid View and Indistinguishability Lemma $\mathcal{D} \approx \mathcal{D}'$

Recall that a "view" contains the randomness, messages, and query-answer pairs to Γ made by all parties. A view may include multiple master keys $k \in K$ and constrained keys $k' \in K'$ and many constrained relations $k[c] = k'$. A Keygen query $\mathsf{Keygen}(k, c) = k'$ is equivalent to one such relation: $k[c] = k'$. An R query $\mathsf{R}(k, k') = c$ is equivalent to one such relation $k[c] = k'$. An R query $\mathsf{R}(k'_1, k'_2) = (k, c_1, c_2)$ is equivalent to 2 such relations: $k[c_1] = k'_1, k[c_2] = k'_2$. Since the behavior of the oracles in a view can be arbitrary, $k_1[c_1]$ and $k_2[c_2]$ might be the same for $(k_1, c_1) \neq (k_2, c_2)$. We refer to the scenario where $k_1[c_1]$ and $k_2[c_2]$ are identical despite (k_1, c_1) being different from (k_2, c_2) as a **collision**.

In this section, we will study a special type of view called *valid view* that only allows "mild" collisions. Then we introduce another distribution $\mathcal{D}'$ defined over valid views and show that $\mathcal{D}$ and $\mathcal{D}'$ are statistically close which is the key lemma in proving our main theorem.

6.1 Valid View

In this section, we will define a special type of view called "valid view". Formally, we make the following definition:

▶ **Definition 5.** *A view V is a valid view if it satisfies the following conditions:*

1. *For two distinct master keys $k_1, k_2 \in K_\kappa$, if they collide at k' in V, i.e. $k_1[c_1] = k_2[c_2] = k'$, then there will be no collisions related to k_1, k_2.*
2. *There are no self-collisions. Here, a self-collision means that one master key generates the same constrained key for two different predicates, i.e. $k[c_1] = k[c_2]$ for $c_1 \neq c_2$.*

In a valid view, master keys and constrained keys can be divided into the following 3 types of groups:

1. **Isolated constrained key:** This group only contains 1 constrained key k' which does not appear to be any master key's constrained key in the view. Formally speaking, there exists no k, c such that $k' = k[c]$ in this view.
2. **Normal group:** this group contains one master key k and the constrained keys generated from k and k does not collide with any other master keys.
3. **Hybrid group:** this group can be viewed as two normal groups colliding at a single constrained key. Suppose k_0, k_1 are two different master keys, $\{c_0, c_1, \ldots, c_s\}$ and $\{c'_0, c'_1, \ldots, c'_t\}$ are two sets of predicates (the predicates in each set are different from each other). $k_0[c_0] = k_1[c'_0] = k'$ represents the colliding constrained key. The rest constrained keys $k_0[c_i], k_1[c'_j], i, j \geq 1$ are different from each other, and all different from k'.

To better understand the concept of valid view, we can think of the master keys and constrained keys in a view as nodes. A master key is represented by a filled node and a constrained key is represented by an unfilled node. A constrained relation $k[c] = k'$ is represented by an edge c connecting k and k'. The following example is a valid view that contains a collision.

▶ **Example 6.** Suppose a view V contains the following query-answer pairs: $\mathsf{R}(k'_1, k'_3) = \perp$, $\mathsf{f}(k'_2, x) = y_1$, $\mathsf{R}(k'_4, k'_5) = (k_1, c_2, c_3)$, $\mathsf{Keygen}(k_1, c_1) = k'_3$, $\mathsf{f}(k'_6, x) = y_2$, $\mathsf{R}(k'_6, k'_8) = (k_2, c_4, c_6)$, $\mathsf{R}(k_2, k'_7) = c_5$, $\mathsf{R}(k'_8, k'_9) = (k_3, c_7, c_8)$, $\mathsf{f}(k'_9, x) = y_3$. Then, figure 1 is the graph illustration of the relations between the keys in V. Note that, V contains a collision: $k_2[c_6] = k_3[c_7] = k'_8$, this means that $k_2, k_3, k'_6, k'_7, k'_8, k'_9$ form a hybrid group.

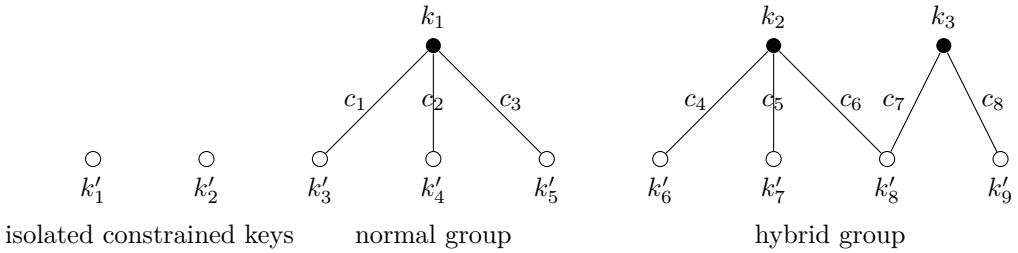


Figure 1 Example of a valid view V which contains 2 isolated constrained keys, 1 normal group and 1 hybrid group.

If a view is not valid, we can easily deduce that at least one of the following 4 complicated collisions must occur:

- Self-collision:

$$k[c_1] = k[c_2] = k'$$

- 3-to-1 collision: three master keys collide at one constrained key:

$$k_1[c_1] = k_2[c_2] = k_3[c_3] = k'$$

- 2-to-2 collision: two master keys collide at two different constrained keys:

$$k_1[c_1] = k_2[c_2] = k'_1, k_1[c'_1] = k_2[c'_2] = k'_2$$

- "W" collision: one master key simultaneously collides with two other master keys at two different constrained keys:

$$k_1[c_1] = k_2[c_2] = k'_1, k_1[c'_1] = k_3[c_3] = k'_2$$

We remark that valid views may contain collisions in the form of hybrid groups, this makes them different from the views in $\mathsf{sup}(\mathcal{D})$ which cannot contain any collisions because Keygen is sampled from injective functions. Thus, valid views can be seen as an extension over $\mathrm{sup}(\mathcal{D})$.

The readers may wonder why we allow a valid view to contain collisions. After all, collisions will never happen in a real execution because Keygen is sampled from all injective functions. The reason is that Eve's simulated view of Alice, combined with the view of Eve and Bob, might contain a collision. Looking ahead, we can see that the simulated view of Alice, combined with the view of Bob, forms a valid view if the simulation is a "miss".

6.2 Equivalent F or f Queries

Notice that in the definition of valid views, the correctness of PCPRF is not considered. Recall the correctness of PCPRF states that $\mathsf{F}(k,x)=\mathsf{f}(k[c],x)$ if $c(x)=1$. The reason that we have not yet considered the correctness of PCPRF is that, in order to evaluate $c(x)$, we might need to make extra H queries that are not in V. Thus, the correctness of PCPRF must be considered under a fixed instantiation H of the oracle H.

Suppose V is a valid view. H is an instantiation of the oracle H that is consistent with the H queries made in V. Suppose $k[c]=k'$ in V. The correctness of PCPRF under V,H states that $\mathsf{F}(k,x)=\mathsf{f}(k',x)$ if $c^H(x)=1$. We say that two F or f queries are equivalent under V and H if they can be deduced to have the same answer from the correctness of PCPRF under V,H.

Still take Example 6 as an illustration. Suppose H is an instantiation of the oracle H. Since no H queries are made in V, H is consistent with V. If $c_4^H(x)=c_6^H(x)=c_7^H(x)=c_8^H(x)=1$, we can say that $\mathsf{f}(k'_6,x)$ and $\mathsf{f}(k'_9,x)$ are equivalent, because the correctness of PCPRF states that $\mathsf{f}(k'_6,x)=\mathsf{F}(k_2,x)=\mathsf{f}(k'_8,x)=\mathsf{F}(k_3,x)=\mathsf{f}(k'_9,x)$. However, if at least one of $c_4^H(x),c_6^H(x),c_7^H(x),c_8^H(x)$ is 0, $\mathsf{f}(k'_6,x)$ and $\mathsf{f}(k'_9,x)$ are not equivalent under V,H. This is clearer in the graph illustration Figure 1. Note that the edges on the shortest path from k'_6 to k'_9 are exactly c_4,c_6,c_7,c_8. Thus, $\mathsf{f}(k'_6,x)$ and $\mathsf{f}(k'_9,x)$ are equivalent under V and H if and only if $c(x)=1$ for every c on this shortest path.

We remark that the answer of $\mathsf{f}(k'_6,x)$ and $\mathsf{f}(k'_9,x)$ in V, i.e. y_2,y_3 are irrelevant to whether $\mathsf{f}(k'_6,x)$ and $\mathsf{f}(k'_9,x)$ are equivalent under V,H.

6.3 Weights of Valid Views and the Indistinguishability Lemma

In this section, we define the notion of *weight* of valid views. With this notion, we may define a new distribution of views called $\mathcal{D}'$ which is distributed over all valid views, thus contains views with hybrid groups. Then, we will prove that $\Delta(\mathcal{D},\mathcal{D}')\le \mathrm{negl}(\lambda)$ (Lemma 8), which is essential to our black-box separation result.

First, even though the input of H can be any string, each party cannot make H queries with arbitrarily long input. Thus, we think of H as finite and denote the size of its domain as M. For a fixed instantiation H of the oracle H and a valid view V, we define $w_H(V)$ as follows:

1. If the H queries made in V are not consistent with H, then $w_H(V)=0$.
2. If two F or f queries made in V are equivalent under V,H, but their answers are different in V, then $w_H(V)=0$.
3. Otherwise, define $w_H(V)$ as follows: each constrained relation $k[c]=k'$ in V contributes a factor of $|K'_\kappa|^{-1}$. Here $k\in K_\kappa, c\in C_\kappa, k'\in K'_\kappa$. Each different F_κ or f_κ query in V contributes a factor of $|Y_\kappa|^{-1}$. However, equivalent F or f queries under V and H are counted only once. $w_H(V)$ is the product of all contributions.

Still take Example 6 as an illustration. Suppose the queries made in V are all in Γ_κ. Suppose H is an instantiation of the oracle H such that $c_4^H(x)=c_6^H(x)=c_7^H(x)=c_8^H(x)=1$. Now we want to compute $w_H(V)$:

1. Since no H queries are made in V, H is naturally consistent with V.
2. There are 3 F or f queries in V: $\mathsf{f}(k'_2,x),\mathsf{f}(k'_6,x),\mathsf{f}(k'_9,x)$. Since k'_2 is an isolated constrained key in V, $\mathsf{f}(k'_2,x)$ is nonequivalent to the other two queries. As we have shown in Section 6.2, $\mathsf{f}(k'_6,x),\mathsf{f}(k'_9,x)$ are equivalent under V, H. Thus, if $y_2\neq y_3$, $w_H(V)=0$. In the following, we assume $y_2=y_3$.

3. There are 8 constrained relations in V, each of which contributes a factor of $|K'_\kappa|^{-1}$. Next, there are 3 F or f queries in V, and two of them are equivalent. Thus, these 3 queries contribute a factor of $|Y_\kappa|^{-2}$ in total. Therefore, $w_H(V) = |K'_\kappa|^{-8}|Y_\kappa|^{-2} = 2^{-2\kappa^3-8\kappa^2}$.

Finally, the weight of a valid view V is defined as

$$w(V) = \frac{1}{2^M}\sum_H w_H(V)$$

Here, the summation is taken over all instantiations of H.

Then, the distribution $\mathcal{D}'$ is defined as follows:

▶ **Definition 7.** *$\mathcal{D}'$ is distributed over all valid views, the probability of a valid view V in $\mathcal{D}'$ is proportional to $w(V)$.*

We can, on some level, think of $w_H(V)$ as an approximated probability of the view V conditioned on H instantiated as H: for each pair of $k \in K_\kappa, c \in C_\kappa$, the constrained key $k[c]$ is sampled from K'_κ. Thus, for a specific k', the probability that $k[c] = k'$ is $|K'_\kappa|^{-1}$. Also, for each pair of $k \in K_\kappa, x \in X_\kappa$, the value of $\mathsf{F}(k, x)$ can be seen as sampled from Y_κ. Thus, for a specific y, the probability that $\mathsf{F}(k, x) = y$ is $|Y_\kappa|^{-1}$. Following this line of thought, we can think of $w(V)$ as an estimated probability of the view V, since $w(V)$ is the average value of $w_H(V)$ over all possible H. Therefore, it is plausible that $\mathcal{D}'$ and $\mathcal{D}$ should be close. Indeed, we can prove the following indistinguishability lemma:

▶ **Lemma 8.** *For any secret-key agreement protocol that each party never queries Γ_κ for $\kappa < \log_2 \lambda$ as we required in Section 5.1.2, it holds that $\Delta(\mathcal{D}, \mathcal{D}') \le negl(\lambda)$.*

The proof of Lemma 8 can be found in appendix B of the full version.

7 Breaking Secret-Key Agreement Protocol Relative to Oracle Γ

In this section, we will construct an adversary Eve who can break any secret-key agreement protocol in the normal form stated in the previous section.

7.1 Eve's Goal

In the normal form of KA protocol (see Section 5.1.2) Alice and Bob queries $\mathsf{H}(s_A)$ and $\mathsf{H}(s_B)$ respectively in the last round, where s_A and s_B are Alice and Bob's output. With probability $\alpha(\lambda)$, $s_A = s_B = s$, then both Alice and Bob would have made the same query $\mathsf{H}(s)$ at the end of the protocol. This means, $s \in Q(V_A) \cap Q(V_B)$, here V_A and V_B denote the view of Alice and Bob at the end of the protocol. Thus, Eve's mission can be reduced to find all the intersection elements $Q(V_A) \cap Q(V_B)$. If Eve accomplishes to include all the intersection elements with probability $1 - \alpha/2$, then she finds all the intersection elements with probability at least $1/2$ conditioned on $s_A = s_B$. Thus, because Eve's view only contains polynomially many elements, Eve can simply choose one element randomly from her view as her guess. The guess would be s with at least $1/\mathsf{poly}(\lambda)$ probability. This means the protocol is broken. In the following, Eve's goal is to find all the intersection elements at the end of the protocol.

7.2 Eve's Algorithm

Suppose r is even, then Bob speaks in round r and Alice speaks in round $r-1$. Eve's algorithm in round r can be seen as running simultaneously with Bob's algorithm in round r. Eve's algorithm in round r consists of many repeated segments. Before segment i, Eve's partial view is $V_E^{r,i-1}$ which is complete. Then, segment i is divided into the following two phases:

- **Simulation Phase**: sample a view of Alice up to the end of round $r-1$: $\mathrm{Sim}_A \leftarrow \mathsf{Sim}(M^r, V_E^{r,i-1})$. Here, the distribution $\mathsf{Sim}(M^r, V_E^{r,i-1})$ is defined as follows: $\mathsf{SIM}(M^{r-1}, V_E^{r,i-1})$ is distributed over all possible partial views of Alice up to the end of round $r-1$ (denoted as Sim_A) that are consistent with M^{r-1}. Besides, $\mathrm{Sim}_A \cup V_E^{r,i-1}$ must form a collision-free valid view. The probability of Sim_A in $\mathsf{SIM}(M^r, V_E^{r,i-1})$ is proportional to $w(\mathrm{Sim}_A \cup V_E^{r,i-1})$.
- **Update Phase**: Eve includes all the elements in $Q(\mathrm{Sim}_A)$ in her view. Here, Sim_A is the sampled view of Alice in the last simulation phase. Then, Eve will try to acquire informations related to $Q(V_E)$ as much as possible. Specifically, for each pair of $k \in K, c \in C$ in her view, she will ask $\mathsf{Keygen}(k, c)$, the answer is automatically included in her view; for each pair of $x \in X, k \in K$ (resp. $k' \in K'$), she will ask $\mathsf{F}(k, x)$ (resp. $\mathsf{f}(k', x)$); Eve always make best efforts to recover the relationships of the master-keys and constrained keys in her view, this can be done by the recovery oracle R. Additionally, Eve will always maintain the completeness of her view. After the update, Eve's view is $V_E^{r,i}$. Naturally, $V_E^{r,i}$ is also complete.

Then, round r proceeds as follows (including Bob and Eve's algorithm):

1. Eve runs m segments described above. (Here $m = m(\lambda)$ is polynomial over λ. Its value is determined later.)
2. Bob makes the first query in round r.
3. We assume that there is a fourth party Charlie. She has the view of all three parties. Charlie is designated for one purpose: if Bob's first query in the last step is an R query and the answer is not $\perp$, then the answer must contain 1 or 2 predicates, say c_1 (and c_2). If c_1 has never appeared in Bob and Eve's view before, Charlie will send c_1 to Eve. Same for c_2. Otherwise Charlie simply sends $\perp$ to Eve.
4. If Charlie's message is not $\perp$ in the last step, Eve will include the received predicates in her view and make her view full, same as the update phase of a segment.
5. Eve runs m segments.
6. Bob makes the rest H queries in round r.
7. Bob sends the message M^r to Alice.

Here, Bob's queries are divided into 2 parts: the first query and the rest $2n\kappa_{\max}$ H queries. Before each part, Eve runs m segments. Between these two parts, we introduces a fourth party Charlie that might send one or two predicates to Eve. If Eve does not receive $\perp$, she will run another update phase to include the received predicates into her view. Indeed, Eve might receive private informations from Charlie, which renders Eve's attack invalid. Therefore, we make the following rule: as long as Charlie sends a message other than $\perp$, Eve is considered failed. In other words, we say Eve fails if (1), Eve does not include all the intersection elements in her view; (2), Charlie sends a message other than $\perp$. We will show that the probability that Eve fails is small. This means that the probability that Eve fails is also small even without Charlie.

7.3 Distribution $\mathcal{D}^{r,i,j}$ and $\mathcal{D}'^{r,i,j}$

The definition of $\mathcal{D}^{r,i,j}, \mathcal{D}'^{r,i,j}$ is the same as $\mathcal{D}, \mathcal{D}'$, except that we make the algorithms of Alice and Bob stop after the j th query of round r and Eve's algorithm stops after segment i of round r. (Especially, if $i = m$, the view sampled from $\mathcal{D}^{r,m,j}$ or $\mathcal{D}'^{r,m,j}$ should also contain the part that Charlie sends messages to Eve and Eve makes updates.) Proof of Lemma 8 also shows $\Delta(\mathcal{D}^{r,i,j}, \mathcal{D}'^{r,i,j}) \leq \text{negl}(\lambda)$.

Recall that the sampling process of $\mathcal{D}$ is as follows: first sample the randomness of three parties r_A, r_B, r_E (Charlie is deterministic) and the whole oracle Γ, then run the protocol. The only difference between $\mathcal{D}$ and $\mathcal{D}^{r,i,j}$ is that the protocol of $\mathcal{D}^{r,i,j}$ stops earlier than $\mathcal{D}$. Thus, we can sample $V^{r,i,j} \leftarrow \mathcal{D}^{r,i,j}$ in the following way: sample $V \leftarrow \mathcal{D}$ first, then output the partial view $V^{r,i,j}$. [2]

7.4 Main Result

Under certain parameter regime, we prove that Eve can break the secret-key agreement protocol with high probability. Let $S(\lambda) = 1/\text{poly}(\lambda)$, we have

▶ **Theorem 9.** *At the end of the protocol, Eve finds all the intersection elements of Alice and Bob's view while Charlie always sends $\perp$ with high probability. Formally, for some sufficiently large polynomial m depending on $S(\lambda)$ (determined later),*

$$\Pr_{V \leftarrow \mathcal{D}}[Q(V_A^n) \cap Q(V_B^n) \subseteq Q(V_E^{n,2m}) \wedge \textit{Charlie always sends } \perp] > 1 - S(\lambda)$$

for sufficiently large λ.

Thus, as long as $S(\lambda) < \alpha(\lambda)/2$, Eve can break the KA protocol as discussed in Section 7.1.

7.5 Proof of Theorem 9

In the rest of this section, we will give a proof sketch for our main result Theorem 9.

Recall that Eve's goal is to include all the intersection elements in her view, while Charlie always sends $\perp$. We instead prove a stronger result: before an element q becomes an intersection element, Eve will anticipate q with high probability. We say Eve fails if

1. An element q becomes an intersection element as an input to a query, but Eve's view does not include it.
2. An element q becomes an intersection element as an output of a query, but Eve's view does not include it.
3. Charlie sends a message other than $\perp$.

If Eve never fails through the whole execution of the protocol, all intersection elements must be included in Eve's view and Charlie always sends $\perp$. This means that Eve accomplishes her goal. If Eve fails, there must be a first failure. We first show the probability that Eve's first failure is type 2 or 3 is negligible.

▶ **Lemma 10.** *Suppose V is sampled from $\mathcal{D}$. Then the probability that Eve fails in V and the first failure is type 2 or 3 is negligible.*

[2] However, the same does not necessarily hold for $\mathcal{D}'$ and $\mathcal{D}'^{r,i,j}$ because $\mathcal{D}'$ and $\mathcal{D}'^{r,i,j}$ are defined in a completely different way than $\mathcal{D}$ and $\mathcal{D}'^{r,i,j}$.

Here, we only give a proof sketch of this lemma. We refer the interested readers to the full version. Note that the size of K'_κ is 2^{κ^2}. In K'_κ, there are only $|K_\kappa| \cdot |C_\kappa| = 2^{2\kappa}$ constrained keys. This proportion is negligible as long as $\kappa \geq \log \lambda$. Therefore, it is almost impossible to find a constrained key in $|K'_\kappa|$ without using Keygen oracle. We denote this event as spoof. The key observation is that, as long as Eve fails and the first failure is type 2 or 3, then the event spoof occurs.

With this lemma, we only need to concern type 1 failures. Recall that Alice and Bob can make at most $1+2n\kappa_{\max}$ queries in each round, and there are n rounds in total, Alice and Bob can make at most $n(1+2n\kappa_{\max})$ queries in total. Denote $p(\lambda) = S(\lambda)/n(\lambda)(1+2n(\lambda)\kappa_{\max}(\lambda))$. If we can bound the probability that the input of the jth query of round r is Eve's first failure by p, we can bound the probability that Eve fails. Here, we consider the first query and the rest H queries separately. Formally, we prove the following 2 lemmas:

▶ **Lemma 11.** *For some sufficiently large polynomial $m = m(\lambda)$ depending on $p(\lambda)$, for each round number $r \in [1, n]$ (without loss of generality assume Bob speaks in round r), the probability that either*

1. *$Q(V_A^{r-1}) \cap Q(V_B^{r-1}) \not\subseteq Q(V_E^{r,0})$. (At the end of round $r-1$, Eve failed to include all intersection elements. This is stronger than Eve failing before round r).*
2. *$q_{r,1} \notin Q(V_A^{r-1})$. (For $q_{r,1}$ to be an intersection element, it must have appeared in Alice's view before round r.)*
3. *$q_{r,1} \in Q_E^{r,m}$. (The definition of Eve finding $q_{r,1}$ at the end of the first m segments.)*

is greater than $1 - p(\lambda)/2$ for sufficiently large λ.

▶ **Lemma 12.** *For some sufficiently large polynomial $m = m(\lambda)$ depending on $p(\lambda)$, for each round number $r \in [1, n]$ (without loss of generality assume Bob speaks in round r) and query number $j \in [2, 1 + 2n\kappa_{\max}]$, the probability that either*

1. *$Q(V_A^{r-1}) \cap Q(V_B^{r,j-1}) \not\subseteq Q(V_E^{r,m})$. (Right before the jth query of round r, Eve failed to include all intersection elements with the first m segments. This is stronger than Eve failing before the jth query of round r).*
2. *$q_{r,j} \notin Q(V_A^{r-1})$. (For $q_{r,j}$ to be an intersection element, it must have appeared in Alice's view before round r.)*
3. *$q_{r,j} \in Q(V_E^{r,2m})$. (The definition of Eve finding $q_{r,j}$ at the end of round r.)*

is greater than $1 - p(\lambda)/2$ for sufficiently large λ.

Note that, if Condition 1 $\vee$ Condition 2 $\vee$ Condition 3 has probability greater than $1-p/2$, then $\neg$Condition 1 $\wedge$ $\neg$Condition 2 $\wedge$ $\neg$Condition 3 has probability smaller than $p/2$, and this is weaker than Eve first failing at this query.

The detailed proofs of Lemma 11 and Lemma 12 can be found in appendix C of the full version.

Now Theorem 9 can be proved by Lemma 11 and Lemma 12:

Proof. Combining Lemma 11 and Lemma 12, for each round number r and query number j, the probability that Eve's first failure is the input of the jth query of round r and the failure is type 1 is bounded by $p/2$. Apply a union bound, the probability that Eve fails and the first failure is type 1 is bounded by $(p/2) \cdot n(1 + 2n\kappa_{\max}) = S/2$. Combined with Lemma 10, the probability that Eve fails is bounded by $S/2 + \mathrm{negl}(\lambda) < S$ for sufficiently large λ. This is stronger than Theorem 9. ◀

References

1 Gilad Asharov and Gil Segev. Limits on the power of indistinguishability obfuscation and functional encryption. *SIAM Journal on Computing*, 45(6):2117–2176, 2016. doi:10.1137/15M1034064.

2 Nuttapong Attrapadung, Takahiro Matsuda, Ryo Nishimaki, Shota Yamada, and Takashi Yamakawa. Constrained prfs for NC^1 in traditional groups. In Hovav Shacham and Alexandra Boldyreva, editors, *Advances in Cryptology – CRYPTO 2018*, pages 543–574, Cham, 2018. Springer International Publishing.

3 Nuttapong Attrapadung, Takahiro Matsuda, Ryo Nishimaki, Shota Yamada, and Takashi Yamakawa. Adaptively single-key secure constrained prfs for NC^1. In *Public Key Cryptography (2)*, volume 11443 of *Lecture Notes in Computer Science*, pages 223–253. Springer, 2019.

4 Boaz Barak and Mohammad Mahmoody-Ghidary. Merkle puzzles are optimal — an $O(n^2)$-query attack on any key exchange from a random oracle. In Shai Halevi, editor, *Advances in Cryptology - CRYPTO 2009*, pages 374–390, Berlin, Heidelberg, 2009. Springer Berlin Heidelberg. doi:10.1007/978-3-642-03356-8_22.

5 Dan Boneh, Kevin Lewi, and David J. Wu. Constraining pseudorandom functions privately. In *Public Key Cryptography (2)*, volume 10175 of *Lecture Notes in Computer Science*, pages 494–524. Springer, 2017. doi:10.1007/978-3-662-54388-7_17.

6 Dan Boneh and Brent Waters. Constrained pseudorandom functions and their applications. In *ASIACRYPT (2)*, volume 8270 of *Lecture Notes in Computer Science*, pages 280–300. Springer, 2013. doi:10.1007/978-3-642-42045-0_15.

7 Dan Boneh and Mark Zhandry. Multiparty key exchange, efficient traitor tracing, and more from indistinguishability obfuscation. In Juan A. Garay and Rosario Gennaro, editors, *Advances in Cryptology – CRYPTO 2014*, pages 480–499, Berlin, Heidelberg, 2014. Springer Berlin Heidelberg. doi:10.1007/978-3-662-44371-2_27.

8 Elette Boyle, Shafi Goldwasser, and Ioana Ivan. Functional signatures and pseudorandom functions. In Hugo Krawczyk, editor, *Public-Key Cryptography – PKC 2014*, pages 501–519, Berlin, Heidelberg, 2014. Springer Berlin Heidelberg. doi:10.1007/978-3-642-54631-0_29.

9 Zvika Brakerski, Jonathan Katz, Gil Segev, and Arkady Yerukhimovich. Limits on the power of zero-knowledge proofs in cryptographic constructions. In Yuval Ishai, editor, *Theory of Cryptography*, pages 559–578, Berlin, Heidelberg, 2011. Springer Berlin Heidelberg. doi:10.1007/978-3-642-19571-6_34.

10 Zvika Brakerski, Rotem Tsabary, Vinod Vaikuntanathan, and Hoeteck Wee. Private constrained prfs (and more) from lwe. In Yael Kalai and Leonid Reyzin, editors, *Theory of Cryptography*, pages 264–302, Cham, 2017. Springer International Publishing. doi:10.1007/978-3-319-70500-2_10.

11 Zvika Brakerski and Vinod Vaikuntanathan. Constrained key-homomorphic prfs from standard lattice assumptions - or: How to secretly embed a circuit in your PRF. In *TCC (2)*, volume 9015 of *Lecture Notes in Computer Science*, pages 1–30. Springer, 2015. doi:10.1007/978-3-662-46497-7_1.

12 Ran Canetti and Yilei Chen. Constraint-Hiding Constrained PRFs for NC^1 from LWE. In Jean-Sébastien Coron and Jesper Buus Nielsen, editors, *Advances in Cryptology – EUROCRYPT 2017*, volume 10210, pages 446–476. Springer International Publishing, Cham, 2017. Series Title: Lecture Notes in Computer Science. doi:10.1007/978-3-319-56620-7_16.

13 Alex Davidson, Shuichi Katsumata, Ryo Nishimaki, Shota Yamada, and Takashi Yamakawa. Adaptively secure constrained pseudorandom functions in the standard model. In Daniele Micciancio and Thomas Ristenpart, editors, *Advances in Cryptology – CRYPTO 2020*, pages 559–589, Cham, 2020. Springer International Publishing. doi:10.1007/978-3-030-56784-2_19.

14 Sanjam Garg, Mohammad Hajiabadi, Mohammad Mahmoody, and Ameer Mohammed. Limits on the power of garbling techniques for public-key encryption. Cryptology ePrint Archive, Paper 2018/555, 2018. URL: https://eprint.iacr.org/2018/555.

15 R. Gennaro and L. Trevisan. Lower bounds on the efficiency of generic cryptographic constructions. In *Proceedings 41st Annual Symposium on Foundations of Computer Science*, pages 305–313, 2000. doi:10.1109/SFCS.2000.892119.

16 Y. Gertner, S. Kannan, T. Malkin, O. Reingold, and M. Viswanathan. The relationship between public key encryption and oblivious transfer. In *Proceedings 41st Annual Symposium on Foundations of Computer Science*, pages 325–335, 2000. doi:10.1109/SFCS.2000.892121.

17 Yael Gertner, Tal Malkin, and Omer Reingold. On the impossibility of basing trapdoor functions on trapdoor predicates. In *Proceedings 42nd IEEE Symposium on Foundations of Computer Science*, pages 126–135. IEEE, 2001. doi:10.1109/SFCS.2001.959887.

18 Oded Goldreich, Shafi Goldwasser, and Silvio Micali. How to construct random functions. *J. ACM*, 33(4):792–807, 1986. doi:10.1145/6490.6503.

19 Dennis Hofheinz, Akshay Kamath, Venkata Koppula, and Brent Waters. Adaptively secure constrained pseudorandom functions. In Ian Goldberg and Tyler Moore, editors, *Financial Cryptography and Data Security*, pages 357–376, Cham, 2019. Springer International Publishing. doi:10.1007/978-3-030-32101-7_22.

20 R. Impagliazzo and S. Rudich. Limits on the provable consequences of one-way permutations. In *Proceedings of the Twenty-First Annual ACM Symposium on Theory of Computing*, STOC '89, pages 44–61, New York, NY, USA, 1989. Association for Computing Machinery. doi:10.1145/73007.73012.

21 Aggelos Kiayias, Stavros Papadopoulos, Nikos Triandopoulos, and Thomas Zacharias. Delegatable pseudorandom functions and applications. In *Proceedings of the 2013 ACM SIGSAC Conference on Computer & Communications Security*, CCS '13, pages 669–684, New York, NY, USA, 2013. Association for Computing Machinery. doi:10.1145/2508859.2516668.

22 Arthur Lazzaretti and Charalampos Papamanthou. Near-optimal private information retrieval with preprocessing. In Guy Rothblum and Hoeteck Wee, editors, *Theory of Cryptography*, pages 406–435, Cham, 2023. Springer Nature Switzerland. doi:10.1007/978-3-031-48618-0_14.

23 Mohammad Mahmoody, Hemanta K. Maji, and Manoj Prabhakaran. *On the Power of Public-Key Encryption in Secure Computation*, pages 240–264. Springer Berlin Heidelberg, 2014. doi:10.1007/978-3-642-54242-8_11.

24 Takahiro Matsuda and Kanta Matsuura. On black-box separations among injective one-way functions. In *Theory of Cryptography Conference*, pages 597–614. Springer, 2011. doi:10.1007/978-3-642-19571-6_36.

25 Steven Rudich. The use of interaction in public cryptosystems (extended abstract). In *Advances in Cryptology - CRYPTO '91, 11th Annual International Cryptology Conference, Santa Barbara, California, USA, August 11-15, 1991, Proceedings*, volume 576 of *Lecture Notes in Computer Science*, pages 242–251. Springer, 1991. doi:10.1007/3-540-46766-1_19.

26 Amit Sahai and Brent Waters. How to use indistinguishability obfuscation: deniable encryption, and more. In *Proceedings of the Forty-Sixth Annual ACM Symposium on Theory of Computing*, STOC '14, pages 475–484, New York, NY, USA, 2014. Association for Computing Machinery. doi:10.1145/2591796.2591825.

27 Elaine Shi, Waqar Aqeel, Balakrishnan Chandrasekaran, and Bruce Maggs. Puncturable pseudorandom sets and private information retrieval with near-optimal online bandwidth and time. In Tal Malkin and Chris Peikert, editors, *Advances in Cryptology – CRYPTO 2021*, pages 641–669, Cham, 2021. Springer International Publishing. doi:10.1007/978-3-030-84259-8_22.

28 Daniel R. Simon. Finding collisions on a one-way street: Can secure hash functions be based on general assumptions? In Kaisa Nyberg, editor, *Advances in Cryptology — EUROCRYPT'98*, pages 334–345, Berlin, Heidelberg, 1998. Springer Berlin Heidelberg. doi:10.1007/BFB0054137.

Interactive Proofs for Batch Polynomial Evaluation

Gal Arnon
Bocconi University, Milan, Italy

Alessandro Chiesa
EPFL, Lausanne, Switzerland

Giacomo Fenzi
EPFL, Lausanne, Switzerland

Eylon Yogev
Bar-Ilan University, Ramat Gan, Israel

Abstract

Polynomials are a fundamental mathematical object underlying virtually all of theoretical computer science. In proof systems, a common task for the verifier is to evaluate a polynomial of degree d at m distinct points. The best known algorithm for this problem performs $O((m+d)\cdot\log^2(m+d))$ field operations.

We present a concretely efficient MA protocol for this problem in which the verifier runs in *linear time*: the prover sends a single message consisting of $d-1$ field elements, and the verifier performs only $O(m+d)$ field operations. We further extend our protocol to handle the more general setting of evaluating multiple polynomials at multiple points, and for this problem, we construct an AMA protocol.

Our protocols improve the verifier time in several interactive proofs. Most notable are the sumcheck protocol over a large summation domain and protocols that rely on polynomial quotienting. In particular, by a straightforward application of our results, we reduce the verifier's runtime in the STIR protocol (CRYPTO 2024) to match that of WHIR (EUROCRYPT 2025), despite WHIR being highly optimized for verification time.

As an additional application, we show that any univariate polynomial commitment scheme (PCS) can be transformed, in a black-box manner, into a new scheme that efficiently supports batch openings at multiple points. In particular, opening m points incurs only a constant overhead compared to opening a single point.

2012 ACM Subject Classification Theory of computation → Interactive proof systems; Theory of computation → Cryptographic protocols

Keywords and phrases interactive proofs, polynomial evaluation

Digital Object Identifier 10.4230/LIPIcs.ITC.2026.3

Related Version *Full Version*: https://eprint.iacr.org/2026/448

Funding Alessandro Chiesa and Giacomo Fenzi are supported in part by the Ethereum Foundation.
Gal Arnon: Gal Arnon is supported by the European Research Council (ERC) under the EU's Horizon 2020 research and innovation programme (Grant agreement No. 101019547), and Stellar Foundation Grant.
Eylon Yogev: Eylon Yogev is supported by the Israel Science Foundation (Grant No. 2302/22), European Research Union (ERC, CRYPTOPROOF, 101164375).

Acknowledgements This work was done in part while Gal Arnon and Giacomo Fenzi were visiting the Simons Institute for the Theory of Computing, and Giacomo Fenzi was visiting Succinct.

1 Introduction

Polynomials play a central role in the design of cryptographic protocols, where they are used as the algebraic backbone of numerous primitives and constructions. These include important notions such as error-correcting codes, secret sharing schemes, commitment

schemes, probabilistically checkable proofs, interactive proofs, and succinct non-interactive arguments. In particular, the algebraic structure of polynomials is at the heart of the efficiency and soundness of many modern proof systems where polynomials are used to encode computation, and their algebraic structure enables the construction of crucial tools such as consistency checks via low-degree testing, interpolation, and evaluation arguments.

As a result, the task of verifying polynomial evaluations – i.e., checking that one or more claimed values correspond to the evaluation of a specific low-degree polynomial at a given set of points – is ubiquitous across interactive and non-interactive proof systems. This task is especially prominent in polynomial commitment schemes and related primitives, where a prover commits to a polynomial and later proves evaluations at points of the verifier's choosing.

Batch polynomial evaluation. In this work, we study the problem $\mathsf{PolyEval}_{\mathbb{F}}$ of batch verification of polynomial evaluation claims over a field $\mathbb{F}$. Given a list of m pairs (x_i, y_i), the verifier must efficiently check that $p(x_i) = y_i$ for all i, where p is a degree d polynomial provided to the verifier in some form (we later give a generalization that considers also the evaluation of multiple polynomials). We usually think of m as being much larger than d. The $\mathsf{PolyEval}_{\mathbb{F}}$ problem arises in various cryptographic protocols, particularly where an untrusted prover is involved. Notable applications include SNARK proof aggregation, the construction of lookup arguments, and protocols leveraging the sumcheck or multilinear extension paradigms (e.g., [20, 27, 18, 16, 17, 25, 2]).

A straightforward approach uses Horner's method [8], which evaluates a degree d polynomial at a single point using d field multiplications and d additions. Applying Horner's method independently to each of the m evaluation points results in a total cost of $O(m \cdot d)$ field operations. While Horner's method is optimal for evaluating a single point, more efficient algorithms exist for the multipoint setting.

A classical result in algebraic complexity (see [29, Chapter 10] and [15]) is an algorithm that performs $O\left((d+m) \cdot \log^2(d+m)\right)$ field operations. However, upon investigation, the hidden constant in the big-O notation is large in practice, limiting the practicality of this approach for moderate input sizes. Alternatively, the Barycentric interpolation method gives a linear time algorithm of $O(d+m)$, but with a quadratic $O(m^2)$ preprocessing phase (that depends on the points and not the polynomials) [28]. This method is employed across various components of the Ethereum network [9].

When the evaluation points lie in a structured domain (e.g., FFT-friendly sets), even faster algorithms are possible. However, this paper focuses on the general case of arbitrary evaluation points – a necessity in many applications, for example, when the evaluation points are chosen uniformly at random. As the $\mathsf{PolyEval}_{\mathbb{F}}$ verification problem frequently arises in the context of interactive proofs, we ask:

Can $\mathsf{PolyEval}_{\mathbb{F}}$ *be decided more efficiently with the help of an untrusted prover?*

In this work, we answer this question in the affirmative.

1.1 Our results

We give an efficient single-message protocol for $\mathsf{PolyEval}_{\mathbb{F}}$ where the verifier runs in strictly linear time: it evaluates the polynomial p at a *single* location and, additionally, makes a small number of field operations.

▶ **Theorem 1** (informal, Theorem 9). *There is an MA proof for* $\mathsf{PolyEval}_{\mathbb{F}}$ *where:*

- *the soundness error is* $\frac{m+d}{|\mathbb{F}|-m}$,
- *the prover sends a single degree* $d-1$ *polynomial and makes* $O(d \cdot m)$ *field operations; and*
- *the verifier makes* $O(d+m)$ *field operations. More precisely, the verifier makes* $2d+3m$ *multiplications and additions and a single inversion.*

The above theorem immediately extends to speeding up polynomial *interpolation*, where the goal is to compute the coefficients of the unique low-degree univariate polynomial that passes through a given set of points. In this setting, the prover can send the interpolated polynomial in coefficient form and then prove to the verifier that it evaluates correctly at all the specified points.

Multiple polynomials and multiple points. We also consider the problem of batch polynomial evaluation with multiple polynomials as well as multiple evaluation points.

In the problem $\mathsf{MultPolyEval}_{\mathbb{F}}$, given degree d polynomials $p_1, \ldots, p_\ell$ and a list of m tuples of the form $(x_i, y_{i,1}, \ldots, y_{i,\ell})$,[1] we check that for all i and j it holds that $p_i(x_j) = y_{i,j}$. Applying Theorem 1 for each of the ℓ polynomials would incur an ℓ multiplicative overhead to the communication. Our next theorem shows that these proofs can be batched together at the cost of a single verifier message:

▶ **Theorem 2** (informal, Theorem 10). *There is an AMA protocol for* $\mathsf{MultPolyEval}_{\mathbb{F}}$ *where:*

- *the round-by-round soundness*[2] *error is* $\frac{\ell+m+d}{|\mathbb{F}|-m}$,
- *the prover sends a single degree* $d-1$ *polynomial and makes* $O(d \cdot m + d \cdot \ell)$ *field operations; and*
- *the verifier makes* $O(m \cdot \ell + d \cdot \ell)$ *field operations. More precisely, the verifier makes* $d\ell + 5m\ell$ *multiplications,* $d\ell + 4m\ell$ *additions, and a single inversion.*

Large fields. The soundness errors of Theorem 1 and Theorem 2 require the size of the field $|\mathbb{F}|$ to be large relative to the size of the input polynomials. Similarly to prior work, the requirement can be relaxed by sampling challenges in the protocols from a sufficiently large extension field of $\mathbb{F}$.

Immediate applications. Our protocols allow for optimizing the verifier in many protocols where the verifier needs to evaluate polynomials. We describe two main such applications.

- *The sumcheck protocol.* The sumcheck protocol [20] is an influential protocol used both in theory and practice for proving claims of the form

$$\sum_{x_1,\ldots,x_v \in H} p(x_1, \ldots, x_v) = \gamma \ ,$$

  for a v-variate polynomial p of individual degree d. During the protocol, the verifier must check v claims of the form $\sum_{x \in H} q_i(x) = \gamma_i$ where q_i is a degree d univariate polynomial. This step is typically implemented naively in time $\Omega(v \cdot d \cdot |H|)$.
  Using Theorem 2, the running time is reduced to $O(v \cdot d + v \cdot |H|)$ at the cost of an additional round of communication with $v \cdot |H| + d - 1$ field elements.

[1] Actually, we can generalize this further so that not all polynomials must take part in each evaluation point and the polynomials are not required to have the same degree.

[2] Round-by-round soundness is a stronger variant of soundness, which is used when making protocols non-interactive using the Fiat–Shamir heuristic.

- *Quotienting.* Polynomial quotienting is a technique used in practice by many protocols, such as STIR [2] and ARC [7]. At a high level, in these protocols, the verifier must evaluate at many points a function

$$f(X) = \frac{g(X) - \mathsf{Ans}(X)}{V(X)} \ ,$$

  where g is an oracle function, and Ans and V are degree d polynomials. This computation is often an efficiency bottleneck in the running time of the verifier. Indeed, as a result, the recent WHIR protocol [3] was developed, in part, to improve the verification time of STIR by avoiding quotient computations on the verifier's side.
  Our protocol can be used to directly improve the computation of the quotient, and thus the verifier complexity. We adapt STIR using this technique, resulting in the STIR-SHAKE protocol, which reduces the verifier's running time to asymptotically match that of WHIR. See Section 6 for a more in-depth discussion and analysis.

Immediate generalizations. The techniques in our protocol directly yield a MA protocol to check whether, for a given list of polynomials $u_1, \ldots, u_m$ and $w_1, \ldots, w_m$ (such that the zeros of w_i are pairwise disjoint) the zeros of w_i are contained within the zeros of u_i *up to multiplicities*. The verifier in the generalized protocol performs a single evaluation of each of the u_i, w_i polynomials, an evaluation of a single additional polynomial and $O(m)$ additional field operations. We refer the reader to Section 3 for details.

1.2 Application: batching polynomial commitment schemes

The protocols underlying Theorem 1 and Theorem 2 have a useful structure: they can be interpreted as *polynomial IOPs* (PIOP). In a PIOP, the prover's messages are bounded-degree polynomials, and the verifier queries these polynomials via evaluation queries. Polynomial IOPs [24, 11, 6] are a fundamental building block of most modern SNARGs. By utilizing this fact about our IPs, we show a generic compiler for reducing the query complexity of any PIOP:

▶ **Theorem 3** (informal, Theorem 15). *Let* Π *be a* k*-round PIOP for a relation* $\mathcal{R}$*. Then there is a* $(k+1)$*-round PIOP* Π' *where the prover sends a single polynomial more than* Π*, and the verifier makes exactly one query to every polynomial sent by the prover in* Π'*.*

Theorem 3 has a surprising consequence for (univariate) polynomial commitment schemes. A polynomial commitment scheme (PCS) [19, 23] is a cryptographic primitive that enables a sender to succinctly commit to a polynomial and then subsequently succinctly open desired evaluations of the committed polynomial. PCS are commonly used to compile PIOPs into arguments. In this context, a highly desirable property of a PCS is "batch opening", where the sender succinctly opens a large set of points (with an opening size that is independent of the number of points). However, not all PCS schemes support batch opening, and often designing a PCS with such property is significantly harder than designing a non-batchable PCS (this was raised in [1] in the context of batching openings of lattice-based PCSs [14, 1, 22, 13]).

When compiling PIOPs into arguments, *designing a PCS with batch openings is not necessary*, as the underlying PIOP can be first transformed into a PIOP where each polynomial sent by the prover is queried exactly once by Theorem 3, and then compiled with a PCS. Since the PIOP queries each polynomial exactly once, no batching property of the PCS is required.

1.3 Comparison with prior work

The efficient batching of polynomial commitments and their openings is a widely studied notion in constructing succinct arguments from PIOPs. We present and compare our results to two main examples: Halo Infinite [5] in the univariate setting, and Hyperplonk [10] in the multivariate setting.

- *Univariate.* [5] considers batching univariate polynomial commitments in the case where the commitments are additively homomorphic (for example, if the commitments are elements of an abelian group) or possesses a linear combination scheme (i.e., it is possible from ℓ commitments and ℓ coefficients to compute a commitment to the linear combination of the committed values with the specified coefficients). Using the language introduced in the recent work [4], it yields a version of Theorem 3 for *homomorphic PIOPs*.
  In contrast, our protocols require *no assumption* on the underlying PIOP (and, consequently, on the underlying polynomial commitment scheme). Further, our protocol requires fewer rounds of interaction.
  Taking as an example batching the evaluation of a single polynomial at many points (the setting of Theorem 9), our protocols yield an MA proof for batch evaluation of a single polynomial at multiple points. In contrast, the protocol of [5] requires an additional round of interaction.
- *Multivariate.* [10, 26] deal with reducing query complexity of *multivariate/multilinear PIOPs*. While the problem statement is similar to the univariate case, batching multilinear polynomial evaluations uses significantly different techniques. In particular, all known techniques to solve this problem utilize the multivariate sumcheck protocol [20], which significantly increases round complexity (typically increasing round complexity to at least $O(m)$ where m is the number of variables) compared to solutions for the univariate case. Our protocol (for univariate polynomials) does not suffer from such an overhead. On the other hand, multilinear batching technique are typically more efficient for the prover (as it needs to perform only a linear amount of field operations). We do not know how to adapt our techniques to achieve linear time complexity, nor how to use them to reduce the round complexity of multilinear polynomial batching, and consider them interesting open problems.

2 Preliminaries

We define objects and notation that we use in this paper.

- For $n \in \mathbb{N}$ we let $[n] \coloneqq \{1, \ldots, n\}$.
- For two functions $f\colon \mathcal{L} \to \mathbb{F}$ and $g\colon \mathcal{D} \to \mathbb{F}$, $\Delta(f, g)$ denotes the fraction of points in $\mathcal{L} \cap \mathcal{D}$ in which they disagree. For a set $\mathcal{S} \subseteq \mathbb{F}^{\mathcal{D}}$, $\Delta(f, \mathcal{S}) := \min_{h \in \mathcal{S}} \Delta(f, h)$.

2.1 Zero multiplicities of polynomials

We give definitions regarding the zeroes of polynomials and their multiplicities.

▶ **Definition 1.** *Let $p \in \mathbb{F}[X]$ and $z \in \mathbb{F}$. We define the following:*

- $\mathsf{zeros}(p) \coloneqq \{x \in \mathbb{F} \ : \ p(x) = 0\}$,
- $\mathsf{mult}(p, z) \coloneqq \max\{t \in \mathbb{N} \ : \ (X - z)^t \mid p(X)\}$,
- $\mathsf{zmult}(p) \coloneqq \{(z, t) \ : \ z \in \mathsf{zeros}(p) \wedge t = \mathsf{mult}(p, z)\}$.

Additionally, we write

$$\mathsf{zmult}(p) \sqsubseteq \mathsf{zmult}(q)$$

if for every $(z, t) \in \mathsf{zmult}(p)$ there exists t' such that $(z, t') \in \mathsf{zmult}(q)$ and $t' \geq t$.

We recall the following fact on the multiplicities of the product of polynomials.

▶ **Fact 2.** *For every $z \in \mathbb{F}$ and $p, q \in \mathbb{F}[X]$,* $\mathsf{mult}(p \cdot q, z) = \mathsf{mult}(p, z) + \mathsf{mult}(q, z)$.

2.2 Interactive proofs and PIOPs

A (public-coin) *interactive proof (IP)* for a language L is a tuple of interactive algorithms $(\mathbf{P}, \mathbf{V})$ that works as follows: $\mathbf{P}$ and $\mathbf{V}$ receive as input an instance $\mathbb{x}$ and then interact over k rounds, where in each round $i \in [\mathsf{k}]$ the prover sends a message m_i and the verifier sends a random message α_i. We require the following properties.

- **Completeness.** For every $\mathbb{x} \in L$,

$$\Pr_{\alpha_1,\ldots,\alpha_{\mathsf{k}}}\left[\begin{array}{l|l} \mathbf{V}(\mathbb{x}, \alpha_1, m_1, \ldots, \alpha_{\mathsf{k}}, m_{\mathsf{k}}) = 1 & \begin{array}{l} m_1 \leftarrow \mathbf{P}(\mathbb{x}) \\ \vdots \\ m_{\mathsf{k}} \leftarrow \mathbf{P}(\mathbb{x}, \alpha_1, \ldots, \alpha_{\mathsf{k}}) \end{array} \end{array}\right] = 1 \ .$$

- **Soundness with error β.** For every $\mathbb{x} \notin L$ and unbounded malicious prover $\tilde{\mathbf{P}}$,

$$\Pr_{\alpha_1,\ldots,\alpha_{\mathsf{k}}}\left[\begin{array}{l|l} \mathbf{V}(\mathbb{x}, \alpha_1, m_1, \ldots, \alpha_{\mathsf{k}}, m_{\mathsf{k}}) = 1 & \begin{array}{l} m_1 \leftarrow \tilde{\mathbf{P}}(\alpha_1) \\ \vdots \\ m_{\mathsf{k}} \leftarrow \tilde{\mathbf{P}}(\alpha_1, \ldots, \alpha_{\mathsf{k}}) \end{array} \end{array}\right] \leq \beta(\mathbb{x}) \ .$$

MA and AMA. An IP is an *MA proof* if $\mathsf{k} = 1$ in the above definition (the interaction consists of a single prover message followed by a single verifier message). An IP is an *AMA proof* if $\mathsf{k} = 2$ and m_1 is empty in the above definition (the interaction consists of a verifier message, a prover message, and a verifier message).

PIOP. A *polynomial IOP* is an IP where the (honest and malicious) prover is restricted to sending only polynomials (over field and of degree determined as part of the definition of the IP), and the verifier is restricted to reading the prover's messages only by evaluating these polynomials on points in the field. In a PIOP, in addition to running times of the parties, we keep track of the number of polynomials sent, their degrees, and the number of evaluations (queries) made by the verifier.

Round-by-round soundness. A *state function* State for an IP $(\mathbf{P}, \mathbf{V})$ has the following structure for any instance $\mathbb{x}$ and partial transcript tr:

- If $\mathbb{x} \notin L$ and $\mathsf{tr} = \emptyset$ then $\mathsf{State}(\mathbb{x}, \mathsf{tr}) = 0$.
- If tr ends in a verifier message and $\mathsf{State}(\mathbb{x}, \mathsf{tr}) = 0$ then $\mathsf{State}(\mathbb{x}, \mathsf{tr} \| m) = 0$ for every prover message m.
- If tr is a full transcript of $(\mathbf{P}, \mathbf{V})$ then $\mathsf{State}(\mathbb{x}, \mathsf{tr}) = \mathbf{V}(\mathbb{x}, \mathsf{tr})$.

A k-round IP has round-by-round soundness error $(\varepsilon_1, \ldots, \varepsilon_{\mathsf{k}})$ if there exists a state function State such that for every $\mathbb{x}$ and transcript tr ending in the prover's i-th message where $\mathsf{State}(\mathbb{x}, \mathsf{tr}) = 0$,

$$\Pr_{\alpha_{i+1}}\left[\mathsf{State}(\mathbb{x}, \mathsf{tr} \| \alpha_{i+1}) = 1\right] \leq \varepsilon_i \ .$$

We sometimes overload the term, and say that the IP has round-by-round soundness error ε if it has round-by-round soundness error $(\varepsilon_1, \ldots, \varepsilon_{\mathsf{k}})$ and $\varepsilon \leq \max\{\varepsilon_1, \ldots, \varepsilon_{\mathsf{k}}\}$.

3 Zero multiplicity distance lemma

We state and prove a technical lemma stating that the sum of quotients of polynomials is close to a low-degree polynomial if and only if each numerator vanishes exactly where the denominator does. Consider polynomials $u_1, \ldots, u_m$ and $w_1, \ldots, w_m$ and the quotient function $f := \sum_{i \in [m]} u_i / w_i$. The lemma says that if f is sufficiently close to a low-degree polynomial then, for every $i \in [m]$, the zeroes of w_i are zeros of u_i and this holds while respecting multiplicities (if z is a zero of w_i with multiplicity t then z is a zero of u_i with multiplicity at least t).

We use this lemma multiple times by setting u_i to be $p(X) - b_i$ and w_i to be $X - a_i$; in this case, by the lemma, if $p(a_i) \neq b_i$ (in which case $X - a_i$ does not divide $p(X) - b_i$) then the function $\sum_i (p(X) - b_i)/(X - a_i)$ is far from *any* low-degree polynomial.

The lemma is formally stated below using the notation of Definition 1. We deem the lemma of independent interest, especially in the context of prover-aided computation (e.g., one can extend our IPs for batch polynomial evaluation to cover any polynomials for which Lemma 3 holds).

▶ **Lemma 3.** *Let* $(u_1, w_1) \ldots, (u_m, w_m)$ *be such that* $u_i \in \mathbb{F}^{\leq d_i}[X]$ *and* $w_i \in \mathbb{F}^{\leq e_i}[X]$, *and assume that the sets* $(\mathsf{zeros}(w_i))_{i \in [m]}$ *are pairwise disjoint. Define* $\mathcal{Z} := \bigcup_i \mathsf{zeros}(w_i)$, $k := \max_i \{d_i - e_i\}$, *and* $s := \sum_i e_i$. *Let* $f \colon \mathbb{F} \setminus \mathcal{Z} \to \mathbb{F}$ *be*

$$f(X) := \sum_{i \in [m]} \frac{u_i(X)}{w_i(X)} \ .$$

If $\Delta(f, \mathbb{F}^{\leq k}[X]) \leq 1 - \frac{k+s+1}{|\mathbb{F}| - |\mathcal{Z}|}$, *then* $\mathsf{zmult}(w_i) \sqsubseteq \mathsf{zmult}(u_i)$ *for every* $i \in [m]$.

Proof. Suppose $\Delta(f, \mathbb{F}^{\leq k}[X]) \leq 1 - \frac{k+s+1}{|\mathbb{F}| - |\mathcal{Z}|}$ and let $v \in \mathbb{F}^{\leq k}[X]$ be the closest polynomial to f (in terms of Hamming distance over $\mathbb{F} \setminus \mathcal{Z}$ and breaking ties arbitrarily). Letting $S := \{z \in \mathbb{F} \setminus \mathcal{Z} : f(z) = v(z)\}$ be the agreement domain between f and v, observe that $|S| > k + s$. Define $A(X) := \prod_{i \in [m]} w_i(X)$, and note that $A(X) \in \mathbb{F}^{\leq s}[X]$, and that $A(z) = 0$ for every $z \in \mathcal{Z}$. By definition of f, for every $z \in \mathbb{F} \setminus \mathcal{Z}$,

$$f(z) \cdot A(z) = \sum_{i \in [m]} \frac{u_i(z)}{w_i(z)} \cdot \prod_{j \in [m]} w_j(z) = \sum_{i \in [m]} u_i(z) \cdot \prod_{j \neq i} w_j(z) \ .$$

Define $g_1, g_2 \in \mathbb{F}^{\leq k+s}[X]$ as follows:

$$g_1(X) := v(X) \cdot A(X) \ ,$$
$$g_2(X) := \sum_{i \in [m]} u_i(X) \cdot \prod_{j \neq i} w_j(X) \ .$$

For every $z \in S$, $g_1(z) = v(z) \cdot A(z) = f(z) \cdot A(z) = g_2(z)$. That is, g_1 and g_2 agree on all points in S, and have degree $\leq k + s$. Since $|S| > k + s$, g_1 and g_2 are the same polynomial. Consider $(z_\ell, t_\ell) \in \mathsf{zmult}(w_\ell)$. Then, $w_\ell(X) = (X - z_\ell)^{t_\ell} \cdot z(X)$, and we can write $g_2(X) = (X - z_\ell)^{t_\ell} \cdot s(X) + u_\ell \cdot \prod_{j \neq \ell} w_j(X)$. Further, by Fact 2,

$$\mathsf{mult}(g_2, z_\ell) = \mathsf{mult}(g_1, z_\ell) = \mathsf{mult}(v, z_\ell) + \mathsf{mult}(A, z_\ell) \geq t_\ell \ ,$$

which implies that $(X - z_\ell)^{t_\ell}$ divides $u_\ell(X) \cdot \prod_{j \neq \ell} w_j(X)$, and since the sets $\mathsf{zeros}(w_j)$ are pairwise disjoint and thus $z_\ell \notin \mathsf{zeros}(w_j)$ for every $j \neq \ell$, we must have that $(X - z_\ell)^{t_\ell}$ divides u_ℓ and thus $\mathsf{mult}(u_\ell, z_\ell) \geq t_\ell$. ◀

We restate Lemma 3 as a protocol for the following language.

▶ **Definition 4.** *Let $\mathbb{F}$ be a field and let $(w_i)_{i\in[m]}$ be a list of polynomials such that the sets $(\mathsf{zeros}(w_i))_{i\in[m]}$ are pairwise disjoint. We define the following language:*

$$\mathsf{Mult}_{\mathbb{F}}[w_1,\ldots,w_m] \coloneqq \left\{ ((d_i,u_i))_{i\in[m]} \;\middle|\; \begin{array}{l} \forall\, i\in[m]: \\ u_i \in \mathbb{F}^{<d_i}[X] \;\wedge\; \mathsf{zmult}(w_i) \sqsubseteq \mathsf{zmult}(u_i) \end{array} \right\} .$$

The construction follows almost immediately.

▶ **Construction 5.** *Fix polynomials $(w_1,\ldots,w_m)$ such that $w_i \in \mathbb{F}^{<e_i}[X]$ and the sets $(\mathsf{zeros}(w_i))_{i\in[m]}$ are pairwise disjoint. Given input $((d_i,u_i))_{i\in[m]}$, the protocol proceeds as follows:*

1. *Prover sends a polynomial $q \in \mathbb{F}^{\le k}[X]$ where $k \coloneqq \max\{d_i - e_i\}$. In the honest case*

$$q(X) := \sum_{i\in[m]} \frac{u_i(X)}{w_i(X)} \ .$$

2. *Verifier samples a random $z \leftarrow \mathbb{F} \setminus \mathcal{Z}$ where $\mathcal{Z} \coloneqq \bigcup_{i\in[m]} \mathsf{zeros}(w_i)$ and accepts if and only if:*

$$q(z) = \sum_{i\in[m]} \frac{u_i(z)}{w_i(z)} \ .$$

Prover and verifier running times. We analyze the running times of the prover and verifier.

- *Prover running time.* For $i \in [m]$, the prover computes the quotients $q_i(X) = u_i(X)/w_i(X)$. This can be done in $O(d_i \cdot e_i)$ field operations via Horner's method for synthetic division [8] or in $O(d_i \cdot \log d_i + e_i \cdot \log e_i)$ field operations via fast Fourier transformations. Obtaining the final quotient can then be done in $O(m \cdot k)$ field operations. The total prover runtime is then

$$O\left(m\cdot k + \sum_{i\in[m]} \min\{d_i\cdot e_i, d_i \log d_i + e_i \log e_i\} \right) \ .$$

- *Verifier running time.* The verifier evaluates $u_1,\ldots,u_m$, $w_1,\ldots,w_m$ and q at a single point and performs m divisions and $3m$ additions. Using the batch inversion method of [21], the divisions can be replaced with $3(m-1)$ multiplications and a single inversion. In total we get:
  - Evaluations: $2m$
  - Additions: $3m$
  - Multiplications: $3m$
  - Inversions: 1

Completeness and soundness. We prove completeness and soundness of Construction 5.

▶ **Lemma 6.** *Construction 5 has perfect completeness.*

Proof. Fix $((d_i,u_i))_{i\in[m]} \in \mathsf{Mult}_{\mathbb{F}}[w_1,\ldots,w_m]$. Since $\mathsf{zmult}(w_i) \sqsubseteq \mathsf{zmult}(u_i)$, u_i/w_i is a polynomial of degree $d_i - e_i \le k$. Then, $q = \sum_{i\in[m]} u_i/w_i$ is indeed a polynomial of degree $\le k$, and thus the verifier will always accept. ◀

▶ **Lemma 7.** *Construction 5 has soundness error at most*

$$\frac{k+s+1}{|\mathbb{F}|-|\mathcal{Z}|} ,$$

where $k, (e_i)_{i\in[m]}, \mathcal{Z}$ *are as is in the construction and* $s := \sum_i e_i$.

Proof. Fix $((d_i, u_i))_{i\in[m]} \notin \mathsf{Mult}_{\mathbb{F}}[w_1, \ldots, w_m]$, let q be the prover's message, and define $f : \mathbb{F} \setminus \mathcal{Z} \to \mathbb{F}$ as

$$f(z) := \sum_{i\in[m]} \frac{u_i(z)}{w_i(z)} .$$

Observe that

$$\Pr[\text{Verifier accepts}] = 1 - \Delta(f, q) \leq 1 - \Delta\left(f, \mathbb{F}^{\leq k}[X]\right) .$$

Since **(1)** $u_i \in \mathbb{F}^{\leq d_i}[X]$, **(2)** $w_i \in \mathbb{F}^{\leq e_i}[X]$, **(3)** $\mathsf{zeros}(w_i)_{i\in[m]}$ are pairwise disjoint, by applying Lemma 3 we have $\Delta\left(f, \mathbb{F}^{\leq k}[X]\right) \geq 1 - \frac{k+s+1}{|\mathbb{F}|-|\mathcal{Z}|}$, and so

$$\Pr[\text{Verifier accepts}] \leq \frac{k+s+1}{|\mathbb{F}|-|\mathcal{Z}|} . \qquad ◀$$

4 Protocols for batch polynomial evaluation

We use Lemma 3 to construct interactive proofs for batch-evaluation of univariate polynomials. In fact, in all of our protocols, the prover sends a single message, which represents a polynomial. The verifier only utilizes this polynomial by evaluating it on a point. Thus, we describe them as PIOPs. Moving to the standard interactive proof model can be done trivially by the prover explicitly sending the polynomials as coefficients, and the verifier can now evaluate them directly.

For the reader's convenience, we have added a description of the verifier's running times in the IP model in the efficiency analysis section of each protocol.

We begin by defining the languages for which we construct PIOPs:

▶ **Definition 8.** *Let* $\mathbb{F}$ *be a field. We define the following languages.*

- ***Single polynomial.***

$$\mathsf{PolyEval}_{\mathbb{F}} := \left\{ (\mathcal{S}, d, p, (b_x)_{x\in\mathcal{S}}) \;\middle|\; \begin{array}{l} \mathcal{S} \subseteq \mathbb{F},\ p \in \mathbb{F}^{\leq d}[X] \\ \forall x \in \mathcal{S} : b_x \in \mathbb{F} \ \wedge\ p(x) = b_x \end{array} \right\} .$$

- ***Multiple polynomials.***

$$\mathsf{MultPolyEval}_{\mathbb{F}} := \left\{ \left(\mathcal{S},\ \{(d_{x,i}, p_{x,i}, b_{x,i})\}_{x\in\mathcal{S}, i\in[n_x]} \right) \;\middle|\; \begin{array}{l} \mathcal{S} \subseteq \mathbb{F} \\ \forall x \in \mathcal{S}, i \in [n_x] : \left(\begin{array}{c} p_{x,i} \in \mathbb{F}^{\leq d_{x,i}}[X],\ b_{x,i} \in \mathbb{F} \\ p_{x,i}(x) = b_{x,i} \end{array} \right) \end{array} \right\} .$$

We can now state our theorems: an MA protocol for $\mathsf{PolyEval}_{\mathbb{F}}$ and AMA protocol for $\mathsf{MultPolyEval}_{\mathbb{F}}$.

▶ **Theorem 9.** *Given any field* $\mathbb{F}$*, there is an MA PIOP for* $\mathsf{PolyEval}_{\mathbb{F}}$ *with the following properties.*

- *Soundness error:* $\frac{|\mathcal{S}|+d}{|\mathbb{F}|-|\mathcal{S}|}$.
- *Polynomials sent: one polynomial of degree* $d-1$.

- *Prover efficiency:* $O(d \cdot |\mathcal{S}|)$ *field operations.*
- *Verifier randomness:* $O(\log |\mathbb{F}|)$ *bits.*
- *Verifier efficiency: One evaluation of* p *and of the polynomial sent by the prover,* $3|\mathcal{S}|$ *field additions and multiplications, and one field inversion.*

▶ **Theorem 10.** *Given any field* $\mathbb{F}$*, there is an AMA PIOP for* $\mathsf{MultPolyEval}_{\mathbb{F}}$ *with the following properties.*

- *Round-by-round soundness error:* $\left(\frac{\max_{x\in\mathcal{S}} n_x - 1}{|\mathbb{F}|}, \frac{|\mathcal{S}|+d}{|\mathbb{F}|-|\mathcal{S}|}\right)$ *where* $d := \max_{x\in\mathcal{S}, i\in[n_x]} d_{x,i}$.
- *Prover message length:* d *field elements.*
- *Prover efficiency:* $O(d \cdot |\mathcal{S}| + \sum_{s\in\mathcal{S}, i\in[n_x]} d_{x,i})$ *field operations.*
- *Verifier randomness:* $O(\log |\mathbb{F}|)$ *bits.*
- *Verifier efficiency: One evaluation of each* $p_{x,i}$ *(all at the same point) and of the polynomial sent by the prover, and, additionally,* $4n$ *additions,* $5n$ *multiplications, and a single inversion (where* $n := \sum_{x\in\mathcal{S}} n_x$*).*

4.1 Proof of Theorem 9

▶ **Construction 11.** *Given input* $(\mathcal{S}, d, p, (b_x)_{x\in\mathcal{S}})$*, the protocol proceeds as follows:*

1. *Prover sends a polynomial* $q \in \mathbb{F}^{\leq d-1}[X]$*. In the honest case*

$$q(X) := \sum_{x\in\mathcal{S}} \frac{p(X) - b_x}{X - x} \ .$$

2. *Verifier samples a random* $z \leftarrow \mathbb{F} \setminus \mathcal{S}$ *and accepts if and only if:*

$$q(z) = \sum_{x\in\mathcal{S}} \frac{p(z) - b_x}{z - x} \ .$$

Prover and verifier running times. We analyze the running times of the prover and verifier.

- *Prover running time.* The prover computes the quotients $q_x(X) = (p(X) - b_x)/(X - x)$ for every $x \in \mathcal{S}$, which can be done in $O(d)$ field operations per quotient via Horner's method for synthetic division [8]. Obtaining q then entails summing these $|\mathcal{S}|$ quotients, for a total of $O(d \cdot |\mathcal{S}|)$ field operations.
- *Verifier running time.* The verifier evaluates p and q at a single point and performs $|\mathcal{S}|$ divisions and $3|\mathcal{S}|$ additions. Using Montgomery's batch inversion method [21], the divisions can be replaced with $3(|\mathcal{S}| - 1)$ multiplications and a single inversion. In total we get:
  - Evaluations: 2
  - Additions: $3|\mathcal{S}|$
  - Multiplications: $3|\mathcal{S}|$
  - Inversions: 1

  Alternatively, if the polynomials are given explicitly by their coefficients, then evaluating them can be done using Horner's method (degree many multiplications and additions) [8]. In this case, the total number of operations that the verifier performs is:
  - Additions: $d + d - 1 + 3|\mathcal{S}| \leq 2d + 3|\mathcal{S}|$
  - Multiplications: $d + d - 1 + 3(|\mathcal{S}| - 1) \leq 2d + 3|\mathcal{S}|$
  - Inversions: 1

Completeness. Fix $(\mathcal{S}, d, p, (b_x)_{x\in\mathcal{S}}) \in \mathsf{PolyEval}_{\mathbb{F}}$. We show that the verifier always accepts when interacting with the honest prover. More specifically, we show that the polynomial

$$q(X) := \sum_{x\in\mathcal{S}} \frac{p(X) - b_x}{X - x} \ ,$$

as defined in the protocol has degree smaller than $d-1$, and so the honest prover can send it to the verifier. Given that the polynomial $\sum_{x\in\mathcal{S}} \frac{p(X)-b_x}{X-x}$ is sent to the verifier, its check will pass with probability 1.

Fix $x \in \mathcal{S}$. Recall that $p(x) = b_x$, and so $p \in \mathbb{F}^{\leq d}[X]$ can be rewritten as

$$p(X) = (X - x) \cdot q_x(X) + b_x \ ,$$

where $q_x \in \mathbb{F}^{\leq d-1}[X]$. Thus,

$$q(X) = \sum_{x\in\mathcal{S}} \frac{p(X) - b_x}{X - x} = \sum_{x\in\mathcal{S}} \frac{((X - x) \cdot q_x(X) + b_x) - b_x}{X - x} = \sum_{x\in\mathcal{S}} q_x(X) \ ,$$

and so $\mathsf{deg}(q) = \max\{\mathsf{deg}(q_x)\} \leq d - 1$.

Soundness. Fix $(\mathcal{S}, d, p, (b_x)_{x\in\mathcal{S}}) \notin \mathsf{PolyEval}_{\mathbb{F}}$, let q be the prover's message, and define $f \colon \mathbb{F} \setminus \mathcal{S} \to \mathbb{F}$ as

$$f(z) := \sum_{x\in\mathcal{S}} \frac{p(z) - b_x}{z - x} \ .$$

Observe that

$$\Pr\left[\text{Verifier accepts}\right] = 1 - \Delta\left(f, q\right) \leq 1 - \Delta\left(f, \mathbb{F}^{\leq d-1}[X]\right) \ .$$

For $x \in \mathcal{S}$ let $u_x(X) := p(X) - b_x$ and $w_x(X) := X - x$. Since $(\mathcal{S}, p, (b_x)_{x\in\mathcal{S}}) \notin \mathsf{PolyEval}_{\mathbb{F}}$, there exists x' so that $p(x') \neq b_{x'}$. Observe that (1) $u_x \in \mathbb{F}^{\leq d}[X]$, (2) $w_x \in \mathbb{F}^{\leq 1}[X]$, (3) $\mathsf{zeros}(w_x) = \{x\}$ are pairwise distinct, and (4) $\mathsf{zeros}(w_{x'}) \not\subseteq \mathsf{zeros}(u_{x'})$.

By applying Lemma 3 we have $\Delta\left(f, \mathbb{F}^{\leq d-1}[X]\right) \geq 1 - \frac{|\mathcal{S}|+d}{|\mathbb{F}|-|\mathcal{S}|}$, and so

$$\Pr\left[\text{Verifier accepts}\right] \leq \frac{|\mathcal{S}| + d}{|\mathbb{F}| - |\mathcal{S}|} \ .$$

4.2 Proof of Theorem 10

▶ **Construction 12.** *Fix inputs $\mathcal{S}$, and $\{(d_{x,i}, p_{x,i}, b_{x,i})\}_{x\in\mathcal{S}, i\in[n_x]}$ for $\mathsf{MultPolyEval}_{\mathbb{F}}$.*

1. *Verifier samples $r \leftarrow \mathbb{F}$ and sends it to the prover.*
2. *Prover sends a polynomial $q \in \mathbb{F}^{\leq d-1}[X]$ where $d := \max_{x\in\mathcal{S}, i\in[n_x]} d_{x,i}$. In the honest case, q defined as follows:*

$$q(X) := \sum_{x\in\mathcal{S}} \frac{\sum_{i\in[n_x]} r^{i-1} \cdot (p_{x,i}(X) - b_{x,i})}{X - x} \ .$$

3. *Verifier samples a random $z \leftarrow \mathbb{F} \setminus \mathcal{S}$ and accepts if and only if:*

$$q(z) = \sum_{x\in\mathcal{S}} \frac{\sum_{i\in[n_x]} r^{i-1} \cdot (p_{x,i}(z) - b_{x,i})}{z - x} \ .$$

Prover and verifier running times. We analyze the prover and verifier running times:

- *Prover running time.* For $x \in \mathcal{S}$ the prover computes the polynomial $t_x(X) = \sum_{i\in[n_x]} r^{i-1} \cdot (p_{x,i}(X) - b_{x,i})$ in $O(\sum_{s\in\mathcal{S},i\in[n_x]} d_{x,i})$ field operations. For $x \in \mathcal{S}$, the prover then computes $q_x(X) = t_x(X)/(X - x)$ in $O(d)$ operations per quotient again via Horner's rule for division [8]. Finally, computing q requires $O(|\mathcal{S}| \cdot d)$ field operations. In total, this results in $O(|\mathcal{S}| \cdot d + \sum_{s\in\mathcal{S},i\in[n_x]} d_{x,i})$ field operations.
- *Verifier running time.* The verifier evaluates each polynomial $\{p_{x,i}\}_{x\in\mathcal{S},\, i\in[n_x]}$ once (note that if a polynomial appears twice in the list of polynomials, it is still evaluated only once), and evaluates q once. It then computes the powers r^i using at most n multiplications (recall that $n = \sum_{x\in\mathcal{S}} n_x$). It then performs $|\mathcal{S}|$ divisions, n multiplications, and $2n+2|\mathcal{S}|$ additions. Using Montgomery's batch inversion method [21], the divisions can be replaced with $3(|\mathcal{S}| - 1)$ multiplications and a single inversion. In total we get:
  - Evaluations: $n + 1$
  - Additions: $2n + 2|\mathcal{S}| \le 4n$
  - Multiplications: $2n + 3|\mathcal{S}| \le 5n$
  - Inversions: 1

  Alternatively, if the polynomials are given explicitly by their coefficients, then evaluating them can be done using Horner's method (degree many multiplications and additions) [8]. In this case, the total number of operations that the verifier performs is:
  - Additions: $dn + d - 1 + 2n + 2|\mathcal{S}| \le (n + 1)d + 4n$
  - Multiplications: $dn + d - 1 + 2n + 3(|\mathcal{S}| - 1) \le (n + 1)d + 5n$
  - Inversions: 1

Completeness. Fix $(\mathcal{S}, (d_{x,i}, p_{x,i}, b_{x,i})_{x\in\mathcal{S},i\in[n_x]}) \in \mathsf{MultPolyEval}_{\mathbb{F}}$. We show that for every r the polynomial

$$q(X) := \sum_{x\in\mathcal{S}} \frac{\sum_{i\in[n_x]} r^{i-1} \cdot (p_{x,i}(X) - b_{x,i})}{X - x} \ ,$$

as defined in the protocol has degree bounded by $d - 1$, and so can be sent by the prover. Once this is established, then by definition the verifier will accept with probability 1.

For every $x \in \mathcal{S}$ and $p_{x,i} \in \mathbb{F}^{\le d_{x,i}}[X]$, we have that $p_{x,i}(x) = b_{x,i}$. Thus, $p_{x,i}$ can be rewritten as

$$p_{x,i}(X) \coloneqq (X - x) \cdot q_{x,i}(X) + b_{x,i} \ ,$$

where $q_{x,i} \in \mathbb{F}^{\le d_{x,i}-1}[X]$. Therefore, for every $r \in \mathbb{F}$ we can rewrite

$$\begin{aligned} q(X) &= \sum_{x\in\mathcal{S}} \frac{\sum_{i\in[n_x]} r^{i-1} \cdot (p_{x,i}(X) - b_{x,i})}{X - x} \\ &= \sum_{x\in\mathcal{S}} \frac{\sum_{i\in[n_x]} r^{i-1} \cdot (((X - x) \cdot q_{x,i}(X) + b_{x,i}) - b_{x,i})}{X - x} \\ &= \sum_{x\in\mathcal{S}} \sum_{i\in[n_x]} r^{i-1} \cdot q_{x,i}(X) \ . \end{aligned}$$

Therefore $\mathsf{deg}(q) = \max\{\mathsf{deg}(q_{x,i})\} \le d - 1$.

Round-by-round soundness. We analyze the round-by-round soundness of our protocol. Before describing the state function, we prove a useful claim:

▷ Claim 13. Fix $\mathcal{S} \subseteq \mathbb{F}$, $x \in \mathcal{S}$, and for $i \in [n]$, $p_{x,i} \in \mathbb{F}^{\leq d_{x,i}}[X]$ and $b_{x,i} \in \mathbb{F}$. Set $d \coloneqq \max_{x,i} d_{x,i}$ and define the function $f_r \colon \mathbb{F} \setminus \mathcal{S} \to \mathbb{F}$ as follows

$$f_r(X) = \sum_{x \in \mathcal{S}} \frac{\sum_{i \in [n_x]} r^{i-1} \cdot (p_{x,i}(X) - b_{x,i})}{X - x} \quad . \tag{1}$$

If for some $y \in \mathcal{S}$ and $i \in [n_y]$ we have that $p_{y,i}(y) \neq b_{y,i}$ then

$$\Pr\left[\Delta(f_r, \mathbb{F}^{\leq d-1}[X]) \leq 1 - \frac{|\mathcal{S}| + d}{|\mathbb{F}| - |\mathcal{S}|}\right] \leq \frac{\max_{x \in \mathcal{S}} n_x - 1}{|\mathbb{F}|} \quad .$$

Proof. Let $y \in \mathcal{S}$ and $i \in [n_y]$ be such that $p_{y,i}(y) \neq b_{y,i}$ Then $g_y(X) \coloneqq \sum_{i \in [n_y]} X^{i-1} \cdot (p_{y,i}(y) - b_{y,i})$ is not the zero polynomial. Noting that $\mathsf{deg}(g_y)) \leq n_y - 1$, we have that

$$\Pr_{r \leftarrow \mathbb{F}}[g_y(r) = 0] \leq \frac{n_y - 1}{|\mathbb{F}|} \quad .$$

For $x \in \mathcal{S}$ and $r \in \mathbb{F}$ let $u_x^{(r)}(X) \coloneqq \sum_{i \in [n_x]} r^{i-1} \cdot (p_{x,i}(X) - b_{x,i})$ and observe that $g_y(r) = u_y^{(r)}(y)$. Letting $w_x(X) \coloneqq X - x$, notice that $\mathsf{zeros}(w_x) = \{x\}$ and so $\{\mathsf{zeros}(w_x)\}_{x \in \mathcal{S}}$ are pairwise disjoint. Since $f_r(X) = \sum_{x \in \mathcal{S}} u_x^{(r)}(X)/w_x(X)$, by Lemma 3 we have that if the conditions of Equation (1) hold then it must be the case that $\mathsf{zeros}(w_x) \subseteq \mathsf{zeros}(u_x^{(r)})$ for every $x \in \mathcal{S}$ and thus the condition must in particular hold for $x = y$ i.e. $\{y\} = \mathsf{zeros}(w_y) \subseteq \mathsf{zeros}(u_y^{(r)})$. In other words, this can only happen if $g_y(r) = u_y^{(r)}(y) = 0$. Then,

$$\Pr_{r \leftarrow \mathbb{F}}\left[\Delta(f_r, \mathbb{F}^{\leq d-1}[X]) \leq 1 - \frac{|\mathcal{S}| + d}{|\mathbb{F}| - |\mathcal{S}|}\right] = \Pr_{r \leftarrow \mathbb{F}}[g_y(r) = 0] \leq \frac{n_y - 1}{|\mathbb{F}|} \leq \max_{x \in \mathcal{S}} \frac{n_x - 1}{|\mathbb{F}|} \quad . \quad \triangleleft$$

▶ **Lemma 14.** *The protocol has round-by-round soundness error*

$$(\varepsilon^{\mathsf{rand}}, \varepsilon^{\mathsf{prox}}) = \left(\frac{\max_{x \in \mathcal{S}} n_x - 1}{|\mathbb{F}|}, \frac{|\mathcal{S}| + d}{|\mathbb{F}| - |\mathcal{S}|}\right) \quad .$$

Proof. We define a state function State, and bound each term individually. Below, we denote the input to the protocol by $\mathbb{x} \coloneqq \left(\mathcal{S}, \{(d_{x,i}, p_{x,i}, b_{x,i})\}_{x \in \mathcal{S}, i \in [n_x]}\right)$.

0. Initial state. We set $\mathsf{State}(\mathbb{x}, \emptyset) = 1$ if and only if $\mathbb{x} \in \mathsf{MultPolyEval}_{\mathbb{F}}$.

1. Bounding $\varepsilon^{\mathsf{rand}}$. At this stage, tr is empty, and the verifier sends $r \leftarrow \mathbb{F}$.

- *State function.* Define the function $f_r : \mathbb{F} \setminus \mathcal{S} \to \mathbb{F}$ as follows

$$f_r(X) \coloneqq \sum_{x \in \mathcal{S}} \frac{\sum_{i \in [n_x]} r^{i-1} \cdot (p_{x,i}(X) - b_{x,i})}{X - x} \quad .$$

  We set $\mathsf{State}(\mathbb{x}, r) = 1$ if and only if

$$\Delta(f_r, \mathbb{F}^{\leq d-1}[X]) \leq 1 - \frac{|\mathcal{S}| + d}{|\mathbb{F}| - |\mathcal{S}|} \quad .$$

- *Bounding the error.* Suppose that $\mathsf{State}(\mathbb{x}, \emptyset) = 0$. This implies that there exist $x \in \mathcal{S}$ and $i \in [n_x]$ with $p_{x,i}(x) \neq b_{x,i}$. Then, we are exactly in the setting of Claim 13 and

$$\begin{aligned}\Pr\left[\mathsf{State}(\mathbb{x}, r) = 1 | \mathsf{State}(\mathbb{x}, \emptyset) = 0\right] &= \Pr\left[\Delta(f_r, \mathbb{F}^{\leq d-1}[X]) \leq 1 - \frac{|\mathcal{S}| + d}{|\mathbb{F}| - |\mathcal{S}|} \;\middle|\; \mathsf{State}(\mathbb{x}, \emptyset) = 0\right] \\ &\leq \max_{x \in \mathcal{S}} \frac{n_x - 1}{|\mathbb{F}|} \quad .\end{aligned}$$

2. **Bounding** $\varepsilon^{\text{prox}}$. At this stage $\mathsf{tr} := (r, q)$ and the verifier sends $z \leftarrow \mathbb{F} \setminus \mathcal{S}$.
   - *State function.* We set $\mathsf{State}(\mathbb{x}, \mathsf{tr} \| z) = 1$ if and only if the verifier accepts given $\mathbb{x}$, tr, and z, i.e. if $q(z) = f_r(z)$.
   - *Bounding the error.* Suppose that $\mathsf{State}(\mathbb{x}, \mathsf{tr}) = 0$. Then $\Delta(f_r, q) \geq \Delta(f_r, \mathbb{F}^{\leq d-1}[X]) > 1 - \frac{|\mathcal{S}|+d}{|\mathbb{F}|-|\mathcal{S}|}$, and thus the probability that $f_r(z) = q(z)$ for $z \leftarrow \mathbb{F} \setminus \mathcal{S}$ is at most $\frac{|\mathcal{S}|+d}{|\mathbb{F}|-|\mathcal{S}|}$. ◀

5 Query reduction for PIOPs

We describe a compiler that, using Theorem 10, transforms a PIOP into a PIOP where each proof polynomial is queried exactly once.

▶ **Theorem 15.** *Given a* k*-round PIOP* Π *for a relation* $\mathcal{R}$ *with maximal degree* d *and total query complexity* q *we construct a* $\mathsf{k}+1$ *PIOP* Π' *for* $\mathcal{R}$ *with the following properties:*

- *The prover in* Π' *sends exactly the same messages as* Π *with the addition of* q *field elements and a single polynomial of degree* $d-1$.
- *Every polynomial sent by the prover in* Π' *is queried exactly once, all on the same field element.*

▶ Remark 16. For generality, we write the protocol in a way that enables the following generalizations of the above theorem:

- *Polynomial IOPs of proximity.* The verifier is given oracle access to part of its input $\mathbb{y}$. We further note that if the implicit input contains polynomials with evaluation access, then they can also be batched into the batch evaluation protocol (these can be considered as an initial prover message).
- *Polynomial IOP for Relations.* The IOP is for a relation $\mathcal{R} = \{(\mathbb{x}, \mathbb{w})\}$ and the honest prover is given the witness $\mathbb{w}$ (in a proof of proximity we would consider $\mathcal{R} = \{(\mathbb{x}, \mathbb{y}), \mathbb{w}\}$). In many use-cases this allows the honest prover to be more efficient.
- *Round-by-round knowledge.* We show that if the original PIOP has round-by-round knowledge soundness with an extractor $\mathbf{E}$, then the new IOP also has round-by-round knowledge soundness.

See [12] for formal definitions of all of the above generalizations.

Proof. We detail the transformation and then prove its properties. While our transformation uses Theorem 10 directly, for convenience and minor optimizations we spell out below the corresponding changes from Theorem 10. Let $\Pi = (\mathbf{P}, \mathbf{V})$ we begin by giving notation to the parameters of Π:

- In round $i \in [\mathsf{k}]$, the prover sends n_i polynomials, where for $j \in [n_i]$ the j-th polynomial $f_{i,j}$ has degree $d_{i,j}$. Let $\mathsf{q}_{i,j}$ be the number of queries made by the verifier to $f_{i,j}$.
- In round i, the verifier sends r_i bits of randomness.
- Let t be the maximal number of polynomials that are evaluated on the same point, and m be the maximal number of distinct points evaluated on any of the polynomials sent.
- Π has round-by-round knowledge soundness error $(\varepsilon_1, \ldots, \varepsilon_{\mathsf{k}})$ with state function $\mathsf{State}_{\mathsf{poly}}$, extractor $\mathbf{E}$, and extraction time et.
- For simplicity, the protocol described below does not explicitly state non-oracle messages sent by the prover, but these are straightforward to add. l denotes the total number of such field elements sent.

▶ **Construction 17.** *We construct a PIOP for $\mathcal{R}$ given a PIOP $(\mathbf{P}, \mathbf{V})$ for $\mathcal{R}$.*

0. ***Inputs:*** *The honest prover receives as input $(\mathbb{x}, \mathbb{y}, \mathbb{w}) \in \mathcal{R}$, and the verifier receives $\mathbb{x}$ as an explicit input and $\mathbb{y}$ as an oracle input.*
1. ***PIOP interaction phase:*** *For $i = 1, \ldots, \mathsf{k}$:*
   a. ***PIOP prover message:*** *The prover computes and sends $f_{i,1}, \ldots, f_{i,n_i}$ where $f_{i,j} \in \mathbb{F}^{\leq d_{i,j}}[X]$. In the honest case $(f_{i,1}, \ldots, f_{i,n_i}) := \mathbf{P}(\mathbb{x}, \mathbb{y}, \mathbb{w}, \alpha_1, \ldots, \alpha_{i-1})$.*
   b. ***PIOP verifier message:*** *The verifier samples and send $\alpha_i \leftarrow \{0,1\}^{\mathsf{r}_i}$.*
2. ***Batching interaction phase:***
   a. ***Send query results:*** *The prover sends an array of field elements $(A_{i,j})_{i \in [\mathsf{k}], j \in [n_i]}$ with $|A_{i,j}| = \mathsf{q}_{i,j}$. In the honest case, the prover simulates the execution of*

$$\mathbf{V}^{\mathbb{y}, (f_{i,j})_{i \in [\mathsf{k}], j \in [n_i]}}(\mathbb{x}, \alpha_1, \ldots, \alpha_{\mathsf{k}}) \ .$$

   For every $i \in [\mathsf{k}]$ and $j \in [n_i]$ set:
   - *$\mathcal{Q}_{i,j} \subseteq \mathbb{F}$ be the set of queries made by $\mathbf{V}$ to $f_{i,j}$.*
   - *$\phi_{i,j} \colon [\mathsf{q}_{i,j}] \to \mathcal{Q}_{i,j}$ be the mapping where $\phi_{i,j}(k)$ returns the k-th query made to $f_{i,j}$;*

   Then, for $k \in [\mathsf{q}_{i,j}]$, the prover sets $A_{i,j}[k] := f_{i,j}(\phi_{i,j}(k))$.
   b. ***Sample combination randomness:*** *the verifier sends randomness $r \leftarrow \mathbb{F}$.*
   c. ***Send interpolation polynomial:*** *the prover sends a polynomial $q \in \mathbb{F}^{\leq d-1}[X]$ where $d := \max_{i,j} d_{i,j}$. In the honest case,*

$$q(X) := \sum_{q \in \mathcal{Q}} \frac{\sum_{k \in [t_q]} r^{k-1} \cdot \left(f_{\psi_q(k)}(X) - f_{\psi_q(k)}(q)\right)}{X - q} \tag{2}$$

   where:
   - *$\mathcal{Q} = \bigcup_{i,j} \mathcal{Q}_{i,j}$ is the set of all queries made by the verifier, and*
   - *For $q \in \mathcal{Q}$: $S_q := \{(i,j) : q \in \mathcal{Q}_{i,j}\}$, $t_q := |S_q|$, and $\psi_q : [t_q] \to S_q$ is an arbitrary ordering of S_q.*

   d. ***Choose proximity randomness:*** *the verifier determines $\mathcal{Q}$ as before, samples a random $z \leftarrow \mathbb{F} \setminus \mathcal{Q}$ and sends it to the prover.*
3. ***Verifier decision phase:***
   a. ***PIOP decision:*** *Check that $\mathbf{V}^{\mathbb{y}, (f_{i,j})_{i \in [\mathsf{k}], j \in [n_i]}}(\mathbb{x}, \alpha_1, \ldots, \alpha_{\mathsf{k}}) = 1$, where the k-th query to $f_{i,j}$ is answered by $A_{i,j}[k]$ and queries to $\mathbb{y}$ are answered by querying $\mathbb{y}$ directly (reject if $\mathbf{V}$ rejects).*
   b. ***Batching decision:*** *Letting ψ_q be as before, check that*

$$q(z) = \sum_{q \in \mathcal{Q}} \frac{\sum_{k \in [t_q]} r^{k-1} \cdot \left(f_{\psi_q(k)}(z) - A_{\psi_q(k)}[\phi^{-1}(q)]\right)}{z - q} \ ,$$

   querying q and $(f_{i,j})_{i,j}$ accordingly.

Complexity measures. The new PIOP has the following parameters:

- *Rounds.* $\mathsf{k} + 1$.
- *Polynomials sent.* In round $i \in [\mathsf{k}]$ the prover sends n_i polynomials, where, for $j \in [n_i]$, the function has degree $d_{i,j}$. In round $\mathsf{k} + 1$, the prover sends a single polynomial of degree $d - 1$.
- *Non-oracle messages.* The prover sends $\mathsf{l}_{\mathsf{poly}}$ field elements in the interaction of Π, and q additional field elements.
- *Evaluation queries.* The verifier makes one evaluation query to each of the $1 + \sum_{i \in [\mathsf{k}]} n_i$ polynomials sent in the protocol.

Round-by-round soundness. We prove that Π' has round-by-round soundness error

$$(\varepsilon_1, \ldots, \varepsilon_{\mathsf{k}}, \varepsilon_{\mathsf{com}}, \varepsilon_{\mathsf{prx}}) = \left(\varepsilon_1, \ldots, \varepsilon_{\mathsf{k}}, \frac{t-1}{|\mathbb{F}|}, \frac{d+m}{|\mathbb{F}|-m}\right) \ ,$$

with extraction time $O(\mathsf{et})$.

We define a state function State, and bound each term individually.

0. State function for empty transcript We set $\mathsf{State}(\mathbb{x}, \mathbb{y}, \emptyset) = \mathsf{State}_{\mathsf{poly}}(\mathbb{x}, \mathbb{y}, \emptyset)$.

1. PIOP verifier message: At this point of the interaction, the transcript has the form

$$\mathsf{tr} = \begin{pmatrix} ((f_{\ell,1}, \ldots, f_{\ell,m_\ell}), \alpha_\ell)_{\ell<i} \\ (f_{i,1}, \ldots, f_{i,m_i}), \end{pmatrix} \ .$$

The verifier chooses $\alpha_i \leftarrow \{0,1\}^{\mathsf{r}_i}$.

- *State function:* We set $\mathsf{State}(\mathbb{x}, \mathbb{y}, \mathsf{tr}\|\alpha_i) = \mathsf{State}_{\mathsf{poly}}(\mathbb{x}, \mathbb{y}, \mathsf{tr}\|\alpha_i)$.
- *Bounding the error:* Suppose that

$$\Pr[\mathsf{State}(\mathbb{x}, \mathbb{y}, \mathsf{tr}\|\alpha_i) = 1 | \mathsf{State}(\mathbb{x}, \mathbb{y}, \mathsf{tr}) = 0] > \varepsilon_i$$

Then, $\Pr[\mathsf{State}_{\mathsf{poly}}(\mathbb{x}, \mathbb{y}, \mathsf{tr}\|\alpha_i) = 1 | \mathsf{State}_{\mathsf{poly}}(\mathbb{x}, \mathbb{y}, \mathsf{tr}) = 0] > \varepsilon_i$ and by round-by-round knowledge soundness of Π we have $(\mathbb{x}, \mathbb{y}, \mathbf{E}(\mathbb{x}, \mathbb{y}, \mathsf{tr})) \in \mathcal{R}$.

2. Combination randomness: At this point of the interaction, the transcript has the form

$$\mathsf{tr} = \begin{pmatrix} ((f_{i,1}, \ldots, f_{i,m_i}), \alpha_i)_{i \le \mathsf{k}} \\ (A_{i,j})_{i \in [\mathsf{k}], j \in [m_i]} \end{pmatrix} \ .$$

The verifier chooses $r \leftarrow \mathbb{F}$.

- *State function:* We set $\mathsf{State}(\mathbb{x}, \mathbb{y}, \mathsf{tr}\|r) = 1$ if both of the following hold:
  - **a.** $\mathbf{V}$ accepts when given access to $\mathbb{y}$ and given query answers according to $A_{i,j}$.
  - **b.** Letting h_r be the right-hand side of Equation (2),

$$\Delta(h_r, \mathbb{F}^{\le d-1}[X]) \le 1 - \frac{d+m}{|\mathbb{F}|-m} \ .$$

- *Bounding the error:* We show that

$$\Pr[\mathsf{State}(\mathbb{x}, \mathbb{y}, \mathsf{tr}\|r) = 1 | \mathsf{State}(\mathbb{x}, \mathbb{y}, \mathsf{tr}) = 0] \le \frac{\max_{q \in \mathcal{Q}} t_q - 1}{|\mathbb{F}|} = \frac{t-1}{|\mathbb{F}|} \ .$$

Since the probability is always bounded from above, we do not require the extractor to be able to extract. Suppose that $\mathsf{State}(\mathbb{x}, \mathbb{y}, \mathsf{tr}) = \mathsf{State}_{\mathsf{poly}}(\mathbb{x}, \mathbb{y}, \mathsf{tr}) = 0$. Then, if it holds that for every $i \in \mathsf{k}$, $j \in [n_i]$ and $k \in [\mathsf{q}_{i,j}]$ $A_{i,j}[k] = f_{i,j}(\phi_{i,j}(k))$ the $\mathbf{V}$ can never accept by the round-by-round knowledge soundness of Π, and we are done. Thus, there must exists a triple (i,j,k) such that $A_{i,j}[k] \neq f_{i,j}(\phi_{i,j}(k))$, and we are left to bound the second item in this case. The claimed error follows then directly by applying Claim 13.

3. Proximity randomness: At this point of the interaction, the transcript has the form

$$\mathsf{tr} = \begin{pmatrix} ((f_{i,1}, \ldots, f_{i,m_i}), \alpha_i)_{i \le \mathsf{k}} \\ (A_{i,j})_{i \in [\mathsf{k}], j \in [m_i]}, r, q \end{pmatrix} \ .$$

The verifier chooses $z \leftarrow \mathbb{F} \setminus \mathcal{Q}$.

- *State function:* We set $\mathsf{State}(\mathbb{x}, \mathbb{y}, \mathsf{tr}\|r) = 1$ if both of the following hold:
  - **a.** $\mathbf{V}$ accepts when given access to $\mathbb{y}$ and given query answers according to $A_{i,j}$.
  - **b.** $q(z) = f_r(z)$ where f_r is the right hand side of Equation (2) with r fixed.

- *Bounding the error:* Since the probability is always bounded from above, we do not require the extractor to be able to extract. Suppose that $\mathsf{State}(\mathbb{x}, \mathbb{y}, \mathsf{tr}) = 0$. Then, by definition, $\Delta(f_r, q) \geq \Delta(f_r, \mathbb{F}^{\leq d}[X]) > 1 - \frac{d+m}{|\mathbb{F}|-m}$, and thus the probability that $f_r(z) = q(z)$ for $z \leftarrow \mathbb{F} \setminus \mathcal{Q}$ is at most $\frac{d+m}{|\mathbb{F}|-m}$. ◀

6 Improving the verifier time of STIR

Our techniques can be used to improve the verifiers that use quotienting of polynomials. We discuss a concrete example: the STIR protocol [2]. Specifically, we show how to improve the STIR protocol so to that the verifier time matches that of the WHIR protocol [3] (which was designed specifically with verification time in mind).

Overview of STIR. The STIR protocol is an interactive oracle proof of proximity[3] for proving that a function $f \colon \mathcal{L} \to \mathbb{F}$ is close to the Reed–Solomon code $\mathsf{RS}[d, \mathcal{L}]$ (i.e., it is close in Hamming weight to the evaluation over $\mathcal{L}$ of a degree d polynomial). Given a security parameter λ and a (small constant) "folding parameter" $k > 2$, the protocol consists of $M = O(\log d)$ iterations where, in iteration i, the prover sends a function g_i, and the verifier has two degree t_i polynomials Ans_i and V_i where $t_i \approx \lambda/i$. The verifier then virtually defines the function

$$f_i(X) := \frac{g_i(X) - \mathsf{Ans}_i(X)}{V_i(X)} \ .$$

The verifier then evaluates f_i on $k \cdot t_{i+1}$ points (where $t_{i+1} \approx \lambda/(i+1)$), which reduces to making $k \cdot t_{i+1}$ queries to g_i, and evaluating Ans_i and V_i on t_{i+1} points. It then uses the answers to generate Ans_{i+1}, V_{i+1}, and (in the final round) do a small number of efficient equality tests for final verification.

In [2], in order to evaluate Ans_i and V_i on all points, the verifier performs roughly $\Omega((t_i + kt_{i-1}) \cdot \log^2(t_i + kt_{i-1}) = \Omega(kt_i \cdot \log^2(kt_i))$ field operations using the classical batch polynomial evaluation algorithm. In fact, in practice the hidden constants in the classical batch polynomial evaluation algorithm are large, and so it is more common to use Horner's method, which yields asymptotic running time $\Omega(kt_i^2)$. Thus, overall, the verifier complexity is $\Omega\left(\sum_{i=1}^{M} kt_i \cdot \log^2(kt_i)\right)$.

Improving verification time. Our proofs for batch polynomial evaluation facilitate the reduction of the STIR verifier complexity by utilizing the prover to compute Ans_i and V_i much faster. Following the interaction phase, the prover derives the queries $\mathcal{Q}_i$ that the verifier will make to f_i and sends to the verifier the points $y_{i,x}$ and $z_{i,x}$ where (in the honest case), for every $x \in \mathcal{Q}_i$, it holds that $\mathsf{Ans}_i(x) = y_{i,x}$ and $V_i(x) = z_{i,x}$. The prover and verifier then engage in the batch polynomial evaluation protocol of Theorem 10 in order to prove that these are, indeed, the correct evaluations of the polynomials. Now that the verifier is convinced of the validity of these points, it can run the STIR verification procedure as if it had computed the evaluations independently.

[3] An interactive oracle proof of proximity is a generalization of an interactive proof, where the prover messages and the original input are oracles to which the verifier makes a small number of queries (rather than reading them in their entirety).

Overall, the verifier performs $O(k \cdot \sum_{i \in [M]} t_i)$ field operations, reducing the factors of $\log^2(t_i + k \cdot t_{i+1})$. This comes at a minor cost to prover communication as the prover must send an additional $O(k \cdot \sum_{i \in [M]} t_i)$ field elements. We further note that this optimization is likely to be useful in the implementation of other protocols that utilize similar techniques, such as the ARC accumulation scheme [7].

Comparison with WHIR. This optimized protocol, which we refer to as STIR-SHAKE, asymptotically matches the verifier complexity of the recently introduced WHIR [3] protocol. WHIR is also an interactive oracle proof of proximity for Reed–Solomon[4] which has a similar basic structure to STIR-SHAKE, and where the verifier has the same asymptotic complexity of $O(\sum_{i=1}^{M} k \cdot t_i)$ field operations.

References

1 Martin R. Albrecht, Giacomo Fenzi, Oleksandra Lapiha, and Ngoc Khanh Nguyen. SLAP: succinct lattice-based polynomial commitments from standard assumptions. In *Proceedings of the 43rd Annual International Conference on Theory and Application of Cryptographic Techniques*, EUROCRYPT '24, pages 90–119, 2024. doi:10.1007/978-3-031-58754-2_4.

2 Gal Arnon, Alessandro Chiesa, Giacomo Fenzi, and Eylon Yogev. STIR: Reed–solomon proximity testing with fewer queries. In *Proceedings of the 44th Annual International Cryptology Conference*, CRYPTO '24, pages 380–413, 2024. doi:10.1007/978-3-031-68403-6_12.

3 Gal Arnon, Alessandro Chiesa, Giacomo Fenzi, and Eylon Yogev. WHIR: Reed–solomon proximity testing with super-fast verification. In *Proceedings of the 44th Annual International Conference on Theory and Application of Cryptographic Techniques*, EUROCRYPT '25, 2025.

4 Dan Boneh, Trisha Datta, Fernando Rex, Kamilla Nazirkhanova, and Alin Tomescu. Dekart-proof: Efficient vector range proofs and their applications. *IACR Cryptol. ePrint Arch.*, page 1159, 2025. URL: https://eprint.iacr.org/2025/1159.

5 Dan Boneh, Justin Drake, Ben Fisch, and Ariel Gabizon. Halo infinite: Proof-carrying data from additive polynomial commitments. In *Proceedings of the 41st Annual International Cryptology*, CRYPTO '21, pages 649–680, 2021. doi:10.1007/978-3-030-84242-0_23.

6 Benedikt Bünz, Ben Fisch, and Alan Szepieniec. Transparent snarks from DARK compilers. In *Proceedings of the 39th Annual International Conference on the Theory and Applications of Cryptographic Techniques*, EUROCRYPT '20, pages 677–706, 2020. doi:10.1007/978-3-030-45721-1_24.

7 Benedikt Bünz, Pratyush Mishra, Wilson Nguyen, and William Wang. Arc: Accumulation for reed-solomon codes. *IACR Cryptol. ePrint Arch.*, page 1731, 2024. URL: https://eprint.iacr.org/2024/1731.

8 C Sidney Burrus, James W Fox, Gary A Sitton, and Sven Treitel. Horner's method for evaluating and deflating polynomials. *DSP Software Notes, Rice University, Nov*, 26, 2003.

9 Vitalik Buterin. Barycentric evaluation trees for polynomial commitments. https://hackmd.io/@vbuterin/barycentric_evaluation, 2023. Accessed: 2025-05-21.

10 Binyi Chen, Benedikt Bünz, Dan Boneh, and Zhenfei Zhang. Hyperplonk: Plonk with linear-time prover and high-degree custom gates. In *Proceedings of the 42nd Annual International Conference on the Theory and Applications of Cryptographic Techniques*, EUROCRYPT '23, pages 499–530, 2023. doi:10.1007/978-3-031-30617-4_17.

11 Alessandro Chiesa, Yuncong Hu, Mary Maller, Pratyush Mishra, Noah Vesely, and Nicholas Ward. Marlin: Preprocessing zkSNARKs with universal and updatable SRS. In *Proceedings of the 39th Annual International Conference on the Theory and Applications of Cryptographic Techniques*, EUROCRYPT '20, 2020.

[4] WHIR also introduces the ability to verify a more general class of codes called "constrained" Reed–Solomon codes, but we focus on testing proximity to Reed–Solomon codes in this comparison.

12 Alessandro Chiesa and Eylon Yogev. Building cryptographic proofs from hash functions. *URL: https://github. com/hash-based-snargs-book*, 2024.

13 Valerio Cini, Giulio Malavolta, Ngoc Khanh Nguyen, and Hoeteck Wee. Polynomial commitments from lattices: Post-quantum security, fast verification and transparent setup. In *Proceedings of the 44th Annual International Cryptology Conference*, CRYPTO '24, pages 207–242, 2024. doi:10.1007/978-3-031-68403-6_7.

14 Giacomo Fenzi, Hossein Moghaddas, and Ngoc Khanh Nguyen. Lattice-based polynomial commitments: Towards asymptotic and concrete efficiency. *J. Cryptol.*, 37(3):31, 2024. doi:10.1007/S00145-024-09511-8.

15 Charles M. Fiduccia. Polynomial evaluation via the division algorithm the fast fourier transform revisited. In *Proceedings of the Fourth Annual ACM Symposium on Theory of Computing*, STOC '72, pages 88–93, 1972. doi:10.1145/800152.804900.

16 Oded Goldreich and Guy N. Rothblum. Simple doubly-efficient interactive proof systems for locally-characterizable sets, 2017. ECCC TR17-018.

17 Oded Goldreich and Guy N. Rothblum. Counting t-cliques: Worst-case to average-case reductions and direct interactive proof systems. In *Proceedings of the 59th Annual IEEE Symposium on Foundations of Computer Science*, FOCS '18, pages 77–88, 2018. doi:10.1109/FOCS.2018.00017.

18 Shafi Goldwasser, Yael Tauman Kalai, and Guy N. Rothblum. Delegating computation: Interactive proofs for muggles. *Journal of the ACM*, 62(4):27:1–27:64, 2015. doi:10.1145/2699436.

19 Aniket Kate, Gregory M. Zaverucha, and Ian Goldberg. Constant-size commitments to polynomials and their applications. In *Proceedings of the 16th International Conference on the Theory and Application of Cryptology and Information Security*, ASIACRYPT '10, pages 177–194, 2010. doi:10.1007/978-3-642-17373-8_11.

20 Carsten Lund, Lance Fortnow, Howard J. Karloff, and Noam Nisan. Algebraic methods for interactive proof systems. *Journal of the ACM*, 39(4):859–868, 1992. doi:10.1145/146585.146605.

21 Peter L Montgomery. Speeding the pollard and elliptic curve methods of factorization. *Mathematics of computation*, 48(177):243–264, 1987.

22 Ngoc Khanh Nguyen and Gregor Seiler. Greyhound: Fast polynomial commitments from lattices. In *Proceedings of the 44th Annual International Cryptology Conference*, CRYPTO '24, pages 243–275, 2024. doi:10.1007/978-3-031-68403-6_8.

23 Charalampos Papamanthou, Elaine Shi, and Roberto Tamassia. Signatures of correct computation. In *Proceedings of the 10th Theory of Cryptography Conference*, TCC '13, pages 222–242, 2013. doi:10.1007/978-3-642-36594-2_13.

24 Omer Reingold, Ron Rothblum, and Guy Rothblum. Constant-round interactive proofs for delegating computation. In *Proceedings of the 48th ACM Symposium on the Theory of Computing*, STOC '16, pages 49–62, 2016. doi:10.1145/2897518.2897652.

25 Noga Ron-Zewi and Ron Rothblum. Local proofs approaching the witness length. In *Proceedings of the 61st Annual IEEE Symposium on Foundations of Computer Science*, FOCS '20, pages 846–857, 2020.

26 Noga Ron-Zewi and Ron D. Rothblum. Proving as fast as computing: Succinct arguments with constant prover overhead. In *Proceedings of the 54th ACM Symposium on the Theory of Computing*, STOC '22, pages 1353–1363, 2022. doi:10.1145/3519935.3519956.

27 Adi Shamir. IP = PSPACE. *Journal of the ACM*, 39(4):869–877, 1992. doi:10.1145/146585.146609.

28 Lloyd N. Trefethen. Barycentric lagrange interpolation. *SIAM Review*, 46(3):501–517, 2004. doi:10.1137/S0036144502417715.

29 Joachim von zur Gathen and Jürgen Gerhard. *Modern Computer Algebra (3. ed.)*. Cambridge University Press, 2013.

Fiat-Shamir for Bounded-Depth Adversaries

Liyan Chen
MIT, Cambridge, MA, USA

Yilei Chen
Tsinghua University, Beijing, China
Shanghai Qi Zhi Institute, China

Zikuan Huang
Shanghai Qi Zhi Institute, China

Nuozhou Sun
Carnegie Mellon University, Pittsburgh, PA, USA

Tianqi Yang
Columbia University, New York, NY, USA

Yiding Zhang
Boston University, MA, USA

Abstract

We study how to construct hash functions that can securely instantiate the Fiat-Shamir transformation against *bounded-depth* adversaries. The motivation is twofold. First, given the recent fruitful line of research of constructing cryptographic primitives against bounded-depth adversaries under *worst-case complexity assumptions*, and the rich applications of Fiat-Shamir, instantiating Fiat-Shamir hash functions against bounded-depth adversaries under worst-case complexity assumptions might lead to further applications (such as SNARG for P, showing the cryptographic hardness of PPAD, etc.) against bounded-depth adversaries. Second, we wonder whether it is possible to overcome the impossibility results of constructing Fiat-Shamir for arguments [Goldwasser, Kalai, FOCS '03] in the setting where the depth of the adversary is bounded, given that the known impossibility results (against p.p.t. adversaries) are contrived.

Our main results give new insights for Fiat-Shamir against bounded-depth adversaries in both the positive and negative directions. On the positive side, for Fiat-Shamir for proofs with certain properties, we show that weak worst-case assumptions are enough for constructing explicit hash functions that give $\mathsf{AC}^0[2]$-soundness. In particular, we construct an $\mathsf{AC}^0[2]$-computable correlation-intractable hash family for constant-degree polynomials against $\mathsf{AC}^0[2]$ adversaries, assuming $\oplus\mathsf{L}/\mathsf{poly} \not\subseteq \widetilde{\mathsf{Sum}}_{n^{-c}} \circ \mathsf{AC}^0[2]$ for some $c > 0$. This is incomparable to all currently-known constructions, which are typically useful for larger classes and against stronger adversaries, but based on arguably stronger assumptions. Our construction is inspired by the Fiat-Shamir hash function by Peikert and Shiehian [CRYPTO '19] and the fully-homomorphic encryption scheme against bounded-depth adversaries by Wang and Pan [EUROCRYPT '22].

On the negative side, we show Fiat-Shamir for arguments is still impossible to achieve against bounded-depth adversaries. In particular,

- Assuming the existence of $\mathsf{AC}^0[2]$-computable CRHF against p.p.t. adversaries, for every poly-size hash function, there is a (p.p.t.-sound) interactive argument that is not $\mathsf{AC}^0[2]$-sound after applying Fiat-Shamir with this hash function.
- Assuming the existence of $\mathsf{AC}^0[2]$-computable CRHF against $\mathsf{AC}^0[2]$ adversaries, there is an $\mathsf{AC}^0[2]$-sound interactive argument such that for every hash function computable by $\mathsf{AC}^0[2]$ circuits, the argument does not preserve $\mathsf{AC}^0[2]$-soundness when applying Fiat-Shamir with this hash function. This is a low-depth variant of Goldwasser and Kalai.

2012 ACM Subject Classification Security and privacy → Hash functions and message authentication codes

Keywords and phrases Fiat-Shamir, Correlation Intractability

7th Conference on Information-Theoretic Cryptography (ITC 2026).
Editor: Yevgeniy Dodis; Article No. 4; pp. 4:1–4:22

Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

Digital Object Identifier 10.4230/LIPIcs.ITC.2026.4

Related Version *Full Version*: https://eprint.iacr.org/2024/256 [13]

Funding *Yilei Chen*: Research supported by Tsinghua University startup funding, and Shanghai Qi Zhi Institute Innovation Program.
Tianqi Yang: Research supported in part by NSF Grant CCF-2106429, CCF-2211238, CCF-2212136, CCF-2312224, an Amazon Research Award, and Google CyberNYC.

1 Introduction

The Fiat-Shamir transform [23] is a generic method for converting public-coin interactive protocols into non-interactive ones while preserving the original protocol's functionality. First proposed as a practical method to compile identification schemes into digital signatures, the Fiat-Shamir transform was then realized to be an extremely general technique for minimizing interaction in any public-coin protocol. The basic idea of Fiat-Shamir is to replace the verifier's message in each round (which consists of random coin tosses) with a deterministic hash of the protocol transcript so far. As a typical example, consider a three-round public-coin interactive proof/argument system (P, V) for a language L. When proving a statement $x \in L$, the prover P first sends a message α, the verifier V responds with random coins β, and then P sends the last message γ. To convert the interactive protocol (P, V) into a non-interactive protocol (P^{FS}, V^{FS}), the Fiat-Shamir transform uses a hash function h (randomly chosen from a hash family $\mathcal{H}$) and removes the verifier's coin tosses β by setting $\beta = h(x\|\alpha)$[1]. Then P^{FS} can directly compute β and send $(\alpha, \beta = h(x\|\alpha), \gamma)$ in one shot without any interaction (actually β can be omitted since it is clear from x, α and the public hash function h). See Figure 1 below for a visual explanation.

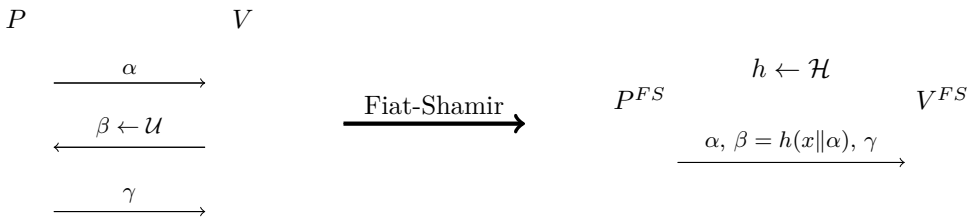


Figure 1 Instantiating Fiat-Shamir on (P, V) for proving $x \in L$.

1.1 The soundness of Fiat-Shamir

The central problem of Fiat-Shamir is its soundness – justifying whether and when the Fiat-Shamir transform results in a sound non-interactive protocol has long been a focus of research in cryptography. It was shown by Pointcheval and Stern [41] that the soundness of Fiat-Shamir holds in the *random oracle model* (ROM), i.e., the soundness of any constant-round public-coin protocol can be preserved if we model the Fiat-Shamir hash function as a random oracle. However, it is more desirable to prove the soundness of Fiat-Shamir

[1] "‖" means the concatenation of two strings. Here we use $h(x\|\alpha)$ but not $h(x, \alpha)$ to emphasize that h takes as input the entire transcript till now (consisting of x and α) instead of taking x and α separately as two inputs.

transformation in the *plain model*: the Fiat-Shamir hash should be instantiated by an explicit hash function in contrast to an idealized one, and the security in the random oracle model cannot provide any guarantee here.

In the plain model, the difference between *proofs* and *arguments* becomes crucial. By using proofs, we mean protocols that are sound against *any* cheating provers (can be computationally unbounded), while by using arguments, we mean protocols that are sound against only computationally bounded (probabilistic polynomial-time) cheating provers. We summarize the main positive results here.

- **Fiat-Shamir for proofs.** If a hash function is correlation intractable (CI) then it can securely instantiate Fiat-Shamir for proofs [11]. In the past few years there has been a long line of work on correlation intractable (CI) hash functions [10, 34, 11, 27, 9, 40, 7, 28, 31, 38, 15, 19] showing that we can securely instantiate Fiat-Shamir on interactive proofs of which the "bad relation" (which is roughly the messages (α, β) that may allow the prover to cheat) meets some special constraints. CI hash functions have already been used for applying Fiat-Shamir to some practical interactive proofs, e.g., the sumcheck protocol and the GKR protocol [32], and also for constructing non-interactive cryptographic primitives from standard cryptographic assumptions, e.g., non-interactive zero-knowledge (NIZK) [40, 7, 28, 31, 19] and succinct non-interactive arguments (SNARGs) [32, 17, 18, 29, 33, 15].
- **Fiat-Shamir for arguments.** In the context of arguments, however, the soundness of Fiat-Shamir is still poorly understood. In some applications of Fiat-Shamir, we do know how to instantiate Fiat-Shamir securely on some special arguments using just CI hash functions (see, e.g., the SNARG constructions from [35, 17, 18, 29, 15]). However, the security mainly comes from the statistical property in their soundness such that these arguments can somehow be regarded as proofs – actually we are still using the framework for proofs. Outside the scope of "proof-like" arguments, there are only few positive results about some specific protocols, e.g., Fiat-Shamir for Schnorr's or Lyubashevsky's identification schemes [14].

As we can see, positive results showing the soundness of Fiat-Shamir in the plain model concentrate almost all on proofs, while the soundness of Fiat-Shamir for arguments remains largely open (instantiating Fiat-Shamir soundly on proofs is intuitively easier since proofs have stronger security guarantees than arguments). Furthermore, there have been impossibility results or strong negative indications on the soundness of Fiat-Shamir for arguments in the plain model. Goldwasser and Kalai [26] constructed a set of three (contrived) arguments such that Fiat-Shamir using *any* explicit hash function must be insecure on at least one of them. In addition, Dwork et al. [21] showed that (constant-round, public-coin) zero-knowledge arguments can also serve as counter-examples for Fiat-Shamir[2], and constructions of such protocols are known based on various cryptographic assumptions (see, e.g., [3, 6, 37]).

1.2 Fiat-Shamir and cryptography against bounded-depth adversaries

In this work, we study the soundness of Fiat-Shamir in a new setting where the adversaries' *parallel time* is further limited. In contrast to the standard security notion in cryptography that the adversaries run in probabilistic polynomial time (p.p.t.) with no further limitation, we consider a relaxed version of security that the adversaries are from *bounded-depth* (and

[2] The intuition of [21] is: we can construct a cheating verifier that sends messages according to the Fiat-Shamir hash function, and the zero-knowledge property guarantees that a simulator can efficiently simulate the view of such a cheating verifier. If the protocol is sound after Fiat-Shamir, the simulator can then be used to decide the language. Therefore, Fiat-Shamir can only be sound when the zero-knowledge protocol proves a language in BPP.

thus bounded parallel time) circuit classes like $\mathsf{AC}^0[2]$ (poly-size constant-depth circuits with NOT gates and unbounded fan-in OR, AND, and MOD_2 gates) and TC^0 (poly-size constant-depth circuits with NOT gates and unbounded fan-in OR, AND, and threshold (THR) gates, where a threshold gate returns 1 if at least half of its inputs are 1, and 0 otherwise). If we can get positive results for Fiat-Shamir against bounded-depth adversaries, it will be very exciting: given the rich applications of (or inspired by) Fiat-Shamir in recent years (such as SNARGs for P, showing the cryptographic hardness of PPAD [16], etc.), instantiating Fiat-Shamir hash functions against bounded-depth adversaries might lead to further applications against bounded-depth adversaries. Furthermore, given the strong negative indications and the extreme lack of positive results on Fiat-Shamir for arguments against p.p.t adversaries, we would like to see if it is possible to bypass the known impossibility results in the bounded-depth setting.

Recently there has been a fruitful sequence of work on constructing cryptographic primitives with security against only bounded-depth adversaries[3], including one-way functions, pseudorandom generators, collision-resistant hash functions [20], one-way permutations [22], fully homomorphic encryption schemes [8, 42], and non-interactive zero-knowledge [42]. Compared to their analogs with the standard p.p.t. security, these primitives are indeed less secure but can rely only on worst-case complexity assumptions or even be unconditional (see, e.g., the construction of CRHF against AC^0 in [20], where AC^0 is the class of poly-size constant-depth circuits with NOT gates and unbounded fan-in OR and AND gates). Here we continue this line of work with a specific focus on Fiat-Shamir in the same setting: the soundness is rather weak (but still non-trivial) against circuit classes as low as $\mathsf{AC}^0[2]$, while the underlying assumption is likewise weak. The choice of bounded-depth circuit classes is also motivated by the developments in circuit complexity, where strong lower bounds for bounded-depth classes are known. Therefore, it is very likely that we can translate these lower bounds into good hash constructions that are only useful for Fiat-Shamir against the same bounded-depth classes.

Also note that the lowest circuit class we care about is $\mathsf{AC}^0[2]$, which stands for poly-size constant-depth circuits with NOT gates and OR, AND, MOD_2 gates with unbounded fan-in. We choose $\mathsf{AC}^0[2]$ because our results (as shown in the next section) can indeed work for classes as low as $\mathsf{AC}^0[2]$, and $\mathsf{AC}^0[2]$ is arguably the smallest circuit class that makes Fiat-Shamir reasonable. In cryptography, almost all hash functions (and other cryptographic primitives) use some kind of linearity or algebra over finite fields, and constructions without the use of parity or modulo gates are rare. To the best of our knowledge, there are only two exceptions about classes lower than $\mathsf{AC}^0[2]$. The first one is the cryptography in AC^0 by Degwekar et al.[20], but their construction is still inherently using parity, except they use some clever tricks to bound the number of bits in the parity by $\mathsf{polylog}$. The other one is the cryptography in NC^0 by Applebaum et al. [2] (where NC^0 is the same as AC^0 except that OR and AND gates have only bounded fan-in), but their construction is not robust enough, e.g., their PRG can only have additive stretch.

1.3 Our results

Our results include two independent parts that make progress on both the positive side and the negative side. On the positive side, we give new constructions of Fiat-Shamir hash functions for proofs in the bounded-depth setting. Inspired by the correlation intractable hash

[3] This research topic is also referred to as "fine-grained cryptography" like in [20]. We do not use the notion "fine-grained" because in complexity theory it usually refers to unrobust classes like $\mathsf{DTIME}[n^2]$. We use the name "cryptography against bounded-depth adversaries" to avoid any possible ambiguity.

by Peikert and Shiehian [40] and the FHE with bounded-depth security by Wang and Pan [42], we construct a new family of correlation intractable hash functions against bounded-depth adversaries. It allows us to generally instantiate Fiat-Shamir, at least on a class of interactive proofs, to obtain non-interactive cryptographic primitives in the bounded-depth setting. On the negative side, unfortunately, we observe that our positive results on proofs are almost the best we can do in the bounded-depth setting. In particular, we show that the known impossibility results for Fiat-Shamir for arguments like Goldwasser and Kalai [26] can be extended to the bounded-depth setting.

1.3.1 Positive results

We first present the positive results for Fiat-Shamir for *proofs* against bounded-depth cheating provers. In particular, we show new constructions of correlation-intractable (CI) hash functions for relations representable by constant-degree polynomials that are secure against bounded-depth adversaries.

Our construction is inspired by the CI hash function constructed by Peikert and Shiehian [40], which uses the FHE scheme of Gentry, Sahai, Waters [24] (henceforth GSW-FHE) as a building block. Our main observation is that the FHE scheme against low-depth adversaries constructed by Wang and Pan [42] bears a striking similarity with GSW-FHE. The similarity allows us to replace GSW-FHE with WP-FHE to obtain a CI hash function against low-depth adversaries assuming worst-case complexity assumptions. Our main positive result is:

▶ **Theorem 1** (Informally stated, see Theorem 28)**.** *Assuming* $\mathsf{NC}^1 \neq \oplus\mathsf{L}/\mathsf{poly}$*, there exists a CI hash function family for constant-degree polynomials against* NC^1 *that is computable in* $\mathsf{AC}^0[2]$.

The circuit class NC^1 (which stands for poly-size $O(\log n)$-depth circuits with bounded fan-in NOT, OR, and AND gates) is just a typical example and can actually be replaced with lower circuit classes. See Section 3.3 for more details. The worst-case complexity assumption is a standard assumption in the area of cryptography against bounded-depth adversaries: previous constructions of, e.g., one-way functions, pseudorandom generators, collision-resistant hash functions [20] are all based on the same assumption.

1.3.2 Negative results

On the negative side, we get two (incomparable) impossibility results in the bounded-depth setting. The first one shows the non-existence of a *universal* Fiat-Shamir hash against bounded-depth adversaries; while the second one extends the work by Goldwasser and Kalai [26] showing the existence of a protocol that makes every Fiat-Shamir hash fail even in the bounded-depth setting.

$\mathcal{C}$-soundness and $\mathcal{C}$-arguments. For simplicity, we say a protocol has $\mathcal{C}$-soundness if it is sound against adversaries computable in the complexity class $\mathcal{C}$. Also, we use the notation $\mathcal{C}$-arguments for protocols with $\mathcal{C}$-soundness. Typical choices of $\mathcal{C}$ include bounded-depth circuit classes $\mathsf{AC}^0[2]$, TC^0, or NC^1. When just saying an argument, we refer to a P/poly-argument.

We first (informally) define what a universal Fiat-Shamir hash is in the bounded-depth setting.

▶ **Definition 2** (Universal Fiat-Shamir hash, informal). *For circuit classes $\mathcal{D} \subseteq \mathcal{C}$, we say a hash function family $\mathcal{H}$ is a universal Fiat-Shamir hash from $\mathcal{C}$ to $\mathcal{D}$ if Fiat-Shamir using $\mathcal{H}$ always results in a sound non-interactive $\mathcal{D}$-argument starting from any (3-round public-coin) interactive $\mathcal{C}$-argument that matches the input and output lengths of $\mathcal{H}$ (i.e., the protocol's instance + first message length should be $\mathcal{H}$'s input length, and the protocol's second message length should be $\mathcal{H}$'s output length).*

Note that Definition 2 is already enough to capture a large class of hash functions, which is a fixed hash function family as long as the input and output lengths are determined. As an example, the CI hash function in [11] fits well into this format. However, there also exist hash functions outside the scope of Definition 2. For example, the CI hash from [40] does not fit into this definition[4] because its key length depends on some specific property of the interactive proofs we want to collapse, which cannot be fixed in advance given only the input and output lengths (more precisely, its key length depends on how fast we can compute the "bad function" of the protocol). Based on this definition, we give our first negative result:

▶ **Theorem 3** (Informally stated, see Theorem 5.2 of [13]). *Assuming the existence of a CRHF family that is computable in $\mathcal{D}$ and collision-resistant against $\mathcal{C}$ with any polynomial shrinkage, there does not exist a universal Fiat-Shamir hash from $\mathcal{C}$ to $\mathcal{D}$ with any polynomial input and output lengths.*

Here we further justify our assumption of a low-complexity CRHF. One can examine that the CRHF construction $f_A(x) = Ax \bmod q$ based on the hardness of short integer solution (SIS) [1] is actually computable in TC^0 (and has polynomial shrinkage). Also, the discrete-log (DLOG)-based CRHF construction $f_{g,h}(x_1, x_2) = g^{x_1} \cdot h^{x_2}$ can also be computed by a TC^0 circuit using some standard pre-processing techniques. Therefore, low-complexity CRHFs are known from these well-studied cryptographic assumptions, which leads to the following corollary:

▶ **Corollary 4.** *Assuming SIS/DLOG, there does not exist a universal Fiat-Shamir hash that can soundly transform any interactive argument into a non-interactive TC^0-argument.*

Actually, similar results hold also for circuit classes lower than TC^0. We present a candidate $\mathsf{AC}^0[2]$-computable poly-shrinkage CRHF based on a variant of SIS, indicating the impossibility of a universal Fiat-Shamir hash achieving even only $\mathsf{AC}^0[2]$-soundness. See Appendix B of the full version [13] for more discussions on the $\mathsf{AC}^0[2]$-computable CRHF.

Even more generally, we show that the result of Goldwasser and Kalai [26] – albeit looks complicated and requires super-constant depth at a first glance – can be generalized to the bounded-depth setting with adversaries from circuit classes as low as $\mathsf{AC}^0[2]$.

▶ **Theorem 5** (Informally stated, see Theorem 6.1 of [13]). *For any circuit class $\mathcal{C} \supseteq \mathsf{AC}^0[2]$, assuming the existence of a CRHF family with arbitrary polynomial shrinkage that is computable in $\mathcal{C}$ and collision-resistant against $\mathcal{C}$, there exists a 3-round public-coin $\mathcal{C}$-argument such that instantiating Fiat-Shamir using any hash function computable in $\mathcal{C}$ does not result in a sound non-interactive $\mathcal{C}$-argument.*

Compared to the previous result in Theorem 3 showing impossibility even when we require only some weaker soundness on the Fiat-Shamir-collapsed protocol, Theorem 5 shows only the impossibility of Fiat-Shamir hash functions preserving the same level of soundness

[4] Here we are only talking about the formats of these hash functions (e.g., how the key length is chosen) instead of their specific functionalities – of course, we do not have such a universal Fiat-Shamir hash with security against p.p.t. due to [26].

due to some technical issues. However, instead of saying every FS hash must fail on some tailored protocol, Theorem 5 is stronger in the sense that it shows the existence of some fixed protocol that makes every Fiat-Shamir hash fail. The Fiat-Shamir hash is allowed to depend arbitrarily on the fixed interactive protocols (e.g., the third message's length $|\gamma|$ or the prover/verifier running time, which are not allowed in Definition 2 and Theorem 3).

Also note that the CRHF in our assumption can be instantiated (at least for $\mathsf{AC}^0[2] \subseteq \mathcal{C} \subseteq \mathsf{NC}^1$) by the CRHF from [20] that is computable in $\mathsf{AC}^0[2]$ and secure against NC^1 based on the worst-case complexity assumption $\mathsf{NC}^1 \neq \oplus\mathsf{L}/\mathsf{poly}$.

1.4 Related works

Aside from the results like Goldwasser and Kalai [26] showing that there is an interactive argument such that no explicit hash function can collapse the protocol while preserving soundness, there have also been negative results for some restricted classes of protocols, in particular, for protocols related to SNARGs like Kilian's protocol [36]. Kilian's protocol is a 4-message succinct argument for any NP language. The idea of collapsing Kilian's protocol by Fiat-Shamir comes from Micali's CS proofs [39], which are now usually referred to as succinct non-interactive arguments (SNARGs). While Micali's construction applies Fiat-Shamir in the random oracle model, there have been negative results on Fiat-Shamir for Kilian's protocol in the plain model. Gentry and Wichs [25] showed a substantial barrier to constructing SNARGs in general (not limited to collapsing Kilian's protocol), and the problem of securely instantiating Fiat-Shamir on Kilian's protocol also suffers from this barrier. More recently, Bartusek et al. [5] showed that Fiat-Shamir with any explicit hash function can never be secure on some contrived instantiation of Kilian's protocol.

CI hash functions for constant degree polynomial relations against all ppt adversaries exist under various assumptions (for example, in [9, 7]), but none of them is computable in $\mathsf{AC}^0[2]$. After the initial publication of our work, Dao et al. [19] construct a CI hash function for degree-d polynomial relations against all ppt adversaries, assuming the hardness of solving random degree-d equations. When d is a constant, their hash function is also computable in $\mathsf{AC}^0[2]$. Let us remark that the assumptions used in their work and our work are technically incomparable, but we think our assumption is arguably weaker since it is a worst-case complexity assumption.

Organization. The rest of this paper is organized as follows. We first give an overview of the intuition and techniques underlying our results in Section 2. Then in Section 3, we present details of the main positive result, which is the construction of CI hash functions. Details of the negative results are deferred to the full version (see Sections 5 and 6 of [13]).

2 Technical Overview

2.1 Positive results

Our positive result is inspired by the CI hash function in [40], which crucially relies on the FHE scheme of Gentry, Sahai, Waters [24] (henceforth GSW-FHE). Our main observation is that the FHE scheme against low-depth adversaries constructed by Wang and Pan [42] (henceforth WP-FHE) bears a striking similarity with GSW-FHE, therefore it can be used as in [40] to obtain a CI hash function against low-depth adversaries.

Let us start by explaining the intuition of [40]. They consider sparse relations defined by efficient functions i.e. $\{(x, f(x))\}$ for some efficiently computable function f. Designing a hash function $h(k, x)$ that is correlation intractable for a single function-defined relation

is trivial: simply define $h(k, x) = f(x) + 1$. The construction [40] utilized the idea: for any f, a random hash key will look indistinguishable from a hash key that is specifically designed to be correlation intractable with respect to the relation defined by f, i.e. the construction achieves the notion of "somewhere correlation intractability". This idea can be implemented by using any fully homomorphic encryption scheme, i.e., the hash key is the homomorphic encryption of the circuit that computes a fixed function g_0 (hence for any f, it looks indistinguishable from the encryption of the circuit that computes f), and the hash function homomorphically computes $g_0(x) + 1$ using a universal circuit $\mathcal{U}(x, \cdot)$ s.t. $\mathcal{U}(x, f) = f(x)$. The result of homomorphic evaluation would be a ciphertext of $g_0(x) + 1$, so it remains as the last step to decrypt this ciphertext. One idea, as shown in [9], is to put the encryption of the FHE decryption function inside the public key.

$$h(k, x) = \mathsf{Eval}(\mathsf{pk}, \mathcal{U}(x, \cdot), \mathsf{ct}), \text{ where } k = (\mathsf{pk}, \mathsf{ct}), \mathsf{ct} = \mathsf{Enc}(\mathsf{pk}, g_0) \approx \mathsf{Enc}(\mathsf{pk}, \mathsf{Dec}(\mathsf{sk}, f(\cdot)) \oplus 1).$$

Then for any function f, if k is sampled according to $(\mathsf{pk}, \mathsf{Enc}(\mathsf{pk}, \mathsf{Dec}(\mathsf{sk}, f(\cdot)) \oplus 1))$, we have

$$f(x) = h(k, x) \implies f(x) = \mathsf{Enc}(\mathsf{pk}, \mathsf{Dec}(\mathsf{sk}, f(x)) \oplus 1) \implies \mathsf{Dec}(\mathsf{sk}, f(x)) = \mathsf{Dec}(\mathsf{sk}, f(x)) \oplus 1.$$

However, in this approach, in order to show the hash key k does not leak FHE secret key, we need to assume circular security. [40] bypasses this "decryption problem" based on an observation about the ciphertext of GSW-FHE. Namely, for any vector $(y_i)_{i \in [n]}$, if we have the ciphertext of each y_i, then we can compute $Ar + y$ where A is the FHE public key and r is derived from the randomness used to encrypt each y_i. Coarsely, if $y = f(x)$ as the result of homomorphic evaluation, then from $f(x) = Ar + f(x)$ we find a solution for $Ar = 0$. [40] manages to reduce breaking SIS assumption to breaking correlation intractability of their construction.

Our construction of CI against low-depth adversary can be explained in a single sentence: replace GSW-FHE in the construction [40] by WP-FHE. We claim that these two FHE schemes, though under different assumptions and are against different adversaries, are similar to each other in a precise sense: WP-FHE can be regarded as the mod-2, noise-free version of GSW-FHE. We provide the following table to sustain this intuition.

Table 1 Comparison between GSW-FHE and WP-FHE.

	GSW-FHE	WP-FHE
$(\mathsf{pk}, \mathsf{sk})$	$k^T A = e \approx 0$	$k^T A = 0$
Enc	$AR + mG$	$AR + mI$
Dec	$k^T \mathsf{ct} \cdot G^{-1}((0, \ldots, 0, q/2)^T)$	$k^T \mathsf{ct} \cdot (0, \ldots, 0, 1)^T$
$\mathsf{Eval}^{\mathrm{add}}$	$\mathsf{ct}_0 + \mathsf{ct}_1 - 2\mathsf{ct}_0 \cdot G^{-1}(\mathsf{ct}_1)$	$\mathsf{ct}_0 + \mathsf{ct}_1$
$\mathsf{Eval}^{\mathrm{mul}}$	$\mathsf{ct}_0 \cdot G^{-1}(\mathsf{ct}_1)$	$\mathsf{ct}_0 \cdot \mathsf{ct}_1$

The replacement allows the hash function to be computable in AC^0[2]. Also, we use a result in [22] to show that the security of our construction follows from the worst-case complexity assumption.

2.2 Negative results

Our main impossibility result comes from extending the impossibility result by Goldwasser and Kalai [26] to the bounded-depth settings. For readers familiar with [26], we first highlight the main technical challenges and techniques behind our extension.

- **Merkle tree hash**: The construction of [26] relies on a Merkle tree hash to succinctly commit to messages that may have an arbitrary polynomial size. In our bounded-depth setting, however, the depth of the Merkle tree is inherently limited to a constant. To handle limitation on depth, we base our construction on the stronger assumption of a low-depth CRHF with polynomial shrinkage – then a constant-depth Merkle tree is enough to handle any polynomial-size message.
- **PCP in $\mathsf{AC}^0[2]$**: [26] also uses probabilistic checkable proofs (PCPs) to guarantee the efficiency of the protocols. Since the prover in our setting is only allowed to do bounded-depth computations, we need a PCP satisfying all the required properties (including efficient oracle-construction, proof of knowledge, and reverse sampling as in [26, 4]) within a bounded-depth circuit class. To meet the efficiency requirements here, we construct such a PCP in $\mathsf{AC}^0[2]$, i.e., a PCP with $\mathsf{AC}^0[2]$-oracle-construction, $\mathsf{AC}^0[2]$-proof-of-knowledge, and $\mathsf{AC}^0[2]$-reverse-sampling (see Appendix C of the full version [13]).

Now we present the intuition of how to make Fiat-Shamir fail and our first impossibility result that directly follows the intuition (see Theorem 3). There is a simple intuition behind all existing contrived impossibility results: the prover in an interactive protocol is never able to predict the verifier's choice of the next message, but Fiat-Shamir makes the verifier's messages "predictable". Consider the standard 3-round public-coin setting where the protocol transcript is (α, β, γ). While sending the first message α, the prover can never predict the next message β in advance – no matter what prediction it makes, the verifier's random choice of β coincides with the prediction only with negligible probability. However, things become totally different when we collapse the protocol by Fiat-Shamir: with the help of a randomly chosen hash function $h \leftarrow \mathcal{H}_{FS}$, the prover now directly generates $(\alpha, \beta = h(x\|\alpha), \gamma)$ in one shot without the verifier's random choice. In other words, the next "verifier's message" $\beta = h(x\|\alpha)$ is just a fixed value after generating α, and of course, the malicious prover can predict β.

Based on the above observation, we can design an interactive protocol that simply asks the prover to predict the next message: in the first round, the prover tries to predict β by submitting the description of a function $\alpha = \hat{C}$ (we use C to denote the circuit of the function and $\hat{C}$ to denote the encoding of C), which means his prediction of β is $C(\alpha) = C(\hat{C})$. Then the verifier randomly chooses β and sends it to the prover, ignores the prover's third message, and finally accepts iff $\beta = C(\alpha)$ holds (which means the prediction is correct). This is an interactive proof for the empty language: the verifier always rejects except the randomly-chosen β happens to be the fixed value $C(\alpha)$, which holds with only negligible probability[5]. After applying Fiat-Shamir, unfortunately, the non-interactive protocol becomes totally broken. Given a Fiat-Shamir hash $h \leftarrow \mathcal{H}_{FS}$, the cheating prover can simply send the description of this hash function (i.e., send $\alpha = \widehat{h(x\|\cdot)}$ with the instance x hardcoded), and then the Fiat-Shamir transformation requires β to be $\beta = h(x\|\alpha)$, which makes the verifier accept.

This counter-example can work in the bounded-depth setting (since what the cheating prover does is simply copying the Fiat-Shamir hash function). However, there is an implicit limitation: when saying $\alpha = \widehat{h(x\|\cdot)}$, $\beta = h(x\|\alpha)$, we implicitly require the Fiat-Shamir

[5] Note that in an interactive protocol for the empty language, there is no *honest* prover since no instance can be proved (except with negligible probability). Or we can say the honest prover is just a machine that sends all-0 strings of some fixed lengths in each round. We clarify here that when describing the prover's behaviors, we actually mean what the prover is supposed to do (and the verifier will check it later). For example, by saying the prover sends $\alpha = \hat{C}$, we actually mean the cheating prover sends an arbitrarily α and the verifier will parse α into the description of a circuit C when deciding to accept/reject.

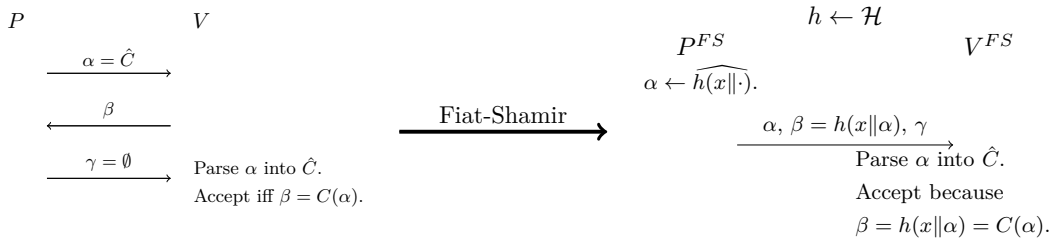


Figure 2 A protocol for the empty language that is not sound after Fiat-Shamir.

hash h to have a short enough description that can fit into its input, i.e. $|h| \leq |\alpha|$. As a result, this counter-example fails to capture a large class of Fiat-Shamir hash functions with the description length longer than the input length. This limitation somehow makes the counter-example worthless, since Fiat-Shamir hash functions usually have larger sizes: as shown in the work of Canetti et al. [12], a hash function cannot even be correlation intractable (i.e., preserving soundness for proofs) if its key length is shorter than the input length.

In order to bypass this limitation, we made some modifications based on the protocol above. Instead of sending the whole circuit description $\hat{C}$ in the first round, the prover only commits to such a description by sending a succinct commitment $\alpha = f(\hat{C})$ (f is the commitment function) – although the description may be long, its commitment can still be short enough to fit into the Fiat-Shamir hash's input. After that, the prover in the third round sends $\gamma = \hat{C}$ in clear to open the commitment, and the verifier decides to accept or reject by checking if $\alpha = f(\hat{C})$ (the opening is correct) and $\beta = C(\alpha)$ (the prediction is correct). See Figure 3 for an illustration of the modified protocol. In this case, a cheating prover after Fiat Shamir transformation can still set C to be the Fiat Shamir hash function $h(x||\cdot)$, and it is clear to see the verifier will accept the proof regardless the input x.

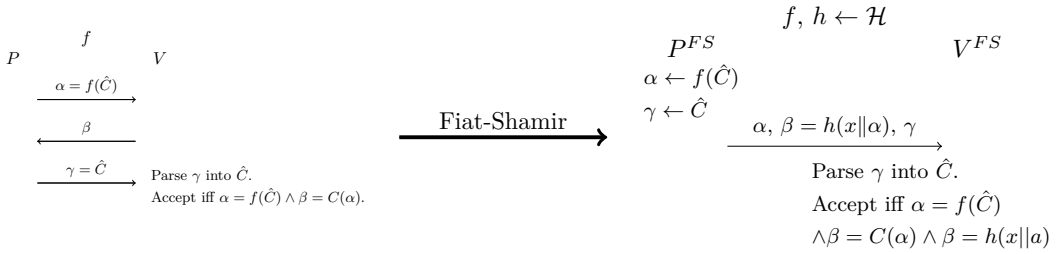


Figure 3 The modified protocol for the empty language.

In order to fit into the bounded-depth setting, we instantiate the succinct commitment f with a CRHF that can be computed in low depth and has enough shrinkage, which leads to our first negative result as shown in Theorem 3. Note that the exact shrinkage of CRHF matters here because we cannot boost the shrinkage arbitrarily in the bounded-depth setting: the circuit depth grows as we stack a CRHF several times to build a Merkle tree. The detailed security proof of this protocol is deferred to the full version (see Section 5 of [13]).

In the aforementioned counterexample, the hash function size remains bounded by a predetermined polynomial. This paper also explores adapting the findings of Goldwasser and Kalai [26] to achieve a more general negative result. Drawing inspiration from their construction, and by carefully addressing circuit depth and the challenges previously men-

tioned, we present three protocols. We demonstrate that at least one of these protocols must serve as a counterexample to the Fiat-Shamir transformation. Our strategy hinges on a win-win argument involving these three protocols: either an adversary can break the security of one protocol after the Fiat-Shamir transformation, thus invalidating it, or no adversary can compromise the security of another. In the latter case, we can leverage a protocol that is easily broken post-transformation but challenging to prove secure as our counterexample. The actual win-win argument is more intricate, involving all three protocols. For a detailed explanation, please refer to Section 6 of [13].

3 Correlation Intractability Against Bounded-Depth Adversaries

In this section we show positive results for Fiat-Shamir for proofs against bounded-depth adversaries. We first observe that the construction of correlation intractable (CI) hash functions of Peikert and Shiehian [40] is computable in TC^0, therefore it has already implied the existence of a CI hash function family computable in TC^0 against adversaries in circuit class $\mathcal{D} \supseteq \mathsf{TC}^0$, assuming SIS is hard against $\mathcal{D}$. We then show our main result: the construction of CI hash functions computable in $\mathsf{AC}^0[2]$ against adversaries in any circuit class $\mathcal{C} \supseteq \mathsf{AC}^0[2]$, assuming $\oplus\mathsf{L/poly} \not\subseteq \widetilde{\mathsf{Sum}}_{n^{-c}} \circ \mathcal{C}$ for some constant $c > 0$, which is a worst-case complexity assumption.

The rest of Section 3 is organized as follows. In Section 3.1 we provide the background needed for this section. In Section 3.2 we present the construction in [40]. The reasons we choose to present the construction in [40] are:

- We emphasize that the construction and hardness reduction can be computed in TC^0;
- The full construction makes the latter construction (which is our main result) more understandable.

Then in Section 3.3 we present our main construction of CI hash function computable in $\mathsf{AC}^0[2]$ against low-depth adversaries.

3.1 Selected Backgrounds

3.1.1 GSW Fully Homomorphic Encryption

▶ **Definition 6** (Lattice Gadgets over $\mathbb{Z}_q$). *For a positive integer modulus q, let $l = \lceil \lg q \rceil$. We define the "gadget" vector as*

$$g^T = (1, 2, \ldots, 2^{l-1}) \in \mathbb{Z}_q^l.$$

For every $u \in \mathbb{Z}_q$, there is an efficiently computable binary vector $g^{-1}(u) \in \{0,1\}^l$ such that $g \cdot g^{-1}(u) \equiv u \pmod q$, i.e., the binary representation of u. We treat $g^{-1} : \mathbb{Z}_q \to \{0,1\}^l$ as a function that computes binary decomposition.

The gadget matrix is defined as $G_n = I_n \otimes g^T$, and sometimes we drop the subscript n. Similarly, we define the function $G^{-1} = (I \otimes g^{-1}) : \mathbb{Z}_q^n \to \{0,1\}^{nl}$, which applies g^{-1} to each coordinate and concatenates the results. For any $u \in \mathbb{Z}_q^n$, we have

$$G \cdot G^{-1}(u) = u.$$

▶ **Construction 7** (GSW-FHE). *GSW-FHE is given by the function family* $\mathsf{GSW\text{-}FHE} = \{\mathsf{GSW\text{-}FHEGen}_n, \mathsf{GSW\text{-}Enc}_n, \mathsf{GSW\text{-}Dec}_n, \mathsf{GSW\text{-}Eval}_n\}_{n\in\mathbb{N}}$*, parameterized by the lattice dimension n, the error distribution χ, the modulus q, and the number of samples $m = n\lceil \log q \rceil$. We assume these parameters are chosen properly so that $LWE_{n-1,2m,q,\chi}$ is believed to be hard.*

- $\mathsf{GSW\text{-}FHEGen}_n$:
  1. *Take* $A' \leftarrow \mathbb{Z}_q^{(n-1)\times 2m}$, $s' \leftarrow \mathbb{Z}_q^{n-1}$, *and* $e \leftarrow \chi^{2m}$.
  2. *Compute* $b = A'^T s' + e \in \mathbb{Z}_q^{2m}$.
  3. *Return*

$$\mathsf{pk} = A = \begin{pmatrix} A' \\ b^T \end{pmatrix} \in \mathbb{Z}_q^{n\times 2m}, \mathsf{sk} = k = \begin{pmatrix} -s' \\ 1 \end{pmatrix} \in \mathbb{Z}_q^n.$$

  (Note that $k^T A = e^T \approx 0$*.)*
- $\mathsf{GSW\text{-}FHEGen}'_n$: *return* $\mathsf{pk} = A \leftarrow \mathbb{Z}_q^{n\times 2m}$ *sampled uniformly at random.*
- $\mathsf{GSW\text{-}Enc}_n(\mathsf{pk} = A, \mu \in \{0,1\})$:
  1. *Take* $R \leftarrow \{0,1\}^{2m\times m} \in \mathbb{Z}_q^{2m\times m}$.
  2. *Return* $\mathsf{ct} = AR + \mu G \in \mathbb{Z}_q^{n\times m}$.

  We allow $\mathsf{GSW\text{-}Enc}_n$ *to take a vector in* $\mathbb{Z}_2^m$ *as the input, instead of a single bit. In this case, the encryption of each dimension of that vector (using independent randomness) is concatenated together by each row. That is, for a length-*m *vector* x, $\mathsf{GSW\text{-}Enc}(\mathsf{pk} = A, x) = AR + x^T \otimes I_n \otimes g^T$, *where* $R \leftarrow \mathbb{Z}_2^{2m\times m^2}$.
- $\mathsf{GSW\text{-}Dec}_n(\mathsf{sk} = k, \mathsf{ct})$:
  1. *Take* $w = (0, 0, \ldots, 0, \lceil q/2 \rceil)^T \in \mathbb{Z}_q^n$.
  2. *Compute the value of* $V = k^T \mathsf{ct} \cdot G^{-1}(w)$, *output the decrypted message as* $\left\lceil \frac{V}{q/2} \right\rfloor$.
- $\mathsf{GSW\text{-}Eval}_n$. *We describe it by giving how to homomorphically compute addition and multiplication in* $\mathbb{Z}_q$.
  - $\mathsf{GSW\text{-}Eval}_n^{add}(\mathsf{pk} = A, (\mathsf{ct}_0, \mathsf{ct}_1))$: *return* $\mathsf{ct}_0 + \mathsf{ct}_1 - 2\mathsf{ct}_0 \cdot G^{-1}(\mathsf{ct}_1) \in \mathbb{Z}_q^{n\times m}$.
  - $\mathsf{GSW\text{-}Eval}_n^{mul}(\mathsf{pk} = A, (\mathsf{ct}_0, \mathsf{ct}_1))$: *return* $\mathsf{ct}_0 \cdot G^{-1}(\mathsf{ct}_1) \in \mathbb{Z}_q^{n\times m}$.

▶ Remark 8. For any constant-degree polynomial function family $\{f_n\}$, the corresponding circuit family computing $\mathsf{GSW\text{-}Eval}_n(\mathsf{pk}, f_n, (\mathsf{ct}_1, \ldots))$ is in TC^0. WP-FHE, which will be defined later, has a stronger property: the circuit family computing homomorphic evaluation of constant-degree polynomial function is in $\mathsf{AC}^0[2]$.

3.1.2 Inert Commitment

Let $A \in \mathbb{Z}_q^{n\times 2m}$ be the public key of GSW-FHE, and let $C_i = AR_i + x_i G$ be the homomorphic encryption of scalar $x_i \in \{0,1\}$, for $i \in [m]$. Then we concatenate all C_i's together, and view it as a commitment (relative to A) to $x^T = (x_1, \cdots, x_m)^T$ under randomness $R = \begin{pmatrix} R_1, & \cdots, & R_m \end{pmatrix}$, i.e.

$$C = \begin{pmatrix} C_1, & \cdots & , C_m \end{pmatrix} = AR + x^T \otimes G = AR + x^T \otimes I_n \otimes g^T = \mathsf{GSW\text{-}Enc}(A, x).$$

Since any matrix $M \in \mathbb{Z}_q^{n\times m}$ can be "vectorized" as an $v_M \in \mathbb{Z}_q^{nm}$, so that $(x^T \otimes I_n) \cdot v_M = Mx$, we have

$$c_M = C \cdot G^{-1}(v_M) = A \underbrace{(R \cdot G^{-1}(v_M))}_{r_M} + (x^T \otimes I_n \otimes g^T) \cdot (I_m \otimes I_n \otimes g^{-t}) v_M = Ar_M + Mx \in \mathbb{Z}_q^n.$$

We view c_M as an "inert commitment" to $Mx \in \mathbb{Z}_q^n$, and we write it as $\mathsf{InertEval}(M, C)$.

▶ **Definition 9** (Inert Commitment). *Given* $C \in \mathbb{Z}_q^{n\times m^2}$ *as the homomorphic encryption of some length-*m *vector, and* $M \in \mathbb{Z}_q^{n\times m}$ *as a matrix, we write*

$$\mathsf{InertEval}(M, C) := c_M = C \cdot G^{-1}(v_M),$$

where $v_M \in \mathbb{Z}_2^{nm}$ *is the vector satisfies* $(x^t \otimes I_n) \cdot v_M = Mx$ *for any length-*m *vector* x*, obtained by reordering the elements of* M*.*

3.1.3 Correlation intractability

▶ **Definition 10.** *We say a relation $\mathcal{R} \subseteq \mathcal{X} \times \mathcal{Y}$ is searchable in a set of functions $\mathcal{F}$ if there exists a function $f : \mathcal{X} \to \mathcal{Y}$ in $\mathcal{F}$ such that if $(x, y) \in \mathcal{R}$ then $y = f(x)$.*

We say a relation class $\mathcal{S} = \{\mathcal{S}_n\}_{n\in\mathbb{N}}$ (i.e. a set of relations for each n) is searchable in a function family $\mathcal{F} = \{\mathcal{F}_n\}_{n\in\mathbb{N}}$ if for each $n \in \mathbb{N}$, for all $\mathcal{R} \in \mathcal{S}_n$, $\mathcal{R}$ is searchable in $\mathcal{F}_n$.

▶ **Definition 11.** *Let $\mathcal{S} = \{\mathcal{S}_n\}$ be a relation class and $\mathcal{C}$ be a circuit class. A hash function family*

$$\mathcal{H} = \left\{ h_n : \{0,1\}^{l_k(n)} \times \{0,1\}^{l_{in}(n)} \to \{0,1\}^{l_{out}(n)} \right\}$$

with a key generation algorithm

$$\mathsf{Gen}_{\mathcal{H}} = \left\{ \mathsf{Gen}_{\mathcal{H},n} : \{0,1\}^* \to \{0,1\}^{l_k(n)} \right\}$$

is correlation intractable for $\mathcal{S}$ against $\mathcal{C}$ if for every circuit family $\mathcal{A} = \{\mathcal{A}_n\} \in \mathcal{C}$, there exists a negligible function $\mathsf{negl}(\cdot)$ such that for every $n \in \mathbb{N}$, for every $\mathcal{R} \in \mathcal{S}_n$,

$$\Pr\left[(x, h_n(k, x)) \in \mathcal{R} \;\middle|\; \begin{array}{l} k \leftarrow \mathsf{Gen}_{\mathcal{H},n} \\ x \leftarrow \mathcal{A}_n(k) \end{array} \right] < \mathsf{negl}(n).$$

We omit the polynomials $l_k(n), l_{in}(n), l_{out}(n)$ when they are clear from the explicit construction.

3.1.4 Cryptographic primitives against bounded-depth circuits

We consider the capability of adversaries as some circuit class that at least contains $\mathsf{AC}^0[2]$ (thus can compute matrix multiplication in $\mathbb{Z}_2$), with the assumption that the same circuit class augmented by an error-tolerant majority gate is not able to compute $\oplus\mathsf{L/poly}$.

▶ **Assumption 12.** *For a certain circuit class $\mathcal{C}$ s.t. $\mathsf{AC}^0[2] \subseteq \mathcal{C}$, there exists a constant $c > 0$ s.t. $\oplus\mathsf{L/poly} \not\subseteq \widetilde{\mathsf{Sum}}_{1/n^c} \circ \mathcal{C}$.*

For example, Assumption 12 is widely believed to be true for $\mathcal{C} \in \{\mathsf{AC}^0[2], \mathsf{TC}^0, \mathsf{NC}^1\}$.

▶ Remark 13. When $\mathcal{C} = \mathsf{NC}^1$, the assumption above is equivalent to $\oplus\mathsf{L/poly} \neq \mathsf{NC}^1$ as in [20].

In the following we introduce two distributions ZeroSamp and OneSamp. By using the result from randomized encoding [30, 2], any algorithm that distinguishes these two distributions can be used to compute functions in $\oplus\mathsf{L/poly}$. Therefore, from the hardness of $\oplus\mathsf{L/poly}$ (as in Assumption 12), we can show that it is hard to distinguish ZeroSamp from OneSamp. Please see [20] for more detailed discussion.

▶ **Definition 14** (ZeroSamp, OneSamp)**.** *Let n be the security parameter. For $r^{(1)} \in \mathbb{Z}_2^{n(n-1)/2}$, and $r^{(2)} \in \mathbb{Z}_2^{n-1}$, denote*

$$L(r^{(1)}) = \begin{pmatrix} 1 & r_1^{(1)} & r_2^{(1)} & \cdots & r_{n-1}^{(1)} \\ & 1 & r_n^{(1)} & \cdots & r_{2n-3}^{(1)} \\ & & \ddots & \ddots & \vdots \\ & & & 1 & r_{n(n-1)/2}^{(1)} \\ & & & & 1 \end{pmatrix}, R(r^{(2)}) = \begin{pmatrix} 1 & 0 & 0 & 0 & r_1^{(2)} \\ & 1 & 0 & 0 & r_2^{(2)} \\ & & \ddots & \vdots & \vdots \\ & & & 1 & r_{n-1}^{(2)} \\ & & & & 1 \end{pmatrix}.$$

Also, let M_0^n, M_1^n be the following $n \times n$ matrices:

$$M_0^n = \begin{pmatrix} \boldsymbol{0}^T & 0 \\ I_{n-1} & \boldsymbol{0} \end{pmatrix}, M_1^n = \begin{pmatrix} \boldsymbol{0}^T & 1 \\ I_{n-1} & \boldsymbol{0} \end{pmatrix} \quad (I_{n-1} \text{ is the identity matrix of rank } n-1).$$

Then we define two sampling procedures:

- $\mathsf{ZeroSamp}(n)$*: take* $r^{(1)} \leftarrow \{0,1\}^*, r^{(2)} \leftarrow \{0,1\}^*$ *and output* $L(r^{(1)})M_0^n R(r^{(2)})$.
- $\mathsf{OneSamp}(n)$*: take* $r^{(1)} \leftarrow \{0,1\}^*, r^{(2)} \leftarrow \{0,1\}^*$ *and output* $L(r^{(1)})M_1^n R(r^{(2)})$.

Note that the output of $\mathsf{ZeroSamp}(n)$ *always has rank* $n-1$*, and the output of* $\mathsf{OneSamp}(n)$ *always has rank* n*. What's more, the random bits used by* $A \leftarrow \mathsf{ZeroSamp}(n)$ *also gives the kernel of* A*, and therefore we sometimes use* $A, k \leftarrow \mathsf{ZeroSamp}(n)$ *where* k *s.t.* $Ak = 0$ *is additionally given by* $\mathsf{ZeroSamp}(n)$.

If the security parameter n *is clear from context, we may drop it and use the notation* ZeroSamp, OneSamp.

Previous literature discovered that NC^1 circuits cannot distinguish ZeroSamp and OneSamp if $\oplus\mathsf{L/poly} \neq \mathsf{NC}^1$. We extend this idea from NC^1 to other circuit classes that contain $\mathsf{AC}^0[2]$, as stated in the following lemma.

▶ **Lemma 15** (See Lemma 4.3 of [20] for $\mathcal{C} = \mathsf{NC}^1$)**.** *For any circuit class* $\mathcal{C} \supseteq \mathsf{AC}^0[2]$*, if Assumption 12 holds for* $\mathcal{C}$*, there is a negligible function* $\mathsf{negl}(\cdot)$ *such that for any family* $\mathcal{F} = \{f_n\}$ *in* $\mathcal{C}$*, for an infinite number of values of* n*,*

$$|\Pr[f_n(M) = 1 \mid M \leftarrow \mathsf{ZeroSamp}(n)] - \Pr[f_n(M) = 1 \mid M \leftarrow \mathsf{OneSamp}(n)]| < \mathsf{negl}(n).$$

▶ Remark 16. As Remark 3.1 of [20] points out, from worst-case complexity assumptions, we can only achieve infinitely-often primitives, as if we assume $\mathcal{D} \nsubseteq \mathcal{C}$ where $\mathcal{C}, \mathcal{D}$ are circuit classes, we only mean that there exists a function family $\{f_n\} \in \mathcal{D}$, such that for any function family $\{g_n\} \in \mathcal{C}$, $f_n \neq g_n$ for infinitely many n.

However, as long as we strengthen the assumption from the previous one to that there exists $\{f_n\} \in \mathcal{D}$, for all $\{g_n\} \in \mathcal{C}$, $\exists n_0$ such that $f_n \neq g_n$ for all $n > n_0$, we can achieve the conventional almost-everywhere security. We state this observation below. All our constructions assuming 12 automatically achieve almost-everywhere security using the strengthened assumption, and we only state theorems in the infinitely-often secure version for brevity.

▶ **Assumption 17** (Strengthened assumption)**.** *For a certain circuit class* $\mathcal{C}$ *s.t.* $\widetilde{\mathsf{AC}^0}[2] \subseteq \mathcal{C}$*, there exists* $c > 0$ *s.t.* $\exists \{f_n\} \in \oplus\mathsf{L/poly}$ *such that for any function family* $\{g_n\} \in \widetilde{\mathsf{Sum}}_{1/n^c} \circ \mathcal{C}$*,* $\exists n_0$ *such that* $f_n \neq g_n$ *for all* $n > n_0$*.*

▶ **Lemma 18.** *For any circuit class* $\mathcal{C} \supseteq \mathsf{AC}^0[2]$*, if Assumption 17 holds for* $\mathcal{C}$*, there is a negligible function* $\mathsf{negl}(\cdot)$ *such that for any family* $\mathcal{F} = \{f_n\}$ *in* $\mathcal{C}$*, for all but infinitely many* n*,*

$$|\Pr[f_n(M) = 1 \mid M \leftarrow \mathsf{ZeroSamp}(n)] - \Pr[f_n(M) = 1 \mid M \leftarrow \mathsf{OneSamp}(n)]| < \mathsf{negl}(n).$$

Since we extend previous work from NC^1 to general circuit classes $\mathcal{C}$, and our previous lemmas 15, 18 are slightly different from those in literature (see lemma 4.3 of [20], where $n = n(\lambda)$ is polynomial of security parameter), we include proofs of those lemmas in Appendix A of [13].

▶ **Construction 19** (WP-FHE [42]). *WP-FHE is given by the function family* $\mathsf{WP\text{-}FHE} = \{\mathsf{WP\text{-}FHEGen}_n, \mathsf{WP\text{-}Enc}_n, \mathsf{WP\text{-}Dec}_n, \mathsf{WP\text{-}Eval}_n\}_{n\in\mathbb{N}}$.

- $\mathsf{WP\text{-}FHEGen}_n$: *Take* $A^T, k \leftarrow \mathsf{ZeroSamp}(n)$, *and set* $\mathsf{pk} = A, \mathsf{sk} = k$.
- $\mathsf{WP\text{-}FHEGen}'_n$: *Take* $A^T \leftarrow \mathsf{OneSamp}(n)$, *and then output* $\mathsf{pk} = A$.
- $\mathsf{WP\text{-}Enc}_n(\mathsf{pk} = A, m \in \{0,1\})$: *Take* $R \leftarrow \mathbb{Z}_2^{n\times n}$, *then return* $\mathsf{ct} = AR + mI_n \in \mathbb{Z}_2^{n\times n}$. *We allow* $\mathsf{WP\text{-}Enc}_n$ *to take a vector in* $\mathbb{Z}_2^n$ *as input, instead of a single bit. In this case, the encryption of each dimension of that vector (using independent randomness) is concatenated together by each row. That is, for a length-*n *vector* x, $\mathsf{WP\text{-}Enc}(\mathsf{pk} = A, x) = AR + x^T \otimes I_n$, *where* $R \leftarrow \mathbb{Z}_2^{n\times n^2}$.
- $\mathsf{WP\text{-}Dec}_n(\mathsf{sk} = k, \mathsf{ct})$: *Let* c *be the* n*-th column vector of* ct, *and then return* $k \cdot c$.
- $\mathsf{WP\text{-}Eval}_n$. *We describe it by giving how to homomorphically compute addition and multiplication in* $\mathbb{Z}_2$.
  - $\mathsf{WP\text{-}Eval}_n^{add}(\mathsf{pk} = A, (\mathsf{ct}_0, \mathsf{ct}_1))$: *return* $\mathsf{ct}_0 + \mathsf{ct}_1 \in \mathbb{Z}_2^{n\times n}$.
  - $\mathsf{WP\text{-}Eval}_n^{mul}(\mathsf{pk} = A, (\mathsf{ct}_0, \mathsf{ct}_1))$: *return* $\mathsf{ct}_0\mathsf{ct}_1 \in \mathbb{Z}_2^{n\times n}$.

▶ Remark 20. We emphasize the striking similarity between GSW-FHE and WP-FHE. The (pk, sk) pairs of both schemes are (pk= A, sk= k) where A is a matrix, k is a vector with $k^TA \approx 0$ in GSW-FHE, and $k^TA = 0$ in WP-FHE. What's more, if we treat the identity matrix I as G's analog in $\mathbb{Z}_2$, then the encryption of both schemes is $\mathsf{Enc}(\mathsf{pk} = A, m) = AR + mG$. Similarly, homomorphic addition of both schemes is adding two ciphertexts together, and $\mathsf{Eval}^{\mathrm{mul}}(\mathsf{ct}_0, \mathsf{ct}_1) = \mathsf{ct}_0G^{-1}(\mathsf{ct}_1)$ for both schemes. Recall Table 1 as a summary of the above observations.

Our proof uses the following lemma:

▶ **Lemma 21** ([22]). *For any circuit class* $\mathcal{C} \supseteq \mathsf{AC}^0[2]$, *if Assumption 12 holds for* $\mathcal{C}$, *then* $\{p_A(x) = Ax : A \leftarrow \mathit{OneSamp}\}$ *is a collection of one-way permutations against* $\mathcal{C}$, *that is, for any adversary family* $\{\mathcal{A}_n\}_{n\in\mathbb{N}} \in \mathcal{C}$, *there exists a negligible function* $\mathsf{negl}(\cdot)$ *such that for infinitely many* n,

$$\Pr\left[Ay = Ax \;\middle|\; \begin{array}{l} A \leftarrow \mathsf{OneSamp}(n) \\ x \leftarrow \mathbb{Z}_2^n \\ y \leftarrow \mathcal{A}_n(A, Ax) \end{array}\right] < \mathsf{negl}(n).$$

[22] only proves the lemma for $\mathcal{C} = \mathsf{NC}^1$, but actually their proof can be applied to any circuit class $\mathcal{C}$ that satisfies requirement 12. We give a sketch of their proof.

Proof sketch. The idea is to use the adversary $\{\mathcal{A}_n\}_{n\in\mathbb{N}}$ that breaks one-wayness with probability $1/p(n)$ to distinguish ZeroSamp and OneSamp. First, checking $Ay = Ax$ involves only matrix multiplication, so we can assume $\mathcal{A}_n$ returns y with $Ay = Ax$ or $\perp$. Since $\{A_n\}_{n\in\mathbb{N}} \in \mathcal{C}$ cannot distinguish ZeroSamp and OneSamp, we have

$$\Pr\left[Ay = Ax \;\middle|\; \begin{array}{l} A \leftarrow \mathsf{ZeroSamp}(n) \\ x \leftarrow \mathbb{Z}_2^n \\ y \leftarrow \mathcal{A}_n(A, Ax) \end{array}\right] > \frac{1}{p(n)} - \mathsf{negl}(n) > \frac{1}{p'(n)}.$$

For A sampled from ZeroSamp, whenever $\mathcal{A}_n$ finds y s.t. $Ay = Ax$, we have probability $\frac{1}{2}$ that $x \neq y$ due to the randomness of x (recall A is of rank $n-1$, so y has two solutions), in which case we have $x - y$ as a non-trivial kernel of A. Therefore by invoking $\mathcal{A}_n$ we have non-negligible probability to obtain the kernel of A, which can be used to distinguish ZeroSamp and OneSamp (recall that the output of OneSamp is always full-rank).

We emphasize that only matrix multiplication in $\mathbb{Z}_2$ is used in this reduction, and thus the reduction is computable in $\mathcal{C} \supseteq \mathsf{AC}^0[2]$. ◀

3.2 CI from SIS against bounded-depth adversaries

In this subsection, we follow the construction of [40] to show that it can be computed in TC^0, yielding a CI hash against bounded-depth adversary under the super-weak assumption of SIS against bounded-depth adversaries.

▶ **Theorem 22.** *For any circuit class $\mathcal{C}$ that contains TC^0, for any polynomial $S(n)$ and constant d, assuming the hardness of $\mathsf{SIS}_{n,m+1,q,\beta}$ for $q = \mathsf{poly}(n), l = \lceil \log q \rceil, m = nl$ and sufficiently large $\beta = m^{O(d)}$ against $\mathcal{C}$, there exists a hash family computable in TC^0 that is CI for any relation class searchable by size-$S(n)$[6], degree-d polynomials $\{f_n : \{0,1\}^n \to \{0,1\}^m\}_{n\in\mathbb{N}}$, against adversaries in $\mathcal{C}$.*

This theorem is proven by the construction exhibited in this subsection (see Construction 24, Lemma 26). By lemma B.3 of [13], when n is large enough, by setting $q \geq n^{\Omega(d)}$ s.t. $q \geq \beta \cdot n^{\Omega(1)}$, there is an polynomial time reduction from $n^{\Omega(d)}$-approximate SIVP to $\mathsf{SIS}_{n,m+1,q,\beta}$.

▶ Remark 23. The hash function is only CI for constant-degree polynomials because TC^0 circuits can only perform a constant number of homomorphic multiplications (which involve matrix multiplications). Therefore, achieving constant-degree polynomials is the best we can do with FHE-based constructions.

▶ **Construction 24.** *The hash family $\mathcal{H} = \{h_n\}_{n\in\mathbb{N}}$ with key generation algorithm $\mathsf{Gen}_{\mathcal{H}} = \{\mathsf{Gen}_{\mathcal{H},n}\}_{n\in\mathbb{N}}$ is parameterized by an arbitrary circuit size $S(n) = \mathsf{poly}(n)$ and depth d. Let $\mathcal{U}(C, x) = C(x)$ denote the universal circuit for size-S, degree-d polynomials (hence, it is also a constant degree polynomial itself). We write $q = q(n), l = \lceil \log q \rceil, m = nl$.*

- $\mathsf{Gen}_{\mathcal{H},n}$*: generate $A \leftarrow \mathbb{Z}_q^{n\times 2m}$ and $C \leftarrow \mathsf{GSW\text{-}Enc}(A, 0^{S(n)}) = AR + 0^{S(n)} \otimes G$, where $R \in \mathbb{Z}^{2m\times m}$. Choose a uniform random $a \leftarrow \mathbb{Z}_q^n$, output the hash key $k = (a, C)$.*
- $h_n(k = (a, C), x \in \{0,1\}^n)$*: let circuit $U_x(\cdot) = \mathcal{U}(x, \cdot)$, and output*

$$G_n^{-1}\left[a + \mathsf{InertEval}(G_n, \mathsf{GSW\text{-}Eval}(U_x, C))\right] \in \{0,1\}^m.$$

We first show this hash function family can be computed in TC^0, then show its correlation intractability. The latter property directly follows the proof of Theorem 3.4 in [40].

▶ **Lemma 25.** *Construction 24 can be computed by TC^0 circuits.*

Proof. Computing $\mathsf{Gen}_{\mathcal{H},n}$ and h_n only requires:

- Computing iterative sum mod $q = \mathsf{poly}(n)$; (In this construction, we only need to compute the product of a normal matrix (with each element in $\mathbb{Z}_q$) and a binary matrix (with each element in $\{0,1\}$), so it suffices to compute iterative sum in $\mathbb{Z}_q$.)
- Generating universal circuit U_x from x; (we only need to replace some input gates with binary values.)
- Computing G_n^{-1}, i.e. bit decomposition, which is trivial since we store a number in $\mathbb{Z}_q$ as a binary number.

All these tasks can be computed in TC^0. ◀

▶ **Lemma 26.** *Assuming the hardness of $\mathsf{SIS}_{n,m,q,\beta}$ for a sufficiently large $\beta = m^{O(d)}$ against $\mathcal{C}$ adversary, Construction 24 is correlation intractable (cf. Theorem 22).*

[6] Here "size" means the encoding length of the constant degree polynomial.

Proof. Let $\mathcal{A} = \{\mathcal{A}_n\}_{n\in\mathbb{N}} \in \mathcal{C}$ be any adversary that breaks correlation intractability against $\{f_n\}$, i.e.

$$\Pr\left[h_n(k=(a,C),x) = f_n(x) \;\middle|\; \begin{array}{l} A \leftarrow \mathbb{Z}_q^{n\times 2m} \\ C \leftarrow \mathsf{GSW\text{-}Enc}(A, 0^{S(n)}) \\ a \leftarrow \mathbb{Z}_q^n \\ x \leftarrow \mathcal{A}_n(k) \end{array}\right]$$

is non-negligible.

First, we change $C \leftarrow \mathsf{GSW\text{-}Enc}(A, 0^{S(n)})$ to $C \leftarrow \mathsf{GSW\text{-}Enc}(A, f)$ for $f = f_n$ by hybrid argument. It can be done because the distribution of C is statistically indistinguishable by the following lemma.

▶ **Lemma 27** (Item 1 of Proposition 2.10 in [40]). *For any $x \in \mathbb{Z}_q^n$, the statistical distance between the distribution of $(A \leftarrow \mathbb{Z}_q^{n\times 2m}, C = \mathsf{GSW\text{-}Enc}(A, x))$ and uniformly random is in $\mathsf{negl}(m)$ by the leftover hash lemma.*

Therefore,

$$\Pr\left[h_n(k=(a,C),x) = f_n(x) \;\middle|\; \begin{array}{l} A \leftarrow \mathbb{Z}_q^{n\times 2m} \\ C \leftarrow \mathsf{GSW\text{-}Enc}(A, f_n) \\ a \leftarrow \mathbb{Z}_q^n \\ x \leftarrow \mathcal{A}_n(k) \end{array}\right]$$

is also non-negligible. Then we can use $\mathcal{A}$ to break SIS. Since $h_n(k,x) = f_n(x)$ implies that

$$\begin{aligned} G_n \cdot f_n(x) =& G_n \cdot h_n(k,x) \\ =& a + \mathsf{InertEval}(G_n, \mathsf{GSW\text{-}Eval}(U_x, C)) \\ =& a + (Ar + G_n \cdot f_n(x)) \\ =& A'z + G_n \cdot f_n(x), \end{aligned}$$

where

$$A' = \begin{bmatrix} a & A \end{bmatrix} \in \mathbb{Z}_q^{n\times(m+1)}, z = (1,r) \in \mathbb{Z}^{m+1}.$$

Note that $||z|| = m^{O(d)}$ as U_x is done by universally computing depth-d circuits.

So our SIS attacker works as follows: given an SIS instance A', the attacker takes A as the last m columns of A' and generates $C \leftarrow \mathsf{GSW\text{-}Enc}(A, f)$ with randomness R retained. After taking $x \leftarrow \mathcal{A}_n(k)$, it simulates the computation of $h_n(k,x)$ to get r, then outputs $z = (1,r)$. The attacker is in $\mathcal{C}$ since the reduction is in TC^0. ◀

3.3 CI from worst-case complexity assumptions

In this subsection, we bypass the reliance on the lattice assumptions, which therefore also allows the hash function to be computable in $\mathsf{AC}^0[2]$, by applying techniques from [40] to WP-FHE (see Construction 19) instead of GSW-FHE (see Construction 7). This approach provides the construction of a hash function family that is computable in $\mathsf{AC}^0[2]$ and correlation-intractable for any relation searchable in constant-degree polynomials (with bounded circuit size) against any circuit class $\mathcal{C} \supseteq \mathsf{AC}^0[2]$, assuming $\oplus\mathsf{L/poly} \not\subseteq \widetilde{\mathsf{Sum}}_{n^{-c}} \circ \mathcal{C}$ for some $c > 0$. The result can be stated as follows:

▶ **Theorem 28.** *For any circuit class $\mathcal{C} \supseteq \mathsf{AC}^0[2]$, if Assumption 12 holds for $\mathcal{C}$, for any polynomial $S(n)$ and constant d, there exists a hash family computable in $\mathsf{AC}^0[2]$ that is CI for any relation class searchable by degree-d polynomials $\{f_n : \{0,1\}^n \to \{0,1\}^n\}_{n\in\mathbb{N}}$ computable by circuits of size $S(n)$, against adversaries in $\mathcal{C}$ (following the "infinitely many" version of security).*

3.3.1 Adaptation of inert commitment

In our construction, all computation are performed in $\mathbb{Z}_2$, hence we restate the inert commitment in $\mathbb{Z}_2$, i.e, set $q = 2$. Although it is just the special case for $q = 2$, we emphasize that in this case, G is simplified to the identify matrix (as what we observed in WP-FHE), hence the inert commitment is greatly simplified.

▶ **Definition 29** (Inert Commitment in $\mathbb{Z}_2$)**.** *Given $C = \mathsf{WP\text{-}Enc}(A, x) \in \mathbb{Z}_2^{n\times n^2}$ as the homomorphic encryption of some length-n vector, and $M \in \mathbb{Z}_2^{n\times n}$ as a matrix, we write*

$$\mathsf{InertEval}(M, C) := c_M = C \cdot v_M,$$

where $v_M \in \mathbb{Z}_2^{n^2}$ is the vector satisfies $(x^T \otimes I_n) \cdot v_M = Mx$ for any length-n vector x, obtained by reordering the elements of M.

3.3.2 Construction

Here we present the construction of the CI hash family to prove Theorem 28.

▶ **Construction 30.** *The hash family $\mathcal{H} = \{h_n\}_{n\in\mathbb{N}}$ with the key generation algorithm $\mathsf{Gen}_{\mathcal{H}} = \{\mathsf{Gen}_{\mathcal{H},n}\}_{n\in\mathbb{N}}$ is parameterized by an arbitrary circuit size $S(n) = \mathsf{poly}(n)$ and depth d. Let $U(C, x) = C(x)$ denote a universal circuit for size-S, degree-d polynomials.*

- *$\mathsf{Gen}_{\mathcal{H},n}$: generate $A \leftarrow \mathsf{OneSamp}$ and $C \leftarrow \mathsf{WP\text{-}Enc}(A, 0^{S(n)})$, choose a uniformly random $a \leftarrow \mathbb{Z}_2^n$ and output the hash key $k = (a, C)$.*
- *$h_n(k = (a, C), x \in \{0,1\}^n)$: define the circuit $U_x(\cdot)$ as $U_x(\cdot) = U(x, \cdot)$ and output*

$$a + \mathsf{InertEval}(I_n, \mathsf{WP\text{-}Eval}(U_x, C)) \in \{0,1\}^n.$$

Proof of Theorem 28. We prove that for any circuit class $\mathcal{C} \supseteq \mathsf{AC}^0[2]$, Construction 30 is a CI hash family against $\mathcal{C}$ if Assumption 12 holds for $\mathcal{C}$. Let $\mathcal{A} = \{\mathcal{A}_n\} \in \mathcal{C}$ be any adversary family that breaks CI for some sequence of polynomials $\{f_n\}$. Since given x, we can check whether $h_n((a, C), x) = f_n(x)$ in $\mathcal{C}$ (which only involves matrix multiplication and computing $f_n(x)$), w.l.o.g., we assume that $\mathcal{A}_n$ always outputs some $x \in \{0,1\}^n$ s.t. $h_n((a, C), x) = f_n(x)$ with non-negligible probability or outputs $\perp$ after taking $k = (a, C)$ sampled according to $\mathsf{Gen}_{\mathcal{H},n}$ as input.

We abuse the notation to let $f_n \in \{0,1\}^{S(n)}$ also denote the encoding of the circuit that computes this function. The idea is to apply the hybrid argument to change the input distribution of $\mathcal{A}_n$. We observe that the following two distribution are equivalent, since A (sampled from $\mathsf{OneSamp}$) is always full-rank, which makes AR below being uniformly random. Hence the two distributions are indistinguishable for $\mathcal{C}$ circuits.

- $\mathcal{D}_n^0 = \{(a, \mathsf{WP\text{-}Enc}(A, 0^{S(n)}) = AR + 0^{S(n)} \otimes I_n) : a \leftarrow \mathbb{Z}_2^n, A \leftarrow \mathsf{OneSamp}, R \leftarrow \mathbb{Z}_2^{n\times nS(n)}\}$.
- $\mathcal{D}_n^1 = \{(a, \mathsf{WP\text{-}Enc}(A, f_n) = AR + f_n \otimes I_n) : a \leftarrow \mathbb{Z}_2^n, A \leftarrow \mathsf{OneSamp}, R \leftarrow \mathbb{Z}_2^{n\times nS(n)}\}$.

▶ **Lemma 31.** *For any circuit class $\mathcal{C} \supseteq \mathsf{AC}^0[2]$, if Assumption 12 holds for $\mathcal{C}$, any circuit family $\{\mathcal{A}_n\} \in \mathcal{C}$ cannot distinguish $\mathcal{D}_n^0$ and $\mathcal{D}_n^1$, i.e,*

$$\left| \Pr\left[\mathcal{A}_n(k) = 1 \;\middle|\; k \leftarrow \mathcal{D}_n^0\right] - \Pr\left[\mathcal{A}_n(k) = 1 \;\middle|\; k \leftarrow \mathcal{D}_n^1\right] \right| < \mathsf{negl}(n).$$

So after switching the input to $\mathcal{A}_n$ from $k \leftarrow \mathcal{D}_n^0$ to $k' \leftarrow \mathcal{D}_n^1$, $\mathcal{A}_n$ should also output $x \in \{0,1\}^n$ s.t. $h_n(k' = (a, \mathsf{WP\text{-}Enc}(A, f_n)), x) = f_n(x)$ with non-negligible probability. Since

$$\begin{aligned} f_n(x) =& h_n(k', x) \\ =& a + \mathsf{InertEval}(I_n, \mathsf{WP\text{-}Eval}(U_x, \mathsf{WP\text{-}Enc}(A, f_n))) \\ =& a + \mathsf{InertEval}(I_n, AR + f_n(x)^t \otimes I_n) \\ =& a + Ar_{I_n} + f_n(x), \end{aligned}$$

we have $Ar_{I_n} = a$ (in $\mathbb{Z}_2$). So if we record the randomness used to compute $\mathsf{WP\text{-}Enc}(A, f_n)$, by calculating $h_n(k', x)$, we are able to compute $r_{I_n} = A^{-1}a$. However, this allows us to invert $p_A(x) = Ax$, which is impossible by Lemma 21. ◀

References

1. Miklós Ajtai. Generating hard instances of lattice problems (extended abstract). In *STOC*, pages 99–108, 1996. doi:10.1145/237814.237838.
2. Benny Applebaum, Yuval Ishai, and Eyal Kushilevitz. Cryptography in nc^0. *SIAM J. Comput.*, 36(4):845–888, 2006. doi:10.1137/S0097539705446950.
3. Boaz Barak. How to go beyond the black-box simulation barrier. In *FOCS*, pages 106–115. IEEE Computer Society, 2001. doi:10.1109/SFCS.2001.959885.
4. Boaz Barak and Oded Goldreich. Universal arguments and their applications. *SIAM J. Comput.*, 38(5):1661–1694, 2008. doi:10.1137/070709244.
5. James Bartusek, Liron Bronfman, Justin Holmgren, Fermi Ma, and Ron D. Rothblum. On the (in)security of kilian-based snargs. In Dennis Hofheinz and Alon Rosen, editors, *Theory of Cryptography - 17th International Conference, TCC 2019, Nuremberg, Germany, December 1-5, 2019, Proceedings, Part II*, volume 11892 of *Lecture Notes in Computer Science*, pages 522–551. Springer, 2019. doi:10.1007/978-3-030-36033-7_20.
6. Nir Bitansky, Yael Tauman Kalai, and Omer Paneth. Multi-collision resistance: a paradigm for keyless hash functions. In *STOC*, pages 671–684. ACM, 2018. doi:10.1145/3188745.3188870.
7. Zvika Brakerski, Venkata Koppula, and Tamer Mour. NIZK from LPN and trapdoor hash via correlation intractability for approximable relations. In Daniele Micciancio and Thomas Ristenpart, editors, *Advances in Cryptology - CRYPTO 2020 - 40th Annual International Cryptology Conference, CRYPTO 2020, Santa Barbara, CA, USA, August 17-21, 2020, Proceedings, Part III*, volume 12172 of *Lecture Notes in Computer Science*, pages 738–767. Springer, 2020. doi:10.1007/978-3-030-56877-1_26.
8. Matteo Campanelli and Rosario Gennaro. Fine-grained secure computation. In Amos Beimel and Stefan Dziembowski, editors, *Theory of Cryptography - 16th International Conference, TCC 2018, Panaji, India, November 11-14, 2018, Proceedings, Part II*, volume 11240 of *Lecture Notes in Computer Science*, pages 66–97. Springer, 2018. doi:10.1007/978-3-030-03810-6_3.
9. Ran Canetti, Yilei Chen, Justin Holmgren, Alex Lombardi, Guy N. Rothblum, Ron D. Rothblum, and Daniel Wichs. Fiat-shamir: from practice to theory. In *STOC*, pages 1082–1090. ACM, 2019. doi:10.1145/3313276.3316380.
10. Ran Canetti, Yilei Chen, and Leonid Reyzin. On the correlation intractability of obfuscated pseudorandom functions. In Eyal Kushilevitz and Tal Malkin, editors, *Theory of Cryptography - 13th International Conference, TCC 2016-A, Tel Aviv, Israel, January 10-13, 2016, Proceedings, Part I*, volume 9562 of *Lecture Notes in Computer Science*, pages 389–415. Springer, 2016. doi:10.1007/978-3-662-49096-9_17.

11 Ran Canetti, Yilei Chen, Leonid Reyzin, and Ron D. Rothblum. Fiat-shamir and correlation intractability from strong kdm-secure encryption. In Jesper Buus Nielsen and Vincent Rijmen, editors, *Advances in Cryptology - EUROCRYPT 2018 - 37th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Tel Aviv, Israel, April 29 - May 3, 2018 Proceedings, Part I*, volume 10820 of *Lecture Notes in Computer Science*, pages 91–122. Springer, 2018. doi:10.1007/978-3-319-78381-9_4.

12 Ran Canetti, Oded Goldreich, and Shai Halevi. The random oracle methodology, revisited. *J. ACM*, 51(4):557–594, 2004. doi:10.1145/1008731.1008734.

13 Liyan Chen, Yilei Chen, Zikuan Huang, Nuozhou Sun, Tianqi Yang, and Yiding Zhang. Fiat-shamir for bounded-depth adversaries. Cryptology ePrint Archive, Paper 2024/256, 2024. URL: https://eprint.iacr.org/2024/256.

14 Yilei Chen, Alex Lombardi, Fermi Ma, and Willy Quach. Does fiat-shamir require a cryptographic hash function? In Tal Malkin and Chris Peikert, editors, *Advances in Cryptology - CRYPTO 2021 - 41st Annual International Cryptology Conference, CRYPTO 2021, Virtual Event, August 16-20, 2021, Proceedings, Part IV*, volume 12828 of *Lecture Notes in Computer Science*, pages 334–363. Springer, 2021. doi:10.1007/978-3-030-84259-8_12.

15 Arka Rai Choudhuri, Sanjam Garg, Abhishek Jain, Zhengzhong Jin, and Jiaheng Zhang. Correlation intractability and snargs from sub-exponential DDH. In Helena Handschuh and Anna Lysyanskaya, editors, *Advances in Cryptology - CRYPTO 2023 - 43rd Annual International Cryptology Conference, CRYPTO 2023, Santa Barbara, CA, USA, August 20-24, 2023, Proceedings, Part IV*, volume 14084 of *Lecture Notes in Computer Science*, pages 635–668. Springer, 2023. doi:10.1007/978-3-031-38551-3_20.

16 Arka Rai Choudhuri, Pavel Hubácek, Chethan Kamath, Krzysztof Pietrzak, Alon Rosen, and Guy N. Rothblum. Finding a nash equilibrium is no easier than breaking fiat-shamir. In *STOC*, pages 1103–1114. ACM, 2019. doi:10.1145/3313276.3316400.

17 Arka Rai Choudhuri, Abhishek Jain, and Zhengzhong Jin. Non-interactive batch arguments for NP from standard assumptions. In Tal Malkin and Chris Peikert, editors, *Advances in Cryptology - CRYPTO 2021 - 41st Annual International Cryptology Conference, CRYPTO 2021, Virtual Event, August 16-20, 2021, Proceedings, Part IV*, volume 12828 of *Lecture Notes in Computer Science*, pages 394–423. Springer, 2021. doi:10.1007/978-3-030-84259-8_14.

18 Arka Rai Choudhuri, Abhishek Jain, and Zhengzhong Jin. Snargs for P from LWE. In *FOCS*, pages 68–79. IEEE, 2021.

19 Quang Dao, Aayush Jain, and Zhengzhong Jin. Non-interactive zero-knowledge from LPN and MQ. In *CRYPTO (9)*, volume 14928 of *Lecture Notes in Computer Science*, pages 321–360. Springer, 2024. doi:10.1007/978-3-031-68400-5_10.

20 Akshay Degwekar, Vinod Vaikuntanathan, and Prashant Nalini Vasudevan. Fine-grained cryptography. In Matthew Robshaw and Jonathan Katz, editors, *Advances in Cryptology - CRYPTO 2016 - 36th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 14-18, 2016, Proceedings, Part III*, volume 9816 of *Lecture Notes in Computer Science*, pages 533–562. Springer, 2016. doi:10.1007/978-3-662-53015-3_19.

21 Cynthia Dwork, Moni Naor, Omer Reingold, and Larry J. Stockmeyer. Magic functions. In *FOCS*, pages 523–534. IEEE Computer Society, 1999. doi:10.1109/SFFCS.1999.814626.

22 Shohei Egashira, Yuyu Wang, and Keisuke Tanaka. Fine-grained cryptography revisited. *J. Cryptol.*, 34(3):23, 2021. doi:10.1007/s00145-021-09390-3.

23 Amos Fiat and Adi Shamir. How to prove yourself: Practical solutions to identification and signature problems. In Andrew M. Odlyzko, editor, *Advances in Cryptology - CRYPTO '86, Santa Barbara, California, USA, 1986, Proceedings*, volume 263 of *Lecture Notes in Computer Science*, pages 186–194. Springer, 1986. doi:10.1007/3-540-47721-7_12.

24 Craig Gentry, Amit Sahai, and Brent Waters. Homomorphic encryption from learning with errors: Conceptually-simpler, asymptotically-faster, attribute-based. In Ran Canetti and Juan A. Garay, editors, *Advances in Cryptology - CRYPTO 2013 - 33rd Annual Cryptology*

Conference, Santa Barbara, CA, USA, August 18-22, 2013. Proceedings, Part I, volume 8042 of *Lecture Notes in Computer Science*, pages 75–92. Springer, 2013. doi:10.1007/978-3-642-40041-4_5.

25 Craig Gentry and Daniel Wichs. Separating succinct non-interactive arguments from all falsifiable assumptions. In *STOC*, pages 99–108. ACM, 2011. doi:10.1145/1993636.1993651.

26 Shafi Goldwasser and Yael Tauman Kalai. On the (in)security of the fiat-shamir paradigm. In *44th Symposium on Foundations of Computer Science (FOCS 2003), 11-14 October 2003, Cambridge, MA, USA, Proceedings*, pages 102–113. IEEE Computer Society, 2003. doi:10.1109/SFCS.2003.1238185.

27 Justin Holmgren and Alex Lombardi. Cryptographic hashing from strong one-way functions (or: One-way product functions and their applications). In Mikkel Thorup, editor, *59th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2018, Paris, France, October 7-9, 2018*, pages 850–858. IEEE Computer Society, 2018. doi:10.1109/FOCS.2018.00085.

28 Justin Holmgren, Alex Lombardi, and Ron D. Rothblum. Fiat-shamir via list-recoverable codes (or: parallel repetition of GMW is not zero-knowledge). In Samir Khuller and Virginia Vassilevska Williams, editors, *STOC '21: 53rd Annual ACM SIGACT Symposium on Theory of Computing, Virtual Event, Italy, June 21-25, 2021*, pages 750–760. ACM, 2021. doi:10.1145/3406325.3451116.

29 James Hulett, Ruta Jawale, Dakshita Khurana, and Akshayaram Srinivasan. Snargs for P from sub-exponential DDH and QR. In Orr Dunkelman and Stefan Dziembowski, editors, *Advances in Cryptology - EUROCRYPT 2022 - 41st Annual International Conference on the Theory and Applications of Cryptographic Techniques, Trondheim, Norway, May 30 - June 3, 2022, Proceedings, Part II*, volume 13276 of *Lecture Notes in Computer Science*, pages 520–549. Springer, 2022. doi:10.1007/978-3-031-07085-3_18.

30 Yuval Ishai and Eyal Kushilevitz. Randomizing polynomials: A new representation with applications to round-efficient secure computation. In *FOCS*, pages 294–304. IEEE Computer Society, 2000. doi:10.1109/SFCS.2000.892118.

31 Abhishek Jain and Zhengzhong Jin. Non-interactive zero knowledge from sub-exponential DDH. In Anne Canteaut and François-Xavier Standaert, editors, *Advances in Cryptology - EUROCRYPT 2021 - 40th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Zagreb, Croatia, October 17-21, 2021, Proceedings, Part I*, volume 12696 of *Lecture Notes in Computer Science*, pages 3–32. Springer, 2021. doi:10.1007/978-3-030-77870-5_1.

32 Ruta Jawale, Yael Tauman Kalai, Dakshita Khurana, and Rachel Yun Zhang. Snargs for bounded depth computations and PPAD hardness from sub-exponential LWE. In Samir Khuller and Virginia Vassilevska Williams, editors, *STOC '21: 53rd Annual ACM SIGACT Symposium on Theory of Computing, Virtual Event, Italy, June 21-25, 2021*, pages 708–721. ACM, 2021. doi:10.1145/3406325.3451055.

33 Yael Tauman Kalai, Alex Lombardi, and Vinod Vaikuntanathan. Snargs and PPAD hardness from the decisional diffie-hellman assumption. In Carmit Hazay and Martijn Stam, editors, *Advances in Cryptology - EUROCRYPT 2023 - 42nd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Lyon, France, April 23-27, 2023, Proceedings, Part II*, volume 14005 of *Lecture Notes in Computer Science*, pages 470–498. Springer, 2023. doi:10.1007/978-3-031-30617-4_16.

34 Yael Tauman Kalai, Guy N. Rothblum, and Ron D. Rothblum. From obfuscation to the security of fiat-shamir for proofs. In Jonathan Katz and Hovav Shacham, editors, *Advances in Cryptology - CRYPTO 2017 - 37th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 20-24, 2017, Proceedings, Part II*, volume 10402 of *Lecture Notes in Computer Science*, pages 224–251. Springer, 2017. doi:10.1007/978-3-319-63715-0_8.

35 Yael Tauman Kalai, Vinod Vaikuntanathan, and Rachel Yun Zhang. Somewhere statistical soundness, post-quantum security, and snargs. In Kobbi Nissim and Brent Waters, editors, *Theory of Cryptography - 19th International Conference, TCC 2021, Raleigh, NC, USA, November 8-11, 2021, Proceedings, Part I*, volume 13042 of *Lecture Notes in Computer Science*, pages 330–368. Springer, 2021. doi:10.1007/978-3-030-90459-3_12.

36 Joe Kilian. A note on efficient zero-knowledge proofs and arguments (extended abstract). In *STOC*, pages 723–732. ACM, 1992. doi:10.1145/129712.129782.

37 Susumu Kiyoshima. Public-coin 3-round zero-knowledge from learning with errors and keyless multi-collision-resistant hash. In *CRYPTO (1)*, volume 13507 of *Lecture Notes in Computer Science*, pages 444–473. Springer, 2022. doi:10.1007/978-3-031-15802-5_16.

38 Alex Lombardi and Vinod Vaikuntanathan. Correlation-intractable hash functions via shift-hiding. In *ITCS*, volume 215 of *LIPIcs*, pages 102:1–102:16. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2022. doi:10.4230/LIPIcs.ITCS.2022.102.

39 Silvio Micali. Computationally sound proofs. *SIAM J. Comput.*, 30(4):1253–1298, October 2000. doi:10.1137/S0097539795284959.

40 Chris Peikert and Sina Shiehian. Noninteractive zero knowledge for NP from (plain) learning with errors. In Alexandra Boldyreva and Daniele Micciancio, editors, *Advances in Cryptology - CRYPTO 2019 - 39th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 18-22, 2019, Proceedings, Part I*, volume 11692 of *Lecture Notes in Computer Science*, pages 89–114. Springer, 2019. doi:10.1007/978-3-030-26948-7_4.

41 David Pointcheval and Jacques Stern. Security proofs for signature schemes. In Ueli M. Maurer, editor, *Advances in Cryptology - EUROCRYPT '96, International Conference on the Theory and Application of Cryptographic Techniques, Saragossa, Spain, May 12-16, 1996, Proceeding*, volume 1070 of *Lecture Notes in Computer Science*, pages 387–398. Springer, 1996. doi:10.1007/3-540-68339-9_33.

42 Yuyu Wang and Jiaxin Pan. Non-interactive zero-knowledge proofs with fine-grained security. In Orr Dunkelman and Stefan Dziembowski, editors, *Advances in Cryptology - EUROCRYPT 2022 - 41st Annual International Conference on the Theory and Applications of Cryptographic Techniques, Trondheim, Norway, May 30 - June 3, 2022, Proceedings, Part II*, volume 13276 of *Lecture Notes in Computer Science*, pages 305–335. Springer, 2022. doi:10.1007/978-3-031-07085-3_11.

Weak Zero-Knowledge and One-Way Functions

Rohit Chatterjee
Department of Computer Science, National University of Singapore, Singapore

Yunqi Li
Department of Computer Science, National University of Singapore, Singapore

Prashant Nalini Vasudevan
Department of Computer Science, National University of Singapore, Singapore

Abstract

We study the implications of the existence of weak Zero-Knowledge (ZK) protocols for worst-case hard languages. These are protocols that have completeness, soundness, and zero-knowledge errors (denoted ϵ_c, ϵ_s, and ϵ_z, respectively) that might not be negligible. Under the assumption that there are worst-case hard languages in NP, we show the following:

1. If all languages in NP have NIZK proofs or arguments satisfying $\epsilon_c + \epsilon_s + \epsilon_z < 1$, then One-Way Functions (OWFs) exist. This covers all possible non-trivial values for these error rates. It additionally implies that if all languages in NP have such NIZK proofs and ϵ_c is negligible, then they also have NIZK proofs where all errors are negligible. Previously, these results were known under the more restrictive condition $\epsilon_c + \sqrt{\epsilon_s} + \epsilon_z < 1$ [Chakraborty et al., CRYPTO 2025].
2. If all languages in NP have k-round public-coin ZK proofs or arguments satisfying $\epsilon_c + \epsilon_s + (2k-1) \cdot \epsilon_z < 1$, then OWFs exist.
3. If, for some constant k, all languages in NP have k-round public-coin ZK proofs or arguments satisfying $\epsilon_c + \epsilon_s + k \cdot \epsilon_z < 1$, then infinitely-often OWFs exist.

2012 ACM Subject Classification Theory of computation → Computational complexity and cryptography

Keywords and phrases One-way functions, zero-knowledge

Digital Object Identifier 10.4230/LIPIcs.ITC.2026.5

Related Version *Full Version*: https://eprint.iacr.org/2026/299

Funding This work was supported by the National Research Foundation, Singapore, under its NRF Fellowship programme, award no. NRF-NRFF14-2022-0010.

Acknowledgements We thank the anonymous reviewers for their comments.

Declaration on GenAI/LLM use Commercial AI tools (ChatGPT, Claude) were used as typing assistants for grammar and basic editing, and for mild assistance with LaTeX formatting.

1 Introduction

The notion of Zero-Knowledge (ZK) protocols is a vital part of modern cryptography. These are interactive proof systems in which a *prover* proves to a *verifier* the validity of a given statement, with the additional guarantee that even a possibly cheating verifier cannot obtain any secrets that might have been used in the proof. More precisely, such protocols guarantee *soundness*, i.e. cheating provers cannot prove false statements, and *zero knowledge*, i.e. a cheating verifier cannot glean anything beyond the validity of the statement over the course of the protocol.

Hardness from Zero-Knowledge

One natural question arising in the context of relating zero-knowledge to other cryptographic notions is that of which other cryptographic primitives are implied by it. This was first studied in the work of Ostrovsky [11], who showed that a statistical ZK proof system for any average-case hard language implies the existence of One-Way Functions (OWFs). This was later generalized by Ostrovsky and Wigderson [12] to computational ZK proofs, and they showed in addition that such proofs for even a worst-case hard language implies a weaker form of OWFs called auxiliary-input OWFs.

Building on these, the recent work of Hirahara and Nanashima [7] showed that if all languages in NP have computational ZK proofs (or even arguments, which only have computational soundness) and NP is hard in the worst-case, then OWFs exist.

Weak Zero-Knowledge

The above results all work with proof systems where the completeness, soundness, and zero-knowledge errors are guaranteed to be negligible. There are, however, a number of natural and useful ZK protocols, such as the commonly taught protocols for 3-Coloring and Graph Non-isomorphism, that do not natively have negligible error rates. Such *Weak ZK* protocols may have completeness error ϵ_c, soundness error of ϵ_s, and zero-knowledge error ϵ_z, that may be as large as a constant number. Along the same lines as above, it is natural and important to study the power of such protocols as well.

Amplifying Weak NIZKs

The work of Goyal et al. [6] was a first step in this direction, investigating the power of weak *Non-Interactive ZK* (NIZK) arguments. NIZKs consider a non-interactive setting where the prover and verifier have access to a common random string, and the protocol only involves a single prover message, following which the verifier decides whether to accept or reject. They showed that weak NIZKs with negligible ε_c satisfying $\epsilon_s + \epsilon_z < 1 - \delta$ for any non-zero constant δ can be amplified to a standard NIZK argument with negligible errors if we additionally have access to a sub-exponentially secure Public Key Encryption (PKE) scheme.

Bitansky and Geier [2] improved this result to show that standard PKE suffices for this amplification. They also showed that if the NIZK system is a proof (i.e., has statistical soundness), then amplification is possible assuming only OWFs. Applebaum and Kachlon [1] improved on this to allow for δ to be as small as an inverse-polynomial function.

Hardness from Weak ZK

Seeking to reduce the assumptions needed for such amplification, Chakraborty et al. [4] showed that in certain settings, weak NIZKs can be used to derive OWFs. Specifically, they show that if all languages in NP have NIZK arguments satisfying $\epsilon_c + \sqrt{\epsilon_s} + \epsilon_z < 1$, then OWFs exist under just the worst-case assumption that $\mathsf{NP} \not\subseteq \mathsf{ioP/poly}$[1]. They then combined this with the amplification results of [2, 1] to show that under the additional hypotheses that these are NIZK proofs with negligible ε_c, they could get NIZK proofs with negligible errors for all of NP.

[1] That is, there are no polynomial-size circuit families for NP, even if they are only required to be correct infinitely often.

While this result helps characterize the hardness of a broad class of weak NIZKs, it is still somewhat unsatisfactory as it does not cover *all* possible non-trivial weak NIZK parameters. As noted in [4], the setting $\epsilon_c + \epsilon_s + \epsilon_z \geq 1$ is not meaningful for NIZK protocols. Thus, to complete the picture, what is required is to understand the implications of any weak NIZK protocol satisfying $\epsilon_c + \epsilon_s + \epsilon_z < 1$.

Another important question that has not been studied in this recent line of work is that of the complexity of weak interactive ZK protocols. The earlier implications of ZK shown in [12, 7] work for interactive ZK protocols with negligible errors. But the extent of their validity for weak ZK protocols was not understood.

Our Results

Our first result addresses the first question above, extending the construction of OWFs from NIZKs to the most general parameters, under the same assumptions as in prior work.

▶ **Theorem 1** (Informally, Theorem 43). *If* NP $\not\subseteq$ ioP/poly*, and every language in* NP *has an* $(\epsilon_c, \epsilon_s, \epsilon_z)$*-NIZK proof (or argument) with* $\epsilon_c + \epsilon_s + \epsilon_z < 1$*, then one-way functions exist.*

Similar to [4], we can in turn use the amplification results of [2, 1] to obtain amplification of NIZK proofs with near-perfect completeness, from the most general setting of the remaining errors.

▶ **Corollary 2** (NIZK Amplification). *If* NP $\not\subseteq$ ioP/poly*, and every language in* NP *has an* $(\epsilon_c, \epsilon_s, \epsilon_z)$*-NIZK proof with* $\epsilon_c + \epsilon_s + \epsilon_z < 1$ *and negligible* ϵ_c*, then every language in* NP *has a NIZK proof with negligible errors. Here, the soundness of the NIZK proofs is required to be adaptive.*[2]

We also address the second question raised above for *public-coin* protocols, where the verifier's messages consist solely of uniform random bits.

▶ **Theorem 3** (Informally, Theorem 44). *If* NP $\not\subseteq$ ioP/poly*, and every language in* NP *has a* t*-message* $(\epsilon_c, \epsilon_s, \epsilon_z)$*-public-coin ZK proof (or argument) with* $\epsilon_c + \epsilon_s + (t-1) \cdot \epsilon_z < 1$*, then one-way functions exist.*

In the above case, the techniques of [12] alone would have resulted in the condition being $\epsilon_c + \epsilon_s + t \cdot \epsilon_z < 1$ instead. For constant-round protocols, we improve this condition much more significantly, though in this case we only obtain infinitely-often one-way functions. Below, a *round* refers to one pair of messages in the protocol – one from the verifier and its response from the prover.

▶ **Theorem 4** (Informally, Theorem 45). *If* NP $\not\subseteq$ P/poly*, and for some constant* k*, every language in* NP *has a* k*-round* $(\epsilon_c, \epsilon_s, \epsilon_z)$*-public-coin ZK proof (or argument) with* $\epsilon_c + \epsilon_s + k \cdot \epsilon_z < 1$*, then infinitely-often one-way functions exist.*

Our results apply to protocols with computational (weak) zero-knowledge and computational (weak) soundness (i.e., arguments), and thus capture the most general class of such protocols.

[2] In the rest of the paper, we exclusively use the weaker non-adaptive definition of soundness for NIZK protocols. This only strengthens our results, as we use NIZKs to construct other things. Here, the stronger adaptive notion of soundness is needed. See, e.g., [2] for the definition of this notion.

Concurrent Work

In concurrent and independent work, Chakraborty et al. [5] obtain results similar to ours. They prove Theorem 1 and a stronger version of Theorem 4, but they do not show Theorem 3. Most remarkably, they are able to relax the condition on the errors for constant-round public-coin ZK protocols required in Theorem 4 to $\epsilon_c + \epsilon_s + \epsilon_z < 1$, which covers all non-trivial protocols. The techniques in the two papers are considerably different.

1.1 Technical Overview

In this section, we provide a high-level overview of the main ideas and techniques behind our results. We start by reviewing the construction of [12] with their analysis for non-interactive ZKs. We then introduce our improved construction and outline the ideas that enable improvement. Further, we find that our approach naturally generalizes to a broader setting – specifically that of public-coin ZK protocols, which we will discuss later.

Throughout, we use the notation $\mathcal{U}$ to denote the uniform distribution over strings whose length will be clear from the context.

Non-interactive zero-knowledge

We first consider non-interactive ZK (NIZK) arguments. A NIZK argument for a language $\mathcal{L}$ allows a prover to produce a single proof π to certify that an input $x \in \mathcal{L}$ while preserving zero-knowledge. Specifically, given a uniformly random string r, also known as the *common reference string*, a polynomial-time prover holding with a valid NP witness w for x computes a proof $\pi \leftarrow \mathsf{P}(x, w; r)$; upon receiving the prover's message, the verifier computes $a \leftarrow \mathsf{V}(x; r, \pi)$ to decide whether $x \in \mathcal{L}$.

The protocol is required to satisfy completeness and computational soundness, where the errors are correspondingly denoted by ϵ_c and ϵ_s. Besides these, it also satisfies computational zero-knowledge. In particular, there is a polynomial-time simulator Sim that on input x generates a distribution (r, π) such that no polynomial-time distinguisher can distinguish between this and the (r, π) from the actual protocol with advantage greater than the zero-knowledge error ϵ_z. For simplicity, we assume that the NIZK protocol has perfect completeness ($\epsilon_c = 0$) and we have a deterministic verification algorithm V.

The Ostrovsky-Wigderson approach

The key observation underlying [12] is that an inverter for the NIZK simulator can be used to construct a distinguisher to decide the language, contradicting its hardness.

Suppose the language $\mathcal{L}$ that has the NIZK argument $(\mathsf{P}, \mathsf{V}, \mathsf{Sim})$ is worst-case hard. Let Sim be the simulator that runs on the input x and randomness ρ, and outputs the common reference string r and a proof π. The candidate hard-to-invert function f_x is defined to be

$$\begin{aligned} f_x(\rho): \quad & (r, \pi) \leftarrow \mathsf{Sim}(x; \rho) \\ & \text{output } r \end{aligned}$$

Since the randomness is treated explicitly as part of the input, the above construction is deterministic and therefore the function is well defined.

Towards a contradiction, assume that there are no auxiliary-input one-way functions. In fact, suppose that there is an adversary $\mathcal{A}$ that perfectly inverts f_x distributionally – that is, given y, $\mathcal{A}$ samples a uniformly random pre-image $f_x^{-1}(y)$. It is known that distributional

OWFs imply OWFs [8], so the only loss of generality here is the assumption that the inverter is perfect, but this is not difficult to remove at the cost of an inverse polynomial loss in parameters.

We have that the joint distributions

$$(\mathcal{A}(f_x(\mathcal{U})), f_x(\mathcal{U})) \approx (\mathcal{U}, f_x(\mathcal{U}))$$

are close, where the left-hand side represents the distribution of first computing f_x on random inputs and applying $\mathcal{A}$, while the right-hand side denotes the distribution of sampling a random input r and then outputting $(r, f_x(r))$.

The algorithm $\mathcal{D}$ that decides $\mathcal{L}$ is as follows. Given input x, it invokes $\mathcal{A}$ to decide whether $x \in \mathcal{L}$ or not: it samples $r \leftarrow \mathcal{U}$, runs $\hat{\rho} \leftarrow \mathcal{A}(r)$, then computes $(\hat{r}, \hat{\pi}) \leftarrow \mathsf{Sim}(x; \hat{\rho})$, and accepts if and only if $\mathsf{V}(x; r, \hat{\pi}) = 1$. Note that under our assumptions, we will always have $r = \hat{r}$. We now analyze the performance of $\mathcal{D}$ in the two possible cases.

1. When $x \in \mathcal{L}$, completeness implies that when (r, π) is generated following the protocol, $\mathsf{V}(x; r, \pi) = 1$ holds with probability 1. Zero-knowledge guarantees that the probability that $\mathsf{V}(x; r, \pi) = 1$ when $(r, \pi) \leftarrow \mathsf{Sim}(x; \mathcal{U})$ is at least $1 - \epsilon_z$.
   In the algorithm $\mathcal{D}(x)$, we run V on (r, π) generated from $\mathsf{Sim}(x; \mathcal{A}(\mathcal{U}))$. When the inverter $\mathcal{A}$ is run on r sampled from $\mathsf{Sim}(x; \mathcal{U})$, the resulting inverse is uniformly random (due to the definition of f_x). Again by zero-knowledge, the uniform distribution of r is ϵ_z-indistinguishable from the distribution of r sampled by $\mathsf{Sim}(x; \mathcal{U})$. So the distribution of $\mathcal{A}(\mathcal{U})$ is also ϵ_z-indistinguishable from uniform. This implies that the distribution of $\mathsf{Sim}(x; \mathcal{A}(\mathcal{U}))$ is ϵ_z-indistinguishable from $\mathsf{Sim}(x; \mathcal{U})$. Altogether, we have

   $$\Pr[\mathcal{D}(x) = 1] \geq 1 - 2\epsilon_z.$$

2. When $x \notin \mathcal{L}$, soundness ensures that for any efficient method of generating π for random r, the probability that V accepts is bounded by ϵ_s, and so

   $$\Pr[\mathcal{D}(x) = 1] \leq \epsilon_s.$$

Consequently, if $\epsilon_s < 1 - 2\epsilon_z$, the inverter $\mathcal{A}$ can be used to determine whether x is in $\mathcal{L}$. If such an inverter works for every x, then we can decide the language, contradicting its hardness. Thus, the family of functions $\{f_x\}$ must be a family of auxiliary-input one-way functions.

Improving the condition on errors

More recently, such analysis has seen further progress. In particular, this bound was improved by [4], where it was shown that $\sqrt{\epsilon_s} + \epsilon_z < 1$ suffices. This was shown with sophisticated arguments using a one-sided version of universal approximation, where the inverter is used to estimate the probabilities of certain outputs.

Our starting point is the observation that with rather simple but careful arguments, it is feasible to relax this bound to $\epsilon_s + \epsilon_z < 1$, which is the most general it can be. We avoid paying for the zero-knowledge error twice by involving the verification procedure inside the candidate one-way function. Our one-way function is as follows

$f_x(\rho)$: $(r, \pi) \leftarrow \mathsf{Sim}(x; \rho)$
$a \leftarrow \mathsf{V}(x; r, \pi)$
output (r, a)

Suppose again that there is a near-perfect polynomial-time inverter $\mathcal{A}$ for f_x (it need not be a distributional inverter). That is,

$$\Pr_{(r,a)\leftarrow f_x(\mathcal{U})}[f_x(\mathcal{A}(r,a)) = (r,a)] \approx 1,$$

The algorithm $\mathcal{D}$ for $\mathcal{L}$ can be constructed as follows. Given input x, sample r uniformly at random, compute $\hat{\rho} \leftarrow \mathcal{A}(r,1)$, and accept if and only if this is a valid pre-image of $(r,1)$ under f_x – that is, iff $f_x(\hat{\rho}) = (r,1)$.

1. For $x \in \mathcal{L}$, consider the following procedure $g(r,\pi)$: it computes $a \leftarrow \mathsf{V}(x;r,\pi)$ and outputs (r,a). When (r,π) is sampled from the simulator $\mathsf{Sim}(x;\mathcal{U})$, the distribution of $g(r,\pi)$ is equivalent to $f_x(\mathcal{U})$. When (r,π) follows the protocol view, the distribution of $g(r,\pi)$ is the same as $(r,1)$ in $\mathcal{D}$. So by ZK and the data processing inequality, these distributions are ϵ_z-indistinguishable.
   Further, when (r,π) is sampled from the simulator, the inverter $\mathcal{A}$ will almost always find a valid pre-image of (r,a). Therefore, given $(r,1)$ as in $\mathcal{D}$, it finds a valid pre-image with overall probability at least $\approx 1-\epsilon_z$.

   $$\Pr[\mathcal{D}(x) = 1] = \Pr_{r\leftarrow\mathcal{U}}[f_x(\mathcal{A}(r,1)) = (r,1)] \geq 1-\epsilon_z.$$

2. For $x \notin \mathcal{L}$, whenever a valid pre-image $\hat{\rho}$ for $(r,1)$ is found, the corresponding $(r,\pi) \leftarrow \mathsf{Sim}(x;\hat{\rho})$ is accepted by V. As both $\mathcal{A}$ and Sim are efficient algorithms, soundness guarantees that a valid inverse cannot be found with probability more than ϵ_s. So

   $$\Pr[\mathcal{D}(x) = 1] \leq \epsilon_s.$$

Therefore, following the same remaining arguments as earlier, $\epsilon_s + \epsilon_z < 1$ suffices to imply the existence of auxiliary-input one-way functions.

The key idea behind our improvement is that we implicitly utilize the verification V while restricting the inverter to output a good pre-image corresponding to a valid proof. Thus as long as the inverter succeeds, there is no need to perform an additional explicit verification. This saves us the extra zero-knowledge error penalty.

The distinguisher: an alternate view

Our analysis reveals that in both the [12] construction and our new one, these distinguishers can be regarded as providing efficient simulations of the protocol, where the original honest prover algorithm is replaced with an efficient algorithm without the witness and V serves to certify correctness. For example, the Ostrovsky-Wigderson distinguisher admits an equivalent formulation as a protocol between a prover $\tilde{\mathsf{P}}$ and the verifier V. The prover $\tilde{\mathsf{P}}$ performs: on randomness r, find the message $\tilde{\pi}$ corresponding to the simulator randomness $\tilde{\rho}$, where $\tilde{\rho}$ is found efficiently by the inverter. Our construction yields a similar prover as well. The only difference is the modification to the inverter since the correctness of a pre-image ensures the validity of corresponding proof.

Multiple-round public-coin zero-knowledge

Our approach naturally extends to the setting of multiple-round public-coin zero-knowledge protocols. Such a protocol allows interaction between the prover and the verifier, where all of the verifier's communication consists of uniform random bits. In particular, all its randomness is public and available to the prover.

For the sake of exposition, we assume below perfect completeness with soundness error ϵ_s and zero-knowledge error ϵ_z. By adapting our approach for the NIZK case, we obtain that the existence a k-round public-coin ZK protocol for a hard language $\mathcal{L}$ with parameters $\epsilon_s + (2k-1)\epsilon_z < 1$ implies the existence of one-way functions. We use the term *round* to denote one interaction where the verifier sends a random challenge and the prover responds with a proof message.

Moreover, the bound can be further improved to $\epsilon_s + k \cdot \epsilon_z < 1$ using a more sophisticated argument when k is only a constant.

For illustration, we focus here on the simplest setting in which the language $\mathcal{L}$ admits a two-round public-coin ZK protocol. Here both parties share a common input x while the honest prover additionally holds a witness w. In the first round, the verifier first samples a random string r_1 and the prover replies with a message $\pi_1 \leftarrow \mathsf{P}_1(x, w; r_1)$. In the second round, the verifier sends another random string r_2 and the prover responds with $\pi_2 \leftarrow \mathsf{P}(x, w; r_1, \pi_1, r_2)$. Finally, the verifier checks the transcript by computing $0/1 \leftarrow \mathsf{V}(x; r_1, \pi_1, r_2, \pi_2)$, where 1 denotes acceptance.

The zero-knowledge condition implies the existence of a simulator Sim that on input x and randomness ρ, outputs a transcript (r_1, π_1, r_2, π_2); for any $x \in \mathcal{L}$ and a valid witness w, $\mathsf{Sim}(x)$ is ϵ_z-indistinguishable from the protocol view.

In the following, we will follow a recursive approach to construct our candidate one-way functions. More precisely, we begin by defining a function, and any algorithm that breaks the one-wayness of this function will be incorporated into the construction of a second function. If the second one is not one-way either, then the resulting inverters can be combined to decide the language. Accordingly, we first define a function $f_{2,x}$ as follows:

$$\begin{aligned} f_{2,x}(\rho): \quad & (r_1, \pi_1, r_2, \pi_2) \leftarrow \mathsf{Sim}(x; \rho) \\ & a \leftarrow \mathsf{V}(x; r_1, \pi_1, r_2, \pi_2) \\ & \text{output } (r_1, \pi_1, r_2, a) \end{aligned}$$

Assume that there is a poly-time algorithm $\mathcal{A}_2$ that inverts $f_{2,x}$

$$\Pr_{(r_1,\pi_1,r_2,a)\leftarrow f_{2,x}(\mathcal{U})} [f_{2,x}(\mathcal{A}_2(x; r_1, \pi_1, r_2, a)) = (r_1, \pi_1, r_2, a)] \approx 1. \tag{1}$$

As in the NIZK case, the deciding procedure is essentially equivalent to efficiently simulating the protocol by constructing an efficient prover $\tilde{\mathsf{P}}$: this queries the inverter $\mathcal{A}$ on input $(r, 1)$ and generates the prover's message π accordingly. For the second round of the protocol, we define the strategy $\tilde{\mathsf{P}}_2$ similarly.

$$\begin{aligned} \tilde{\mathsf{P}}_2(x; r_1, \pi_1, r_2): \quad & \tilde{\rho} \leftarrow \mathcal{A}_2(x; r_1, \pi_1, r_2, 1) \\ & (\tilde{r}_1, \tilde{\pi}_1, \tilde{r}_2, \tilde{\pi}_2) \leftarrow \mathsf{Sim}(x; \tilde{\rho}) \\ & \text{output } \tilde{\pi}_2 \end{aligned}$$

We measure the performance of $\tilde{\mathsf{P}}_2$ formally by the following quantity:

$$\mathsf{Succ}_3(x; r_1, \pi_1, r_2) = 1[f_{2,x}(\mathcal{A}_2(x; r_1, \pi_1, r_2, 1)) = (r_1, \pi_1, r_2, 1)]$$

where $\mathcal{A}_2$ is assumed to be deterministic for simplicity. The value of Succ_3 represents the probability that $\mathcal{A}_2$ finds a valid pre-image with respect to $f_{2,x}$. This provides a lower bound for the acceptance probability of the protocol conditioned on the first 3 messages being (r_1, π_1, r_2), since $\tilde{\pi}_2$ is valid as long as $\mathcal{A}_2$ finds a valid pre-image of $f_{2,x}$.

$$f_{2,x}(\tilde{\rho}) = (r_1, \pi_1, r_2, 1) \Rightarrow \mathsf{V}(x; r_1, \pi_1, r_2, \tilde{\pi}_2) = 1.$$

Since the verifier samples r_2 uniformly, we can equivalently lower bound the success probability of this prover by the following expression (where the first two messages are (r_1, π_1) and $\tilde{\mathsf{P}}_2$ is the second round strategy of the prover):

$$\mathsf{Succ}_2(x; r_1, \pi_1) = \mathop{\mathbb{E}}_{r_2 \leftarrow \mathcal{U}} [\mathsf{Succ}_3(x; r_1, \pi_1, r_2)]$$

Now consider the protocol $\langle \mathsf{P}, \mathsf{V} \rangle$ that replaces the prover algorithm in the second round with $\tilde{\mathsf{P}}_2$. We can establish an upper bound for the completeness error of this modified protocol.

Denote by $D_{2,x}$ the distribution that samples (r_1, π_1, r_2, π_2) from the protocol $\langle \mathsf{P}, \mathsf{V} \rangle(x, w)$, sets $a \leftarrow \mathsf{V}(x; r_1, \pi_1, r_2, \pi_2)$ and outputs (r_1, π_1, r_2, a). By the perfect completeness of $\langle \mathsf{P}_w, \mathsf{V} \rangle$, $a = 1$ always holds. It follows by the data processing inequality that, for $x \in \mathcal{L}$

$$\Delta(f_{2,x}(\mathcal{U}); D_{2,x}) \leq \Delta(\mathsf{Sim}(x); \langle \mathsf{P}, \mathsf{V} \rangle(x)) \leq \epsilon_z.$$

Combining above with our assumption (1), the acceptance probability is at least

$$\begin{aligned}
&\mathop{\mathbb{E}}_{\substack{r_1 \leftarrow \mathcal{U} \\ \pi_1 \leftarrow \mathsf{P}_1(x,w;r_1)}} [\mathsf{Succ}_2(x; r_1, \pi_1)] \\
&= \Pr_{(r_1, \pi_1, r_2, a) \leftarrow D_{2,x}} [f_{2,x}(\mathcal{A}_2(x; r_1, \pi_1, r_2, a)) = (r_1, \pi_1, r_2, a)] \\
&\geq \Pr_{(r_1, \pi_1, r_2, a) \leftarrow f_{2,x}(\mathcal{U})} [f_{2,x}(\mathcal{A}_2(x; r_1, \pi_1, r_2, a)) = (r_1, \pi_1, r_2, a)] - \epsilon_z \\
&\geq 1 - \epsilon_z. \qquad (2)
\end{aligned}$$

The above implies that when the prover applies $\mathsf{P}_1(x, w)$ and $\tilde{\mathsf{P}}_2(x)$ in each round, respectively, the acceptance probability is at least $1 - \epsilon_z$ when $x \in \mathcal{L}$. However, because the witness for P_1 is unavailable to our deciding strategy, we need a different polynomial-time algorithm as a replacement to obtain an efficient simulation for the protocol.

We now turn to building our first-round strategy. A natural attempt is to define a new function $f_{1,x}(\rho)$ analogously to $f_{2,x}$ and apply the inverter to produce the message π_1, where $f_{1,x}(\rho)$ can be defined as follows: sample $(r_1, \pi_1, r_2, \pi_2) \leftarrow \mathsf{Sim}(x; \rho)$, $a \leftarrow \mathsf{V}(x; r_1, \pi_1, r_2, \pi_2)$, and output (r_1, a) (or only output r_1). However, our analysis of the resulting prover algorithms shows that the best achievable result is $\epsilon_s + 3\epsilon_z < 1$, which matches $\epsilon_s + (2k-1)\epsilon_z < 1$ for $k = 2$. We refer the reader to Section 3.2 for the details.

Instead, we introduce a new construction of $f_{1,x}$ that leads to an improved bound. After fixing $\tilde{\mathsf{P}}_2$, our objective is to find an efficient way $\tilde{\mathsf{P}}_1$ to generate π_1 that maximizes the value $\mathsf{Succ}_2(x; r_1, \pi_1)$ optimally for $x \in \mathcal{L}$. The value of Succ_2 can be efficiently approximated up to any arbitrary inverse-polynomial error due to our assumption on $\mathcal{A}_2$ and standard concentration arguments. For convenience, we ignore the accuracy issue here and assume that we are able to compute Succ_2 precisely in polynomial-time. The key is to instead have the new function compute Succ_2 for the relevant partial transcript. More precisely, we define:

$$\begin{aligned}
f_{1,x}(\rho): \quad & (r_1, \pi_1, r_2, \pi_2) \leftarrow \mathsf{Sim}(x; \rho) \\
& \text{output } (r_1, \mathsf{Succ}_2(x; r_1, \pi_1))
\end{aligned}$$

Suppose now that there is an efficient algorithm $\mathcal{A}_1$

$$\Pr_{(r_1, a) \leftarrow f_{1,x}(\mathcal{U})} [f_{1,x}(\mathcal{A}_1(x; r_1, a)) = (r_1, a)] \approx 1.$$

We are now ready to formally specify $\tilde{\mathsf{P}}_1$, which works as follows:

$$\begin{aligned}\tilde{\mathsf{P}}_1(x;r_1):\quad &\tilde{a}\leftarrow \arg\max_a \{a\cdot \mathbf{1}[f_{1,x}(\mathcal{A}_1(x;r_1,a))=(r_1,a)\}]\\ &\tilde{\rho}\leftarrow \mathcal{A}_1(x;r_1,\tilde{a})\\ &\tilde{\pi}_1\leftarrow \mathsf{Sim}(\tilde{\rho})\\ &\text{output } \tilde{\pi}_1\end{aligned}$$

where we take the support size of a to be polynomial, enabling us to efficiently iterate over all possible values and find the maximum $\tilde{a}$. In fact, in general one can efficiently find an approximate maximum value instead, which suffices.

On input $(x;r_1)$, the prover $\tilde{\mathsf{P}}_1$ goes through all possible values of a, and selects the highest one on which $\mathcal{A}_1$ inverts $f_{1,x}$ successfully. Note that when $\mathcal{A}_1$ finds a correct pre-image of (r_1,a), it implies that there exists a consistent π_1 with value $\mathsf{Succ}_2(x;r_1,\pi_1)=a$; so $\tilde{\mathsf{P}}_1$ takes the maximal a and outputs its associated prover message.

With the construction of $\tilde{\mathsf{P}}_1$ and $\tilde{\mathsf{P}}_2$, it remains to analyze the performance of the resulting protocol $\langle\tilde{\mathsf{P}},\mathsf{V}\rangle$. More specifically, we are interested in

$$\mathop{\mathbb{E}}_{\substack{r_1\leftarrow\mathcal{U},\pi_1\leftarrow\tilde{\mathsf{P}}_1(x;r_1)\\ r_2\leftarrow\mathcal{U},\pi_2\leftarrow\tilde{\mathsf{P}}_2(x;r_1,\pi_1,r_2)}}[\mathsf{V}(x;r_1,\pi_1,r_2,\pi_2)] \tag{3}$$

Recall that

$$(3)\geq \mathop{\mathbb{E}}_{\substack{r_1\leftarrow\mathcal{U}\\ \pi_1\leftarrow\tilde{\mathsf{P}}_1(x;r_1)}}[\mathsf{Succ}_2(x;r_1,\pi_1)] = \mathop{\mathbb{E}}_{r_1\leftarrow\mathcal{U}}\left[\max_a\{a\cdot\mathbf{1}[f_{1,x}(\mathcal{A}_1(x;r_1,a))=(r_1,a)]\}\right] \tag{4}$$

where the last equality is ensured since by definition of $\tilde{\mathsf{P}}_1(x;r_1)$, the Succ_2 estimate a associated with π_1 is the largest such value with respect to which $\mathcal{A}_1$ can succeed, i.e.,

$$\mathsf{Succ}_2(x;r_1,\pi_1)=\max_a\{a\cdot\mathbf{1}[f_{1,x}(\mathcal{A}_1(x;r_1,a))=(r_1,a)]\}.$$

Now observe that, for any r_1 and any particular a^*

$$\max_a\{a\cdot\mathbf{1}[f_{1,x}(\mathcal{A}_1(x;r_1,a))=(r_1,a)]\}\geq a^*\cdot\mathbf{1}[f_{1,x}(\mathcal{A}_1(x;r_1,a^*))=(r_1,a^*)]. \tag{5}$$

Let a^* above be sampled as $a^*\leftarrow\mathsf{Succ}_2(x;r_1,\pi_1)$ where $\pi_1\leftarrow\mathsf{P}_1(x,w;r_1)$. We now investigate the new completeness error to obtain a lower bound for (4) when $x\in\mathcal{L}$. Note that the value of Succ_2 falls in the range $[0,1]$, thus we have

$$\begin{aligned}(4)&\geq \mathop{\mathbb{E}}_{\substack{r_1\leftarrow\mathcal{U},\pi_1\leftarrow\mathsf{P}_1(x,w;r_1)\\ a\leftarrow\mathsf{Succ}_2(x;r_1,\pi_1)}}[a\cdot\mathbf{1}[f_{1,x}(\mathcal{A}_1(x;r_1,a))=(r_1,a)]]\\ &\geq \mathop{\mathbb{E}}_{\substack{r_1\leftarrow\mathcal{U},\pi_1\leftarrow\mathsf{P}_1(x,w;r_1)\\ a\leftarrow\mathsf{Succ}_2(x;r_1,\pi_1)}}[a]-\mathop{\mathbb{E}}_{\substack{r_1\leftarrow\mathcal{U},\pi_1\leftarrow\mathsf{P}_1(x,w;r_1)\\ a\leftarrow\mathsf{Succ}_2(x;r_1,\pi_1)}}[\mathbf{1}[f_{1,x}(\mathcal{A}_1(x;r_1,a))\neq(r_1,a)]]\\ &\geq 1-\epsilon_z-\Pr_{\substack{r_1\leftarrow\mathcal{U},\pi_1\leftarrow\mathsf{P}_1(x,w;r_1)\\ a\leftarrow\mathsf{Succ}_2(x;r_1,\pi_1)}}[f_{1,x}(\mathcal{A}_1(x;r_1,a))\neq(r_1,a)],\end{aligned} \tag{6}$$

where the first inequality is obtained by the observation (5) and the last inequality holds by (2). Now define the distribution $D_{1,x}$: sample $r_1\leftarrow\mathcal{U}$ and $\pi_1\leftarrow\mathsf{P}_1(x,w;r_1)$, and finally output $(r_1,\mathsf{Succ}_2(x;r_1,\pi_1))$.

For $x\in\mathcal{L}$, by the data processing inequality and zero-knowledge condition

$$\Delta(f_{1,x}(\mathcal{U});D_{1,x})\leq\Delta(\mathsf{Sim}(x);\langle\mathsf{P}_w,\mathsf{V}\rangle(x))\leq\epsilon_z.$$

Since we assume that $\mathcal{A}_1$ inverts the function almost perfectly, we get

$$\begin{aligned}
&\Pr_{\substack{r_1\leftarrow\mathcal{U},\pi_1\leftarrow\mathsf{P}_1(x,w;r_1)\\ a\leftarrow\mathsf{Succ}_2(x;r_1,\pi_1)}}[f_{1,x}(\mathcal{A}_1(x;r_1,a))\neq(r_1,a)]\\
&\leq \Pr_{(r_1,a)\leftarrow f_{1,x}(\mathcal{U})}[f_{1,x}(\mathcal{A}_1(x;r_1,a))\neq(r_1,a)]+\epsilon_z\\
&\leq \epsilon_z. \qquad (7)
\end{aligned}$$

Therefore, by combining (3), (6) and (7), we derive that for $x\in\mathcal{L}$

$$\mathop{\mathbb{E}}_{\substack{r_1\leftarrow\mathcal{U},\pi_1\leftarrow\tilde{\mathsf{P}}_1(x;r_1)\\ r_2\leftarrow\mathcal{U},\pi_2\leftarrow\tilde{\mathsf{P}}_2(x;r_1,\pi_1,r_2)}}[\mathsf{V}(x;r_1,\pi_1,r_2,\pi_2)]\geq 1-2\epsilon_z.$$

On the other hand, soundness of the original protocol $\langle\mathsf{P}_w,\mathsf{V}\rangle$ ensures that, for $x\notin\mathcal{L}$

$$\mathop{\mathbb{E}}_{\substack{r_1\leftarrow\mathcal{U},\pi_1\leftarrow\tilde{\mathsf{P}}_1(x;r_1)\\ r_2\leftarrow\mathcal{U},\pi_2\leftarrow\tilde{\mathsf{P}}_2(x;r_1,\pi_1,r_2)}}[\mathsf{V}(x;r_1,\pi_1,r_2,\pi_2)]\leq \epsilon_s.$$

When $\epsilon_s+2\epsilon_z$ is noticeably less than 1, we thus obtain an efficient algorithm that decides if $x\in\mathcal{L}$, which yields the desired result.

One catch here is that while this approach is conceptually complete, it only gives us a construction of an infinitely often one-way function. This is because in our approach, we will require that the adversaries $\mathcal{A}_1$ and $\mathcal{A}_2$ must both succeed in their inversion so that we successfully decide on an instance. This means that the sequence of input lengths on which our assumed inverters work must overlap when we aim for a contradiction - and the formal negation for this only implies infinitely-often hardness. See Section 4.2.2 and Remark 42 for details.

2 Preliminaries

Denote by $\mathcal{U}_\ell$ the uniform distribution over the length-ℓ binary strings $\{0,1\}^\ell$. For a language $\mathcal{L}$, let $\mathcal{L}_n=\mathcal{L}\cap\{0,1\}^n$ be the set of all the length-n strings in the language. For $k\in\mathbb{N}$, denote $[k]=\{1,\ldots,k\}$. We define a distribution ensemble $\mathcal{X}=\{X_n\}_{n\in\mathbb{N}}$ to be a collection of distributions where each X_n is defined over $\{0,1\}^{m(n)}$ where $m:\mathbb{N}\to\mathbb{N}$ is some arithmetic function. We say a function ν is negligible if for every polynomial p there exists an $n_0\in\mathbb{N}$ such that for all $n\geq n_0$, $\mu(n)<\frac{1}{p(n)}$. Similarly, we call a function μ noticeable if there exists a polynomial p and $n_0\in\mathbb{N}$ such that for all $n\geq n_0$, $\mu(n)\geq\frac{1}{p(n)}$.

For $\epsilon_1=\epsilon_1(n),\epsilon_2=\epsilon_2(n)$, we say $\epsilon_1<_n\epsilon_2$ (or $\epsilon_2>_n\epsilon_1$) to represent the noticeable gap between ϵ_1 and ϵ_2, if there exists a polynomial p such that for all sufficiently large $n\in\mathbb{N}$, we have $\epsilon_1(n)+\frac{1}{p(n)}<\epsilon_2(n)$, i.e. there is an inverse-polynomial gap between ϵ_1 and ϵ_2.

We use the Hoeffding bound, stated as follows.

▶ **Lemma 5** (Hoeffding's inequality). *For independent random variables $X_1,\ldots,X_q$ with support $[0,1]$, let $X=\sum_{i\in[q]}X_i$, the following holds*

$$\Pr\left[|X-\mathbb{E}X|>t\cdot q\right]\leq 2e^{-2t^2q}.$$

2.1 Indistinguishability

Assume that we have two distributions X and Y defined over a common universe U. We first consider statistical indistinguishability.

▶ **Definition 6** (Statistical Distance). *The* statistical distance *between distributions X and Y (defined over the support of X, denoted by $\mathsf{Supp}(X)$) is defined as*

$$\Delta_s(X;Y) = \frac{1}{2} \sum_{u \in \mathsf{Supp}(X)} \big|\Pr[X=u] - \Pr[Y=u]\big|.$$

The following properties hold.

▶ **Lemma 7** (Triangle Inequality). *For any three distributions X, Y and Z, we have $\Delta_s(X;Z) \le \Delta_s(X;Y) + \Delta_s(Y;Z)$.*

▶ **Lemma 8** (Data Processing Inequality). *For any two probability distributions X, Y (on a common universe U), and any (possibly randomized) process f, we have $\Delta_s(f(X);f(Y)) \le \Delta_s(X;Y)$.*

Suppose that we have a given distinguishing algorithm D to distinguish between X and Y, taking inputs in U and outputting a bit to indicate whether it identifies a given input as being sampled from X or Y. Define the *distinguishing advantage* $\mathsf{Adv}_D(X,Y)$ of D as:

$$\mathsf{Adv}_D(X;Y) = \left| \Pr_{x \leftarrow X}[D(x)=1] - \Pr_{y \leftarrow Y}[D(y)=1] \right|.$$

In fact, the statistical distance implicitly provides an upper bound on the advantage that any distinguisher can obtain, that is $\Delta_s(X;Y) = \max_D \mathsf{Adv}_D(X;Y)$, where D can be any possible algorithm. Usually, for a constant ϵ, when $\Delta_s(X;Y) < \epsilon$, X and Y are said to be ϵ-statistically indistinguishable. Next, we formally define the statistical indistinguishability asymptotically.

▶ **Definition 9** (Statistical Indistinguishability). *Consider a function $\epsilon : \mathbb{N} \to [0,1]$, and distribution ensembles $\mathcal{X} = \{X_n\}_{n\in\mathbb{N}}$ and $\mathcal{Y} = \{Y_n\}_{n\in\mathbb{N}}$. We say that $\mathcal{X}$ and $\mathcal{Y}$ are* ϵ-statistically indistinguishable *(or have statistical distance at most ϵ), denoted by $\Delta_s(\mathcal{X};\mathcal{Y}) \le \epsilon$, if for all $n \in \mathbb{N}$, we have $\Delta_s(X_n;Y_n) \le \epsilon(n)$.*

Notice that we do not impose any computational constraint on the distinguisher above, thus the statistical notion implies that even computationally unbounded algorithms cannot achieve an advantage better than ϵ. It is also natural to restrict attention to polynomial-time algorithms. Next, we define computational indistinguishability.

▶ **Definition 10** (Computational Indistinguishability). *Consider a function $\epsilon : \mathbb{N} \to [0,1]$, and distribution ensembles $\mathcal{X} = \{X_n\}_{n\in\mathbb{N}}$ and $\mathcal{Y} = \{Y_n\}_{n\in\mathbb{N}}$. If for every non-uniform probabilistic polynomial-time algorithm D, there is an $n_D \in \mathbb{N}$ such that for all $n \ge n_D$ we have $\mathsf{Adv}_D(X_n;Y_n) \le \epsilon(n)$, then we say that $\mathcal{X}$ and $\mathcal{Y}$ are* ϵ-computationally indistinguishable *(with respect to polynomial-time algorithms). We denote this by $\Delta_c(\mathcal{X};\mathcal{Y}) \le \epsilon$.*

In the course of our technical arguments, for the sake of simplicity, we often make statements of the form $\Delta_c(X_n;Y_n) \le \epsilon(n)$. These statements and their implications are to be interpreted in the asymptotic sense, as holding for all large enough n rather than for all n.

The definition ensures that $\Delta_c(\mathcal{X};\mathcal{Y}) \le \Delta_s(\mathcal{X};\mathcal{Y})$. Versions of the triangle inequality and data processing inequality hold computational indistinguishability as well. The former is easily implied by essentially a hybrid argument. We state it formally as follows.

▶ **Lemma 11** (Triangle Inequality). *For any three distribution ensembles $\mathcal{X}$, $\mathcal{Y}$, $\mathcal{Z}$, we have*

$$\Delta_c(\mathcal{X};\mathcal{Y}) \le \epsilon_1, \Delta_c(\mathcal{Y};\mathcal{Z}) \le \epsilon_2 \Rightarrow \Delta_c(\mathcal{X};\mathcal{Z}) \le \epsilon_1 + \epsilon_2.$$

▶ **Lemma 12** (Data Processing Inequality). *For any distribution ensembles $\mathcal{X}, \mathcal{Y}$ and any probabilistic polynomial-time procedure f, we have*

$$\Delta_c(\mathcal{X}; \mathcal{Y}) \leq \epsilon \Rightarrow \Delta_c(f(\mathcal{X}); f(\mathcal{Y})) \leq \epsilon.$$

This is a simple consequence of the observation that any such efficient function f can simply be run on top of samples from $\mathcal{X}$ or $\mathcal{Y}$ and then fed into a distinguisher for $f(\mathcal{X})$ and $f(\mathcal{Y})$.

We slightly abuse the notation by writing $\Delta_c(\mathcal{X}; \mathcal{Z}) \leq \Delta_c(\mathcal{X}; \mathcal{Y}) + \Delta_c(\mathcal{Y}; \mathcal{Z})$ and $\Delta_c(f(\mathcal{X}); f(\mathcal{Y})) \leq \Delta_c(\mathcal{X}; \mathcal{Y})$, by which we mean the properties defined above.

2.2 Circuits and Oracles

We define notions of circuits and functions computed with respect to certain oracles.

▶ **Definition 13** (Oracle-Aided Circuits and Algorithms). *Let $\mathcal{O} : \{0,1\}^* \to \{0,1\}^*$ be an oracle (an arbitrary function). An oracle circuit (or algorithm) C with respect to $\mathcal{O}$, denoted by $C^{\mathcal{O}}$ is a circuit (or algorithm) where in addition to standard operations, C also has oracle gates (or oracle operations) where it can make a query to $\mathcal{O}$, and expect its output as response.*

▶ **Definition 14** (Oracle-Aided Functions). *Let $\mathcal{O} : \{0,1\}^* \to \{0,1\}^*$ be an oracle (an arbitrary function). An oracle aided function f with respect to $\mathcal{O}$, denoted $f^{\mathcal{O}}$, is a function computable by a deterministic oracle-aided algorithm $A^{\mathcal{O}}$.*

▶ Remark 15. When $\mathcal{O}$ is a randomized algorithm, we view the oracle gates for $\mathcal{O}$ as deterministic ones that take an additional randomness as input.

2.3 One-Way Functions

In the following, we present the definition of one-way functions, along with several weaker variants, which will serve as intermediate steps in our later proofs.

▶ **Definition 16** (One-Way Function, OWF). *For m_1, m_2 being polynomials, a function family $\mathcal{F} = \{f_n : \{0,1\}^{m_1(n)} \to \{0,1\}^{m_2(n)}\}_{n\in\mathbb{N}}$ is said to be a* One-Way Function (OWF) *if $\mathcal{F}$ is efficiently computable and for every non-uniform PPT algorithm $\mathcal{A}$ there is a negligible function $\nu(\cdot)$ such that for all large enough $n \in \mathbb{N}$,*

$$\Pr_{x \leftarrow \mathcal{U}_{m_1(n)}} \left[f_n(\mathcal{A}(f_n(x))) = f_n(x) \right] \leq \nu(n).$$

▶ **Definition 17** (Weak One-Way Function). *For m_1, m_2 being polynomials, a function family $\mathcal{F} = \{f_n : \{0,1\}^{m_1(n)} \to \{0,1\}^{m_2(n)}\}_{n\in\mathbb{N}}$ is said to be a* Weak One-Way Function *if $\mathcal{F}$ is efficiently computable and there is a polynomial p such that for every non-uniform PPT algorithm $\mathcal{A}$, for all large enough $n \in \mathbb{N}$,*

$$\Pr_{x \leftarrow \mathcal{U}_{m_1(n)}} \left[f_n(\mathcal{A}(f_n(x))) = f_n(x) \right] \leq 1 - \frac{1}{p(n)}.$$

▶ **Definition 18** (Distributional One-Way Function, dOWF). *For m_1, m_2 being polynomials, a function family $\mathcal{F} = \{f_n : \{0,1\}^{m_1(n)} \to \{0,1\}^{m_2(n)}\}_{n\in\mathbb{N}}$ is a* Distributional One-Way Function (dOWF) *if $\mathcal{F}$ is efficiently computable and there is a polynomial p such that for every non-uniform PPT algorithm $\mathcal{A}$, for all large enough n, $\Delta_s(X_n; Y_n) > \frac{1}{p(n)}$ holds where X_n and Y_n are defined as:*

1. $X_n : \{(x, f(x)) : x \leftarrow \mathcal{U}_{m_1(n)}\}$
2. $Y_n : \{(\mathcal{A}(f(x)), f(x)) : x \leftarrow \mathcal{U}_{m_1(n)}\}$

▶ Remark 19. In our arguments, we will consider adversaries against distributional one-way functions. We will refer to such an adversary $\mathcal{A}$ as inverting the function in a distributional sense (or distributionally inverting it as shorthand), with deviation say $\frac{1}{q(n)}$ (where $q(\cdot)$ is a polynomial), to mean that $\Delta_s\big((x, f(x)); (\mathcal{A}(f(x), f(x)))\big) \leq \frac{1}{q(n)}$ for uniformly sampled $x \leftarrow \mathcal{U}_{m_1(n)}$.

▶ **Definition 20** (Auxiliary-Input One-Way Functions, ai-OWF). *For m_1, m_2 being polynomials, a function family $\mathcal{F} = \{f_a : \{0,1\}^{m_1(|a|)} \to \{0,1\}^{m_2(|a|)}\}_{a\in\{0,1\}^*}$ is said to be an* Auxiliary-Input One-Way Function (ai-OWF) *if $\mathcal{F}$ is efficiently computable and for every non-uniform PPT machine $\mathcal{A}$ there is a negligible function $\nu(\cdot)$ such that for all large enough $n \in \mathbb{N}$, there exists some $a \in \{0,1\}^n$ such that we have*

$$\Pr_{x\leftarrow\mathcal{U}_{m_1(n)}}\big[f_a(A(a, f_a(x))) = f_a(x)\big] \leq \nu(n).$$

▶ Remark 21. If in the above definition the string a (for a given n) is always fixed to be 0^n, then the definition collapses to that of a standard one-way function.

▶ Remark 22. Similar to the above variant of (standard) OWFs, we can also extend the definitions of weak and distributional one-way functions to the auxiliary input setting in the natural manner.

▶ **Definition 23** (Infinitely-Often One-Way Functions, ioOWF). *For m_1, m_2 being polynomials, a function family $\mathcal{F} = \{f_n : \{0,1\}^{m_1(n)} \to \{0,1\}^{m_2(n)}\}$ is an* Infinitely Often One-Way Function (ioOWF) *if $\mathcal{F}$ is efficiently computable and if for every non-uniform PPT algorithm A there is a negligible function $\nu(\cdot)$ and an infinite set $S_A \subseteq \mathbb{N}$ such that we have*

$$\Pr_{r\leftarrow\{0,1\}^{m_1(n)}}\big[f_n(A(f_n(r))) = f_n(r)\big] \leq \nu(n)$$

for all $n \in S_A$.

▶ Remark 24. When the properties of any of the primitives above hold only for infinitely many $n \in \mathbb{N}$, we refer to them as infinitely-often versions. Similarly, infinitely-often versions of complexity classes also be defined – e.g., ioP is the set of languages that have deterministic polynomial-time algorithms that are correct on some infinite set of input lengths.

▶ Remark 25. Similar to above, we can also define weak and distributional infinitely often one-way functions with straightforward modifications.

▶ **Lemma 26** ([8]). *There is an explicit, efficient transformation from any distributional one-way function to a standard one-way function.*

▶ **Lemma 27** ([13]). *There is an explicit, efficient transformation from any weak one-way function to a standard one-way function.*

▶ Remark 28. Both results cited above work as stated for converting (infinitely-often or auxiliary-input) weak or distributional one-way functions to (infinitely-often or auxiliary-input) one-way functions.

2.4 Zero-Knowledge Protocols

In this section, we formally define zero-knowledge protocols, specifically *non-interactive zero-knowledge* (NIZK) and *public-coin zero-knowledge* proofs and arguments.

2.4.1 Non-Interactive Zero-Knowledge

We describe the non-interactive zero-knowledge proofs or arguments in the *common reference string* model. In particular, there is no interaction between the prover and the verifier. Both parties refer to a common reference string r, which is randomly sampled, to run the protocol. For an NP language $\mathcal{L}$, on input x, the honest prover holds the NP witness w and generates a message $\pi \leftarrow \mathsf{P}(x, w; r)$; then the verifier computes $0/1 \leftarrow \mathsf{V}(x; r, \pi)$, where by convention, 1 denotes the acceptance of the proof.

▶ **Definition 29** (Non-Interactive Zero-Knowledge, NIZK). *For $\epsilon_c, \epsilon_s, \epsilon_z : \mathbb{N} \to [0, 1]$ and a language $\mathcal{L} \in \mathsf{NP}$, an $(\epsilon_c, \epsilon_s, \epsilon_z)$-*Non-Interactive Zero-Knowledge (NIZK) proof *for $\mathcal{L}$ consists of algorithms $(\mathsf{Gen}, \mathsf{P}, \mathsf{V})$, where Gen samples the* common reference string *in polynomial time, V is a deterministic polynomial-time verifier and P is a computationally unbounded prover. Let n be the length of the input x and $\mathcal{R}_\mathcal{L}$ denote the corresponding NP relation. The protocol should satisfy the following properties.*

1. Completeness. *For any $x \in \mathcal{L}_n$ and any witness w such that $(x, w) \in \mathcal{R}_\mathcal{L}$*

$$\Pr_{\substack{r \leftarrow \mathsf{Gen}(1^n) \\ \pi \leftarrow \mathsf{P}(x,w;r)}} [\mathsf{V}(x; r, \pi) = 1] \geq 1 - \epsilon_c(n).$$

2. Soundness. *For any $x \in \{0, 1\}^n \setminus \mathcal{L}_n$ and any prover algorithm P^*, the following holds*

$$\Pr_{\substack{r \leftarrow \mathsf{Gen}(1^n) \\ \pi^* \leftarrow \mathsf{P}^*(x;r)}} [\mathsf{V}(x; r, \pi^*) = 1] \leq \epsilon_s(n),$$

3. Computational Zero-Knowledge. *There exists a probabilistic polynomial-time simulator Sim such that for any $x \in \mathcal{L}_n$ and $(x, w) \in \mathcal{R}_\mathcal{L}$*

$$\Delta_c(\mathsf{Sim}(x); \mathsf{View}\langle \mathsf{P}, \mathsf{V}\rangle(x, w)) \leq \epsilon_z(n),$$

where the transcript $\mathsf{View}\langle \mathsf{P}, \mathsf{V}\rangle(x, w)$ represents the view of the verifier in the protocol with input x and witness w given to the prover, consisting of the common reference string r and the prover's message π.

If the honest prover P is constrained to be computationally efficient, and the soundness condition is required to hold only against computationally efficient provers P^*, the protocol is called an NIZK argument.

▶ Remark 30. The simulator $\mathsf{Sim}(x)$ is randomized on input x. When we need a deterministic description, we explicitly expose the random coins and include them as part of the input, which is written as $\mathsf{Sim}(x; \rho)$. Equivalently, $\mathsf{Sim}(x)$ represents the distribution of $\mathsf{Sim}(x; \rho)$ when ρ is drawn uniformly randomly. For convenience, we denote by $\mathsf{Sim}_i(x)$ the simulator $\mathsf{Sim}(x)$ restricted to outputting only the i-th message. For example, in the NIZK protocols, $\mathsf{Sim}_1(x)$ only samples the marginal distribution on the common reference string r while $\mathsf{Sim}_2(x)$ simulates the distribution of π.

2.4.2 Public-Coin Zero-Knowledge

Beyond non-interactive protocols, we study the broader class of public-coin zero-knowledge arguments or proofs.

▶ **Definition 31** (Public-Coin Zero-Knowledge). *For $\epsilon_c, \epsilon_s, \epsilon_z : \mathbb{N} \to [0, 1]$ and a language $\mathcal{L}$, an $(\epsilon_c, \epsilon_s, \epsilon_z)$-*public-coin Zero-Knowledge (ZK) proof *for $\mathcal{L}$ consists of algorithms (P, V), where V is a deterministic polynomial-time verifier and P is a computationally unbounded prover. In an execution of the protocol V is given as input the instance x and P the instance x and a witness w. The protocol should satisfy the following.*

1. Syntax and notation. *In each round, first a uniformly random string r_i of pre-specified length is sampled and sent to the prover, and the prover responds with a message π_i computed as $\pi_i \leftarrow \mathsf{P}_i(x, w; r_1, \pi_1, \ldots, r_i)$. At the end, the verifier decides whether to accept the transcript $(r_1, \pi_1, \ldots, r_k, \pi_k)$ by computing $0/1 \leftarrow \mathsf{V}(x; r_1, \pi_1, \ldots, r_k, \pi_k)$. Denote by $\langle \mathsf{P}, \mathsf{V} \rangle(x, w)$ the output of the protocol on a common input x and a witness w (held only by the prover), and $\mathsf{View}\langle \mathsf{P}, \mathsf{V} \rangle(x, w)$ represents the transcript of the execution, consisting of $(r_1, \pi_1, \ldots, r_k, \pi_k)$. Here, k is the number of rounds of the protocol.*
2. Completeness. *For any $x \in \mathcal{L}_n$, for any w that $(x, w) \in \mathcal{R}_{\mathcal{L}}$, $\Pr[\langle \mathsf{P}, \mathsf{V} \rangle(x, w) = 1] \geq 1 - \epsilon_c(n)$.*
3. Soundness. *For any $x \in \{0,1\}^n \setminus \mathcal{L}_n$, for any prover P^*, $\Pr[\langle \mathsf{P}^*, \mathsf{V} \rangle(x) = 1] \leq \epsilon_s(n)$.*
4. Computational Zero-knowledge. *There exists a probabilistic polynomial-time simulator Sim, such that for any $x \in \mathcal{L}_n$ and $(x, w) \in \mathcal{R}_{\mathcal{L}}$, $\Delta_c(\mathsf{Sim}(x); \mathsf{View}\langle \mathsf{P}, \mathsf{V} \rangle(x, w)) \leq \epsilon_z(n)$. If the honest prover P is constrained to be computationally efficient, and the soundness condition is required to hold only against computationally efficient provers P^*, the protocol is called a* ZK argument.

▶ Remark 32. We often think of the public coins r_i's as being sent by the verifier to the prover. We consider the process of the verifier sending a randomness and the prover replying with a message as one round in the protocol, where each round contains two messages. For convenience in notation, when the first message in the protocol is from the prover, we sometimes pretend that there is an empty message from the verifier before that.

3 Auxiliary-Input One-Way Functions

In this section, we show that if a language has a Zero-Knowledge proof or argument with errors satisfying certain conditions, then for any instance, we can define a function such that any inverter for that function can be used to decide the membership of that instance in the language. Worst-case hardness of the language then gives us an auxiliary-input one-way function, which will be used in later sections to construct one-way functions from hard languages that have such proof systems.

We do this for Non-Interactive ZK protocols in Section 3.1 and for general public-coin ZK protocols in Section 3.2. In Section 3.3, we use additional ideas to improve the range of errors that can be used, at the cost of the proof being non-black-box, and yielding only infinitely often secure OWFs.

▶ Remark 33. In this section, we state the main lemmas for both ZK proofs and ZK arguments. To avoid excessive repetition, in the proofs of these lemmas, we only deal with the case of arguments. It may be verified that these proofs do not rely on the efficiency of the honest prover, and work nearly as is for proof systems as well.

3.1 Reductions from NIZK

▶ **Lemma 34.** *For some $\epsilon_c, \epsilon_s, \epsilon_z : \mathbb{N} \to [0, 1]$, suppose a language $\mathcal{L} \in \mathsf{NP}$ has an $(\epsilon_c, \epsilon_s, \epsilon_z)$-NIZK proof or argument (with deterministic verification). Then there exists a reduction R, which is a polynomial-time oracle-aided algorithm, and a polynomial-time computable function family $\mathcal{F} = \{f_x\}_{x \in \{0,1\}^*}$ such that, for any probabilistic polynomial-time algorithm $\mathcal{A}$, any polynomial p, and all large enough $n \in \mathbb{N}$,*

1. *For any $x \in \mathcal{L}_n$, if $\mathcal{A}$ inverts f_x with probability at least $(1 - 1/p(n))$, then*

$$\Pr\left[R^{\mathcal{A}}(x) = 1\right] \geq 1 - \epsilon_c(n) - \epsilon_z(n) - \frac{1}{p(n)}.$$

2. *For any $x \in \{0,1\}^n \setminus \mathcal{L}_n$,*

$$\Pr\left[R^{\mathcal{A}}(x) = 1\right] \leq \epsilon_s(n).$$

▶ Remark 35. Owing to space constraints, we omit this proof and several subsequent ones, and refer the readers to the full version of our paper.

▶ **Corollary 36.** *For $\epsilon_c, \epsilon_s, \epsilon_z : \mathbb{N} \to [0,1]$, suppose there is an* NP *language $\mathcal{L} \notin$* ioP/poly *that has an $(\epsilon_c, \epsilon_s, \epsilon_z)$-NIZK proof or argument with $\epsilon_c + \epsilon_s + \epsilon_z <_n 1$. Then auxiliary-input one-way functions exist.*

Proof. Let R be the oracle-aided algorithm and $\mathcal{F}$ be the function family guaranteed by Lemma 34. There exists a polynomial q such that:

$$\epsilon_c(n) + \epsilon_s(n) + \epsilon_z(n) + \frac{1}{q(n)} < 1$$

holds for all sufficiently large $n \in \mathbb{N}$. Then, let $p = 2q$. Suppose that there is a (non-uniform) PPT algorithm $\mathcal{A}$ that, for infinitely many $n \in \mathbb{N}$, for every $x \in \mathcal{L}_n$, inverts f_x with probability at least $(1 - 1/p(n))$. By Lemma 34, for every $x \in \mathcal{L}_n$,

$$\Pr\left[R^{\mathcal{A}}(x) = 1\right] \geq 1 - \epsilon_c(n) - \epsilon_z(n) - \frac{1}{p(n)};$$

and for every $x \in \{0,1\}^n \setminus \mathcal{L}_n$,

$$\Pr\left[R^{\mathcal{A}}(x) = 1\right] \leq \epsilon_s(n).$$

Since both $\mathcal{A}$ and R are polynomial-time algorithms and

$$1 - \epsilon_c(n) - \epsilon_z(n) - \frac{1}{2q(n)} >_n \epsilon_s(n),$$

it contradicts $\mathcal{L} \notin$ ioP/poly. Hence, $\mathcal{F}$ is a weak auxiliary-input one-way function: for any efficient non-uniform adversary, for all sufficiently large n, there exists a function f_x, $x \in \{0,1\}^n$ on which the adversary cannot successfully invert f_x with probability at least $(1 - 1/p(n))$. By [13], the existence of weak ai-OWFs implies the existence of (standard) ai-OWFs, which concludes the proof. ◀

3.2 Reductions from Public-Coin ZK

▶ **Lemma 37.** *For some $\epsilon_c, \epsilon_s, \epsilon_z : \mathbb{N} \to [0,1]$, $t : \mathbb{N} \to \mathbb{N}$, suppose a language $\mathcal{L} \in$* NP *has an $(\epsilon_c, \epsilon_s, \epsilon_z)$-public-coin ZK proof or argument with t messages. Then there exists a reduction R, which is a polynomial-time oracle-aided algorithm, and a polynomial-time computable function family $\mathcal{F} = \{f_x\}_{x \in \{0,1\}^*}$ such that, for any probabilistic polynomial-time algorithm $\mathcal{A}$, any polynomial p, and all large enough $n \in \mathbb{N}$,*

1. *For any $x \in \mathcal{L}_n$, if $\mathcal{A}$ distributionally inverts f_x with deviation at most $1/p(n)$, then*

$$\Pr\left[R^{\mathcal{A}}(x) = 1\right] \geq 1 - \epsilon_c(n) - (t(n) - 1) \cdot \epsilon_z(n) - \frac{t(n)}{p(n)}.$$

2. *For any $x \in \{0,1\}^n \setminus \mathcal{L}_n$,*

$$\Pr\left[R^{\mathcal{A}}(x) = 1\right] \leq \epsilon_s(n).$$

▶ **Corollary 38.** *For $\epsilon_c, \epsilon_s, \epsilon_z : \mathbb{N} \to [0,1]$ and $t : \mathbb{N} \to \mathbb{N}$, suppose there is an* NP *language $\mathcal{L} \notin$* ioP/poly *that has a t-message public-coin $(\epsilon_c, \epsilon_s, \epsilon_z)$-ZK proof or argument with $\epsilon_c + \epsilon_s + (t-1)\epsilon_z <_n 1$. Then auxiliary-input one-way functions exist.*

3.3 Reductions from Constant-Round Public-Coin ZK

For the lemma below, as opposed to the other (similar) ones, we consider the number of rounds in the protocol rather than the number of messages. Recall that a round consists of a message from the verifier, followed by a message from the prover. If the protocol starts with the prover sending a message, we think of it as starting with an empty message from the verifier instead.

▶ **Lemma 39.** *For some $\epsilon_c, \epsilon_s, \epsilon_z : \mathbb{N} \to [0,1]$ and any constant $k \in \mathbb{N}$, suppose a language $\mathcal{L} \in \mathsf{NP}$ has a k-round public-coin $(\epsilon_c, \epsilon_s, \epsilon_z)$-ZK proof or argument. Then there exists a polynomial-time oracle-aided algorithm R and families of oracle-aided functions $\mathcal{F}_1, \ldots, \mathcal{F}_k$ satisfying the following.*

For $i \in [k]$, the family $\mathcal{F}_i = \{f_{i,x}\}_{x \in \{0,1\}^*}$ consists of functions that require access to $(k-i)$ oracles, and are polynomial-time computable given these oracles. For any sequence of probabilistic polynomial-time algorithms $\mathcal{A}_1, \ldots, \mathcal{A}_k$, any polynomial p, and all large enough n, we have the following.

1. *For any $x \in \mathcal{L}_n$, if for all $i \in [k]$, $\mathcal{A}_i$ distributionally inverts $f_{i,x}^{\mathcal{A}_{i+1},\ldots,\mathcal{A}_k}$ with deviation at most $1/p(n)$, then*

$$\Pr\left[R^{\mathcal{A}_1 \ldots \mathcal{A}_k}(x) = 1\right] \geq 1 - \epsilon_c(n) - k \cdot \epsilon_z(n) - \frac{3k-2}{p(n)} - \mathsf{negl}(n).$$

2. *For any $x \in \{0,1\}^n \setminus \mathcal{L}_n$,*

$$\Pr\left[R^{\mathcal{A}_1 \ldots \mathcal{A}_k}(x) = 1\right] \leq \epsilon_s(n).$$

▶ Remark 40. In this approach, the functions are defined recursively and the number of function families is closely related to the round number k, so we can only achieve the implication result for any constant k. We do not know how to make it work beyond a finite number of rounds. This issue has also been observed in other work that employ similar recursive constructions [3, 10].

▶ **Corollary 41.** *For $\epsilon_c, \epsilon_s, \epsilon_z : \mathbb{N} \to [0,1]$ and a constant $k \in \mathbb{N}$, suppose there is an NP language $\mathcal{L} \notin \mathsf{P/poly}$ that has a k-round public-coin $(\epsilon_c, \epsilon_s, \epsilon_z)$-ZK proof or argument with $\epsilon_c + \epsilon_s + k \cdot \epsilon_z <_n 1$. Then infinitely-often auxiliary-input one-way functions exist.*

▶ Remark 42. Our approach to obtaining a contradiction relies on the overlap of input lengths on which all of the algorithms $\mathcal{A}_1, \ldots, \mathcal{A}_k$ succeed in their respective inversions. The negation of this assumption only implies the existence of infinitely-often one-way functions. Given that we only get infinitely-often security anyway, we are able to weaken our assumption to $\mathcal{L} \notin \mathsf{P/poly}$ instead of needing $\mathcal{L} \notin \mathsf{ioP/poly}$ as in our other results.

4 One-Way Functions

We complete our transformation by extending the result of [7] that shows that assuming there are zero-knowledge arguments for NP, auxiliary-input one-way functions imply standard one-way functions. We follow the approach taken by [4] in the context of weak zero-knowledge arguments, showing that the auxiliary-input OWFs obtained in the previous sections also imply standard one-way functions. The resulting theorems are as follows.

▶ **Theorem 43.** *If $\mathsf{NP} \not\subseteq \mathsf{ioP/poly}$ and, for some $\epsilon_c, \epsilon_s, \epsilon_z : \mathbb{N} \to [0,1]$ such that $\epsilon_c + \epsilon_s + \epsilon_z <_n 1$, every language in NP has an $(\epsilon_c, \epsilon_s, \epsilon_z)$-NIZK proof, then one-way functions exist. The same holds for NIZK arguments.*

▶ **Theorem 44.** *If* NP $\not\subseteq$ ioP/poly *and, for some* $\epsilon_c, \epsilon_s, \epsilon_z : \mathbb{N} \to [0,1]$ *and* $t : \mathbb{N} \to \mathbb{N}$ *such that* $\epsilon_c + \epsilon_s + (t-1) \cdot \epsilon_z <_n 1$*, every language in* NP *has a* t*-message public-coin* $(\epsilon_c, \epsilon_s, \epsilon_z)$*-ZK proof, then one-way functions exist. The same holds for ZK arguments.*

▶ **Theorem 45.** *If* NP $\not\subseteq$ P/poly *and, for some* $\epsilon_c, \epsilon_s, \epsilon_z : \mathbb{N} \to [0,1]$ *and constant* $k \in \mathbb{N}$ *such that* $\epsilon_c + \epsilon_s + k \cdot \epsilon_z <_n 1$*, every language in* NP *has a* k*-round public-coin* $(\epsilon_c, \epsilon_s, \epsilon_z)$*-ZK proof, then infinitely often one-way functions exist. The same holds for ZK arguments.*

The proofs of these theorems follow from combining the respective lemmas in Section 3 with lemmas stated later in this section. Theorems 43 and 44 are proven in Section 4.2.1, and Theorem 45 in Section 4.2.2.

As observed in [4, Section 5] (which in turn draws on the approach in [9]), the overall implication from weak ZK to OWFs can be broken up into three parts. The first part consists of showing that having such a weak ZK protocol for worst-case hard language implies an auxiliary-input OWF, and this we have shown in Section 3 (for various flavors of ZK protocols). The two remaining steps are the following:

1. Show that auxiliary-input OWFs imply that there exist what are called in [4] as *one-sided average-case hard* languages.
2. Show that (the corresponding flavor of) weak ZK protocols for such one-sided average-case hard languages imply standard OWFs.

We outline these transformations below for each flavor of weak ZK that we have considered so far. Some of the proofs below are along the lines of those of similar lemmas from prior work, especially from [4].

4.1 One-Sided Average-Case Hardness from ai-OWF

This lemma appears in [4, Lemma 4]. This result is somewhat independent, and does not have anything to do with weak zero-knowledge. We can thus use it as is.

▶ **Definition 46** (One-Sided Average-Case BPP)**.** *The class* 1AvgBPP/poly *consists of average-case problems* $(\mathcal{L}, \mathcal{D})$ *for which there is a non-uniform probabilistic polynomial-time algorithm* $\mathcal{A}$ *and functions* $a, b : \mathbb{N} \to [0,1]$ *with* $a >_n b$*, such that for all sufficiently large* n

1. *For every* $x \in \mathcal{L}_n$*,* $\Pr[\mathcal{A}(x) = 1] \geq a(n)$*.*
2. $\Pr_{x \leftarrow \mathcal{D}_n}[\mathcal{A}(x) = 1 | x \notin \mathcal{L}_n] \leq b(n)$*.*

▶ **Lemma 47** ([4, Lemma 4])**.** *Suppose that there exists an auxiliary-input one-way function. Then there exists an* $\mathcal{L} \in$ NP *such that* $(\mathcal{L}, \mathcal{U}) \notin$ io1AvgBPP/poly *and* $\Pr_{x \leftarrow \mathcal{U}_n}[x \notin \mathcal{L}_n] \geq 1/2$ *for all* $n \in \mathbb{N}$*.*

The following variant also follows from the proof of the above lemma.

▶ **Lemma 48.** *Suppose that there exists an infinitely-often ai-OWF, then there exists a language* $\mathcal{L} \in$ NP *such that* $(\mathcal{L}, \mathcal{U}) \notin$ 1AvgBPP/poly *and* $\Pr_{x \leftarrow \mathcal{U}_n}[x \notin \mathcal{L}_n] \geq 1/2$ *for all* $n \in \mathbb{N}$*.*

4.2 OWF from One-Sided Average-Case Hardness

4.2.1 NIZK and Public-Coin ZK

Combining Lemma 34 and Lemma 37 with the amplification for converting weak and distributional one-way functions into standard form [13, 8], we obtain the following lemma.

▶ **Lemma 49.** *For some $\epsilon_c, \epsilon_s, \epsilon_z : \mathbb{N} \to [0,1]$ and $t : \mathbb{N} \to \mathbb{N}$, consider a language $\mathcal{L} \in \mathsf{NP}$ with an $(\epsilon_c, \epsilon_s, \epsilon_z)$-NIZK protocol (in which case $t = 2$) or a t-message $(\epsilon_c, \epsilon_s, \epsilon_z)$-public-coin ZK protocol. For any polynomials p_1, p_2, there is a reduction R, which is a polynomial-time oracle-aided algorithm, and a polynomial-time computable function family $\mathcal{F} = \{f_x\}_{x\in\{0,1\}^*}$ such that for any probabilistic polynomial-time algorithm $\mathcal{A}$ and all large enough n,*

1. *When $x \in \mathcal{L}_n$, if $\mathcal{A}$ inverts f_x with probability at least $1/p_1(n)$, then*

$$\Pr\left[R^{\mathcal{A}}(x) = 1\right] \geq 1 - \epsilon_c(n) - (t(n) - 1) \cdot \epsilon_z(n) - 1/p_2(n).$$

2. *When $x \in \{0,1\}^n \setminus \mathcal{L}_n$,*

$$\Pr\left[R^{\mathcal{A}}(x) = 1\right] \leq \epsilon_s(n).$$

Note that our constructions of R and $\mathcal{F}$ in the proofs of Lemma 34 and Lemma 37 do not satisfy the above conditions. However, reductions and functions satisfying the above lemma can be obtained from the previous construction by applying the techniques in [13, 8]. Next, we use the following analogue of [4, Lemma 5].

▶ **Lemma 50.** *For $\epsilon_c, \epsilon_s, \epsilon_z : \mathbb{N} \to [0,1]$ and $t : \mathbb{N} \to \mathbb{N}$, consider any language $\mathcal{L}$ such that $(\mathcal{L},\mathcal{U}) \notin \mathsf{io1AvgBPP/poly}$, and $\Pr_{x\leftarrow\mathcal{U}_n}[x \notin \mathcal{L}_n] \geq 1/2$ for all $n \in \mathbb{N}$. If there is an $(\epsilon_c, \epsilon_s, \epsilon_z)$-NIZK protocol (in which case $t = 2$) or a t-message $(\epsilon_c, \epsilon_s, \epsilon_z)$-public-coin ZK protocol for $\mathcal{L}$ with $\epsilon_c + \epsilon_s + (t-1) \cdot \epsilon_z <_n 1$, then one-way functions exist.*

Combining Corollary 36, Lemma 47 and Lemma 50 yields Theorem 43. Analogously, for any round public-coin ZK protocol, by putting Corollary 38, Lemma 47 and Lemma 50 together, we obtain Theorem 44.

4.2.2 Constant-Round Public-Coin ZK

Lemma 39 combined with [8] yields the following lemma.

▶ **Lemma 51.** *For some $\epsilon_c, \epsilon_s, \epsilon_z : \mathbb{N} \to [0,1]$ and any constant $k \in \mathbb{N}$, suppose a language $\mathcal{L} \in \mathsf{NP}$ has a k-round public-coin $(\epsilon_c, \epsilon_s, \epsilon_z)$-ZK proof or argument. For any polynomials p_1, p_2, there exists a polynomial-time oracle-aided algorithm R and families of oracle-aided functions $\mathcal{F}_1, \ldots, \mathcal{F}_k$ satisfying the following.*

For $i \in [k]$, the family $\mathcal{F}_i = \{f_{i,x}\}_{x\in\{0,1\}^*}$ consists of functions that require access to $(k-i)$ oracles, and are polynomial-time computable given these oracles. For any sequence of probabilistic polynomial-time algorithms $\mathcal{A}_1, \ldots, \mathcal{A}_k$ and all large enough n, we have the following.

1. *For every $x \in \mathcal{L}_n$, if for all $i \in [k]$, $\mathcal{A}_i$ inverts $f_{i,x}$ with probability at least $1/p_1(n)$, then*

$$\Pr\left[R^{\mathcal{A}_1 \ldots \mathcal{A}_k}(x) = 1\right] \geq 1 - \epsilon_c(n) - k \cdot \epsilon_z(n) - 1/p_2(n)$$

2. *For every $x \in \{0,1\}^n \setminus \mathcal{L}_n$,*

$$\Pr\left[R^{\mathcal{A}_1 \ldots \mathcal{A}_k}(x) = 1\right] \leq \epsilon_s(n).$$

▶ **Lemma 52.** *For $\epsilon_c, \epsilon_s, \epsilon_z : \mathbb{N} \to [0,1]$ and a constant $k \in \mathbb{N}$, consider any language $\mathcal{L}$ such that $(\mathcal{L},\mathcal{U}) \notin \mathsf{1AvgBPP/poly}$, and $\Pr_{x\leftarrow\mathcal{U}_n}[x \notin \mathcal{L}_n] \geq 1/2$ for all $n \in \mathbb{N}$. If there is a k-round $(\epsilon_c, \epsilon_s, \epsilon_z)$-public-coin ZK protocol for $\mathcal{L}$ with $\epsilon_c + \epsilon_s + k \cdot \epsilon_z <_n 1$, then infinitely-often one-way functions exist.*

Theorem 45 follows directly from Corollary 41, Lemma 48, and Lemma 52.

All the omitted proofs can be found in the full version of our paper.

References

1 Benny Applebaum and Eliran Kachlon. NIZK amplification via leakage-resilient secure computation. In Yael Tauman Kalai and Seny F. Kamara, editors, *Advances in Cryptology - CRYPTO 2025 - 45th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 17-21, 2025, Proceedings, Part VII*, volume 16006 of *Lecture Notes in Computer Science*, pages 462–479. Springer, 2025. doi:10.1007/978-3-032-01907-3_15.

2 Nir Bitansky and Nathan Geier. Amplification of non-interactive zero knowledge, revisited. In Leonid Reyzin and Douglas Stebila, editors, *Advances in Cryptology - CRYPTO 2024 - 44th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 18-22, 2024, Proceedings, Part IX*, volume 14928 of *Lecture Notes in Computer Science*, pages 361–390. Springer, 2024. doi:10.1007/978-3-031-68400-5_11.

3 Jan Buzek and Stefano Tessaro. Collision resistance from multi-collision resistance for all constant parameters. In Leonid Reyzin and Douglas Stebila, editors, *Advances in Cryptology - CRYPTO 2024 - 44th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 18-22, 2024, Proceedings, Part V*, volume 14924 of *Lecture Notes in Computer Science*, pages 429–458. Springer, 2024. doi:10.1007/978-3-031-68388-6_15.

4 Suvradip Chakraborty, James Hulett, and Dakshita Khurana. On weak nizks, one-way functions and amplification. In Yael Tauman Kalai and Seny F. Kamara, editors, *Advances in Cryptology - CRYPTO 2025 - 45th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 17-21, 2025, Proceedings, Part VII*, volume 16006 of *Lecture Notes in Computer Science*, pages 580–610. Springer, 2025. doi:10.1007/978-3-032-01907-3_19.

5 Suvradip Chakraborty, James Hulett, Dakshita Khurana, and Kabir Tomer. Non-trivial zero-knowledge implies one-way functions. Cryptology ePrint Archive, Paper 2026/331, 2026. URL: https://eprint.iacr.org/2026/331.

6 Vipul Goyal, Aayush Jain, and Amit Sahai. Simultaneous amplification: The case of non-interactive zero-knowledge. In Alexandra Boldyreva and Daniele Micciancio, editors, *Advances in Cryptology - CRYPTO 2019 - 39th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 18-22, 2019, Proceedings, Part II*, volume 11693 of *Lecture Notes in Computer Science*, pages 608–637. Springer, 2019. doi:10.1007/978-3-030-26951-7_21.

7 Shuichi Hirahara and Mikito Nanashima. One-way functions and zero knowledge. In Bojan Mohar, Igor Shinkar, and Ryan O'Donnell, editors, *Proceedings of the 56th Annual ACM Symposium on Theory of Computing, STOC 2024, Vancouver, BC, Canada, June 24-28, 2024*, pages 1731–1738. ACM, 2024. doi:10.1145/3618260.3649701.

8 Russell Impagliazzo and Michael Luby. One-way functions are essential for complexity based cryptography (extended abstract). In *30th Annual Symposium on Foundations of Computer Science, Research Triangle Park, North Carolina, USA, 30 October - 1 November 1989*, pages 230–235. IEEE Computer Society, 1989. doi:10.1109/SFCS.1989.63483.

9 Yanyi Liu, Noam Mazor, and Rafael Pass. A note on zero-knowledge for NP and one-way functions. *IACR Cryptol. ePrint Arch.*, page 800, 2024. URL: https://eprint.iacr.org/2024/800.

10 Changrui Mu and Prashant Nalini Vasudevan. Instance-hiding interactive proofs - (extended abstract). In Elette Boyle and Mohammad Mahmoody, editors, *Theory of Cryptography - 22nd International Conference, TCC 2024, Milan, Italy, December 2-6, 2024, Proceedings, Part I*, volume 15364 of *Lecture Notes in Computer Science*, pages 3–34. Springer, 2024. doi:10.1007/978-3-031-78011-0_1.

11 Rafail Ostrovsky. One-way functions, hard on average problems, and statistical zero-knowledge proofs. In *Proceedings of the Sixth Annual Structure in Complexity Theory Conference, Chicago, Illinois, USA, June 30 - July 3, 1991*, pages 133–138. IEEE Computer Society, 1991. doi:10.1109/SCT.1991.160253.

12 Rafail Ostrovsky and Avi Wigderson. One-way fuctions are essential for non-trivial zero-knowledge. In *Second Israel Symposium on Theory of Computing Systems, ISTCS 1993, Natanya, Israel, June 7-9, 1993, Proceedings*, pages 3–17. IEEE Computer Society, 1993. doi:10.1109/ISTCS.1993.253489.

13 Andrew Chi-Chih Yao. Theory and applications of trapdoor functions (extended abstract). In *23rd Annual Symposium on Foundations of Computer Science, Chicago, Illinois, USA, 3-5 November 1982*, pages 80–91. IEEE Computer Society, 1982. doi:10.1109/SFCS.1982.45.

Towards Characterizing Secure Samplability

Hari Krishnan P. Anilkumar
TIFR, Mumbai, India

Keval Jain
IIIT Hyderabad, India

Manoj Prabhakaran
IIT Bombay, Mumbai, India

Vinod M. Prabhakaran
TIFR, Mumbai, India

Abstract

This work deals with the fundamental problem of characterizing which multiparty distributions can be securely sampled with information-theoretic security against passive corruption, when there is no setup or honest-majority. We focus on the case of 4-party distributions with boolean outputs for each party. We show that such a distribution is securely samplable if and only if every 2-party distribution derived from it by partitioning the parties into two sets is securely samplable. This extends a similar characterization previously known for 3-party distributions.

2012 ACM Subject Classification Security and privacy → Information-theoretic techniques

Keywords and phrases secure sampling, information-theoretic MPC

Digital Object Identifier 10.4230/LIPIcs.ITC.2026.6

Funding *Hari Krishnan P. Anilkumar*: supported by DAE, Govt. of India, under RTI4014.
Keval Jain: supported by IIT Bombay Trust Lab internship and by STCS, TIFR.
Manoj Prabhakaran: supported in part by IIT Bombay Trust Lab.
Vinod M. Prabhakaran: supported by DAE, Govt. of India, under RTI4014 and by ANRF through ANRF/ARGM/2025/000821/TS.

Declaration on GenAI/LLM use Computer searches and AI tools (specifically ChatGPT) were extensively used to arrive at conjectures and proof ideas throughout the project, but the final proofs were derived by hand.

1 Introduction

A fundamental question in the theory of secure multi-party computation (MPC) is to understand which functionalities can be securely realized. This question, posed in different settings, has led to a large and influential body of work in cryptography [7, 4, 5, 10, 3, 8, 9, 6, 11]. However, despite these major advances, a most basic question remains open:

> Which multi-party functions have information-theoretically secure protocols without any setup or restrictions on the set of parties that can be corrupted?

Here, by information-theoretic security we mean statistical security against passive corruption (but other variants are also interesting and remain open).

As a stepping stone towards answering this question, one can investigate this question restricted to functionalities which have no input – i.e., the setting of secure sampling. In this case, we fully understand the landscape of 2-party and 3-party distributions. Specifically, 2-party distributions that are securely samplable are exactly those which are "simple" – a combinatorial characterization which we define later (see [11] and references therein).

Further, 3-party distributions that are securely samplable are exactly those which when partitioned into 2-party distributions (by merging any two parties) always result in simple 2-party distributions [12]. However, these works leave open the question of 4 or more parties.

In this work we investigate the question of characterizing 4-party *boolean* distributions (i.e., each party receives a bit as its output). We show that the combinatorial characterization for 3-party distributions generalizes to this case.

Our proof involves a careful case analysis tailored to 4-party boolean distributions. In particular, our positive results (protocols for certain distributions) rely on sets satisfying a certain combinatorial condition having an "additive representation" over an abelian group. As we show in Appendix A, with a larger number of parties, not all securely samplable boolean distributions are additively representable.

1.1 Our Contribution

To state our result for securely samplable 4-party distributions, we recall that a 2-party distribution is securely samplable if and only if it is *simple*.[1] Given an n-party distribution, we can derive 2-party distributions by partitioning the set of parties into two; the original distribution is said to be *simple** if every such partition results in a 2-party distribution that is simple.

Clearly, if an n-party distribution is securely samplable, so is every distribution derived using a 2-way partition; hence only simple* distributions are securely samplable. In [12] it was shown that 3-party distributions that are securely samplable are exactly simple* distributions, and the question of whether this characterization holds for a larger number of parties was raised.

In this work we show that for 4-party *boolean* distributions (i.e., where each party gets a single bit), this is a sufficient condition as well:

▶ **Theorem 1.** *A 4-party boolean distribution is statistically securely samplable against passive corruption of any number of parties if and only if it is simple***.*

We leave open the question for 5-party distributions, as well as non-boolean 4-party distributions, whether being simple* is sufficient for secure samplability. Another open problem we leave is to extend our sufficiency results to *perfectly secure* sampling (as was done in [12] for 3-party distributions samplable using a circuit over uniformly random bits).

1.2 Related Work

As mentioned above, a large body of work has looked into the broader question of characterization of the class of 2-party and multi-party functions that can be information-theoretically securely computed, under several settings. Here we shall mention just a few of them which are closest in spirit to our setting.

In our setting of passive corruption without setup or honest-majority restrictions, the 2-party case is largely well understood (see [11] for a survey). However, for randomized 2-party functions there appear to be no simple combinatorial characterizations; instead a recent line of work has given algorithmic characterizations [1, 2].

[1] A 2-party distribution $P_{X,Y}$ is said to be simple if there exists a "common information" random variable Q (that is, Q can be expressed as a deterministic function of X and also as a deterministic function of Y), such that X and Y are independent conditioned on Q.

In the same setting, a characterization of securely computable multi-party functions (with more than 2 parties) remains elusive. Chor and Ishai [6] investigated a question similar to the one we explore, for deterministic functions (with inputs). Specifically, they asked if every 2-party function derived from an m-party function by partitioning the parties into two sets is securely computable, then whether the original function is guaranteed to be securely computable. In their case, they answered this problem in the negative. (For us, the answer is positive.)

The work most closely related to ours is [12], which considered the same sampling as ours. Their result in our setting (point-to-point channels and no setup) was limited to 3 parties, leaving the case of more parties open.

1.3 Overview

In this section, we give a detailed technical overview of our proof. Our main goal is to show that all simple* 4-party boolean distributions are securely samplable.

One may hope for a generic secure protocol for sampling from an n-party distribution that relies only on the fact that the distribution is simple*. Indeed, in the case of 3-party distributions, such a generic protocol was identified in [12]. Unfortunately, for 4-party distributions we do not know if such a generic protocol is possible. Even restricted to boolean outputs, we rely on a careful case analysis to construct secure protocols for all simple* distributions. At a high level, there are two parts to our proof: (1) analyzing simple* 4-party boolean distributions, and identifying a small class of explicit distributions for which protocols do not follow from prior work, and (2) showing that each of these distributions does indeed have a secure sampling protocol. We give an overview of these two parts below.

An Example of a Simple* Distribution

Before we proceed further, to familiarize the reader with the notions of simple and simple* distributions, we present an example.

It will be convenient to reinterpret a 2-party distribution $P_{\mathrm{X,Y}}$ as a distribution over edges in a bipartite graph: an edge (x, y) corresponds to $P_{\mathrm{X,Y}}(x, y) > 0$.[2] If $P_{\mathrm{X,Y}}$ is a simple distribution, then this graph breaks up into connected components each of which is a complete bipartite graph (a.k.a. biclique). When the distribution is uniform over its support, this also is a sufficient condition for it to be simple.

Now, we consider an example of a 4-party simple* distribution: Let $(\mathrm{X}_1, \ldots, \mathrm{X}_4)$ be uniformly distributed over $S = \{(0,0,0,0), (0,0,1,1), (0,1,0,1), (1,0,0,1)\}$. To verify that it is simple*, we consider each subset $J \subset \{1,2,3,4\}$ and check that the pair of random variables $(\mathrm{X}_J, \mathrm{X}_{\overline{J}})$ is simple.

For example, (see Figure 1) for $J = \{1\}$, the 2-party distribution of $(\mathrm{X}_1, \mathrm{X}_{\{2,3,4\}})$ is uniform over $\{(0,000), (0,011), (0,101), (1,001)\}$; the bipartite graph of this distribution has two connected components, each has a single node on the left-hand side and hence is a biclique. For $J = \{1,4\}$, we have $(\mathrm{X}_{\{1,4\}}, \mathrm{X}_{\{2,3\}})$ uniformly distributed over $\{(00,00), (01,01), (01,10), (11,00)\}$; in this case, the graph has two connected components – one component with 00 in the right-hand side and the other with 01 in the left-hand side – both of which are bicliques. Verifying this for all non-empty $J \subset \{1,2,3,4\}$ confirms that the distribution is simple*.

[2] For simplicity, for now we consider distributions which are uniform over their support. In the more general case, we consider a weighted graph with the weight of the edge (x, y) being $P_{\mathrm{X,Y}}(x, y)$.

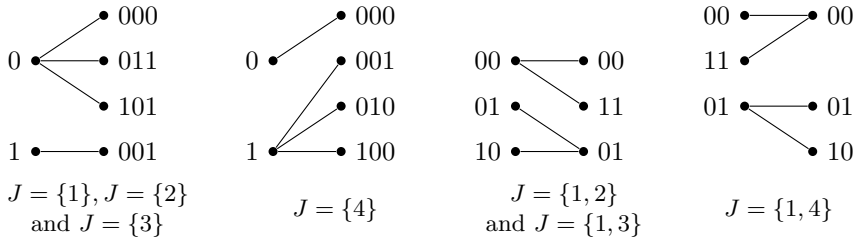


Figure 1 Bipartite graphs corresponding to the different partitions $(\mathrm{X}_J, \mathrm{X}_{\bar{J}})$ for the support $S = \{(0,0,0,0), (0,0,1,1), (0,1,0,1), (1,0,0,1)\}$.

Understanding 4-Party Boolean Simple* Distributions

Reducible and Non-reducible Distributions

We start by dividing the simple* distributions into two cases: (1) *reducible* distributions which can be securely sampled using a secure sampling protocol for a 3−party distribution, and (2) *non-reducible* distributions, which we will analyze further.

Informally, a distribution is reducible if one of the parties has an output independent of the others, or if two parties have the same (or opposite) output. If a 4-party simple* distribution is reducible, it is easy to see that removing the independent/duplicate party results in a 3-party simple* distribution. Next, by [12], we obtain a secure sampling protocol for this 3-party distribution. Finally, this 3-party protocol can be readily transformed to a 4-party secure protocol for the original distribution.

Non-reducible simple* distributions require more work. We catalog all such 4-party boolean distributions, so that we can show that each one of them is securely samplable. We do this in two steps: first we characterize the support of such distributions ("admissible supports"), and then we characterize the probability mass over such a support.

Support of Non-reducible Simple* Distributions

Firstly, for a 2-party simple distribution, the support can be identified with a bipartite graph whose connected components are bicliques. For a simple* distribution, the requirement on its support is that for every partition of the parties into two, the resulting 2-party distribution's support forms a disjoint collection of bicliques.

Being simple* *and non-reducible* imposes several combinatorial constraints on the support. In particular, we argue that the support, viewed as a subset of $\{0,1\}^4$, forms a "code" of distance at least 2, and it should be of size 4, 5, 6 or 8.

To be able to easily enumerate the supports, we consider two supports equivalent if they are related to each other by a permutation of the parties followed by complementing the outputs of a subset of the parties. Then, up to this equivalence (which we call relabeling), we exhaustively argue that there is only one support each of size 5, 6 and 8, and only two supports of size 4 that are possible for a 4-party boolean simple* distribution. In the sequel we refer to these five possible supports as the *admissible supports*.

Non-reducible Simple* Distributions

To complete our explicit characterization of non-reducible simple* 4-party boolean distributions, we need to understand how probabilities could be assigned to the elements in an admissible support while maintaining simple* nature. One obvious choice is the uniform

distribution. This can be further generalized to a *product-form*: we associate a non-negative weight to each output (0 and 1) of each party, and the probability of any output tuple in the support is proportional to the product of the weights assigned to each party's outcome. If the uniform distribution is simple*, such a transformation over the same support keeps it simple*.

Remarkably, these product-form distributions turn out to be the only distributions that are simple*. We prove this by analyzing each of the five admissible supports separately.

Protocols for Simple* Distributions

Since a secure protocol for a reducible 4-party simple* distribution can be obtained from the secure protocol for a simple* 3-party distribution (from [12]), we focus on the non-reducible 4-party boolean simple* distributions. Then it suffices to give a statistically secure protocol for simple* distributions with each of the 5 admissible supports identified above.

Firstly, we note that due to the product form of these distributions the following "sample-and-check" template suffices for a statistically secure protocol:

- Each party locally samples an independent bit according to its fixed marginal distribution.
- They will "semi-securely" compute whether the bits they collectively sampled lie in the support of the desired distribution. If it does, they output the sampled bits and terminate; else, they repeat the whole process. (After security parameter many attempts, one party may sample an element and send it to the others; this occurs with negligible probability.)

Here a "semi-secure" membership test is a protocol which provides privacy guarantees when the inputs do form a member of the set (i.e., if two distinct elements of the set have the same bits restricted to a subset of the parties, then their views are distributed identically when the protocol is executed with those two inputs). When the inputs form an element outside the set, there is no privacy guarantee. Note that such a protocol suffices in the above template since we reject samples which are not elements of the support.

To instantiate the second step of this template, we need a *semi-secure set-membership test*, for each of the five admissible supports. The key to our protocols is the observation that all the admissible supports are "additively representable," as defined below. (In our case, $\mathcal{X}_i = \{0, 1\}$ and $n = 4$.)

▶ **Definition 2.** *A set $S \subseteq \mathcal{X}_1 \times \ldots \times \mathcal{X}_n$ is said to be* additively representable *if there exist a finite abelian group G, a set of maps $\mu_i : \mathcal{X}_i \to G$ and an element $g \in G$, such that $S = \{(x_1, \ldots, x_n) | \sum_{i=1}^{n} \mu_i(x_i) = g\}$.*

We note that every additively representable set has a simple semi-secure membership protocol (against passive corruption). It is well known that the summation task over any finite abelian group can be securely computed. To check membership in an additively representable set S, each party locally maps their input x_i to $\mu_i(x_i)$, and then they securely compute the sum $\sum_{i=1}^{n} \mu_i(x_i)$. If this sum equals g, then the parties only learn that their inputs form an element of the set; otherwise, they do learn information beyond the fact that the inputs are not in the set, but this is permitted in a semi-secure protocol.

To complete our proof, we need to show that each of the five admissible supports we identified as supports for simple* distributions are additively representable. We show that this is indeed the case (using groups $\mathbb{Z}_2, \mathbb{Z}_3$ and $\mathbb{Z}_6$). In particular, for the example from the beginning of this section, $S = \{(0,0,0,0), (0,0,1,1), (0,1,0,1), (1,0,0,1)\}$, we use the group $\mathbb{Z}_6$, the target element $g = 0$, and maps $\mu_1 = \mu_2 = \mu_3$ which map $0 \mapsto 0$ and $1 \mapsto 1$, and $\mu_4 : 0 \mapsto 0$ and $\mu_4 : 1 \mapsto -1$.

2 Preliminaries

For any positive integer n, we write $[n]$ to denote the set $\{1, \ldots, n\}$. We use uppercase letters in italics to denote sets and uppercase letters in upright to denote random variables.

Multi-Party Distributions

A distribution P over a (finite) alphabet $\mathcal{X}$ is simply a function $P : \mathcal{X} \to [0, 1]$ such that $\sum_{x \in \mathcal{X}} P(x) = 1$. The basic objects of interest in this work are n-party distributions. These are simply distributions defined over some alphabet $\mathcal{X}_1 \times \cdots \times \mathcal{X}_n$, with the connotation that an element $(x_1, \ldots, x_n)$ sampled from this distribution will be distributed among n parties $\mathsf{P}_1, \ldots, \mathsf{P}_n$, so that P_i gets x_i. We shall often write such a distribution as $P_{\mathrm{X}_1,\ldots,\mathrm{X}_n}$, where the random variable X_i denotes the element given to P_i. The distribution $P_{\mathrm{X}_1,\ldots,\mathrm{X}_n}$ is said to be *boolean* if each X_i is a boolean random variable – i.e., its alphabet is $\{0, 1\}$.

For any $J \subseteq [n]$, we will use X_J to denote the tuple of variables $(\mathrm{X}_i)_{i \in J}$. We also define, for $x \in \mathcal{X}_1 \times \ldots \times \mathcal{X}_n$ and subset $J \subseteq [n]$, the element x_J to be the tuple obtained by restricting x to the indices in J. When there is no ambiguity, we shall often represent a tuple by a string (e.g., 0000 instead of $(0, 0, 0, 0)$).

Protocols

The other basic object in this work is a secure sampling protocol. In an (inputless) protocol, parties $\mathsf{P}_1, \ldots, \mathsf{P}_n$ have access to private randomness at the beginning of the protocol. The protocol proceeds in rounds; in every round, each party sends a message to each of the other parties, where the message is a function of the sender's private randomness and all the messages that it received in all the previous rounds in the protocol. After a finite number of rounds, the protocol should terminate and each party P_i should produce an output.

For a protocol Π, the random variable View_i^{Π} denotes the view of a party P_i, which consists of the private randomness of P_i and all the messages received by P_i during the entire protocol. The random variable out_i^{Π} denotes the output produced by P_i. out_i^{Π} is a deterministic function of View_i^{Π}.

The statistical distance (total variation distance) $\mathrm{SD}(\mathrm{X}_1, \mathrm{X}_2)$ between two random variables X_1 and X_2 defined on some alphabet $\mathcal{X}$ is

$$\mathrm{SD}(\mathrm{X}_1, \mathrm{X}_2) = \frac{1}{2} \sum_{x \in \mathcal{X}} |\Pr[\mathrm{X}_1 = x] - \Pr[\mathrm{X}_2 = x]| = \max_{T \subseteq \mathcal{X}} (\Pr[\mathrm{X}_1 \in T] - \Pr[\mathrm{X}_2 \in T]).$$

The support of an n-party distribution P over $\mathcal{X}_1 \times \ldots \mathcal{X}_n$ is $\mathrm{supp}(P) = \{x \in \mathcal{X}_1 \times \ldots \times \mathcal{X}_n : P(x) > 0\}$. For any distribution $P_{\mathrm{X,Y,Q}}$, the statement that X and Y are independent conditioned on Q will be denoted by the Markov chain $\mathrm{X} \leftrightarrow \mathrm{Q} \leftrightarrow \mathrm{Y}$.

▶ **Definition 3** (Securely Samplable Distribution). *For $\epsilon > 0$, a protocol Π is said to ϵ-securely sample from $P_{X_1,\ldots,X_n}$ if for each $C \subset [n]$ there exists a random variable Sim_C (jointly distributed with the ideal outputs $\mathrm{X}_{[n]}$) such that*

$$\mathsf{Sim}_C \leftrightarrow \mathrm{X}_C \leftrightarrow \mathrm{X}_{\overline{C}}, \text{ and} \tag{1}$$

$$\mathrm{SD}\left(\left(\mathrm{X}_{\overline{C}}, \mathsf{Sim}_C\right), \left(\mathsf{out}_{\overline{C}}^{\Pi}, \mathsf{View}_C^{\Pi}\right)\right) \leq \epsilon. \tag{2}$$

$P_{\mathrm{X}_1,\ldots,\mathrm{X}_n}$ is said to be securely samplable *if for every $\epsilon > 0$ there exists a protocol that ϵ-securely samples from it.*

When $\epsilon = 0$, we say that $P_{\mathrm{X}_1,\ldots,\mathrm{X}_n}$ is samplable under perfect security.

▶ **Definition 4** (Bipartite Graph of a 2-Party Distribution)**.** *The bipartite graph of a 2-party probability distribution P_{XY} over the alphabet $\mathcal{X} \times \mathcal{Y}$ is defined as the weighted graph $G_{P_{\mathrm{XY}}} = (\mathcal{X}, \mathcal{Y}, P_{\mathrm{XY}})$. That is, the weight of an edge $(x, y) \in \mathcal{X} \times \mathcal{Y}$ is $P_{\mathrm{XY}}(x, y)$.*

▶ **Definition 5** (Simple 2-Party Distribution)**.** *A 2-party distribution $P_{\mathrm{X,Y}}$ over the alphabet $\mathcal{X} \times \mathcal{Y}$ is said to be* simple *if there are functions $w_1 : \mathcal{X} \to \mathbb{R}^+$, $w_2 : \mathcal{Y} \to \mathbb{R}^+$, $q_1 : \mathcal{X} \to \mathbb{Z}$ and $q_2 : \mathcal{Y} \to \mathbb{Z}$ such that for all $x \in \mathcal{X}, y \in \mathcal{Y}$,*

$$P_{\mathrm{X,Y}}(x, y) = \begin{cases} w_1(x)w_2(y) & \text{if } q_1(x) = q_2(y) \\ 0 & \text{otherwise} \end{cases}. \tag{3}$$

The following alternate view of a simple distribution is often convenient: $P_{\mathrm{X,Y}}$ is simple if and only if there exists a random variable Q over $\mathbb{Z}$ jointly distributed with X and Y such that

1. Q is a deterministic function of X and a deterministic function of Y.
2. The Markov chain $\mathrm{X} \leftrightarrow \mathrm{Q} \leftrightarrow \mathrm{Y}$ holds; i.e., X and Y are independent conditioned on Q.

Indeed, if P_{XY} is simple as defined, taking $\mathrm{Q} = q_1(\mathrm{X})$ it holds that $\mathrm{Q} = q_2(\mathrm{Y})$ also w.p. 1. Further, for a given q, for all x, y s.t. $q_1(x) = q_2(y) = q$, $P_{\mathrm{X,Y|Q}}(x, y|q) = \frac{w_1(x)w_2(y)}{P_{\mathrm{Q}}(\mathrm{Q}=q)} = \frac{w_1(x)w_2(y)}{(\sum_{x':q_1(x')=q} w_1(x'))(\sum_{y':q_2(y')=q} w_2(y'))}$ and 0 for all other x, y; then $P_{\mathrm{X|Q}}(x|q) = \sum_{y:q_2(y)=q} P_{\mathrm{X,Y|Q}}(x, y|q) = \frac{w_1(x)}{\sum_{x':q_1(x')=q} w_1(x')}$, and similarly $P_{\mathrm{Y|Q}}(y|q) = \frac{w_2(y)}{\sum_{y':q_2(y')=q} w_2(y')}$, so that $P_{\mathrm{X,Y|Q}}(x, y|q) = P_{\mathrm{X|Q}}(x|q)P_{\mathrm{Y|Q}}(y|q)$. (This factorization also holds for (x, y) for which it is not the case that $q_1(x) = q_2(y) = q$, as the LHS and at least one of the factors in the RHS are 0.)

Conversely, if such a Q exists, we use the given deterministic functions that define Q as q_1, q_2 and set $w_1(x) = P_{\mathrm{X}}(x), w_2(y) = P_{\mathrm{Y|Q}}(y|q_2(y))$, so that, for x, y pairs such that $P_{\mathrm{X,Y}}(x, y) > 0$, we have

$$\begin{aligned} P_{\mathrm{X,Y}}(x, y) = P_{\mathrm{X,Y,Q}}(x, y, q_1(x) = q_2(y)) = P_{\mathrm{X,Q}}(x, q_1(x))P_{\mathrm{Y|Q}}(y|q_2(y)) \\ = P_{\mathrm{X}}(x)P_{\mathrm{Y|Q}}(y|q_2(y)) = w_1(x)w_2(y). \end{aligned} \tag{4}$$

We will often refer to the variable Q as the *common information* for a simple $P_{\mathrm{X,Y}}$.

▶ **Proposition 6** ([13])**.** *A two-party distribution is simple if and only if it is securely samplable.*

The notion of a simple distribution can be extended to n-party distributions for $n \geq 2$ as follows.

▶ **Definition 7** (Simple* Distributions)**.** *An n-party distribution $P_{\mathrm{X}_{[n]}}$ is a simple* distribution if for every set $J \subset [n]$, the 2-party distribution $P_{\mathrm{X}_J\mathrm{X}_{\overline{J}}}$ is a simple distribution.*

More explicitly, a distribution $P_{\mathrm{X}_{[n]}}$ is simple* if, for every non-empty $J \subset [n]$, there is a weight function $w_J : \mathrm{X}_J \to \mathbb{R}^+$ and an index function $q_J : \mathrm{X}_J \to \mathbb{Z}$ such that

$$P_{\mathrm{X}_{[n]}}(x) = \begin{cases} w_J(x_J)w_{\overline{J}}(x_{\overline{J}}) & \text{if } q_J(x_J) = q_{\overline{J}}(x_{\overline{J}}) \\ 0 & \text{otherwise.} \end{cases}$$

▶ **Lemma 8.** *If an n-party distribution is securely samplable, then it is simple*.*

Proof. If an n-party distribution $P_{\mathrm{X}_1,\ldots,\mathrm{X}_n}$ is securely samplable using a protocol Π, for every $J \subset [n]$, consider the 2-party protocol in which one party simulates the parties $(\mathsf{P}_i)_{i\in J}$ in Π, and the other party simulates $(\mathsf{P}_i)_{i\in\overline{J}}$. This is a secure sampling protocol for $P_{\mathrm{X}_J\mathrm{X}_{\overline{J}}}$. By Proposition 6, it follows that for every $J \subset [n]$, the distribution $P_{\mathrm{X}_J\mathrm{X}_{\overline{J}}}$ is simple, which means that $P_{\mathrm{X}_1,\ldots,\mathrm{X}_n}$ is simple*. ◀

3 4-Party Boolean Simple* Distributions are Securely Samplable

In this section we present the overall structure of our proof of Theorem 1, that a 4-party boolean distribution is securely samplable if and only if it is simple*.

By Lemma 8, the "only if" direction is immediate. Therefore, it remains to prove that every simple* 4-party boolean distribution is securely samplable.

The proof is divided into two parts. In this section we first identify families of simple* 4-party distributions whose secure samplability can be obtained from the secure samplability of a suitable 3-party distribution. We call these distributions reducible. In the next section we then analyze the remaining simple* 4-party distributions.

▶ **Definition 9.** *An n-party distribution $P_{\mathrm{X}_1,\ldots,\mathrm{X}_n}$ is said to* reduce to *the $(n-1)$-party distribution $P_{\mathrm{X}_{[n]\setminus\{i\}}}$ for an $i \in [n]$ if either of the following conditions hold:*

1. *X_i is distributed independently of the other variables, i.e., $\mathrm{X}_i \perp\!\!\!\perp \mathrm{X}_{[n]\setminus\{i\}}$*
2. *For some $i' \neq i$ and some deterministic function f, $\mathrm{X}_i = f(\mathrm{X}_{i'})$.*

$P_{\mathrm{X}_1,\ldots,\mathrm{X}_n}$ is said to be reducible *if it reduces to $P_{\mathrm{X}_{[n]\setminus\{i\}}}$ for some $i \in [n]$. Otherwise, we call it* non-reducible.

We shall establish that if a 4-party simple* distribution is reducible then it is securely samplable. For this, we first need the following lemma.

▶ **Lemma 10.** *Suppose P reduces to P'. Then,*

1. *If P is simple*, then so is P'.*
2. *If P' is securely samplable, so is P.*

Proof. Suppose P is $P_{\mathrm{X}_1,\ldots,\mathrm{X}_n}$ and P' is $P_{\mathrm{X}_{[n]\setminus\{i\}}}$. W.l.o.g, we shall take $i = n$, so that P' is $P_{\mathrm{X}_{[n-1]}}$. We prove each of the items in the claim.

P simple* $\implies$ P' simple*. To show that P' is simple*, for any subset $J \subset [n-1]$ and its complement $\overline{J} = [n-1] \setminus J$, we need to show that the 2-party distribution $P_{\mathrm{X}_J,\mathrm{X}_{\overline{J}}}$ is simple. For this we can use the fact that $P_{\mathrm{X}_J,\mathrm{X}_{\overline{J}\cup\{n\}}}$ is simple. We fix a J now.

Recall that there are two ways in which P can reduce to P' – either X_n is independent of $\mathrm{X}_{[n-1]}$, or X_n is a function of $\mathrm{X}_{i'}$ for some $i' \neq n$.

In the former case, we invoke the following claim, with $\mathrm{U} = \mathrm{X}_J, \mathrm{V} = \mathrm{X}_{\overline{J}}$ and $\mathrm{W} = \mathrm{X}_n$, to conclude that $P_{\mathrm{X}_J,\mathrm{X}_{\overline{J}}}$ is simple.

▷ Claim 11. Let $P_{\mathrm{UVW}} = P_{\mathrm{UV}}P_{\mathrm{W}}$. Then, if the two party distribution $P_{\mathrm{U,VW}}$ is simple, so is the two-party distribution $P_{\mathrm{U,V}}$.

Proof. Let Q be common information of $P_{\mathrm{U,VW}}$. Note that since $\mathrm{W} \perp\!\!\!\perp (\mathrm{U},\mathrm{V})$, $\mathrm{U} \leftrightarrow \mathrm{V} \leftrightarrow \mathrm{W}$, which further implies that $\mathrm{Q} \leftrightarrow \mathrm{V} \leftrightarrow \mathrm{W}$ as Q is a function of U. Using this conditional independence, we have that since, Q is a function of (V,W), Q must be a function of only V. Moreover, Q is a function of U. This, along with the fact that $\mathrm{U} \leftrightarrow \mathrm{Q} \leftrightarrow (\mathrm{V},\mathrm{W}) \implies \mathrm{U} \leftrightarrow \mathrm{Q} \leftrightarrow \mathrm{V}$ completes the proof that $P_{\mathrm{U,V}}$ is simple. ◁

In case X_n is a function of $\mathrm{X}_{i'}$ for some $i' \neq n$, we invoke the following claim, again with $\mathrm{X}_J = \mathrm{U}$, $\mathrm{X}_{\overline{J}} = \mathrm{V}$ and $\mathrm{X}_n = \mathrm{W}$, and assuming w.l.o.g, that $i' \in J$ (swapping the roles of $J, \overline{J}$ if necessary).

▷ Claim 12. Suppose $P_{\mathrm{UW,V}}$ defined on $\mathcal{U} \times \mathcal{V} \times \mathcal{W}$ is simple, where $\mathrm{W} = f(\mathrm{U})$ for a deterministic function $f : \mathcal{U} \to \mathcal{W}$. Then $P_{\mathrm{U,V}}$ is simple.

Proof. Let Q be the common information for $P_{\mathrm{UW,V}}$. Since Q is a function of (U, W), it is a function of U as well; and Q is also a function of V since it is the common information. Furthermore, since $(\mathrm{U}, \mathrm{W}) \leftrightarrow \mathrm{Q} \leftrightarrow \mathrm{V}$, therefore, $\mathrm{U} \leftrightarrow \mathrm{Q} \leftrightarrow \mathrm{V}$. ◁

Thus in either case, if P is simple*, so is P'.

P' securely samplable $\implies$ P securely samplable. Given an $(n-1)$-party secure protocol for sampling P', we shall upgrade it to an n-party protocol for P. Again, we separately consider the two different ways in which P can reduce to P'.

If X_n is independent of $\mathrm{X}_{[n-1]}$, we obtain a secure protocol for P in which the n^{th} party simply outputs a sample from the marginal distribution P_{X_n} and independently, the other $n-1$ parties run the given protocol for sampling from $P_{\mathrm{X}_{[n-1]}}$.

If X_n is a function of $\mathrm{X}_{i'}$ for some $i' \neq n$, say, $\mathrm{X}_n = f(\mathrm{X}_{i'})$, we obtain a secure protocol for P in which the first $n-1$ parties run the given protocol for sampling from $P_{\mathrm{X}_{[n-1]}}$, after which party i' sends the value $f(\mathrm{X}_{i'})$ to the n^{th} party, who outputs it.

It is easy to see that in either case, the protocol obtained above is a secure protocol for sampling from P. ◀

We now need the following result about secure samplability of 3-party distributions from [12].

▶ **Proposition 13** ([12]). *A 3-party boolean distribution is securely samplable against passive corruption of any subset of parties if and only if it is simple*.*

Now we state the main result of this section.

▶ **Lemma 14.** *If a 4-party distribution is simple* and reducible, then, it is securely samplable.*

Proof. Suppose a 4-party distribution P is simple* and reduces to a 3-party distribution P'. Then by Lemma 10 (item 1), P' is simple*. Then, by Proposition 13, P' is securely samplable. Invoking Lemma 10 (item 2), we conclude that P is securely samplable, as desired. ◀

4 Supports of Simple* Non-Reducible 4-Party Boolean Distributions

▶ **Definition 15** (Relabeling a Distribution). *A* relabeling *of n-party distributions over $\mathcal{X}^n$ is a bijection $\Phi_\varphi : \mathcal{X}^n \to \mathcal{X}^n$ specified by a tuple $\varphi = (\pi, \sigma_1, \ldots, \sigma_n)$ where $\pi : [n] \to [n]$ is a permutation and, for each $i \in [n]$, each $\sigma_i : \mathcal{X} \to \mathcal{X}$ is a bijection. The bijection $\Phi_\varphi : \mathcal{X}^n \to \mathcal{X}^n$ is defined as $\Phi_\varphi(x_1, \ldots, x_n) := \big(\sigma_1(x_{\pi(1)}), \ldots, \sigma_n(x_{\pi(n)})\big)$. We define the relabeled distribution $\varphi(P)$ by*

$$\varphi(P)(v) := P\big(\Phi_\varphi^{-1}(v)\big), \qquad \forall\, v \in \mathcal{X}^n.$$

Example. Let $\mathcal{X} = \{0,1\}$ and $n = 4$. Let π swap coordinates 1 and 2 (so $\pi(1) = 2, \pi(2) = 1$), let σ_1 be bit-flip ($\sigma_1(b) = 1 - b$), and let $\sigma_2, \sigma_3, \sigma_4$ be the identity. Then $\Phi_\varphi(x_1, x_2, x_3, x_4) = (1-x_2,\, x_1,\, x_3,\, x_4)$, so for any $(v_1, v_2, v_3, v_4) \in \{0,1\}^4$ we have $\varphi(P)(v_1, v_2, v_3, v_4) = P(v_2,\, 1-v_1,\, v_3,\, v_4)$. It is easy to see that relabeling does not affect the combinatorial property of being simple*, nor the cryptographic property of being securely samplable. We record this in the following lemma.

▶ **Lemma 16** (Invariance under relabeling). *Let P be an n-party distribution $P_{\mathrm{X}_1,\ldots,\mathrm{X}_n}$ and φ a relabeling for it. Then P is simple* if and only if $\varphi(P)$ is simple*, and P is securely samplable if and only if $\varphi(P)$ is securely samplable.*

In this section we show that a simple* non-reducible 4-party boolean distribution, up to relabeling, must have one of the following five sets as its support. In the sequel, we shall refer to them as the *admissible supports*.

$$
\begin{aligned}
S_1 &:= \{0000, 0011, 0101, 0110, 1001, 1010, 1100, 1111\},\\
S_2 &:= \{0000, 0011, 0101, 1010, 1100, 1111\},\\
S_3 &:= \{0000, 0011, 0101, 1001, 1110\},\\
S_4 &:= \{0000, 0011, 0101, 1001\},\\
S_5 &:= \{0000, 0011, 0101, 1110\}.
\end{aligned} \tag{5}
$$

▶ **Proposition 17.** *Let P be a simple* non-reducible 4-party boolean distribution. Then, up to relabeling, supp(P) must be one of S_1, S_2, S_3, S_4, S_5 defined in Equation* (5).

The proof proceeds by a case analysis on $|\mathrm{supp}(P)|$. We first rule out supports of size smaller than 4, then analyze the cases of support sizes 4 through 8, and finally rule out supports of size larger than 8. This yields Proposition 17.

Case Analysis of Supports of Different Sizes

Before proceeding with our analysis, we introduce the following unweighted bipartite graph for representing the support of a 2-party distribution derived from an n-party distribution with a given support.

▶ **Definition 18** (Support graph). *Let $S \subseteq \mathcal{X}_1 \times \cdots \times \mathcal{X}_n$ and let $J \subset [n], J \neq \emptyset$. The support graph B_J^S is the bipartite graph whose left vertex set is $\mathcal{X}_J$, whose right vertex set is $\mathcal{X}_{[n]\setminus J}$, and whose edge set is $E\big(B_J^S\big) := \{(x_J, x_{[n]\setminus J}) : x \in S\}$.*

We shall also refer to a few combinatorial properties of the support of a simple* non-reducible boolean distribution. For this it will be useful to define minimum Hamming distance of a subset $S \subseteq \{0,1\}^n$ as $\Delta(S) = \min_{\substack{x,y\in S\\ x\neq y}} d_H(x,y)$, where, $d_H(x,y)$ denotes the Hamming distance between x and y, namely, the number of coordinates on which x and y differ.

▶ **Lemma 19.** *Let S be the support of a simple* non-reducible boolean distribution. Then:*

1. *$\Delta(S) \geq 2$;*
2. *no coordinate is constant on S, i.e., there is no $i \in [n]$ such that for a fixed b, $x_i = b$ for all $x \in S$.*
3. *no two coordinates are identical on all of S or complementary on all of S.*

Proof. Let P be a non-reducible simple* distribution over $\{0,1\}^n$ such that $S = \mathrm{supp}(P)$. For item (1), suppose there exist distinct $x, y \in S$ such that $d_H(x,y) = 1$. Let $i \in [n]$ be the unique coordinate on which x and y differ, and let $r := x_{[n]\setminus\{i\}} = y_{[n]\setminus\{i\}}$. Since P is simple*,

the two-party distribution $(\mathrm{X}_i, \mathrm{X}_{[n]\setminus\{i\}})$ is simple. Let Q be its common information. Then Q is a deterministic function of X_i, and also of $\mathrm{X}_{[n]\setminus\{i\}}$. Since both $\Pr[(\mathrm{X}_i, \mathrm{X}_{[n]\setminus\{i\}}) = (0, r)] > 0$ and $\Pr[(\mathrm{X}_i, \mathrm{X}_{[n]\setminus\{i\}}) = (1, r)] > 0$, it follows that the two values $\mathrm{X}_i = 0$ and $\mathrm{X}_i = 1$ give rise to the same value of Q and hence Q is constant. Therefore $\mathrm{X}_i \perp\!\!\!\perp \mathrm{X}_{[n]\setminus\{i\}}$, so P is reducible which gives a contradiction.

For item (2), if some coordinate i were constant on S, then X_i would be constant, and hence independent of $X_{[n]\setminus\{i\}}$. Thus P would be reducible, a contradiction.

For item (3), if coordinates i and j were identical on S or complementary on S, then X_i would be a deterministic function of X_j. Thus again P would be reducible, a contradiction. ◀

Now we are ready to start our case analysis of supports of non-reducible simple* 4-party boolean distributions, organized by the size of the support. We start by ruling out supports of size smaller than 4.

▶ **Lemma 20.** *Let P be a simple* non-reducible 4-party boolean distribution, and let $S := supp(P)$. Then $|S| \geq 4$.*

Proof. By Lemma 19, no coordinate is constant on S, and no two coordinates are identical or complementary on S.

Suppose $|S| \leq 3$, and write $S = \{x^{(1)}, \ldots, x^{(t)}\}$ with $t \leq 3$. For each coordinate $i \in [4]$, consider the length-t bitstring $\left(x_i^{(1)}, \ldots, x_i^{(t)}\right) \in \{0,1\}^t$. Since no coordinate is constant on S, this bitstring is neither all-0 nor all-1. Hence there are at most $2^t - 2$ possibilities. Identifying a bitstring with its complement leaves at most $\frac{2^t-2}{2} \leq 3$ equivalence classes. Since there are 4 coordinates, two coordinates lie in the same class, so they are identical or complementary on S, a contradiction. ◀

▶ **Lemma 21.** *Let $S \subseteq \{0,1\}^4$ be such that $\Delta(S) \geq 2$, and let $J \subseteq [4]$ satisfy $|J| = 2$. Then every vertex of B_J^S has degree at most 2.*

Proof. Let u be a left vertex of B_J^S. If u had three distinct neighbors $v^{(1)}, v^{(2)}, v^{(3)}$, then among the three 2-bit strings $v^{(1)}, v^{(2)}, v^{(3)}$ there would be two that differ in exactly one coordinate. The corresponding two points of S would then differ in exactly one coordinate, contradicting the assumption that $\Delta(S) \geq 2$. The same argument applies to the right vertices. ◀

▶ **Lemma 22.** *Let P be a simple* non-reducible 4-party boolean distribution, and let $S := supp(P)$. If $|S| = 4$, then, up to relabeling, S is either S_4 or S_5.*

Proof. By Lemma 19, $\Delta(S) \geq 2$, no coordinate is constant on S, and no two coordinates are identical or complementary on S.

Since $|S| = 4$, $\Delta(S) \leq 2$. Indeed, if $\Delta(S) \geq 3$, then the radius-1 Hamming balls around the four points of S would be disjoint, each of size 5, which is impossible in $\{0,1\}^4$. Thus there exist $x, y \in S$ with $d_H(x, y) = 2$. Up to relabeling, we may assume that $0000, 0011 \in S$.

Consider the support graph $B_{\{1,2\}}^S$. The left vertex 00 is adjacent to the right vertices 00 and 11. If some other point $z \in S$ satisfied $z_{\{3,4\}} \in \{00, 11\}$, then the connected component containing the left vertex 00 would contain a second left vertex adjacent to both 00 and 11 (to both 00 and 11 due to the fact that P is a simple* distribution). Since no coordinate is constant on S, this second left vertex cannot be 01 or 10, and hence it must be 11. Therefore $\{0000, 0011, 1100, 1111\} \subseteq S$. But then the coordinates 1 and 2 are identical on S, a contradiction. Let x, y be the two points in $S \setminus \{0000, 0011\}$. We have $x_{\{3,4\}}, y_{\{3,4\}} \in \{01, 10\}$.

If $x_{\{3,4\}} = y_{\{3,4\}}$, then, after swapping coordinates 3 and 4 if necessary, this common value is 01. Since $\Delta(S) \geq 2$, the points x and y must then be 0101 and 1001. Thus $S = S_4$.

Suppose instead that $x_{\{3,4\}} \neq y_{\{3,4\}}$. Then $\{x_{\{3,4\}}, y_{\{3,4\}}\} = \{01, 10\}$. Neither $x_{\{1,2\}}$ nor $y_{\{1,2\}}$ is 00 since $x, y \in S \setminus \{0000, 0011\}$ and $\Delta(S) \geq 2$. If $\{x_{\{1,2\}}, y_{\{1,2\}}\} = \{01, 10\}$, then, up to relabeling, $S = \{0000, 0011, 0101, 1010\}$. In $B^S_{\{1,3\}}$, the left vertices 00 and 11 have neighborhoods $\{00, 11\}$ and $\{00\}$, respectively. These two left vertices have a common neighbor but different neighborhoods, so this graph is not a union of bicliques, contradicting that P is simple*. Hence, one of the points x, y satisfies $x_{\{1,2\}} = 11$ or $y_{\{1,2\}} = 11$. Up to swapping coordinates 3 and 4, we may assume that $1110 \in S$. The remaining point must then be 0101 or 1001, and a further swap of coordinates 1 and 2 if necessary, gives 0101. Hence, S is a relabeling of S_5. ◀

▶ **Lemma 23.** *Let P be a simple* non-reducible 4-party boolean distribution, and let $S := supp(P)$. If $|S| = 5$, then, up to relabeling, $S = S_3$.*

Proof. By Lemma 19, $\Delta(S) \geq 2$, no coordinate is constant on S, and no two coordinates are identical or complementary on S.

Consider the support graph $B^S_{\{1,2\}}$. By Lemma 21, every connected component is one of $K_{1,1}$, $K_{1,2}$, $K_{2,1}$, or $K_{2,2}$. Let n_{11} be the number of $K_{1,1}$ components, let n_{12} be the number of components which are either $K_{1,2}$ or $K_{2,1}$, and let n_{22} be the number of $K_{2,2}$ components. Counting edges gives $n_{11}+2n_{12}+4n_{22} = 5$, and counting vertices gives $2n_{11}+3n_{12}+4n_{22} \leq 8$. The only nonnegative integer solutions are $(n_{11}, n_{12}, n_{22}) = (1, 0, 1)$ and $(1, 2, 0)$.

Assume first that $(n_{11}, n_{12}, n_{22}) = (1, 0, 1)$. Because $\Delta(S) \geq 2$, the two left vertices in the $K_{2,2}$ component are opposite vertices of $\{0, 1\}^2$, and the same holds on the right. Up to relabeling, we may therefore assume that $S = \{0000\} \cup \{0101, 0110, 1001, 1010\}$. In $B^S_{\{1,3\}}$, the left vertices 00 and 11 have neighborhoods $\{00, 11\}$ and $\{00\}$, respectively. These two left vertices have a common neighbor but different neighborhoods, so this graph is not a union of bicliques, contradicting that P is simple*. Hence $(n_{11}, n_{12}, n_{22}) = (1, 2, 0)$. Up to relabeling, this forces $S = \{0001, 0010\} \cup \{0100, 1000\} \cup \{1111\}$. Flipping coordinate 4 maps this set to S_3. ◀

▶ **Lemma 24.** *Let P be a simple* non-reducible 4-party boolean distribution, and let $S := supp(P)$. If $|S| = 6$, then, up to relabeling, $S = S_2$.*

Proof. By Lemma 19, $\Delta(S) \geq 2$, no coordinate is constant on S, and no two coordinates are identical or complementary on S.

Consider the support graph $B^S_{\{1,2\}}$. By Lemma 21, every connected component is one of $K_{1,1}$, $K_{1,2}$, $K_{2,1}$, or $K_{2,2}$. Let n_{11}, n_{12}, and n_{22} be the corresponding component counts. Counting edges gives $n_{11}+2n_{12}+4n_{22} = 6$, and counting vertices gives $2n_{11}+3n_{12}+4n_{22} \leq 8$. The only nonnegative integer solutions are $(n_{11}, n_{12}, n_{22}) = (0, 1, 1)$ and $(2, 0, 1)$.

Assume first that $(n_{11}, n_{12}, n_{22}) = (0, 1, 1)$. Because $\Delta(S) \geq 2$, the two left vertices in the $K_{2,2}$ component are opposite, and the same holds on the right. Up to relabeling, we may assume that $S = \{0000, 0011\} \cup \{0101, 0110, 1001, 1010\}$. In $B^S_{\{1,3\}}$, the left vertices 01 and 10 have neighborhoods $\{01, 10\}$ and $\{01\}$, respectively. These two left vertices have a common neighbor but different neighborhoods, so this graph is not a union of bicliques, contradicting that P is simple*.

Hence $(n_{11}, n_{12}, n_{22}) = (2, 0, 1)$. Up to relabeling, this forces

$S = \{0011\} \cup \{1100\} \cup \{0101, 0110, 1001, 1010\}$. Flipping coordinates 2 and 3 maps this set to S_2. ◀

▶ **Lemma 25.** *There is no set $S \subseteq \{0,1\}^4$ with $|S| = 7$ such that $\Delta(S) \geq 2$ and B^S_J is a union of bicliques for every $J \subseteq [4]$ with $|J| = 2$.*

Proof. Consider the support graph $B^S_{\{1,2\}}$. By Lemma 21, every connected component is one of $K_{1,1}$, $K_{1,2}$, $K_{2,1}$, or $K_{2,2}$. Let n_{11} be the number of $K_{1,1}$ components, let n_{12} be the number of $K_{1,2}$ or $K_{2,1}$ components, and let n_{22} be the number of $K_{2,2}$ components. Counting edges gives $n_{11} + 2n_{12} + 4n_{22} = 7$, while counting vertices gives $2n_{11} + 3n_{12} + 4n_{22} \leq 8$. Subtracting yields $n_{11} + n_{12} \leq 1$.

If $n_{22} \geq 2$, then $|S| \geq 8$, a contradiction. If $n_{22} = 1$, then $n_{11} + 2n_{12} = 3$, but $n_{11} + n_{12} \leq 1$ implies $n_{11} + 2n_{12} \leq 2$, a contradiction. If $n_{22} = 0$, then $n_{11} + 2n_{12} = 7$, but again $n_{11} + 2n_{12} \leq 2$, a contradiction. Thus no such set S exists. ◀

▶ **Lemma 26.** *Let $S \subseteq \{0,1\}^4$ be such that $\Delta(S) \geq 2$. Then $|S| \leq 8$. Moreover, if $|S| = 8$, then, up to relabeling, $S = S_1$.*

Proof. Join two strings in $\{0,1\}^4$ by an edge if they differ in exactly one coordinate. Since $\Delta(S) \geq 2$, no such edge has both endpoints in S. Every point of S has exactly four neighbors in $\{0,1\}^4$, so the number of edges with one endpoint in S and the other in $\{0,1\}^4 \setminus S$ is $4|S|$. On the other hand, every point of $\{0,1\}^4 \setminus S$ is incident to at most four such edges, so

$$4|S| \leq 4(16 - |S|).$$

Hence $|S| \leq 8$.

Now assume that $|S| = 8$. Then equality holds above, so every point of $\{0,1\}^4 \setminus S$ is adjacent to four points of S. In particular, $\Delta(\{0,1\}^4 \setminus S) \geq 2$. Thus $\{0,1\}^4$ is partitioned into two sets of size 8, with $\Delta(S) \geq 2$ and $\Delta(\{0,1\}^4 \setminus S) \geq 2$. Since every edge of $\{0,1\}^4$ joins an even-weight vector and an odd-weight vector, membership in S alternates with parity along every edge. Therefore S is exactly one parity class. If necessary, flipping one coordinate maps the odd-parity class to the even-parity class, which is S_1. ◀

Proof of Proposition 17. Let $S := \mathrm{supp}(P)$. By Lemma 19, $\Delta(S) \geq 2$, no coordinate is constant on S, and no two coordinates are identical or complementary on S. By Lemma 20, $|S| \geq 4$. By Lemma 25, $|S| \neq 7$. By Lemma 26, $|S| \leq 8$. Thus $|S| \in \{4, 5, 6, 8\}$. If $|S| = 4$, apply Lemma 22. If $|S| = 5$, apply Lemma 23. If $|S| = 6$, apply Lemma 24. If $|S| = 8$, apply Lemma 26. ◀

5 Securely Sampling Non-Reducible Simple* 4-Party Boolean Distributions

To complete the proof of Theorem 1, it remains to show that all non-reducible simple* 4-party boolean distributions are securely samplable. By Proposition 17 and Lemma 16, it suffices to consider simple* distributions whose support is one of the 5 admissible supports S_1, S_2, S_3, S_4, S_5.

The argument in this section has three parts. First, we show that any "product-form distribution" is statistically securely samplable, provided its support admits a "semi-secure membership-test"; both these notions are defined below and the result is stated as Lemma 29. Next, as stated in Lemma 30, we show that each of the five admissible supports admits a semi-secure membership-test because each is additively representable. Finally, we show that every simple* distribution supported on one of the five admissible supports is a product-form distribution; this is stated in Lemma 31.

We present the above definitions and the statements of these three lemmas below.

▶ **Definition 27** (Product-form distribution). *A distribution P on $\mathcal{X}_1 \times \ldots \times \mathcal{X}_n$ is said to be a* product-form distribution *if there exist a nonempty set $S \subseteq \mathcal{X}_1 \times \ldots \times \mathcal{X}_n$ and weight functions $w_i : \mathcal{X}_i \to \mathbb{R}^+$ such that $supp(P) = S$ and, for every $x \in S$,*

$$P(x_1, \ldots, x_n) = \frac{1}{Z} \prod_{i=1}^{n} w_i(x_i) \tag{6}$$

where $Z := \sum_{(x_1,\ldots,x_n)\in S} \prod_{i=1}^{n} w_i(x_i)$ is a normalization factor.

▶ **Definition 28** (Semi-secure Membership-Test). *An n-party protocol Π is said to be a* semi-secure membership-test *for a set $S \subseteq \mathcal{X}_1 \times \ldots \times \mathcal{X}_n$ if, on running Π with inputs $x_i \in \mathcal{X}_i$ to each party, all parties learn whether $(x_1, \ldots, x_n) \in S$, and further, for all $C \subseteq [n]$, the joint views of the parties in C are identically distributed on executions of Π with inputs $x, x' \in S$ such that $x_C = x'_C$.*

Note that above, there is no privacy guarantee for inputs not in S.

We prove the following three lemmas in Sections 5.1–5.3.

▶ **Lemma 29.** *Let P be an n-party product-form distribution with support $S \subseteq \mathcal{X}_1 \times \ldots \mathcal{X}_n$. Also, suppose there exists a semi-secure membership-test for S. Then P is securely samplable.*

More precisely, for any $\epsilon > 0$, there is an ϵ-secure sampling protocol for P that invokes the membership-test $O(\log 1/\epsilon)$ times.

▶ **Lemma 30** (Admissible Supports have Semi-Secure Membership-tests). *Each of the supports $S_1, \ldots, S_5$ admits a semi-secure membership test.*

▶ **Lemma 31** (Non-Reducible Simple* Distributions are Product-Form). *Let P be a 4-party simple* boolean distribution whose support is one of $S_1, \ldots, S_5$. Then P is a product-form distribution.*

Combining Lemmas 29–31 with Proposition 17 and Lemma 16, we directly obtain the following, which is then used to complete the proof of Theorem 1.

▶ **Lemma 32.** *Every non-reducible simple* 4-party boolean distribution is securely samplable.*

Proof. Let P be such a distribution. Then by Proposition 17 and Lemma 16, it can be relabeled to a simple* boolean distribution P' whose support is one of the admissible supports S_1, S_2, S_3, S_4, S_5. Then, by Lemmas 30 and 31, P' satisfies the requirements of Lemma 29, and hence is securely samplable. By Lemma 16, so is P. ◀

Proof of Theorem 1. Let P be a simple* 4-party boolean distribution. If P is reducible, then by Lemma 14, it is securely samplable. On the other hand if P is non-reducible, by Lemma 32, it is securely samplable. Thus every simple* 4-party boolean distribution is securely samplable. The converse holds by Lemma 8. ◀

In the rest of this section, we prove Lemmas 29–31.

5.1 Secure Sampling via Rejection Sampling

A distribution in product-form is amenable to being sampled using a rejection sampling procedure:

Rejection Sampling Step. Given a distribution P_{X} in product-form with support $S \subseteq \mathcal{X}_1 \times \ldots \times \mathcal{X}_n$ and weight functions $w_i : \mathcal{X}_i \to \mathbb{R}^+$.

- **Independent Sampling:** For $i = 1$ to n, independently sample X_i such that $\Pr[X_i = x] = w_i(x)/Z_i$, where $Z_i = \sum_{x \in \mathcal{X}_i} w_i(x)$.
- **Accept or Reject:** If the values sampled $(x_1, \ldots, x_n)$ are such that $(x_1, \ldots, x_n) \in S$, output it, else output $\perp$.

Note that the probability of accepting is $\sum_{(x_1,\ldots,x_n)\in S} \prod_{i=1}^{n} \frac{w_i(x_i)}{Z_i} = \frac{Z}{\prod_i Z_i}$.

It is easy to see that conditioned on accepting, the output is distributed according to P_{X}.[3] Now consider a process that repeats the rejection sampling step at most k times until a step accepts; if it accepts, then we output the sample in that step, and if it rejects in all k steps, then we output a sample from P_{X}. (Later, we shall implement this procedure using a protocol.)

The output of this entire process is also distributed as P_{X}. Further, the probability of this procedure requiring the final direct sampling step is $(1 - \frac{Z}{\prod_i Z_i})^k$. Note that $0 < Z \leq \prod_i Z_i$ (as follows from the definition of a product-form distribution), and hence this probability decreases exponentially with k.

Proof of Lemma 29. Let the weight functions associated with the product-form distribution P be $w_i : \mathcal{X}_i \to \mathbb{R}^+$, for $i = 1$ to n. Also, let $Z = \sum_{(x_1,\ldots,x_n)\in S} \prod_{i=1}^{n} w_i(x_i)$ and $Z_i = \sum_{x_i \in \mathcal{X}_i} w_i(x_i)$.

Our protocol is to use the rejection procedure above, for some number of iterations k (to be specified below), with the rejection step implemented using a membership-test. If the final direct sampling step is required (because all k iterations of the rejection sampling step rejected), this will be carried out by a fixed party.

The view of the adversary consists of an integer t (the number of iterations of the rejection sampling step that are rejected in the protocol), a view of the membership test in the first t iterations, and the view in the final iteration (either an accepting iteration of the rejection sampling step, or a direct sampling step). We argue that this entire view can be perfectly or ϵ-statistically simulated (depending on whether the party designated for direct sampling is corrupt or not). The integer t is distributed according to a fixed geometric distribution and can be simulated. Given t, to simulate the view of the first t iterations, we faithfully simulate the protocol for all parties conditioned on their inputs not being in S; each of these executions is independent of the others and independent of t itself. For the final iteration, if $t < k$, then we simulate the view of the adversary as follows: the x_i sampled by each corrupt party is set to be exactly the ideal output given to the simulator; and, the view of the corrupt parties in the membership test is sampled from the fixed distribution of views when inputs are in the set. In the case $t = k$, there are two possibilities. If the party designated for direct sampling is not corrupt, the view simply consists of the outputs x_i sent to each corrupt party; however, if the designated party is corrupt, the simulation fails.

This simulation is perfect when the designated party is not corrupt. In case it is corrupt, we note that conditioned on $t < k$ (in both simulation and real execution), the simulation is perfect; so the statistical difference between the simulated view and the real view is at

[3] Probability of the procedure outputting $(x_1, \ldots, x_n) \in S$ is $\frac{\prod_i w_i(x_i)}{\prod_i Z_i}$, and the probability of accepting is $\frac{Z}{\prod_i Z_i}$. The conditional probability is their ratio, which is the desired probability $\frac{\prod_i w_i(x_i)}{Z}$.

most the probability that $t = k$, which corresponds to all k iterations of rejection sampling rejecting. This, as discussed earlier, has probability α^k where $\alpha = 1 - \frac{Z}{\prod_i Z_i} < 1$. To make this simulation error at most ϵ, we set $k \geq \log(1/\epsilon)/\log(1/\alpha) = O(\log(1/\epsilon))$ (since α is a constant that depends only on the given distribution P). ◀

5.2 Semi-Secure Membership-Test for Admissible Supports

To prove Lemma 30, we rely on the following observation.

▶ **Lemma 33** (Semi-secure membership-test protocol for an additively representable set)**.** *Let $S \subseteq \mathcal{X}_1 \times \ldots \times \mathcal{X}_n$ be additively representable (Definition 2). Then S admits a semi-secure membership test.*

Proof. Since S is additively representable, there is a finite abelian group G, maps $\mu_i : \mathcal{X}_i \to G$ and an element $g \in G$ such that

$$S = \Big\{ x \in \mathcal{X}_1 \times \ldots \times \mathcal{X}_n : \sum_{i=1}^{n} \mu_i(x_i) = g \Big\}.$$

The semi-secure membership test proceeds as follows: Each party with input x_i locally computes $\mu_i(x_i) \in G$. A standard perfectly secure protocol for addition in G allows the parties to compute the sum $\sum_{i=1}^{n} \mu_i(x_i)$. They output 1 if and only if the revealed sum equals g. If the input lies in S, then the revealed sum is always g, and so on such inputs the protocol reveals nothing beyond membership in S. No security is required when the input lies outside S. ◀

Now to prove that the admissible supports in Equation (5) have semi-secure membership tests, we can show that they are additively representable. We do this below.

Proof of Lemma 30. Each of the five admissible supports is additively representable. In particular,

$$\begin{aligned}
S_1 &= \{x \in \{0,1\}^4 : \ x_1 + x_2 + x_3 + x_4 \equiv 0 \pmod 2\},\\
S_2 &= \{x \in \{0,1\}^4 : \ x_1 - x_2 - x_3 + x_4 \equiv 0 \pmod 6\},\\
S_3 &= \{x \in \{0,1\}^4 : \ x_1 + x_2 + x_3 - x_4 \equiv 0 \pmod 3\},\\
S_4 &= \{x \in \{0,1\}^4 : \ x_1 + x_2 + x_3 - x_4 \equiv 0 \pmod 6\},\\
S_5 &= \{x \in \{0,1\}^4 : \ 4x_1 + x_2 + x_3 - x_4 \equiv 0 \pmod 6\}.
\end{aligned}$$

A direct check on the 16 points of $\{0,1\}^4$ shows that each displayed congruence holds exactly on the corresponding support. Note that each equation corresponds to an additive representation where multiplying by a coefficient c corresponds to the local map $\mu(0) = 0, \mu(1) = c$, and the modular addition corresponds to addition in an abelian group. Therefore each S_i is additively representable, and the conclusion follows from Lemma 33. ◀

5.3 Proof of Lemma 31: Case Analysis of Admissible Supports

We shall invoke the following observation in the arguments below.

▶ **Lemma 34** (The $K_{2,2}$ constraint)**.** *If $P_{\mathrm{U,V}}$ is a simple two-party distribution over $\mathcal{U} \times \mathcal{V}$ such that there exist $u_0, u_1 \in \mathcal{U}$ and $v_0, v_1 \in \mathcal{V}$ with $P_{\mathrm{U,V}}(u_i, v_j) > 0$ for all $i, j \in \{0,1\}$ (i.e., the biclique $\{u_0, u_1\} \times \{v_0, v_1\}$ is a subgraph of the support graph), then*

$$P_{\mathrm{U,V}}(u_0, v_0) P_{\mathrm{U,V}}(u_1, v_1) = P_{\mathrm{U,V}}(u_0, v_1) P_{\mathrm{U,V}}(u_1, v_0). \tag{7}$$

Proof. Since $P_{\mathrm{U,V}}(u_i, v_j) > 0$, we have $P_{\mathrm{U,V}}(u_i, v_j) = w_1(u_i)w_2(v_j)$ for some weight functions w_1, w_2 as required in Equation (3). Then the LHS and RHS of Equation (7) are both $\prod_{i=0}^{1} w_1(u_i)w_2(v_i)$. ◀

▶ **Lemma 35** (Support S_1). *If P is a simple* 4-party boolean distribution with $supp(P) = S_1$, then it is a product-form distribution.*

Proof. Recall that $S_1 = \{0000, 0011, 0101, 0110, 1001, 1010, 1100, 1111\}$. Note that for all the three subsets up to complement $J \subset [4]$ with $|J| = 2$, namely, $\{1,2\}, \{1,3\}$ and $\{1,4\}$, the support graph $B_J^{S_1}$ has two $K_{2,2}$ components, one of them defined on $\{00, 11\} \times \{00, 11\}$ and the other one defined on $\{01, 10\} \times \{01, 10\}$. Applying Lemma 34 to the components in $B_{\{1,2\}}^{S_1}$ gives

$$P(0000)P(1111) = P(0011)P(1100), \quad P(0101)P(1010) = P(0110)P(1001).$$

Symmetrically (since the support is symmetric along coordinates $2, 3$ and 4), for $B_{\{1,3\}}^{S_1}$ and $B_{\{1,4\}}^{S_1}$, we have

$$P(0000)P(1111) = P(0101)P(1010), \quad P(0011)P(1100) = P(0110)P(1001), \text{ and}$$
$$P(0000)P(1111) = P(0110)P(1001), \quad P(0011)P(1100) = P(0101)P(1010)$$

respectively. Now, if P is simple*, then every one of these equalities must hold, and hence we have the following:

$$P(0000)P(1111) = P(0011)P(1100) = P(0101)P(1010) = P(0110)P(1001). \tag{8}$$

For brevity, let $a := P(0000)$, $b := P(1100)$, $c := P(0011)$, $d := P(1010)$, $e := P(0101)$, $f := P(1001)$, $g := P(0110)$ and $h := P(1111)$. Then

$$ah = bc = de = fg. \tag{9}$$

Define weights ω_i as follows: $\omega_1 := \sqrt{\frac{bd}{ag}}$, $\omega_2 := \frac{b}{a\,\omega_1}$, $\omega_3 := \frac{d}{a\,\omega_1}$, $\omega_4 := \frac{f}{a\,\omega_1}$. Then, the following can be verified directly (from the definition of ω_i for b, d, f, and using Equation (9) as well for the rest): $b = a\,\omega_1\omega_2$, $c = a\,\omega_3\omega_4$, $d = a\,\omega_1\omega_3$, $e = a\,\omega_2\omega_4$, $f = a\,\omega_1\omega_4$, $g = a\,\omega_2\omega_3$, $h = a\,\omega_1\omega_2\omega_3\omega_4$. That is, we have for each $x \in S_1$, $P(x) = a\prod_{i=1}^{4}\omega_i^{x_i}$. Defining functions $w_i : \{0,1\} \to \mathbb{R}^+$ as $w_i(\alpha) = \omega_i^{\alpha}$, and noting that $\sum_{x\in S_1} P(x) = 1$, we can see that (6) holds. ◀

▶ **Lemma 36** (Support S_2). *If P is a simple* 4-party boolean distribution with $supp(P) = S_2$, then it is a product-form distribution.*

Proof. Recall that $S_2 = \{0000, 0011, 0101, 1010, 1100, 1111\}$. For $J \in \{\{1,2\}, \{1,3\}\}$, the $K_{2,2}$ component of $B_J^{S_2}$ is on $\{00, 11\} \times \{00, 11\}$. For $B_{\{1,4\}}^{S_2}$, the $K_{2,2}$ component is over $\{01, 10\} \times \{01, 10\}$. Applying Lemma 34 gives, respectively,

$$P(0000)P(1111) = P(0011)P(1100),$$
$$P(0000)P(1111) = P(0101)P(1010),$$
$$P(0011)P(1100) = P(0101)P(1010).$$

If P is simple*, then the displayed equalities must hold, and hence we have

$$P(0000)P(1111) = P(0011)P(1100) = P(0101)P(1010). \tag{10}$$

Now let $a := P(0000)$, $b := P(0011)$, $c := P(0101)$, $d := P(1010)$, $e := P(1100)$, and $f := P(1111)$. Then $af = be = cd$. Set $\omega_1 := 1$, $\omega_2 := \frac{e}{a}$, $\omega_3 := \frac{d}{a}$, and $\omega_4 := \frac{c}{e}$. By construction, $a\,\omega_1\omega_2 = e$, $a\,\omega_1\omega_3 = d$, and $a\,\omega_2\omega_4 = c$. The remaining nontrivial values are $a\,\omega_3\omega_4 = \frac{cd}{e} = b$ and $a\,\omega_1\omega_2\omega_3\omega_4 = \frac{cd}{a} = f$, using $be = cd$ and $af = cd$. Thus $P(x) = a\prod_{i=1}^{4}\omega_i^{x_i}$ for every $x \in S_2$. With $w_i(\alpha) = \omega_i^{\alpha}$, and noting $\sum_{x\in S_2} P(x) = 1$ we get (6). ◀

For the remaining three supports, every distribution on the support already has the product form (6); no additional simple* constraints need to be imposed.

▶ **Lemma 37** (Supports S_3, S_4, S_5)**.** *If P is a 4-party boolean distribution with supp$(P) = S_3, S_4$ or S_5, then P is a product-form distribution.*

Proof. We shall denote the probabilities of the following elements: $a := P(0000), b := P(0011), c := P(0101), d := P(1001),$ and $e := P(1110)$ for all these supports.

Case S_3. Recall that $S_3 = \{0000, 0011, 0101, 1001, 1110\}$. Set $\omega_4 := \left(\frac{bcd}{ea^2}\right)^{1/3}$, $\omega_1 := \frac{d}{a\,\omega_4}$, $\omega_2 := \frac{c}{a\,\omega_4}$, and $\omega_3 := \frac{b}{a\,\omega_4}$. Then $a\,\omega_3\omega_4 = b$, $a\,\omega_2\omega_4 = c$, and $a\,\omega_1\omega_4 = d$ by construction, and $a\,\omega_1\omega_2\omega_3 = a \cdot \frac{d}{a\,\omega_4} \cdot \frac{c}{a\,\omega_4} \cdot \frac{b}{a\,\omega_4} = \frac{bcd}{a^2\omega_4^3} = e$. Thus $P(x) = a\prod_{i=1}^{4}\omega_i^{x_i}$ for every $x \in S_3$, and with $w_i(\alpha) = \omega_i^{\alpha}$, we get (6).

Case S_4. Recall that $S_4 = \{0000, 0011, 0101, 1001\}$. Set $\omega_1 := \frac{d}{a}$, $\omega_2 := \frac{c}{a}$, $\omega_3 := \frac{b}{a}$, and $\omega_4 := 1$. Then $a\,\omega_3\omega_4 = b, a\,\omega_2\omega_4 = c,$ and $a\,\omega_1\omega_4 = d$, so $P(x) = a\prod_{i=1}^{4}\omega_i^{x_i}$ for every $x \in S_4$, and with $w_i(\alpha) = \omega_i^{\alpha}$, we get (6).

Case S_5. Recall that $S_5 = \{0000, 0011, 0101, 1110\}$. Set $\omega_1 := \frac{ae}{bc}$, $\omega_2 := \frac{c}{a}$, $\omega_3 := \frac{b}{a}$, and $\omega_4 := 1$. Then $a\,\omega_2\omega_4 = c$, $a\,\omega_3\omega_4 = b$, and $a\,\omega_1\omega_2\omega_3 = e$. Hence, $P(x) = a\prod_{i=1}^{4}\omega_i^{x_i}$ for every $x \in S_5$, and with $w_i(\alpha) = \omega_i^{\alpha}$, we get (6). ◀

Proof of Lemma 31. The proof follows immediately from Lemma 35, Lemma 36 and Lemma 37. ◀

References

1 Saugata Basu, Hamidreza Amini Khorasgani, Hemanta K. Maji, and Hai H. Nguyen. Geometry of secure two-party computation. In *63rd IEEE Annual Symposium on Foundations of Computer Science, FOCS 2022, Denver, CO, USA, October 31 - November 3, 2022*, pages 1035–1044. IEEE, 2022. doi:10.1109/FOCS54457.2022.00101.

2 Saugata Basu, Hamidreza Amini Khorasgani, Hemanta K. Maji, and Hai H. Nguyen. Solving linear inequalities over the space of convex sets & its applications to cryptography and hydrodynamics. In *66th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2025, Sydney, Australia, December 14-17, 2025*, pages 2369–2380. IEEE, 2025. doi:10.1109/FOCS63196.2025.00124.

3 Donald Beaver. Foundations of secure interactive computing. In *Advances in Cryptology — CRYPTO '91*, pages 377–391. Springer, 1991. doi:10.1007/3-540-46766-1_31.

4 Michael Ben-Or, Shafi Goldwasser, and Avi Wigderson. Completeness theorems for non-cryptographic fault-tolerant distributed computation. In *Proceedings of the 20th Annual ACM Symposium on Theory of Computing (STOC)*, pages 1–10. ACM, 1988.

5 David Chaum, Claude Crépeau, and Ivan Damgård. Multiparty unconditionally secure protocols. In *Proceedings of the 20th Annual ACM Symposium on Theory of Computing (STOC)*, pages 11–19. ACM, 1988.

6 Benny Chor and Yuval Ishai. On privacy and partition arguments. *Inf. Comput.*, 167(1):2–9, 2001. doi:10.1006/inco.2000.3013.

7 Oded Goldreich, Silvio Micali, and Avi Wigderson. How to play any mental game or a completeness theorem for protocols with honest majority. In *Proceedings of the 19th Annual ACM Symposium on Theory of Computing (STOC)*, pages 218–229. ACM, 1987. doi:10.1145/28395.28420.

8 Joe Kilian. Founding cryptography on oblivious transfer. In *Proceedings of the 20th Annual ACM Symposium on Theory of Computing (STOC)*, pages 20–31. ACM, 1988. doi:10.1145/62212.62215.

9 Joe Kilian. More general completeness theorems for secure two-party computation. In *Proceedings of the 32nd Annual ACM Symposium on Theory of Computing (STOC)*, pages 316–324. ACM, 2000. doi:10.1145/335305.335342.

10 Eyal Kushilevitz. Privacy and communication complexity. *SIAM Journal on Discrete Mathematics*, 5(2):273–284, 1992. Originally presented at FOCS 1989. doi:10.1137/0405021.

11 Hemanta K. Maji, Manoj Prabhakaran, and Mike Rosulek. Complexity of multi-party computation functionalities. In Manoj Prabhakaran and Amit Sahai, editors, *Secure Multi-Party Computation*, volume 10 of *Cryptology and Information Security Series*, pages 249–283. IOS Press, 2013. doi:10.3233/978-1-61499-169-4-249.

12 Manoj Prabhakaran and Vinod M. Prabhakaran. On secure multiparty sampling for more than two parties. In *2012 IEEE Information Theory Workshop, Lausanne, Switzerland, September 3-7, 2012*, pages 99–103. IEEE, 2012. doi:10.1109/ITW.2012.6404773.

13 Vinod M. Prabhakaran and Manoj M. Prabhakaran. Assisted common information with an application to secure two-party sampling. *IEEE Transactions on Information Theory*, 60(6):3413–3434, 2014. doi:10.1109/TIT.2014.2316011.

A Beyond 4 Parties: A 6-Party Simple* Distribution without Additive Representation

In Section 5, our positive results for 4-party boolean simple* distributions relied on the fact that all 5 admissible supports are *additively representable* (Definition 2), which naturally enables a semi-secure membership test. A fundamental question is whether all boolean simple* supports have an additive representation.

In this appendix, we show that this is not the case. We present a 6-party simple* boolean support that is not additively representable, but the distribution uniformly distributed over the support is statistically securely samplable.

A.1 The 6-Party Support S_6

Consider the 6-party boolean distribution uniformly distributed over the support $S_6 \subset \{0,1\}^6$, where $S_6 = \{000000, 101100, 110010, 010101, 001011, 111001, 100111\}$. One can verify that this distribution is simple* by checking every bipartite support graph across bipartitions.

▶ **Lemma 38.** *The support S_6 is not additively representable.*

Proof. Assume for contradiction that S_6 is additively representable. That is, there exist maps $\mu_i : \{0,1\} \to G$, where G is some finite abelian group G, and a target element $g \in G$, such that $x \in S_6$ if and only if $\sum_{i=1}^{6} \mu_i(x_i) = g$.

Since $000000 \in S_6$, we have $\sum_{i=1}^{6} \mu_i(0) = g$. We can shift the maps without loss of generality by defining $\tilde{\mu}_i(x_i) = \mu_i(x_i) - \mu_i(0)$. This ensures that $\tilde{\mu}_i(0) = 0$ for all i, and the target natively becomes $\tilde{g} = 0$. Thus, $x \in S_6 \iff \sum_{i=1}^{6} \tilde{\mu}_i(x_i) = 0$.

Let $a_i = \tilde{\mu}_i(1) \in G$. The inclusion of the sequence of six non-zero points of S_6 yields the following additive relations in G:

$$a_1 + a_3 + a_4 = 0 \tag{11}$$
$$a_1 + a_2 + a_5 = 0 \tag{12}$$
$$a_2 + a_4 + a_6 = 0 \tag{13}$$
$$a_3 + a_5 + a_6 = 0 \tag{14}$$
$$a_1 + a_2 + a_3 + a_6 = 0 \tag{15}$$
$$a_1 + a_4 + a_5 + a_6 = 0 \tag{16}$$

Adding equations (11), (12), (13), and (14) and subtracting equations (15) and (16) yields:

$$\begin{aligned}&(a_1 + a_3 + a_4) + (a_1 + a_2 + a_5) + (a_2 + a_4 + a_6) + (a_3 + a_5 + a_6)\\ &- (a_1 + a_2 + a_3 + a_6) - (a_1 + a_4 + a_5 + a_6)\\ &= a_2 + a_3 + a_4 + a_5 = 0.\end{aligned}$$

This implies that the point 011110 is evaluated to 0, and thus must also belong to the set. However, $011110 \notin S_6$, which yields a contradiction. Consequently, S_6 is not additively representable over any finite abelian group. ◀

A.2 Securely Sampling S_6

Let $m = 011110$ be the specific point that we saw in the proof of Lemma 38. Define the set $T_6 = S_6 \cup \{m\}$, resulting in

$$T_6 = \{000000, 101100, 110010, 010101, 001011, 111001, 100111, 011110\}.$$

Note that T_6 has an additive representation over the group $\mathbb{Z}_2^3$. By setting the target element $g = (0, 0, 0)$ and defining the maps $\mu_i : \{0, 1\} \to \mathbb{Z}_2^3$ such that $\mu_i(0) = (0, 0, 0)$ for all i and the non-zero mappings are $\mu_1(1) = (1, 0, 0)$, $\mu_2(1) = (0, 1, 0)$, $\mu_3(1) = (1, 0, 1)$, $\mu_4(1) = (0, 0, 1)$, $\mu_5(1) = (1, 1, 0)$, and $\mu_6(1) = (0, 1, 1)$, the set T_6 is defined exactly by the single equation:

$$\sum_{i=1}^{6} \mu_i(x_i) = (0, 0, 0)$$

Because T_6 is additively representable, it admits a semi-secure membership test (by Lemma 34), and the uniform distribution supported on T_6 is securely samplable.

We construct the secure sampling protocol for uniform distribution over S_6 via the following protocol:

1. The 6 parties sample a joint outcome $\mathrm{X} = (\mathrm{X}_1, \ldots, \mathrm{X}_6)$ uniformly spanning T_6 enabled through a semi-secure membership test of T_6.
2. Let $d = \mathrm{X} \oplus m$. Note that $\mathrm{X} = m$ if and only if $d = 000000$. Every party i samples an independent, uniformly random bit $r_i \in \{0, 1\}$, and computes $u_i = r_i d_i$.
3. The parties securely compute the sum $\beta = \bigoplus_{i=1}^{6} u_i$ over $\mathbb{Z}_2$ and reveal it via a standard secure summation protocol.
4. If $\beta = 1$, they accept X and terminate. Otherwise, they reject and repeat.

▶ **Theorem 39.** *The uniform distribution over S_6 is statistically securely samplable using the protocol described above.*

Proof. First, we analyze the output distribution. In step 1, the protocol samples X uniformly from T_6. If $\mathrm{X} = m$, then $d = \mathrm{X} \oplus m = 000000$. This implies $u_i = 0$ for all $i \in \{1, \ldots, 6\}$, and the revealed sum is always $\beta = 0$, so the sample is rejected. If $\mathrm{X} \in S_6$, then $d \neq 000000$, and there is at least one coordinate i where $d_i = 1$. Since r_i is sampled uniformly and independently in step 2, the term $u_i = r_i d_i$ is uniformly and independently distributed over $\{0, 1\}$ for every such coordinate i. This makes β uniformly distributed as well. Hence, every $\mathrm{X} \in S_6$ is accepted with probability exactly $1/2$. Conditioned on acceptance, the output is exactly the uniform distribution over S_6.

Next, we establish privacy. Fix a corrupt coalition $C \subset \{1, \ldots, 6\}$. We construct a simulator for the view of C. On a given output X_C, the simulator generates the local random bits r_C uniformly. It remains to simulate the transcript of the secure summation for the accepted condition $\beta = 1$.

Let $D = \{x \oplus m \mid x \in T_6\}$. Observe two properties of D:

1. D is a linear space under bitwise XOR.
2. For any two distinct, non-zero vectors $d, e \in D$, their sets of non-zero coordinates intersect. That is, no two non-zero vectors in D have disjoint supports.

In the real protocol, the honest parties' contribution to β is $\bigoplus_{i \notin C} u_i = \bigoplus_{i \notin C} r_i d_i$. The secure summation transcript is perfectly simulatable from the corrupt inputs u_C and the final sum $\beta = 1$, provided that the honest contribution to β is uniformly random. This uniformity holds as long as $d_{C^c} \neq \mathbf{0}$ (meaning that there is at least one honest party $i \notin C$ with $d_i = 1$).

Suppose, for contradiction, that there exist two valid outcomes $x, y \in S_6$ corresponding to the same output $\mathrm{X}_C = x_C = y_C$ for the coalition, and it happens that $d = x \oplus m$ satisfies $d_{C^c} = \mathbf{0}$. This means d is entirely supported on C. Let $e = y \oplus m$. Because $x_C = y_C$, the vector $d \oplus e = x \oplus y$ is zero on C, meaning it is entirely supported on C^c. By property 1, both d and $d \oplus e$ are in D. Since their supports are disjoint, property 2 implies that one must be the zero vector. Since $x \in S_6$, it follows that $d \neq 000000$, and therefore $d \oplus e = 000000$, which implies $x = y$.

Therefore, for any given output X_C for the coalition, exactly one of two cases applies:

- $d_{C^c} \neq \mathbf{0}$ for all valid x consistent with X_C. In this case, the honest parties always supply an independent, uniform bit to β. The simulator simply sets $\beta = 1$ and perfectly simulates the secure sum transcript.
- X_C is consistent with exactly one outcome $x \in S_6$. In this case, the simulator knows the entire vector X, and can simulate the secure sum transcript by simulating the honest parties as well.

Furthermore, rejected iterations are independent and the view for such iterations can be simulated by simulating all parties conditional on $\beta = 0$. The statistical distance between the simulated view and the real view approaches zero exponentially with the number of rounds. This satisfies the security requirement. ◀

Compressing Correlations via Secret Replication: PCFs from Symmetric Cryptography

Yuval Ishai
Technion, Haifa, Israel
Amazon Web Services, New York, NY, USA

Hugo Krawczyk
Amazon Web Services, New York, NY, USA

Tal Rabin
Amazon Web Services, New York, NY, USA

Abstract

We revisit the question of securely compressing multiparty correlations using only symmetric cryptography. A *linear correlation* $\mathcal{C}$, defined by a linear subspace $C \subseteq \mathbb{F}^n$, samples a secret random $\mathbf{c} \in C$ and assigns to each party a fixed subset of the entries of $\mathbf{c}$. Gilboa and Ishai (Crypto 1999) and Cramer, Damgård and Ishai (TCC 2005) provide a general technique for securely compressing many independent samples from $\mathcal{C}$ by replicating independent keys of a pseudorandom function (PRF) among the parties. This implies a *pseudorandom correlation function* (PCF) for $\mathcal{C}$ from any PRF, where the PCF key size scales with the number of minimal-support codewords in C.

We observe that the above generalizes to other types of useful target correlations $\mathcal{C}_T$ by using a *secret* replication pattern obtained via a random secret assignment of parties in $\mathcal{C}$ to parties in $\mathcal{C}_T$.

We present several corollaries of this general blueprint. These include a re-derivation of two-party PCF constructions for VOLE and subfield-VOLE over small domains (Roy, Crypto 2022) as well as new multiparty PCFs for small-domain VOLE-style correlations, including scalar-vector multiplication triples and their authenticated variants. Finally, we discuss applications to secure computation.

2012 ACM Subject Classification Theory of computation → Cryptographic primitives; Theory of computation → Cryptographic protocols

Keywords and phrases Pseudorandom correlation functions, correlated randomness, secure computation, symmetric cryptography

Digital Object Identifier 10.4230/LIPIcs.ITC.2026.7

Related Version *Full Version*: `https://eprint.iacr.org/2026/1355`

Funding *Yuval Ishai*: Work done while at Amazon Web Services.

1 Introduction

Never Underestimate the Power of Replication.

Jacob Jacobson

Secure multiparty computation (MPC) without an honest majority inherently requires public-key cryptography. However, this is no longer the case if the parties have access to secret, correlated randomness. That is, we imagine a trusted dealer who generates a sample $(r_1, \ldots, r_n)$ from a correlated randomness distribution $\mathcal{C}$, and sends each r_i to party P_i before the inputs are known. Many MPC protocols can benefit from vast amounts of such correlated randomness, generated during a preprocessing phase, to realize fast and "information-theoretic" online protocols [5, 6, 23].

To mitigate the communication costs and trust assumptions associated with distributing this correlated randomness directly by an external party, pseudorandom correlation generators (PCGs) and pseudorandom correlation functions (PCFs) enable a *secure compression* of

correlated randomness. A PCG [12, 14] allows the parties to locally expand short, correlated keys into a polynomially larger source of correlated pseudorandomness, whereas a PCF [15] extends this to enable random access to a virtually unlimited source. PCGs and PCFs are the correlated randomness analogues of pseudorandom generators (PRGs) and pseudorandom functions (PRFs), respectively. Crucially, the correlated keys viewed by any subset of the parties should not reveal more information about the outputs of the remaining parties beyond what the ideal correlation allows. The correlated keys can be securely generated by an offline protocol, whose cost is amortized away in the online phase. While the combined offline/online protocol is no longer information-theoretic, it still inherits the simplicity and low online cost of a protocol that has an ideal access to $\mathcal{C}$.

The above paradigm motivates the design of PCGs and PCFs for useful multiparty correlations under the weakest possible assumptions and with the best concrete efficiency. While this is a very active area of research (see Section 1.4), current results still leave much room for improvement. Most constructions for useful correlations, such as random oblivious transfer, are based on different flavors of the Learning Parity with Noise (LPN) assumption [12, 14, 15, 19] or alternatively on public-key cryptography assumptions based on lattices [24, 14, 18] or groups [39, 20]. Since a PCG for nontrivial correlations implies a PRG, it is natural to ask when the converse is true:

Which correlations can be compressed using symmetric cryptography?

Here, the term "symmetric cryptography" can stand for a PRG or a PRF, which from a feasibility viewpoint are both equivalent to a one-way function [28, 30].[1]

For the case of two-party correlations, Boyle et al. [14] show that any PRG implies a PCG (in fact, a PCF) for a single long instance of a *unit-vector correlation*, an additively shared vector containing 1 in a random position and 0's elsewhere. Roy [40] obtained a similar result for a *vector oblivious linear evaluation* (VOLE) correlation over a small field $\mathbb{F}$, where a "sender" party gets random long vectors $\mathbf{a}, \mathbf{b} \in \mathbb{F}^N$ and a "receiver" party gets a random scalar $x \in \mathbb{F}$ along with $\mathbf{a}x + \mathbf{b}$. This was generalized to the case where only the scalar x comes from a small domain [36], and used as a basis for low-communication oblivious transfer [40], non-interactive zero-knowledge proofs and post-quantum digital signatures [4, 3], threshold signatures [36, 35], and arithmetic garbling [31]. These isolated positive results call for a broader study of the possibility of compressing useful correlations using only symmetric cryptography.

1.1 Our Contribution

We obtain feasibility results for compressing certain types of correlated randomness using only symmetric cryptography. These are motivated by several applications we discuss in Section 1.2 below. Our results include a conceptually simple explanation for the VOLE-related results from [40] discussed above, as well as solutions for other useful related correlations that were not covered by previous works.

The results are based on a new general framework that utilizes *secure information-theoretic reductions* between correlations. Our main observation is that several useful and inherently nonlinear target correlations $\mathcal{C}_T$ can be compressed via a perfectly secure non-interactive reduction to a *linear* source correlation $\mathcal{C}_S$, namely one obtained by drawing a random secret

[1] A broader interpretation of this term allows for the use of a random oracle. Here we will only use assumptions equivalent to a one-way function.

codeword $\mathbf{c}$ from a publicly known linear code $C_S \subseteq \mathbb{F}^n$ and assigning a coordinate of $\mathbf{c}$ to each of the n parties. More generally, a linear correlation can assign any set of coordinates to each party, as long as this set is fixed and public.

The core mechanism of this reduction is a random projection, which assigns to each party in $\mathcal{C}_T$ a *secretly chosen* subset of virtual parties from the source correlation $\mathcal{C}_S$. We combine such a projection reduction with an existing PCF for linear correlations (Lin-PCF) due to Gilboa and Ishai [25] and Cramer, Damgård and Ishai [22] (see Section 1.4), which is based on a public replication of secret PRF keys. Using this combination, we obtain PCFs for useful *nonlinear* correlations that use *secretly replicated* PRF keys.

Overview of results. We present several concrete instances of the above general blueprint, revisiting previous results and expanding the class of useful correlations that can be compressed using symmetric cryptography. Our positive results apply to different flavors of "VOLE-style" correlations, namely ones that support secure scalar-vector multiplication. While their time complexity is polylogarithmic in the field size, it is polynomial in the scalar domain. This restricts scalars to come from a feasible-size domain. Concretely, our main results are summarized as follows:

- **Two-party VOLE:** We re-derive the previous PCF from [40, 36] for the VOLE correlation, also allowing the scalar to be restricted to any subset $X \subseteq \mathbb{F}$. This relies on an n-party linear correlation $\mathcal{C}_S$ obtained from a Reed-Solomon code of length $n = |X|$ defined by the evaluations of lines $L_{a,b}(\alpha) = a\alpha + b$ on all $\alpha \in X$. The evaluation time of this PCF scales linearly with $|X|$. We extend this to the subfield variant of VOLE, where $\mathbf{a}$ is over a subfield $\mathbb{F}'$ [14, 40].
- **Multiparty SVMT:** We obtain the first PCFs for *multiparty* VOLE-style correlations. Concretely, a k-party *scalar-vector multiplication triple* (SVMT) additively shares between k parties a random vector $\mathbf{a} \in \mathbb{F}^N$, scalar x with shares in $X \subseteq \mathbb{F}$, and their product $\mathbf{a}x$. This is the scalar-vector analogue of a standard Beaver-triple correlation [5]. Here we use a linear correlation $\mathcal{C}_S$ defined by a "multi-line" code of length $n = (k-1) \cdot |X|$, consisting of evaluations of $k-1$ lines L_{a,b_i} with a common slope a on all $\alpha \in X$. Our PCF provides random access to any instance of the SVMT correlation using $O(k \cdot |X|^{k-1})$ PRF calls.
- **Authenticated correlations:** To support authenticated VOLE-style correlations, the vector $\mathbf{a} \in \mathbb{F}^N$ needs to be multiplied by a second scalar y whose additive shares are taken from $Y \subseteq \mathbb{F}$. Our SVMT construction extends to this case, replacing $|X|$ by $|X| \cdot |Y|$ in the cost analysis.

1.2 Applications

Our new PCF constructions naturally support MPC protocols that consume VOLE-style correlations. However, one should account for the limitations on the scalar domain size and the number of parties imposed by our constructions. While these limitations rule out some use cases, many others remain.

Our PCF for k-party SVMT is useful for MPC tasks in which the same secret scalar is multiplied by many secret values or a long secret vector. This is a very common scenario. For example, it applies to any computation that involves matrix-vector multiplications or repeated cryptographic operations with a fixed key. The $|X|^{k-1}$ computational overhead is tolerable when the number of parties k is small and scalars can be taken from a small domain X. For example, the latter applies to standard secret-sharing based MPC for Boolean circuits, where we have $|X| = |\mathbb{F}| = 2$.

The usefulness of small-domain VOLE goes beyond computations over a small domain. This was already demonstrated, for the case of standard 2-party VOLE, by applications to OT extension [40] and threshold cryptography [35], which increase the domain size exponentially via repetition or CRT representation.

Some other applications *natively* involve multiplications of vectors over a large field $\mathbb{F}$ with scalars from a small subset $X \subseteq \mathbb{F}$. For example, in a recent protocol for computing *encrypted matrix-vector product* (EMVP) [8], a client decrypts the product by multiplying a public matrix with a secret decryption vector. Each entry of this vector can potentially come from a small domain, as long as the entire vector has sufficient entropy. Our PCF for k-party SVMT can help distribute an EMVP client.

Finally, while our authenticated SVMT variant is even more limited because of the additional overhead depending on $|Y|$, a small Y suffices for achieving a notion of "covert security" that may provide sufficient deterrence against cheating [2, 36]. See Section 5 for additional discussion of applications.

1.3 Overview of Techniques

Our results use an information-theoretic *projection reduction* of a k-party VOLE-style target correlation $\mathcal{C}_T$ to N instances of an n-party *linear* source correlation $\mathcal{C}_S$, for some $n > k$. This projection reduction can assign each of the k parties in $\mathcal{C}_T$ to a secretly chosen subset of the n parties in $\mathcal{C}_S$, and have each of the k parties apply some local post-processing. Using the Lin-PCF for linear correlations [25, 22], we obtain a PCF for $\mathcal{C}_T$ that involves a *secret replication* of PRF keys.

Beyond the simple special case of generating additive shares of 0, which is used extensively in the context of secure federated learning [9], linear correlations were mostly used for threshold cryptography and general MPC in the *honest-majority* setting [22, 10]. In particular, linear correlations are trivial in the two-party setting. Our results give the first nontrivial applications of linear correlations in the two-party and dishonest-majority settings.

The efficiency of our constructions is closely linked to the specifics of the Lin-PCF, which we briefly explain here. Recall that an instance of an n-party linear correlation $\mathcal{C}_S$ is distributed uniformly over a linear code $C_S \subseteq \mathbb{F}^n$. The Lin-PCF can generate a virtually unbounded number of instances of $\mathcal{C}$ by replicating independently chosen PRF keys between the parties. The replication pattern is determined by a public collection of sets $H \subseteq [n]$. The collection is defined by the *minimal-support sets* $W(C_S)$: For each minimal nonempty set $H \subseteq [n]$ such that there is a codeword $\mathbf{c} \in C_S$ for which H is the set of nonzero coordinates, the set H is in $W(C_S)$. The Lin-PCF distributes the PRF key r_H to each party in H. To generate an instance of $\mathcal{C}_S$, each party evaluates the PRF on its keys, obtaining a fresh field element for each key, and applies a linear combination determined by C_S to the resulting elements. The time required to generate an instance of $\mathcal{C}_S$ using the PCF thus scales linearly with $|W(C_S)|$.

See Figure 1 for a pictorial overview of the combination of the Lin-PCF for $\mathcal{C}_S$ with the secret assignment induced by our projection reduction from $\mathcal{C}_T$ to $\mathcal{C}_S$.

We now describe the concrete projection reductions used for obtaining our PCF for vector oblivious linear evaluation (VOLE) and its multiparty generalization to scalar-vector multiplication triples (SVMT).

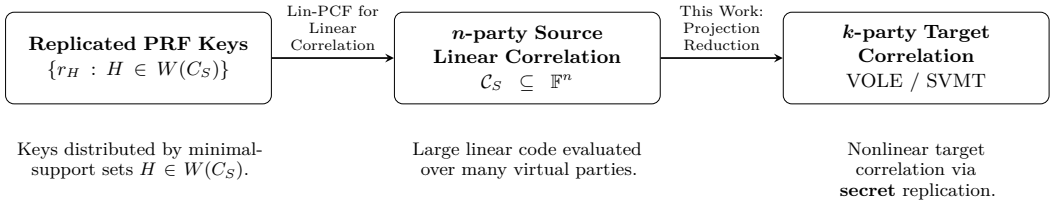


Figure 1 Overview of technique for PCF via secret replication.

1.3.1 Small-Domain VOLE from Line Correlation

Recall that in a 2-party VOLE correlation over $\mathbb{F}$, the sender gets a random vector $\mathbf{a} \in \mathbb{F}^N$ and an intercept vector $\mathbf{b} \in \mathbb{F}^N$. The receiver gets a random scalar $x \in X$ (for scalar domain $X \subseteq \mathbb{F}$) and the evaluated vector $\mathbf{y} = \mathbf{a}x + \mathbf{b} \in \mathbb{F}^N$.

To compress this inherently nonlinear correlation, we use the linear source correlation $\mathcal{C}_S$ of a Reed-Solomon code C_S of length $n = |X|$ defined by degree-1 polynomials evaluated on X. The N instances of $\mathcal{C}_S$ can be viewed as the evaluations of N random lines $L(\alpha) = \mathbf{a}\alpha + \mathbf{b}$ at all points in X. The secret assignment then works as follows:

- **The Sender** is assigned the entire set of parties $[n] = X$. By observing the evaluations at all points in X, the sender can easily interpolate the N lines to recover the slope $\mathbf{a}$ and the intercept $\mathbf{b}$.
- **The Receiver** is assigned only a single, secret coordinate $\{x\}$. By reading the evaluation of the lines at this specific point, the receiver directly obtains $\mathbf{y} = \mathbf{a}x + \mathbf{b}$.

Because the receiver's coordinate x is completely hidden from the sender, this constitutes a perfectly secure reduction. Combined with the Lin-PCF for $\mathcal{C}_S$, the number of PRF keys distributed to each party corresponds to the number of minimal-support sets $H \in W(C_S)$, which is exactly $|X|$. Indeed, for each $x \in X$, the set $H = X \setminus \{x\}$ supports the nonzero lines L such that $L(x) = 0$, and no strict subset of H can support a nonzero line.

▶ **Theorem 1** (Informal: PCF for small-domain VOLE). *Let $\mathbb{F}$ be a finite field and $X \subseteq \mathbb{F}$ be a scalar domain. There exists a 2-party PCF for VOLE with scalar $x \in X$ relying solely on the existence of PRFs. In this PCF, the sender holds $|X|$ PRF keys, the receiver holds $|X| - 1$ PRF keys. The PCF evaluation involves a single PRF evaluation for each key.*

As in [40], the PCF keys can be compressed via the use of a puncturable PRF [11, 33, 17], where the sender only holds the root of a GGM tree [28] and the receiver holds a punctured key. Moreover, to distribute the PCF key generation between the two parties, one can use concretely efficient protocols that make a black-box use of a PRG [13, 41, 40]. Similar optimizations and supporting protocols apply to other PCFs in this work, but we focus here on the core PCF results for simplicity.

Finally, similarly to [40], Theorem 1 can be extended to the case of *subfield* VOLE, in which $\mathbf{a} \in (\mathbb{F}')^N$ for a subfield $\mathbb{F}' \subset \mathbb{F}$, where typically $\mathbb{F}' = \mathbb{F}_2$. To minimize the PCF cost in this case, we utilize a generalization of the Lin-PCF construction to the case where each party can own multiple coordinates in a codeword. This generalized case was implicitly considered in [22] in the context of converting between secret-sharing schemes.

1.3.2 SVMT from Multi-Line Correlation

We now consider the more challenging case of a k-party scalar-vector multiplication triple (SVMT) correlation. Here, k parties hold additive secret shares of a random vector $\mathbf{a} \in \mathbb{F}^N$, scalar $x \in \mathbb{F}$, and their product $\mathbf{a}x$. As before, we will sometimes restrict each share of x to come from a smaller domain $X \subseteq \mathbb{F}$.

In the two-party case ($k = 2$), one can directly reduce SVMT to two instances of VOLE via a standard reduction (cf. [12]): Letting the sharing of $\mathbf{a}$ and x be $\mathbf{a} = \mathbf{a}_0 + \mathbf{a}_1$ and $x = x_0 + x_1$, we assign to each party i a random vector $\mathbf{a}_i$ and scalar x_i. To compute the additive shares of $\mathbf{a}x = \mathbf{a}_0x_0 + \mathbf{a}_0x_1 + \mathbf{a}_1x_0 + \mathbf{a}_1x_1$, each party i can locally compute $\mathbf{a}_ix_i$. To additively share the cross-terms $\mathbf{a}_0x_1$ and $\mathbf{a}_1x_0$, the parties jointly use two VOLE instances. Now each party i can locally add its shares of the 2 cross-terms given by the VOLE instances with its locally computed share, yielding an additive sharing of $\mathbf{a}x$.

Extending this to $k \geq 3$ parties seems challenging. A direct generalization would require $k \cdot (k-1)$ instances of VOLE to share the cross-terms $\mathbf{a}_ix_j$, but the values of the vector $\mathbf{a}_i$ used by a VOLE sender to multiply different receiver scalars x_j must be identical to ensure the cross-terms sum correctly to $\mathbf{a}x$. The repeated use of the same $\mathbf{a}_i$ in VOLE invocations with different parties, often referred to as "programmability" in the PCG context [15], is not supported by our PCF for VOLE.

A natural way around this would be for a sender party i to use a single source linear correlation representing one line, $L(\alpha) = \mathbf{a}_i\alpha + \mathbf{b}_i$, and have all other $k-1$ parties, acting as VOLE receivers, evaluate this exact same line at their respective scalar shares x_j. However, this approach critically fails to provide security against colluding parties. Because a degree-1 polynomial is uniquely determined by just two points, any coalition of two or more receiving parties could pool their evaluations, interpolate the line, and easily extract the shared slope $\mathbf{a}_i$, thereby entirely compromising party i's secret vector share.

An alternative approach would be to avoid the use of pairwise VOLE and attempt at a direct generalization using polynomials p of degree $< k$. For instance, an entry a of $\mathbf{a}$ can be encoded by a random p with a as its free coefficient, thereby hiding a from any strict subset of parties. The projection reduction, given the scalar x, can then assign to the parties random distinct interpolation points x_i along with interpolation coefficients λ_i, subject to the constraint that for any polynomial p with free coefficient a, we have $\sum \lambda_i p(x_i) = ax$. Then, each party can output $\lambda_i \cdot p(x_i)$ as a correct additive share of ax. While this reduction achieves perfect correctness and perfectly hides a, it does not hide x.

To get around the above difficulties, we go back to a line-based solution, but use $k-1$ "fresh" lines with the same slope to encode each entry of $\mathbf{a}$. Concretely, define a linear source correlation $\mathcal{C}_S$ consisting of k independent blocks. Block i is assigned to party i acting as the "sender" for its vector share $\mathbf{a}_i$. To enforce consistency, block i consists of the evaluations of $k-1$ distinct lines that all share the *same* slope $\mathbf{a}_i$, but have independent intercepts $\mathbf{b}_{i\to j}$ for each receiver $j \neq i$. The secret assignment now proceeds as follows:

- **As a Sender:** Party i is assigned the entirety of block i. It interpolates the $k-1$ lines to extract its vector share $\mathbf{a}_i$ and the $k-1$ intercepts $\mathbf{b}_{i\to j}$.
- **As a Receiver:** In every other block $\ell \neq i$, party i is assigned a single coordinate corresponding to its secret scalar share x_i. It evaluates the line designated for it in block ℓ at the point x_i, yielding the cross-term $\mathbf{a}_\ell x_i + \mathbf{b}_{\ell\to i}$.

Applying the Lin-PCF to this specific block structure yields a combinatorial blow-up in the minimal-support sets due to the shared slope constraint across the $k-1$ lines. Counting these sets gives the exact PRF complexity for the multiparty protocol.

▶ **Theorem 2** (Informal: PCF for SVMT). *Let $k \geq 3$ be the number of parties, and let $X \subseteq \mathbb{F}$ be the support of each scalar share x_i. There exists a k-party PCF for the SVMT correlation relying solely on PRFs. In this PCF, each party holds $k|X|^{k-1} - (k-1)|X|^{k-2} + 2k - 2$ PRF keys. The PCF evaluation involves a single PRF evaluation for each key.*

We finally note that a k-party SVMT correlation over $\mathbb{F}_2$ implies the kind of gadget used in the multi-party DPF construction from [16].[2] The gadget enables compressing k-party additive shares of either a long random vector r or a long 0-vector, such that no strict subset of parties can distinguish between the two cases, nor learn anything about r in the former case. Indeed, an SVMT correlation over $\mathbb{F}_2$ achieves exactly this by having the scalar act as a selector between the two cases.

1.4 Related Work

Pseudorandom secret sharing. A closely related primitive to PCFs for linear correlations is *pseudorandom secret sharing* (PRSS), introduced by Cramer et al. [22]. PRSS enables a set of parties to locally generate shares of pseudorandom secrets drawn from a given secret-sharing scheme. A PCF for linear correlations is a means for realizing linear PRSS. The general PRSS construction from [22] is closely related to the PCF implicit in [25], but uses the semantics of secret sharing and allows parties to hold multiple field elements as shares (as opposed to a single element in [25]). Its analysis, based on canonical span programs, implies a *share conversion* transformation that locally converts replicated (aka CNF) shares of a secret s according to an access structure $\mathcal{A}$ to shares of the *same* secret s according to any other linear scheme that realizes $\mathcal{A}$. This PRSS technique was further extended by Benhamouda et al. [7], who introduced generalized PRSS for variants of Shamir's secret-sharing scheme that provide better efficiency for threshold PRSS in case there are gaps between the security threshold and the polynomial degree. This was primarily motivated by the case of packed secret sharing, where such gaps are inherent. These techniques can potentially be useful for improving our multi-party PCFs when settling for security against bounded collusion size.

Pseudorandom correlations. The quest to compress useful nonlinear correlations, such as VOLE and oblivious transfer, has led to the recent line of work of Boyle et al. on *pseudorandom correlation generators* (PCGs) [12, 14] and *pseudorandom correlation functions* (PCFs) [15]. Most PCG and PCF constructions in the literature target concrete efficiency and are based on different flavors of the LPN assumption. See [34, 29, 37, 19] and references therein. Our PCG and PCF constructions rely on the weaker (and minimal) assumption that one-way functions exist. However, we are much more limited in the type of correlations we can support. In particular, even in the simple case of VOLE, we can only efficiently support instances in which the scalar comes from a feasible-size domain. The possibility of a PCG for general VOLE from symmetric cryptography is still open, and so is the possibility of basing PCG for oblivious transfer and other useful correlations on symmetric cryptography.

2 Preliminaries

We index n parties and entries of length-n vectors by $i = 0, 1, \ldots, n-1$ and let $[n] = \{0, 1, \ldots, n-1\}$. Similarly, for any finite field $\mathbb{F}$ of size q, we identify the elements of $\mathbb{F}$ with $x = 0, 1, \ldots q-1$.

[2] The multiparty DPF from [16] was asymptotically improved in [26]. The improvement was obtained by first constructing a variant of the gadget that only requires polynomially many keys by settling for an inverse-polynomial security error, and then amplifying security at the level of the DPF application. It is not clear how to apply a similar security amplification step in the context of PCF for SVMT.

2.1 Multiparty Correlations

An m-party correlation is a joint distribution $\mathcal{C} = (R_0, \ldots, R_{m-1})$ over $D_0 \times \cdots \times D_{m-1}$, where each D_i is a finite domain, typically a vector space over a finite field $\mathbb{F}$. We will use k to denote the number of parties in a *target correlation* that we want to securely generate and n to denote the number of parties in a *source correlation* that we use towards this end. We will use $\mathcal{C}^N$ to denote an m-party correlation obtained via N independent samples of $\mathcal{C}$.

2.1.1 Linear Correlations

A typical class of source correlations we will use in this work are n-party correlations defined by a linear code, namely a linear subspace $C \subseteq \mathbb{F}^n$. We will start with the simple notion of linear correlations from [25] in which each party receives exactly one coordinate and later generalize it to allow a general assignment of coordinates to parties.

▶ **Definition 3** (Linear correlation)**.** *Let $\mathbb{F}$ be a finite field and $C \subseteq \mathbb{F}^n$ be a linear subspace. The n-party (simple) linear correlation over $\mathbb{F}$ specified by C is the correlation $\mathcal{C}$ that provides the parties with a joint sample drawn uniformly at random from C. Namely, it samples a vector $(c_0, \ldots, c_{n-1}) \leftarrow C$ and outputs the i-th coordinate c_i to party i, for $i \in \{0, \ldots, n-1\}$.*

The above notion of a linear correlation can be generalized to allow each entry to contain multiple field elements. A useful instance of this is a correlation obtained by replicating independently chosen field elements. These correlations are useful because many (pseudo-)independent instances of such a correlations can be compressed using pseudorandomness.

▶ **Definition 4** (Replicated randomness correlation)**.** *A* replicated randomness correlation *over $\mathbb{F}$ of dimension d is an n-party correlation $\mathcal{C}$ in which each entry contains a fixed subset of the entries of a uniformly random vector $\mathbf{r} \in \mathbb{F}^d$.*

2.1.2 VOLE-Style Correlations

The main target correlations we consider in this work are "VOLE-style" correlations that support secure scalar-vector products. In such a correlation, we have a random scalar $x \in \mathbb{F}$ and long random vector $\mathbf{a} \in \mathbb{F}^N$, both of which may either be given in the clear to different parties or additively secret-shared between all parties. (We will sometimes have more than one scalar multiplying the same $\mathbf{a}$.)

▶ **Definition 5** (Standard and subfield VOLE)**.** *In a standard 2-party Vector Oblivious Linear Evaluation (VOLE) correlation over $\mathbb{F}$, the sender P_0 holds random vectors $\mathbf{a}, \mathbf{b} \in \mathbb{F}^N$, and the receiver P_1 holds a random scalar $x \in \mathbb{F}$. The correlation provides the receiver with the output vector $\mathbf{a}x + \mathbf{b} \in \mathbb{F}^N$. The definition requires that $\mathbf{a}$ and $\mathbf{b}$ remain hidden from the receiver, and x remain hidden from the sender. We also consider a variant where x is sampled from a subset $X \subseteq \mathbb{F}$, where $|X| \geq 2$. We denote this correlation by $\mathsf{VOLE}_{\mathbb{F},N,X}$, omitting X when $X = \mathbb{F}$. Finally, a* subfield VOLE *correlation $\mathsf{VOLE}_{\mathbb{F},\mathbb{F}',N,X}$ is defined similarly except that $\mathbf{a} \in (\mathbb{F}')^N$ for a subfield $\mathbb{F}' \subseteq \mathbb{F}$.*

The following correlation is analogous to a standard (Beaver-style [5]) multiplication triple, except for supporting a scalar-vector multiplication instead of a scalar-scalar multiplication. It differs from the above in that $\mathbf{a}$ and x are additively shared.

▶ **Definition 6** (Scalar-Vector Multiplication Triple (SVMT))**.** *A k-party Scalar-Vector Multiplication Triple (SVMT) correlation consists of a k-party additive secret sharing of a random scalar $x \in \mathbb{F}$ and a random vector $\mathbf{a} \in \mathbb{F}^N$, along with a k-party additive secret sharing of their scalar-vector product $\mathbf{a}x \in \mathbb{F}^N$.*

More generally, we may restrict the scalar shares x_i to be sampled uniformly from $X \subseteq \mathbb{F}^k$ where $|X| \geq 2$ (in which case the shared scalar x is not necessarily uniform and may not be perfectly hidden). We denote this by $\mathsf{SVMT}_{k,\mathbb{F},N,X}$, omitting X when $X = \mathbb{F}^k$.

The following extension of SVMT enables multiplying the same vector $\mathbf{a}$ by another scalar y to achieve an authenticated variant of scalar-vector multiplication. Indeed, keeping the ratio between x and y secret enables the type of consistency checks applied in "SPDZ-style" MPC protocols [6, 23].

▶ **Definition 7** (Authenticated SVMT). *A k-party* authenticated *scalar-vector MT (ASVMT) extends SVMT by introducing an additional random scalar $y \in \mathbb{F}$, possibly restricting its shares to $Y \subseteq \mathbb{F}^k$. The correlation provides the parties with k-party additive shares of the scalars $x, y \in \mathbb{F}$, the vector $\mathbf{a} \in \mathbb{F}^N$, and the two scaled vectors $\mathbf{a}x \in \mathbb{F}^N$ and $\mathbf{a}y \in \mathbb{F}^N$. We denote this by $\mathsf{ASVMT}_{k,\mathbb{F},N,X,Y}$.*

Supporting random access. To support fast random access to a correlation instance when N is big (e.g., when $[N]$ is the input domain of a PRF), it will be convenient to view a k-party VOLE-style correlation $\mathcal{C}$ as a k-tuple of *correlated random functions*. Concretely, we view $\mathcal{C}$ as a joint distribution over k functions $(F_0, \ldots, F_{k-1})$, where $F_i : [N] \to D_i$, such that for an instance identifier $\mathsf{id} \in [N]$, the output $F_i(\mathsf{id})$ includes entry id of each vector share, along with the scalar shares which are independent of id.

2.2 Secure Non-Interactive Reductions

A central tool we use in this work is a perfectly secure and non-interactive *reduction* from a source correlation $\mathcal{C}_S$ to a target correlation $\mathcal{C}_T$, which provides a distributed way of converting a sample from $\mathcal{C}_S$ into a sample from $\mathcal{C}_T$. Here we consider a reduction that only involves local computation and no interaction. Such a reduction is a simple special case of the general notion of secure reductions, as formalized in [21, 27], where here a reduction only involves local computation and no interaction. The power of such (statistically secure) reductions in the two-party setting was studied in a recent line of work, originating from [32, 1]. We give an explicit definition below, focusing on the perfect case.

▶ **Definition 8** (Secure Non-Interactive Reduction (SNIR) [25, 32, 1]). *Let $k \geq 2$ be an integer. Let $\mathcal{C}_S$ be a k-party source correlation defined over the domain $\mathcal{S}_0 \times \cdots \times \mathcal{S}_{k-1}$, and let $\mathcal{C}_T$ be a k-party target correlation defined over the domain $\mathcal{T}_0 \times \cdots \times \mathcal{T}_{k-1}$.*

A (perfectly) secure non-interactive reduction *(SNIR) from the target correlation $\mathcal{C}_T$ to the source correlation $\mathcal{C}_S$ is a tuple of deterministic local functions $\mathcal{F} = (f_0, \ldots, f_{k-1})$, where $f_i \colon \mathcal{S}_i \to \mathcal{T}_i$, satisfying the following condition:*

For any strict subset of corrupted parties $A \subsetneq \{0, \ldots, k-1\}$ (with complement $\bar{A} = \{0, \ldots, k-1\} \setminus A$), there exists a randomized simulator Sim such that the following real distribution $\mathsf{Real}_{\mathcal{F},A}$ and the ideal distribution $\mathsf{Ideal}_{\mathsf{Sim},A}$ are identical. The distributions are defined by the following experiments:

- ***The Real Execution ($\mathsf{Real}_{\mathcal{F},A}$):***
  1. *Sample the source correlation: $\mathbf{s} \leftarrow \mathcal{C}_S$.*
  2. *Define the adversary's view of the source correlation: $\mathbf{s}_A = (s_i)_{i \in A}$.*
  3. *For all honest parties $j \in \bar{A}$, locally compute the target shares: $t_j = f_j(s_j)$. Let $\mathbf{t}_{\bar{A}} = (t_j)_{j \in \bar{A}}$.*
  4. *Output the joint distribution: $(\mathbf{s}_A, \mathbf{t}_{\bar{A}})$.*

- ***The Ideal Execution ($\mathsf{Ideal}_{\mathsf{Sim},A}$):***
  1. *Sample the target correlation:* $\mathbf{t} \leftarrow \mathcal{C}_T$.
  2. *Define the target shares for the corrupted and honest parties:* $\mathbf{t}_A = (t_i)_{i\in A}$ *and* $\mathbf{t}_{\bar{A}} = (t_j)_{j\in\bar{A}}$.
  3. *Run the simulator on the corrupted target shares to generate the simulated source shares:* $\mathbf{s}_A \leftarrow \mathsf{Sim}(\mathbf{t}_A)$.
  4. *Output the joint distribution:* $(\mathbf{s}_A, \mathbf{t}_{\bar{A}})$.

▶ Remark 9 (Correctness). The above simulation-based security definition implies the natural correctness requirement, namely that the local functions f_i map the source correlation to the target correlation. Indeed, when $A = \emptyset$, the adversary's view $\mathbf{s}_\emptyset$ is empty. The requirement that $\mathsf{Real}_{\mathcal{F},\emptyset} \equiv \mathsf{Ideal}_{\mathsf{Sim},\emptyset}$ dictates that the locally computed shares $\mathbf{t}_{\bar{A}} = (f_0(s_0), \ldots, f_{k-1}(s_{k-1}))$ generated from a fresh $\mathcal{C}_S$ sample are distributed identically to a fresh sample $\mathbf{t}$ drawn directly from $\mathcal{C}_T$.

▶ Remark 10 (Efficient simulation for linear SNIR). The above SNIR definition is purely information-theoretic and does not require the simulator Sim to be computationally efficient. However, in all of the SNIR instances considered in this work, the simulator is computationally efficient. In particular, this always holds for a *linear* SNIR, where both $\mathcal{C}_S$ and $\mathcal{C}_T$ are linear (in the sense of Definition 3), and the local mappings f_i are linear functions. This condition can be generalized using a suitable efficient reverse-sampling requirement that depends not only on the correlations but also on the f_i (see full version). The general condition will be met for all instances of SNIR considered in this work.

▶ Remark 11 (Corruptible SNIR). Finally, we note that any perfect SNIR is also perfectly secure in the *corruptible correlation* setting [14, 38]. A corruptible variant $\mathcal{C}'$ of a k-party correlation $\mathcal{C}$ enables the adversary to fix its own entries (subject to being in the support of the distribution) and samples the honest parties' entries conditioned on this choice. Access to $\mathcal{C}'$ can be viewed as a weaker resource than access to $\mathcal{C}$ since $\mathcal{C}'$ gives the adversary partial control over the outputs. However, a perfect SNIR from $\mathcal{C}_T$ to $\mathcal{C}_S$ is also a perfect *corruptible* SNIR, namely SNIR from $\mathcal{C}'_T$ to $\mathcal{C}'_S$. See full version for details.

2.3 Pseudorandom Correlations

In this section, we review the concepts of Pseudorandom Correlation Generators (PCG) and Pseudorandom Correlation Functions (PCF), which allow for securely compressing long cryptographic correlations via short, correlated seeds. We only outline the definitions and refer the reader to [14, 15] for the formal versions.

▶ **Definition 12** (Pseudorandom Correlation Generator (PCG) [12, 14], informal)**.** *A 2-party PCG for a target correlation $\mathcal{C}$ is a pair of algorithms* $(\mathsf{Gen}, \mathsf{Expand})$*. The randomized* Gen *algorithm outputs a pair of short, correlated seeds (s_0, s_1). The deterministic* Expand *algorithm takes as input a party index $\sigma \in \{0,1\}$ and a seed s_σ, and outputs a long pseudorandom string R_σ. The security requirement guarantees that the expanded outputs (R_0, R_1) are computationally indistinguishable from $\mathcal{C}$, even to an adversary that corrupts one party and learns their seed. This is formalized by requiring that* Expand *be a* computationally secure *SNIR from the* corruptible *variant $\mathcal{C}'$ of $\mathcal{C}$ (see Remark 11) to the seed correlation generated by* Gen*. This naturally generalizes to a k-party PCG.*

The use of a corruptible variant of $\mathcal{C}$ is necessary, since without it the above PCG definition cannot be realized [14]. In the context of applications, it means that the application should be secure even when relying on $\mathcal{C}'$. Fortunately, this is the case for natural applications.

A PCF can be viewed as a PCG for an exponentially big correlation, but where both the honest parties and the adversary have random access to the correlation. Here we will consider VOLE-style correlations with vector length N, where a correlation instance is indexed by an identifier $\mathsf{id} \in [N]$ that serves as the PCF input and gives random access to entry id of each vector.

▶ **Definition 13** (Pseudorandom Correlation Function (PCF) [15], informal)**.** *A PCF generalizes a PCG by providing oracle access to correlation instances indexed by* $\mathsf{id} \in [N]$*. It consists of a seed generation algorithm that outputs correlated keys* (s_0, s_1) *and a keyed evaluation function* $F_s(\mathsf{id})$*. For a suitable joint key distribution, the evaluated outputs* $(F_{s_0}(\mathsf{id}), F_{s_1}(\mathsf{id}))$ *are computationally indistinguishable from instances of the (corruptible) target correlation even when allowing an adaptively chosen sequence of indices* id*. Crucially, this indistinguishability holds against insiders, remaining secure even when the adversary is given one of the two keys* s_σ*. This naturally generalizes to a* k*-party PCF.*

▶ Remark 14 (PCGs and PCFs via SNIR). A corollary of Remark 10 and Remark 11 is that for reverse-samplable distributions $\mathcal{C}$ and $\mathcal{C}_T$, a PCF (resp., PCG) for $\mathcal{C}$ together with a perfect SNIR from $\mathcal{C}_T$ to $\mathcal{C}$ imply a PCF (resp., PCG) for $\mathcal{C}_T$.

From here on, we will formulate all of our results for the stronger PCF primitive, based on a PRF, with the understanding that a similar PCG analogue can be based on a PRG. The PCG variant is often sufficient for applications and can yield better concrete efficiency in some cases.

2.4 PCF for Linear Correlations

We now recall the replication-based approach from [25] for compressing linear correlations, formulated as a PCF for linear correlations based on any PRF.

▶ **Definition 15** (Minimal-support codeword)**.** *Let* $\mathbb{F}$ *be a finite field and* $C \subseteq \mathbb{F}^n$ *be a linear subspace. The support of a vector* $v \in \mathbb{F}^n$*, denoted* $supp(v)$*, is the set of indices* $i \in \{0, \ldots, n-1\}$ *for which* $v_i \neq 0$*. A non-zero codeword* $w \in C$ *is a minimal-support codeword if there is no non-zero* $w' \in C$ *such that* $supp(w') \subsetneq supp(w)$*. We denote by* $W(C)$ *the set of all minimal-support codewords in* C*, normalized by selecting exactly one representative for each support set (e.g., the one whose first nonzero entry is 1).*

The main technical result from [25] is a perfectly secure reduction from a linear correlation $\mathcal{C}$ to a more redundant linear correlation $\widehat{\mathcal{C}}$ which consists of replicated, independent field elements, with replication pattern determined by $W(C)$.

▶ **Definition 16** (Replicated minimal-support correlation $\widehat{\mathcal{C}}$)**.** *Let* $\mathbb{F}$ *be a finite field and* $C \subseteq \mathbb{F}^n$ *be a linear subspace, defining a linear correlation* $\mathcal{C}$*. Let* $W(C)$ *be the set of (normalized) minimal-support codewords of* C*. The* minimal-support correlation $\widehat{\mathcal{C}}$ *for* C *is a replicated* n*-party correlation containing* $|W(C)|$ *independent field elements* r_w *which are replicated as follows. For each minimal-support codeword* $w \in W(C)$ *and party* $i \in \{0, \ldots, n-1\}$*, party* i *is given* r_w *if and only if* $i \in supp(w)$*.*

The main technical result from [25] is captured by the following theorem.

▶ **Theorem 17** (Linear correlations from replicated randomness [25])**.** *For any linear correlation* $\mathcal{C}$*, there is a perfect SNIR from* $\mathcal{C}$ *to* $\widehat{\mathcal{C}}$*.*

The above reduction is motivated by the possibility of generating many (pseudo-) independent copies of each r_w using a dedicated PRF key s_w. This can be formalized as a simple direct construction of a PCF for N independent instances of $\widehat{\mathcal{C}}$ using any PRF, by combining Theorem 17 and Remark 14. This implies:

▶ **Corollary 18** (Compressing linear correlations [25, 7]). *Let $C \subseteq \mathbb{F}^n$ be a linear subspace defining a linear correlation $\mathcal{C}$. Let $W(C)$ be the set of (normalized) minimal-support codewords of C. There exists a PCF for $\mathcal{C}^N$ that requires $|W(C)|$ independent keys s_w for a PRF with input domain $[N]$ and output domain $\mathbb{F}$. Concretely, each s_w corresponds to $w \in W(C)$ and is received by each party $i \in supp(w)$. The PCF evaluation function (per instance) involves a single PRF evaluation per party per key.*

3 Secret Projection Reductions

The main idea that we introduce to obtain new PCF constructions is to use secretly chosen random *projections*, assigning to each party of a target correlation $\mathcal{C}_T$ one or more parties in a source correlation $\mathcal{C}_S$, and providing each party in $\mathcal{C}_T$ with all the data available to the parties in $\mathcal{C}_S$ it emulates. When applying this to the previous PCFs for linear correlation, we get PCFs whose keys consist of secretly replicated PRF keys.

We start by formalizing the notion of projection reductions.

▶ **Definition 19** (Projection reduction). *Let Q be a distribution over $(2^{[n]})^k$, namely k-tuples of subsets of $[n]$. For an n-party correlation $\mathcal{C}_S$, the Q-induced correlation $Q(\mathcal{C}_S)$ is a k-party correlation defined via the following process:*

- *Sample $(Q_0, Q_1, \ldots, Q_{k-1}) \leftarrow Q$;*
- *Sample $\mathbf{s} = (s_0, s_1, \ldots, s_{n-1}) \leftarrow \mathcal{C}_S$;*
- *Output $\mathbf{t} = (t_0, t_1, \ldots, t_{k-1})$ where $t_i = (Q_i, \mathbf{s}|_{Q_i})$.*

A projection reduction *from a k-party $\mathcal{C}_T$ to an n-party $\mathcal{C}_S$ is a pair (Q, R), where R is a SNIR from $\mathcal{C}_T$ to $Q(\mathcal{C}_S)$.*

One could generalize the above notion to allow the reduction to provide additional partial information to each party beyond the set Q_i. However, the above notion suffices for our purposes.

The key observation is that a projection reduction can be used to directly convert a PCF for an n-party correlation $\mathcal{C}_S$ to a similar PCF for a k-party $\mathcal{C}_T$. Indeed, the PCF for $\mathcal{C}_T$ is obtained by simply replicating the n keys of a PCF for $\mathcal{C}_S$ according to the secret pattern induced by Q. (We assume that all distributions satisfy the reverse-sampling conditions required for efficient simulation.) This is captured by the following theorem, whose formal statement and proof are deferred to the full version.

▶ **Theorem 20** (PCF via projection, informal). *Let $\mathcal{C}_S$ be an n-party correlation and $\mathcal{C}_T$ be a k-party correlation. Let Q be a distribution over $(2^{[n]})^k$. If there is a projection reduction (Q, R) from $\mathcal{C}_T$ to $\mathcal{C}_S$ and a PCF for $\mathcal{C}_S$, then the natural composition, where the key generator samples Q and distributes the PCF keys according to Q_i, is a PCF for $\mathcal{C}_T$.*

In the case of linear correlations, combining the PCF from Corollary 18 with a projection reduction leads to PCF whose keys include a *secret replication* of independent PRF keys.

4 From Linear to VOLE-Style Correlations

In this section we use linear n-party source correlations $\mathcal{C}_S$ to securely derive useful VOLE-style k-party target correlations $\mathcal{C}_T$ via projection reductions.

4.1 Small-Domain VOLE from Line Correlation

We start by reproducing the PCF for standard VOLE of Roy [40]. Here we take as the target correlation $\mathcal{C}_T = \mathsf{VOLE}_{\mathbb{F},N}$, namely a standard 2-party VOLE over $\mathbb{F}$ of length N. To obtain a PCF for VOLE, we let the source correlation $\mathcal{C}_S$ be an n-party correlation, for $n = |\mathbb{F}|$, corresponding to a length-n degree-1 Reed-Solomon code. Concretely, we use the code C_{Line}, which consists of evaluations of lines $L(\alpha) = a\alpha + b$ on the entire field. This can be generalized to a subset X of evaluation points, resulting in a code of length $n = |X|$, and to the case where $a \in \mathbb{F}'$ for a subfield $\mathbb{F}' \subseteq \mathbb{F}$. In the latter case, which serves as the basis for the SoftSpokenOT protocol [40], we get a linear correlation over $\mathbb{F}'$.

▶ **Theorem 21** (VOLE from line correlation). *There is a projection reduction from* $\mathsf{VOLE}_{\mathbb{F},N}$ *to* $\mathcal{C}^N_{\mathsf{Line},\mathbb{F}}$*, where* $\mathcal{C}_{\mathsf{Line},\mathbb{F}}$ *is the* $\mathbb{F}$*-linear correlation defined by*

$$C_{\mathsf{Line},\mathbb{F}} = \{(L(\alpha))_{\alpha\in\mathbb{F}} \,:\, L(\alpha) = a\alpha + b,\, a, b \in \mathbb{F}\}.$$

More generally, there is a projection reduction from $\mathsf{VOLE}_{\mathbb{F},N,X}$ *to* $\mathcal{C}^N_{\mathsf{Line},\mathbb{F},\mathsf{X}}$*, for the linear correlation* $C_{\mathsf{Line},\mathbb{F},X} = \{(L(\alpha))_{\alpha\in X} \,:\, L(\alpha) = a\alpha + b,\, a, b \in \mathbb{F}\}$.

Proof. We prove the generalized statement, which directly implies the base case by setting $X = \mathbb{F}$. We must construct a projection reduction (Q, R) from the target correlation $\mathcal{C}_T = \mathsf{VOLE}_{\mathbb{F},N,X}$ to the source correlation $\mathcal{C}_S = \mathcal{C}^N_{\mathsf{Line},\mathbb{F},X}$. The target $\mathcal{C}_T$ has $k = 2$ parties, where party 0 is the sender and party 1 is the receiver.

We define the projection distribution Q over $(2^X)^2$ as follows:

- Q_0 is deterministically the entire set X.
- Q_1 is a singleton set $\{x\}$, where x is sampled uniformly at random from X.

Next, we define the non-interactive reduction $R = (f_0, f_1)$. In the source correlation, each coordinate $\alpha \in X$ contains the evaluation of N independent lines: $\mathbf{s}_\alpha = \mathbf{a}\alpha + \mathbf{b} \in \mathbb{F}^N$, where $\mathbf{a}, \mathbf{b} \in \mathbb{F}^N$.

- **Sender (f_0):** The sender receives $Q_0 = X$ and the evaluations $(\mathbf{s}_\alpha)_{\alpha\in X}$. Assuming $|X| \geq 2$, the sender efficiently interpolates these lines to recover the unique slopes $\mathbf{a} \in \mathbb{F}^N$ and intercepts $\mathbf{b} \in \mathbb{F}^N$. It outputs the target shares $t_0 = (\mathbf{a}, \mathbf{b})$.
- **Receiver (f_1):** The receiver receives $Q_1 = \{x\}$ and the single evaluation $\mathbf{s}_x \in \mathbb{F}^N$. It directly outputs the target shares $t_1 = (x, \mathbf{s}_x)$.

To prove that this (Q, R) forms a perfect SNIR, we construct a simulator Sim for any corrupted subset of parties $A \subsetneq \{0, 1\}$.

Case 1: Corrupted Sender ($A = \{0\}$). In the ideal execution, the simulator receives the sender's target shares $t_0 = (\mathbf{a}, \mathbf{b}) \leftarrow \mathcal{C}_T$. It must simulate the sender's view of the projected source correlation $(Q_0, \mathbf{s}_{Q_0})$. The simulator sets $Q_0 = X$ and computes $\mathbf{s}_\alpha = \mathbf{a}\alpha + \mathbf{b}$ for all $\alpha \in X$. It outputs $(X, (\mathbf{s}_\alpha)_{\alpha\in X})$. This perfectly matches the real execution, where $\mathbf{a}$ and $\mathbf{b}$ are sampled uniformly, and the honest receiver's output t_1 is exactly the evaluation at a uniformly random $x \in X$. Thus, $\mathsf{Real}_{\mathcal{F},\{0\}} \equiv \mathsf{Ideal}_{\mathsf{Sim},\{0\}}$.

Case 2: Corrupted Receiver ($A = \{1\}$). In the ideal execution, the simulator receives the receiver's target shares $t_1 = (x, \mathbf{y}) \leftarrow \mathcal{C}_T$, where $\mathbf{y} = \mathbf{a}x + \mathbf{b}$. It must simulate the receiver's view of the projected source correlation $(Q_1, \mathbf{s}_{Q_1})$. The simulator sets $Q_1 = \{x\}$ and $\mathbf{s}_x = \mathbf{y}$. It outputs $(\{x\}, \mathbf{y})$. In the real execution, the receiver's assigned coordinate is $\{x\}$ for a uniformly random $x \leftarrow X$, and its source share is exactly the evaluation at x, which is $\mathbf{a}x + \mathbf{b}$. Conditioned on $\mathbf{a}x + \mathbf{b} = \mathbf{y}$, the honest sender's output $(\mathbf{a}, \mathbf{b})$ in the real execution is perfectly uniform, matching the ideal execution. Thus, $\mathsf{Real}_{\mathcal{F},\{1\}} \equiv \mathsf{Ideal}_{\mathsf{Sim},\{1\}}$.

Since perfect simulation holds for all strict subsets, (Q, R) is a secure projection reduction. ◀

Combining the above reduction with Theorem 20 and the PCF for line correlation implied by Corollary 18, we get a PCF for VOLE.

▶ **Corollary 22** (PCF for VOLE, implicit in [40])**.** *Let $\mathbb{F}$ be a finite field and N be the domain size of a PRF. There exists a PCF for $\mathsf{VOLE}_{\mathbb{F},N}$, where the sender holds $|\mathbb{F}|$ PRF keys and the receiver holds all keys except the one indexed by x. More generally, there is a PCF for $\mathsf{VOLE}_{\mathbb{F},N,X}$, where the sender holds $|X|$ PRF keys and the receiver holds all keys except one. Evaluating a PCF instance requires each party to perform a single PRF evaluation per key.*

We conclude this section with a few remarks on extensions and optimizations.

▶ Remark 23 (Subfield VOLE). The above PCF can be naturally extended to support subfield-VOLE $\mathsf{VOLE}_{\mathbb{F},\mathbb{F}',N,X}$. This uses a projection reduction from $\mathsf{VOLE}_{\mathbb{F},\mathbb{F}',N,X}$ to $\mathcal{C}^N_{\mathsf{Line},\mathbb{F},\mathbb{F}',X}$, for the $\mathbb{F}'$-linear correlation $C_{\mathsf{Line},\mathbb{F},\mathbb{F}',X} = \{(L(\alpha))_{\alpha\in X} : L(\alpha) = a\alpha + b, a \in \mathbb{F}', b \in \mathbb{F}\}$. Here each party owns multiple entries of the correlation over $\mathbb{F}'$. To match the cost of the above VOLE solution, we use a generalization of the GI construction that can assign multiple codeword coordinates to each party. See full version for details.

▶ Remark 24 (Smooth VOLE). The above construction is not efficient for super-polynomial size $\mathbb{F}$, unless the set of evaluation points X is of polynomial size. However, by concatenating independent copies of $\mathsf{VOLE}_{\mathbb{F}_{p_i},N}$ for small primes p_i and using the Chinese Remainder Theorem, one can get a PCF for VOLE over an exponentially large smooth integer modulus $M = \prod p_i$. This type of PCF serves as the basis for Kondi's threshold ECDSA protocol [35].

▶ Remark 25 (Key compression and distributed setup). The PCF for VOLE obtained directly via Corollary 22 requires the VOLE sender to hold $|\mathbb{F}|$ PRF keys and the receiver to hold $|\mathbb{F}| - 1$ keys. The keys can be compressed via the use of a puncturable PRF [11, 33, 17], where the sender only holds the root of a GGM tree [28] and the receiver holds a punctured key including a logarithmic number of nodes along the path to a leaf. Moreover, to distribute the key generation, one can use concretely efficient protocols that make a black-box use of a PRG [13, 41, 40].

4.2 Scalar-Vector MT from Multi-Line Correlation

When generalizing from 2-party VOLE to a k-party Scalar-Vector Multiplication Triple (SVMT), a natural approach to generate the cross-terms $x_i\mathbf{a}_j$ would be for a sender party j to use a single source linear correlation representing one line, $L(\alpha) = \mathbf{a}_j\alpha + \mathbf{b}_j$, and have all other $k-1$ parties evaluate this exact same line at their respective scalar shares x_i. However, this approach critically fails to provide security against colluding parties. Because a degree-1 polynomial is uniquely determined by just two points, any coalition of two or more receiving parties could pool their evaluations, interpolate the line, and easily extract the shared slope $\mathbf{a}_j$, thereby entirely compromising party j's secret vector share.

To securely generate these cross-terms without leaking $\mathbf{a}_j$ to colluding subsets, we execute k symmetric instances of a random projection utilizing multiple lines with independent intercepts. For each party $j \in \{1,\dots,k\}$ acting as the "sender", we define a source linear correlation C_S of length $M = (k-1)|\mathbb{F}|$. The code C_S consists of the evaluations of $k-1$ independent lines, denoted $L_{j\to i}(\alpha) = \mathbf{a}_j\alpha + \mathbf{b}_{j\to i}$ for each $i \neq j$, over all points $\alpha \in \mathbb{F}$. Crucially, all $k-1$ lines share the identical slope $\mathbf{a}_j$ (representing party j's vector share) but feature independent, uniformly random intercepts $\mathbf{b}_{j\to i}$.

In the random projection from C_S to C_T, party j is assigned all entries of C_S, thereby holding the keys that generate the intercepts $\mathbf{b}_{j\to i}$. Each receiving party $i \neq j$ is assigned the single entry corresponding to the evaluation of their specific line $L_{j\to i}$ at the secret point $\alpha = x_i$. Consequently, party i non-interactively evaluates the PRFs to obtain $\mathbf{c}_{i,j} = \mathbf{a}_j x_i + \mathbf{b}_{j\to i} \in \mathbb{F}^N$.

The values $\mathbf{c}_{i,j}$ held by party i and $-\mathbf{b}_{j\to i}$ held by party j directly constitute a 2-party additive sharing of the cross-term $x_i\mathbf{a}_j$. Summing these local shares across all pairs yields the final MT. This approach guarantees full k-party privacy: a coalition of $k-1$ corrupted parties aiming to learn the honest party j's vector $\mathbf{a}_j$ will collectively observe exactly one evaluated point on each of the $k-1$ lines. Due to the independent random intercepts, these evaluations are uniformly distributed and perfectly hide the shared slope.

▶ **Theorem 26** (SVMT from multi-line correlations). *Let $S \subseteq \mathbb{F}$ be the union of the marginal supports of $X \subseteq \mathbb{F}^k$ (i.e., the set of all possible scalar shares x_i). There is a projection reduction from the scalar-vector multiplication triple correlation $\mathsf{SVMT}_{k,\mathbb{F},N,X}$ to the source correlation $\mathcal{C}^N_{\mathsf{SVMT-Line},\mathbb{F},\mathsf{S}}$. The source correlation $\mathcal{C}_{\mathsf{SVMT-Line},\mathbb{F},\mathsf{S}}$ is defined over k independent blocks, where the j-th block is an $\mathbb{F}$-linear correlation consisting of $k-1$ lines over S with a common slope:*

$$C^{(j)}_{\mathsf{Line},\mathbb{F},\mathsf{S}} = \{(L_{j\to i}(\alpha))_{\alpha\in S, i\neq j} : L_{j\to i}(\alpha) = a_j\alpha + b_{j\to i},\ a_j, b_{j\to i} \in \mathbb{F}\}$$

Proof. Let $\mathcal{C}_T = \mathsf{SVMT}_{k,\mathbb{F},N,X}$ be the target correlation. In this correlation, a joint sample provides each party $\ell \in \{0,\dots,k-1\}$ with a scalar share x_ℓ (where the joint vector $(x_0,\dots,x_{k-1})$ is drawn from X), a uniformly random vector share $\mathbf{a}_\ell \in \mathbb{F}^N$, and additive shares of the cross-terms $\mathbf{a}_j x_i$ for all $i \neq j$. Specifically, party ℓ acts as a sender of its vector share to all $i \neq \ell$ by holding uniformly random intercepts $\mathbf{b}_{\ell\to i} \in \mathbb{F}^N$, and acts as a receiver from all $j \neq \ell$ by holding the evaluated cross-terms $\mathbf{c}_{\ell,j} = \mathbf{a}_j x_\ell + \mathbf{b}_{j\to\ell} \in \mathbb{F}^N$. Party ℓ's final share of the SVMT product is computed locally as $\mathbf{a}_\ell x_\ell + \sum_{j\neq\ell}\mathbf{c}_{\ell,j} - \sum_{i\neq\ell}\mathbf{b}_{\ell\to i}$.

We construct a projection reduction (Q,R) to the source correlation $\mathcal{C}_S = \mathcal{C}^N_{\mathsf{SVMT-Line},\mathbb{F},\mathsf{S}}$. The domain of $\mathcal{C}_S$ consists of k independent blocks. Each block $j \in \{0,\dots,k-1\}$ comprises $k-1$ disjoint sub-domains $S_{j\to i}$ (each a copy of S) for all $i \neq j$.

We define the projection distribution Q over the subsets of this domain. For each party ℓ:

- **In its own block ℓ:** $Q^{(\ell)}_\ell$ is the entire domain of block ℓ, namely $\bigcup_{i\neq\ell} S_{\ell\to i}$.
- **In all other blocks $j \neq \ell$:** $Q^{(j)}_\ell$ consists of a single point in the sub-domain $S_{j\to\ell}$, specifically $Q^{(j)}_\ell = \{x_\ell\}$. The scalar share x_ℓ is consistent across all $j \neq \ell$.

We define the non-interactive reduction $R = (f_0,\dots,f_{k-1})$. Given the projected source evaluations, each party ℓ locally computes its target shares $t_\ell \in \mathcal{C}_T$ as follows:

- By interpolating the $k-1$ lines in its full block ℓ, party ℓ recovers the common slope $\mathbf{a}_\ell \in \mathbb{F}^N$ and the $k-1$ independent intercepts $\mathbf{b}_{\ell\to i} \in \mathbb{F}^N$.
- From its singleton evaluations in the blocks $j \neq \ell$, party ℓ directly reads the vectors $\mathbf{c}_{\ell,j} \in \mathbb{F}^N$.
- Party ℓ computes

$$\mathbf{z}_\ell = \mathbf{a}_\ell x_\ell + \sum_{j\neq\ell}\mathbf{c}_{\ell,j} - \sum_{i\neq\ell}\mathbf{b}_{\ell\to i} \in \mathbb{F}^N$$

and outputs

$$t_\ell = (x_\ell, \mathbf{a}_\ell, \mathbf{z}_\ell).$$

To argue correctness, note that:

$$\sum_{\ell=0}^{k-1}\mathbf{z}_\ell = \sum_\ell \mathbf{a}_\ell x_\ell + \sum_\ell\sum_{j\neq\ell}(\mathbf{a}_j x_\ell + \mathbf{b}_{j\to\ell}) - \sum_\ell\sum_{i\neq\ell}\mathbf{b}_{\ell\to i} = \left(\sum_j \mathbf{a}_j\right)\left(\sum_\ell x_\ell\right) = \mathbf{a}x.$$

Moreover, conditioned on the shares $(x_\ell,\mathbf{a}_\ell)_\ell$, the vector $(\mathbf{z}_\ell)_\ell$ is uniform subject to $\sum_\ell \mathbf{z}_\ell = \mathbf{a}x$, owing to the refreshing induced by the independent directed-edge masks $\mathbf{b}_{j\to i}$.

To prove that (Q, R) is a perfect SNIR, we define a simulator Sim for any strict subset of corrupted parties $A \subsetneq \{0, \ldots, k-1\}$.

Fix a strict corrupted coalition $A \subsetneq [k]$. The simulator receives

$$(x_i, \mathbf{a}_i, \mathbf{z}_i)_{i \in A}.$$

It constructs the projected source view as follows. For every corrupted sender block $j \in A$, it samples all intercepts $\mathbf{b}_{j \to i} \leftarrow \mathbb{F}^N$ for $i \neq j$ uniformly, and outputs the full line evaluations

$$L_{j \to i}(\alpha) = \mathbf{a}_j \alpha + \mathbf{b}_{j \to i} \qquad \alpha \in S.$$

For corrupted receivers $i \in A$ and corrupted sender blocks $j \in A, j \neq i$, this determines

$$\mathbf{c}_{i,j} = \mathbf{a}_j x_i + \mathbf{b}_{j \to i}.$$

It remains to simulate the values $\mathbf{c}_{i,j}$ seen by corrupted receivers $i \in A$ in honest sender blocks $j \notin A$. For each fixed $i \in A$, define

$$\mathbf{K}_i = \mathbf{a}_i x_i + \sum_{j \in A,\, j \neq i} \mathbf{c}_{i,j} - \sum_{h \neq i} \mathbf{b}_{i \to h}.$$

Since A is strict, choose one honest index $h^\star \notin A$. For every $j \notin A, j \neq h^\star$, sample $\mathbf{c}_{i,j} \leftarrow \mathbb{F}^N$ uniformly, and set

$$\mathbf{c}_{i,h^\star} = \mathbf{z}_i - \mathbf{K}_i - \sum_{j \notin A,\, j \neq h^\star} \mathbf{c}_{i,j}.$$

The simulator places $\mathbf{c}_{i,j}$ at the projected coordinate x_i in the line $j \to i$.

In the real execution, for honest sender blocks $j \notin A$, the values $\mathbf{c}_{i,j} = \mathbf{a}_j x_i + \mathbf{b}_{j \to i}$ seen by corrupted receivers are independent uniform vectors because the intercepts $\mathbf{b}_{j \to i}$ are independent uniform pads. The only constraints involving these values and the corrupted target shares are the linear equations defining $\mathbf{z}_i$ above. The simulator samples exactly the uniform affine distribution subject to these equations, hence the simulation is perfect. ◀

▶ **Corollary 27** (PCF for SVMT). *Let $\mathbb{F}$ be a finite field, N be the domain size of a PRF, and $X \subseteq \mathbb{F}^k$ be the support of the scalar shares. Let $S \subseteq \mathbb{F}$ be the union of the marginal supports of X. There exists a PCF for the k-party scalar-vector multiplication triple correlation $\mathsf{SVMT}_{k,\mathbb{F},N,X}$, where each party holds $k|S|^{k-1} - (k-1)|S|^{k-2} + 2k - 2$ PRF keys. Evaluating a PCF instance requires each party to perform a single PRF evaluation per key.*

Proof. The PCF is obtained by applying the Gilboa-Ishai construction to the source linear correlation C_S, which consists of k independent blocks. By the properties of this construction, a PCF for a linear code requires exactly $|W(C)|$ PRF keys, where $W(C)$ is the set of minimal-support codewords up to scalar multiplication. A party holds the key corresponding to $w \in W(C)$ if and only if its assigned evaluation coordinate lies in the support of w.

Each block C_{block} is an $\mathbb{F}$-linear code of length $(k-1)|S|$ corresponding to the evaluation of $k-1$ lines over S sharing a common slope. The dimension of C_{block} is k. We can classify its minimal-support codewords $W(C_{block})$ into two mutually exclusive categories:

1. **Zero slope:** Codewords supported entirely on a single line's domain. There are exactly $k-1$ such codewords (one for each line).
2. **Non-zero slope:** Codewords that evaluate to zero at exactly one point per line. Because there are $|S|$ choices for the zero on each of the $k-1$ lines, there are exactly $|S|^{k-1}$ such codewords.

Thus, the total number of minimal-support codewords for a single block is $|W(C_{block})| = |S|^{k-1} + k - 1$.

In the projection reduction, each of the k parties acts as a sender for exactly one block and as a receiver for the remaining $k-1$ blocks.

- **As a sender:** The party is assigned the entire domain of its designated block. Since its coordinates cover the entire block, it lies in the support of all minimal-support codewords. Therefore, it holds all $|S|^{k-1} + k - 1$ keys for this block.
- **As a receiver:** In each of the other $k-1$ blocks, the party is assigned a single coordinate $x_i \in S$ on its designated line. It holds the key for a zero-slope codeword only if the word is supported on its line (1 key). It holds the key for a non-zero-slope codeword only if the word is non-zero at x_i. Since exactly $|S|^{k-2}$ of the non-zero-slope words evaluate to zero at x_i, the point lies in the support of the remaining $|S|^{k-1} - |S|^{k-2}$ words. Thus, the party holds $1 + |S|^{k-1} - |S|^{k-2}$ keys per receiver block.

Summing the keys across the single sender block and the $k-1$ receiver blocks yields the total PRF key count per party:

$$(|S|^{k-1} + k - 1) + (k-1)(|S|^{k-1} - |S|^{k-2} + 1) = k|S|^{k-1} - (k-1)|S|^{k-2} + 2k - 2$$

completing the proof. ◀

4.3 Authenticated Scalar-Vector MT

Building on the k-party scalar-vector MT, we now extend the construction to an authenticated scalar-vector MT. In this setting, the parties hold a k-party additive sharing of a random vector $\mathbf{v} = \sum_{i=1}^{k} \mathbf{a}_i$ alongside two scalars $x = \sum_{i=1}^{k} x_i$ and $y = \sum_{i=1}^{k} y_i$. The goal is to non-interactively generate additive sharings of both products, $x\mathbf{v}$ and $y\mathbf{v}$. The scalar y may serve as an information-theoretic MAC key for authenticating the shared vector and subsequent operations on it. (Given our constraints on the scalar domains, this can only provide weak authentication; see Section 5 for discussion of how ASVMT may be used in applications.)

To achieve the above without compromising the secrecy of the sender's vector share $\mathbf{a}_j$, we naturally generalize the SVMT construction by expanding the source linear correlation C_S. For each party $j \in \{1, \ldots, k\}$, the correlation C_S is now defined over a length of $M = 2(k-1)|\mathbb{F}|$. It consists of the evaluations of $2(k-1)$ independent lines over $\mathbb{F}$, all of which share the exact same slope $\mathbf{a}_j$. Specifically, for each receiving party $i \neq j$, we define two lines: $L_{x,j\to i}(\alpha) = \mathbf{a}_j\alpha + \mathbf{b}_{x,j\to i}$ and $L_{y,j\to i}(\alpha) = \mathbf{a}_j\alpha + \mathbf{b}_{y,j\to i}$. The intercepts $\mathbf{b}_{x,j\to i}$ and $\mathbf{b}_{y,j\to i}$ are chosen independently and uniformly at random.

▶ **Corollary 28** (PCF for ASVMT). *Let $\mathbb{F}$ be a finite field, N be the domain size of a PRF and $X, Y \subseteq \mathbb{F}^k$ be the supports of the scalar shares. Let $S_X, S_Y \subseteq \mathbb{F}$ be the unions of the marginal supports of X, Y, respectively. There exists a PCF for the k-party authenticated scalar-vector multiplication triple correlation $\mathsf{ASVMT}_{k,\mathbb{F},N,X,Y}$, where each party holds $k(|S_X||S_Y|)^{k-1} - (k-1)(|S_X||S_Y|)^{k-2} + 4(k-1)$ PRF keys. Evaluating a PCF instance requires each party to perform a single PRF evaluation per key.*

The construction, security proof and key count closely follow the SVMT case, applied simultaneously to the two independent families of cross-terms for $\mathbf{a}x$ and $\mathbf{a}y$. The shared slope $\mathbf{a}_j$ is hidden by the independent intercepts $\mathbf{b}_{x,j\to i}, \mathbf{b}_{y,j\to i}$.

5 Applications

Our new PCF constructions have broad applications for different flavors of secure multiparty computation (MPC), particularly in settings where multiplication triples (a.k.a. Beaver triples) or Vector Oblivious Linear Evaluations (VOLE) are employed. Our results are especially relevant for functionalities that involve repeated use of the same secret value. This includes matrix-vector multiplications (where the same scalar multiplies the coordinates of an entire vector), evaluating parallel SIMD-style arithmetic circuits, circuits that compute repeatedly with a fixed value (e.g., multiple encryptions/decryptions under a fixed key), and similar scenarios.

A caveat that should be kept in mind is that the computational cost of our PCFs typically scales with $|X|^{k-1}$, where $|X|$ is the scalar domain size and k is the number of parties. (In contrast, the PCF key size can be compressed using puncturable PRFs.) This requires the number of parties to be small and the repeated scalar values to be represented by elements from a small subset of the (potentially very large) field. Still, even with this limitation, there are quite a few applications that can benefit from the new PCF constructions. We discuss some concrete use cases below.

5.1 Secret-Sharing Based MPC

A central hub for applications is a secure scalar-vector multiplication primitive, where a vector $\mathbf{v} \in \mathbb{F}^N$ and scalar $w \in \mathbb{F}$ are either owned by different parties or additively shared between k parties. The output $w\mathbf{v}$ can either be revealed in the clear or secret-shared between the parties to be further processed by the protocol. Our PCG for SVMT can be used to support all such scenarios by having the parties generate a big instance of $\mathsf{SVMT}_{\mathbb{F},N}$ from short PCG seeds. Concretely, by revealing the masked vector $\mathbf{v}' = \mathbf{v} + \mathbf{a}$ and masked scalar $w' = w + x$ (for $\mathbf{a}, x$ from SVMT), the parties can locally compute additive shares of $w\mathbf{v}$.

Such a secret-shared scalar-vector multiplication can serve as a general-purpose tool for circuit-based MPC, since it applies whenever the same value feeds into many multiplication gates. For instance, this is the case in the common scenario of circuits that involve matrix-vector multiplications. Our PCF for SVMT is particularly appealing when both k and $\mathbb{F}$ are small, e.g., in the case of 3-party evaluation of Boolean circuits. Moreover, the authenticated variant can be useful to provide lightweight (SPDZ-style) malicious security, though this use case is limited by the need to use MAC keys taken from a small domain.

5.2 Encrypted Matrix-Vector Product

A more concrete use case comes from a recent work on Encrypted Matrix-Vector Product (EMVP) [8]. The EMVP primitive has an offline phase and an online phase. In the offline phase, a client uses a secret key sk to encrypt a big matrix $\mathbf{M} \in \mathbb{F}^{N \times \ell}$ and store the encrypted matrix $\widehat{\mathbf{M}}$ on a server. In each invocation of the online phase, the client uses sk to encrypt a query vector $\mathbf{v} \in \mathbb{F}^{\ell}$ and send the encrypted vector $\widehat{\mathbf{v}}$ to the server. During the encryption of $\mathbf{v}$, the client also generates a short decryption key $\mathbf{d} \in \mathbb{F}^s$, where $s \ll \ell$. The server uses $\widehat{\mathbf{M}}$ and $\widehat{\mathbf{v}}$ to compress $\widehat{\mathbf{M}}$ into a much smaller encrypted matrix $\widehat{\mathbf{M}'} \in \mathbb{F}^{N \times s}$, which is sent back to the client. The client can recover $\mathbf{Mv}$ by computing the matrix-vector product $\widehat{\mathbf{M}'} \cdot \mathbf{d}$. Applications of EMVP include privacy-preserving linear algebra, similarity search, and machine-learning inference.

To ensure that the server's answer does not reveal anything beyond the output $\mathbf{Mv}$, or alternatively produce additive shares of $\mathbf{Mv}$ for further processing, the EMVP protocol from [8] suggests using linearly homomorphic encryption (LHE) to have the server multiply

the compressed encrypted matrix $\widehat{\mathbf{M}'}$ by an encrypted version of the decryption key $\mathbf{d}$. An alternative approach proposed in [8] is to use a PCG for VOLE. This typically involves higher communication cost than the LHE-based approach (comparable to Ns), but can have faster computational cost and easier distributed encryption and decryption.

Our PRG-based PCG for SVMT (Section 4.2) can be used as an alternative implementation of the VOLE-based approach. In particular, it can be used for efficiently distributing the EMVP client among k parties along the lines of the generic MPC application discussed above, using only symmetric cryptography. More concretely, using s PCG instances for length-N SVMT, each corresponding to a different entry in $\mathbf{d}$, the distributed client can send to the server a masked version of $\mathbf{d}$ and the server send back a masked version of $\mathbf{M}'\mathbf{d}$ that enables locally computing additive shares of $\mathbf{M}'\mathbf{d} = \mathbf{M}\mathbf{v}$ using the s SVMT correlations.

The small-domain restriction of our PCGs is often not a problem for this use case. First, for some applications a small field $\mathbb{F}$ suffices. Most importantly, the PIR application from [8] (which has the lowest server circuit complexity among all known PIR protocols) specifically uses $\mathbb{F} = \mathbb{F}_4$. Second, even for EMVP over large fields $\mathbb{F}$, security can plausibly hold even if the entries of the decryption key $\mathbf{d}$ come from a small domain $X \subset \mathbb{F}$. In this case, our PCG for $\mathsf{SVMT}_{\mathbb{F},N,X}$ will have good concrete efficiency when k is not too large.

6 Open Questions

Similarly to previous constructions from the literature, the computation and storage costs of our PCF constructions for VOLE-style correlations scale exponentially both with the bit-length of the scalar and with the number of parties. Whether this is inherent is an interesting open question.

A natural first step is to study this question within the framework proposed in this work, namely information-theoretic projection reductions to linear correlations. For VOLE, if we consider the strict notion of projection where the receiver only gets a single coordinate in the codeword, then restricting to a small domain is clearly necessary because computation scales with the number of coordinates. But for more general projection reductions that allow the receiver to get multiple codeword coordinates and apply local postprocessing, the answer seems less obvious.

References

1 Pratyush Agarwal, Varun Narayanan, Shreya Pathak, Manoj Prabhakaran, Vinod M. Prabhakaran, and Mohammad Ali Rehan. Secure non-interactive reduction and spectral analysis of correlations. In *Advances in Cryptology – EUROCRYPT 2022, Part III*, volume 13277 of *Lecture Notes in Computer Science*, pages 276–305. Springer, 2022. doi:10.1007/978-3-031-07088-4_10.

2 Yonatan Aumann and Yehuda Lindell. Security against covert adversaries: Efficient protocols for realistic adversaries. *J. Cryptol.*, 23(2):281–343, 2010. doi:10.1007/S00145-009-9040-7.

3 Carsten Baum, Ward Beullens, Cyprien de Saint Guilhem, Shibam Mukherjee, Emmanuela Orsini, Sebastian Ramacher, Christian Rechberger, Lawrence Roy, and Peter Scholl. One tree to rule them all: Optimizing GGM trees and OWFs for post-quantum signatures. *Lecture Notes in Computer Science*, pages 463–493, 2024. doi:10.1007/978-981-96-0875-1_15.

4 Carsten Baum, Lennart Braun, Cyprien Delpech de Saint Guilhem, Michael Klooß, Emmanuela Orsini, Lawrence David Roy, and Peter Scholl. Publicly verifiable zero-knowledge and post-quantum signatures from vole-in-the-head. *Lecture Notes in Computer Science*, pages 581–615, 2023. doi:10.1007/978-3-031-38554-4_19.

5 Donald Beaver. Efficient multiparty protocols using circuit randomization. In *Advances in Cryptology – CRYPTO '91*, pages 420–432. Springer, 1991. doi:10.1007/3-540-46766-1_34.

6 Rikke Bendlin, Ivan Damgård, Claudio Orlandi, and Sarah Zakarias. Semi-homomorphic encryption and multiparty computation. In Kenneth G. Paterson, editor, *Advances in Cryptology - EUROCRYPT 2011 - 30th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Tallinn, Estonia, May 15-19, 2011. Proceedings*, volume 6632 of *Lecture Notes in Computer Science*, pages 169–188. Springer, 2011. doi:10.1007/978-3-642-20465-4_11.

7 Fabrice Benhamouda, Elette Boyle, Niv Gilboa, Shai Halevi, Yuval Ishai, and Ariel Nof. Generalized pseudorandom secret sharing and efficient straggler-resilient secure computation. In *Theory of Cryptography – 19th International Conference, TCC 2021, Proceedings, Part II*, volume 13043 of *Lecture Notes in Computer Science*, pages 129–161. Springer, 2021. doi:10.1007/978-3-030-90453-1_5.

8 Fabrice Benhamouda, Caicai Chen, Shai Halevi, Yuval Ishai, Hugo Krawczyk, Tamer Mour, Tal Rabin, and Alon Rosen. Encrypted matrix-vector products from secret dual codes. In *Proceedings of the 2025 ACM SIGSAC Conference on Computer and Communications Security, CCS 2025, Taipei, Taiwan, October 13-17, 2025*, pages 394–408. ACM, 2025. doi:10.1145/3719027.3765194.

9 Kallista A. Bonawitz, Vladimir Ivanov, Ben Kreuter, Antonio Marcedone, H. Brendan McMahan, Sarvar Patel, Daniel Ramage, Aaron Segal, and Karn Seth. Practical secure aggregation for privacy-preserving machine learning. In Bhavani Thuraisingham, David Evans, Tal Malkin, and Dongyan Xu, editors, *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security, CCS 2017, Dallas, TX, USA, October 30 - November 03, 2017*, pages 1175–1191. ACM, 2017. doi:10.1145/3133956.3133982.

10 Dan Boneh, Elette Boyle, Henry Corrigan-Gibbs, Niv Gilboa, and Yuval Ishai. Zero-knowledge proofs on secret-shared data via fully linear pcps. In Alexandra Boldyreva and Daniele Micciancio, editors, *Advances in Cryptology - CRYPTO 2019 - 39th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 18-22, 2019, Proceedings, Part III*, volume 11694 of *Lecture Notes in Computer Science*, pages 67–97. Springer, 2019. doi:10.1007/978-3-030-26954-8_3.

11 Dan Boneh and Brent Waters. Constrained pseudorandom functions and their applications. In *Advances in Cryptology – ASIACRYPT 2013*, pages 280–300. Springer, 2013. doi:10.1007/978-3-642-42045-0_15.

12 Elette Boyle, Geoffroy Couteau, Niv Gilboa, and Yuval Ishai. Compressing vector OLE. In David Lie, Mohammad Mannan, Michael Backes, and XiaoFeng Wang, editors, *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security, CCS 2018, Toronto, ON, Canada, October 15-19, 2018*, pages 896–912. ACM, 2018. doi:10.1145/3243734.3243868.

13 Elette Boyle, Geoffroy Couteau, Niv Gilboa, Yuval Ishai, Lisa Kohl, Peter Rindal, and Peter Scholl. Efficient two-round OT extension and silent non-interactive secure computation. In *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security (CCS)*, pages 291–308. ACM, 2019. doi:10.1145/3319535.3354255.

14 Elette Boyle, Geoffroy Couteau, Niv Gilboa, Yuval Ishai, Lisa Kohl, and Peter Scholl. Efficient pseudorandom correlation generators: Silent OT extension and more. In *Annual International Cryptology Conference (CRYPTO)*, pages 489–518. Springer, 2019. doi:10.1007/978-3-030-26954-8_16.

15 Elette Boyle, Geoffroy Couteau, Niv Gilboa, Yuval Ishai, Lisa Kohl, and Peter Scholl. Correlated pseudorandom functions from variable-density LPN. In *61st Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 1069–1080. IEEE, 2020. doi:10.1109/FOCS46700.2020.00103.

16 Elette Boyle, Niv Gilboa, and Yuval Ishai. Function secret sharing. In *Advances in Cryptology – EUROCRYPT 2015*, pages 337–367. Springer, 2015. doi:10.1007/978-3-662-46803-6_12.

17 Elette Boyle, Shafi Goldwasser, and Ioana Ivan. Functional signatures and pseudorandom functions. In *Public-Key Cryptography – PKC 2014*, pages 501–519. Springer, 2014. doi:10.1007/978-3-642-54631-0_29.

18 Elette Boyle, Lisa Kohl, and Peter Scholl. Homomorphic secret sharing from lattices without FHE. In *EUROCRYPT 2019*, volume 11477 of *LNCS*, pages 3–33, 2019. doi:10.1007/978-3-030-17656-3_1.

19 Lennart Braun, Geoffroy Couteau, Kelsey Melissaris, Mahshid Riahinia, and Elahe Sadeghi. Fast pseudorandom correlation functions from sparse LPN. In *Advances in Cryptology – ASIACRYPT 2025*. Springer, 2025. doi:10.1007/978-981-95-5122-4_14.

20 Dung Bui, Geoffroy Couteau, Pierre Meyer, Alain Passelègue, and Mahshid Riahinia. Fast public-key silent OT and more from constrained Naor-Reingold. In Marc Joye and Gregor Leander, editors, *Advances in Cryptology - EUROCRYPT 2024 - 43rd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Zurich, Switzerland, May 26-30, 2024, Proceedings, Part VI*, volume 14656 of *Lecture Notes in Computer Science*, pages 88–118. Springer, 2024. doi:10.1007/978-3-031-58751-1_4.

21 Ran Canetti. Security and composition of multiparty cryptographic protocols. *Journal of Cryptology*, 13(1):143–202, 2000. doi:10.1007/s001459910006.

22 Ronald Cramer, Ivan Damgård, and Yuval Ishai. Share conversion, pseudorandom secret-sharing and applications to secure computation. In *Theory of Cryptography, Second Theory of Cryptography Conference, TCC 2005, Proceedings*, volume 3378 of *Lecture Notes in Computer Science*, pages 342–362. Springer, 2005. doi:10.1007/978-3-540-30576-7_19.

23 Ivan Damgård, Valerio Pastro, Nigel P. Smart, and Sarah Zakarias. Multiparty computation from somewhat homomorphic encryption. In *Advances in Cryptology – CRYPTO 2012*, pages 643–662. Springer, 2012. doi:10.1007/978-3-642-32009-5_38.

24 Yevgeniy Dodis, Shai Halevi, Ron D. Rothblum, and Daniel Wichs. Spooky encryption and its applications. In *CRYPTO 2016*, volume 9816 of *LNCS*, pages 93–120. Springer, 2016.

25 Niv Gilboa and Yuval Ishai. Compressing cryptographic resources. In *Advances in Cryptology – CRYPTO '99*, volume 1666 of *Lecture Notes in Computer Science*, pages 591–608. Springer, 1999. doi:10.1007/3-540-48405-1_38.

26 Aarushi Goel, Mingyuan Wang, and Zhiheng Wang. Multiparty distributed point functions. In *Advances in Cryptology – CRYPTO 2025*, 2025.

27 Oded Goldreich. *Foundations of Cryptography: Volume 2, Basic Applications*. Cambridge University Press, 2004.

28 Oded Goldreich, Shafi Goldwasser, and Silvio Micali. How to construct random functions. *Journal of the ACM (JACM)*, 33(4):792–807, 1986. doi:10.1145/6490.6503.

29 Sebastian Hasler and Pascal Reisert. Pseudorandom correlation functions for multiparty beaver triples from sparse LPN. Cryptology ePrint Archive, Paper 2025/2002, 2025. URL: https://eprint.iacr.org/2025/2002.

30 Johan Håstad, Russell Impagliazzo, Leonid A Levin, and Michael Luby. A pseudorandom generator from any one-way function. *SIAM Journal on Computing*, 28(4):1364–1396, 1999. doi:10.1137/S0097539793244708.

31 Nakul Khambhati, Anwesh Bhattacharya, and David Heath. Duty-free bits: Projectivizing garbling schemes. Cryptology ePrint Archive, Paper 2026/476, 2026. URL: https://eprint.iacr.org/2026/476.

32 Hamidreza Amini Khorasgani, Hemanta K. Maji, and Hai H. Nguyen. Secure non-interactive simulation from arbitrary joint distributions. In *Advances in Cryptology – EUROCRYPT 2022, Part III*, volume 13277 of *Lecture Notes in Computer Science*, pages 306–335. Springer, 2022. doi:10.1007/978-3-031-07088-4_11.

33 Aggelos Kiayias, Stavros Papadopoulos, Nikos Triandopoulos, and Thomas Zacharias. Delegatable pseudorandom functions and applications. In *Proceedings of the 2013 ACM SIGSAC Conference on Computer and Communications Security (CCS)*, pages 669–684. ACM, 2013. doi:10.1145/2508859.2516668.

34 Vladimir Kolesnikov, Stanislav Peceny, Srinivasan Raghuraman, and Peter Rindal. Stationary syndrome decoding for improved pcgs. In Yael Tauman Kalai and Seny F. Kamara, editors, *Advances in Cryptology - CRYPTO 2025 - 45th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 17-21, 2025, Proceedings, Part I*, volume 16000 of *Lecture Notes in Computer Science*, pages 284–317. Springer, 2025. doi:10.1007/978-3-032-01855-7_10.

35 Yashvanth Kondi. Two-party ECDSA signing at constant communication overhead. *IACR Cryptol. ePrint Arch.*, page 1813, 2025. URL: https://eprint.iacr.org/2025/1813.

36 Yashvanth Kondi, Claudio Orlandi, and Lawrence Roy. Two-round stateless deterministic two-party schnorr signatures from pseudorandom correlation functions. In Helena Handschuh and Anna Lysyanskaya, editors, *Advances in Cryptology - CRYPTO 2023 - 43rd Annual International Cryptology Conference, CRYPTO 2023, Santa Barbara, CA, USA, August 20-24, 2023, Proceedings, Part I*, Lecture Notes in Computer Science, pages 646–677. Springer, 2023. doi:10.1007/978-3-031-38557-5_21.

37 Zhe Li, Chaoping Xing, Yizhou Yao, and Chen Yuan. Efficient pseudorandom correlation generators over Z/p^k. In Yael Tauman Kalai and Seny F. Kamara, editors, *Advances in Cryptology - CRYPTO 2025 - 45th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 17-21, 2025, Proceedings, Part IV*, volume 16003 of *Lecture Notes in Computer Science*, pages 207–240. Springer, 2025. doi:10.1007/978-3-032-01884-7_7.

38 Daniel Masny and Peter Rindal. Endemic oblivious transfer. In Lorenzo Cavallaro, Johannes Kinder, XiaoFeng Wang, and Jonathan Katz, editors, *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security, CCS 2019, London, UK, November 11-15, 2019*, pages 309–326. ACM, 2019. doi:10.1145/3319535.3354210.

39 Claudio Orlandi, Peter Scholl, and Sophia Yakoubov. The rise of paillier: Homomorphic secret sharing and public-key silent OT. In *Advances in Cryptology - EUROCRYPT 2021 - 40th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Zagreb, Croatia, October 17-21, 2021, Proceedings, Part I*, volume 12696 of *Lecture Notes in Computer Science*, pages 678–708. Springer, 2021. doi:10.1007/978-3-030-77870-5_24.

40 Lawrence Roy. SoftSpokenOT: Communication–computation tradeoffs in OT extension. In *Annual International Cryptology Conference (CRYPTO)*, pages 255–285. Springer, 2022.

41 Phillipp Schoppmann, Adrià Gascón, Leonie Reichert, and Mariana Raykova. Distributed vector-OLE: Improved constructions and implementation. In *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security (CCS)*, pages 1055–1072. ACM, 2019. doi:10.1145/3319535.3363228.

Fast Bounded-Independence Functions and Their Duals

Martijn Brehm
Informatics Institute, University of Amsterdam, The Netherlands

Yuval Ishai
Technion, Haifa, Israel
AWS, New York, NY, USA

Nicolas Resch
Informatics Institute, University of Amsterdam, The Netherlands

Abstract

We continue the study of *fast* functions, computable by linear-size circuits, that share useful properties of random functions. Motivated by cryptographic applications, we generalize and improve on previous results in this area, obtaining the following results:

- For any constant t, we construct a fast t-wise independent hash function with algebraic degree $\log_2 t$ (over $\mathbb{F}_2$), simultaneously optimizing both asymptotic circuit size and degree.
- We simplify and improve a recent construction (ITCS 2026) of a family of fast codes with fast duals, both meeting the Gilbert-Varshamov bound. Unlike the previous construction, our construction has negligible failure probability, can accommodate general fields and rates, supports a systematic encoding, and admits fast universal encoders.
- We strengthen the above to support stronger random-like properties, such as optimal combinatorial list-decoding. This is achieved by constructing, for any constant t, a family of fast linear functions that map any t linearly independent inputs to uniform and statistically independent outputs. Prior to our work, this was only known for $t = 1$.

We demonstrate the usefulness of the above results to cryptography. This includes the first nontrivial protocols for perfectly secure multiparty computation whose circuit complexity scales linearly with the number of parties, as well as protocols for computing encrypted matrix-vector products with optimal asymptotic circuit complexity.

2012 ACM Subject Classification Theory of computation → Cryptographic primitives; Theory of computation → Error-correcting codes; Theory of computation → Cryptographic protocols

Keywords and phrases Linear codes, hash function families, efficient encoding, secure computation

Digital Object Identifier 10.4230/LIPIcs.ITC.2026.8

Related Version *Full Version*: https://doi.org/10.48550/arXiv.2606.07009 [10]

Funding *Yuval Ishai*: Research supported by ISF grant 3527/24 and BSF grant 2022370. This work is not associated with Amazon.
Nicolas Resch: Research supported by an NWO (Dutch Research Council) grant with number C.2324.0590.

1 Introduction

An important trend in cryptographic research is to develop cryptographic schemes realizing functionalities as efficiently as possible. Ideally, the cost of performing a task *with* security should be on the order of the cost of performing the same task *without* security. That is, security just results in a *constant* multiplicative computational overhead. Originating from [30], this has been the theme of a line of work on cryptography with constant computational overhead; see [7] and the references therein.

As a concrete example, consider the task of hashing a large amount of data, where one generates a short digest from a long string. One typically wants the hash function to satisfy certain desirable properties of a random function. Ideally, the cost of performing the hashing should only scale linearly with the size of the data. Which "random-like" properties can be satisfied while achieving constant computational overhead compared to just reading the input?

Similarly, one can consider the task of *encoding* a large vector in a way that satisfies desirable properties of a random linear code. In this case, one is sometimes also required to apply the encoding function of the *dual* code. Is it possible to implement *both* the primal and the dual encoding with constant overhead?

The above hashing and encoding primitives serve as useful building blocks in many algorithmic and cryptographic applications. Minimizing their overhead is relevant to all of these applications.

1.1 Our Results

To make progress on the above questions, we provide new constructions that achieve *constant computational overhead* for random-like functions. This refers by default to implementations by (fan-in 2) Boolean or arithmetic circuits whose size scales linearly with the input and output length; see Section 2.2 for discussion. We refer to such an implementation as being *fast*.

For hash functions, the work of Ishai, Kushilevitz, Ostrovsky, and Sahai [30] demonstrated the existence of fast *pairwise independent* hash functions with arbitrary input and output length.

We first demonstrate that one can build on this result to instead obtain fast *t-wise independent* hash functions. That is, even after seeing $t-1$ hash values, the t-th value is completely unpredictable (assuming the input is distinct from the previous $t-1$ inputs). This result has already been discussed informally in talks by authors of [30]. Here we spell it out in detail, also achieving near-optimal algebraic degree as a function of t, which is useful for cryptographic applications. This result also acts as a nice warmup for our later results.

▶ **Theorem 1** (Fast t-wise hash function; informal version of Theorem 21)**.** *For any constant $t \geq 2$, finite field $\mathbb{F}_q$, input length k, and output length r, there exists a construction of a fast t-wise independent hash function family $\mathcal{H}$ of functions $h : \mathbb{F}_q^k \to \mathbb{F}_q^r$. For $q = 2$, this can be done with (optimal) algebraic degree of $\log_2 t$.*

Fast dual codes. Partially motivated by cryptographic applications, a recent work of Brehm and Resch [11] constructed a family of linear codes $\mathcal{C} \leq \mathbb{F}_2^n$ of rate $1/2$ with the following properties:

- Both the code and its dual admit *fast* encoding;
- With probability $\geq 1 - 1/\mathrm{poly}(n)$, the code *and its dual* $\mathcal{C}^\perp$ achieve distance ε-close to the Gilbert-Varshamov (GV) bound.

Recall that a code $\mathcal{C}$ of rate R achieves distance ε-close to the GV bound, if its (relative) distance is at least $h_q^{-1}(1 - R - \varepsilon)$, where h_q is the q-ary entropy function (and h_q^{-1} is its inverse); see Section 2.1 for more discussion. The GV bound captures the distance achieved by *uniformly* random linear codes, which is typically the best distance one could reasonably hope for.

The [11] result was based on a rather involved analysis of a code constructed in a manner inspired by repeat-multiple-accumulate codes [19]. While it provided good concrete efficiency that can be useful for applications (see Remark 3), this result also has several drawbacks:

- The code is only constructed over the binary field $\mathbb{F}_2$;
- The code (and its dual) need to have rate exactly $1/2$;
- The randomized construction of the code has a non-negligible failure probability;
- The encoding map is not systematic (which is a problem for some applications);
- The construction does not provide a fast *universal circuit* for encoding given both the message and the randomness defining the code.

Building on the work of Druk and Ishai [21] (which, in turn, builds on [30]), we provide a simpler alternative construction that does not suffer from any of these drawbacks.

▶ **Theorem 2** (Fast code with fast dual; informal version of Theorem 9)**.** *For any $R \in (0,1)$ and any prime power q, there is a family of codes $\mathcal{C} \leq \mathbb{F}_q^n$ of rate R along with their duals $\mathcal{C}^\perp$, for which:*

- *Both $\mathcal{C}$ and $\mathcal{C}^\perp$ admit fast systematic encoding;*
- *$\forall \varepsilon > 0$, $\mathcal{C}$ and $\mathcal{C}^\perp$ are both ε-close to the GV bound with probability $\geq 1 - 2q^{-\varepsilon n}$.*

Furthermore, the encoding matrices for $\mathcal{C}$ and $\mathcal{C}^\perp$ are determined by a linear-sized universal circuit.

▶ Remark 3. A significant advantage of the construction from [11] is concrete efficiency: encoding can be done by circuits of size $12n$. In contrast, the current approach involves much bigger hidden constants that we did not attempt to optimize.

As an additional comment, for both constructions the encoding circuit has depth $O(\log n)$, which is asymptotically optimal in the bounded fan-in case. However, here too we expect our construction to have a much bigger hidden constant than the previous construction from [11].

Fast, random-like codes. By building off the concept of *local similarity* [35, 27, 36] – which informally allows one to argue that A code ensemble "behaves like" a uniformly random linear code – we can show that we can extend upon this construction, to obtain $\mathcal{C}$ and $\mathcal{C}^\perp$ possessing stronger properties. For example, they can be proved to have very good *list-decodability*, up to capacity, with optimal list sizes. Recall that a code $\mathcal{C} \leq \mathbb{F}_q^n$ is said to be (ρ, L)-list-decodable if for all $z \in \mathbb{F}_q^n$ the number of codewords within distance ρ from z is at most L. Additionally, the rate $1 - h_q(\rho)$ is called the list-decoding capacity: informally, it is the maximum rate at which a code can be (ρ, L)-list-decodable with any "reasonable" (i.e., sub-exponential) list-size L. We say a code is ε-close to list-decoding capacity if it has rate at least $1 - h_q(\rho) - \varepsilon$.

▶ **Theorem 4** (Fast list-decodable code with fast dual; informal version of Theorem 10)**.** *For any $R \in (0,1)$ and any prime power q, there is a randomized procedure producing a code $\mathcal{C} \leq \mathbb{F}_q^n$ of rate R for which:*

- *Both $\mathcal{C}$ and $\mathcal{C}^\perp$ admit fast systematic encoding;*
- *$\forall \varepsilon > 0$, both $\mathcal{C}$ and $\mathcal{C}^\perp$ are ε-close to capacity, with list-size $O(1/\varepsilon)$ with probability $\geq 1 - q^{-\Omega(\varepsilon n)}$.*

Furthermore, the encoding matrices for $\mathcal{C}$ and $\mathcal{C}^\perp$ are determined by a linear-sized universal circuit.

We comment here that list-size $O(1/\varepsilon)$ is the best one can "reasonably" hope for, in the sense that there is no known construction of a code ε-close to list-decoding capacity with list-size $o(1/\varepsilon)$. For context, prior work had managed to construct linear-time encodable

codes with such list-decodability [27], and also managed to construct a code where it and its dual achieve such list-decodability [36]: getting both properties simultaneously is, to the best of our knowledge, novel.

Cryptographic applications. While our results are broadly useful in any contexts in which error-correcting codes or hash functions are used, they are particularly motivated by several cryptographic applications.

In the context of information-theoretic cryptography, any linear code along with a dual code can support *secure multiplication* in secure multiparty computation (MPC) [34, 17]. Our results imply the first perfectly secure MPC protocols in which both the security threshold and the computational complexity (measured by Boolean circuit size) scale linearly with the number of parties. Concretely, for any constant-size function f whose inputs and outputs are owned by a constant number of parties, we get a perfectly secure n-party protocol, with a near-optimal (passive) corruption threshold of $t = (1/2 - \varepsilon) \cdot n$, in which the total circuit size of all parties is $O(n)$. Our result on fast and minimal-degree t-wise hash functions can also be useful for simultaneously minimizing communication and round complexity in distributed MPC implementations.

In the context of complexity-based cryptography, our work (as well as the related [11]) is motivated by a recent technique for computing *encrypted matrix-vector products* [3]. This technique relies on two dual codes, where one code is used for encrypting the matrix and the other is used for encrypting the vector. Our construction of fast dual codes gets around some of the limitations of the previous construction from [11] which are relevant to this use case.

See Section 5 for details on these and other cryptographic applications.

1.2 Technical Overview

In this section, we briefly discuss how we obtain our main results. First, we recall the IKOS construction of linear-sized pairwise independent hash function [30], and then the DI construction of a linear-time encodable code getting ε-close to the GV bound [21], which itself builds off the IKOS construction.

Firstly, the IKOS construction essentially goes as follows:

1. First, apply a fast code with large distance to the input. The alphabet for this code will be large, but constant.
2. Next, apply a (constant-sized) pairwise-independent hash function to each of the coordinates.
3. Lastly, apply an extractor for bit-fixing sources, which is obtained from the transpose of a generator matrix for a good code. In particular, one can take the transpose of the code applied in the first step (this also guarantees this step is fast, thanks to a "transposition" principle – cf. Lemma 12).

The intuition for the pairwise independence is as follows. Given two distinct inputs, the first encoding step produces codewords that have many distinct coordinates. On these distinct coordinates, the constant-sized hash functions map the coordinates to independent, uniform values. The last step then extracts from these many coordinates with independent values a single string of independent values.

To get the t-wise independent hash function family, we simply have to replace the constant-sized pairwise-independent hash functions by constant-sized t-wise independent hash functions. The reasoning now is very similar. Suppose now we are given t distinct inputs. If the initial code distance is good enough (greater than $1 - 1/t^2$), then in many

of the coordinates all the values of the encodings will be distinct, allowing the inner hash functions to output independent, uniform randomness which can be extracted in the last step.

We now turn to the constructions of the codes. Here, we make use of the concept of a *linear uniform output family (LUOF)*, as defined in [21]. This is a random variable A distributed over $k \times r$ matrices such that, for any fixed *nonzero* vector $x \in \mathbb{F}_q^k$, xA is distributed uniformly over $\mathbb{F}_q^r$. We recall that if A is uniformly random, then it has this property. In this work – as in [21] – we look for distributions over $k \times r$ matrices such that the map $x \mapsto xA$ is *fast* (informally, implementable by a linear-sized circuit). For brevity, we will call such an LUOF *fast*.

The basic idea of [21] is to construct a fast LUOF over $k \times n$ matrices. It is relatively straightforward that this suffices for obtaining codes ε-close to the GV bound.[1] To get the fast LUOF, [21] keep the first and third steps the same as in the IKOS construction, but instead of applying pairwise independent hash functions, they simply apply constant-sized matrices to each of the coordinates, which are viewed as length-β vectors over the base field $\mathbb{F}_q$ (recall the first encoding step may increase the alphabet size, but only by a constant amount). The reasoning for why this gives an LUOF is very similar. Given a nonzero message vector, the first encoding step gives many coordinates on which it is nonzero, and on these coordinates the small matrices map them to uniform values, which the final step can extract into a single uniform string, as required.

Now, to get codes where both it and its dual have good distance, we look at the code with systematic encoding map $x \mapsto (x, xA)$, where now A is supported on $k \times r$ matrices. This gives a code $\mathcal{C}$ of block-length $n := k + r$ and dimension k, and if A is fast, then so is the map $x \mapsto (x, xA)$. Additionally, it is easy to verify that the code defined by the encoding map $y \mapsto (-yA^\top, y)$ is dual to $\mathcal{C}$, and the transposition principle guarantees that computing $y \mapsto -yA^\top$ is also fast. Furthermore, we show that if A is LUOF, then so is $A^\top$: the crucial ingredient in this argument is Vazirani's XOR Lemma [15, 39]. Finally, a simple argument shows then that both $\mathcal{C}$ and $\mathcal{C}^\perp$ get ε-close to the GV bound with high probability, as desired.

To consider more sophisticated properties like list-decodability, we observe that it suffices to consider the following generalization of the LUOF property. We say a random matrix A supported on $k \times r$ matrices is t-LUOF if, given t *linearly independent* vectors $x_1, \ldots, x_t \in \mathbb{F}_q^k$, the vectors $x_1A, \ldots, x_tA$ are uniform and independent over $\mathbb{F}_q^r$.[2] Again, we observe that if A is a uniformly random matrix, then it has this property for any $t \leq k$ (for $t > k$, the concept is vacuous, as there are no sets of $k + 1$ or more linearly independent vectors in $\mathbb{F}_q^k$).

Our first observation is that, by assuming the first code of the DI construction has sufficiently good distance ($> 1 - 1/q^t$), we can obtain a t-LUOF via their construction. The argument is similar to above: now, we are given t linearly independent vectors $x_1, \ldots, x_t$, and we need to argue that in many coordinates we still have linearly independent vectors (recall that each coordinate of this first encoding step is a length-β vector). This indeed follows assuming the distance is sufficiently large. Then, on these good coordinates, the inner matrices map the linearly independent vectors to independent, uniform values, which are then extracted in the final step.

[1] Indeed, one can inspect the "standard" argument that random linear codes – i.e., codes defined by sampling a uniformly random generator matrix – get ε-close to the GV-bound with high probability, and observe that the only property one uses is that uniformly random matrices give an LUOF.

[2] In this second case, we mean independent in the "stochastic" sense.

Having obtained thus a t-LUOF A, we can then consider the (random) code $\mathcal{C}$ with encoding map $x \mapsto (x, xA)$, along with its dual defined by the encoding $y \mapsto (-yA^\top, y)$. That we can then prove these codes achieve sophisticated properties like list-decoding then follows from the theory of local similarity, as developed in recent works [35, 27, 36]. Morally speaking, one observes that if one wishes to prove a code is (ρ, L)-list-decodable, then one is interested in computing quantities like $\mathbb{P}[x_1, \ldots, x_{L+1} \in \mathcal{C}]$, where $x_1, \ldots, x_{L+1}$ are all within distance ρ of some vector z. Suppose one has an argument that shows uniformly random linear codes $\mathcal{C}'$ are with high probability (ρ, L)-list-decodable, where $\mathcal{C}'$ is defined as the kernel of a uniformly random $(n-k) \times n$ matrix. In this case, we know that $\mathbb{P}[x_1, \ldots, x_{L+1} \in \mathcal{C}'] = q^{-(n-k)\dim(\mathrm{span}\{x_1,\ldots,x_{L+1}\})}$. One can verify that for the random $\mathcal{C}$ defined as above, we have $\mathbb{P}[x_1, \ldots, x_{L+1} \in \mathcal{C}'] = q^{-(n-k)\dim(\mathrm{span}\{x_1,\ldots,x_{L+1}\})}$, assuming A is $(L+1)$-LUOF. That is, if we take $t = L+1$, then we get list-decodability for $\mathcal{C}$ (or, more precisely, whatever list-decodability is known to hold for $\mathcal{C}'$ whp, also holds for $\mathcal{C}$ whp). Also, generalizing the reasoning above we can show that if A is t-LUOF, then so is $A^\perp$, in which case we can derive analogous list-decodability guarantees for $\mathcal{C}^\perp$.

1.3 Related Work

While we have already mentioned the papers most directly connected to our work, we now mention a few more relevant related works.

Work by Gál, Hansen, Koucký, Pudlák and Viola [23] studies circuit sizes for encoding maps of asymptotically good codes over the binary alphabet. However, their model allows for gates with *unbounded* fan-in (whereas we allow only fan-in 2), and additionally they insist of small depth (constant, or $\log^* n$), whereas we allow for larger depth (say, $O(\log n)$).

Another related work by Li [33] considers the *unbounded* arithmetic circuit complexity (i.e., gates are arbitrary and have unbounded fan-in) of secret-sharing schemes. The main result is a demonstration that the graph underlying the circuit computing the shares for such a scheme must be a *superconcentrator*. Based on this fact, lower bounds on the size of such a sharing circuit are derived.

A final related work is due to Fan, Li and Yang [22], which builds a pairwise independent hash function with optimal size of $\approx 2n$ (where n is the input length) in the B_2 circuit model (i.e., Boolean circuits with arbitrary fan-in 2 gates).[3] Unlike the IKOS construction (and ours), their construction only yields highly compressive mappings (that is, the output length is $\approx n^{0.1}$), it is not *perfectly* pairwise independent (only statistically close), and only achieves linear size when the circuit can depend on the key (rather than having a single universal circuit of linear size that takes both the key and the input).

2 Preliminaries

We use $\mathbb{N}$ to denote the set of positive integers, and for $n \in \mathbb{N}$ we abbreviate $[n] := \{1, 2, \ldots, n\}$. Throughout, q denotes a prime power, and $\mathbb{F}_q$ a finite field with q elements. By default, we view vectors $x \in \mathbb{F}_q^n$ as row vectors. We will also use tuple notation to denote concatenation: given $x \in \mathbb{F}_q^n$ and $y \in \mathbb{F}_q^m$, $z = (x, y)$ denotes the length $n+m$ vector such that $z_i = x_i$ for $i \in [n]$, and $z_i = y_{i-n}$ for $i \in \{n+1, \ldots, n+m\}$.

[3] They obtain additional bounds for other circuit classes.

For a distribution $\mathcal{D}$ over a finite set U (which itself is a function $\mathcal{D} : U \to [0,1]$ satisfying $\sum_{u \in U} \mathcal{D}(u) = 1$), we denote by $X \sim \mathcal{D}$ a random variable distributed according to $\mathcal{D}$, i.e., for all $u \in U$, $\mathbb{P}[X = u] = \mathcal{D}(u)$. For a finite set S, we write $X \sim S$ to denote that X is a random variable distributed uniformly at random over the set S, i.e., $\mathbb{P}[X = u] = \frac{1}{|S|}$ if $u \in S$, and 0 otherwise.

For vectors $x, y \in \mathbb{F}_q^n$, their *inner-product* is defined as $\langle x, y \rangle := \sum_{i=1}^n x_i y_i \in \mathbb{F}_q$. We recall that the inner-product thus defined is commutative ($\langle x, y \rangle = \langle y, x \rangle$), and that additionally it is linear in both arguments, e.g., $\langle \alpha x + \beta y, z \rangle = \alpha \langle x, z \rangle + \beta \langle y, z \rangle$, where also $z \in \mathbb{F}_q^n$ and $\alpha, \beta \in \mathbb{F}_q$. For a subspace $V \leq \mathbb{F}_q^n$, we denote its *dual space* by $V^\perp := \{u \in \mathbb{F}_q^n : \forall v \in V,\ \langle u, v \rangle = 0\}$, which we recall is also a subspace and has dimension $\dim(V^\perp) = n - \dim(V)$, and that additionally $(V^\perp)^\perp = V$. Lastly, we recall that given a matrix $M \in \mathbb{F}_q^{a \times b}$ and vectors $x \in \mathbb{F}_q^a$ and $y \in \mathbb{F}_q^b$, we always have $\langle xM, y \rangle = \langle x, yM^\top \rangle$, where $M^\top \in \mathbb{F}_q^{b \times a}$ is the transpose of M.

2.1 Coding Theory

An *$\mathbb{F}_q$-linear code* $\mathcal{C}$ is an $\mathbb{F}_q$-linear subspace of $\mathbb{F}_q^n$ for some $n \in \mathbb{N}$. The *block-length* of $\mathcal{C}$ is the value n. If $\dim(\mathcal{C}) = k$, then $\mathcal{C}$'s *rate* is $R(\mathcal{C}) := \frac{k}{n}$. Given two vectors $x, y \in \mathbb{F}_q^n$, the *(relative) Hamming distance* between them is $d(x, y) := \frac{1}{n} |\{i \in [n] : x_i \neq y_i\}|$. Additionally, for a vector $x \in \mathbb{F}_q^n$ its *(relative) Hamming weight* is $\mathrm{wt}(x) := \frac{1}{n} |\{i \in [n] : x_i \neq 0\}| = \frac{1}{n} |\mathrm{supp}(x)|$, where we've also defined the *support* of a vector as the set of nonzero coordinates. Note that $\mathrm{wt}(x) = d(x, 0)$ and that $d(x, y) = \mathrm{wt}(x - y)$. Because of this latter equality, for any $\mathbb{F}_q$-linear code we have that the *(relative) minimum distance* $\delta(\mathcal{C}) := \min\{d(c, c') : c, c' \in \mathcal{C}, c \neq c'\}$ is also equal to $\min\{\mathrm{wt}(c) : c \in \mathcal{C} \setminus \{0\}\}$. Lastly, for a linear code $\mathcal{C}$, we refer to $\mathcal{C}^\perp$ as its *dual code*.

Every $\mathbb{F}_q$-linear code of dimension k admits a generator matrix $G \in \mathbb{F}_q^{k \times n}$, which is a rank-$k$ matrix such that $\mathcal{C} = \{xG : x \in \mathbb{F}_q^k\}$. It additionally admits a parity-check matrix $H \in \mathbb{F}_q^{(n-k) \times n}$, which is a rank-$n - k$ matrix such that $\mathcal{C} = \{x \in \mathbb{F}_q^n : Hx^\top = 0\}$. It follows that if H is a parity-check matrix for $\mathcal{C}$, then H is a generator matrix for $\mathcal{C}^\perp$. Additionally, the matrices satisfy $GH^\top = 0$. If G is of the form $G = \left[I_k \mid A\right]$ where $I_k \in \mathbb{F}_q^{k \times k}$ is the $k \times k$ identity matrix and $A \in \mathbb{F}_q^{k \times (n-k)}$, then G is said to be *systematic*. Note that if $G = \left[I_k \mid A\right]$ is a systematic generator matrix for $\mathcal{C}$, then $H = \left[-A^\top \mid I_{n-k}\right]$ is a parity-check matrix for $\mathcal{C}$, as

$$GH^\top = \left[I_k \mid A\right] \begin{bmatrix} -A \\ I_{n-k} \end{bmatrix} = -A + A = 0 \ .$$

We will also be interested in codes where the underlying alphabet is in fact a vector space over $\mathbb{F}_q$. Specifically, for $\beta \in \mathbb{N}$ we will refer to an $\mathbb{F}_q$-subspace $\mathcal{C} \leq \left(\mathbb{F}_q^\beta\right)^n$ as an *$\mathbb{F}_q$-additive code over $\mathbb{F}_q^\beta$*. The block-length is still n, while the rate of such a code is now $\frac{k}{\beta n}$, and we define its minimum distance with respect to the alphabet $\mathbb{F}_q^\beta$. That is, given a codeword $c \in \mathcal{C}$, writing it as $c = (c^{(1)}, \ldots, c^{(n)})$ with each $c^{(i)} \in \mathbb{F}_q^\beta$, we define $\mathrm{wt}_\beta(c) := \frac{1}{n} |\{i \in [n] : c^{(i)} \neq 0\}|$, and then $\delta(\mathcal{C}) := \min\{\mathrm{wt}_\beta(c) : c \in \mathcal{C} \setminus \{0\}\}$. Note that all $\mathbb{F}_q$-additive codes over $\mathbb{F}_q^\beta$ admit a generator matrix G of size $k \times \beta n$.

We often deal not with a single code $\mathcal{C}$, but rather with an infinite family of codes $\{\mathcal{C}_i\}_{i \geq 1}$ for an increasing sequence of block-lengths $n_i \in \mathbb{N}$, each with dimension $\{k_i\}_{i \geq 1}$ and distance $\{\delta_i\}_{i \geq 1}$. The rate of the family of codes can then be defined as $R := \liminf_{i \to \infty} k_i / n_i$ and the relative distance as $\delta := \liminf_{i \to \infty} \delta_i$. We call a family of codes *asymptotically good* whenever both $R > 0$ and $\delta > 0$. We assume the field $\mathbb{F}_q$ is the same within a family.

The q-ary Gilbert-Varshamov bound gives a lower bound on the asymptotic relative distance δ given the rate R for a family of codes over $\mathbb{F}_q$. In particular, it represents the minimum distance achieved by uniformly random (linear) codes (defined, e.g., by sampling a uniformly random generator matrix). Its definition makes use of the *q-ary entropy function* $h_q : [0, 1 - 1/q] \to [0, 1]$, which itself is defined as

$$h_q(x) := x \log_q(q-1) + x \log_q \frac{1}{x} + (1-x) \log_q \frac{1}{1-x}$$

for $x \in (0, 1-1/q]$ and $h_q(0) = 0$ at the left endpoint. This function increases continuously on its domain, and therefore admits a continuous inverse $h_q^{-1} : [0, 1] \to [0, 1 - 1/q]$. The *Gilbert-Varshamov (GV) bound* states that there exist codes of distance $\geq \delta$ and rate $\geq 1 - h_q(\delta)$. In other words, codes of rate R can achieve distance $\geq h_q^{-1}(1 - R)$. If a rate R code achieves distance $h_q^{-1}(1 - R - \varepsilon)$ for some $\varepsilon > 0$, we will say that it is *ε-close to the GV bound*. We recall that uniformly random codes are ε-close to the GV bound with probability $\geq 1 - q^{-\varepsilon n}$.

Lastly, we recall the concept of *list-decodability*. For $\rho \in (0, 1 - 1/q)$ and $L \in \mathbb{N}$, we say that a code $\mathcal{C} \subseteq \Sigma^n$ (for an arbitrary finite alphabet Σ) is *(ρ, L)-list-decodable* if, for all $z \in \Sigma^n$, we have

$$|\{c \in \mathcal{C} : d(c, z) \leq \rho\}| \leq L \ .$$

It is known codes of rate R can achieve decoding radius ρ so long as $\rho < h_q^{-1}(1 - R)$. More precisely, there exist codes of rate $1 - h_q(\rho) - \varepsilon$ that are $(\rho, O(1/\varepsilon))$-list-decodable; however, any code of rate $1 - h_q(\rho) + \varepsilon$ is not (ρ, L)-list-decodable for any $L \leq q^{o(n)}$. For this reason, $1 - h_q(\rho)$ is called *list-decoding capacity*. We will say that a code is *ε-close to list-decoding capacity* if it has rate $1 - h_q(\rho) - \varepsilon$. It is known that there exist codes ε-close to list-decoding capacity with list-size $O(1/\varepsilon)$; in fact, random linear codes achieve this with high probability [26, 32]. Furthermore, it is not known if codes ε-close to list-decoding capacity exist with list-size $o(1/\varepsilon)$; thus, hoping for list-size $O(1/\varepsilon)$ is a reasonable target.

2.2 Computational Model

We will be interested in encoding functions and hash functions that admit "fast" algorithms. This refers by default to having a uniform family of linear-size arithmetic circuits, but can also refer to other computational models in which Spielman's error correcting codes can be implemented in linear time. See [40] for discussion.

More concretely, our default complexity measure refers to the size of an arithmetic circuit over a finite field $\mathbb{F}_q$. Wires carry field elements into gates. Gates have two input wires and output the addition, multiplication or subtraction of the incoming field elements. We allow gates to have arbitrary fan-out. We also allow gates with a fan-in of 0, which represent either an input variable, a constant scalar, or (for randomized circuits) an independently uniformly random value from $\mathbb{F}_q$. We will typically consider *linear* circuits in which one of the inputs for each multiplication gate is a scalar

The *size* of the circuit is defined as the number of wires, the *depth* as the length of the longest path from an input to an output, and the *algebraic degree* as the highest (total) degree of an output as an $\mathbb{F}_q$-polynomial in the inputs.

We consider infinite families of circuits C_k over $\mathbb{F}_q$ where C_k has k input variables. We consider uniform families, which means there is a polynomial time algorithm that maps 1^k to a description of C_k. We call a function $f : \mathbb{F}_q^k \to \mathbb{F}_q^r$ *fast* if there exists a uniform circuit family C_k which computes f, where each C_k has size $O(k)$. We will sometimes apply this notion also to families of *randomized* functions computed by randomized circuits. Finally, for a family of *linear* functions, we assume by default that the family of linear-size circuits are *linear* in the sense defined above.

2.3 Hash Functions

First, we recall the definition of a hash function with bounded independence.

▶ **Definition 5** (t-wise independent hash function). *Let $q, k, r \in \mathbb{N}$ and let $\mathcal{H}$ denote a distribution over functions from $[q]^k \to [q]^r$. We say that $\mathcal{H}$ is a* t-wise independent hash function family *if, for all $x_1, \ldots, x_t \in [q]^k$* distinct*, we have, over $H \sim \mathcal{H}$,*

$$(H(x_1), \ldots, H(x_t)) \sim [q]^{rt} \ .$$

In other words, for all $y_1, \ldots, y_t \in [q]^r$ it holds that

$$\mathbb{P}\left[\bigwedge_{i=1}^{t} H(x_i) = y_i\right] = q^{-rt} \ .$$

We recall the following classical construction of t-wise independent hash functions, based on degree $\leq (t-1)$-polynomials [12, 41].

▶ **Proposition 6** (Construction of t-wise independent hash functions). *Let q be a prime power, let $\beta \in \mathbb{N}$ with $q^\beta \geq t$, and let $\mathcal{H}$ denote the uniform distribution over functions $\mathbb{F}_{q^\beta} \to \mathbb{F}_{q^\beta}$ of the form $h(x) = \sum_{i=0}^{t-1} h_i x^i$. That is, the domain and range are interpreted as the finite field with q^β elements, and the sampled functions are all degree $\leq (t-1)$ polynomials. Then $\mathcal{H}$ is a t-wise independent hash function family.*

Observe that sampling according to $\mathcal{H}$ as defined above requires $t\beta \log_2 q$ bits of uniform randomness. By considering the entropy of the resulting distribution $(H(x_1), \ldots, H(x_t))$, this is optimal, as the uniform distribution over $\mathbb{F}_q^{\beta \cdot t}$ has entropy $\beta t \log_2(q)$, and by the data-processing inequality entropy cannot increase.

In particular, if $\mathcal{H}$ only ever output functions represented by polynomials of degree $\leq t'-1$ with $t' < t$, then $\mathcal{H}$ would be supported on a set of size $\leq q^{\beta t'}$, and therefore $\mathcal{H}$ would have entropy at most $\log q^{\beta t'} = \beta t' \log(q)$, a contradiction to the above. Hence, guaranteeing that every hash function has degree $\leq t-1$ as a polynomial over $\mathbb{F}_{q^\beta}$ is optimal.

However, we can also think of the functions as being maps from $\mathbb{F}_q^\beta \to \mathbb{F}_q^\beta$ – that is, inputs and outputs are length-β vectors with coordinates in $\mathbb{F}_q$ – then the functions can be chosen to have smaller degree. That is, one can fix an $\mathbb{F}_q$-linear isomorphism $\mathbb{F}_q^\beta \to \mathbb{F}_{q^\beta}$ such that, under this identification, the function

$$F : \mathbb{F}_{q^\beta} \to \mathbb{F}_{q^\beta}, \quad x \mapsto \sum_{i=0}^{t-1} F_i \cdot x^i$$

is represented by a function $G = (G_1, \ldots, G_\beta) : \mathbb{F}_q^\beta \to \mathbb{F}_q^\beta$ where each G_j has degree $\leq (q-1)\log_q(t)$ [5, Section 2]. More precisely, one can bound the degree of each G_j by

$$\max\{|i|_q : F_i \neq 0\} \ ,$$

where we've defined $|i|_q$ as $\sum_{j=0}^{\beta-1} x_j$ where $i = \sum_{j=0}^{\beta-1} x_j q^j$. In other words, it is the sum of the coordinates in the base-q expansion of i.

Thus, we can have t-wise independent hash functions with degree $(q-1)\log_q(t)$: if $i \leq t-1$, one can verify that $|i|_q \leq (q-1)\log_q(t)$. When $q = 2$, this bound is just $\log_2(t)$. We record this fact below.

▶ **Lemma 7.** *Let q be a prime power and let $\beta \in \mathbb{N}$ with $q^\beta \geq t$. There exists a t-wise independent hash function family $\mathcal{H}$ of functions from $\mathbb{F}_q^\beta \to \mathbb{F}_q^\beta$ such that every function $h \in \mathrm{supp}(\mathcal{H})$ has algebraic degree $\leq (q-1)\log_q(t)$.*

2.4 The q-ary XOR Lemma

Finally, we record a useful lemma, sometimes referred to as Vazirani's XOR Lemma [15]. We will use the generalization to larger fields, for which a proof can be found in Rao's exposition of Bourgain's 2-source extractor [39, Lemma 4.2]. In fact, we just need the $\varepsilon = 0$ version of that statement, which results in the following.

▶ **Lemma 8.** *Let x be a random vector distributed over $\mathbb{F}_q^n$. Then $x \sim \mathbb{F}_q^n$ if and only if for all $\xi \in \mathbb{F}_q^n \setminus \{0\}$, $\langle \xi, x \rangle \sim \mathbb{F}_q$.*

In words: in order to test if a random vector is uniform over $\mathbb{F}_q^n$, it suffices to check that its inner-product with each (fixed) non-zero vector is uniform over $\mathbb{F}_q$.

3 Constructions of Fast Codes with Fast Duals

In this section we concern ourselves with the construction of a "fast good code" with a "fast good dual," as coined by [11]. These are pairs of linear codes that are dual to each other, that are both fast, and that are both asymptotically good with a rate-distance tradeoff at the GV-bound. Applications of such a pair of objects include protocols for information-theoretic secure multiparty computation (MPC) and encrypted matrix vector products with asymptotically optimal circuit size, as we discuss further in Section 5.

Recalling from the introduction, such a pair of codes was first given by [11], but their construction had a number of downsides, like being restricted to binary alphabets and both codes needing to have rate 1/2. We present a new such construction in Section 3.1 which amends the five drawbacks described in the introduction. The construction relies on a fast *linear uniform-output family* (LUOF), a distribution over matrices A such that xA is uniform for any nonzero x (see Definition 11). The properties of this new construction are summarized in the following theorem. When we refer to a map as being *systematic* we mean that it maps an input vector x to an output vector of the form (x, y) or (y, x), i.e. the output vector contains the input vector.

▶ **Theorem 9.** *Let q be a prime power and fix $R, \varepsilon \in (0, 1)$ with $\varepsilon < \min\{R, 1 - R\}$. There exist uniform families of randomized linear arithmetic circuits $\{C_i : \mathbb{F}_q^{R_i n_i} \to \mathbb{F}_q^{n_i}\}_{i \in \mathbb{N}}$ and $\{C_i^\perp : \mathbb{F}_q^{(1-R_i)n_i} \to \mathbb{F}_q^{n_i}\}_{i \in \mathbb{N}}$, where $(n_i)_{i \in \mathbb{N}}$ is an increasing sequence of positive integers and $(R_i)_{i \in \mathbb{N}}$ is a sequence of real numbers with each $R_i n_i \in \mathbb{N}$, satisfying the following properties.*

- ***Rate:*** *$\lim_{i \to \infty} R_i = R$ (and hence $\lim_{i \to \infty}(1 - R_i) = 1 - R$).*
- ***Efficient encodability:*** *For each $i \in \mathbb{N}$, $C_i : \mathbb{F}_q^{R_i n_i} \to \mathbb{F}_q^{n_i}$ is of size $O(n_i)$ with $O(n_i)$ uniformly random field elements. The map is systematic.*
- ***Dual efficient encodability:*** *For each $i \in \mathbb{N}$, $C_i^\perp : \mathbb{F}_q^{(1-R_i)n_i} \to \mathbb{F}_q^{n_i}$ of size $O(n_i)$ with $O(n_i)$ uniformly random field elements. The map is systematic.*
- ***Good distance:*** *For each $i \in \mathbb{N}$, let $\mathcal{C}_i$ denote the random code with encoding map determined by C_i, where the randomness is over the internal randomness of the circuit. Then $\mathcal{C}_i$ has rate R_i and distance $h_q^{-1}(1 - R_i - \varepsilon)$ except with probability $q^{-\varepsilon n_i}$ (GV-bound).*
- ***Good dual distance:*** *For each $i \in \mathbb{N}$, let $\mathcal{C}_i^\perp$ denote the random code with encoding map determined by $C_i^\perp$, where the randomness is over the internal randomness of the circuit. Then $\mathcal{C}_i^\perp$ has rate $1 - R_i$ and distance $h_q^{-1}(R_i - \varepsilon)$ except with probability $q^{-\varepsilon n_i}$ (GV-bound).*
- ***Duality:*** *For each $i \in \mathbb{N}$, the codes $\mathcal{C}_i$ and $\mathcal{C}_i^\perp$ are dual to one another, assuming the encoder circuits use the same randomness.*

The distance analysis of these codes is much simpler than that of the construction of [11]. This gives a better prospect of generalizing to other properties; to prove that our code construction doesn't just generate a "fast good code" with "fast good dual," but a code with additional interesting properties. We indeed manage to do so, and in Section 3.2 prove that our construction yields a pair of codes that additionally are *t-locally similar* to a random linear code (albeit, at some cost in terms of parameters). Intuitively, this means that any *t-local property* of random linear codes also applies to this pair of codes, which are features that can be ruled out by a small, size t subset of vectors. This includes distance ($t = 1$) and list-of-L-decodability ($t = L + 1$). Note that this generalizes the construction from the first subsection, which we leave mostly as a warm-up.

We conclude with the following result, which, to the best of our knowledge, is the first (randomized) construction of a code achieving list-decoding capacity with fast (*linear-size*) encoding. This holds for both the code we construct *and* its dual. In fact, the codes we construct have the best-known list-size of $L = O(1/\varepsilon)$.

▶ **Theorem 10.** *Let q be a prime power and fix $R, \varepsilon \in (0, 1)$. There is a $c > 0$ such that, assuming $\min\{1 - R - c\varepsilon, R - c\varepsilon\} > 0$, the following holds. There exist uniform families of randomized linear arithmetic circuits $\{C_i : \mathbb{F}_q^{R_i n_i} \to \mathbb{F}_q^{n_i}\}_{i\in\mathbb{N}}$ and $\{C_i^\perp : \mathbb{F}_q^{(1-R_i)n_i} \to \mathbb{F}_q^{n_i}\}_{i\in\mathbb{N}}$, where $(n_i)_{i\in\mathbb{N}}$ is an increasing sequence of positive integers and $(R_i)_{i\in\mathbb{N}}$ is a sequence of real numbers with each $R_i n_i \in \mathbb{N}$, satisfying the following properties.*

- ***Rate:*** *$\lim_{i\to\infty} R_i = R$ (and hence $\lim_{i\to\infty}(1 - R_i) = 1 - R$).*
- ***Efficient encodability:*** *For each $i \in \mathbb{N}$, $C_i : \mathbb{F}_q^{R_i n_i} \to \mathbb{F}_q^{n_i}$ is of size $O(n_i)$ with $O(n_i)$ uniformly random field elements. The map is systematic.*
- ***Dual efficient encodability:*** *For each $i \in \mathbb{N}$, $C_i^\perp : \mathbb{F}_q^{(1-R_i)n_i} \to \mathbb{F}_q^{n_i}$ of size $O(n_i)$ with $O(n_i)$ uniformly random field elements. The map is systematic.*
- ***Good list-decodability:*** *For each $i \in \mathbb{N}$, let $\mathcal{C}_i$ denote the random code with encoding map determined by C_i, where the randomness is over the internal randomness of the circuit. Then $\mathcal{C}_i$ has rate R_i and is $(h_q^{-1}(1 - R_i - c\varepsilon), \lceil 1/\varepsilon \rceil)$-list-decodable except with probability $q^{-\Omega(n_i)}$ (list-decoding capacity).*
- ***Good dual list-decodability:*** *For each $i \in \mathbb{N}$, let $\mathcal{C}_i^\perp$ denote the random code with encoding map determined by $C_i^\perp$, where the randomness is over the internal randomness of the circuit. Then $\mathcal{C}_i^\perp$ has rate $1 - R_i$ and is $(h_q^{-1}(R_i - c\varepsilon), \lceil 1/\varepsilon \rceil)$-list-decodable except with probability $q^{-\Omega(n_i)}$ (list-decoding capacity).*
- ***Duality:*** *For each $i \in \mathbb{N}$, the codes $\mathcal{C}_i$ and $\mathcal{C}_i^\perp$ are dual to one another, assuming the encoder circuits use the same randomness.*

3.1 Fast Good Code with Fast Good Dual

Our construction for a "fast good code" with "fast good dual" is based on an object known as a linear uniform output family. This is any matrix A which maps any fixed non-zero input vector to a uniformly random vector in the output space.

▶ **Definition 11** (Linear Uniform Output Family, LUOF). *Let A be a random matrix distributed over $\mathbb{F}_q^{k\times r}$. We call A a* linear uniform output family (LUOF) *if, for all $x \in \mathbb{F}_q^k \setminus \{0\}$, we have $xA \sim \mathbb{F}_q^r$.*

Classical constructions of LUOF, such as the family of Toeplitz matrices, require circuits of quasilinear size. Here we use the fast (i.e., linear-size universal circuit) LUOF construction of Druk and Ishai [21], which works over any finite field $\mathbb{F}_q$ and for any choice of k and r. This LUOF can be used to construct a fast good code: simply use the LUOF $A \in \mathbb{F}_q^{k\times n}$ itself as a generator matrix for a linear code. Since the LUOF is fast, so is the code. The code

also is asymptotically good, with rate-distance tradeoff at the GV-bound, which follows in a standard way from the fact that each message is mapped to a uniformly random vector in the output space. We briefly recall this argument as we will use a similar argument for our code constructions.

The probability the code attains some distance δ can then (by a union bound) be upper bounded by the probability that any message attains weight at most δ, times the number of messages. Since each message ends up as a uniformly random output vector, the probability it has weight at most δ is equal to the number of weight at most δ output vectors divided by the size of the output space: $q^{n \cdot h_q(d/n)-n}$, where h_q is the q-ary entropy function. Multiplying by the number of messages q^k gives an upper bound on the probability the code attains distance δ of $q^{n(h_q(\delta)-1+k/n)}$, which is $q^{-\Omega(n)}$ (negligible in n) whenever $k/n < 1 - h_q(\delta)$. This tradeoff between the rate $R := k/n$ and distance δ is exactly the q-ary GV-bound.

The fast LUOF of Druk and Ishai thus generates a fast good code. But the authors note that the dual of any code generated by a LUOF likewise is good: it also attains the GV-bound. However, it is not clear what the encoding function of such a dual code looks like, and in particular, whether this dual code is fast. We now give a slightly more complicated code construction which does explicate what the dual code looks like, and indeed lets us guarantee that the dual code is fast.

Instead of having the primal code be generated by a $k \times n$ LUOF, we suppose that we have a $k \times r$ LUOF called A. We then consider the linear maps $x \mapsto (x, xA)$ and $x \mapsto (-xA^\top, x)$. These are implemented by the generator matrices $G = [I_k|A]$ and $H = [-A^\top|I_r]$. Here, I_k and I_r are the $k \times k$ and $r \times r$ identity matrices. Note that G is a $k \times (r+k)$ matrix and H is a $r \times (r+k)$ matrix. Let us write $n := r + k$ for the block length, so that the primal code generated by G has rate k/n and the dual code generated by H has rate $(n-k)/n$.

We now claim that G is a fast good code with a fast good dual H satisfying the properties listed in Theorem 9. We spend the rest of this subsection verifying each of the properties listed in this theorem.

Proof of Theorem 9. To start, we can choose G to have any rate $R \in (0, 1)$, the codes G and H clearly have systematic encoding maps, and G and H generate dual codes as H is the parity-check matrix of G:

$$GH^\top = [I_k \mid A] \begin{bmatrix} -A \\ I_r \end{bmatrix} = -I_k A + A I_r = -A + A = 0 \ .$$

The two harder properties that remain are the fact that G and H admit fast encoding maps (arithmetic circuits of linear size in the input) and attain the GV-bound. Starting with the fast encoding maps, recall that that $xG = (x, xA)$ and that $xH = (-xA^\top, x)$. Copying x and applying a minus sign can of course both be done fast. If we sample the LUOF A according to the construction of [21], then computing xA is fast too. From this fact it actually follows that computing $xA^\top$ is fast too:

▶ **Lemma 12** (Transposition principle [4]). *There exists a constant $c_4 > 0$ such that for every finite field $\mathbb{F}$ the following holds. Let C be an arithmetic circuit over $\mathbb{F}$ of size t which consists of only addition and scalar multiplication gates and computes the function $f(x) = xA$ (where xA is a matrix-vector product). Then there exists an arithmetic circuit C' of size at most $c_4 t$ that computes the function $f'(x) = xA^\top$.*

All that remains is to show that G and H attain the GV-bound. We start with the code G, which we recall encoded a vector x to (x, xA). What we need to show is that the probability that G fails to have distance $h_q^{-1}(1-R-\varepsilon)$ for some $\varepsilon > 0$ is at most $q^{-\varepsilon n}$, where we recall that n is the block length of G. Since our code is linear, this event is equivalent to G having a codeword of weight at most $h_q^{-1}(1-R-\varepsilon)$.

There are at most $q^{nh_q(\delta)}$ length n vectors of weight at most δ. Recall the earlier high-level proof that a random linear code attains the GV-bound (in fact, any generator matrix that is a LUOF). There, we could argue that any message vector is mapped to a uniformly random output vector, so that any specific vector $y \in \mathbb{F}_q^n$ is in the code with probability q^{k-n}: there are q^k messages, and each has a probability of q^{-n} of being mapped to a some specific output vector. Applying a union bound over all bad codewords (with weight below the target d) yields an upper bound on the probability that such a code fails to have distance δ:

$$q^{nh_q(\delta)+k-n} = q^{n(h_q(\delta)-1+k/n)} = q^{n(h_q(\delta)-1+R)} = q^{n(1-R-\varepsilon-1+R)} = q^{-\varepsilon n} ,$$

where in the last step we substituted in our target distance of $\delta = h_q^{-1}(1-R-\varepsilon)$. Now, we cannot directly apply this reasoning to our code, as our code does not map each message to a uniformly random output vector. The first k entries of our output will be a copy of the message, and only the last r entries will be uniformly random. However, it is not hard to see that we can derive the same upper bound of $q^{k-n} = q^{-r}$ on the probability that a given vector $y \in \mathbb{F}_q^n$ is in our code $\mathcal{C}$. This then immediately yields the GV-bound by the above derivation.

Recall that $y \in \mathcal{C}$ whenever $yH^\top = 0$ where $H = [-A^\top | I_k]$ is the parity-check matrix of $\mathcal{C}$. Let us split up $y = (y^{(1)}, y^{(2)})$ with $y^{(1)} \in \mathbb{F}_q^k$ and $y^{(2)} \in \mathbb{F}_q^r$. Then we can write

$$\mathbb{P}[y \in \mathcal{C}] = \mathbb{P}\left[yH^\top = 0\right] = \mathbb{P}\left[y^{(2)} - y^{(1)}A = 0\right] = \mathbb{P}\left[y^{(2)} = y^{(1)}A\right] \leq q^{-r} ,$$

where the last step is because A is a LUOF, so that the output $y^{(1)}A$ is uniformly random over $\mathbb{F}_q^r$ if $y^{(1)} \neq 0$; if $y^{(1)} = 0$, then $y^{(2)} \neq 0$ (since $y = (y^{(1)}, y^{(2)}) \neq 0$), so then $\mathbb{P}\left[y^{(2)} = y^{(1)}A\right] = 0$. Thus, in our code we obtain the same (upper bound on the) probability that a given codeword is in our code as we had above for a random linear code. It follows that we obtain the exact same bound on the probability that our code achieves distance d, and hence we recover the GV-bound.

It remains to prove that the dual code, generated by H with the linear map $x \mapsto (-xA^\top, x)$, also attains the GV-bound. To do this, we prove Lemma 13 just below which states that if A is a LUOF, then $-A^\top$ is also a LUOF. The proof that H achieves the GV-bound is then analogous to that of G, establishing the theorem. ◀

▶ **Lemma 13.** *Let $r, k \in \mathbb{N}$. Let A be a random matrix distributed over $\mathbb{F}_q^{k\times r}$. If A is LUOF, then so is $-A^\top$.*

Proof. It is immediate that if A is LUOF, then so is $-A$. Thus, it suffices to prove that if A is LUOF, then so is $A^\top$. That is, we must show that for all $x \in \mathbb{F}_q^r \setminus \{0\}$, we have $xA^\top \sim \mathbb{F}_q^k$. By Lemma 8, it suffices to prove that for all $\xi \in \mathbb{F}_q^k \setminus \{0\}$, we have $\langle \xi, xA^\top \rangle \sim \mathbb{F}_q$. Now, $\langle \xi, xA^\top \rangle = \langle \xi A, x \rangle$. Since $\xi \neq 0$ and A is LUOF, we know that $\xi A \sim \mathbb{F}_q^r$, so now by the other direction of Lemma 8 we have $\langle \xi A, x \rangle \sim \mathbb{F}_q$, as desired. ◀

3.2 Fast List-Decodable Code with Fast List-Decodable Dual

We now generalize the previous construction to yield not just a "fast good code" with "fast good dual", but codes that are in addition *t-locally similar* to a random linear code. We won't define t-local similarity formally, which would require some additional definitions. Instead, we make use of Proposition II.8 of [36] (a result which was implicit in earlier works [35, 27]) which states that a code is t-locally similar to a random linear code whenever any selection

of t linearly independent vectors are in $\mathcal{C}$ with probability at most $q^{-n(1-R)t}$, where R is the rate, q is the field size and n is the block length.[4] We will establish that (a slight generalization of) our code construction satisfies this property.

As explained in the introduction to this section, this roughly implies that our code shares with random linear codes all features that can be ruled out by size t subsets of vectors. Our goal will be to show that our code construction yields a pair of codes that have good list-decodability. Recall that a code $\mathcal{C} \subseteq \mathbb{F}_q^n$ is (ρ, L)-list-decodable if for all $z \in \mathbb{F}_q^n$, $|\{c \in \mathcal{C} : d(c, z) \leq \rho\}| \leq L$. Observe that proving a code $\mathcal{C}$ is (ρ, L)-list-decodable amounts to showing that certain sets of vectors of size L are *not* contained in $\mathcal{C}$: in this specific case, the vectors that must be shown to not lie in $\mathcal{C}$ are all $L+1$-subsets of some Hamming ball of radius ρ. This aligns exactly with the intuitive idea that a local property is one that can be ruled out by a small set of vectors.

Proposition II.9 of [36] makes this precise by showing that any linear code that is t-locally similar to a random linear code has good list-decodability. We state it in full below.

▶ **Proposition 14.** *Let q be a prime power. Let $k, n, L \in \mathbb{N}$, let $R = k/n$ and let $\varepsilon > 1/(L+1)$. For n sufficiently large compared to L, there exists a constant $c > 0$ such that the following holds. If $\rho = h_q^{-1}(1 - R - c\varepsilon)$ and $\mathcal{C} \leq \mathbb{F}_q^n$ is a random code of rate R that is $(L+1)$-locally similar to a random linear code, then with probability at least $1 - q^{-\varepsilon(n - o_{n\to\infty}(1))}$, $\mathcal{C}$ is (ρ, L)-list-decodable.*

We note that the maximum radius up to which one can hope to list-decode is $h_q^{-1}(1-R-\varepsilon)$, and existentially at radius $h_q^{-1}(1 - R - \varepsilon)$ lists of size $O(1/\varepsilon)$ are sufficient. Thus, if we can prove that our codes are t-locally similar to random linear codes, they essentially obtain the best-known list-decodability. We now turn to proving this, thereby establishing Theorem 10.

We recall once more that to prove t-local similarity it suffices to prove that any choice of $b \leq t$ linearly independent vectors are in the codes with probability at most $q^{-n(1-R)b}$. Our original code construction used a fast LUOF which guaranteed that any fixed non-zero input vector is mapped to a uniformly random output vector. This property is not sufficient for our current purposes, as it doesn't guarantee that multiple distinct inputs are mapped to uniform outputs independently.

To obtain this guarantee, we generalize the notion of a LUOF. Instead of requiring a single non-zero input to be mapped to a uniform random output, we require t linearly independent inputs to be mapped to uniformly and independent outputs.

▶ **Definition 15.** *Let $t, r, k \in \mathbb{N}$. Let A be a random matrix distributed over $\mathbb{F}_q^{k\times r}$. We call A a* t-wise linear uniform output family (t-LUOF) *if, for all $x_1, x_2, \ldots, x_t \in \mathbb{F}_q^k$ linearly independent, we have $(x_1A, x_2A, \ldots, x_tA) \sim \mathbb{F}_q^{tr}$. That is, the collection of random variables x_iA for $i \in [t]$ are all uniform over $\mathbb{F}_q^r$, and independent.*

▶ Remark 16. Note that if a random matrix is t-LUOF, then it is also s-LUOF for any $s \leq t$. Indeed, if $(x_1A, x_2A, \ldots, x_tA) \sim \mathbb{F}_q^{tr}$, then also $(x_1A, x_2A, \ldots, x_sA) \sim \mathbb{F}_q^{sr}$.

In Section 4.2 we will show that the DI construction of a fast LUOF can be generalized to yield a fast t-LUOF as well. For now, we show that our code construction, set up with a t-LUOF instead of a 1-LUOF, gives us t-local similarity. Or more specifically, we prove that our construction yields a fast list-decodable code with fast list-decodable dual, as described in Theorem 10.

[4] Note that this result requires $\frac{n}{\log_q n} \geq \omega_{n\to\infty}(q^{2t})$, i.e. the block length needs to be large enough compared to q and t.

Proof of Theorem 10. We use the codes G and H according to the code construction outlined in Section 3.1, but instead of using the fast 1-LUOF of [21], we use the fast t-LUOF which we construct in Section 4.2. We recall that proof of Theorem 9 in that same subsection which shows us that these codes are dual, can be of any rates (as long as they sum to 1, of course), and have fast, systematic encoding maps. But now, instead of proving good distance, we need to argue that our codes are t-locally similar to random linear codes, for which we recall it suffices to show that any choice of $b \leq t$ linearly independent vectors are in the codes with probability at most $q^{-n(1-R)b}$. Note that $R = k/n$ and $n = k + r$, so that $n(1-R) = n(1-k/n) = n-k = r$. The probability upper bound then becomes q^{-rb}.

We now show that G and H indeed have probability $\leq q^{-rb}$ that any fixed selection of b non-zero and linearly independent vectors is in our code, establishing the required list-decodability by Proposition 14. This will require not just that A is t-LUOF, but that $A^\top$ is t-LUOF as well, which we establish in Lemma 17. Then, in Lemma 18 we show that G satisfies the earlier probability bound. A completely analogous proof establishes this result for H, which in turn completed the proof of our theorem. The proofs for these lemmata are provided in the full version [10]. ◀

▶ **Lemma 17.** *Let $t, r, k \in \mathbb{N}$. Let A be a random matrix distributed over $\mathbb{F}_q^{k \times r}$ which is t-LUOF. Then $A^\top$ is also t-LUOF.*

▶ **Lemma 18.** *Let $b, t, r, k \in \mathbb{N}$ with $b \leq t$, and let A be a random matrix distributed over $\mathbb{F}_q^{k \times r}$ which is t-LUOF. Consider systematic linear code $\mathcal{C} := \{(x, xA) : x \in \mathbb{F}_q^k\}$, which has block-length $n := k + r$. Then, for all $x_1, \ldots, x_b \in \mathbb{F}_q^n$ linearly independent, we have $\mathbb{P}[\forall j \in [b], x_j \in \mathcal{C}] \leq q^{-rb}$.*

4 Fast Bounded-Independence Functions

In this section we extend previous results from [30, 21] on hash functions and linear codes by considering a higher independence parameter t.

4.1 Fast t-Wise Independent Hash Functions

In this section we prove that, with minor modifications, the IKOS construction [30] can yield t-wise independent hash functions. This generalized variant is still fast as long as t is constant. We summarized the IKOS construction for pairwise independent hash functions ($t = 2$) in Section 1.2. We now briefly recall that the construction consisted first of an encoding step (determined by code with very good distance over a larger alphabet); then, constant-sized hash functions are applied to the coordinates of the encoding; lastly, an extraction step (implemented by the transpose of a generator matrix for a good code) is applied.

To generalize this to t-wise independence, the first natural change that one must make is that the small, constant-sized hashes $h_1, \ldots, h_m$ should now be sampled so as to be t-wise independent (where m is the block-length of the first code), which we can naturally do (and, assuming $t = O(1)$, this will only affect the circuit-size by a constant factor).

Now, recalling the proof of pairwise independence for the IKOS construction, the good distance of the first code guaranteed that, in many coordinates, the encodings of the two distinct inputs will be different. That is, if the code's distance was $1 - \varepsilon$, that exactly means that in at least a $1 - \varepsilon$ fraction of coordinates, the encodings will differ.

For t-wise independence, we are given t inputs $x_1, \ldots, x_t$, and in order to use the t-wise independence of the little hashes $h_1, \ldots, h_m$, we need many coordinates $\ell \in [m]$ in which *all* the encodings differ. In this case, an averaging argument will establish this holds in at least

a $1 - \binom{t}{2} \cdot \varepsilon$ fraction of coordinates. Thus, so long as $\varepsilon > 0$ is chosen small enough (which can be done at the cost of increasing the alphabet size for the initial code), the argument goes through.

We now formally prove the construction works. We will use the following codes from [21]:

▶ **Theorem 19** ([21, Theorem 2])**.** *For every prime power q and $0 < \delta < 1$ there exists $0 < \rho < 1$, $\beta \in \mathbb{N}$ and a family of $\mathbb{F}_q$-additive codes $\mathcal{D}_k \leq (\mathbb{F}_q^\beta)^m$ with rate $R = \frac{k}{\beta m} \geq \rho$ and minimum β-distance $\geq \delta$. Recall this means that, for all $c \in \mathcal{D}_k \setminus \{0\}$, we have $\mathrm{wt}_\beta(c) \geq \delta$.*

Furthermore, the encoding map $E_{\mathcal{D}_k}$ can be computed by a uniform family of $O(k)$-size arithmetic circuits over $\mathbb{F}_q$.

▶ Remark 20. Theorem 19 above promises codes of rate at least some value ρ; however, in some cases we will in fact require codes of a smaller rate, while preserving the same distance and running time. We can easily arrange for this by, say, zeroing out some coordinates from the messages. E.g., if $\mathcal{D}_k$ has dimension k and we'd instead like a code with dimension $k' < k$, we just append $k - k'$ 0's to the message $m \in \mathbb{F}_q^{k'}$ prior to encoding via $E_{\mathcal{D}_k}$.

▶ **Theorem 21.** *Let $k, r \in \mathbb{N}$ be growing parameters, fix $t \in \mathbb{N}$, and let q be a prime power. There exists an arithmetic circuit $C_{k,r} : \mathbb{F}_q^k \to \mathbb{F}_q^r$ of size $O(k + r)$ with $O(k + r)$ uniformly random field elements such that, over the uniformly random field elements, the circuit implements a t-wise independent hash function of degree $(q - 1)\log_q(t)$.*

The proof is provided in the full verion.

▶ Remark 22. In Theorem 21, we argued that we can achieve $O(k+r)$ circuit size for constant values of t. Naturally, the constant hidden in the big-O notation depends on the choice of t. An inspection of the proof shows that the dependence is roughly $\Theta(t^7)$. Firstly, in order to guarantee $1 - \binom{t}{2} \cdot \varepsilon$ is not too small, we require $\varepsilon = O(1/t^2)$. The codes promised by Theorem 19 can be verified to require $\beta = O(1/\varepsilon^3)$ (this itself stems from using the best-known explicit constructions of lossless expander graphs, whose degree are of this order). Thus, in the end we require $\beta = \Omega(t^6)$, and since the total size is about $t \cdot \beta$, we get $\Theta(t^7)$. In particular, with $t = \mathrm{poly}(\log(k + r))$ we could still achieve size $\tilde{O}(k + r)$. This means that when $t = \log^{o(1)} k$, we get better asymptotic size than an FFT-based implementation of a hash function using degree-t polynomials.

▶ Remark 23. We remark that having smaller algebraic degree is not possible, at least in the case $q = 2$. Here, we use the following standard fact (see, e.g., [29, Section 3.1]): if $f : \mathbb{F}_2^\beta \to \mathbb{F}_2$ is a polynomial map with algebraic degree at most d and $d < \beta$, then for any $x_0, x_1, \ldots, x_{d+1} \in \mathbb{F}_2^\beta$, we have $\sum_{S \subseteq [d+1]} f\left(x_0 + \sum_{i \in S} x_i\right) = 0$. That is: the sum over any $(d + 1)$-dimensional affine space is 0.

Now, suppose one had $\mathcal{H}$ supported on maps $G = (G_1, \ldots, G_\beta) : \mathbb{F}_2^\beta \to \mathbb{F}_2^\beta$, where each G_j is of degree d with $2^{d+1} \leq t$. Then, choose $x_1, \ldots, x_{d+1} \in \mathbb{F}_2^\beta$ linearly independent (this is possible since $d < \beta$, i.e., $d + 1 \leq \beta$), and x_0 arbitrarily; say, $x_0 = 0$. The linear independence implies the values $\sum_{i \in S} x_i$ are distinct for over $S \subseteq [d + 1]$. Then we know that, no matter what G is sampled, we have

$$\begin{aligned}\sum_{S \subseteq [d+1]} G\left(\sum_{i \in S} x_i\right) &= \sum_{S \subseteq [d+1]} \left(G_1\left(\sum_{i \in S} x_i\right), \ldots, G_\beta\left(\sum_{i \in S} x_i\right)\right) \\ &= \left(\sum_{S \subseteq [d+1]} G_1\left(\sum_{i \in S} x_i\right), \ldots, \sum_{S \subseteq [d+1]} G_\beta\left(\sum_{i \in S} x_i\right)\right) = (0, \ldots, 0) = 0\ .\end{aligned}$$

Thus, we can find $2^{d+1} \leq t$ distinct input values such that, after revealing the hash value on the first $2^{d+1} - 1$ of them, the last value is determined (as the sum of the previous values). So the distribution cannot define a t-wise independent hash function family.

4.2 Fast Bounded-Independence LUOF

Recall the definitions of a linear uniform output family (LUOF) (Section 3.1) and a t-wise LUOF (Section 3.2). The t-wise LUOF can be thought of as a linear analogue of t-wise independent hash functions. Of course, when restricting to a linear function, one cannot hope to achieve a t-wise independent hash function, as any distinct yet linearly dependent inputs will be mapped to linearly dependent outputs (and hence not stochastically independent). This motivated us to define a t-LUOF as any matrix which maps t distinct *linearly independent* non-zero inputs to uniformly random and independent outputs.

Our goal in this subsection is to give a construction for a fast t-wise LUOF. Recall that we required this for the fast list-decodable codes with fast list-decodable duals of Section 3.2. Akin to how we showed that the fast t-wise independent hash function by generalizing the IKOS construction for fast pairwise independent hash function (assuming one uses a code with large enough distance), we will generalize the fast LUOF construction of Druk and Ishai [21] to give a fast t-LUOF. Here also this will rely on the codes from Theorem 19 having sufficiently large distance. We remark that the encoding circuits for the codes we construct only have linear size if $q = O(1)$ (see Remark 25).

▶ **Theorem 24.** *Let $k, r \in \mathbb{N}$ be growing parameters, fix $t \in \mathbb{N}$, and let q be a prime power. There exists an arithmetic circuit $C_{k,r} : \mathbb{F}_q^k \to \mathbb{F}_q^r$ of size $O(k + r)$ with $O(k + r)$ uniformly random field elements such that, over the $O(k + r)$ uniformly random field elements, the circuit implements a t-LUOF.*

The proof is provided in the full version [10].

▶ Remark 25. Again, as in Remark 22 we can roughly say that since we require $\varepsilon < q^{-t}$ and $\beta > 1/\varepsilon^3$, we at the very least require $\beta > q^{3t}$, showing that the constant in the big-O notation for the circuits in Theorem 24 is at least $q^{O(t)}$. In particular, this forces both q and t to be constant to obtain a meaningful bound.

5 Cryptographic Applications

In this section we discuss several cryptographic applications of fast bounded-independence functions, most of which also require fast duals.

5.1 Fast Information-Theoretic MPC

We start by applying asymptotically good fast codes with fast duals towards secure multiparty computation (MPC) in the client-server model. For constant-size functions, our protocol is perfectly secure against almost $1/2$ of the n servers, and has total circuit complexity of $O(n)$. This should be contrasted with similar protocols based on algebraic MPC-friendly codes, such as Reed-Solomon codes, whose circuit complexity is (at least) quasi-linear in n.

▶ **Theorem 26** (Near-optimal perfect MPC for constant-size functions)**.** *Let m be a constant number of clients, X and Y finite input and output domains, and $\varepsilon > 0$ an arbitrarily small constant. For every (finite) $f : X^m \to Y^m$ and $\varepsilon > 0$, there is an m-client, n-server protocol for f with the following features.*

- *The protocol has perfect (information-theoretic) security against a passive adversary corrupting (at most) one client and $t = (1/2 - \varepsilon) \cdot n$ of the n servers.*
- *The total computational complexity of all parties, measured by Boolean circuit size, is $O(n)$.*

The above result is essentially the best one could hope for in the case of constant-size functions. First, the security threshold is nearly optimal, since a similar protocol with security threshold $t > n/2$ would imply information-theoretic oblivious transfer. Second, the circuit complexity is asymptotically optimal, since if a client sends messages to $o(n)$ servers, the adversary can corrupt all of these servers and learn the client's input. (In the perfect security case, this holds even in random access models where we only charge for messages actually sent.)

Protocols in the above client-server model can be used as a building block for other cryptographic applications. For example, the above result can be used to obtain an *oblivious transfer combiner* with similar near-optimality features using the MPC-based approach from [28].

An OLE protocol. We will start with the simpler special case of (1) $m = 2$ clients, (2) the function $f = \mathsf{OLE}$ that takes one bit x from a receiver client, two bits (y, z) from a sender client and delivers $xy + z$ (over $\mathbb{F}$) to the receiver, and (3) sub-optimal but constant fractional security threshold $t = \Omega(n)$. This case already captures the core difficulty of the problem. We will later outline how to bootstrap from this case to general functions and near-optimal security threshold using standard techniques. In the following $\mathbb{F}$ can be any finite field, but the case $\mathbb{F} = \mathbb{F}_2$ is sufficient for our purposes.

▶ **Proposition 27.** *Let $f : \mathbb{F} \times \mathbb{F}^2 \to \mathbb{F}$ be the OLE functionality, which takes $x \in \mathbb{F}$ from a Receiver client and $y, z \in \mathbb{F}$ from a Sender client, and delivers $f(x, (y, z)) = xy + z$ to Receiver. Let $\mathcal{C} \leq \mathbb{F}^{n+1}$ and its dual $\mathcal{C}^\perp \leq \mathbb{F}^{n+1}$ be linear codes with coordinates indexed $0, 1, \ldots, n$. Assume the following:*

1. *$\mathcal{C}$ has minimum distance $\mathsf{dist} > t + 1$.*
2. *$\mathcal{C}^\perp$ has minimum distance $\mathsf{dist}^\perp > t + 1$.*
3. *Both $\mathcal{C}$ and $\mathcal{C}^\perp$ have encoder circuits, E and $E^\perp$, of size s.*

Then, there exists a perfectly secure protocol that computes f against any passive adversary corrupting up to t servers and at most one client. Furthermore, the total arithmetic circuit size of all parties is $O(n + s)$.

To prove the above proposition, we start with a lemma that applies the well-known connection between linear error correcting codes and secret sharing [34] while respecting the circuit complexity of the encoder. The proof is provided in Appendix A.1 of the full version [10].

▶ **Lemma 28** (Secret sharing from linear codes)**.** *Let $\mathcal{C} \leq \mathbb{F}^{n+1}$ be a linear code with coordinates indexed $0, 1, \ldots, n$ and dual distance $\mathsf{dist}^\perp \geq 2$. Let $E : \mathbb{F}^k \to \mathbb{F}^{n+1}$ be an arithmetic encoder circuit for $\mathcal{C}$ of size s. Then, the secret sharing scheme that shares a secret $\sigma \in \mathbb{F}$ into n shares by sampling a uniformly random codeword $c \in \mathcal{C}$ subject to $c_0 = \sigma$ and dropping c_0 satisfies the following:*

1. *Perfect privacy: Any subset of shares $T \subset \{1, \ldots, n\}$ of size $|T| \leq \mathsf{dist}^\perp - 2$ perfectly hides σ.*
2. *Linear complexity: The generation of the shares from σ can be computed by a randomized arithmetic circuit of size at most $s + n$.*

The protocol and its analysis are provided in the full version [10].

From OLE to general constant-size functions. To extend the above protocol from two-party OLE to a protocol with $m = O(1)$ clients for any constant-size function f, we can use the known completeness of OLE (an arithmetic variant of OT) for secure computation of Boolean or arithmetic circuits [25, 31]. At a high level, the clients start by additively sharing their inputs among each other. Then, each addition gate is processed by having the clients locally add their shares, and each multiplication gate is processed via $O(m^2)$ invocations of the OLE protocol to additively share each cross-term $a_i b_j$ between the client holding a_i and the client holding b_j. The protocol concludes by having the clients exchange their output shares. When the field $\mathbb{F}$ is of constant size, the Boolean circuit complexity of the protocol is $O(n)$ for any constant-size circuit.

Achieving near-optimal security. The security threshold t of the protocol is close to the smaller minimal distance of the primal code and its dual, namely $\min(\mathsf{dist}, \mathsf{dist}^{\perp})$. One approach for getting it arbitrarily close to $1/2$ is to let $\mathbb{F}$ be an arbitrarily large constant, in which case Theorem 9 gives a pair of codes with relative distance $1/2 - \varepsilon$. Another approach is to start with a protocol that has an arbitrarily small (but constant) fractional security threshold, say by using $\mathbb{F} = \mathbb{F}_2$, and apply a general threshold amplification technique based on virtualization [9, 37, 18].

Putting everything together. With the above extensions of Proposition 27, we can use our constructions of fast dual codes (cf. Theorem 9) to obtain the main result of this section captured by Theorem 26. This construction is semi-explicit in the sense that there is a PPT algorithm that generates circuits implementing the parties with $negl(n)$ failure probability.

5.2 Fast Encrypted Matrix-Vector Product

Switching to the computational security setting, a recent application of fast dual codes with random-like properties was recently given in the context of computing on encrypted data [3]. Concretely, an *encrypted matrix-vector product* (EMVP) protocol enables a client to encode a matrix M using a secret code $\mathcal{C}$ and then compute an unbounded number of matrix-vector products Mq_i by encrypting each query vector q_i using the dual code $\mathcal{C}^{\perp}$. Linear algebra attacks are prevented by adding a suitable type of noise. Under a variant of the Learning Subspace with Noise (LSN) assumption [20, 14], the protocol hides both M and q_i from a server who stores the encrypted matrix and processes the encrypted queries.

Using our new families of fast dual codes, we obtain the first candidate EMVP protocols in which both the matrix encoding and the query encoding have asymptotically optimal circuit complexity. The recent construction of fast dual codes from [11] is limited to $\mathbb{F}_2$ and has a non-negligible failure probability.

5.3 Fast and Conservative Substitutes for Random Linear Codes

The previous two applications inherently relied on having a fast code coupled with a fast dual. A broader use case, which does not require duality, is providing a fast substitute to a random linear code in the context of applications that rely on the hardness of "noisy linear algebra." Our new construction of fast t-LUOF (Section 4.2) can serve as a more conservative candidate than the previous 1-LUOF construction from [21].

In more detail, in most cryptographic applications of noisy linear algebra, one typically considers the minimal distance of the code or its dual as a heuristic measure of hardness.

Thus, it is perhaps not clear why the t-local similarity provided by t-LUOF (for $t \geq 2$) is interesting. To illustrate this, consider code-based constructions that rely on the LPN assumption,[5] where one assumes that the distributions

$$(A, Ae) \quad \text{and} \quad (A, b)$$

are computationally indistinguishable, where e is a random sparse vector and b is a uniformly random vector (in this case, one should think of $A \in \mathbb{F}_q^{k \times r}$ with $r > k$). This use of minimal distance as a proxy for conjectured LPN hardness is common in the recent line of works on pseudorandom correlation generators (PCGs) [6, 8]. Indeed, minimal distance is sufficient for provably defeating any attack that fits under a natural "linear attack" framework [16].

However, it is arguably a more conservative to use an A that shares more nuanced combinatorial properties with random linear codes, such as list-decodability/-recoverability. This is exactly what the t-LUOF property guarantees.

5.4 MPC-Friendly t-Wise Hash Functions

Finally, when distributing the computation of a function f between two or more parties, either via classical MPC techniques [42, 25, 2, 13] or via fully homomorphic encryption [24], the cost of the protocol depends on both the circuit size and the algebraic degree. For $t = O(1)$, our constructions of fast t-wise independent hash functions from Section 4.1 *simultaneously* optimize both asymptotic size and degree.

There are many algorithmic applications that benefit from t-wise independent hashing for constant $t > 2$. Two well-known examples are the use of 4-wise independence in sketching [1] and 5-wise independence in linear probing [38].

References

1 Noga Alon, Yossi Matias, and Mario Szegedy. The space complexity of approximating the frequency moments. *Journal of Computer and System Sciences*, 58(1):137–147, 1999. doi:10.1006/JCSS.1997.1545.

2 Michael Ben-Or, Shafi Goldwasser, and Avi Wigderson. Completeness theorems for non-cryptographic fault-tolerant distributed computation. In *Proceedings of the Twentieth Annual ACM Symposium on Theory of Computing (STOC)*, pages 1–10. ACM, 1988.

3 Fabrice Benhamouda, Caicai Chen, Shai Halevi, Yuval Ishai, Hugo Krawczyk, Tamer Mour, Tal Rabin, and Alon Rosen. Encrypted matrix-vector products from secret dual codes. In Chun-Ying Huang, Jyh-Cheng Chen, Shiuh-Pyng Shieh, David Lie, and Véronique Cortier, editors, *Proceedings of the 2025 ACM SIGSAC Conference on Computer and Communications Security, CCS 2025, Taipei, Taiwan, October 13-17, 2025*, pages 394–408. ACM, 2025. doi:10.1145/3719027.3765194.

4 J. L. Bordewijk. Inter-reciprocity applied to electrical networks. *Applied Scientific Research, Section B*, 6(1):1–74, 1956.

5 Christina Boura and Anne Canteaut. On the influence of the algebraic degree of F^{-1} on the algebraic degree of $G \circ F$. *IEEE Transactions on Information Theory*, 59(1):691–702, 2012. doi:10.1109/TIT.2012.2214203.

6 Elette Boyle, Geoffroy Couteau, Niv Gilboa, and Yuval Ishai. Compressing vector OLE. In David Lie, Mohammad Mannan, Michael Backes, and XiaoFeng Wang, editors, *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security, CCS 2018, Toronto, ON, Canada, October 15-19, 2018*, pages 896–912. ACM, 2018. doi:10.1145/3243734.3243868.

[5] What we will give below is often called *dual*-LPN, which is equivalent to the more standard variant.

7 Elette Boyle, Geoffroy Couteau, Niv Gilboa, Yuval Ishai, Lisa Kohl, Nicolas Resch, and Peter Scholl. Oblivious transfer with constant computational overhead. In Carmit Hazay and Martijn Stam, editors, *Advances in Cryptology - EUROCRYPT 2023 - 42nd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Lyon, France, April 23-27, 2023, Proceedings, Part I*, volume 14004 of *Lecture Notes in Computer Science*, pages 271–302. Springer, 2023. doi:10.1007/978-3-031-30545-0_10.

8 Elette Boyle, Geoffroy Couteau, Niv Gilboa, Yuval Ishai, Lisa Kohl, and Peter Scholl. Efficient pseudorandom correlation generators: Silent OT extension and more. In *Annual International Cryptology Conference (CRYPTO)*, pages 489–518. Springer, 2019. doi:10.1007/978-3-030-26954-8_16.

9 Gabriel Bracha. An $O(\log n)$ expected rounds randomized Byzantine generals protocol. *Journal of the ACM (JACM)*, 34(4):910–920, 1987. doi:10.1145/31846.42229.

10 Martijn Brehm, Yuval Ishai, and Nicolas Resch. Fast bounded-independence functions and their duals. *arXiv preprint*, 2026. doi:10.48550/arXiv.2606.07009.

11 Martijn Brehm and Nicolas Resch. Linear Time Encodable Binary Code Achieving GV Bound with Linear Time Encodable Dual Achieving GV Bound. In Shubhangi Saraf, editor, *17th Innovations in Theoretical Computer Science Conference (ITCS 2026)*, volume 362 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 28:1–28:19, Dagstuhl, Germany, 2026. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.ITCS.2026.28.

12 Larry Carter and Mark N. Wegman. Universal classes of hash functions. *Journal of Computer and System Sciences*, 18(2):143–154, 1979. doi:10.1016/0022-0000(79)90044-8.

13 David Chaum, Claude Crépeau, and Ivan Damgård. Multiparty unconditionally secure protocols. In *Proceedings of the Twentieth Annual ACM Symposium on Theory of Computing (STOC)*, pages 11–19. ACM, 1988.

14 Caicai Chen, Yuval Ishai, Tamer Mour, and Alon Rosen. Secret-key PIR from random linear codes. *Cryptology ePrint Archive*, 2025. To appear in STOC 2026.

15 Benny Chor, Oded Goldreich, Johan Håstad, Joel Friedman, Steven Rudich, and Roman Smolensky. The bit extraction problem of t-resilient functions (preliminary version). In *26th Annual Symposium on Foundations of Computer Science, Portland, Oregon, USA, 21-23 October 1985*, pages 396–407. IEEE Computer Society, 1985. doi:10.1109/SFCS.1985.55.

16 Geoffroy Couteau, Peter Rindal, and Srinivasan Raghuraman. Silver: Silent VOLE and oblivious transfer from hardness of decoding structured LDPC codes. In Tal Malkin and Chris Peikert, editors, *Advances in Cryptology - CRYPTO 2021 - 41st Annual International Cryptology Conference, CRYPTO 2021, Virtual Event, August 16-20, 2021, Proceedings, Part III*, volume 12827 of *Lecture Notes in Computer Science*, pages 502–534. Springer, 2021. doi:10.1007/978-3-030-84252-9_17.

17 Ronald Cramer, Ivan Damgård, and Ueli M. Maurer. General secure multi-party computation from any linear secret-sharing scheme. In Bart Preneel, editor, *Advances in Cryptology - EUROCRYPT 2000, International Conference on the Theory and Application of Cryptographic Techniques, Bruges, Belgium, May 14-18, 2000, Proceeding*, volume 1807 of *Lecture Notes in Computer Science*, pages 316–334. Springer, 2000. doi:10.1007/3-540-45539-6_22.

18 Ivan Damgård, Yuval Ishai, Mikkel Krøigaard, Jesper Buus Nielsen, and Adam Smith. Scalable multiparty computation with nearly optimal work and resilience. In *Advances in Cryptology – CRYPTO 2008*, volume 5157 of *Lecture Notes in Computer Science*, pages 241–261. Springer, 2008. doi:10.1007/978-3-540-85174-5_14.

19 Dariush Divsalar, Hui Jin, and Robert J McEliece. Coding theorems for" turbo-like" codes. In *Proceedings of the annual Allerton Conference on Communication control and Computing*, volume 36, pages 201–210. University Of Illinois, 1998.

20 Yevgeniy Dodis, Yael Tauman Kalai, and Shachar Lovett. On cryptography with auxiliary input. In Michael Mitzenmacher, editor, *Proceedings of the 41st Annual ACM Symposium on Theory of Computing, STOC 2009, Bethesda, MD, USA, May 31 - June 2, 2009*, pages 621–630. ACM, 2009. doi:10.1145/1536414.1536498.

21 Erez Druk and Yuval Ishai. Linear-time encodable codes meeting the gilbert-varshamov bound and their cryptographic applications. In *Proceedings of the 5th Conference on Innovations in Theoretical Computer Science*, ITCS '14, pages 169–182, New York, NY, USA, 2014. Association for Computing Machinery. doi:10.1145/2554797.2554815.

22 Zhiyuan Fan, Jiatu Li, and Tianqi Yang. The exact complexity of pseudorandom functions and the black-box natural proof barrier for bootstrapping results in computational complexity. In Stefano Leonardi and Anupam Gupta, editors, *STOC '22: 54th Annual ACM SIGACT Symposium on Theory of Computing, Rome, Italy, June 20 - 24, 2022*, pages 962–975. ACM, 2022. doi:10.1145/3519935.3520010.

23 Anna Gál, Kristoffer Arnsfelt Hansen, Michal Koucký, Pavel Pudlák, and Emanuele Viola. Tight bounds on computing error-correcting codes by bounded-depth circuits with arbitrary gates. *IEEE Trans. Inf. Theory*, 59(10):6611–6627, 2013. doi:10.1109/TIT.2013.2270275.

24 Craig Gentry. Fully homomorphic encryption using ideal lattices. In Michael Mitzenmacher, editor, *Proceedings of the 41st Annual ACM Symposium on Theory of Computing, STOC 2009, Bethesda, MD, USA, May 31 - June 2, 2009*, pages 169–178. ACM, 2009. doi:10.1145/1536414.1536440.

25 Oded Goldreich, Silvio Micali, and Avi Wigderson. How to play any mental game or a completeness theorem for protocols with honest majority. In *Proceedings of the Nineteenth Annual ACM Symposium on Theory of Computing (STOC)*, pages 218–229. ACM, 1987. doi:10.1145/28395.28420.

26 Venkatesan Guruswami, Johan Håstad, and Swastik Kopparty. On the list-decodability of random linear codes. *IEEE Transactions on Information Theory*, 57(2):718–725, 2011. doi:10.1109/TIT.2010.2095170.

27 Venkatesan Guruswami and Jonathan Mosheiff. Punctured low-bias codes behave like random linear codes. In *63rd IEEE Annual Symposium on Foundations of Computer Science, FOCS 2022, Denver, CO, USA, October 31 - November 3, 2022*, pages 36–45. IEEE, 2022. doi:10.1109/FOCS54457.2022.00011.

28 Danny Harnik, Yuval Ishai, Eyal Kushilevitz, and Jesper Buus Nielsen. Ot-combiners via secure computation. In Ran Canetti, editor, *Theory of Cryptography, Fifth Theory of Cryptography Conference, TCC 2008, New York, USA, March 19-21, 2008*, volume 4948 of *Lecture Notes in Computer Science*, pages 393–411. Springer, 2008. doi:10.1007/978-3-540-78524-8_22.

29 Hamed Hatami, Pooya Hatami, and Shachar Lovett. Higher-order fourier analysis and applications. *Foundations and Trends® in Theoretical Computer Science*, 13(4):247–448, 2019. doi:10.1561/0400000064.

30 Yuval Ishai, Eyal Kushilevitz, Rafail Ostrovsky, and Amit Sahai. Cryptography with constant computational overhead. In Cynthia Dwork, editor, *Proceedings of the 40th Annual ACM Symposium on Theory of Computing, Victoria, British Columbia, Canada, May 17-20, 2008*, pages 433–442. ACM, 2008. doi:10.1145/1374376.1374438.

31 Yuval Ishai, Manoj Prabhakaran, and Amit Sahai. Secure arithmetic computation with no honest majority. In Omer Reingold, editor, *Theory of Cryptography, 6th Theory of Cryptography Conference, TCC 2009, San Francisco, CA, USA, March 15-17, 2009. Proceedings*, volume 5444 of *Lecture Notes in Computer Science*, pages 294–314. Springer, 2009. doi:10.1007/978-3-642-00457-5_18.

32 Ray Li and Mary Wootters. Improved list-decodability of random linear binary codes. *IEEE Transactions on Information Theory*, 67(3):1522–1536, 2020. doi:10.1109/TIT.2020.3041650.

33 Yuan Li. Secret sharing on superconcentrator, 2026. arXiv:2302.04482.

34 James L Massey. Some applications of coding theory in cryptography. In *Codes and Cyphers: Cryptography and Coding IV*, pages 33–47. Forma, 1995.

35 Jonathan Mosheiff, Nicolas Resch, Noga Ron-Zewi, Shashwat Silas, and Mary Wootters. Low-density parity-check codes achieve list-decoding capacity. *SIAM J. Comput.*, 53(6):S20–38, 2024. doi:10.1137/20M1365934.

36 Jonathan Mosheiff, Nicolas Resch, Kuo Shang, and Chen Yuan. Randomness-efficient constructions of capacity-achieving list-decodable codes. In *2025 IEEE International Symposium on Information Theory (ISIT)*, pages 1–6, 2025. doi:10.1109/ISIT63088.2025.11195274.

37 Rafail Ostrovsky, Sridhar Rajagopalan, and Umesh V. Vazirani. Simple and efficient leader election in the full information model. In Frank Thomson Leighton and Michael T. Goodrich, editors, *Proceedings of the Twenty-Sixth Annual ACM Symposium on Theory of Computing, 23-25 May 1994, Montréal, Québec, Canada*, pages 234–242. ACM, 1994. doi:10.1145/195058.195141.

38 Anna Pagh, Rasmus Pagh, and Milan Ružić. Linear probing with constant independence. In *Proceedings of the thirty-ninth annual ACM symposium on Theory of computing*, pages 318–327, 2007.

39 Anup Rao. An exposition of bourgain's 2-source extractor. *Electron. Colloquium Comput. Complex.*, TR07-034, 2007. URL: https://eccc.weizmann.ac.il/eccc-reports/2007/TR07-034/index.html.

40 Daniel A. Spielman. Linear-time encodable and decodable error-correcting codes. *IEEE Transactions on Information Theory*, 42(6):1723–1731, 1996. doi:10.1109/18.556668.

41 Salil P. Vadhan. Pseudorandomness. *Foundations and Trends in Theoretical Computer Science*, 7(1–3):1–336, 2012. doi:10.1561/0400000010.

42 Andrew Chi-Chih Yao. How to generate and exchange secrets. In *27th Annual Symposium on Foundations of Computer Science (FOCS 1986)*, pages 162–167. IEEE, 1986.

Resilience of Inner-Product Masking Scheme Against Hamming Weight Leakage

Aniruddha Biswas
Department of Computer Science, Purdue University, West Lafayette, IN, USA

Jihun Hwang
Department of Computer Science, Purdue University, West Lafayette, IN, USA

Hemanta K. Maji
Department of Computer Science, Purdue University, West Lafayette, IN, USA

Xiuyu Ye
Department of Computer Science, Purdue University, West Lafayette, IN, USA

Abstract

Additive masking is a widely used countermeasure against side-channel attacks in which a secret is additively split into multiple random shares. However, over binary fields, the number of 1's in the binary representation (i.e., the Hamming weight) of the shares reveals information about the original secret. Inner-product masking scheme has been proposed as a promising alternative that is secure against such information leakage.

In this work, we establish that inner-product masking over a binary extension field is provably secure against Hamming weight leakage, and it translates into security against arbitrary symmetric function leakage from the shares. In addition, we present an efficiently computable score function that quantifies its security against leakages, enabling users to test and certify its security. Finally, we derive a relationship between the leakage resilience of inner-product masking and additive masking over arbitrary fields; roughly speaking, they are at least as secure as additive masking. Our approach is Fourier-analytic and involves estimating spectral norms of the Hamming slice by studying Krawtchouk polynomials.

2012 ACM Subject Classification Theory of computation → Cryptographic primitives; Security and privacy → Cryptanalysis and other attacks

Keywords and phrases Local leakage resilience, additive masking, inner product masking, Hamming weight leakage, Fourier analysis, Krawtchouk polynomials

Digital Object Identifier 10.4230/LIPIcs.ITC.2026.9

Related Version *Full Version*: `https://www.cs.purdue.edu/homes/hmaji/papers/BHMY26.pdf`

Funding *Aniruddha Biswas*: Partly supported by CCF-2327981.
Hemanta K. Maji: Partly supported by CNS–2055605 and CCF–2327981.

Acknowledgements Authors thank Yuval Filmus for discussions on the Krawtchouk polynomial.

1 Introduction

Side-channel attacks have repeatedly compromised cryptographic secrets by exploiting observable side effects of a device's physical operation. *Masking* [9,21] is a widely prevalent and effective countermeasure; it splits the secret into shares, relying on the noise in measurements from individual shares to accumulate quickly and obscure the secret.

Additive masking, a prominent masking strategy, splits a secret s into random $(s_1, s_2, \ldots, s_n)$ shares satisfying $s_1 + s_2 + \cdots + s_n = s$. The security provided by this scheme has its limitations; even with noisy measurements, a carefully chosen attack can still reveal partial information about the secret. For example, over any finite field, single-bit probes into the devices storing the shares reveal information about the secret [1, 30, 31, 33]. *Over binary*

extension fields, even power analysis and timing attacks [26, 27] *that reveal only the number of set bits of shares (a.k.a., their Hamming weight) compromise this masking – the secret's Hamming weight and the sum of all the leaked Hamming weights always have the same parity.* However, over exceptionally specialized and rare Mersenne-prime fields, weak security guarantees are known against such attacks [17].

In comparison, *inner-product masking* [2,3], a mild generalization of additive masking, has demonstrated significant potential for achieving greater security across all finite fields. Such a scheme is parameterized by its *reconstruction multipliers* $\vec{\beta} \in (F^*)^n$, where F is the ambient finite field, and the corresponding shares $(s_1, s_2, \ldots, s_n)$ satisfy $\beta_1 \cdot s_1 + \beta_2 \cdot s_2 + \cdots + \beta_n \cdot s_n = s$. Observe that $\vec{\beta} = (1, 1, \ldots, 1)$ reduces it to additive masking; however, carefully selecting these multipliers $\vec{\beta}$ can significantly enhance security. For instance, over any field, nearly all multipliers $\vec{\beta}$ offer strong security guarantees when the number of set bits of each share is revealed [18]. While nearly all multipliers offer strong security, we cannot identify them, nor can we determine the level of security provided by a specific multiplier.

This work contributes to the ongoing investigations into the security of inner product masking and, by extension, provides guidelines for NIST's ongoing standardization efforts for more secure secret-sharing schemes [7].

Summary of our results. As in [17, 18], we investigate the *local leakage resilience* [4, 5] of inner-product masking against *Hamming weight leakage* from shares, where the number of set bits in each share is revealed. This is a strong security metric, requiring statistical independence between the secret and the leakage, and it implies other security metrics in the literature, such as statistical bias [13], mutual information [19], and guessing entropy [40].

In this work, we establish that inner-product masking is *always* as secure as additive masking over any field, thereby ruling out the possibility of "vulnerable" multipliers that could undermine security – a gap left open in the analysis of [18]. Over binary extension fields, while additive masking is vulnerable, as indicated above, we prove that it is (essentially) the *only* vulnerable scheme; all other inner product masking schemes are secure. We present an efficiently computable *score function* for multipliers, where a higher score indicates greater security of the corresponding inner-product masking. Our score function enables *certifying* the security achieved by a specific multiplier, facilitating efforts similar to those in [14–16]. It helps explicitly identify multipliers with strong security, whose existence was proven in [18]. From a randomness extractor perspective: the inner product with random (public) multipliers is already a strong extractor; our contribution is to deterministically identify multipliers that yield a good inner-product extractor.

The results above stem from more general technical findings, also applicable to the local leakage resilience of secret-sharing schemes and polynomial masking [20, 39]. Our approach is Fourier-analytic and relies on spectral-norm estimates of the Hamming slice, which, in turn, require the asymptotic properties of Krawtchouk polynomials [29, 41].

1.1 Preliminaries

Basic notations. Here $x = a \pm c$ denotes $x \in [a - c, a + c]$, for reals x and a with $c > 0$. $\mathrm{card}(S)$ denotes the cardinality of the set S. Let F_q denote the finite field of order (cardinality) q and $F_q^* := F_q \setminus \{0\}$. Let $\lambda \in \{1, 2, \ldots\}$ satisfy $2^{\lambda-1} < q \leqslant 2^\lambda$, the number of bits needed to represent the finite field elements.

Field extensions. Elements of an extension field F_{2^λ} are represented as $F_2[Z]/\Pi(Z)$ elements, where $\Pi(Z)$ is a monic irreducible polynomial of degree λ. We can view F_{2^λ} as a vector space F_2^λ after choosing a basis $\mathcal{B} = (b_1, \ldots, b_\lambda)$. The *Hamming weight* $\mathrm{wt}_\mathcal{B}(\zeta) \in \{0, 1, \ldots, \lambda\}$ is

the number of 1's in the vector representation of $\zeta \in F_{2^\lambda}$ in F_2^λ under $\mathcal{B}$; we omit $\mathcal{B}$ unless it has to be specified. Let $\mathsf{Tr}_{F_{2^\lambda}/F_2}\colon F_{2^\lambda} \to F_2$ be the field trace of the field extension F_{2^λ}/F_2. The *trace-dual coordinate vector* of $\zeta \in F_{2^\lambda}$ is $\zeta^\vee := (\mathsf{Tr}_{F_{2^\lambda}/F_2}(\zeta b_1), \ldots, \mathsf{Tr}_{F_{2^\lambda}/F_2}(\zeta b_\lambda)) \in F_2^\lambda$ and its Hamming weight $\mathrm{wt}(\zeta^\vee) =: \mathrm{wt}^\vee(\zeta)$ is called the *trace-dual Hamming weight* of ζ.

Secret sharing and masking. Let Add_n denote the *additive secret sharing* among n parties; to share a secret $s \in F$, it provides random shares $(s_1, s_2, \ldots, s_n) \in F^n$ conditioned on $s_1 + s_2 + \cdots + s_n = s$. The *generalized additive secret sharing* $\mathsf{Gen}(\mathsf{Add}_n, \vec{\beta})$ is parameterized by $\vec{\beta} = (\beta_1, \ldots, \beta_n) \in (F^*)^n$ and its shares satisfy $\beta_1 \cdot s_1 + \beta_2 \cdot s_2 + \cdots + \beta_n \cdot s_n = s$ instead (β_i's need not be all distinct); the additive scheme corresponds to $\vec{\beta} = (1, \ldots, 1)$. The additive and generalized additive secret sharing are equivalent to additive and inner-product masking, respectively.

Leakage resilience. Fix a secret sharing scheme $\mathcal{S}$. Let $\tau_i\colon F \to \Omega_i$ be a leakage function, and let $\vec{\tau}(s)$ denote the joint leakage distribution $(\tau_1(s_1), \ldots, \tau_n(s_n))$, where $\mathcal{S}$ generates shares $(s_1, \ldots, s_n)$ of s. $\vec{\tau}(U_F)$ will denote the leakage distribution when the secret is chosen uniformly at random from F. The *local leakage resilience* of $\mathcal{S}$ against the leakage attack $\vec{\tau}$ is defined by [4, 5]:

$$\varepsilon(\mathcal{S}; \vec{\tau}) \;:=\; \max_{s \in F} \mathrm{SD}(\vec{\tau}(s)\ ,\ \vec{\tau}(U_F)) \tag{1}$$

In local leakage resilience context, the insecurity of $\mathsf{Gen}(\mathsf{Add}_n,\ (c, \ldots, c)\ )$, for $c \in F^*$, is identical to the Add_n's insecurity; hence, they are *essentially equivalent*.

Our score function. Define $\sigma\colon \{0, 1, \ldots, \lambda\} \to \mathbb{R}$ as follows[1]

$$\sigma(w) \;:=\; \begin{cases} 0, & \text{if } w \in \{0, \lambda\} \\ \frac{1}{2}\log\binom{\lambda}{w} - \frac{\log\lambda}{4}, & \text{otherwise} \end{cases}. \tag{2}$$

Note that this function can be computed efficiently [6, 8].

▶ Remark 1 (Behavior of σ). Since $\binom{\lambda}{w} \geqslant (\lambda/w)^w$, we have

$$\sigma(w) \geqslant \frac{w^*}{2} \cdot \log\left(\frac{\lambda}{w^*}\right) - \frac{1}{4} \cdot \log\lambda$$

where $w^* = \min\{w, \lambda - w\}$ and $w \in \{1, 2, \ldots, \lambda - 1\}$. Consequently, either $\sigma(w) = 0$ or $\sigma(w) \geqslant (1/4) \cdot \log\lambda$.

Asymptotically, by central limit theorem, $\binom{\lambda}{\frac{\lambda}{2} \pm x}$ behaves like $\frac{2^\lambda}{\sqrt{\pi\lambda/2}} \cdot \exp\left(-\frac{2x^2}{\lambda}\right)$. So, $\sigma(\lambda/2 \pm x)$ behaves like $\frac{\log 2}{2}\lambda - \frac{1}{2}\log\lambda - \frac{x^2}{\lambda}$. When w is drawn according to the binomial distribution, the expected value of $\sigma(w)$ is roughly $\frac{\log 2}{2}\lambda$, a consequence of the entropy of the binomial distribution.

For $\vec{\beta} \in (F^*)^n$ and $\zeta \in F$, we define $\mathsf{Score}\colon F \times (F^*)^n \to \mathbb{R}$ as follows.

$$\mathsf{Score}(\zeta; \vec{\beta}) \;:=\; \sum_{i=1}^{n} \sigma\big(\ \mathrm{wt}^\vee(\beta_i \cdot \zeta)\ \big). \tag{3}$$

[1] For intuition, `https://www.desmos.com/calculator/zerikstukw` graphs the $\sigma(\cdot)$ function for a representative λ.

Let $\zeta^* \in F$ be the unique element satisfying $\zeta^{*\vee} = (1, \ldots, 1)$, equivalently $\mathrm{wt}^\vee(\zeta^*) = \lambda$. For a tuple $\vec{\beta} = (\beta_1, \beta_2, \ldots, \beta_n) \in (F^*)^n$, define

$$\mathcal{S}_{\vec{\beta}} := \left\{ \zeta^* \cdot \beta_1^{-1},\ \zeta^* \cdot \beta_2^{-1},\ \ldots,\ \zeta^* \cdot \beta_n^{-1} \right\} \subseteq F^*. \tag{4}$$

Looking ahead, our security characterization will depend on the minimum "score" $\mathsf{Score}(\zeta; \vec{\beta})$ over $\zeta \in \mathcal{S}_{\vec{\beta}}$; insecurity will be small when $\vec{\beta}$ has a *large minimum score*.

1.2 Our Results

Over binary extension fields. Over $F = F_{2^\lambda}$, the attack in the introduction demonstrates the vulnerability of additive secret sharing to Hamming weight leakage. We begin by showing that transitioning to the generalized setting eliminates this vulnerability.

▶ **Informal Theorem 4.** *For $n \in \{3, 4, \ldots\}$ parties, $\mathsf{Gen}(\mathsf{Add}_n, \vec{\beta})$ is $\lambda^{-1/4}$-insecure, unless $\vec{\beta} = (b, b, \ldots, b)$ for $b \in F^*$.*

Except for those inner product masking that are essentially equivalent to the additive masking in the local leakage resilience context, everyone else is secure. Theorem 4 states this result formally. Next, we characterize choices of $\vec{\beta} \in (F^*)^n$ that allow us to derive stronger security bounds for $\mathsf{Gen}(\mathsf{Add}_n, \vec{\beta})$, where insecurity decays $\lambda^{-\Theta(n)}$. This characterization builds on the following technical result.

▶ **Informal Theorem 8.** *For $n \in \{3, 4, \ldots\}$ parties and $\vec{\beta} \in (F^*)^n$, $\mathsf{Gen}(\mathsf{Add}_n, \vec{\beta})$ is λ^{-cn}-insecure, where $c \in (0, 1/2 - 1/n)$ satisfies*

$$\min_{\zeta \in \mathcal{S}_{\vec{\beta}}} \mathsf{Score}\left(\zeta; \vec{\beta}\right) \geqslant cn \cdot \log \lambda. \tag{5}$$

and the Score *function is defined in Equation 3.*

We leave the case $n = 2$ as an open problem for both results. For clarity of presentation, we ignore additive $\log\log\lambda$ terms in the right-hand side of the expression above (Equation 5). See Theorem 8 for the full expression.

Given $\vec{\beta}$, one can efficiently compute its trace-dual weights and test whether the minimum score is $\geqslant cn \log \lambda$ or not, i.e. verify Equation 5 (see for example [42]). For example, if all elements of $\vec{\beta}$ are identical, we know already that the $\mathsf{Gen}(\mathsf{Add}_n, \vec{\beta})$ is insecure. In this case, the minimum score is 0 and our technical result cannot upper bound the insecurity by a small quantity.

▶ Remark 2 (Consequences of Theorem 4 and 8). The following corollaries are immediate:

1. If every element in $\vec{\beta}$ occurs at most m times, then $\mathsf{Gen}(\mathsf{Add}_n, \vec{\beta})$ satisfies Equation 5 with $c = (n - m)/4n$. This is because for $\zeta \in \mathcal{S}_{\vec{\beta}}$, $\beta_i \zeta$ is either equal to ζ^* or has dual weight $\in \{1, 2, \ldots, \lambda - 1\}$. Since every element in $\vec{\beta}$ occurs at most m times, at least $(n - m)$ elements in $\{\zeta\beta_1, \ldots, \zeta\beta_n\}$ have dual weight $\in \{1, 2, \ldots, \lambda - 1\}$. Therefore, the score is $\geqslant (n - m) \cdot (1/4) \log \lambda$; whence, the result. In particular, if all elements of $\vec{\beta}$ are distinct, then $m = 1$ and $c = (n - 1)/4n$. Moreover, when $\vec{\beta}$ is chosen uniformly at random from F^n, then with exponentially high probability, it consists of distinct elements from F^*.
2. A *symmetric* function $f \colon F \to \Omega$ satisfies the constraint "$\mathrm{wt}(x) = \mathrm{wt}(y)$ implies $f(x) = f(y)$" for $x, y \in F$. Let $\mathcal{F}$ denote the set of all symmetric functions. A *symmetric local leakage* leaks $f_1, \ldots, f_n$ from the shares $s_1, \ldots, s_n$, where $f_1, \ldots, f_n \in \mathcal{F}$.

Note that given only the $\mathrm{wt}(x)$, one can compute $f(x)$ for a symmetric function; i.e., $x \to \mathrm{wt}(x) \to f(x)$ is a Markov chain. Therefore, by the data processing inequality (see [10, Chapter 2]), any secret sharing scheme that has ε insecurity against the Hamming weight leakage has ε insecurity against *every* symmetric local leakage.

Theorem 8 also gives a systematic way to select reconstruction multipliers $\vec{\beta} \in (F^*)^n$.

▶ **Informal Corollary 9.** *Let $\vec{\beta} = (\beta_1, \ldots, \beta_n) \in (F^*)^n$ If for every β_i there exists another β_j such that $\beta_j \zeta^* \beta_i^{-1}$ has non-extreme (close to neither 0 nor λ) dual Hamming weight, then $\mathsf{Gen}(\mathsf{Add}_n, \vec{\beta})$ satisfies Equation 5 for all sufficiently large λ.*

We give a concrete instantiation below, where we construct multipliers $\vec{\beta} \in (F^*)^n$ from the irreducible polynomial $\Pi(Z) \in F_2[Z]$ defining $F = F_{2^\lambda} \cong F_2[Z]/\Pi(Z)$. Since the relevant trace-dual weights may depend on $\Pi(Z)$, the choice of $\vec{\beta}$ varies with $\Pi(Z)$. See full version for more general recommendations and their proofs.

Recommendations for $\vec{\beta}$: an illustrative example

Suppose we are given an insecurity budget of 2^{-48} and $n = 16$, a fairly reasonable expectation in practice. We make the following recommendations based on our technical findings.

1. If $\Pi(Z) = Z^\lambda + Z + 1$, choose $\vec{\beta} = (\beta, 1, \ldots, 1) \in (F^*)^n$, where $\beta = Z^{\lfloor \lambda/2 \rfloor}$. For $\lambda \geqslant 256$, this gives the desired insecurity bound.
2. If $\Pi(Z) = Z^\lambda + Z^j + 1$ where $j > 1$ is a small constant, choose $\vec{\beta} = (\beta, 1, \ldots, 1) \in (F^*)^n$, where $\beta = Z^i$, where i is the multiple of j closest to $\lambda/2$. Again, for $\lambda \geqslant 256$ and the usual choices of j, this gives the desired insecurity bound.
3. If $\Pi(Z) = Z^\lambda + Z^j + 1$ where j is already comparable to $\lambda/2$, choose $\vec{\beta} = (\beta, 1, \ldots, 1) \in (F^*)^n$, where $\beta = Z^j$. This is the simplest choice in the large-j trinomial case.

Corollary 9 also allows us to extend our results to the security of Shamir's secret sharing with random evaluation points; see Remark 31.

Over arbitrary fields. Next, we show that generalized additive secret sharing inherits its security from the additive secret sharing, and is therefore *at least as secure as* the additive secret sharing, in the sense that its insecurity is bounded above by a quantity that upper bounds the insecurity of additive secret sharing.

▶ **Informal Theorem 10.** *Let F_q be a finite field of order q and $\vec{\beta} \in (F_q^*)^n$. Then $\mathsf{Gen}(\mathsf{Add}_n, \vec{\beta})$ is secure against the leakage $\vec{\tau} = (\tau_1, \tau_2, \ldots, \tau_n)$ if Add_n is "sufficiently secure" on average over $i \in \{1, 2, \ldots, n\}$ against $\vec{\tau_i} = (\tau_i, \tau_i, \ldots, \tau_i)$.*

This result implies that one could verify the security of a generalized additive secret sharing against a certain leakage, by testing whether the additive secret sharing (i.e. when all reconstruction multipliers are 1) is *sufficiently secure* against leakage attacks where the same leakage function is applied to every share. Here, the notion of strong security comes from the fact that we need the upper bound of insecurity to be small, not the insecurity itself. See Theorem 10 for the precise statement with exact expressions.

It is worthwhile to note that Theorem 10 applies to *any* finite field. Consequently, we can also derive security guarantees against Hamming weight leakage for schemes over prime fields, including those based on Mersenne primes, by combining our generalized bound with the results of Faust et al. [17].

▶ Remark 3 (Technical subtlety in Theorem 10). Benhamouda et al. [4] introduced the quantity called *Fourier proxy* that measures the local leakage resilience (Equation 1). Proving that this proxy (or its upper bound) is small is a popular technique to demonstrate that the insecurity is small [4, 5, 24, 25, 30, 33, 34, 37].

In Theorem 10, we show that the proxy of the inner product masking is at most an upper bound of the proxy for additive masking, induced by triangle inequality, hence the terminology *sufficiently secure*. We *did not* prove that the insecurity of the inner product masking is at most the insecurity of the additive scheme directly.

1.3 Technical Overview

Our results are best interpreted by considering the number of parties $n \in \{3, 4, \dots\}$ to be a constant and determining the asymptotics of insecurity as a function of the security parameter λ. Consider $\mathsf{Gen}(\mathsf{Add}_n, \vec{\beta})$, for arbitrary $\vec{\beta} \in (F^*)^n$. In this section, we illustrate our analysis strategy against the Hamming weight leakage; we present formal proofs formally in Section 3.

The analysis relies on spectral-norm estimates of the Hamming slice, which in turn require the asymptotic properties of Krawtchouk polynomials [29, 41]. Existing techniques such as bounding the spectral norm of the Hamming slice via polynomial threshold function estimates, do not yield bounds sufficiently tight for our purposes. See Figure 1 for a visualization of the gap between spectral weights at each level; see full version for the further discussion. One modern tool for estimating the spectral norm of Hamming slices is the use of level-k inequalities; details are likewise deferred to the full version.

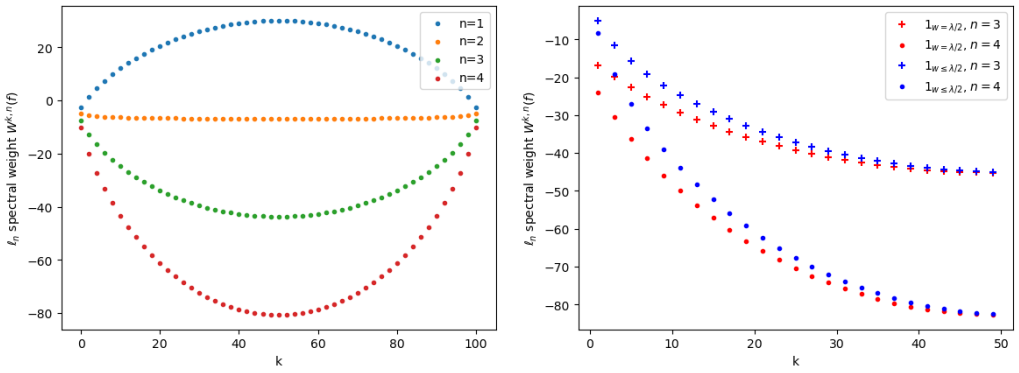


Figure 1 Distribution of ℓ_n-spectral weight at level-k of $f\colon F_{2^\lambda} \to \{0,1\}$, i.e. $W^{k,n}(f) := \sum_{S\colon \mathrm{wt}(S)=k} |\hat{f}(S)|^n$.
[Left] x-axis: $k \in \{0, 1, \dots, \lambda\}$. y-axis: $W^{k,n}(\mathbb{1}_{w=\lambda/2})$ (log-scaled) for $n = 1$ (blue), 2 (orange), 3 (green), and 4 (red). Note the change in trend as n increases.
[Right] x-axis: $k \in \{0, 1, \dots, \lambda/2\}$. y-axis: log-scaled $W^{k,n}(\mathbb{1}_{w=\lambda/2})$ (red) and log-scaled $W^{k,n}(\mathbb{1}_{w\leqslant\lambda/2})$ (blue) for $n = 3$ (+ markers) and 4 (• markers). Note the gap between $\mathbb{1}_{w=\lambda/2}$ (red) and log-scaled $\mathbb{1}_{w\leqslant\lambda/2}$.

Main Technical Result and Proof Outline

▶ **Theorem 4.** *For* $\vec{\beta} \in (F^*)^n \setminus \{(b, \dots, b)\colon b \in F^*\}$ *and* $n \in \{3, 4, \dots\}$, *we have*

$$\varepsilon\Big(\ \mathsf{Gen}(\mathsf{Add}_n, \vec{\beta})\ ;\ \overrightarrow{\mathrm{wt}}\ \Big) = \mathcal{O}\Big(\lambda^{-1/4}\Big).$$

Recall that $\overrightarrow{\mathrm{wt}}(s)$, represents the Hamming weight leakage distribution over the sample space $\{0,1,\ldots,\lambda\}^n$ corresponding to a secret $s \in F$. Similarly, $\overrightarrow{\mathrm{wt}}(U_F)$ represents the joint Hamming weight distribution when the secret is randomly picked from F. We aim to upper-bound the statistical distance between these two probability distributions.

$$2 \cdot \mathrm{SD}\big(\overrightarrow{\mathrm{wt}}(s)\ ,\ \overrightarrow{\mathrm{wt}}(U_F)\big) = \sum_{\vec{w}\in\{0,1,\ldots,\lambda\}^n} \left| \Pr_{\vec{s}\leftarrow \mathsf{Gen}(\mathsf{Add}_n,\vec{\beta};s)}[\overrightarrow{\mathrm{wt}}(\vec{s})=\vec{w}] - \Pr_{\vec{s}\leftarrow \mathsf{Gen}(\mathsf{Add}_n,\vec{\beta};U_F)}[\overrightarrow{\mathrm{wt}}(\vec{s})=\vec{w}] \right|$$

Here, $\mathsf{Gen}(\mathsf{Add}_n, \vec{\beta}; s)$ denotes the distribution of shares over F^n for secret $s \in F$, and $\mathsf{Gen}(\mathsf{Add}_n, \vec{\beta}; U_F)$ the distribution of shares for a uniformly random secret in F. Note that each share s_i is uniformly random over F, irrespective of the secret. Therefore, the (marginal) distribution of the Hamming weight of any share is the binomial distribution $B(\lambda, 1/2)$. By Chernoff bound (Lemma 24), it is unlikely that the Hamming weight of any share is significantly far from $\lambda/2$. In particular, for a parameter $\tau \in (0, 1/2]$, the Hamming weight of a share being $\geqslant \lambda^{1/2+\tau}$ far from $\lambda/2$, is at most $2 \cdot \exp(-2\lambda^{2\tau})$. We can control this probability of "atypical weights" by setting $\tau = \frac{\log((n/4)\cdot\log\lambda)}{2\log\lambda}$ and driving the probability below $\lambda^{-n/2}$.

We define the set of all *typical* leakages $\mathsf{Typical}(n,\tau) \subseteq \{0,1,\ldots,\lambda\}^n$ parameterized by τ:

$$\mathsf{Typical}(n,\tau) := \Big\{\vec{w} :\ |w_i - \lambda/2| \leqslant \lambda^{1/2+\tau},\ \text{for all } i \in \{1,2,\ldots,n\}\Big\}. \tag{6}$$

The probability that the Hamming weight leakage of any share for secret s or random secret U_F is outside this $\mathsf{Typical}(n,\tau)$ set is (at most) $4n \cdot \exp\big(-2\lambda^{2\tau}\big)$ by the union bound. So, we conclude that

$$2 \cdot \mathrm{SD}\big(\overrightarrow{\mathrm{wt}}(s)\ ,\ \overrightarrow{\mathrm{wt}}(U_F)\big) = \sum_{\vec{w}\in\mathsf{Typical}(n,\tau)} \left| \Pr_{\vec{s}\leftarrow \mathsf{Gen}(\mathsf{Add}_n,\vec{\beta};s)}[\overrightarrow{\mathrm{wt}}(\vec{s})=\vec{w}] - \Pr_{\vec{s}\leftarrow \mathsf{Gen}(\mathsf{Add}_n,\vec{\beta};U_F)}[\overrightarrow{\mathrm{wt}}(\vec{s})=\vec{w}] \right| \pm 4n \cdot \lambda^{-n/2}.$$

and therefore it suffices to estimate the leakage probability restricted to typical leakages $\vec{w} \in \mathsf{Typical}(n,\tau)$.

To this end, we will use Fourier analysis. Using the Poisson summation formula (Lemma 25) for the generalized additive secret sharing, we can rewrite the right-hand side expression in terms of Fourier coefficients.

$$2 \cdot \mathrm{SD}\big(\overrightarrow{\mathrm{wt}}(s)\ ,\ \overrightarrow{\mathrm{wt}}(U_F)\big) = \sum_{\vec{w}\in\mathsf{Typical}(n,\tau)} \left| \sum_{\zeta\in F^*} \left(\prod_{i=1}^{n} \widehat{\mathbb{1}}_{w_i}(\beta_i\zeta) \right) \cdot \chi_1\Big(s\cdot\zeta\cdot\big\langle\vec{\beta},\vec{v}\big\rangle\Big) \right| \pm 4n \cdot \lambda^{-n/2}.$$

Here, $\mathbb{1}_w \colon F \to \{0,1\}$ is the indicator function of the set all elements in F with Hamming weight $w \in \{0,1,\ldots,\lambda\}$. The shares $\vec{v} \in F^n$ is an arbitrary share of the secret $1 \in F$. Moreover, $\chi_1 \colon F \to \mathbb{C}$ is defined as $\chi_1(x) := (-1)^{\mathsf{Tr}_{F/F_2}(x)}$ (see Section 2.4 for details). To upper-bound the right-hand side expression, we apply the triangle inequality and conclude:

$$2 \cdot \mathrm{SD}\big(\overrightarrow{\mathrm{wt}}(s)\ ,\ \overrightarrow{\mathrm{wt}}(U_F)\big) \leqslant \sum_{\vec{w}\in\mathsf{Typical}(n,\tau)} \sum_{\zeta\in F^*} \prod_{i=1}^{n} \left|\widehat{\mathbb{1}}_{w_i}(\beta_i\zeta)\right| + 4n \cdot \lambda^{-n/2}.$$

Now, we need to upper-bound the magnitude of $\widehat{\mathbb{1}}_w(\cdot)$, for typical $w \in \{0, 1, \ldots, \lambda\}$. We remind the reader that the right-hand side expression is not small for every $\vec{\beta} \in F^n$. For instance, when $\vec{\beta} = \vec{1} = (1, 1, \ldots, 1) \in F^n$ we know an attack on the corresponding secret sharing scheme. Thus, the resulting secret sharing scheme must be insecure and the right-hand side expression corresponding to that insecure scheme must be large. So, the right-hand side estimate will be small depending on $\vec{\beta}$; our analysis cannot be transparent to this fact.

To approach this estimation problem, and appreciate our key technical components, let us build some elementary intuition of the magnitude of the Fourier coefficients $\widehat{\mathbb{1}}_w(\cdot)$. We remark that $\widehat{\mathbb{1}}_w(x) = 2^{-\lambda} K_w(\mathrm{wt}^\vee(x))$ where $K_w(\cdot)$ is a Krawtchouk polynomial, as defined in [28]. Krawtchouk polynomials find extensive applications in many subfields in mathematics, theoretical computer science, and cryptography; see, for example, [41].

1. First, by symmetry $\widehat{\mathbb{1}}_w(x) = \widehat{\mathbb{1}}_w(y)$, for all $x, y \in F$ satisfying $\mathrm{wt}^\vee(x) = \mathrm{wt}^\vee(y)$.
2. Next, $\widehat{\mathbb{1}}_w(x) = (-1)^w \cdot \widehat{\mathbb{1}}_w(y)$, for all $x, y \in F$ satisfying $\mathrm{wt}^\vee(x) + \mathrm{wt}^\vee(y) = \lambda$, because $\mathbb{1}_w$ is a real-valued function.

As we will see, the contributions of $\widehat{\mathbb{1}}_w(x)$, where $\mathrm{wt}^\vee(x) \in \{1, 2, \ldots, \lambda - 1\}$, is relatively small; *this fact is non-trivial to prove and is one of our technical contributions.* The contributions of $\widehat{\mathbb{1}}_w(0)$ and $\widehat{\mathbb{1}}_w(\zeta^*)$, where $\zeta^* \in F$ is the unique element of F with $\mathrm{wt}^\vee(\zeta^*) = \lambda$, are large. Both of them have magnitude equal to the density of the subset of all weight-w elements of F. However, note that $\widehat{\mathbb{1}}_w(0)$ never occurs on the right-hand side expression, because the right-hand side expression only considers $\zeta \in F^*$ and all β_i are also non-zero. The potential "troublemakers" are those terms where $\widehat{\mathbb{1}}_w(\zeta^*)$ appears.

To account for them, we identify the set of candidate $\zeta \in F$ such that $\zeta \cdot \beta_i = \zeta^*$ for some $i \in \{1, 2, \ldots, n\}$.

$$\mathcal{S}_{\vec{\beta}} \;:=\; \left\{ \zeta^* \cdot \beta_1^{-1},\; \zeta^* \cdot \beta_2^{-1},\; \ldots,\; \zeta^* \cdot \beta_n^{-1} \right\}.$$

Note that the cardinality of $\mathcal{S}_{\vec{\beta}}$ is the number of distinct elements in $\vec{\beta}$, which is at most n. We split the right-hand side expression depending on whether $\zeta \in \mathcal{S}_{\vec{\beta}}$ or not, then we estimate them separately:

$$2 \cdot \mathrm{SD}\big(\overrightarrow{\mathrm{wt}}(s)\;,\; \overrightarrow{\mathrm{wt}}(U_F)\big) \leqslant \underbrace{\sum_{\vec{w} \in \mathsf{Typical}(n,\tau)} \sum_{\zeta \in \mathcal{S}_{\vec{\beta}}} \prod_{i=1}^{n} \left| \widehat{\mathbb{1}}_{w_i}(\beta_i \zeta) \right|}_{\text{first summand}} + \underbrace{\sum_{\vec{w} \in \mathsf{Typical}(n,\tau)} \sum_{\zeta \in F^* \setminus \mathcal{S}_{\vec{\beta}}} \prod_{i=1}^{n} \left| \widehat{\mathbb{1}}_{w_i}(\beta_i \zeta) \right|}_{\text{second summand}} + 4n \cdot \lambda^{-n/2}.$$

Second summand estimation. We prove an upper bound on the magnitude $|\widehat{\mathbb{1}}_{w_i}(x)| \leqslant B(x)$ as defined in Lemma 27, for all $x \in F \setminus \{0, \zeta^*\}$. Using this bound, we have:

$$\begin{aligned} \text{second summand} &\leqslant \sum_{\vec{w} \in \mathsf{Typical}(n,\tau)} \sum_{\zeta \in F^* \setminus \mathcal{S}_{\vec{\beta}}} \prod_{i=1}^{n} B(\beta_i \zeta) \\ &= \left(2\lambda^{1/2+\tau}\right)^n \cdot \left(\sum_{\zeta \in F^* \setminus \mathcal{S}_{\vec{\beta}}} \prod_{i=1}^{n} B(\beta_i \zeta) \right). \end{aligned}$$

The last equality substitutes the cardinality of the $\mathsf{Typical}(n,\tau)$ set. Now, using (generalized) Hölder's inequality (Lemma 26), we have

$$\text{second summand} \leqslant \left(2\lambda^{1/2+\tau}\right)^n \cdot \left(\sum_{\zeta \in F\setminus\{0,\zeta^*\}} B(\zeta)^n\right) \tag{7}$$

We show that this n-th norm, for $n \geqslant 3$, is small. Roughly speaking, Lemma 27 upper bounds the right-hand side expression as follows:

$$\text{second summand} \leqslant \left(2\lambda^{1/2+\tau}\right)^n \cdot \lambda^{-n+1} = (2\lambda^{\tau})^n \cdot \lambda^{-n/2+1}. \tag{8}$$

which is $1/\mathsf{poly}(\lambda)$ for $n \geqslant 3$; here, the exponent of the polynomial depends on n. *This result requires a tight estimate of $B(x)$ such that $\mathrm{wt}^{\vee}(x) = 1$, a technical contribution of our work.*

Substituting $\tau = \frac{\log\left(\frac{n}{4}\log\lambda\right)}{2\log\lambda}$, we get that $2\lambda^{\tau} = (n\log\lambda)^{1/2}$. Henceforth, we will carry this upper bound as $(n\log\lambda)^{n/2} \cdot \lambda^{-n/2+1}$.

▶ Remark 5. Determining the estimates $B(x)$ is equivalent to estimating the evaluations of Krawtchouk polynomials. We require concrete estimates, not asymptotics such as the one in [12]. Elementary estimates suffice for our applications; tighter estimates, for example, those in [28, Section 3], do not yield any qualitative improvement of our results. We present upper bounds on such estimates in the appendix of the full version.

▶ Remark 6 (A world without estimating the higher order spectral norm). Suppose we upper bound the second summation using prior techniques as used in [4, 5, 17, 32, 34]; they did not analyze the ℓ_n norm. They upper bound the second summand using the ℓ_∞ norm and the ℓ_2 norm of the Fourier coefficients. Using our tight estimate of $B(x) = \lambda^{-1}$, when $\mathrm{wt}^{\vee}(x) = 1$, the old strategy will yield an upper bound of $\mathsf{poly}(\log\lambda) \cdot \lambda^{-(n-2)-(1/2)+n/2}$, which will be $o(1)$ only for $n \geqslant 4$. Our upper bound is tighter; it is $o(1)$ for $n \geqslant 3$. Detailed discussions are available in the appendix of the full version.

First summand estimation. To summarize our derivation so far, for $n \geqslant 3$ parties, we have

$$2 \cdot \mathrm{SD}\left(\overrightarrow{\mathrm{wt}}(s)\ ,\ \overrightarrow{\mathrm{wt}}(U_F)\right) \leqslant \underbrace{\sum_{\vec{w}\in\mathsf{Typical}(n,\tau)} \sum_{\zeta\in\mathcal{S}_{\vec{\beta}}} \prod_{i=1}^{n} \left|\widehat{\mathbb{1}}_{w_i}(\beta_i\zeta)\right|}_{\text{first summand}} + \underbrace{\left((n\log\lambda)^{n/2} + 4n/\lambda\right) \cdot \lambda^{-n/2+1}}_{\text{small}}. \tag{9}$$

Note that to upper bound the first summand it suffices to show that the following quantity is small *for every* $\zeta \in \mathcal{S}_{\vec{\beta}}$:

$$\sum_{\vec{w}\in\{0,1,\ldots,\lambda\}^n} \prod_{i=1}^{n} \left|\widehat{\mathbb{1}}_{w_i}(\beta_i\zeta)\right|.$$

Because, if the above quantity is small, then so is the summation restricted to $\vec{w} \in \mathsf{Typical}(n,\tau)$. And, in turn, the expression in the first summand is small too, with an additional multiplicative factor of at most n (since $\mathrm{card}(\mathcal{S}_{\vec{\beta}}) \leqslant n$).

To begin, for any $\zeta \in \mathcal{S}_{\vec{\beta}}$, rewrite

$$\sum_{\vec{w}\in\{0,1,\ldots,\lambda\}^n} \prod_{i=1}^{n} \left|\widehat{\mathbb{1}}_{w_i}(\beta_i\zeta)\right| = \prod_{i=1}^{n} \left(\sum_{w_i\in\{0,1,\ldots,\lambda\}} \left|\widehat{\mathbb{1}}_{w_i}(\beta_i\zeta)\right|\right) \tag{10}$$

Let $H(\zeta;\vec{\beta}) := \{i\colon \beta_i\zeta \neq \zeta^*\}$. Consider an index $i \notin H(\zeta;\vec{\beta})$. For such an i, we have $\beta_i\zeta = \zeta^*$ and, consequently, $\left|\widehat{\mathbb{1}}_{w_i}(\beta_i\zeta)\right| = \left|\widehat{\mathbb{1}}_{w_i}(\zeta^*)\right| = \left|\widehat{\mathbb{1}}_{w_i}(0)\right| = \binom{\lambda}{w_i}\cdot 2^{-\lambda}$, and therefore, $\sum_{w_i}\left|\widehat{\mathbb{1}}_{w_i}(\beta_i\zeta)\right| = \sum_{w_i}\binom{\lambda}{w_i}\cdot 2^{-\lambda} = 1$. As a result, the expression in Equation 10 is

$$\text{(Equation 10)} = \prod_{i\in H(\zeta;\vec{\beta})}\left(\sum_{w_i\in\{0,1,\ldots,\lambda\}}\left|\widehat{\mathbb{1}}_{w_i}(\beta_i\zeta)\right|\right)$$

If all elements in $\vec{\beta}$ are identical, then, observe that $\mathcal{S}_{\vec{\beta}}$ is a singleton set and $H(\zeta;\vec{\beta}) = \emptyset$ – a lost case for us; the expression above is 1. On the other hand, if *it is not the case that all elements in* $\vec{\beta}$ *are identical*, then $H(\zeta;\vec{\beta}) \neq \emptyset$ for every $\zeta \in \mathcal{S}_{\vec{\beta}}$. For any $i \in H(\zeta;\vec{\beta})$, it will suffice to prove that

$$\sum_{w_i\in\{0,1,\ldots,\lambda\}}\left|\widehat{\mathbb{1}}_{w_i}(\beta_i\zeta)\right| = \mathcal{O}\left(\lambda^{-1/4}\right).$$

Define $\zeta' := \beta_i\zeta$. Observe that $\zeta' \in F\setminus\{0,\zeta^*\}$. A standard application of the Parseval's identity (Fact 22) yields the estimate (see Lemma 27 for details):

$$\left|\widehat{\mathbb{1}}_{w_i}(\zeta')\right| \leqslant \sqrt{\binom{\lambda}{w_i}\cdot 2^{-\lambda}\cdot\binom{\lambda}{w'}^{-1}},$$

where $w' = \mathrm{wt}^{\vee}(\zeta')$ and $w' \in \{1,2,\ldots,\lambda-1\}$. Therefore, using the fact $\binom{\lambda}{w'} \geqslant \binom{\lambda}{1} = \lambda$, we have

$$\left|\widehat{\mathbb{1}}_{w_i}(\zeta')\right| \leqslant \sqrt{\binom{\lambda}{w_i}\cdot 2^{-\lambda}\cdot\lambda^{-1}},$$

Using this upper bound on the Fourier coefficient, we have

$$\begin{aligned}\sum_{w_i\in\{0,1,\ldots,\lambda\}}\left|\widehat{\mathbb{1}}_{w_i}(\zeta')\right| &\leqslant 2^{-\lambda/2}\lambda^{-1/2}\sum_{w_i\in\{0,1,\ldots,\lambda\}}\binom{\lambda}{w_i}^{1/2}\\ &< 2^{-\lambda/2}\lambda^{-1/2}\ \cdot\ \pi 2^{\lambda/2}\lambda^{1/4} = \pi\lambda^{-1/4} \qquad \text{(by Claim 29)}\end{aligned}$$

which is what we set out to prove.

▶ Remark 7 (Perspective). Suppose $\vec{\beta}$ has $k \geqslant 2$ distinct elements $\beta_1,\beta_2,\ldots,\beta_k$ and they occur $n_1,n_2,\ldots,n_k$ times in $\vec{\beta}$, respectively. Without loss of generality, assume that $1 \leqslant n_1 \leqslant n_2 \leqslant \cdots \leqslant n_k$. Note that $n_1+n_2+\cdots+n_k = n$ and $\mathcal{S}_{\vec{\beta}} = \left\{\zeta^*\beta_1^{-1},\ldots,\zeta^*\beta_k^{-1}\right\}$, a set of k elements. Furthermore, the set $H(\zeta^*\beta_i^{-1};\vec{\beta})$ has cardinality $(n-n_i)$, for $i \in \{1,2,\ldots,k\}$. Therefore, our upper bound calculated above will be

$$\sum_{i=1}^{k}\left(\pi\cdot\lambda^{-1/4}\right)^{n-n_i}.$$

And, this upper bound is largest (a.k.a., the weakest) when $n_1 = \cdots = n_{k-1} = 1$ and $n_k = (n-k+1)$. For this case, the upper bound becomes

$$\underbrace{\left(\pi\cdot\lambda^{-1/4}\right)^{(k-1)}}_{\text{dominant term}} + (k-1)\left(\pi\cdot\lambda^{-1/4}\right)^{n-1}$$

Overall, the worst upper bound happens where $\vec{\beta}$ has $k=2$ distinct elements, one of them occurring once, and the other occurring $(n-1)$ times. Even in this worst case, the upper bound is $\mathcal{O}\left(\lambda^{-1/4}\right)$.

Putting things together and concluding remarks. Substituting these estimates of the two summands, we get

$$2 \cdot \mathrm{SD}\big(\overrightarrow{\mathrm{wt}}(s)\ ,\ \overrightarrow{\mathrm{wt}}(U_F)\big) \leqslant \pi\lambda^{-1/4} + \underbrace{\Big(\ (n\log\lambda)^{n/2} + 4n/\lambda\ \Big) \cdot \lambda^{-n/2+1}}_{\text{small}}$$

The "small" quantity in the insecurity upper-bound is (roughly) $\lambda^{-n/2+1}$, which is o(1), for $n \geqslant 3$. This wraps up the proof overview of Theorem 4.

Which schemes are most secure?

Theorem 4 shows that $\mathsf{Gen}(\mathsf{Add}_n, \vec{\beta})$ is secure against Hamming weight leakage as long as the reconstruction multiplier $\vec{\beta}$ is not a constant vector; in other words, the only instance where the generalized additive secret sharing becomes insecure is when it is equivalent to the additive secret sharing. Does this mean all choices of $\vec{\beta}$ are equally secure, or can some choices offer even stronger security? In other words:

Which $\vec{\beta}$ provides $\mathsf{Gen}(\mathsf{Add}_n, \vec{\beta})$ the strongest security against Hamming weight leakage?

We identify the *score function* Score that quantifies the security of the $\mathsf{Gen}(\mathsf{Add}_n, \vec{\beta})$. This function captures how the choice of $\vec{\beta}$ influences the insecurity, as formalized in the following theorem.

▶ **Theorem 8.** *For $\vec{\beta} \in (F^*)^n$, the $\mathsf{Gen}(\mathsf{Add}_n, \vec{\beta})$ secret sharing scheme is ε insecure against the Hamming weight leakage, where*

$$\varepsilon = \mathcal{O}(n\log\lambda)^{n/2} \cdot \exp\left(-\min_{\zeta \in \mathcal{S}_{\vec{\beta}}} \mathsf{Score}(\zeta; \vec{\beta})\right) + \mathcal{O}(n\log\lambda)^{n/2} \cdot \lambda^{-n/2+1}.$$

In particular, if $\vec{\beta} \in (F^*)^n$ satisfies the following for $\frac{1}{2} - \frac{1}{n} > c > 0$,

$$\min_{\zeta \in \mathcal{S}_{\vec{\beta}}} \mathsf{Score}(\zeta; \vec{\beta}) \geqslant \mathcal{O}(n\log\log\lambda) + cn\log\lambda \tag{11}$$

then $\mathsf{Gen}(\mathsf{Add}_n, \vec{\beta})$ is $\mathcal{O}(\lambda^{-cn})$-insecure against the Hamming weight leakage.

Choosing $\vec{\beta}$ more carefully, by maximizing the minimum score, can lead to a faster reduction in insecurity (as shown in Corollary 9). We now present a different strategy to upper bound the first summand. Below is a brief overview of the proof of this theorem.

First summand estimation using an alternative approach. We upper bound the first summand in Equation 9 as follows:

$$\text{first summand} \leqslant \left(2\lambda^{1/2+\tau}\right)^n \cdot n \cdot \max_{\substack{\vec{w} \in \mathsf{Typical}(n,\tau) \\ \zeta \in \mathcal{S}_{\vec{\beta}}}} \left(\prod_{i=1}^{n} \left|\widehat{\mathbb{1}}_{w_i}(\beta_i\zeta)\right|\right)$$

The key to upper bounding the first summand is to understand how the monomial $\prod_{i=1}^{n}\left|\widehat{\mathbb{1}}_{w_i}(\beta_i\zeta)\right|$ can become large. Recall that $\widehat{\mathbb{1}}_{w_i}(\zeta^*)$ has a large magnitude. If $\vec{\beta}$ is such that every $\zeta\beta_i$ in the monomial simultaneously becomes ζ^* then the monomial has large magnitude, and we cannot prove the security of the secret-sharing scheme. This is exactly what happens when $\vec{\beta} = \vec{1}$; or, more generally, when all elements in $\vec{\beta}$ are identical.

For $\zeta \in \mathcal{S}_{\vec{\beta}}$, we will assign a score $\mathsf{Score}(\zeta; \vec{\beta})$ such that

$$\prod_{i=1}^{n} \left| \widehat{\mathbb{1}}_{w_i}(\beta_i \zeta) \right| \leqslant \lambda^{-n/2} \cdot \exp\Big(- \mathsf{Score}(\zeta; \vec{\beta}) \Big)$$

Equation 3 presents the definition of our score function and Lemma 30 proves that it satisfies the equation above. This definition does not depend on w_i, only on the dual Hamming weights $\mathrm{wt}^\vee(\zeta\beta_i)$, where $i \in \{1, 2, \dots n\}$. As a consequence, we get the following upper bound for the first summand, upon substituting our value of τ:

$$\text{first summand} \leqslant (n \log \lambda)^{n/2} n \cdot \exp\left(- \min_{\zeta \in \mathcal{S}_{\vec{\beta}}} \mathsf{Score}(\zeta; \vec{\beta}) \right).$$

If the minimum score for $\zeta \in \mathcal{S}_{\vec{\beta}}$ is $\geqslant c \cdot n \log \lambda$, for some $1/2 \geqslant c > 0$, then the scheme will have insecurity $\leqslant \lambda^{-cn}$. When the elements of $\vec{\beta}$ are all identical, then its minimum score is 0; in which case, the first summand is not small. Substituting this bound into Equation 9, we get our desired result.

Theorem 8 then yields a way to choose reconstruction multipliers $\vec{\beta} \in (F^*)^n$ ensuring the security of $\mathsf{Gen}(\mathsf{Add}_n, \vec{\beta})$.

▶ **Corollary 9.** *Let $n \geqslant 3$ and $\vec{\beta} = (\beta_1, \dots, \beta_n) \in (F^*)^n$ where for every β_i, there exists some $\beta_j \neq \beta_i$ such that*

$$4cn \leqslant \mathrm{wt}^\vee(\beta_j \zeta^* \beta_i^{-1}) \leqslant \lambda - 4cn \tag{12}$$

for $c \in (5/36, 1/2 - 1/n)$. Then $\mathsf{Gen}(\mathsf{Add}_n, \vec{\beta})$ is λ^{-cn}-insecure against Hamming weight leakage as long as λ is sufficiently large.

Over arbitrary fields

We show that $\mathsf{Gen}(\mathsf{Add}_n, \vec{\beta})$ inherits its security from the security of Add_n, in the sense that $\mathsf{Gen}(\mathsf{Add}_n, \vec{\beta})$ is no less secure than the additive secret sharing Add_n over *any finite field*. We make this connection precise via a *diagonalization* argument that relates the insecurity bound of $\mathsf{Gen}(\mathsf{Add}_n, \vec{\beta})$ directly to the quantity that is closely related to the insecurity of Add_n.

▶ **Theorem 10** (Diagonalization)**.** *Let F_q be a finite field of order q. Let $\vec{\beta} = (\beta_1, \dots, \beta_n) \in (F_q^*)^n$ be multipliers and $\vec{\tau} = (\tau_1, \dots, \tau_n)$ be the leakage. Define*

$$\mu_i := \sum_{\zeta \in F_q^*} \left(\sum_{\ell_i \in \Omega_i} \left| \widehat{\mathbb{1}}_{\tau_i^{-1}(\ell_i)}(\zeta) \right| \right)^n.$$

Then,

$$\varepsilon(\mathsf{Gen}(\mathsf{Add}_n, \vec{\beta})) \leqslant \prod_{i=1}^{n} \mu_i^{1/n} \leqslant \frac{1}{n} \sum_{i=1}^{n} \mu_i \leqslant \max(\mu_1, \dots, \mu_n)$$

▶ Remark 11. Combining Theorem 10 with the result of Faust et al. [17], we immediately obtain that $\mathsf{Gen}(\mathsf{Add}_n, \vec{\beta})$ over F_p, where $p = 2^\lambda - 1$ is a Mersenne prime, is $\mathcal{O}\big(\lambda^{-n/4}\big)$-insecure against Hamming weight leakage whenever $n \geqslant 5$, for all $\vec{\beta} \in (F_p^*)^n$.

We defer the proof of Theorem 10 and further discussion to the full version.

2 Preliminaries

Basic Notations. Let F_q be a finite field of order (cardinality) q and λ be an integer such that $2^{\lambda-1} < q \leqslant 2^{\lambda}$. Unless the order needs to be specified, we will simply write F to denote the finite field. Let $F^* := F \setminus \{0\}$ denote the multiplicative group of F. We use U_F to denote the uniform distribution over F, and we write $x \leftarrow U_F$ when sampling $x \in F$ uniformly at random.

Given a set S, we denote by $\mathbb{1}_S$ the indicator function for set S, which gives $\mathbb{1}_S(x) = 1$ if $x \in S$ and 0 otherwise. For any function $f\colon D \to R$ where D is the domain and R is the range, we write $f^{-1}(y) := \{x \in D\colon f(x) = y\}$ for the set of all preimages of $y \in R$. For instance, $x \in F$ has $f(x) = w$ if and only if $\mathbb{1}_{f^{-1}(w)}(x) = 1$.

2.1 Secret Sharing (Masking) Schemes

▶ **Definition 12** (Additive Secret Sharing). *Let F be a finite field and n a positive integer. Given a secret $s \in F$, additive secret sharing Add_n shares s by sampling $(s_1, \ldots, s_n) \in F^n$ uniformly at random conditioned on $s_1 + \cdots + s_n = s$. More concretely, Add_n consists of the following two algorithms: given $s \in F$,*

- $\mathsf{Share}(s)$*: Sample $s_1, \ldots, s_{n-1} \leftarrow U_F$, set $s_n = s - (s_1 + \cdots + s_{n-1})$, and return $(s_1, \ldots, s_n)$.*
- $\mathsf{Reconstruct}(\{s_1, \ldots, s_n\})$*: Return $\tilde{s} = s_1 + \cdots + s_n$.*

We denote by $\mathsf{Add}_n(s)$ the resulting distribution of shares $(s_1, \ldots, s_n)$.

▶ **Definition 13** (Generalized Additive Secret Sharing). *Let F be a finite field and n a positive integer. Given reconstruction multipliers $\vec{\beta} = (\beta_1, \ldots, \beta_n) \in (F^*)^n$ and a secret $s \in F$, the generalized additive secret sharing $\mathsf{Gen}(\mathsf{Add}_n, \vec{\beta})$ shares s by sampling $(s_1, \ldots, s_n) \in F^n$ uniformly at random conditioned on $\beta_1 s_1 + \cdots + \beta_n s_n = s$. More concretely, $\mathsf{Gen}(\mathsf{Add}_n, \vec{\beta})$ consists of the following two algorithms:*

- $\mathsf{Share}(s)$*: Sample $s_1, \ldots, s_{n-1} \leftarrow U_F$, set $s_n = \beta_n^{-1}(s - (\beta_1 s_1 + \cdots + \beta_{n-1} s_{n-1}))$, then return $(s_1, \ldots, s_n)$.*
- $\mathsf{Reconstruct}(\{s_1, \ldots, s_n\})$*: Return $\tilde{s} = \beta_1 s_1 + \cdots + \beta_n s_n$.*

We denote by $\mathsf{Gen}(\mathsf{Add}_n, \vec{\beta}; s)$ the resulting distribution of shares $(s_1, \ldots, s_n)$.

Note that, by definition, $\mathsf{Gen}(\mathsf{Add}_n, \vec{\beta})$ sharing $s \in F$ with $\vec{\beta} = (c, \ldots, c) \in (F^*)^n$ where $c \in F^*$ constant is equivalent to Add_n sharing $c^{-1}s$.

The term *generalized* secret sharing was chosen in line with notational conventions in coding theory literature. Observe that, $\mathsf{Gen}(\mathsf{Add}_n, \vec{\beta})$ can be seen as Add_n generating the shares $(s'_1, \ldots, s'_n)$ such that $s'_1 + \cdots + s'_n = s$, then postprocessing them into $(\beta_1^{-1}s'_1, \ldots, \beta_n^{-1}s'_n)$. Interpreting the shares as codewords of a linear code [35], this postprocessing corresponds to the notion of generalized linear codes; see, for example, [22, Chapter 5].

▶ **Definition 14** (Shamir Secret Sharing). *Let F be a finite field and n and k be positive integers such that $k \leqslant n$. Denote by $F[X]_{<k}$ the set of polynomials over F of degree less than k [11, Definition 11.6]. Given evaluation places $\vec{X} = (X_1, \ldots, X_n) \in (F^*)^n$ with $X_i \neq X_j$ for all $i \neq j$ and a secret $s \in F$, Shamir secret sharing $\mathsf{Shamir}(n, k, \vec{X})$ shares s by:*

- $\mathsf{Share}(s)$*: Sample a polynomial $P(X) \leftarrow F[X]_{<k}$ uniformly at random conditioned on $P(0) = s$. Evaluate $P(X)$ at each value in $\vec{X}$, i.e. $s_i = P(X_i)$ for $i = 1, \ldots, n$. Return $(s_1, \ldots, s_n)$.*
- $\mathsf{Reconstruct}(\{s_{i_1}, \ldots, s_{i_k}\})$*: Interpolate the polynomial $\tilde{P}(X) \in F[X]_{<k}$ over points $(X_{i_1}, s_{i_1}), \ldots, (X_{i_k}, s_{i_k})$. Return $\tilde{s} := \tilde{P}(0)$.*

We denote by $\mathsf{Shamir}(n, k, \vec{X}; s)$ the resulting distribution of shares $(s_1, \ldots, s_n)$.

▶ **Proposition 15.** *Shamir secret sharing with $k = n$ is an instance of generalized additive secret sharing. Therefore, proving security of* $\mathsf{Gen}(\mathsf{Add}_n, \vec{\beta})$ *automatically translates to the security of* $\mathsf{Shamir}(n, n, \vec{X})$.

The proof of Proposition 15 is provided in the full version.

2.2 Leakage-Resilient Secret Sharing

We quantify the variation between two distributions P and Q using the *statistical distance*.

▶ **Definition 16** (Statistical Distance). *The statistical distance between two distributions P and Q over a finite space Ω is defined as*

$$\mathrm{SD}(P\ ,\ Q) := \frac{1}{2}\sum_{x\in\Omega}\Big|\Pr[P = x] - \Pr[Q = x]\Big|.$$

▶ **Definition 17** (Local Leakage [4, 5]). *Let F be a finite field and n be a positive integer. Let $\mathcal{S}$ be a secret sharing scheme with sharing algorithm* Share. *Let $\vec{\tau} = (\tau_1, \ldots, \tau_n)$ be a collection of n functions whose domains are F. For any secret $s \in F$, the leakage distribution $\vec{\tau}(\mathsf{Share}(s))$ is defined by the following experiment:*
1. *Sample shares $\vec{s} = (s_1, \ldots, s_n)$ from* $\mathsf{Share}(s)$.
2. *Output $(\tau_1(s_1), \ldots, \tau_n(s_n))$.*

We refer to $\vec{\tau}(\mathsf{Share}(s))$ as the joint leakage distribution induced by all shares of $s \in F$, and for brevity we shall also write it as $\vec{\tau}(s)$ instead.

▶ **Definition 18** (ε-insecurity). *A secret sharing scheme $\mathcal{S}$ is said to be ε-insecure against a local leakage $\vec{\tau}$ if for all secret $s \in F$,*

$$\varepsilon(\mathcal{S}; \vec{\tau}) := 2 \cdot \mathrm{SD}(\vec{\tau}(\mathsf{Share}(s))\ ,\ \vec{\tau}(\mathsf{Share}(U_F))) \leqslant \varepsilon$$

Roughly speaking, if $\mathcal{S}$ has small insecurity against a leakage attack, then the joint distribution of leakage of shares is statistically independent of the secret. In this paper, we primarily focus on the setting where all leakage functions are the Hamming weight, i.e. $\tau_1 = \cdots = \tau_n = \mathrm{wt}$. And in this case, $\vec{\tau}(\vec{s}) = \overrightarrow{\mathrm{wt}}(\vec{s})$ corresponds to the joint Hamming weight leakage of the shares of s.

▶ **Definition 19** (Typical Weights). *We call $w \in \{0, 1, \ldots, \lambda\}$ to be typical if $|w - \lambda/2| \leqslant \lambda^{1/2+\tau}$ for some $\tau > 0$. And we say a vector $\vec{w} = (w_1, w_2, \ldots, w_n) \in \{0, 1, \ldots, \lambda\}^n$ is in a typical set* $\mathsf{Typical}(n, \tau)$ *if every w_i is typical:*

$$\mathsf{Typical}(n, \tau) := \Big\{\vec{w}\ :\ |w_i - \lambda/2| \leqslant \lambda^{1/2+\tau} \quad \forall i \in \{1, 2, \ldots, n\}\Big\}.$$

2.3 Extension Fields and Hamming Weight

The field F_{2^λ} is isomorphic to the vector space F_2^λ upon choosing a basis $\mathcal{B} = (b_1, \ldots, b_\lambda)$. An element $\zeta \in F_{2^\lambda}$ is represented as a vector $(\zeta_1, \ldots, \zeta_\lambda) \in F_2^\lambda$ satisfying $\zeta = \sum_{j=1}^{\lambda} \zeta_j b_j$. We refer to $(\zeta_1, \ldots, \zeta_\lambda) \in F_2^\lambda$ as the *vector representation* of $\zeta \in F_{2^\lambda}$ under the basis $\mathcal{B}$, and the *Hamming weight* $\mathrm{wt}_\mathcal{B}(\zeta)$ of $\zeta \in F_{2^\lambda}$ is the number of 1's this vector representation. For brevity, we write $\mathrm{wt}(\zeta) = \mathrm{wt}_\mathcal{B}(\zeta)$ unless the basis $\mathcal{B}$ must be specified. We also write $\mathbb{1}_w := \mathbb{1}_{\mathrm{wt}^{-1}(w)}$, i.e.

$$\mathbb{1}_w(x) := \begin{cases} 1, & \text{if } \mathrm{wt}(x) = w \\ 0, & \text{otherwise} \end{cases}$$

▶ **Definition 20** (Field Trace). *The trace of an extension field F_{p^d} over the base field F_p, denoted by $\mathsf{Tr}_{F_{p^d}/F_p}\colon F_{p^d} \to F_p$, is defined as*

$$\mathsf{Tr}_{F_{p^d}/F_p}(y) := \sum_{i=0}^{d-1} y^{p^i}$$

The *trace-dual coordinate vector* $\zeta^\vee$ of $\zeta \in F_{2^\lambda}$ is its vector representation under the basis $\mathcal{B}^\vee$ dual to $\mathcal{B} = (b_1, \ldots, b_\lambda)$, and can be computed as

$$\zeta^\vee := \left(\mathsf{Tr}_{F_{2^\lambda}/F_2}(\zeta b_1), \ldots, \mathsf{Tr}_{F_{2^\lambda}/F_2}(\zeta b_\lambda)\right) \in F_2^\lambda$$

because if we write $\mathcal{B}^\vee = (b_1^\vee, \ldots, b_\lambda^\vee)$, then

$$\mathsf{Tr}_{F_{2^\lambda}/F_2}(b_i b_j^\vee) = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{otherwise} \end{cases}$$

We hence also have $\mathsf{Tr}_{F_{2^\lambda}/F_2}(\zeta x) = \langle \zeta^\vee, x\rangle$ for all $\zeta, x \in F_{2^\lambda}$ (the x in the inner product $\langle \alpha^\vee, x\rangle$ is the vector representation of x). The *trace-dual Hamming weight* of ζ is then the Hamming weight of $\zeta^\vee$, i.e. $\mathrm{wt}^\vee(\zeta) = \mathrm{wt}(\zeta^\vee)$. The map $\zeta \mapsto \zeta^\vee$ is an invertible F_2-linear map, and also efficiently computable [42].

2.4 Fourier Analysis on Finite Fields

We will use Fourier analysis on the additive group $(F_{p^d}, +)$ for a prime $p \geqslant 2$ and degree of extension $d \in \{1, 2, \ldots\}$. In this subsection, $F := F_{p^d}$.

Let $\omega_p := \exp(2\pi\imath/p)$. Define the Fourier transformation of $f\colon F \to \mathbb{C}$ over F, denoted $\widehat{f}\colon F \to \mathbb{C}$, as follows:

$$\widehat{f}(\alpha) := \frac{1}{q}\sum_{x\in F} f(x)\cdot \omega_p^{-\mathsf{Tr}_{F/F_p}(\alpha x)} \qquad \forall \alpha \in F$$

We call $\chi_\alpha(x) = \omega_p^{\mathsf{Tr}_{F/F_p}(\alpha x)}$ the *character* and $\widehat{f}(\alpha)$ the *Fourier coefficient* of f at α.

▶ **Example 21.** For $p = 2$, we have $\omega := \omega_2 = \exp(\pi\imath) = -1$ and Fourier coefficients are

$$\widehat{f}(\alpha) := \frac{1}{q}\sum_{x\in F_{2^d}} f(x)\cdot(-1)^{\mathsf{Tr}_{F/F_2}(\alpha x)}.$$

Since $\mathsf{Tr}_{F/F_2}(\alpha x) = \langle \alpha^\vee, x\rangle$ as mentioned in Section 2.3, $\widehat{\mathbb{1}}_w(\alpha)$ is a function of $\mathrm{wt}^\vee(\alpha)$; in particular, for all $\alpha, \alpha' \in F_{2^d}$, if $\mathrm{wt}^\vee(\alpha) = \mathrm{wt}^\vee(\alpha')$ then $\widehat{\mathbb{1}}_w(\alpha) = \widehat{\mathbb{1}}_w(\alpha')$.

▶ **Fact 22** (Parseval's Identity). *For any function $f\colon F \to \mathbb{C}$,*

$$\frac{1}{\mathrm{card}(F)}\sum_{x\in F}|f(x)|^2 = \sum_{\alpha\in F}\left|\widehat{f}(\alpha)\right|^2.$$

▶ **Fact 23** (Character Sum). *For all $\alpha \in F$,*

$$\sum_{x\in F}\chi_\alpha(x) = \begin{cases} \mathrm{card}(F) & \text{if } \alpha = 0 \\ 0 & \text{otherwise} \end{cases}$$

Note also that the modulus $|\chi_\alpha(x)| = 1$ for any α and x in F.

2.5 Imported Lemmas

▶ **Lemma 24** (Chernoff Bound [36, Theorem 2.1]). *Let $X_1, X_2, \ldots, X_\lambda$ be independent Bernoulli random variables with $\mathbb{E}(X_i) = p$ for each i. Then for any $t \geqslant 0$,*

$$\Pr\left(\left|\sum_{i=1}^{\lambda} X_i - \lambda p\right| \geqslant \lambda t\right) \leqslant 2\exp(-2\lambda t^2).$$

In particular, for $t = \lambda^{-\frac{1}{2}+\tau}$, the upper bound is $2\exp(-2\lambda^{2\tau})$.

▶ **Lemma 25** (Poisson Summation Formula [38, Chapter 3.3]). *Let $C \subseteq F^n$ denote the set of all shares of the secret $0 \in F = F_{p^d}$. Let $\vec{v} \in F^n$ denote an arbitrary secret share of the secret $1 \in F$. Consider an arbitrary local leakage $\vec{\tau}\colon F^n \to \Omega^n$. For any secret $s \in F$, let $\vec{\tau}(s)$ denote the distribution of the leakage $\vec{\tau}(\vec{x})$, where $\vec{x}$ is sampled uniformly at random from the set $s \cdot \vec{v} + C$. The following identity holds for any leakage value $\vec{\ell} \in \Omega^n$.*

$$\Pr\left[\vec{\tau}(s) = \vec{\ell}\right] = \sum_{\vec{z} \in C^\perp} \left(\prod_{i=1}^{n} \widehat{\mathbb{1}}_{\tau_i^{-1}(\ell_i)}(z_i)\right) \cdot \chi_1(s \cdot \langle \vec{z}, \vec{v} \rangle).$$

where $\chi_1(x) = \omega_p^{\mathrm{Tr}_{F/F_p}(x)}$, whose modulus is $|\chi_1(x)| = 1$ for any $x \in F$.

▶ **Lemma 26** (Generalized Hölder's Inequality [23, Theorem 11]). *Let $p_1, \ldots, p_n$ be positive real numbers such that $p_1 + \cdots + p_n = 1$, and $(x_i^{(1)})_{i=1}^k$, $(x_i^{(2)})_{i=1}^k$, $\ldots$, $(x_i^{(n)})_{i=1}^k$ be sequences of real numbers. Then,*

$$\sum_{i=1}^{k} \left|x_i^{(1)}\right|^{p_1} \cdots \left|x_i^{(n)}\right|^{p_n} \leqslant \left(\sum_{i=1}^{k} \left|x_i^{(1)}\right|\right)^{p_1} \cdots \left(\sum_{i=1}^{k} \left|x_i^{(n)}\right|\right)^{p_n}$$

3 Insecurity Analysis

In this section, we formally state the technical results introduced in Section 1.2 and their proofs outlined in the overview (Section 1.3). For succinctness, we omit the calculations in the proofs of both theorems; see the full version for complete derivations of each equation.

3.1 Worst-case Analysis (Theorem 4)

▶ **Theorem 4.** *For $\vec{\beta} \in (F^*)^n \setminus \{(b, \ldots, b)\colon b \in F^*\}$ and $n \in \{3, 4, \ldots\}$, we have*

$$\varepsilon\Big(\ \mathsf{Gen}(\mathsf{Add}_n, \vec{\beta})\ ;\ \overrightarrow{\mathrm{wt}}\ \Big) = \mathcal{O}\Big(\lambda^{-1/4}\Big).$$

The proof of Theorem 4 relies on the following lemma that upper bounds our Fourier coefficients; we present its proof in the appendix of the full version.

▶ **Lemma 27.** *For $\lambda \geqslant 2$, $w \in \{1, \ldots, \lambda\}$, and $\zeta \in F$, we have $|\widehat{\mathbb{1}}_w(\zeta)| \leqslant B(\zeta)$ where*

$$B(\zeta) := \begin{cases} \lambda^{-1/2}, & \text{if } \mathrm{wt}^\vee(\zeta) \in \{0, \lambda\} \\ \lambda^{-1}, & \text{if } \mathrm{wt}^\vee(\zeta) \in \{1, \lambda - 1\} \\ 4 \cdot \lambda^{-3/2}, & \text{if } \mathrm{wt}^\vee(\zeta) \in \{2, \lambda - 2\} \\ \lambda^{-\frac{1}{4}} \binom{\lambda}{\mathrm{wt}^\vee(\zeta)}^{-\frac{1}{2}}, & \text{otherwise.} \end{cases}$$

And hence, $\sum_{\mathrm{wt}^\vee(\zeta)=1}^{\lambda-1} (B(\zeta))^n \leqslant 3 \cdot 5^{\frac{5n}{2}-4} \cdot \lambda^{-n+1}$

Proof of Theorem 4. By Definition 18, it suffices to prove that $2 \cdot \mathrm{SD}\big(\overrightarrow{\mathrm{wt}}(s)\ ,\ \overrightarrow{\mathrm{wt}}(U_F)\big)$ is upper bounded by ε. Recall from Definition 19 that a weight $w \in \{0, 1, \ldots, \lambda\}$ is said to be typical if it is only up to $\lambda^{1/2+\tau}$ away from the average $\lambda/2$ for some $\tau > 0$. Chernoff bound (Lemma 24) implies that the probability mass on atypical weights contribute only negligibly to the statistical distance. Hence, the expression for the statistical distance (Definition 18) becomes a summation over the set of typical weights $\mathsf{Typical}(n, \tau)$ plus a negligible quantity.

$$\begin{aligned} &2 \cdot \mathrm{SD}\big(\overrightarrow{\mathrm{wt}}(s)\ ,\ \overrightarrow{\mathrm{wt}}(U_F)\big) \\ &\quad = \sum_{\vec{w} \in \mathsf{Typical}(n,\tau)} \left| \Pr_{\vec{s} \leftarrow \mathsf{Gen}(\mathsf{Add}_n, \vec{\beta}; s)} \left[\overrightarrow{\mathrm{wt}}(\vec{s}) = \vec{w}\right] - \Pr_{\vec{s} \leftarrow \mathsf{Gen}(\mathsf{Add}_n, \vec{\beta}; U_F)} \left[\overrightarrow{\mathrm{wt}}(\vec{s}) = \vec{w}\right] \right| \qquad (13) \\ &\qquad\qquad \pm\ 2 \cdot 2n \cdot \exp(-2\lambda^{2\tau}) \end{aligned}$$

If $\vec{s} \in F^n$ is a share of secret 0, i.e. $\vec{s} \in C := \mathsf{Gen}(\mathsf{Add}_n, \vec{\beta}; 0)$, then $\beta_1 s_1 + \cdots + \beta_n s_n = 0$, and for any $\zeta \in F$, we have $\zeta\beta_1 s_1 + \cdots + \zeta\beta_n s_n = \zeta \cdot 0 = 0$. From this, we can deduce $C^\perp = \{\zeta\vec{\beta} \mid \zeta \in F\}$, then by Poisson summation formula (Lemma 25),

$$\Pr_{\vec{s} \leftarrow \mathsf{Gen}(\mathsf{Add}_n, \vec{\beta}; s)} \left[\overrightarrow{\mathrm{wt}}(\vec{s}) = \vec{w}\right] = \sum_{\zeta \in F} \left(\prod_{i=1}^{n} \widehat{\mathbb{1}}_{w_i}(\beta_i \zeta) \right) \cdot \chi_\zeta(s) \qquad (14)$$

and by Fact 23,

$$\Pr_{\vec{s} \leftarrow \mathsf{Gen}(\mathsf{Add}_n, \vec{\beta}; U_F)} \left[\overrightarrow{\mathrm{wt}}(\vec{s}) = \vec{w}\right] = \sum_{\zeta \in \{0\}} \left(\prod_{i=1}^{n} \widehat{\mathbb{1}}_{w_i}(\beta_i \zeta) \right) \qquad (15)$$

Plugging these into the summand in Equation 13 and applying triangle inequality, we get

$$\begin{aligned} &\sum_{\vec{w} \in \mathsf{Typical}(n,\tau)} \left| \Pr_{\vec{s} \leftarrow \mathsf{Gen}(\mathsf{Add}_n, \vec{\beta}; s)} \left[\overrightarrow{\mathrm{wt}}(\vec{s}) = \vec{w}\right] - \Pr_{\vec{s} \leftarrow \mathsf{Gen}(\mathsf{Add}_n, \vec{\beta}; U_F)} \left[\overrightarrow{\mathrm{wt}}(\vec{s}) = \vec{w}\right] \right| \\ &\quad \leqslant \sum_{\vec{w} \in \mathsf{Typical}(n,\tau)} \sum_{\zeta \in F^*} \prod_{i=1}^{n} \left| \widehat{\mathbb{1}}_{w_i}(\beta_i \zeta) \right| \qquad (16) \end{aligned}$$

Let $\mathcal{S}_{\vec{\beta}} := \left\{\zeta^* \beta_1^{-1}, \zeta^* \beta_2^{-1}, \ldots, \zeta^* \beta_n^{-1}\right\}$. Then for any $\zeta \in \mathcal{S}_{\vec{\beta}}$, at least one of β_i's should give $\beta_i \zeta = \zeta^*$. Consider the following separation of the summation.

(Equation 16)

$$= \sum_{\vec{w} \in \mathsf{Typical}(n,\tau)} \sum_{\zeta \in \mathcal{S}_{\vec{\beta}}} \prod_{i=1}^{n} \left| \widehat{\mathbb{1}}_{w_i}(\beta_i \zeta) \right| + \sum_{\vec{w} \in \mathsf{Typical}(n,\tau)} \sum_{\zeta \in F^* \setminus \mathcal{S}_{\vec{\beta}}} \prod_{i=1}^{n} \left| \widehat{\mathbb{1}}_{w_i}(\beta_i \zeta) \right| \qquad (17)$$

Let us upper bound the second summand (the summation over $F^* \setminus \mathcal{S}_{\vec{\beta}}$). As stated in Lemma 27, we denote $B(\zeta)$ to be a function that upper bounds $|\widehat{\mathbb{1}}_w(\zeta)|$. Then, by Hölder's inequality (Lemma 26) and Lemma 27,

$$\sum_{\vec{w} \in \mathsf{Typical}(n,\tau)} \sum_{\zeta \in F^* \setminus \mathcal{S}_{\vec{\beta}}} \prod_{i=1}^{n} \left| \widehat{\mathbb{1}}_{w_i}(\beta_i \zeta) \right| \leqslant 2^{n+2} \cdot 5^{\frac{5n}{2}-4} \cdot \lambda^{n\tau} \cdot \lambda^{-\frac{n}{2}+1} \qquad (18)$$

Let $H(\zeta; \vec{\beta}) := \{i \colon \beta_i \zeta \neq \zeta^*\}$. The first summand (the summation over $\mathcal{S}_{\vec{\beta}}$) can be rewritten as follows

$$\begin{aligned} &\sum_{\vec{w} \in \mathsf{Typical}(n,\tau)} \sum_{\zeta \in \mathcal{S}_{\vec{\beta}}} \prod_{i=1}^{n} \left| \widehat{\mathbb{1}}_{w_i}(\beta_i \zeta) \right| \\ &\quad \leqslant \sum_{\zeta \in \mathcal{S}_{\vec{\beta}}} \left(\prod_{i \in H(\zeta;\vec{\beta})} \sum_{w_i=0}^{\lambda} \left| \widehat{\mathbb{1}}_{w_i}(\beta_i \zeta) \right| \right) \cdot \left(\prod_{i \notin H(\zeta;\vec{\beta})} \sum_{w_i=0}^{\lambda} \left| \widehat{\mathbb{1}}_{w_i}(\beta_i \zeta) \right| \right) \qquad (19) \end{aligned}$$

Next, we separate bound the ℓ_1-norms on the right-hand side for those indices $i \in H(\zeta; \vec{\beta})$ and $i \notin H(\zeta; \vec{\beta})$. We use the following claims, whose proofs are provided in the full version.

▷ Claim 28. For all $\lambda \in \{1, 2, \dots\}$, $w \in \{1, 2, \dots, \lambda\}$, and $\zeta \in F$,

$$\left|\widehat{\mathbb{1}}_w(\zeta)\right| \leqslant \frac{1}{2^{\lambda/2}}\binom{\lambda}{w}^{1/2}\binom{\lambda}{\mathrm{wt}^\vee(\zeta)}^{-1/2} \leqslant \frac{1}{\lambda^{1/4}}\binom{\lambda}{\mathrm{wt}^\vee(\zeta)}^{-\frac{1}{2}} \qquad \text{if } \zeta \notin \{0, \zeta^*\}$$

$$\left|\widehat{\mathbb{1}}_w(\zeta)\right| = \frac{1}{2^\lambda}\binom{\lambda}{w} \leqslant \frac{1}{\sqrt{\lambda}} \qquad \text{if } \zeta \in \{0, \zeta^*\}$$

▷ Claim 29. For $m \in \{1, 2, \dots\}$, the following bound holds.

$$\sum_{i=0}^{m}\binom{m}{i}^{1/2} < \pi \cdot m^{1/4} \cdot 2^{m/2}.$$

Case 1. Consider $i \in H(\zeta; \vec{\beta})$. This is the non-trivial case, we want a non-trivial upper bound. By Claim 28 and Claim 29,

$$\sum_{w_i=0}^{\lambda}\left|\widehat{\mathbb{1}}_{w_i}(\beta_i\zeta)\right| < \binom{\lambda}{\mathrm{wt}^\vee(\beta_i\zeta)}^{-1/2} \cdot \pi \cdot \lambda^{1/4}$$

Case 2. Consider $i \notin H(\zeta; \vec{\beta})$. In this case, we have the trivial estimate from Claim 28.

$$\sum_{w_i=0}^{\lambda}\left|\widehat{\mathbb{1}}_{w_i}(\beta_i\zeta)\right| = \sum_{w_i=0}^{\lambda}\left|\widehat{\mathbb{1}}_{w_i}(\zeta^*)\right| = \sum_{w_i=0}^{\lambda}\widehat{\mathbb{1}}_{w_i}(0) = \sum_{w_i=0}^{\lambda}\binom{\lambda}{w_i}\cdot 2^{-\lambda} = 1$$

Substituting these values into Equation 19 and continuing the upper bound, we have:

$$(\text{Equation 19}) < \sum_{\zeta\in\mathcal{S}_{\vec{\beta}}}\prod_{i\in H(\zeta;\vec{\beta})}\binom{\lambda}{\mathrm{wt}^\vee(\beta_i\zeta)}^{-1/2} \cdot \pi \cdot \lambda^{1/4} \tag{20}$$

Note that, for any $i \in H(\zeta; \vec{\beta})$, $\mathrm{wt}^\vee(\beta_i\zeta) \in \{1, 2, \dots, \lambda - 1\}$. Therefore, using the fact that $\binom{\lambda}{\mathrm{wt}^\vee(\beta_i\zeta)} \geqslant \binom{\lambda}{1} = \lambda$ for any $i \in H(\zeta; \vec{\beta})$, we obtain the following bound for the above quantity:

$$(\text{Equation 20}) \leqslant \sum_{\zeta\in\mathcal{S}_{\vec{\beta}}}\lambda^{-\frac{\mathrm{card}(H(\zeta;\vec{\beta}))}{2}} \cdot \pi^{\mathrm{card}(H(\zeta;\vec{\beta}))} \cdot \lambda^{\frac{\mathrm{card}(H(\zeta;\vec{\beta}))}{4}} \leqslant (\widetilde{h}+1)\cdot\left(\frac{\pi^4}{\lambda}\right)^{\frac{\widetilde{h}}{4}} \tag{21}$$

because $\mathrm{card}(\mathcal{S}_{\vec{\beta}}) \leqslant \widetilde{h} + 1$, where $\widetilde{h} = \min_{\zeta\in\mathcal{S}_{\vec{\beta}}}\Big(\mathrm{card}(H(\zeta; \vec{\beta}))\Big)$. [2]

Therefore, for a sufficiently large λ, the above sum is small as long as we have $\mathrm{card}(H(\zeta; \vec{\beta})) \geqslant 1$; that is, unless $\beta_1 = \beta_2 = \cdots = \beta_n$ (i.e., unless the secret sharing is the additive secret sharing), the above sum would be at most $\lambda^{-1/4}$. More precisely, Equation 18 and 21 add up to:

$$2 \cdot \mathrm{SD}\big(\overrightarrow{\mathrm{wt}}(s)\ ,\ \overrightarrow{\mathrm{wt}}(U_F)\big) \leqslant 2\pi \cdot \frac{c_n}{\lambda^{1/4}} + 2^{n+2} \cdot 5^{\frac{5n}{2}-4} \cdot \lambda^{n\tau} \cdot \lambda^{-\frac{n}{2}+1} + 2 \cdot 2n \cdot \exp(-2\lambda^{2\tau})$$

which is $\mathcal{O}\big(\lambda^{-1/4}\big)$, for sufficiently large λ. ◀

[2] If $\zeta \in \mathcal{S}_{\vec{\beta}}$, then there should exist $j \in \{1, 2, \dots, n\}$ such that $\zeta = \beta_j^{-1}\zeta^*$, and so $i \in H(\zeta, \vec{\beta})$ iff $\beta_i \neq \beta_j$. Hence, for any $\zeta \in \mathcal{S}_{\vec{\beta}}$, $\mathrm{card}(H(\zeta; \vec{\beta})) \geqslant \mathrm{card}(\mathcal{S}_{\vec{\beta}}) - 1$, making $\widetilde{h} := \min_{\zeta\in\mathcal{S}_{\vec{\beta}}}\mathrm{card}(H(\zeta; \vec{\beta})) \geqslant \mathrm{card}(\mathcal{S}_{\vec{\beta}}) - 1$ as well.

3.2 Security Characterization (Theorem 8)

▶ **Theorem 8.** *For $\vec{\beta} \in (F^*)^n$, the $\mathsf{Gen}(\mathsf{Add}_n, \vec{\beta})$ secret sharing scheme is ε insecure against the Hamming weight leakage, where*

$$\varepsilon = \mathcal{O}(n \log \lambda)^{n/2} \cdot \exp\left(- \min_{\zeta \in \mathcal{S}_{\vec{\beta}}} \mathsf{Score}(\zeta; \vec{\beta}) \right) + \mathcal{O}(n \log \lambda)^{n/2} \cdot \lambda^{-n/2+1}.$$

In particular, if $\vec{\beta} \in (F^*)^n$ satisfies the following for $\frac{1}{2} - \frac{1}{n} > c > 0$,

$$\min_{\zeta \in \mathcal{S}_{\vec{\beta}}} \mathsf{Score}(\zeta; \vec{\beta}) \geqslant \mathcal{O}(n \log \log \lambda) + cn \log \lambda \tag{11}$$

then $\mathsf{Gen}(\mathsf{Add}_n, \vec{\beta})$ is $\mathcal{O}(\lambda^{-cn})$-insecure against the Hamming weight leakage.

For the proof of Theorem 8, we use the following lemma that relates the product of Fourier coefficients to our Score function. Its proof can be found in the full version.

▶ **Lemma 30.** *For any $\vec{\beta} = (\beta_1, \beta_2, \ldots, \beta_n) \in (F^*)^n$ and $\vec{w} = (w_1, w_2, \ldots, w_n) \in \{1, 2, \ldots, \lambda\}^n$,*

$$\prod_{i=1}^{n} \left| \widehat{\mathbb{1}}_{w_i}(\beta_i \zeta) \right| \leqslant \lambda^{-\frac{n}{2}} \cdot \exp\left(- \mathsf{Score}(\zeta; \vec{\beta}) \right)$$

Proof of Theorem 8. The proof goes in the same way as for Theorem 4, with the estimation of the second summand unchanged. The only difference lies in the first summand, which naturally gives rise to the score function quantifying the security of $\mathsf{Gen}(\mathsf{Add}_n, \vec{\beta})$.

The first summand (the summation over $\mathcal{S}_{\vec{\beta}}$) can be rewritten as follows, using Lemma 30.

$$\sum_{\vec{w} \in \mathsf{Typical}(n,\tau)} \sum_{\zeta \in \mathcal{S}_{\vec{\beta}}} \prod_{i=1}^{n} \left| \widehat{\mathbb{1}}_{w_i}(\beta_i \zeta) \right| \leqslant (2\lambda^{\tau})^n \cdot n \cdot \exp\left(- \min_{\zeta \in \mathcal{S}_{\vec{\beta}}} \mathsf{Score}(\zeta; \vec{\beta}) \right) \tag{22}$$

because the maximum of a sequence is at least as large as its average. Hence, upon choosing τ such that $2\lambda^{\tau} = (n \log \lambda)^{1/2}$, we get

(Equation 17)

$$\leqslant (n \log \lambda)^{n/2} \cdot n \cdot \exp\left(- \min_{\zeta \in \mathcal{S}_{\vec{\beta}}} \mathsf{Score}(\zeta; \vec{\beta}) \right) + 4 \cdot 5^{\frac{5n}{2}-4} \cdot (n \log \lambda)^{n/2} \cdot \lambda^{-\frac{n}{2}+1} \tag{23}$$

by Equation 18 and 22, and in summary, for sufficiently large λ,

$$2 \cdot \mathrm{SD}\left(\overrightarrow{\mathrm{wt}}(s) \, , \, \overrightarrow{\mathrm{wt}}(U_F) \right) \leqslant \mathcal{O}(n \log \lambda)^{\frac{n}{2}} \cdot \exp\left(- \min_{\zeta \in \mathcal{S}_{\vec{\beta}}} \mathsf{Score}(\zeta; \vec{\beta}) \right) + \mathcal{O}(n \log \lambda)^{\frac{n}{2}} \cdot \lambda^{-\frac{n}{2}+1}$$

Now, suppose that $\vec{\beta}$ gives

$$\min_{\zeta \in \mathcal{S}_{\vec{\beta}}} \mathsf{Score}(\zeta; \vec{\beta}) \geqslant \left(\frac{n}{2} + 1 \right) \log(n \log \lambda) + cn \log \lambda \tag{24}$$

for some constant $c > 0$. Then,

$$2 \cdot \mathrm{SD}\left(\overrightarrow{\mathrm{wt}}(s) \, , \, \overrightarrow{\mathrm{wt}}(U_F) \right) \leqslant \lambda^{-cn} + \mathcal{O}\left(\lambda^{-c'n} \right) = \mathcal{O}\left(\lambda^{-cn} \right)$$

because $\frac{\log \lambda}{\lambda} \leqslant \lambda^{-c'}$ holds for all $c' \in (0, 1/2)$. This concludes the proof of Theorem 8. ◀

Note that Theorem 8 provides an insecurity bound in terms of reconstruction multipliers $\vec{\beta} \in (F^*)^n$. This consequently gives a method for choosing $\vec{\beta}$ such that $\mathsf{Gen}(\mathsf{Add}_n, \vec{\beta})$ is guaranteed to be secure, by choosing one that makes the bound small.

▶ **Corollary 9.** *Let $n \geqslant 3$ and $\vec{\beta} = (\beta_1, \ldots, \beta_n) \in (F^*)^n$ where for every β_i, there exists some $\beta_j \neq \beta_i$ such that*

$$4cn \leqslant \mathrm{wt}^\vee(\beta_j \zeta^* \beta_i^{-1}) \leqslant \lambda - 4cn \tag{12}$$

for $c \in (5/36, 1/2 - 1/n)$. Then $\mathsf{Gen}(\mathsf{Add}_n, \vec{\beta})$ is λ^{-cn}-insecure against Hamming weight leakage as long as λ is sufficiently large.

Proof of Corollary 9. Consider any $\zeta \in \mathcal{S}_{\vec{\beta}}$. Then, by definition of $\mathcal{S}_{\vec{\beta}}$, we can write $\zeta = \zeta^* \beta_i^{-1}$ for some distinct multiplier β_i in $\vec{\beta}$. By condition, there is another distinct multiplier value $\beta_j \neq \beta_i$ appearing in $\vec{\beta}$ that gives $4cn \leqslant \mathrm{wt}^\vee(\beta_j \zeta^* \beta_i^{-1}) \leqslant \lambda - 4cn$. Therefore,

$$\mathsf{Score}(\zeta; \vec{\beta}) = \sum_{\ell=1}^{n} \sigma(\mathrm{wt}^\vee(\beta_\ell \cdot \zeta)) \geqslant \sigma(\mathrm{wt}^\vee(\beta_j \cdot \zeta)) = \sigma\big(\mathrm{wt}^\vee(\beta_j \zeta^* \beta_i^{-1})\big) \geqslant \left(\frac{8}{5}cn - \frac{1}{4}\right) \log \lambda.$$

The last inequality follows from the definition (Equation 2), and provided that λ is sufficiently large say, $\lambda \geqslant (4cn)^5$. Since $\zeta \in \mathcal{S}_{\vec{\beta}}$ was arbitrary, we get $\min_{\zeta \in \mathcal{S}_{\vec{\beta}}} \mathsf{Score}(\zeta; \vec{\beta}) \geqslant \left(\frac{8}{5}cn - \frac{1}{4}\right) \log \lambda$. It satisfies Equation 24 in the proof of Theorem 8 (as long as $cn > 5/12$ which by choosing $c > 5/36$ it always holds for all $n \geqslant 3$). Therefore, $\mathsf{Gen}(\mathsf{Add}_n, \vec{\beta})$ is at most λ^{-cn}-insecure by Theorem 8. ◀

▶ Remark 31. In light of Proposition 15, Corollary 9 also implies the security of $\mathsf{Shamir}(n, n, \vec{X})$ when the evaluation places $\vec{X} = (X_1, \ldots, X_n)$ are chosen uniformly at random. See the full version for details.

References

1 Donald Q. Adams, Hemanta K. Maji, Hai H. Nguyen, Minh L. Nguyen, Anat Paskin-Cherniavsky, Tom Suad, and Mingyuan Wang. Lower bounds for leakage-resilient secret-sharing schemes against probing attacks. In *2021 IEEE International Symposium on Information Theory (ISIT)*, pages 976–981, 2021. doi:10.1109/ISIT45174.2021.9518230.

2 Josep Balasch, Sebastian Faust, and Benedikt Gierlichs. Inner product masking revisited. In Elisabeth Oswald and Marc Fischlin, editors, *Advances in Cryptology – EUROCRYPT 2015, Part I*, volume 9056 of *Lecture Notes in Computer Science*, pages 486–510, Sofia, Bulgaria, April 26–30 2015. Springer Berlin Heidelberg, Germany. doi:10.1007/978-3-662-46800-5_19.

3 Josep Balasch, Sebastian Faust, Benedikt Gierlichs, Clara Paglialonga, and François-Xavier Standaert. Consolidating inner product masking. In Tsuyoshi Takagi and Thomas Peyrin, editors, *Advances in Cryptology – ASIACRYPT 2017, Part I*, volume 10624 of *Lecture Notes in Computer Science*, pages 724–754, Hong Kong, China, December 3–7 2017. Springer, Cham, Switzerland. doi:10.1007/978-3-319-70694-8_25.

4 Fabrice Benhamouda, Akshay Degwekar, Yuval Ishai, and Tal Rabin. On the local leakage resilience of linear secret sharing schemes. In Hovav Shacham and Alexandra Boldyreva, editors, *Advances in Cryptology – CRYPTO 2018, Part I*, volume 10991 of *Lecture Notes in Computer Science*, pages 531–561, Santa Barbara, CA, USA, August 19–23 2018. Springer, Cham, Switzerland. doi:10.1007/978-3-319-96884-1_18.

5 Fabrice Benhamouda, Akshay Degwekar, Yuval Ishai, and Tal Rabin. On the local leakage resilience of linear secret sharing schemes. *Journal of Cryptology*, 34(2):10, April 2021. doi:10.1007/s00145-021-09375-2.

6 Peter B Borwein. On the complexity of calculating factorials. *Journal of Algorithms*, 6(3):376–380, 1985. doi:10.1016/0196-6774(85)90006-9.

7 Luís T. A. N. Brandão and René Peralta. NIST IR 8214C: NIST first call for multi-party threshold schemes. https://csrc.nist.gov/pubs/ir/8214/c/ipd, Jan 25, 2023.

8 Richard P Brent and Paul Zimmermann. *Modern computer arithmetic*, volume 18. Cambridge University Press, 2010.

9 Suresh Chari, Charanjit S. Jutla, Josyula R. Rao, and Pankaj Rohatgi. Towards sound approaches to counteract power-analysis attacks. In Michael J. Wiener, editor, *Advances in Cryptology – CRYPTO'99*, volume 1666 of *Lecture Notes in Computer Science*, pages 398–412, Santa Barbara, CA, USA, August 15–19 1999. Springer Berlin Heidelberg, Germany. doi:10.1007/3-540-48405-1_26.

10 Thomas M. Cover and Joy A. Thomas. *Elements of information theory*. John Wiley & Sons, 1999.

11 Ronald Cramer, Ivan Bjerre Damgård, and Jesper Buus Nielsen. *Secure multiparty computation and secret sharing*. Cambridge University Press, 2015.

12 Diego Dominici. Asymptotic analysis of the krawtchouk polynomials by the WKB method. *The Ramanujan Journal*, 15:303–338, 2008. doi:10.1007/s11139-007-9078 -9.

13 Alexandre Duc, Stefan Dziembowski, and Sebastian Faust. Unifying leakage models: From probing attacks to noisy leakage. In Phong Q. Nguyen and Elisabeth Oswald, editors, *Advances in Cryptology – EUROCRYPT 2014*, volume 8441 of *Lecture Notes in Computer Science*, pages 423–440, Copenhagen, Denmark, May 11–15 2014. Springer Berlin Heidelberg, Germany. doi:10.1007/978-3-642-55220-5_24.

14 Alexandre Duc, Sebastian Faust, and François-Xavier Standaert. Making masking security proofs concrete - or how to evaluate the security of any leaking device. In Elisabeth Oswald and Marc Fischlin, editors, *Advances in Cryptology – EUROCRYPT 2015, Part I*, volume 9056 of *Lecture Notes in Computer Science*, pages 401–429, Sofia, Bulgaria, April 26–30 2015. Springer Berlin Heidelberg, Germany. doi:10.1007/978-3-662-46800-5_16.

15 François Durvaux, François-Xavier Standaert, and Santos Merino Del Pozo. Towards easy leakage certification. In Benedikt Gierlichs and Axel Y. Poschmann, editors, *Cryptographic Hardware and Embedded Systems – CHES 2016*, volume 9813 of *Lecture Notes in Computer Science*, pages 40–60, Santa Barbara, CA, USA, August 17–19 2016. Springer Berlin Heidelberg, Germany. doi:10.1007/978-3-662-53140-2_3.

16 François Durvaux, François-Xavier Standaert, and Nicolas Veyrat-Charvillon. How to certify the leakage of a chip? In Phong Q. Nguyen and Elisabeth Oswald, editors, *Advances in Cryptology – EUROCRYPT 2014*, volume 8441 of *Lecture Notes in Computer Science*, pages 459–476, Copenhagen, Denmark, May 11–15 2014. Springer Berlin Heidelberg, Germany. doi:10.1007/978-3-642-55220-5_26.

17 Sebastian Faust, Loïc Masure, Elena Micheli, Maximilian Orlt, and François-Xavier Standaert. Connecting leakage-resilient secret sharing to practice: Scaling trends and physical dependencies of prime field masking. In Marc Joye and Gregor Leander, editors, *Advances in Cryptology – EUROCRYPT 2024, Part IV*, volume 14654 of *Lecture Notes in Computer Science*, pages 316–344, Zurich, Switzerland, May 26–30 2024. Springer, Cham, Switzerland. doi:10.1007/978-3-031-58737-5_12.

18 Sebastian Faust, Loïc Masure, Elena Micheli, Hai Hoang Nguyen, Maximilian Orlt, and François-Xavier Standaert. IP masking with generic security guarantees under minimum assumptions, and applications. In *31st International Conference on the Theory and Application of Cryptology and Information Security – ASIACRYPT 2025*, december 8–12 2025.

19 Benedikt Gierlichs, Lejla Batina, Pim Tuyls, and Bart Preneel. Mutual information analysis. In Elisabeth Oswald and Pankaj Rohatgi, editors, *Cryptographic Hardware and Embedded Systems – CHES 2008*, volume 5154 of *Lecture Notes in Computer Science*, pages 426–442, Washington, DC, USA, August 10–13 2008. Springer Berlin Heidelberg, Germany. doi:10.1007/978-3-540-85053-3_27.

20 Louis Goubin and Ange Martinelli. Protecting AES with Shamir's secret sharing scheme. In Bart Preneel and Tsuyoshi Takagi, editors, *Cryptographic Hardware and Embedded Systems – CHES 2011*, volume 6917 of *Lecture Notes in Computer Science*, pages 79–94, Nara, Japan, September 28 – October 1 2011. Springer Berlin Heidelberg, Germany. doi:10.1007/978-3-642-23951-9_6.

21 Louis Goubin and Jacques Patarin. DES and differential power analysis (the "duplication" method). In Çetin Kaya Koç and Christof Paar, editors, *Cryptographic Hardware and Embedded Systems – CHES'99*, volume 1717 of *Lecture Notes in Computer Science*, pages 158–172, Worcester, Massachusetts, USA, August 12–13 1999. Springer Berlin Heidelberg, Germany. doi:10.1007/3-540-48059-5_15.

22 Jonathan I Hall. Notes on coding theory, 2003. URL: https://users.math.msu.edu/users/halljo/classes/CODENOTES/CODING-NOTES.HTML.

23 G. H. Hardy, J. E. Littlewood, and G. Pólya. *Inequalities*. Cambridge Mathematical Library. Cambridge University Press, 1952.

24 Dustin Kasser. An improvement upon the bounds for the local leakage resilience of shamir's secret sharing scheme. In Elette Boyle and Mohammad Mahmoody, editors, *TCC 2024: 22nd Theory of Cryptography Conference, Part IV*, volume 15367 of *Lecture Notes in Computer Science*, pages 395–422, Milan, Italy, December 2–6 2024. Springer, Cham, Switzerland. doi:10.1007/978-3-031-78023-3_13.

25 Ohad Klein and Ilan Komargodski. New bounds on the local leakage resilience of Shamir's secret sharing scheme. In Helena Handschuh and Anna Lysyanskaya, editors, *Advances in Cryptology – CRYPTO 2023, Part I*, volume 14081 of *Lecture Notes in Computer Science*, pages 139–170, Santa Barbara, CA, USA, August 20–24 2023. Springer, Cham, Switzerland. doi:10.1007/978-3-031-38557-5_5.

26 Paul C. Kocher. Timing attacks on implementations of Diffie-Hellman, RSA, DSS, and other systems. In Neal Koblitz, editor, *Advances in Cryptology – CRYPTO'96*, volume 1109 of *Lecture Notes in Computer Science*, pages 104–113, Santa Barbara, CA, USA, August 18–22 1996. Springer Berlin Heidelberg, Germany. doi:10.1007/3-540-68697-5_9.

27 Paul C. Kocher, Joshua Jaffe, and Benjamin Jun. Differential power analysis. In Michael J. Wiener, editor, *Advances in Cryptology – CRYPTO'99*, volume 1666 of *Lecture Notes in Computer Science*, pages 388–397, Santa Barbara, CA, USA, August 15–19 1999. Springer Berlin Heidelberg, Germany. doi:10.1007/3-540-48405-1_25.

28 Ilia Krasikov. Nonnegative quadratic forms and bounds on orthogonal polynomials. *Journal of Approximation Theory*, 111(1):31–49, July 2001. doi:10.1006/jath.2001.3570.

29 Florence Jessie MacWilliams and Neil James Alexander Sloane. *The theory of error-correcting codes*, volume 16. North-Holland Publishing Company, 1977.

30 Hemanta K. Maji, Hai H. Nguyen, Anat Paskin-Cherniavsky, Tom Suad, and Mingyuan Wang. Leakage-resilience of the shamir secret-sharing scheme against physical-bit leakages. In Anne Canteaut and François-Xavier Standaert, editors, *Advances in Cryptology – EUROCRYPT 2021, Part II*, volume 12697 of *Lecture Notes in Computer Science*, pages 344–374, Zagreb, Croatia, October 17–21 2021. Springer, Cham, Switzerland. doi:10.1007/978-3-030-77886-6_12.

31 Hemanta K. Maji, Hai H. Nguyen, Anat Paskin-Cherniavsky, Tom Suad, Mingyuan Wang, Xiuyu Ye, and Albert Yu. Tight estimate of the local leakage resilience of the additive secret-sharing scheme & its consequences. In Dana Dachman-Soled, editor, *ITC 2022: 3rd Conference on Information-Theoretic Cryptography*, volume 230 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 16:1–16:19, Cambridge, MA, USA, July 5–7 2022. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.ITC.2022.16.

32 Hemanta K. Maji, Hai H. Nguyen, Anat Paskin-Cherniavsky, and Mingyuan Wang. Improved bound on the local leakage-resilience of shamir's secret sharing. In *IEEE International Symposium on Information Theory, ISIT 2022, Espoo, Finland, June 26 - July 1, 2022*, pages 2678–2683. IEEE, 2022. doi:10.1109/ISIT50566.2022.9834695.

33 Hemanta K. Maji, Hai H. Nguyen, Anat Paskin-Cherniavsky, and Xiuyu Ye. Constructing leakage-resilient Shamir's secret sharing: Over composite order fields. In Marc Joye and Gregor Leander, editors, *Advances in Cryptology – EUROCRYPT 2024, Part IV*, volume 14654 of *Lecture Notes in Computer Science*, pages 286–315, Zurich, Switzerland, May 26–30 2024. Springer, Cham, Switzerland. doi:10.1007/978-3-031-58737-5_11.

34 Hemanta K. Maji, Anat Paskin-Cherniavsky, Tom Suad, and Mingyuan Wang. Constructing locally leakage-resilient linear secret-sharing schemes. In Tal Malkin and Chris Peikert, editors, *Advances in Cryptology – CRYPTO 2021, Part III*, volume 12827 of *Lecture Notes in Computer Science*, pages 779–808, Virtual Event, August 16–20 2021. Springer, Cham, Switzerland. doi:10.1007/978-3-030-84252-9_26.

35 James L. Massey. Some applications of coding theory in cryptography. *Codes and Ciphers: Cryptography and Coding IV*, pages 33–47, 1995.

36 Colin McDiarmid. *Concentration*, pages 195–248. Springer Berlin Heidelberg, Berlin, Heidelberg, 1998. doi:10.1007/978-3-662-12788-9_6.

37 Hai H. Nguyen. Towards breaking the half-barrier of local leakage-resilient shamir's secret sharing. In Leonid Reyzin and Douglas Stebila, editors, *Advances in Cryptology – CRYPTO 2024, Part V*, volume 14924 of *Lecture Notes in Computer Science*, pages 257–285, Santa Barbara, CA, USA, August 18–22 2024. Springer, Cham, Switzerland. doi:10.1007/978-3-031-68388-6_10.

38 Ryan O'Donnell. *Analysis of boolean functions*. Cambridge University Press, 2021. Available online as arXiv:2105.10386v1. arXiv:2105.10386.

39 Emmanuel Prouff and Thomas Roche. Higher-order glitches free implementation of the AES using secure multi-party computation protocols. In Bart Preneel and Tsuyoshi Takagi, editors, *Cryptographic Hardware and Embedded Systems – CHES 2011*, volume 6917 of *Lecture Notes in Computer Science*, pages 63–78, Nara, Japan, September 28 – October 1 2011. Springer Berlin Heidelberg, Germany. doi:10.1007/978-3-642-23951-9_5.

40 François-Xavier Standaert, Tal Malkin, and Moti Yung. A unified framework for the analysis of side-channel key recovery attacks. In Antoine Joux, editor, *Advances in Cryptology – EUROCRYPT 2009*, volume 5479 of *Lecture Notes in Computer Science*, pages 443–461, Cologne, Germany, April 26–30 2009. Springer Berlin Heidelberg, Germany. doi:10.1007/978-3-642-01001-9_26.

41 Jacobus Hendricus van Lint. *Introduction to coding theory*, volume 86. Springer Science & Business Media, 1998.

42 Joachim von zur Gathen and Victor Shoup. Computing frobenius maps and factoring polynomials. *Computational Complexity*, 2:187–224, September 1992. doi:10.1007/BF01272074.

When Does Quantum Differential Privacy Compose?

Daniel Alabi
University of Illinois at Urbana-Champaign, IL, USA

Theshani Nuradha
University of Illinois at Urbana-Champaign, IL, USA

Abstract

Composition is a cornerstone of classical differential privacy, enabling strong end-to-end guarantees for complex algorithms through composition theorems (e.g., basic and advanced). In the quantum setting, however, privacy is defined operationally against arbitrary measurements, and classical composition arguments based on scalar privacy-loss random variables no longer apply. As a result, it has remained unclear when meaningful composition guarantees can be obtained for quantum differential privacy (QDP).

In this work, we clarify both the limitations and possibilities of composition in the quantum setting. We first show that classical-style composition fails in full generality for POVM-based approximate QDP: even quantum channels that are individually perfectly private can completely lose privacy when combined through correlated joint implementations.

We then identify a setting in which clean composition guarantees can be restored. For tensor-product channels acting on product neighboring inputs, we introduce a *quantum moments accountant* based on an operator-valued notion of privacy loss and a matrix moment-generating function. Although the resulting Rényi-type divergence does not satisfy a data-processing inequality, we prove that controlling its moments suffices to bound measured Rényi divergence, yielding operational privacy guarantees against arbitrary measurements. This leads to advanced-composition-style bounds with the same leading-order behavior as in the classical theory.

Our results demonstrate that meaningful composition theorems for quantum differential privacy require carefully articulated structural assumptions on channels, inputs, and adversarial measurements, and provide a principled framework for understanding which classical ideas do and do not extend to the quantum setting.

2012 ACM Subject Classification Security and privacy → Information-theoretic techniques; Theory of computation → Quantum information theory

Keywords and phrases Quantum Information, Differential Privacy

Digital Object Identifier 10.4230/LIPIcs.ITC.2026.10

Related Version *Full Version*: https://arxiv.org/abs/2601.00337

Funding *Daniel Alabi*: Research supported by the Simons Foundation (965342, D.A.) as part of the Junior Fellowship from the Simons Society of Fellows.
Theshani Nuradha: Research supported by the Department of Mathematics and the IQUIST Postdoctoral Fellowship from the Illinois Quantum Information Science and Technology Center at the University of Illinois Urbana-Champaign.

Acknowledgements DA acknowledges helpful discussions with Guy Rothblum and Salil Vadhan regarding the composition of classical and quantum private mechanisms. TN acknowledges helpful discussions with Sujeet Bhalerao, Felix Leditzky, Vishal Singh, and Mark M. Wilde on the composition of quantum private mechanisms.

1 Introduction

Differential privacy (DP) provides a rigorous framework for limiting information leakage about sensitive inputs under randomized data analysis [11, 10]. With the growth of quantum technologies and the generation of quantum data, it is crucial to protect sensitive information related to the data generated and the data that is fed into these developing hybrid classical–quantum systems. While the theory of classical differential privacy is by now mature, extending its guarantees to quantum information processing raises both conceptual and technical challenges [31]. In fully or hybrid (i.e., classical–quantum) quantum processing pipelines, we need a definition that is native to that setting. The outputs of the algorithms and the processing techniques have to obey the laws of quantum mechanics. Classical DP does not capture this because the released object (if any) from quantum algorithms should be a quantum state, and an adversary may apply arbitrary measurements, including collective measurements across multiple outputs, to that state.

With that, quantum DP is defined in a way to ensure privacy for quantum states such that any two quantum states that are classified as neighbors are indistinguishable under all measurements up to a certain limit demanded by privacy parameters [31, 16] (see also Definition 14). In the quantum setting, outputs of quantum systems are quantum states rather than classical samples, adversaries may perform arbitrary measurements, and correlations or entanglement across multiple outputs can fundamentally alter distinguishability properties [22, 19]. These features complicate even basic questions about composition, a cornerstone of the classical DP theory.

In classical DP, composition guarantees are derived by tracking a scalar *privacy-loss random variable* whose moment generating function (MGF) is additive under independent composition [12]. This structure underlies both basic composition and advanced composition results, including the moments accountant framework of Abadi et al. [2]. In contrast, for quantum channels privacy loss is not a scalar quantity prior to measurement, and naively taking a supremum over all measurements destroys additivity since measured quantum divergences are not always additive or at least sub-additive. As a result, classical composition arguments do not directly extend to quantum differential privacy (QDP).

Much of the existing literature on composition in quantum differential privacy has taken an explicitly adversary-centric viewpoint [16, 25]. In this line of work, composition is analyzed primarily through the lens of what an adversary may do to the outputs of multiple private mechanisms, most notably by quantifying over increasingly powerful classes of measurements [25, 26]. While this perspective is natural from an operational security standpoint, it largely abstracts away how quantum systems are actually composed in practice. In quantum information processing, composition is implemented by specific channel constructions (such as tensor-product composition, factorized releases, or correlated joint implementations) whose structural properties can be as consequential for privacy as the adversary's measurement capabilities [1]. By explicitly separating the model of channel composition from the model of adversarial measurement, our work shifts part of the focus from "what measurements are allowed" to "how mechanisms are combined," and shows that many composition phenomena in quantum differential privacy are driven as much by the structure of the joint channel as by the power of the adversary.

This work develops a principled framework for understanding when and how composition guarantees can be recovered in the quantum setting. Our analysis makes two key points. First, classical-style composition fails in full generality for POVM-based approximate QDP when correlated joint channels or entangled neighboring inputs are allowed. Second, under

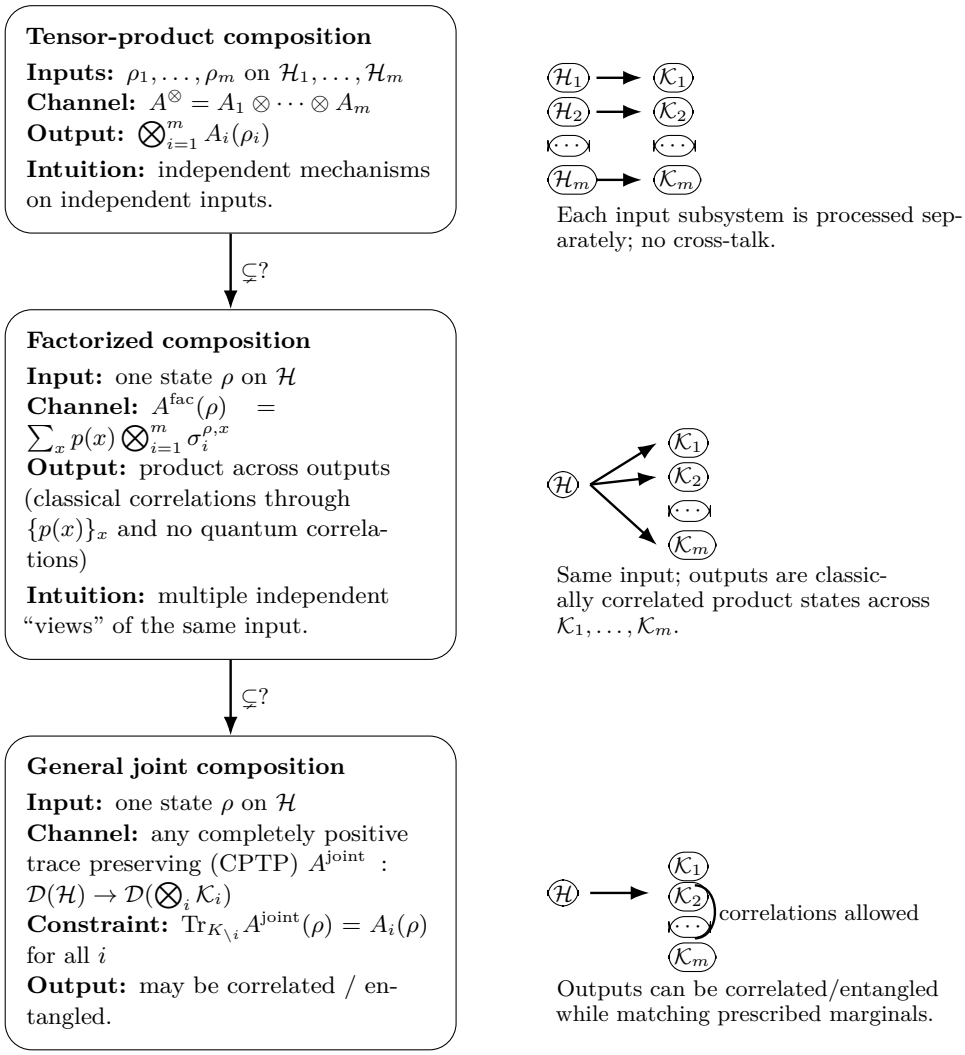


Figure 1 A hierarchy for multi-output composition models.

carefully articulated structural assumptions (most notably tensor-product channels acting on product neighboring inputs) one can recover clean and quantitatively sharp composition guarantees via an operator-level analogue of the classical moments accountant.

To achieve this, we introduce a *quantum moments accountant* (QMA) based on the privacy-loss operator and a matrix moment-generating function. While the resulting Rényi-type divergence does not satisfy a data-processing inequality in general, we show that controlling its moments suffices to bound *measured Rényi divergence*, which by definition captures worst-case distinguishability over all measurements. This yields a direct route from operator-level moment bounds to operational (ε, δ)-QDP guarantees.

Throughout the paper, we emphasize the importance of distinguishing between different models of composition. We propose a strict hierarchy (i.e., tensor-product, factorized $\subsetneq$ general joint composition) and show that failures of classical composition arise precisely when moving beyond tensor-product structure. Our results therefore clarify not only what is possible in quantum composition, but also why additional assumptions are unavoidable.

Contributions

This paper makes the following contributions.

- **A hierarchy of quantum composition models.** We formalize and distinguish tensor-product, factorized, and general joint composition of quantum channels, and show that these form a strict hierarchy. This framework, illustrated in Figure 1, isolates exactly where classical composition guarantees break down in the quantum setting. See Section 2.3.
- **Impossibility of basic composition for general joint channels.** We prove, in Theorem 15, that even pure $(0, 0)$-QDP mechanisms can fail to compose under correlated joint implementations. In particular, we show that, for $\varepsilon_1, \varepsilon_2, \delta_1, \delta_2 \geq 0$ no general $(\varepsilon_1 + \varepsilon_2, \delta_1 + \delta_2)$-type composition theorem can hold for POVM-based approximate QDP without further restrictions. See Section 4 for more details.
- **Safe basic composition under restricted adversaries.** For tensor-product channels on product neighboring inputs, we prove (in Theorem 18) basic composition guarantees against one-way classical communication and local-operations-based (one-way LOCC) adversaries, providing a clean quantum analogue of classical sequential composition under explicitly stated assumptions. See Section 5 for more details.
- **A quantum moments accountant.** In Section 6, we introduce a non-commutative analogue of the classical moments accountant based on the quantum privacy-loss operator and a matrix moment-generating function. We show, via Theorem 27, that this accountant composes additively under tensor-product channels and product neighbors.
- **From operator moments to operational privacy.** We prove (Theorem 30) that bounds on the quantum moments accountant imply bounds on measured Rényi divergence, yielding operational (ε, δ)-QDP guarantees against arbitrary POVMs. This provides an advanced-composition-style bound with the familiar $\sqrt{\sum_i \varepsilon_i^2 \log(1/\delta)}$ scaling over k $(\varepsilon_i, 0)$-QDP mechanisms. See Proposition 28 and Corollary 31.
- **Advanced composition for QDP.** We prove an advanced composition result for tensor-product composition (Theorem 33) of k $(\varepsilon_i, \delta_i)$-QDP mechanisms under all possible adversaries for $\delta_i = 0$, with the composed mechanism satisfying the privacy parameter $\sqrt{\sum_i \varepsilon_i^2 \log(1/\delta)}$ for $\delta \in (0, 1)$.
- **Clarifying the quantum–classical gap.** Our results identify measurement incompatibility and the absence of a joint classical probability space as the fundamental obstacles to classical composition arguments, rather than a failure of tensor-product additivity at the channel level. See Section 2.3.

2 Preliminaries and Definitions

We collect notation, definitions, and structural distinctions that will be used throughout the paper. Because composition behavior in the quantum setting depends sensitively on how channels are combined, how neighboring inputs are defined, and what adversarial measurements are permitted, we make these modeling choices explicit at the outset. In particular, we distinguish several notions of multi-output composition that coincide classically but diverge sharply in the quantum setting.

2.1 Notation and Conventions

All Hilbert spaces are finite-dimensional. For a Hilbert space $\mathcal{H}$, we write $\mathcal{D}(\mathcal{H})$ for the set of density operators on $\mathcal{H}$, i.e., positive semidefinite operators with unit trace.

▶ **Definition 1** (Density operators). *Let $\mathcal{H}$ be a finite-dimensional Hilbert space. We write $\mathcal{D}(\mathcal{H}) := \{\rho \in \mathsf{L}(\mathcal{H}) \mid \rho \succeq 0,\ \mathrm{Tr}(\rho) = 1\}$, where $\mathsf{L}(\mathcal{H})$ denotes the set of linear operators on $\mathcal{H}$. Elements of $\mathcal{D}(\mathcal{H})$ are called* density operators *or* quantum states.

Given a composite system $\mathcal{H}_1 \otimes \cdots \otimes \mathcal{H}_m$, we write $\mathrm{Tr}_{\mathcal{H}\setminus i}$ for the partial trace over all subsystems except $\mathcal{H}_i$. For operators X, Y we write $X \preceq Y$ to denote that $Y - X$ is positive semidefinite.

Throughout, logarithms are natural unless otherwise stated. When defining operator expressions involving inverses or logarithms, we adopt the convention that the expression is $+\infty$ whenever the required support conditions fail.

2.2 Channels

Quantum channels are completely positive, trace-preserving (CPTP) maps between spaces of density operators.

▶ **Definition 2** (Tensor-product channels). *Let $\mathcal{H}_1, \mathcal{H}_2, \ldots, \mathcal{H}_m$ and $\mathcal{K}_1, \mathcal{K}_2, \ldots, \mathcal{K}_m$ be finite-dimensional Hilbert spaces. Let $A_i : \mathcal{D}(\mathcal{H}_i) \to \mathcal{D}(\mathcal{K}_i)$, $i \in \{1, 2, \ldots, m\}$, be quantum channels (i.e., completely positive, trace-preserving maps). The* tensor-product channel *associated with $(A_1, \ldots, A_m)$ is the channel $A_1 \otimes \cdots \otimes A_m : \mathcal{D}(\mathcal{H}_1 \otimes \cdots \otimes \mathcal{H}_m) \longrightarrow \mathcal{D}(\mathcal{K}_1 \otimes \cdots \otimes \mathcal{K}_m)$ defined by $(A_1 \otimes \cdots \otimes A_m)(\rho_1 \otimes \cdots \otimes \rho_m) := A_1(\rho_1) \otimes \cdots \otimes A_m(\rho_m)$, and extended linearly to all inputs $\rho \in \mathcal{D}(\mathcal{H}_1 \otimes \cdots \otimes \mathcal{H}_m)$. Equivalently, $A_1 \otimes \cdots \otimes A_m$ is a CPTP map whose Kraus operators are all tensor products of Kraus operators of $A_1, \ldots, A_m$.*

▶ **Definition 3** (Factorized channels). *Let $\mathcal{H}$ be a finite-dimensional Hilbert space and let $\mathcal{K}_1, \ldots, \mathcal{K}_m$ be finite-dimensional Hilbert spaces. A quantum channel $A : \mathcal{D}(\mathcal{H}) \to \mathcal{D}(\mathcal{K}_1 \otimes \cdots \otimes \mathcal{K}_m)$ is called* factorized *if $A(\rho) = \sum_x p(x) \bigotimes_{i=1}^m \sigma_i^{\rho,x}$, where $\sigma_i^{\rho,x}$ is a quantum state for all i and x, and $\{p(x)\}_x$ is a probability distribution with $p(x) \geq 0$ and $\sum_x p(x) = 1$. In this case, the m outputs are not entangled and independent given the input state.*

Note that a factorized channel can measure the input, copy the classical data, and then prepare quantum states conditioned on the measurement outcome. This has been done in previous work [25].

▶ **Definition 4** (General joint channels). *Let $\mathcal{H}$ be a finite-dimensional Hilbert space and let $\mathcal{K}_1, \ldots, \mathcal{K}_m$ be finite-dimensional Hilbert spaces. A* general joint channel *is any quantum channel $A : \mathcal{D}(\mathcal{H}) \to \mathcal{D}(\mathcal{K}_1 \otimes \cdots \otimes \mathcal{K}_m)$ that is completely positive and trace-preserving, with no further structural restrictions. Equivalently, a general joint channel may produce arbitrary correlations or entanglement among the output subsystems $\mathcal{K}_1, \ldots, \mathcal{K}_m$, and need not admit any factorization or tensor-product decomposition.*

2.3 Models of Composition

Classically, releasing multiple outputs of differentially private mechanisms implicitly defines a joint distribution over all outputs. In the quantum setting, however, there is no unique joint state consistent with a collection of marginals, and different joint implementations can exhibit drastically different privacy behavior. We therefore distinguish three composition models:

1. **Tensor-product composition**, where independent channels act on independent input subsystems.
2. **Factorized composition**, where multiple channels act on the input but the joint output is constrained to be a product state (not entangled).
3. **General joint composition**, where only the marginal behavior of each output is fixed, and the joint output may be arbitrarily correlated or entangled.

While these notions might coincide in the classical setting, they form a hierarchy in the quantum setting. Much of the subtlety of quantum composition arises from the gap between factorized and general joint composition.

▶ **Definition 5** (Composition into tensor-product channels). *Let* $A_i : \mathcal{D}(\mathcal{H}_i) \to \mathcal{D}(\mathcal{K}_i),\ i = 1, \ldots, m$, *be quantum channels acting on (possibly distinct) input systems. The* composition into a tensor-product channel *is the joint channel* $A^{\otimes} := A_1 \otimes \cdots \otimes A_m : \mathcal{D}(\mathcal{H}_1 \otimes \cdots \otimes \mathcal{H}_m) \to \mathcal{D}(\mathcal{K}_1 \otimes \cdots \otimes \mathcal{K}_m)$, *defined by* $A^{\otimes}(\rho_1 \otimes \cdots \otimes \rho_m) = A_1(\rho_1) \otimes \cdots \otimes A_m(\rho_m)$, *and extended linearly to all joint inputs. This models the composition of* independent mechanisms acting on independent inputs*.*

Note that tensor-product composition is the quantum analogue of classical parallel composition.

▶ **Definition 6** (Composition into factorized channels). *The* composition into a factorized channel *is the joint channel* $A^{\mathrm{fac}} : \mathcal{D}(\mathcal{H}) \to \mathcal{D}(\mathcal{K}_1 \otimes \cdots \otimes \mathcal{K}_m)$ *defined by* $A^{\mathrm{fac}}(\rho) = \sum_x p(x) \bigotimes_{i=1}^m \sigma_i^{\rho,x}$ *In this composition, all outputs are conditionally independent given the input state (not entangled), but each output may reveal different information about that input.*

▶ **Definition 7** (Composition into general joint channels). *Let* $A_i : \mathcal{D}(\mathcal{H}) \to \mathcal{D}(\mathcal{K}_i),\ i = 1, \ldots, m$, *be quantum channels with a common input space. A* composition into a general joint channel *is any quantum channel* $A^{\mathrm{joint}} : \mathcal{D}(\mathcal{H}) \to \mathcal{D}(\mathcal{K}_1 \otimes \cdots \otimes \mathcal{K}_m)$ *whose marginals coincide with the individual channels, i.e.,* $\mathrm{Tr}_{\mathcal{K}_{\setminus i}}\big(A^{\mathrm{joint}}(\rho)\big) = A_i(\rho)$ *for all* $\rho \in \mathcal{D}(\mathcal{H})$ *and all* i*, where* $\mathcal{K}_{\setminus i} := \bigotimes_{j \neq i} \mathcal{K}_j$*. Such a composition may introduce arbitrary classical or quantum correlations (including entanglement) among the outputs and strictly generalizes both tensor-product and factorized compositions.*

▶ Remark 8 (Tensor-product vs. factorized composition). Tensor-product and factorized composition impose constraints along different axes. Tensor-product composition restricts how a joint channel is *implemented*, requiring independent local mechanisms, but allows correlated or entangled outputs on entangled inputs. In contrast, factorized composition restricts the *output structure*, requiring a product state for every system (possibly with classical correlations), but does not require that the channel decompose as a tensor product of local channels.

On product inputs $\rho = \rho_1 \otimes \cdots \otimes \rho_m$, the output of a tensor-product channel is product: $(A_1 \otimes \cdots \otimes A_m)(\rho) = A_1(\rho_1) \otimes \cdots \otimes A_m(\rho_m)$, but this coincidence does not extend to entangled inputs.

Examples

▶ **Example 9** (Example of tensor-product channel). Let $\mathcal{H}_1 = \mathcal{H}_2 = \mathbb{C}^2$ and $\mathcal{K}_1 = \mathcal{K}_2 = \mathbb{C}^2$. Let A_1 and A_2 be single-qubit depolarizing channels, $A_i(\rho) = (1 - p_i)\rho + p_i \frac{I}{2},\ i \in \{1, 2\}$. Then the tensor-product channel $A_1 \otimes A_2$ acts on two-qubit inputs as $(A_1 \otimes A_2)(\rho_1 \otimes \rho_2) = A_1(\rho_1) \otimes A_2(\rho_2)$, and introduces no correlations between the two output qubits beyond those already present in the input. If the input is a product state, the output is also a product state.

▶ **Example 10** (Example of factorized channel). Let $\mathcal{H} = \mathbb{C}^2$ and let $\mathcal{K}_i = \mathbb{C}^2$ for $i \in \{1, \ldots, m\}$. Consider measure and prepare channel A that measures the state ρ with the POVM $\{M_x\}_x$ and then depending on the classical outcome, prepare state $\otimes_{i=1}^m \sigma_i^x$. Then the joint channel $\mathcal{A}(\rho) = \sum_x \mathrm{Tr}(M_x \rho)\ \sigma_1^x \otimes \cdots \otimes \sigma_m^x$ is a valid factorized channel.

▶ **Example 11** (Example of general joint channels). Let $\mathcal{H} = \mathbb{C}^2$ and $\mathcal{K}_1 = \mathcal{K}_2 = \mathbb{C}^2$. Define a channel A that appends an ancilla qubit initialized to $|0\rangle$ and applies a CNOT gate:

$$A(\rho) = \mathrm{CNOT}\,(\rho \otimes |0\rangle\langle 0|)\,\mathrm{CNOT}^\dagger.$$

The resulting joint output state on $\mathcal{K}_1 \otimes \mathcal{K}_2$ is generally entangled. This channel is a valid joint channel but is neither tensor-product nor factorized, since the two output subsystems are correlated even when the input is pure.

▶ Remark 12 (Distinguishing the three notions). Tensor-product channels describe independent mechanisms acting on independent inputs. Factorized channels describe multiple independent outputs derived from the same input. General joint channels allow arbitrary correlations or entanglement across outputs. These distinctions are crucial when reasoning about composition and privacy guarantees in the quantum setting.

▶ **Proposition 13** (Containment relations between composition models). *Let $m \geq 2$ and let $\mathcal{H}, \mathcal{H}_1, \ldots, \mathcal{H}_m$ and $\mathcal{K}_1, \ldots, \mathcal{K}_m$ be finite-dimensional Hilbert spaces.*

1. ***Tensor-product implies factorization on product inputs.*** *Let $A^{\otimes} := A_1 \otimes \cdots \otimes A_m$ be a tensor-product composition of channels $A_i : \mathcal{D}(H_i) \to \mathcal{D}(K_i)$. Then for every product input state$\rho = \rho_1 \otimes \cdots \otimes \rho_m$, the output of $A^{\otimes}$ is factorized: $A^{\otimes}(\rho) = A_1(\rho_1) \otimes \cdots \otimes A_m(\rho_m)$. Equivalently, when restricted to product inputs, a tensor-product channel coincides with a factorized channel (notice that in this case we set the probability distribution to a singleton).*
2. ***Factorized $\subseteq$ general joint.*** *Every factorized composition is a special case of general joint composition. Indeed, if $A^{\mathrm{fac}}(\rho) = \sum_x p(x) \bigotimes_{i=1}^m \sigma_i^{\rho,x}$, then A^{fac} is a joint channel whose marginals are exactly the $A_i(\rho) = \sum_x p(x)\sigma_i^{x,\rho}$.*
3. ***Strictness of inclusions.*** *Both inclusions above are strict:*
   - *There exist factorized channels that are not tensor-product channels (e.g., multiple outputs that are independent and possibly only classically correlated for a given input state).*
   - *There exist general joint channels that are not factorized (e.g., channels that produce entangled outputs while preserving the same marginals).*

Proof. Follows from Definition 5, Definition 6, Definition 7 and Example 9,Example 10,Example 11. ◀

2.4 Operational Quantum Differential Privacy

In the classical setting, differential privacy is defined in terms of the output distributions of randomized algorithms. In the quantum setting, the output of a mechanism is a quantum state, and privacy must therefore be defined operationally in terms of the statistics produced by measurements. This leads naturally to a definition that quantifies over all POVM effects applied to the channel output [31, 16]:

▶ **Definition 14** ((Approximate) Quantum Differential Privacy). *A quantum channel (CPTP map) A is (ε, δ)-*QDP *if for all neighboring input states $\rho \sim \sigma$ and for every measurement operator M (i.e., satisfying $0 \preceq M \preceq I$),*

$$\mathrm{Tr}\big(MA(\rho)\big) \;\leq\; e^{\varepsilon}\mathrm{Tr}\big(MA(\sigma)\big) \;+\; \delta. \tag{1}$$

Importantly, this operational definition captures adversaries with unrestricted measurement power. While alternative notions of quantum privacy can be defined (e.g., by restricting measurements or comparing states directly via trace distance [25]), the general quantum differential privacy definition most directly mirrors the classical adversarial model and is the strongest notion considered in this work.

Neighboring Inputs

We use the standard notion of neighboring inputs adapted to quantum states. Two density operators $\rho, \sigma \in \mathcal{D}(\mathcal{H})$ are said to be *neighbors*, denoted $\rho \sim \sigma$, if they differ in the data of a single individual or record. The precise definition is application-dependent and left abstract in this work; all results hold for any fixed neighboring relation. When considering multi-system inputs, we distinguish between:

- *Product neighbors*, where $\rho = \bigotimes_i \rho_i$ and $\sigma = \bigotimes_i \sigma_i$ with $\rho_i \sim \sigma_i$ for each subsystem; and
- *General neighbors*, which may be entangled or classically correlated across subsystems.

This distinction plays a critical role in our composition results and impossibility theorems.

2.5 Roadmap

The distinctions introduced above are not merely definitional. In Section 4, we show that basic composition fails for approximate QDP under general joint composition. In contrast, Sections 5 and 6 show that clean composition guarantees can be recovered for tensor-product channels acting on product neighboring inputs, culminating in an advanced-composition-style bound via the quantum moments accountant. First, we discuss some additional related work.

3 Related Work

Classical differential privacy and composition

Composition theorems are central to classical differential privacy [17]. Basic composition and advanced composition results were established early in the literature, culminating in the moments accountant framework of Abadi et al. [2] for tracking privacy loss in iterative algorithms such as DP-SGD. Rényi differential privacy (RDP) and its variants further systematized composition by working directly with Rényi divergences [20].

Quantum differential privacy

Quantum differential privacy (QDP) was first introduced in [31], and several variants of QDP considering distinguishability under all possible measurements (adversaries) have been studied in [1, 16, 3, 25, 4, 15, 14]. Furthermore, a variant that generalizes QDP by encoding domain knowledge and measurement capabilities (practical possibilities of the adversary) of the systems, known as quantum pufferfish privacy (QPP), was introduced in [25] and further studied in [26] with the inspiration from its classical variants [18, 24]. This variant highlights, in some cases, how certain mechanisms are private when we consider the limitations of the measurements that can be performed, in contrast to those mechanisms being non-private when all measurements are allowed. Prior works have explored relationships between QDP mechanisms, quantum hypothesis testing, and quantum Rényi divergences, as well as connections to classical DP under commuting states [16, 25, 13, 4, 27, 6, 7].

In terms of composition of private mechanisms, for the variant of quantum local differential privacy (Definition 14 for $\delta = 0$ and neighbors declared as $\rho \sim \sigma$ for all pairs of distinct states), (basic) composition of several such private mechanisms when allowed measurements

are having locally motivated structures has been studied in [15], and in [16, 25] for QDP and QPP, some basic composition results were provided. In the setting where one applies private channels one after the other sequentially, sequential composition results for QDP have been studied using strong data-processing inequalities [16, 23]. However, more generally, composition guarantees in the quantum setting remain far less understood in a unified way, particularly in the presence of entanglement and correlated outputs.

Rényi divergences in quantum information

Quantum Rényi divergences, including the Petz and sandwiched variants [21, 5, 30], play a central role in quantum information theory [9]. The sandwiched Rényi divergence satisfies a data-processing inequality and has been used to derive operational guarantees in a variety of settings. In contrast, the MMGF-induced divergence we consider is tailored to moment accounting and exact additivity, rather than monotonicity under channels.

Limits of composition in non-classical settings

Impossibility results related to composition have appeared in other non-classical or non-i.i.d. settings, where joint distributions or correlated releases invalidate union-bound-based arguments [17]. Our no-go results for general joint quantum channels fit squarely into this theme and provide a concrete quantum-mechanical explanation rooted in measurement incompatibility.

Our contribution in context

Relative to prior work, this paper provides the first systematic treatment of when classical-style composition can and cannot be recovered for quantum differential privacy. By explicitly separating structural assumptions on channels, neighboring inputs, and adversaries, we reconcile negative results with positive composition theorems and introduce a moments-accountant-style framework that is both quantum-native and operationally meaningful.

4 Why Basic Composition for Approximate DP Fails for General Joint Channels

Basic composition (classical)

In the classical setting, if M_1 is $(\varepsilon_1, \delta_1)$-DP and M_2 is $(\varepsilon_2, \delta_2)$-DP, then the joint release (M_1, M_2) is $(\varepsilon_1 + \varepsilon_2, \delta_1 + \delta_2)$-DP. The proof relies on representing privacy loss as a scalar random variable and applying a union bound to the "δ-bad" events.

4.1 Failure of Basic Composition for Quantum DP

We now show that the classical basic composition theorem does not extend to Definition 14.

▶ **Theorem 15** (No-go for basic composition under POVM-QDP). *There exist quantum channels A_1, A_2 and neighboring inputs $\rho \sim \sigma$ such that:*

1. *each of A_1 and A_2 is $(0, 0)$-QDP, yet*
2. *there exists a composition of A_1, A_2 into a joint channel (Definition 7) that is not (ε, δ)-QDP for any $\delta < 1$.*

Consequently, no general rule of the form

$$(\varepsilon_1, \delta_1)\text{-}QDP + (\varepsilon_2, \delta_2)\text{-}QDP \Longrightarrow (\varepsilon_1 + \varepsilon_2, \delta_1 + \delta_2)\text{-}QDP$$

can hold for POVM-based quantum differential privacy.

Proof. Let the database be a single bit $b \in \{0, 1\}$ encoded by orthogonal states $\rho_0 = |0\rangle\langle 0|$ and $\rho_1 = |1\rangle\langle 1|$, which we declare neighboring. Define a joint channel A_{12} with two-qubit output systems A and B by

$$A_{12}(\rho_0) = |\Phi^+\rangle\langle\Phi^+|, \qquad A_{12}(\rho_1) = |\Phi^-\rangle\langle\Phi^-|,$$

where $|\Phi^\pm\rangle := \frac{1}{\sqrt{2}}(|00\rangle \pm |11\rangle)$ are Bell states.

Define the individual mechanisms by taking marginals: $A_1 := \mathrm{Tr}_B \circ A_{12},\ A_2 := \mathrm{Tr}_A \circ A_{12}$. Since $\mathrm{Tr}_B(|\Phi^+\rangle\langle\Phi^+|) = \mathrm{Tr}_B(|\Phi^-\rangle\langle\Phi^-|) = \frac{I}{2}$, we have $A_1(\rho_0) = A_1(\rho_1)$, and similarly $A_2(\rho_0) = A_2(\rho_1)$. Hence for every POVM effect M, $\mathrm{Tr}(MA_i(\rho_0)) = \mathrm{Tr}(MA_i(\rho_1))$, $i \in \{1, 2\}$, so both A_1 and A_2 are $(0, 0)$-QDP. However, the joint outputs $A_{12}(\rho_0)$ and $A_{12}(\rho_1)$ are orthogonal. Let $M = |\Phi^+\rangle\langle\Phi^+|$. Then $\mathrm{Tr}(MA_{12}(\rho_0)) = 1,\ \mathrm{Tr}(MA_{12}(\rho_1)) = 0$. If A_{12} were (ε, δ)-QDP with $\delta < 1$, Definition 14 would imply $1 \leq \delta$, a contradiction. ◀

▶ Remark 16 (Clarification about Theorem 15). The joint channel refers to the correlated implementation (see Definition 7) given by the single joint channel A_{12} outputting both subsystems, not to the independent tensor product channel $A_1 \otimes A_2$ (see Definition 5).

Source of the composition failure

The failure of composition in Theorem 15 does *not* arise from entangled neighboring inputs. The neighboring states $\rho_0 = |0\rangle\langle 0|$ and $\rho_1 = |1\rangle\langle 1|$ are single-qubit, orthogonal, and unentangled. Rather, the pathology arises from allowing a general joint channel whose marginals are fixed but whose joint action introduces correlations between outputs. Although each marginal channel is perfectly private, the joint channel can amplify distinguishability through correlated outputs.

4.2 Why This Phenomenon Is Quantum

Classical approximate DP relies on two properties:

1. *Existence of a single joint probability space* supporting all outcomes and all privacy-loss events.
2. *Scalar privacy loss*: the likelihood ratio is a random variable, and δ controls the probability of large deviations via a union bound.

In contrast, Definition 14 quantifies over all POVMs *after* the channel. Different POVMs are generally incompatible and cannot be realized jointly. As a result, there is no single outcome space on which "bad events" can be union-bounded. This lack of joint measurability is the same structural feature responsible for Bell/CHSH violations in quantum mechanics [22, 19].

Basic composition for approximate DP fundamentally relies on classical probabilistic structure. Under POVM-based quantum differential privacy, this structure is absent: local indistinguishability does not imply joint indistinguishability. Consequently, classical $(\varepsilon_1 + \varepsilon_2, \delta_1 + \delta_2)$ composition has no general quantum analogue without additional assumptions.

The impossibility of basic $(\varepsilon_1 + \varepsilon_2, \delta_1 + \delta_2)$ composition for approximate QDP stems from measurement incompatibility and the absence of a joint classical probability space, rather than from a failure of tensor-product additivity at the channel level.

5 Basic Composition for Product Neighbors and Tensor-Product Channels

5.1 Setup

Let $\mathcal{H}_1, \mathcal{H}_2$ be finite-dimensional Hilbert spaces. For $i \in \{1, 2\}$, let $A_i : \mathcal{D}(\mathcal{H}_i) \to \mathcal{D}(\mathcal{K}_i)$ be quantum channels (CPTP maps), and define the tensor-product channel composition (as in Definition 5) $A_{12} := A_1 \otimes A_2 : \mathcal{D}(\mathcal{H}_1 \otimes \mathcal{H}_2) \to \mathcal{D}(\mathcal{K}_1 \otimes \mathcal{K}_2)$.

We use the POVM-based approximate quantum differential privacy definition (Definition 14).

Product-neighbor model

We say that (ρ, σ) are *product neighbors* if $\rho = \rho_1 \otimes \rho_2$, $\sigma = \sigma_1 \otimes \sigma_2$, and $\rho_i \sim \sigma_i$ for each $i \in \{1, 2\}$.

5.2 A Composition Theorem

The clean "δ-adds" basic composition proof in the classical setting relies on the fact that a joint event can be decomposed into conditional events on each release (equivalently, one can implement any test sequentially without changing the underlying probability space). In the quantum setting, this exact argument goes through *verbatim* provided the adversary's test on the joint output is *separable*. In particular, any local operations and classical communications (LOCC) test is separable. For *arbitrary* global POVMs, the corresponding statement is not generally known to hold under the POVM-based definition, and the classical proof technique does not directly apply.

▶ **Definition 17** (One-way LOCC two-outcome tests). *A two-outcome test on $K_1 \otimes K_2$ is one-way LOCC ($K_1 \to K_2$) if it can be implemented as:*

- *measure K_1 with a POVM $\{E_t\}_t$,*
- *on outcome t, measure K_2 with the POVM $\{M_{2,t}, I - M_{2,t}\}$ such that $0 \preceq M_{2,t} \preceq I$ and accept iff it accepts.*

Equivalently, its acceptance probability on product states satisfies $\mathrm{Tr}(M\, \xi_1 \otimes \xi_2) = \sum_t \mathrm{Tr}(E_t \xi_1)\mathrm{Tr}(M_{2,t} \xi_2)$.

▶ **Theorem 18** (Tensor-product composition on product neighbors against one-way LOCC tests). *Let $A_i : \mathcal{D}(H_i) \to \mathcal{D}(K_i)$ be quantum channels for $i \in \{1, 2\}$. Assume that A_i is $(\varepsilon_i, \delta_i)$-QDP: for all neighboring $\rho_i \sim \sigma_i$ and all measurement operators $0 \preceq M_i \preceq I$ on K_i,*

$$\mathrm{Tr}(M_i A_i(\rho_i)) \le e^{\varepsilon_i} \mathrm{Tr}(M_i A_i(\sigma_i)) + \delta_i.$$

Fix any product *neighboring pair $\rho = \rho_1 \otimes \rho_2$, $\sigma = \sigma_1 \otimes \sigma_2$, $\rho_i \sim \sigma_i$. Let $A_{12} := A_1 \otimes A_2$. Then for every one-way LOCC ($K_1 \to K_2$) two-outcome test on $K_1 \otimes K_2$ in the sense of Definition 17 (equivalently, for every measurement operator $0 \preceq M \preceq I$ admitting the factorization in (3) below), we have*

$$\mathrm{Tr}\big(M\, A_{12}(\rho)\big) \;\le\; e^{\varepsilon_1+\varepsilon_2} \mathrm{Tr}\big(M\, A_{12}(\sigma)\big) \;+\; e^{\varepsilon_2}\delta_1 \;+\; \delta_2. \tag{2}$$

By swapping the roles of the two subsystems (i.e., using one-way LOCC tests $K_2 \to K_1$), we also obtain the symmetric bound $\mathrm{Tr}\big(M\, A_{12}(\rho)\big) \;\le\; e^{\varepsilon_1+\varepsilon_2} \mathrm{Tr}\big(M\, A_{12}(\sigma)\big) \;+\; \delta_1 \;+\; e^{\varepsilon_1}\delta_2$. *Consequently, against one-way LOCC adversaries (in either direction) the tensor-product channel $A_1 \otimes A_2$ is $(\varepsilon_1 + \varepsilon_2, \delta_{\mathrm{comp}})$-QDP on product neighbors for $\delta_{\mathrm{comp}} := \min\{\, e^{\varepsilon_2}\delta_1 + \delta_2,\; \delta_1 + e^{\varepsilon_1}\delta_2 \,\}$.*

Proof. Let $\omega_i := A_i(\rho_i)$ and $\eta_i := A_i(\sigma_i)$ for $i \in \{1,2\}$. Then $A_{12}(\rho) = \omega_1 \otimes \omega_2$ and $A_{12}(\sigma) = \eta_1 \otimes \eta_2$.

Fix a one-way LOCC $(K_1 \to K_2)$ two-outcome test with acceptance operator $0 \preceq M \preceq I$. By Definition 17, there exist a POVM $\{E_t\}_t$ on K_1 (so $E_t \succeq 0$ and $\sum_t E_t = I$) and measurement operator $0 \preceq M_{2,t} \preceq I$ on K_2 such that for all product states $\xi_1 \otimes \xi_2$,

$$\mathrm{Tr}\big(M(\xi_1 \otimes \xi_2)\big) \;=\; \sum_t \mathrm{Tr}(E_t \xi_1)\,\mathrm{Tr}(M_{2,t}\xi_2). \tag{3}$$

Applying (3) to $\omega_1 \otimes \omega_2$ gives $\mathrm{Tr}\big(M(\omega_1 \otimes \omega_2)\big) = \sum_t \mathrm{Tr}(E_t\omega_1)\,\mathrm{Tr}(M_{2,t}\omega_2)$. For each t, since $0 \preceq M_{2,t} \preceq I$ and A_2 is $(\varepsilon_2, \delta_2)$-QDP, we have $\mathrm{Tr}(M_{2,t}\omega_2) \le e^{\varepsilon_2}\mathrm{Tr}(M_{2,t}\eta_2) + \delta_2$. Multiplying by $\mathrm{Tr}(E_t\omega_1) \ge 0$ and summing over t yields

$$\begin{aligned}\mathrm{Tr}\big(M(\omega_1 \otimes \omega_2)\big) &\le \sum_t \mathrm{Tr}(E_t\omega_1)\Big(e^{\varepsilon_2}\mathrm{Tr}(M_{2,t}\eta_2) + \delta_2\Big) \\ &= e^{\varepsilon_2}\sum_t \mathrm{Tr}(E_t\omega_1)\mathrm{Tr}(M_{2,t}\eta_2) \;+\; \delta_2 \sum_t \mathrm{Tr}(E_t\omega_1).\end{aligned} \tag{4}$$

Since $\sum_t E_t = I$ and $\mathrm{Tr}(\omega_1) = 1$, we have $\sum_t \mathrm{Tr}(E_t\omega_1) = 1$, so

$$\mathrm{Tr}\big(M(\omega_1 \otimes \omega_2)\big) \le e^{\varepsilon_2}\sum_t \mathrm{Tr}(E_t\omega_1)\mathrm{Tr}(M_{2,t}\eta_2) \;+\; \delta_2. \tag{5}$$

Define the scalars $s_t := \mathrm{Tr}(M_{2,t}\eta_2) \in [0,1]$ and define the operator $N := \sum_t s_t E_t$. Because $0 \le s_t \le 1$ and $\sum_t E_t = I$, it follows that $0 \le N \le I$. Moreover,
$\sum_t \mathrm{Tr}(E_t\omega_1)\mathrm{Tr}(M_{2,t}\eta_2) = \sum_t \mathrm{Tr}(E_t\omega_1)\, s_t = \mathrm{Tr}(N\omega_1)$. Substituting into (5) gives

$$\mathrm{Tr}\big(M(\omega_1 \otimes \omega_2)\big) \le e^{\varepsilon_2}\mathrm{Tr}(N\omega_1) + \delta_2. \tag{6}$$

Now apply $(\varepsilon_1, \delta_1)$-QDP for A_1 to the measurement operator N (valid since $0 \le N \le I$): $\mathrm{Tr}(N\omega_1) \le e^{\varepsilon_1}\mathrm{Tr}(N\eta_1) + \delta_1$. Plugging into (6) yields

$$\mathrm{Tr}\big(M(\omega_1 \otimes \omega_2)\big) \le e^{\varepsilon_2}\Big(e^{\varepsilon_1}\mathrm{Tr}(N\eta_1) + \delta_1\Big) + \delta_2 = e^{\varepsilon_1+\varepsilon_2}\mathrm{Tr}(N\eta_1) + e^{\varepsilon_2}\delta_1 + \delta_2. \tag{7}$$

Finally, apply (3) to $\eta_1 \otimes \eta_2$:
$\mathrm{Tr}\big(M(\eta_1 \otimes \eta_2)\big) = \sum_t \mathrm{Tr}(E_t\eta_1)\mathrm{Tr}(M_{2,t}\eta_2) = \sum_t \mathrm{Tr}(E_t\eta_1)\, s_t = \mathrm{Tr}(N\eta_1)$. Substituting $\mathrm{Tr}(N\eta_1) = \mathrm{Tr}(M(\eta_1 \otimes \eta_2))$ into (7) gives $\mathrm{Tr}\big(M(\omega_1 \otimes \omega_2)\big) \le e^{\varepsilon_1+\varepsilon_2}\mathrm{Tr}\big(M(\eta_1 \otimes \eta_2)\big) + e^{\varepsilon_2}\delta_1 + \delta_2$, which is exactly (2). The symmetric claim follows by repeating the same argument with the roles of the subsystems swapped. ◀

▶ Remark 19. Theorem 18 avoids any appeal to separable decompositions of M. It uses only the *defining* sequential factorization of one-way LOCC tests and the one-shot POVM-based QDP inequalities for each channel. This is the direct quantum analogue of the classical sequential/interactive proof of basic composition.

▶ **Corollary 20** (Small-ε simplification). *In the setting of Theorem 18, let* $\varepsilon_{\max} := \max\{\varepsilon_1, \varepsilon_2\}$. *Then*

$$\delta_{\mathrm{comp}} = \min\{\, e^{\varepsilon_2}\delta_1 + \delta_2,\; \delta_1 + e^{\varepsilon_1}\delta_2 \,\} \le e^{\varepsilon_{\max}}(\delta_1 + \delta_2).$$

In particular, if $\varepsilon_{\max} \le 1$, *then using* $e^x \le 1 + 2x$ *for all* $x \in [0,1]$, $\delta_{\mathrm{comp}} \;\le\; (1 + 2\varepsilon_{\max})(\delta_1 + \delta_2)$.

Proof. Since $e^{\varepsilon_2} \le e^{\varepsilon_{\max}}$ and $e^{\varepsilon_1} \le e^{\varepsilon_{\max}}$,

$$\delta_{\text{comp}} = \min\{\, e^{\varepsilon_2}\delta_1 + \delta_2,\ \delta_1 + e^{\varepsilon_1}\delta_2 \,\} \le e^{\varepsilon_{\max}}\delta_1 + \delta_2 \le e^{\varepsilon_{\max}}(\delta_1 + \delta_2),$$

and similarly using the other order. This proves the first inequality.

If $\varepsilon_{\max} \le 1$, the elementary bound $e^x \le 1 + 2x$ on $[0, 1]$ gives $e^{\varepsilon_{\max}} \le 1 + 2\varepsilon_{\max}$, hence $\delta_{\text{comp}} \le e^{\varepsilon_{\max}}(\delta_1 + \delta_2) \le (1 + 2\varepsilon_{\max})(\delta_1 + \delta_2)$. ◀

▶ Remark 21 (Relation to previous works). One can also arrive at Theorem 18 by using the general result for all joint measurements for the QDP setting in [16, Corollary III.3] using syb-additivity properties of hockey-stick divergence. Theorem 18 can also be obtained by invoking results in the Quantum Pufferfish Framework (i.e., defining potential secrets, discriminative pairs, data distributions, and measurements). Specifically, by using quasi-subadditivity of the Datta-Leditzky information spectrum divergence [8] in Proposition 2 of [25], the same result in Theorem 18 can be obtained. In this work, we provide an alternative proof that works for one-way LOCC measurements.

In this section, we obtained composition guarantees when the adversary is allowed to perform one-way LOCC two-outcome tests as in Definition 17. This can be understood as the composed mechanism satisfying a flexible private variant of QDP (also a special case of quantum pufferfish privacy in [25]). In particular, Theorem 18 shows that the composed mechanism satisfies quantum differential privacy with the neighbors defined as $\rho_1 \otimes \rho_2 \sim \sigma_1 \otimes \sigma_2,\ \rho_i \sim \sigma_i$ and $\mathcal{M}$ having measurement operators corresponding to two-outcome tests belonging to the category of one-way LOCC in Definition 17.

6 Quantum Moments Accountant

In the classical setting, the *moments accountant* tracks the log-moment generating function (MGF) of the privacy-loss random variable and exploits its additivity under independent composition. In the quantum setting, the main obstruction is that privacy loss cannot be represented as a scalar random variable prior to measurement, and taking a supremum over all POVMs destroys additivity. Nevertheless, we show that a natural non-commutative analogue of the MGF (i.e., defined using the privacy-loss operator and a Petz/exponential Rényi divergence induced by the MMGF) restores exact additivity. This yields a clean and composable quantum version of the classical moments accountant over pure DP channels.

The results we provide here are over tensor-product channels.

6.1 Quantum Privacy-Loss Operator

For neighboring states $\rho \sim \sigma$, define the *quantum privacy-loss operator*

$$L(\rho, \sigma) \coloneqq \log\big(\sigma^{-1/2}\rho\,\sigma^{-1/2}\big), \tag{8}$$

which is well defined on the support of σ. For a quantum channel A, we write $L_A(\rho, \sigma) \coloneqq L(A(\rho), A(\sigma))$. This operator is a natural non-commutative analogue of the classical privacy loss $\log \frac{P(o)}{Q(o)}$ for classical probability measures P, Q and outcome/event o.

Support convention

Throughout, when defining $L(\rho, \sigma) = \log\big(\sigma^{-1/2}\rho\,\sigma^{-1/2}\big)$, we implicitly restrict to pairs (ρ, σ) satisfying $\mathsf{supp}(\rho) \subseteq \mathsf{supp}(\sigma)$. If this condition fails, we define the corresponding privacy-loss operator and MMGF to be $+\infty$. Consequently, $\alpha_A(\lambda)$ may take the value $+\infty$ for some channels.

6.2 Matrix Moment-Generating Function

For $\lambda > 0$, quantum channel A, and states $\rho \sim \sigma$, we define the *matrix moment-generating function* (MMGF) of the privacy-loss operator by

$$\mathrm{MMGF}_A(\lambda; \rho, \sigma) \coloneqq \mathrm{Tr}\Big[A(\sigma)^{1/2}\, \exp\big(\lambda L_A(\rho,\sigma)\big)\, A(\sigma)^{1/2}\Big]. \tag{9}$$

Since $e^{\lambda L_A(\rho,\sigma)} = (A(\sigma)^{-1/2}A(\rho)A(\sigma)^{-1/2})^{\lambda}$, we can rewrite this as

$$\mathrm{MMGF}_A(\lambda; \rho, \sigma) = \mathrm{Tr}\Big[A(\sigma)\,\big(A(\sigma)^{-1/2}A(\rho)A(\sigma)^{-1/2}\big)^{\lambda}\Big]. \tag{10}$$

6.3 Quantum Moments Accountant

We now define the quantum analogue of the classical moments accountant.

▶ **Definition 22** (Quantum Moments Accountant)**.** *For a quantum channel A and $\lambda > 0$, define the quantum moments accountant (QMA) by $\alpha_A(\lambda) \coloneqq \sup_{\rho\sim\sigma} \log \mathrm{MMGF}_A(\lambda;\rho,\sigma)$, where $\mathrm{MMGF}_A(\lambda;\rho,\sigma) \coloneqq \mathrm{Tr}\big[A(\sigma)^{1/2}\exp\big(\lambda L_A(\rho,\sigma)\big)A(\sigma)^{1/2}\big]$, $L_A(\rho,\sigma) \coloneqq \log\big(A(\sigma)^{-1/2}A(\rho)A(\sigma)^{-1/2}\big)$.*

As we will discuss, Definition 22 induces an exponential Rényi-type divergence that composes additively under tensor products:

▶ **Definition 23** (Petz–Rényi divergence [28, 29])**.** *Let ρ and σ be density operators on a finite-dimensional Hilbert space $\mathcal{H}$, and let $\alpha \in (0,1)\cup(1,\infty)$. The* Petz–Rényi divergence *of order α is defined as $D_\alpha^{\mathrm{Petz}}(\rho\|\sigma) \coloneqq \frac{1}{\alpha-1}\log \mathrm{Tr}\big[\rho^\alpha\,\sigma^{1-\alpha}\big]$, with the convention that $D_\alpha^{\mathrm{Petz}}(\rho\|\sigma) = +\infty$ if $\mathrm{supp}(\rho) \not\subseteq \mathrm{supp}(\sigma)$.*

Relation to Rényi-type divergences induced by the MMGF

The quantum moments accountant is defined via the log moment generating function

$$\log \mathrm{MMGF}_A(\lambda;\rho,\sigma) = \log \mathrm{Tr}\left[A(\sigma)\left(A(\sigma)^{-1/2}A(\rho)A(\sigma)^{-1/2}\right)^{\lambda}\right].$$

It is convenient to associate to this quantity an *MMGF-induced* (exponential) Rényi-type divergence:

▶ **Definition 24** (MMGF-induced divergence)**.** *Let ρ and σ be density operators on a finite-dimensional Hilbert space $\mathcal{H}$, and let $\alpha \in (0,1)\cup(1,\infty)$. For $\alpha > 1$, the* MMGF-induced divergence *of order α is defined as*

$$D_\alpha^{\mathrm{MMGF}}(\rho\|\sigma) \;\coloneqq\; \frac{1}{\alpha-1}\log \mathrm{Tr}\left[\sigma\left(\sigma^{-1/2}\rho\sigma^{-1/2}\right)^{\alpha-1}\right], \tag{11}$$

with the convention $D_\alpha^{\mathrm{MMGF}}(\rho\|\sigma) = +\infty$ if $\mathsf{supp}(\rho) \not\subseteq \mathsf{supp}(\sigma)$.

With this notation, for $\alpha = 1+\lambda$ we have the identity
$\frac{1}{\lambda}\log \mathrm{MMGF}_A(\lambda;\rho,\sigma) = D_{1+\lambda}^{\mathrm{MMGF}}\big(A(\rho)\|A(\sigma)\big)$.

While D_α^{MMGF} composes additively under tensor products for product inputs (Section 6.4), it need not satisfy a data-processing inequality in general. To obtain an operational privacy guarantee against *arbitrary* POVMs, we instead show in Theorem 30 that bounds on the MMGF moments imply a bound on *measured* Rényi divergence, which directly captures worst-case distinguishability over all measurements:

▶ **Definition 25** (Measured Rényi divergence). *Let $\alpha > 1$. The* measured Rényi divergence *of order α is $D_\alpha^{\mathrm{meas}}(\rho\|\sigma) := \sup_{M\in\mathrm{POVM}} D_\alpha\big(P_M(\rho)\,\|\,P_M(\sigma)\big)$, where $P_M(\rho)$ denotes the classical outcome distribution induced by measuring ρ with POVM M, and $D_\alpha(\cdot\|\cdot)$ is the classical Rényi divergence of order α.*

6.4 Additivity Under Tensor-Product Composition

Let $A_1, \ldots, A_k$ act on disjoint subsystems, and let $A^{(k)} := \bigotimes_{i=1}^k A_i$. For neighboring product states $\rho = \bigotimes_{i=1}^k \rho_i$ and $\sigma = \bigotimes_{i=1}^k \sigma_i$, we have $L_{A^{(k)}}(\rho, \sigma) = \sum_{i=1}^k L_{A_i}(\rho_i, \sigma_i)$, where each summand acts on its own tensor factor.

Because exponentials of tensor-factor sums factorize, and because $\mathrm{Tr}(X \otimes Y) = \mathrm{Tr}(X)\mathrm{Tr}(Y)$, we obtain the following:

▶ **Lemma 26** (Additivity of the Matrix MGF). *For all $\lambda > 0$,* $\mathrm{MMGF}_{A^{(k)}}(\lambda; \rho, \sigma) = \prod_{i=1}^k \mathrm{MMGF}_{A_i}(\lambda; \rho_i, \sigma_i)$.

Taking logarithms and suprema yields:

▶ **Theorem 27** (Additivity of the quantum moments accountant under product neighbors). *Fix $\lambda > 0$. For a channel $\mathcal{A}$ define the* product-neighbor accountant

$$\alpha_{\mathcal{A}}^{\mathrm{prod}}(\lambda) := \sup_{\substack{\rho=\otimes_{i=1}^k \rho_i,\ \sigma=\otimes_{i=1}^k \sigma_i \\ \rho_i \sim \sigma_i\ \forall i}} \log \mathrm{MMGF}_{\mathcal{A}}(\lambda; \rho, \sigma).$$

Let $\mathcal{A}^{(k)} := \bigotimes_{i=1}^k \mathcal{A}_i$ be a tensor-product channel on disjoint subsystems. Then for all $\lambda > 0$, $\alpha_{\mathcal{A}^{(k)}}^{\mathrm{prod}}(\lambda) = \sum_{i=1}^k \alpha_{\mathcal{A}_i}(\lambda)$. Moreover, for the unrestricted *accountant $\alpha_{\mathcal{A}^{(k)}}(\lambda)$ of Definition 22, with neighbors having the same dimension as $\otimes_{i=1}^k \rho_i$, we always have the lower bound $\alpha_{\mathcal{A}^{(k)}}(\lambda) \geq \alpha_{\mathcal{A}^{(k)}}^{\mathrm{prod}}(\lambda) = \sum_{i=1}^k \alpha_{\mathcal{A}_i}(\lambda)$.*

Proof. Let $\rho = \otimes_i \rho_i$ and $\sigma = \otimes_i \sigma_i$ be product neighbors. Then $\mathcal{A}^{(k)}(\rho) = \otimes_i \mathcal{A}_i(\rho_i)$ and $\mathcal{A}^{(k)}(\sigma) = \otimes_i \mathcal{A}_i(\sigma_i)$. The privacy-loss operators add across tensor factors, hence their exponentials factorize. Using $\mathrm{Tr}(X \otimes Y) = \mathrm{Tr}(X)\mathrm{Tr}(Y)$ yields $\mathrm{MMGF}_{\mathcal{A}^{(k)}}(\lambda; \rho, \sigma) = \prod_{i=1}^k \mathrm{MMGF}_{\mathcal{A}_i}(\lambda; \rho_i, \sigma_i)$, so taking log gives additivity for each fixed product neighbor pair. Taking the supremum over product neighbors yields $\alpha_{\mathcal{A}^{(k)}}^{\mathrm{prod}}(\lambda) = \sum_i \alpha_{\mathcal{A}_i}(\lambda)$. Finally, since the unrestricted supremum is over a larger set, $\alpha_{\mathcal{A}^{(k)}}(\lambda) \geq \alpha_{\mathcal{A}^{(k)}}^{\mathrm{prod}}(\lambda)$. ◀

This mirrors the classical moments accountant exactly, but crucially *without requiring any measurement.*

▶ **Proposition 28** (Advanced composition via the quantum moments accountant (measured Rényi route)). *Let $A_1, \ldots, A_k$ be quantum channels and let $A^{(k)} := A_1 \otimes \cdots \otimes A_k$ denote their tensor-product composition. Consider product neighboring inputs $\rho = \bigotimes_{i=1}^k \rho_i$ and $\sigma = \bigotimes_{i=1}^k \sigma_i$ with $\rho_i \sim \sigma_i$. Fix $\alpha > 1$ and write $\lambda := \alpha - 1$.*

Assume that for each $i \in \{1, \ldots, k\}$ there exist parameters $\varepsilon_i \in (0, 1]$, $c_i \geq 0$, and $\alpha_i > 1$ such that for all $\alpha \in (1, \alpha_i]$ and all $\rho_i \sim \sigma_i$,

$$\log \mathrm{Tr}\Big[A_i(\sigma_i)\Big(A_i(\sigma_i)^{-1/2} A_i(\rho_i) A_i(\sigma_i)^{-1/2}\Big)^{\alpha}\Big] \leq \frac{1}{2}\varepsilon_i^2(\alpha-1)^2 + c_i(\alpha-1)^3. \tag{12}$$

Let

$$S := \sum_{i=1}^k \varepsilon_i^2, \qquad C := \sum_{i=1}^k c_i, \qquad \bar{\alpha} := \min_{i\in[k]} \alpha_i, \qquad \bar{\lambda} := \bar{\alpha} - 1.$$

Then for every $\delta \in (0,1)$ and every $\lambda \in (0, \bar{\lambda}]$ the channel $A^{(k)}$ satisfies $(\varepsilon(\lambda), \delta)$-QDP against arbitrary *POVMs, where*

$$\varepsilon(\lambda) \;\coloneqq\; \frac{S}{2}\lambda \;+\; C\lambda^2 \;+\; \frac{\log(1/\delta)}{\lambda}. \tag{13}$$

In particular, letting

$$\lambda^\star \coloneqq \sqrt{\frac{2\log(1/\delta)}{S}} \quad \textit{and} \quad \hat{\lambda} \coloneqq \min\{\bar{\lambda}, \lambda^\star\},$$

we obtain the explicit bound

$$\varepsilon(\hat{\lambda}) \;\le\; \sqrt{2S\log(1/\delta)} \;+\; \frac{2C\log(1/\delta)}{S} \;+\; \frac{S}{2}\,(\bar{\lambda} - \lambda^\star)_+, \tag{14}$$

where $(x)_+ \coloneqq \max\{x, 0\}$.

Proof. Fix $\delta \in (0,1)$ and $\lambda \in (0, \bar{\lambda}]$, and set $\alpha \coloneqq 1 + \lambda$.

Step 1: Moment additivity under tensor products (product neighbors). For each i define

$$X_i \;\coloneqq\; A_i(\sigma_i)^{-1/2} A_i(\rho_i) A_i(\sigma_i)^{-1/2} \succeq 0.$$

Since $A^{(k)} = \bigotimes_{i=1}^k A_i$ and the neighbors are products, we have $A^{(k)}(\rho) = \bigotimes_{i=1}^k A_i(\rho_i)$, $A^{(k)}(\sigma) = \bigotimes_{i=1}^k A_i(\sigma_i)$, and thus $X \;\coloneqq\; A^{(k)}(\sigma)^{-1/2} A^{(k)}(\rho) A^{(k)}(\sigma)^{-1/2} = \bigotimes_{i=1}^k X_i$. Using $(\bigotimes_i X_i)^\alpha = \bigotimes_i X_i^\alpha$ and multiplicativity of the trace,

$$\operatorname{Tr}\big(A^{(k)}(\sigma)\, X^\alpha\big) = \operatorname{Tr}\left(\bigotimes_{i=1}^k A_i(\sigma_i)\; \bigotimes_{i=1}^k X_i^\alpha\right) = \prod_{i=1}^k \operatorname{Tr}\big(A_i(\sigma_i)\, X_i^\alpha\big). \tag{15}$$

Taking logarithms and applying the bound (12) yields

$$\log \operatorname{Tr}\big(A^{(k)}(\sigma)\, X^\alpha\big) \;\le\; \sum_{i=1}^k \left(\frac{1}{2}\varepsilon_i^2\lambda^2 + c_i\lambda^3\right) = \frac{S}{2}\lambda^2 + C\lambda^3. \tag{16}$$

Step 2: Convert the moment bound to measured Rényi DP. By Theorem 30 applied to $A^{(k)}$ at order $\alpha = 1 + \lambda$, for all product neighbors $\rho \sim \sigma$,

$$D_\alpha^{\mathrm{meas}}\big(A^{(k)}(\rho) \,\|\, A^{(k)}(\sigma)\big) \le \frac{1}{\alpha - 1} \log \operatorname{Tr}\big(A^{(k)}(\sigma)\, X^\alpha\big) = \frac{1}{\lambda} \log \operatorname{Tr}\big(A^{(k)}(\sigma)\, X^\alpha\big).$$

Combining with (16) gives

$$D_{1+\lambda}^{\mathrm{meas}}\big(A^{(k)}(\rho) \,\|\, A^{(k)}(\sigma)\big) \;\le\; \frac{1}{\lambda}\left(\frac{S}{2}\lambda^2 + C\lambda^3\right) = \frac{S}{2}\lambda + C\lambda^2. \tag{17}$$

Equivalently, $A^{(k)}$ satisfies $(\alpha, \varepsilon_\alpha)$-measured Rényi DP with $\varepsilon_\alpha = \frac{S}{2}\lambda + C\lambda^2$.

Step 3: Convert measured Rényi DP to (ε, δ)-QDP. Fix an arbitrary POVM M. By definition of measured Rényi divergence, (17) implies that the induced classical distributions $P_M(A^{(k)}(\rho))$ and $P_M(A^{(k)}(\sigma))$ satisfy
$D_\alpha\big(P_M(A^{(k)}(\rho)) \,\|\, P_M(A^{(k)}(\sigma))\big) \le \varepsilon_\alpha$. Applying the standard (classical) conversion from Rényi DP to approximate DP yields that for all $\delta \in (0,1)$,

$$\Pr[M(A^{(k)}(\rho)) = 1] \le \exp\left(\varepsilon_\alpha + \frac{\log(1/\delta)}{\alpha - 1}\right) \Pr[M(A^{(k)}(\sigma)) = 1] + \delta.$$

Since $\alpha - 1 = \lambda$, we obtain $(\varepsilon(\lambda), \delta)$-QDP with $\varepsilon(\lambda) = \varepsilon_\alpha + \frac{\log(1/\delta)}{\lambda} = \big(\frac{S}{2}\lambda + C\lambda^2\big) + \frac{\log(1/\delta)}{\lambda}$, which is exactly (13).

Step 4 (Optimization). For fixed $\delta \in (0,1)$, the conversion yields the bound $\varepsilon(\lambda) := \frac{\lambda}{2} S + C\lambda^2 + \frac{\log(1/\delta)}{\lambda}$, valid for $\lambda \in (0, \bar{\lambda}]$, where $S := \sum_{i=1}^{k} \varepsilon_i^2$. Ignoring the constraint $\lambda \le \bar{\lambda}$ for a moment, the function $g(\lambda) := \frac{\lambda}{2} S + \frac{\log(1/\delta)}{\lambda}$ is minimized over $\lambda > 0$ at $\lambda^\star := \sqrt{\frac{2\log(1/\delta)}{S}}$. We therefore choose the feasible parameter $\hat{\lambda} := \min\{\lambda^\star, \bar{\lambda}\}$. Substituting $\lambda = \hat{\lambda}$ gives the valid bound

$$\varepsilon \le \frac{\hat{\lambda}}{2} S + C\hat{\lambda}^2 + \frac{\log(1/\delta)}{\hat{\lambda}}. \tag{18}$$

In particular, if $\lambda^\star \le \bar{\lambda}$ then $\hat{\lambda} = \lambda^\star$ and $\varepsilon(\lambda^\star) = \sqrt{2S\log(1/\delta)} + \frac{2C\log(1/\delta)}{S}$. If instead $\lambda^\star > \bar{\lambda}$, then $\hat{\lambda} = \bar{\lambda}$ and we simply obtain the explicit feasible bound $\varepsilon(\bar{\lambda}) = \frac{\bar{\lambda}}{2} S + C\bar{\lambda}^2 + \frac{\log(1/\delta)}{\bar{\lambda}}$. Combining these cases yields (18) with $\hat{\lambda} = \min\{\lambda^\star, \bar{\lambda}\}$. ◀

▶ **Lemma 29** (Scalar Jensen bound for operator moments). *Let $X \succeq 0$ be a positive semidefinite operator on a finite-dimensional Hilbert space and let $\tau \in \mathcal{D}(\mathcal{H})$ be a density operator. Then for every $t \ge 1$, $\big(\mathrm{Tr}(\tau X)\big)^t \le \mathrm{Tr}\big(\tau X^t\big)$.*

Proof. Let $X = \sum_j x_j \Pi_j$ be the spectral decomposition of X with eigenvalues $x_j \ge 0$ and orthogonal projectors Π_j. Define a probability distribution r on eigen-indices by $r_j := \mathrm{Tr}(\tau\Pi_j)$, so that $r_j \ge 0$ and $\sum_j r_j = \mathrm{Tr}(\tau) = 1$. Then $\mathrm{Tr}(\tau X) = \sum_j r_j x_j$, $\mathrm{Tr}(\tau X^t) = \sum_j r_j x_j^t$. Since $f(x) = x^t$ is convex on $\mathbb{R}_+$ for $t \ge 1$, Jensen's inequality gives $\left(\sum_j r_j x_j\right)^t \le \sum_j r_j x_j^t$, which is exactly the desired inequality. ◀

▶ **Theorem 30** (Quantum moments accountant ⇒ measured Rényi DP). *Fix $\alpha > 1$ and let A be a quantum channel. Define, for neighbors $\rho \sim \sigma$,*

$$X(\rho,\sigma) := A(\sigma)^{-1/2} A(\rho) A(\sigma)^{-1/2},$$

with the convention that if $\mathsf{supp}(A(\rho)) \not\subseteq \mathsf{supp}(A(\sigma))$, then the expressions below are $+\infty$. Suppose there exists a function $\alpha_A(\alpha) \ge 0$ such that for all neighbors $\rho \sim \sigma$,

$$\log \mathrm{Tr}\big(A(\sigma)\, X(\rho,\sigma)^\alpha\big) \le (\alpha - 1)\,\varepsilon_\alpha \qquad \text{for some } \varepsilon_\alpha \ge 0. \tag{19}$$

Then A satisfies $(\alpha, \varepsilon_\alpha)$-measured Rényi DP, i.e., for all $\rho \sim \sigma$,

$$D_\alpha^{\mathrm{meas}}\big(A(\rho) \,\|\, A(\sigma)\big) \le \varepsilon_\alpha.$$

Proof. Fix neighbors $\rho \sim \sigma$ and an arbitrary POVM $M = \{M_z\}_z$ on the output space. Let the induced classical distributions be $p_z := \mathrm{Tr}(M_z A(\rho))$, $q_z := \mathrm{Tr}(M_z A(\sigma))$. If $\mathsf{supp}(A(\rho)) \not\subseteq \mathsf{supp}(A(\sigma))$, then $\mathrm{Tr}(A(\sigma)X^\alpha) = +\infty$ and the claim is trivial, so assume $\mathsf{supp}(A(\rho)) \subseteq \mathsf{supp}(A(\sigma))$.

Define the positive operators $B_z := A(\sigma)^{1/2} M_z A(\sigma)^{1/2} \succeq 0$. Then $\sum_z B_z = A(\sigma)$, and moreover $q_z = \mathrm{Tr}(B_z)$, $\quad p_z = \mathrm{Tr}(M_z A(\rho)) = \mathrm{Tr}\big(B_z X(\rho,\sigma)\big)$. For each z with $q_z > 0$, define the normalized state $\tau_z := B_z/q_z \in \mathcal{D}$ so that $\frac{p_z}{q_z} = \mathrm{Tr}(\tau_z X)$. Now consider the classical Rényi divergence of order $\alpha > 1$:

$$\exp\big((\alpha-1)D_\alpha(p\|q)\big) = \sum_z p_z^\alpha q_z^{1-\alpha} = \sum_z q_z \left(\frac{p_z}{q_z}\right)^\alpha = \sum_z q_z \big(\mathrm{Tr}(\tau_z X)\big)^\alpha.$$

Apply Lemma 29 with $t = \alpha$ to each term: $\big(\mathrm{Tr}(\tau_z X)\big)^\alpha \le \mathrm{Tr}(\tau_z X^\alpha)$. Hence

$$\sum_z q_z \big(\mathrm{Tr}(\tau_z X)\big)^\alpha \le \sum_z q_z\, \mathrm{Tr}(\tau_z X^\alpha) = \sum_z \mathrm{Tr}(B_z X^\alpha) = \mathrm{Tr}\left(\sum_z B_z\, X^\alpha\right) = \mathrm{Tr}\big(A(\sigma)\, X^\alpha\big).$$

Therefore, $\exp\big((\alpha-1)D_\alpha(p\|q)\big) \le \mathrm{Tr}\big(A(\sigma)\,X^\alpha\big)$. Taking logs and dividing by $\alpha-1$ gives $D_\alpha(p\|q) \le \frac{1}{\alpha-1}\log \mathrm{Tr}\big(A(\sigma)\,X^\alpha\big)$. By the assumed accountant bound (19), the RHS is at most ε_α. Since M was arbitrary, taking the supremum over all POVMs yields $D_\alpha^{\mathrm{meas}}\big(A(\rho)\,\|\,A(\sigma)\big) = \sup_M D_\alpha\big(P_M(A(\rho))\,\|\,P_M(A(\sigma))\big) \le \varepsilon_\alpha$, which is the desired $(\alpha,\varepsilon_\alpha)$ measured Rényi DP guarantee. ◀

▶ **Corollary 31** (Composition via the quantum moments accountant (measured Rényi route))**.** *Let $A_1,\dots,A_k$ be quantum channels and let $A^{(k)} \coloneqq A_1\otimes\cdots\otimes A_k$ be their tensor-product composition. Consider product neighboring inputs $\rho=\bigotimes_{i=1}^k\rho_i$ and $\sigma=\bigotimes_{i=1}^k\sigma_i$ with $\rho_i\sim\sigma_i$. Fix $\alpha>1$ and set $\lambda\coloneqq\alpha-1$.*

Assume each A_i admits a quantum moments accountant at order α, i.e., there exists $\alpha_{A_i}(\alpha)\ge 0$ such that for all $\rho_i\sim\sigma_i$,

$$\log \mathrm{Tr}\Big[A_i(\sigma_i)\,\Big(A_i(\sigma_i)^{-1/2}A_i(\rho_i)A_i(\sigma_i)^{-1/2}\Big)^{\alpha}\Big] \;\le\; (\alpha-1)\,\alpha_{A_i}(\alpha). \tag{20}$$

*Then the composed channel $A^{(k)}$ satisfies $(\alpha,\varepsilon_\alpha)$-*measured Rényi DP *against arbitrary POVMs with*

$$\varepsilon_\alpha = \sum_{i=1}^k \alpha_{A_i}(\alpha). \tag{21}$$

Consequently, for every $\delta\in(0,1)$, the composition satisfies (ε',δ)-QDP with

$$\varepsilon' = \sum_{i=1}^k \alpha_{A_i}(\alpha) \;+\; \frac{\log(1/\delta)}{\alpha-1}. \tag{22}$$

Proof. Fix $\alpha>1$ and write $\lambda\coloneqq\alpha-1$.

Step 1: Additivity of the α-order moment bound under tensor products. Let $\rho=\bigotimes_{i=1}^k\rho_i$ and $\sigma=\bigotimes_{i=1}^k\sigma_i$ be product neighbors. Define

$$X_i \coloneqq A_i(\sigma_i)^{-1/2}A_i(\rho_i)A_i(\sigma_i)^{-1/2}, \qquad X \coloneqq A^{(k)}(\sigma)^{-1/2}A^{(k)}(\rho)A^{(k)}(\sigma)^{-1/2}.$$

Because $A^{(k)}=\bigotimes_{i=1}^k A_i$ and the inputs are products, we have $A^{(k)}(\rho)=\bigotimes_{i=1}^k A_i(\rho_i)$, $A^{(k)}(\sigma)=\bigotimes_{i=1}^k A_i(\sigma_i)$, $X=\bigotimes_{i=1}^k X_i$. Therefore, using $(\bigotimes_i X_i)^\alpha=\bigotimes_i X_i^\alpha$ and multiplicativity of the trace,

$$\mathrm{Tr}\big(A^{(k)}(\sigma)\,X^\alpha\big) = \mathrm{Tr}\left(\bigotimes_{i=1}^k A_i(\sigma_i)\,\bigotimes_{i=1}^k X_i^\alpha\right) = \prod_{i=1}^k \mathrm{Tr}\big(A_i(\sigma_i)\,X_i^\alpha\big),$$

and hence $\log \mathrm{Tr}\big(A^{(k)}(\sigma)\,X^\alpha\big) = \sum_{i=1}^k \log \mathrm{Tr}\big(A_i(\sigma_i)\,X_i^\alpha\big)$. Applying (20) term-by-term yields $\log \mathrm{Tr}\big(A^{(k)}(\sigma)\,X^\alpha\big) \le (\alpha-1)\sum_{i=1}^k \alpha_{A_i}(\alpha)$.

Step 2: Convert the moment bound to measured Rényi DP. Applying Theorem 30 at order α gives, for all product neighbors $\rho\sim\sigma$, $D_\alpha^{\mathrm{meas}}\big(A^{(k)}(\rho)\,\|\,A^{(k)}(\sigma)\big) \le \frac{1}{\alpha-1}\log \mathrm{Tr}\big(A^{(k)}(\sigma)\,X^\alpha\big) \le \sum_{i=1}^k \alpha_{A_i}(\alpha)$, which proves (21).

Step 3: Convert measured Rényi DP to (ε',δ)-QDP. Fix any POVM M. By definition of measured Rényi divergence,

$$D_\alpha\big(P_M(A^{(k)}(\rho))\,\|\,P_M(A^{(k)}(\sigma))\big) \le D_\alpha^{\mathrm{meas}}\big(A^{(k)}(\rho)\,\|\,A^{(k)}(\sigma)\big) \le \varepsilon_\alpha.$$

Applying the standard classical conversion from $(\alpha, \varepsilon_\alpha)$-Rényi DP to (ε', δ)-DP yields

$$\Pr[M(A^{(k)}(\rho)) = 1] \leq e^{\varepsilon'} \Pr[M(A^{(k)}(\sigma)) = 1] + \delta \quad \text{with} \quad \varepsilon' = \varepsilon_\alpha + \frac{\log(1/\delta)}{\alpha - 1}.$$

Substituting (21) gives (22). ◀

▶ Remark 32 (Comparison to the classical moments accountant of Abadi et al.). Proposition 28 and Corollary 31 are most naturally compared to the classical moments accountant analysis of Abadi et al. for DP-SGD.

Classical moments accountant (Abadi et al.)

In the classical setting one considers a sequence of (randomized) mechanisms and tracks the log moment generating function of the *privacy loss random variable*. Under adaptive composition, these log-moments add, and one converts the resulting moment bound to an (ε, δ)-DP guarantee via Markov's inequality, followed by an optimization over the moment order.

Quantum analogue in this work

In the quantum setting, privacy is defined operationally against *all* measurements (POVMs). A direct quantum analogue of the classical privacy loss random variable is not available prior to measurement. Instead, we work with an operator-level moment quantity: for neighbors $\rho \sim \sigma$ we form the positive operator $X(\rho, \sigma) = A(\sigma)^{-1/2} A(\rho) A(\sigma)^{-1/2}$ and track the moment functional $\mathrm{Tr}\big(A(\sigma) X(\rho, \sigma)^\alpha\big)$. Theorem 30 shows that controlling these moments implies an *operational* Rényi-type privacy guarantee, namely measured Rényi DP, which by definition quantifies worst-case distinguishability over all POVMs.

Additivity vs. post-processing

As in the classical analysis, additivity under composition is obtained at the level of moments: for tensor-product channels and product neighboring inputs, the moments factorize exactly by tensor-product identities and multiplicativity of the trace, yielding the additive accountant in Corollary 31. Unlike the classical case, post-processing by measurements is handled *inside* the privacy notion via measured Rényi divergence: the supremum over POVMs is built into D_α^{meas}, so no additional data-processing inequality is needed at this stage.

Resulting advanced-composition behavior

After converting measured Rényi DP to (ε, δ)-QDP using the standard classical Rényi-DP-to-approximate-DP conversion, one recovers an "advanced composition" tradeoff with dominant term $\sqrt{\sum_i \varepsilon_i^2 \log(1/\delta)}$ (cf. Abadi et al.). The conceptual difference is that the quantum analysis separates (i) an operator-level accounting step (moments add under tensor products) from (ii) an operational step (worst-case over POVMs), whereas in the classical setting the privacy loss is classical from the outset.

7 Advanced Composition for Quantum Differential Privacy

In this section, we provide advanced composition results for QDP mechanisms under the tensor product composition (aka parallel composition) and neighboring notion based on product neighbors (i.e., $\otimes_{i=1}^k \rho_i \sim \otimes_{i=1}^k \sigma_i \iff \rho_i \sim \sigma_i \ \forall i \in \{1, \ldots, k\}$).

For this setting, if one uses basic composition of k QDP mechanisms, we would get a composed mechanism satisfying $(k\varepsilon, 0)$-QDP, if each of them satisfies $(\varepsilon, 0)$-QDP [31, 16]. Next, we show that one can even obtain strong privacy guarantees that scale as $\sqrt{k}\varepsilon$ for sufficiently small ε, which may find use in high privacy regimes with $\varepsilon \leq 1$ and iterative protocols where the impact of k is significant.

▶ **Theorem 33.** *Let $\varepsilon_i \in [0,1]$ for all $1 \leq i \leq k$ and $\delta \in (0,1]$. Let the channel A_i satisfy $(\varepsilon_i, 0)$-QDP. Then, the parallel composition (Definition 5) of $A_1, \ldots, A_k$ (i.e., $A_1 \otimes \cdots \otimes A_k$) satisfies (ε', δ)-QDP with $\varepsilon' \coloneqq \frac{1}{2}\sum_{i=1}^{k}\varepsilon_i^2 + \sqrt{2\log\left(\frac{1}{\delta}\right)\sum_{i=1}^{k}\varepsilon_i^2}$. Furthermore, for $\varepsilon_i = \varepsilon \leq 1$ for all i, we have that $\varepsilon' = \frac{k\varepsilon^2}{2} + \sqrt{k}\varepsilon\sqrt{2\log\left(\frac{1}{\delta}\right)}$.*

Proof. Choose D_α satisfying data-processing and additivity for $\alpha > 1$ (For example, Sandwiched Rényi divergence) We also have $i \in \{1, \ldots, k\}$. A_i satisfying $(\varepsilon_i, 0)$-QDP [1] implies that we have from [25, Proposition 9, Proposition 13]

$$\sup_{\rho\sim\sigma} D_\alpha(A_i(\rho)\|A_i(\sigma)) \leq \min\left\{\frac{\varepsilon_i^2\alpha}{2}, \varepsilon_i\right\}. \tag{23}$$

Then, consider the following:

$$\sup_{\rho_i\sim\sigma_i\ \forall i} D_\alpha\left(A_1(\rho_1)\otimes\cdots\otimes A_k(\rho_k)\|A_1(\sigma_1)\otimes\cdots\otimes A_k(\sigma_k)\right) = \sup_{\rho_i\sim\sigma_i\ \forall i}\sum_{i=1}^{k} D_\alpha\left(A_i(\rho_i)\|A_i(\sigma_i)\right) \tag{24}$$

$$\leq \sum_{i=1}^{k}\min\left\{\frac{\varepsilon_i^2\alpha}{2}, \varepsilon_i\right\}, \tag{25}$$

where the equality follows by the additivity of Rényi divergence and the inequality follows from (23).

Let $0 \preceq M_x \preceq I$ (note that M_x is a measurement operator on k sub-systems), define the following: $p_x(\rho) \coloneqq \mathrm{Tr}[M_x\left(A_1(\rho_1)\otimes\cdots\otimes A_k(\rho_k)\right)]$, $q_x(\sigma) \coloneqq \mathrm{Tr}[M_x\left(A_1(\sigma_1)\otimes\cdots\otimes A_k(\sigma_k)\right)]$. Then, by the data-processing of Rényi divergence, we have that

$$\sup_{\rho_i\sim\sigma_i\ \forall i} D_\alpha\left(A_1(\rho_1)\otimes\cdots\otimes A_k(\rho_k)\|A_1(\sigma_1)\otimes\cdots\otimes A_k(\sigma_k)\right) \geq \sup_{\rho_i\sim\sigma_i\ \forall i}\sup_{\{M_x\}_x} D_\alpha(p_x(\rho)\|q_x(\sigma)), \tag{26}$$

where the second supremization is over all POVMs over k-subsystems. So we have that for all $\{M_x\}_x$ and neighboring pairs, we have

$$D_\alpha(p_x(\rho)\|q_x(\sigma)) \leq \sum_{i=1}^{k}\min\left\{\frac{\varepsilon_i^2\alpha}{2}, \varepsilon_i\right\} \leq \sum_{i=1}^{k}\frac{\varepsilon_i^2}{2}\alpha = \zeta\alpha, \tag{27}$$

where $\zeta \coloneqq \sum_{i=1}^{k}\varepsilon_i^2/2$. This is the point where we can utilize various classical procedures to obtain the desired composition result.

By following the reasoning of the proof of [20, Proposition 3] by utilizing [20, Proposition 10], we have that $p_x(\rho) \leq e^{\varepsilon_\alpha}q_x(\sigma) + \delta$ for $\delta \in (0,1)$ and $\varepsilon_\alpha \coloneqq \zeta\alpha + \frac{\log(1/\delta)}{\alpha-1}$ This holds for all $0 \leq M_x \leq I$ and all neighboring pairs, so we have the guarantees of $(\varepsilon_\alpha, \delta)$-QDP.

Note that the above analysis is valid for all $\alpha > 1$. With that the privacy parameter can be optimized by obtaining $\varepsilon' = \min_{\alpha>1}\left(\zeta\alpha + \frac{\log(1/\delta)}{\alpha-1}\right)$. Let $f(\alpha' \equiv \alpha - 1) = \zeta(\alpha'+1) + b/\alpha'$ by denoting $b \equiv \log(1/\delta)$. Then, $f'(\alpha') = \zeta - b/(\alpha')^2$. With that, the minimum is achieved at $\alpha' = \sqrt{b/\zeta}$. So that, we have $\varepsilon' = \zeta + 2\sqrt{\zeta\log\left(\frac{1}{\delta}\right)}$, and plugging that $\zeta = \sum_{i=1}^{k}\varepsilon_i^2/2$, we conclude the proof. ◀

[1] In [25], the result is shown for the Quantum Pufferfish Privacy (QPP) framework, which generalizes the QDP definition.

8 Conclusion

This work clarifies the landscape of composition guarantees for quantum differential privacy. We showed that classical composition theorems fail in full generality for POVM-based approximate QDP, due to measurement incompatibility and correlated joint channels unique to the quantum setting. At the same time, we demonstrated that these failures are not inherent to quantum channels per se, but arise from specific structural features absent in classical analysis. By restricting attention to tensor-product channels acting on product neighboring inputs, we recovered clean composition guarantees using a quantum moments accountant. Our framework separates operator-level accounting from operational privacy guarantees, enabling advanced-composition-style bounds against arbitrary measurements.

Several directions remain open. It would be valuable to understand whether variants of the quantum moments accountant can handle broader classes of channels, such as factorized but non-tensor-product compositions, or whether approximate DP guarantees can be incorporated without losing additivity. More generally, our results suggest that progress on quantum differential privacy will require careful alignment between mathematical structure and operational threat models, rather than direct transplantation of classical proofs. This work provides both technical tools and conceptual clarity for future investigations into privacy in quantum information processing.

References

1 Scott Aaronson and Guy N. Rothblum. Gentle measurement of quantum states and differential privacy. In *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing, STOC 2019, Phoenix, AZ, USA, June 23-26, 2019*, pages 322–333. ACM, 2019. doi:10.1145/3313276.3316378.

2 Martin Abadi, Andy Chu, Ian Goodfellow, H. Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. Deep learning with differential privacy. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, CCS '16, pages 308–318, New York, NY, USA, 2016. ACM. doi:10.1145/2976749.2978318.

3 Armando Angrisani, Mina Doosti, and Elham Kashefi. A unifying framework for differentially private quantum algorithms, 2023. doi:10.48550/arXiv.2307.04733.

4 Armando Angrisani and Elham Kashefi. Quantum differential privacy in the local model. *IEEE Transactions on Information Theory*, 71(5):3675–3692, 2025. doi:10.1109/TIT.2025.3552671.

5 Mario Berta, Kaushik P. Seshadreesan, and Mark M. Wilde. Rényi generalizations of the conditional quantum mutual information. *Journal of Mathematical Physics*, 56(2):022205, February 2015.

6 Hao-Chung Cheng, Christoph Hirche, and Cambyse Rouzé. Sample complexity of locally differentially private quantum hypothesis testing, 2024. doi:10.48550/arXiv.2406.18658.

7 Ayanava Dasgupta, Naqueeb Ahmad Warsi, and Masahito Hayashi. Quantum information ordering and differential privacy, 2025. arXiv:2511.01467.

8 Nilanjana Datta and Felix Leditzky. Second-order asymptotics for source coding, dense coding, and pure-state entanglement conversions. *IEEE Trans. Inf. Theory*, 61(1):582–608, 2015. doi:10.1109/TIT.2014.2366994.

9 D. Ding and M. M. Wilde. Strong converse for the feedback-assisted classical capacity of entanglement-breaking channels. *Probl. Inf. Transm.*, January 2018.

10 Cynthia Dwork, Krishnaram Kenthapadi, Frank McSherry, Ilya Mironov, and Moni Naor. Our data, ourselves: Privacy via distributed noise generation. In *EUROCRYPT*, 2006.

11 Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam D. Smith. Calibrating noise to sensitivity in private data analysis. In *TCC*, 2006.

12 Cynthia Dwork, Guy Rothblum, and Salil Vadhan. Boosting and differential privacy. In *Proceedings of the 51st Annual IEEE Symposium on Foundations of Computer Science (FOCS '10)*, pages 51–60. IEEE, 23–26 october 2010. doi:10.1109/FOCS.2010.12.

13 Farhad Farokhi. Quantum privacy and hypothesis-testing. In *2023 62nd IEEE Conference on Decision and Control (CDC)*, pages 2841–2846, 2023. doi:10.1109/CDC49753.2023.10384192.

14 Theshani Nuradha Piliththuwasam Gallage. *Theory of Privacy and Testing in a Quantum World*. PhD thesis, Cornell University, 2025. URL: https://www.proquest.com/openview/2cb580e718241481b84185ba6d289ce7/.

15 Ji Guan. Optimal mechanisms for quantum local differential privacy, 2024. arXiv:2407.13516.

16 Christoph Hirche, Cambyse Rouzé, and Daniel Stilck França. Quantum differential privacy: An information theory perspective. *IEEE Transactions on Information Theory*, 69(9):5771–5787, 2023. doi:10.1109/TIT.2023.3272904.

17 P. Kairouz, S. Oh, and P. Viswanath. The composition theorem for differential privacy. *IEEE Transactions on Information Theory*, 63(6):4037–4049, June 2017. doi:10.1109/TIT.2017.2685505.

18 D. Kifer and A. Machanavajjhala. Pufferfish: A framework for mathematical privacy definitions. *ACM Transactions on Database Systems*, 39(1):1–36, 2014. doi:10.1145/2514689.

19 A. Yu. Kitaev, A. H. Shen, and M. N. Vyalyi. *Classical and Quantum Computation*. American Mathematical Society, USA, 2002.

20 Ilya Mironov. Rényi differential privacy. In *30th IEEE Computer Security Foundations Symposium*, 2017.

21 Martin Müller-Lennert, Frédéric Dupuis, Oleg Szehr, Serge Fehr, and Marco Tomamichel. On quantum rényi entropies: A new generalization and some properties. *Journal of Mathematical Physics*, 54(12):122203, December 2013. doi:10.1063/1.4838856.

22 Michael A. Nielsen and Isaac L. Chuang. *Quantum Computation and Quantum Information*. Cambridge University Press, 2000.

23 Theshani Nuradha, Ian George, and Christoph Hirche. Non-linear strong data-processing for quantum hockey-stick divergences, 2025. doi:10.48550/arXiv.2512.16778.

24 Theshani Nuradha and Ziv Goldfeld. Pufferfish privacy: An information-theoretic study. *IEEE Transactions on Information Theory*, 69(11):7336–7356, 2023. doi:10.1109/TIT.2023.3296288.

25 Theshani Nuradha, Ziv Goldfeld, and Mark M. Wilde. Quantum pufferfish privacy: A flexible privacy framework for quantum systems. *IEEE Trans. Inf. Theory*, 70(8):5731–5762, 2024. doi:10.1109/TIT.2024.3404927.

26 Theshani Nuradha, Vishal Singh, and Mark M. Wilde. Measured hockey-stick divergence and its applications to quantum pufferfish privacy. In *2025 IEEE International Symposium on Information Theory (ISIT)*, pages 1–6, 2025. doi:10.1109/ISIT63088.2025.11195501.

27 Theshani Nuradha and Mark M. Wilde. Contraction of private quantum channels and private quantum hypothesis testing. *IEEE Transactions on Information Theory*, 71(3):1851–1873, March 2025. arXiv:2406.18651v2. doi:10.1109/TIT.2025.3527859.

28 Dénes Petz. Quasi-entropies for States of a von Neumann Algebra. *Publications of the Research Institute for Mathematical Sciences*, 21:787–800, 1985. doi:10.2977/prims/1195178929.

29 Dénes Petz. Quasi-entropies for finite quantum systems. *Reports in Mathematical Physics*, 23:57–65, 1986.

30 Tobias Rippchen, Sreejith Sreekumar, and Mario Berta. Locally-measured rényi divergences. In *2024 IEEE International Symposium on Information Theory (ISIT)*, pages 351–356, 2024. doi:10.1109/ISIT57864.2024.10619141.

31 Li Zhou and Mingsheng Ying. Differential privacy in quantum computation. In *Proceedings of IEEE Computer Security Foundations Symposium (CSF)*, pages 249–262. IEEE, 2017. doi:10.1109/CSF.2017.23.

Adaptive Garbled Circuits and Garbled RAM from Non-Programmable Random Oracles

Cruz Barnum
University of Illinois Urbana-Champaign, IL, USA

David Heath
University of Illinois Urbana-Champaign, IL, USA

Vladimir Kolesnikov
Georgia Institute of Technology, Atlanta, GA, USA

Rafail Ostrovsky
University of California Los Angeles, CA, USA

Abstract

Garbled circuit techniques secure in the adaptive setting – where inputs are chosen after a garbled program is sent – are motivated by practice, but they are difficult to achieve. Prior adaptive garbling is either impractically expensive or encrypts the garbled program with the output of a programmable random oracle (PRO). This latter approach introduces a strong model *and* incurs computational overhead.

We present a simple framework for proving adaptive security of garbling schemes in the non-programmable random oracle (NPRO) model. NPRO is a milder model than PRO, and it is close to the assumption required by the widely used Free XOR extension. Our framework is applicable to a number of existing GC techniques, which are proved adaptively secure *without modification* (and hence incurring no overhead).

As our main application, we construct and prove adaptively secure a garbling scheme for *tri-state circuits*, a model that captures both Boolean circuits and RAM programs (Heath et al., Crypto 2023). For TSC C, our garbling of C is at most $|C| \cdot \lambda$ bits long, for security parameter λ. This implies both an adaptively secure garbled Boolean circuit scheme, and an adaptively secure garbled RAM scheme where the garbling of a T-step RAM program has size $O(T \cdot \log^3 T \cdot \log \log T \cdot \lambda)$ bits.

Our scheme is concretely efficient: its Boolean circuit handling matches the performance of half-gates, and it is adaptively secure from NPRO.

2012 ACM Subject Classification Security and privacy → Cryptography; Security and privacy → Symmetric cryptography and hash functions

Keywords and phrases Adaptive Garbling, Garbled RAM, Random Oracle Model

Digital Object Identifier 10.4230/LIPIcs.ITC.2026.11

Related Version *Full Version*: `https://eprint.iacr.org/2023/1527` [4]

1 Introduction

Yao's Garbled Circuit (GC) is a cryptographic technique allowing two parties – a garbler G and an evaluator E – to securely evaluate arbitrary programs $\mathcal{P}$ on their joint private inputs [37]. GC is a foundational cryptographic primitive with various applications, especially in the fields of secure two-party computation (2PC) and multiparty computation (MPC). The technique is noteworthy because it allows 2PC and MPC protocols that use only a small constant number of rounds, and because it relies almost entirely on fast symmetric-key primitives. GC is the most efficient secure computation approach in many settings, particularly those that involve two parties; studying its power, performance, and underlying assumptions is well-motivated by both theory and practice.

Garbled RAM

Typically, the evaluated program $\mathcal{P}$ is a Boolean circuit. While Boolean circuits are powerful enough to represent any bounded function, the representation is often inefficient, in the sense that many natural programs blow up to large circuits. This is problematic because the cost of garbling typically scales linearly in the size of the circuit. The common sources of this blow-up are uses of complex looping/branching control flow and of complex data structures.

Garbled RAM (GRAM) is a GC extension that enables garbling of random access machine (RAM) programs [29]. GRAM solves the above sources of blow-up, allowing constant round protocols that handle complex programs.

[21] showed that there exists a relatively simple circuit model – tri-state circuits (TSCs) – that can efficiently emulate both Boolean circuits and RAM programs. Roughly, efficient emulation of RAM is possible because TSC gates evaluate in a data-dependent order. We construct a TSC garbling scheme that in particular handles data-dependent execution order, implying results for both garbled Boolean circuits and for garbled RAM. Our scheme's handling of Boolean circuits matches the cost of state-of-the-art half-gates garbling [38].[1]

Garbling schemes and selective security

For simplicity and modularity, GC techniques are often formalized as *garbling schemes* [6]. A garbling scheme factors evaluation of program $\mathcal{P}$ on joint secret input x into four steps:

1. Parties encode their joint input x into a garbled form $\tilde{x}$. $\tilde{x}$ is given to E.
2. G encodes the program $\mathcal{P}$ as a *garbled* program $\tilde{\mathcal{P}}$, and $\tilde{\mathcal{P}}$ is also sent to E.
3. E *evaluates* $\tilde{\mathcal{P}}$ on the garbled input, yielding garbled output $\tilde{y} \leftarrow \text{Eval}(\tilde{\mathcal{P}}, \tilde{x})$.
4. The parties *decode* the garbled output into its cleartext form y.

Of course, y should be equal to the result of simply running $\mathcal{P}(x)$ in cleartext.

Security of garbling schemes is typically considered in the so-called *selective* setting. Security against (semi-honest) corrupted G is easy, as it essentially reduces to the security of Oblivious Transfer (OT). Security against corrupted E is more detailed. Consider the following interaction between G and E:

1. G garbles $\mathcal{P}$ to obtain $\tilde{\mathcal{P}}$.
2. E sends a cleartext input x to G.
3. G sends to E the garbled input $\tilde{x}$, the garbled program $\tilde{\mathcal{P}}$, and information d needed to decode the output.

Security against a corrupted E is proved by considering this interaction and constructing a simulator that – from the program output y alone – can forge a garbled program, garbled input, and decoding information such that E cannot tell whether they are interacting with G or with the simulator. Existence of such a simulator proves that E learns *nothing beyond* y from GC evaluation.

The crucial detail of the above interaction is that E must select its input x *before* it sees the garbled program $\tilde{\mathcal{P}}$.

Adaptive security

Transmission of the garbled program $\tilde{\mathcal{P}}$ is the main bottleneck of GC. One common practical mitigation is to move GC generation and transmission to the *offline* (or *preprocessing*) phase.

[1] [35] uses less communication than [38], but it uses significantly more computation. We consider both techniques state-of-the-art.

In this way, we can do most of the work "overnight", before inputs are ready. Once the parties obtain their inputs, they enter the *online* phase and quickly compute the desired output.

At first glance, garbling schemes seem ideal for the offline/online setting. Indeed, G can simply garble the program and send $\tilde{\mathcal{P}}$ in advance; then, once inputs become available, G quickly conveys garbled inputs to E, who evaluates $\tilde{\mathcal{P}}$ on $\tilde{x}$ and learns the program output. The security of such usage is *not implied* by our selective security game, so we need an updated game, where E chooses the input x after $\tilde{\mathcal{P}}$ is sent over. This is the *adaptive security* game:

1. G garbles $\mathcal{P}$ and sends $\tilde{\mathcal{P}}$ to E.
2. E sends a cleartext input x to G.
3. G sends $\tilde{x}$ to E, as well as information d needed to decode the output.

Pushing transmission of $\tilde{\mathcal{P}}$ to the offline phase requires that the GC scheme is secure in the context of this game.

Constructing garbling schemes that provably achieve this second notion is *notoriously difficult*. This difficulty arises from the fact that E's input can depend on the garbled program itself. Proving this secure is a significant challenge.

Cost accounting for adaptive GC schemes

When constructing adaptively-secure GC, we consider the cost of the offline and the online phases separately. The crucial metrics are the offline and online communication costs. Ideally, offline phase communication should be close to the cost in the selective security setting. Thus, we ideally want a scheme whose offline communication is equal to the size of the underlying circuit, multiplied by security parameter λ. In the online phase, we wish to pay online in terms of the number of input/output bits of the program.

Computation overhead is, of course, also important, and the computation used by both G and E should ideally be almost identical to their computation in the selective security setting.

Summary of state-of-the-art adaptive GC

The insecurity of standard GC (or, more precisely, the invalidity of existing GC selective-security proofs) when x may depend on $\tilde{\mathcal{P}}$ was first observed by [5]. A number of solutions were proposed, which we review in detail in Section 1.3. Here, leading up to Section 1.1 we highlight two main approaches:

One, based only on one-way functions, requires high computational overhead (multiplicative factor $O(w)$, where w is the width of the evaluated circuit) both in online and offline phases [22]. This effectively negates the benefit of offline transmission in many settings.

The second is far more efficient, simply requiring to XOR the transmitted circuit with an output of a random oracle (RO), as described by [5]. This both requires modification to the selectively secure scheme, and it roughly doubles computational cost. It would be far preferable to obtain adaptive security without increasing computation and, indeed, without modifying implementation. In addition the [5] proof requires that the simulator *program* the RO. This is a strong model, which cannot be met by any fixed function, and which is widely seen as much stronger than a non-programmable RO.

1.1 Our Contribution

Garbled circuit adaptive security is a well motivated and intensely studied problem. As we discuss in Section 1.2, the current state of the art offers to practitioners an unsatisfying menu of options when confronted with the need to use GC in the adaptive setting.

Many practical GC techniques are selectively secure in the non-programmable random oracle (NPRO) model. As our main contribution, we provide a framework that allows one to prove that such schemes are also *adaptively* secure, if they meet two conditions. These are satisfied by existing standard schemes, including Free XOR, half-gates, three-halves gates [35], and even arithmetic techniques [3, 19]; see Appendix C in the full version. Our conditions require that (1) any RO queries issued while garbling the circuit are hidden by the resulting garbled circuit and (2) one can *resample* keys associated with a particular GC, such that the GC still evaluates correctly with these fresh keys – the scheme should be *rekeyable*. We highlight that these conditions are reasonably easy to prove, which would facilitate the adoption of our framework and of the final protocols.

We apply our framework to the tri-state circuit model by (1) constructing a natural NPRO-based garbling scheme and (2) using our framework to prove this scheme achieves adaptive security. Thus, we obtain adaptively-secure garbling of both Boolean circuits and RAM programs in the NPRO model (NPROM).

Our adaptively-secure TSC scheme matches the cost of state-of-the-art selectively secure schemes in terms of both communication and computation. Let C denote a TSC with input x and output y. Let λ be the security parameter, which can be understood as the length of encryption keys (e.g. 128 bits). Our offline communication cost is $\leq |C| \cdot \lambda$, where the actual cost depends on the types of gates used. Our online communication cost is $O((|x| + |y|) \cdot \lambda)$, and is independent of the size of the program. In terms of computation, both G and E expend at most one call to the random oracle per tri-state gate.

When we compile Boolean gates to tri-state gates, our scheme's costs match the cost of the popular selectively-secure half-gates garbling scheme [38] in terms of both computation and communication. [38] is a state-of-the-art Boolean scheme[2], so our TSC-based Boolean handling matches what one would expect from standard Boolean circuit garbling. Additionally, T steps of a RAM program can be compiled to a TSC with $O(T \cdot \log^3 T \cdot \log \log T)$ gates [21].

We emphasize that while RO-based proofs are sometimes straightforward, proving unmodified GC schemes adaptively secure remains surprisingly difficult. Our work proves many such schemes secure, and it provides a framework by which to show security for future schemes. To our knowledge, no other works show adaptive GC from RO, except see Section 1.3's discussion of [5, 18].

1.2 The Practical Importance of Adaptive Security

We feel that prior to our work, the options for practical GC deployment were unsatisfying, and the system was brittle. Practitioners had three options:

Option 1: Obey selective security; perform all work in the online phase

Even in settings where this is acceptable from the perspective of engineering/performance, delaying all work to the online phase is an undesirable constraint that is prone to be inadvertently violated. Envisioning larger systems incorporating MPC/GC, even implemented

[2] The more recent [35] scheme uses less communication than [38], but it uses more computation.

and maintained by a knowledgeable team, it is easy to foresee the temptation to modularize, optimize and multi-thread execution, separating GC generation/transmission from OT execution, eventually leading to order violation.

Relatedly, garbled circuit is a simple, powerful, and convenient object for standardization. There is already a preliminary effort by NIST to standardize threshold schemes, including more complex objects such as GC [32, 33]. Following discussion at the MPTS workshops, it seems impractical to standardize many possible combinations of GC and OT. Rather, GC, OT, and other related primitives are likely to be standardized separately. A more robust, versatile, and resilient GC primitive would be much preferable for standardization than a brittle one, subject to execution order constraints.

Option 2: Use a programmable random oracle (PRO) to mask GCs

Namely, use the PRO-based technique of [5]. This option roughly doubles the total computation cost, both for G and E, compared to selectively-secure GC, and to our solution. Perhaps worse than this increased cost is the fact that it requires modification to the selectively-secure GC scheme. Far more attractive would be to simply show that existing schemes are, in fact, adaptively secure, if the hash function is sufficiently strong; this is what we achieve here.

The approach of [5] also assumes programmability of the RO (PRO). PRO is an unusually strong model that violates several impossibility results, e.g., enabling non-interactive non-committing encryption [31] and adaptive GC with online phase independent of the program output size [5]. Both are impossible in the standard model [2, 31]. In contrast, NPRO is a milder model, which is widely used in practical cryptography. For instance, the standardized RSA-OAEP encryption scheme uses NPRO [7, 9, 10, 30]. Notably, in the GC setting, the widely used Free-XOR key homomorphism relies on circular correlation robustness [8], which can be understood as a weakening of the NPRO model.

Option 3: Implement adaptive GC in the standard model

While standard model adaptive GC remains of theoretical interest, the best known technique incurs multiplicative factor w overhead in terms of computation, where w is the width of the evaluated circuit. In many practical settings (e.g., laptops on the 1Gbps LAN), the speed of GC generation/evaluation is only about $3\times$ faster than transmission. In this scenario, the online phase of the adaptively secure GC will be factor $\approx w/3$ times *slower* than the entire selective GC evaluation.

Improving adaptive GC from just a PRF remains a challenge, and any results that improve the factor w overhead would be highly surprising.

1.3 Related Work

Selective GC Security

Even proofs of selective GC security are subtle. The classic proof of selective security from a PRF [28] proceeds by a hybrid argument where in each step we replace one real gate by a *simulated* gate. This simulated gate is programmed such that it outputs a value consistent with real-world evaluation. Once each gate is replaced, the final distribution is statistically close to simulation, which does not depend on the real-world input x.

One subtle, but central, aspect of this simulation is that we must replace gates in a specific order. This is so that we can base the indistinguishability of each hybrid on the PRF assumption. Indeed, PRF keys are used throughout the circuit, but PRF security only holds if the key had not been used elsewhere.

Jumping ahead, we remark that in the adaptive setting [28]'s intermediate simulated gates should output values that depend on the input x, but syntactically, x simply is not well defined at the time the simulated gate should be programmed. Prior works resolve this problem, but at significant cost.

Adaptive Garbled Circuits

[5] were the first to thoroughly investigate the problem of adaptive GC. They pointed out that existing schemes do not seem to admit a proof of adaptive security, and they gave two constructions that *are* adaptively secure. Their first construction should mostly be viewed as a proof of concept. It requires that G one-time pad the GC $\tilde{C}$ before sending it to E. Then, in the online phase, G sends the one-time-pad mask to E, allowing E to decrypt the circuit and evaluate normally. In terms of online cost, this construction is poor, as it requires that G send a message proportional to the GC.

This said, [5]'s one-time-pad-based construction does give important insight into *how* adaptivity can be achieved. In short, the one-time-pad mask allows G to *equivocate* the GC. Namely, G can unmask to E a (different) GC that *depends on E's choice of input x*. This capability is, of course, not used in the real-world execution, but it *is* used by the simulator. Namely, in intermediate steps of the proof, G uses its ability to equivocate to open to E intermediate hybrid garbled circuits from the *selective* security proof [28]. In this way, the one-time-pad-based scheme admits a natural proof of adaptive security.

[5] also constructed a scheme that they proved secure by using programmable RO. In short, the simulator programs the RO to equivocate the GC, similar to the above. As already noted, this scheme circumvents a known lower bound on online communication cost of adaptively secure GC [2]. In particular, [2] showed that any standard model adaptive garbling scheme must have an online phase scaling at least with the program's output, but [5]'s RO construction only sends information proportional to the program's *input*. In contrast, our simulator *does not* program the RO; in the NPROM, we cannot circumvent the [2] lower bound.[3]

In concurrent work, [18] consider adaptive garbling of specific implementations of popular half-gates and three-halves schemes. They show that these schemes, instantiated with (previously designed) specific encryptions built on the random permutation model [17], achieve adaptive security. Their proof is focused on specific details of encryption based on fixed-key AES, and accordingly their proof details are intricate. In contrast, we aim to develop a framework that may be useful in large class of GC constructions. Our simulation methods are less customized and more general. Our framework applies to garbling of TSCs, and thus our results immediately imply an adaptively secure garbled RAM scheme in the NPROM, which was not shown by [18].

Other Works on Adaptive GC

A number of other works made progress on adaptive GC, including by giving surprising, but expensive, solutions that assume only the existence of one-way functions [22], showing security of schemes for low-depth circuits [24, 23], lower-bounding the security of certain techniques in the adaptive setting [25, 1], and achieving adaptive security from heavy-weight public-key-style methods [14, 22]. The prior work section in the full version discusses these at greater length.

[3] If the decoding table is given in the online phase in the adaptive scheme from [5], then one can show that their scheme is adaptively secure in the NPROM. We emphasize that [5] still prescribes an RO mask on every ciphertext, which our analysis avoids.

Garbled RAM (GRAM)

GRAM [29] upgrades GC to handle RAM programs rather than circuits. In short, GRAM allows the GC to perform oblivious random access to a main memory where each access incurs amortized sublinear cost. Ideally, the per-access overhead should be at most polylogarithmic.

Early GRAM schemes, e.g. [12, 13, 15, 29], demonstrated feasibility results, but they were not concerned with polylog performance factors, so their constructions are expensive. More recent constructions [20, 21, 34, 16] target performance. The best symmetric-key result [21] garbles only $O(\log^3 n \cdot \log\log n)$ fan-in-two gates per access.

More interesting than [21]'s cost is its formalism. [21] shows that RAM computation can be emulated by a relatively simple circuit model called *tri-state circuits* (TSCs). To garble RAM, it suffices to garble TSCs [21]. Our result leverages the TSC model to achieve adaptively secure GRAM. We review the TSC model in Section 3.

2 Technical Overview

This section explains our approach at a high level, providing sufficient detail for informal understanding. Section 4 in this work and Section 5 in the full version of the paper formalize the ideas explained here.

Note that we prove adaptive security of garbled tri-state circuits (TSCs) because the TSC model is more general (and useful – it supports efficient RAM implementation) than Boolean circuits. For those unfamiliar, we explain the TSC model in Section 3.2. Our proofs and proof intuition apply and can be read and understood in the narrower context of Boolean circuits.

2.1 Tri-State Circuit Construction

Our main contribution is a framework for proving adaptive security of garbling schemes in the NPRO model. To make this contribution concrete, we formalize a particularly useful garbling scheme, and prove it fits into our framework. Our scheme garbles the tri-state circuit (TSC) model; see Section 3.2.

In short, a TSC includes three types of gates, and the gates execute in a data-dependent order. This data-dependent execution is powerful enough to support efficient emulation of RAM programs.

Wire keys

Our TSC handling starts with Free-XOR-based wire keys [26]. Namely, to garble a TSC C, the garbler G samples for each circuit wire w a length-λ key k_w^0. This key encodes a logical zero on wire w. G then samples a single length-λ global correlation Δ, and for each wire w, G defines the encoding of logical one as $\mathsf{k}_w^1 = \mathsf{k}_w^0 \oplus \Delta$ Note that this means that if we overload the name of a wire with its runtime value, at runtime E holds the following key:

$$\mathsf{k}_w = \mathsf{k}_w^0 \oplus w \cdot \Delta$$

TSCs also allow wires to hold a value $\mathcal{Z}$. We encode $\mathcal{Z}$ by E holding *no key*.

Gate handling

The function of each TSC gate-type was explained in Section 3.2. Roughly, our garbling of gates is as follows:

- **XOR gates** are handled simply by XORing the input labels. This is the Free XOR optimization [26].
- **Buffers** of the form $z \leftarrow x/y$ have two inputs: a *control* wire y and a *data* wire x. G defines the key for the buffer's output wire as follows:

  $$\mathsf{k}_z^0 = \mathsf{k}_x^0 \oplus \mathcal{O}(\mathsf{k}_y^1, \mathit{gid})$$

  Here *gid* is a nonce. This use of RO ensures that E can compute a key for the output wire iff the control wire y holds logical one.
- **Joins** of the form $z \leftarrow x \bowtie y$ connect together the inputs x and y such that if either wire is non-$\mathcal{Z}$, the output wire acquires the non-$\mathcal{Z}$ input value. G handles joins by simply setting the output key as $\mathsf{k}_z^0 = \mathsf{k}_x^0$. This trivially enables E to translate an x key to a z key. To allow E to translate a y key to a z key, G includes in the GC the particular string $\mathsf{k}_x^0 \oplus \mathsf{k}_y^0$. E XORs this difference with its y key to obtain an appropriate z key.

We refer the reader to Section 5 in the full version for further details on our TSC handling.

2.2 Proof of Security

Our main contribution shows that typical GC schemes built from NPRO are also *adaptively secure*. For example, our above basic TSC scheme is adaptively secure with no change in implementation. Our framework for proving RO-based schemes adaptively secure requires two properties of the GC scheme: (1) the scheme should be *rekeyable* and (2) the scheme should be *query hiding*. In the following, we explain these properties in the context of our TSC scheme.

On our (non-)use of programming

Recall from Section 3.3 that the NPRO model constrains the relationship between interactive Turing machines $\mathcal{S}$ and $\mathcal{A}$. Namely, the simulator $\mathcal{S}$ may not program responses to $\mathcal{A}$'s RO queries.

In our security proof, we perform a standard real/ideal comparison, where we define hybrid distributions bridging the two worlds. In these hybrid worlds, we reason about the content of the random oracle's truth table by "programming" it. For instance, we reason that a certain interaction with a true random oracle is statistically close to one with the same oracle, but where some rows of the truth table have been replaced by fresh randomness.

One natural question is to consider whether or not this use of "programming" should be allowed in the NPROM.

We emphasize that this use of "programming" is *formally* in the NPROM [9]. The model simply constrains that the *simulator* cannot program the oracle. As a proof technique, we introduce hybrid worlds, which – like the real and ideal worlds – are simply descriptions of the output of some program (the adversary). We introduce these hybrid thought experiments only as a means by which to compare and relate two objects – the output of $\mathcal{A}$ in the real and ideal experiments. The tools we use to compare these objects are immaterial. Our definition of security is formalized with respect to an RO that cannot be programmed. *This alone* defines the security properties of our scheme, and the method by which we prove real-ideal indistinguishability is irrelevant.

Indeed, the NPROM can be understood as insisting that we can heuristically instantiate the RO in *both* the real world and the ideal world, and it is in this sense in which the RO is non-programmable – if the RO were programmed, it would not be instantiable by any pre-defined hash function. But of course we can *in a thought experiment* reason about $\mathcal{A}$'s

output in the context of that (heuristically assumed random) truth table; a part of that reasoning is considering what would happen if we were to replace some truth table rows by fresh randomness.

This style of using programming inside hybrids is analogous techniques in the universal composability (UC) framework. In the UC framework, one cannot "rewind" the adversary (aka environment), but some works have found value in rewinding the adversary in the context of hybrid experiments [11, 27].

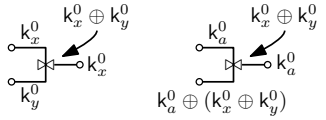
Appendix A in the full version provides more discussion regarding NPRO.

Rekeying a Garbled Circuit

In a standard model GC proof of *selective* security, we would use a hybrid argument to rewrite parts of the GC to "hard-code" its behavior, forcing GC gates to output keys consistent with evaluation under the circuit input x; see e.g. [28]. In the *adaptive* setting, such a hybrid argument is impossible: at the time $\mathcal{A}$ receives the GC, the input x is not defined.

Our proof of adaptive security observes that while we cannot use a hybrid argument to change the GC, we *can* change the *keys* associated with the GC. Consider garbled circuit $\tilde{C}$, and let K be the collection of wire keys chosen by G while garbling $\tilde{C}$. In our TSC construction – and in many RO-based GC schemes – it is possible to choose a fresh collection of keys K' that are independent of K and that preserve circuit semantics when executed with garbling $\tilde{C}$. We call this process of replacing GC keys a *rekeying* of the GC.

We next show how to rekey a TSC join gate (defined in Section 2.1), a gate that is more primitive than Boolean AND/OR. In particular, a Boolean AND gate can be efficiently implemented from a small number of TSC gates, including two joins [21]. In the resulting garbled AND, G and E take the same actions as if they had used the half-gates AND gate construction [38]. The rekeying of a single join gate can be understood as a subprocedure of a rekeying of the more complex Boolean AND gate. (In TSC-based RAM, joins are also used outside the construction of Boolean AND gates.) Consider the following join:



On the left, we depict a join as garbled by G, where input wires are labelled by G's keys $\mathsf{k}^0_x, \mathsf{k}^0_y$; the output is labelled by k^0_x. To enable evaluation in all cases, G includes in $\tilde{C}$ the string $\mathsf{k}^0_x \oplus \mathsf{k}^0_y$. If E only holds a runtime key k_y for the bottom input, it can use this string to *translate* that input key to an appropriate output:

$$(\mathsf{k}^0_y \oplus y \cdot \Delta) \oplus (\mathsf{k}^0_x \oplus \mathsf{k}^0_y) = \mathsf{k}^0_x \oplus y \cdot \Delta$$

We start rekeying this gate by replacing its top input key k^0_x with some freshly sampled key k^0_a. Similarly, we replace the output wire key by k^0_a. To complete the rekeying, we must ensure the semantics of the gate are preserved. Namely, if E obtains some key on the bottom wire, it should be able to *translate* that input key to an appropriate output key. To ensure this works, we rekey the bottom wire as well, replacing k^0_y by a key that is specifically chosen to preserve semantics: $\mathsf{k}^0_a \oplus (\mathsf{k}^0_x \oplus \mathsf{k}^0_y)$. Thus, if E obtains a key on the bottom input wire y, it can use the GC string to appropriately translate to an output key:

$$((\mathsf{k}^0_a \oplus (\mathsf{k}^0_x \oplus \mathsf{k}^0_y)) \oplus y \cdot \Delta) \oplus (\mathsf{k}^0_x \oplus \mathsf{k}^0_y) = \mathsf{k}^0_a \oplus y \cdot \Delta$$

We can prove that this rekeying of the gate is indistinguishable from the original keying of the gate, and by extending this strategy we can replace *all* GC keys.

The benefit of rekeying is that it allows us to define security properties for *fixed* garbled circuits. The standard definitions of selective GC security [6] consider distributions of garbled circuits. For instance, standard GC obliviousness roughly states that a randomly garbled circuit should be indistinguishable from the output of a simulator. Our notion of rekeying allows us to instead *quantify* over GCs. For instance, we can state that for a particular GC, the rekeying of *that* GC should be indistinguishable from some other distribution. This shift from probabilistically-defined GCs to universally quantified GCs is critical, because it allows us to make meaningful claims about $\mathcal{A}$'s ability to distinguish in the adaptive game's *online* phase, when the GC is, indeed, fixed.

Query hiding

Recall that we use a RO $\mathcal{O}$ to construct a garbled circuit $\tilde{C}$. Then, in the adaptive security game's *offline* phase, $\mathcal{A}$ obtains $\tilde{C}$ and is allowed access to the same RO $\mathcal{O}$. One concern is that $\mathcal{A}$ might *guess* a RO query that intersects with queries issued when garbling. If this happens, then $\mathcal{A}$ might, for instance, be able to partially decrypt $\tilde{C}$, and use this side information to choose an input x that in the later *online* phase helps it distinguish real from ideal.

To show security, we therefore must show that such a guess is unlikely, in the sense that the garbled circuit $\tilde{C}$ does not "help" $\mathcal{A}$ to guess problematic queries. More precisely, we formalize a property – query hiding – whereby $\mathcal{A}$ should not be able to detect when we *remove* from the random oracle all queries issued by Garble; see Section 4. At the very highest level, we show that our TSC garbling scheme is query-hiding due to the fact that all keys are uniformly chosen, and not included in the GC itself. By plugging together the ability to rekey the garbled circuit with query hiding, we are able to obtain a proof of adaptive security.

Proof intuition

In short, our security proof combines query hiding and rekeying. By applying query hiding, we show that $\mathcal{A}$'s chosen input x cannot depend on the content of the RO. However, it still might depend on the garbled circuit itself. Rekeyability then decouples the garbled circuit from its wire keys and the RO, allowing us to argue that we can replace $\mathcal{A}$'s chosen input x by, say, the all zeros string, without $\mathcal{A}$ noticing. This ultimately leads to a complete proof.

3 Preliminaries

3.1 Notation

- λ is a security parameter and can be understood as the length of GC labels.
- $x \approx y$ denotes distributions x and y are computationally indistinguishable.
- $x \overset{s}{\approx} y$ denotes distributions x and y are statistically close, i.e. indistinguishable to an unbounded adversary.
- $x \equiv y$ denotes that distributions x and y are identical.
- $\mathcal{O}$ denotes a (non-programmable) random oracle.
- We refer to wire id w. When clear from context, we will *overload* w to also mean the plaintext value on that wire.
- Whenever $\mathcal{A}$ appears more than once in a game, assume $\mathcal{A}$ saves state.

3.2 Garbled RAM and Tri-State Circuits

We provide a framework for achieving adaptive security from NPRO. As part of this contribution, we construct a scheme that captures much of the recent advances in practical GC, and we prove this scheme's security in our framework.

[21] formalized a model of computation called *tri-state circuits* (TSCs). TSCs are interesting for garbling because there exists an efficient (polylog overhead) reduction from RAM programs to the TSC model. The TSC model is straightforward to garble, and hence TSCs lead to natural constructions of Garbled RAM [29]. Our presented garbling scheme handles TSCs. Thus, we review relevant definitions. All definitions in this section are adapted from [21].

▶ **Definition 1** (Tri-state Circuit). *A **tri-state circuit** (TSC) is a circuit allowing cycles (i.e., its graph need not be acyclic) with three gate types: XORs ($\oplus$), buffers ($/$), and joins ($\bowtie$). Each wire carries a value in the set $\{0, 1, \mathcal{Z}, \text{✗}\}$. The semantics of each gate type are as follows:*

$\oplus$	$\mathcal{Z}$	0	1	✗
$\mathcal{Z}$	$\mathcal{Z}$	$\mathcal{Z}$	$\mathcal{Z}$	✗
0	$\mathcal{Z}$	0	1	✗
1	$\mathcal{Z}$	1	0	✗
✗	✗	✗	✗	✗

$/$	$\mathcal{Z}$	0	1	✗
$\mathcal{Z}$	$\mathcal{Z}$	$\mathcal{Z}$	$\mathcal{Z}$	✗
0	$\mathcal{Z}$	$\mathcal{Z}$	0	✗
1	$\mathcal{Z}$	$\mathcal{Z}$	1	✗
✗	✗	✗	✗	✗

$\bowtie$	$\mathcal{Z}$	0	1	✗
$\mathcal{Z}$	$\mathcal{Z}$	0	1	✗
0	0	0	✗	✗
1	1	✗	1	✗
✗	✗	✗	✗	✗

We assume each gate has some distinct gate ID gid. A TSC has n input wires and m output wires. Circuit execution on input $x \in \{0,1\}^n$ proceeds as follows: (1) Store $\mathcal{Z}$ on each non-input wire, (2) store x on the input wires, (3) repeatedly and arbitrarily choose some gate g and update g's output wire according to g's function and input wires. Once there remains no gate whose execution would change a wire, halt and output the state of the circuit output wires.

Buffer gates act as "switches". Each buffer x/y has two inputs: a *control wire* y and a *data wire* x. If the control wire y holds one, then the buffer "closes", connecting its data wire x to its output; if the control wire holds zero, the buffer remains open, and the output wire is unassigned. Join gates allow us to connect wires together. For instance, we can connect the output of two buffers such that the joined output wire takes the value of whichever buffer is closed.

The tri-state value $\mathcal{Z}$ roughly denotes the idea "this wire does not have a value" and the value ✗ denotes "an error has occurred". In this work we consider a natural restriction of TSCs which requires that (1) no errors occur and (2) every wire ultimately acquires a Boolean value:

▶ **Definition 2** (Total Tri-State Circuit). *A tri-state circuit C is **total** if on every input x, the following holds. Change the semantics of joins such that they are* multidirectional[4]*. To execute gate $z \leftarrow x \bowtie y$, update the value of* each *wire x, y, z with joined value $(x \bowtie y \bowtie z)$. C is **total** if after completing circuit execution with these semantics, every circuit wire is assigned a Boolean value.*

The interesting capability of TSCs is that gates execute in data-dependent orders. This capability is precisely what enables efficient RAM emulation. To take advantage of this in the GC setting, we must inform the GC evaluator E of the order they should execute gates.

[4] i.e. if one input or output is boolean, propagate the boolean value to each other wire, even if the output propagates to the input wires.

[21] show that to make this work, it suffices to reveal to E every buffer control wire. These (plaintext) control bits allow E to determine which gates are ready for evaluation. G reveals control bits by including in the GC one extra bit per control wire.

Revealing control wires to E complicates simulation of E's view. We must somehow argue that even though E learns all control wires, we can still simulate. [21] solve this by extending TSC input to additionally include randomness. The random part of the input can be used as masks on control bits. An oblivious TSC guarantees that if randomness r is properly sampled, the circuit correctly evaluates with overwhelming probability. When the oblivious TSC is garbled, randomness r is sampled locally by G and included as part of his input to the circuit. With the addition of masks and careful circuit design, particular tri-state circuits can then be shown to be *oblivious*, i.e. that the control wire values can be simulated. We garble oblivious TSCs, so we give the relevant definitions:

▶ **Definition 3** (Oblivious Tri-state Circuit)**.** *A **randomized tri-state circuit** is a pair consisting of a tri-state circuit C and a distribution of bitstrings D. The execution of a randomized tri-state circuit on input x is defined by randomly sampling a string r from D, then running C on x and r:*

$$(C, D)(x) = C(x; r) \qquad \text{where } r \leftarrow\!\!\$\ D$$

*Let C be a tri-state circuit with input $x \in \{0,1\}^n$. The **controls of C on x**, denoted* $\text{controls}(C, x) \in \{0,1\}^*$*, is the set of all buffer control wire values (each labeled by its gate ID) after executing $C(x)$. Let $\{(C_i, D_i) : i \in \mathbb{N}\}$ denote a* family *of randomized tri-state circuits. The family is considered **oblivious** if for any two inputs $x, y \in \{0,1\}^n$ the following holds:*

$$\{\ \text{controls}(C_\sigma, (x; r))\ \mid\ r \leftarrow\!\!\$\ D_\sigma\ \} \overset{s}{\approx} \{\ \text{controls}(C_\sigma, (y; r))\ \mid\ r \leftarrow\!\!\$\ D_\sigma\ \}$$

3.3 Definition of Non-Programmable Random Oracle Model

We consider security in the non-programmable RO model (NPROM), as defined by [9]. The NPROM specifies a constraint on formal reductions from one interactive Turing machine to another. In our case, it constrains the relationship between our ideal-world simulator $\mathcal{S}$ and the real-world adversary $\mathcal{A}$.

Formally, the adversary $\mathcal{A}$ interacts with the RO by writing queries to/reading responses from its oracle tape. In the *programmable* RO model's ideal world, the simulator controls this tape, such that it may read $\mathcal{A}$'s queries and arbitrarily choose RO responses – namely, it may program the RO. In the *non-programmable* RO model, even in the ideal world, $\mathcal{A}$'s tape is instead connected to an externally defined oracle $\mathcal{O}$, and $\mathcal{O}$ responds directly.

The informal rationale underlying the NPROM is that it more closely resembles the setting where we heuristically instantiate RO with an externally-defined hash function. From a philosophical point of view, the simulator $\mathcal{S}$ can be viewed as an explanation of the interaction observed by the real-world adversary. In the NPROM, we, as the designers of $\mathcal{S}$, must explain this interaction in the context of a hash function that we cannot control. This seems to more closely resemble real life than the PROM because we cannot control, e.g., the definition of SHA-3.

More formally, the NPROM is motivated by a separation between it and the PROM, as there exist constructions that are (1) possible in the PROM and (2) impossible in both the standard model and NPROM, suggesting that the NPROM is closer to reality. Our results also hold for an even weaker model called the Auxiliary Input ROM [36]. This model and proof that our scheme is compatible can be found in Appendix D in the full version.

3.4 Garbling Schemes

We consider security for *garbling schemes*, as defined by [6]:

▶ **Definition 4** (Garbling Scheme [6]). *A **garbling scheme** for a class of circuits $\mathcal{C}$ is a tuple of four procedures $(Garble, Encode, Eval, Decode)$:*

- $Garble(1^\lambda, C) \to (\tilde{C}, e, d)$*: Garble a circuit $C \in \mathcal{C}$, producing garbled circuit $\tilde{C}$,* input encoding string e*, and* output decoding string d*.*
- $Encode(e, x) \to \tilde{x}$*: Use the input encoding string e to encode input x.*
- $Eval(\tilde{C}, \tilde{x}) \to \tilde{y}$*: Evaluate $\tilde{C}$ on encoded input $\tilde{x}$, yielding encoded output $\tilde{y}$.*
- $Decode(d, \tilde{y}) \to y$*: Use the output decoding string d to decode output $\tilde{y}$. If $\tilde{y}$ is not a valid encoding, then Decode outputs $\perp$.*

*A garbling scheme is (perfectly) **correct** if for all circuits $C \in \mathcal{C}$, all inputs x, and for security parameter λ:*

$$Decode(d, Eval(\tilde{C}, Encode(e, x))) = C(x) \qquad \text{where } (\tilde{C}, e, d) \leftarrow Garble(1^\lambda, C)$$

Typically, garbling schemes are shown to satisfy selective notions called *obliviousness* and *privacy*. We consider *adaptive* variants of these; see next.

3.5 Definition of Adaptive Security

Our notion of adaptivity is based on definitions from [5] and [6]:

▶ **Definition 5** (Adaptive Privacy). *A garbling scheme is **adaptively private** if for all circuits C, there exists a simulator $\mathcal{S}$ such that for all stateful PPT adversaries $\mathcal{A}$ the following quantity is negligible in λ:*

$$\left|\Pr\left[\mathrm{Real}_{\mathrm{prv}}^{\mathcal{A},C}(1^\lambda) = 1\right] - \Pr\left[\mathrm{Ideal}_{\mathrm{prv}}^{\mathcal{A},\mathcal{S},C}(1^\lambda) = 1\right]\right|$$

where Real, Ideal *are as follows:*

$\mathrm{Real}_{\mathrm{prv}}^{\mathcal{A},C}(1^\lambda)$	$\mathrm{Ideal}_{\mathrm{prv}}^{\mathcal{A},\mathcal{S},C}(1^\lambda)$
1 : $(\tilde{C}, e, d) \leftarrow Garble^{\mathcal{O}}(1^\lambda, C)$	1 : $\tilde{C} \leftarrow \mathcal{S}^{\mathcal{O}}(1^\lambda, C)$
2 : $x \leftarrow \mathcal{A}^{\mathcal{O}}(1^\lambda, \tilde{C})$	2 : $x \leftarrow \mathcal{A}^{\mathcal{O}}(1^\lambda, \tilde{C})$
3 : $\tilde{x} \leftarrow Encode(e, x)$	3 : $(\tilde{x}, d) \leftarrow \mathcal{S}^{\mathcal{O}}(C(x))$
4 : **return** $\mathcal{A}^{\mathcal{O}}(\tilde{x}, d)$	4 : **return** $\mathcal{A}^{\mathcal{O}}(\tilde{x}, d)$

Adaptive privacy roughly states that $\mathcal{A}$ cannot distinguish the real garbled circuit from a simulated one, even when it is allowed to adaptively choose its input, and even when it is given the string d that decodes the output.

The literature also considers an alternative notion to privacy which is called *obliviousness*. Obliviousness differs from privacy in that the adversary is not allowed access to the output. That is, obliviousness roughly states that the garbled circuit alone should leak no information.

Formally, obliviousness and privacy are incomparable; one does not imply the other. However, in typical garbling schemes (including all considered in this work), privacy is proved as a consequence of obliviousness. Since privacy is the more relevant property for applications in MPC, we leave definitions and proofs of obliviousness – which are almost identical to those of privacy – to Appendix B in the full version.

4 Adaptive Security of Rekeyable Garbling Schemes

This section formalizes **query-hiding garbling schemes** and **rekeyable garbling schemes**, and we show schemes satisfying these notions are adaptively secure. In Section 5 in the full version of the paper, we see that our garbling of TSCs is query-hiding and rekeyable; Appendix C in the full version discusses other garbling schemes that satisfy our notions.

4.1 Additional Notation

Adaptive GC splits evaluation into an offline and an online phase. $\mathcal{A}$ accesses the RO even in the offline phase, when it has seen the circuit garbling, but before we send the encoded input. In our security proof, we show that the adversary cannot detect if we remove oracle queries used to garble the circuit. This is useful, because it allows us to reason that $\mathcal{A}$ cannot use work it performs in the offline phase to help it distinguish in the online phase. Formalizing this requires us to change rows of an RO, so we define appropriate notation.

▶ **Notation 1.** *Let $\mathcal{O}$ be a random oracle outputting strings in $\{0,1\}^\lambda$, and let δ be a partial map from oracle queries to oracle responses. We denote by $\mathcal{O}^\delta$ the following programming of the oracle's truth table:*

$$\mathcal{O}^\delta(x) = \begin{cases} r & \text{if } (x,r) \in \delta \\ \mathcal{O}(x) & \text{otherwise} \end{cases}$$

We will also sometimes rerandomize certain rows in $\mathcal{O}$'s truth table:

▶ **Notation 2.** *Let $\mathcal{O}$ be a random oracle outputting strings in $\{0,1\}^\lambda$, and let δ be a partial map from oracle queries to oracle responses. We denote by $\mathcal{O}^{-\delta}$ the following rerandomization of the oracle's truth table:*

$$\mathcal{O}^{-\delta}(x) = \begin{cases} \mathcal{O}'(x) & \text{if } \exists r \text{ s.t. } (x,r) \in \delta \\ \mathcal{O}(x) & \text{otherwise} \end{cases}$$

for second sampled random oracle $\mathcal{O}'$. Namely, $\mathcal{O}^{-\delta}$ replaces outputs from queries in δ with fresh uniform values.

It will also be convenient to extract from the formal procedure Garble its oracle queries. From here on, we will write $\delta, (\tilde{C}, e, d) \leftarrow \mathrm{Garble}^{\mathcal{O}}(1^\lambda, C)$ to denote that δ captures RO queries/responses occurring in Garble.

Our security properties of rekeyable garbling schemes rely on the ability to rekey any fixed garbled circuit $\tilde{C}$ (we universally quantify over all GCs). To properly formalize such notions, we will need the ability to talk about the set of all possible garbled circuits from a particular garbling scheme:

▶ **Notation 3** (Garble Support)**.** *Let $C \in \mathcal{C}$ be a circuit. Let $Supp(Garble(1^\lambda, C))$ denote the **support** of Garble on input C. Formally, $Supp(Garble(1^\lambda, C))$ is the support of the distribution defined by the procedure that (1) samples a fresh RO $\mathcal{O}$, (2) uses $\mathcal{O}$ to construct a garbled circuit and associated encoding/decoding strings $(\tilde{C}, e, d) \leftarrow Garble^{\mathcal{O}}(1^\lambda, C)$, and (3) returns $\tilde{C}$:*

4.2 Properties on Garbling Schemes

This section formalizes important definitions and properties we require on a garbling scheme to show adaptive privacy.

We start with *query hiding*, which roughly states that a garbled circuit $\tilde{C}$ does not leak the oracle queries that were used to construct it:

▶ **Definition 6** (Query Hiding). *Let* Π *be a garbling scheme. We say that* Π *is* ***query hiding*** *if for all polynomially-query-bounded adversaries* $\mathcal{A}$ *and all circuits* C*, then for the following game:*

$QueryHiding^{\mathcal{A},C}(\lambda)$	
1 :	*Sample random oracle* $\mathcal{O}$
2 :	$\delta, (\tilde{C}, e, d) \leftarrow Garble^{\mathcal{O}}(1^\lambda, C)$
3 :	**return** $\mathcal{A}^{\mathcal{O}}(1^\lambda, \tilde{C})$

The probability that $\mathcal{A}$ *makes a query in* δ *is* $\mathsf{negl}(\lambda)$*. That is, an adversary that only observes* $\tilde{C}$ *cannot make queries that the garbler made.*

Next, we define *rekeyability* of a garbling scheme. Roughly, a garbling scheme is rekeyable if there is a way to sample fresh keys that are consistent with $\tilde{C}$:

▶ **Definition 7** (Rekeyability). *Let* Π *be a garbling scheme.* Π *is* ***rekeyable*** *if the following holds. There must exist a poly-time*[5] *procedure Rekey:*

$$(\delta, e, d) \leftarrow Rekey(\lambda, C, \tilde{C}),$$

On input a circuit C *and a garbled circuit* $\tilde{C}$*, Rekey outputs a query map* δ*, an input encoding string* e*, and an output decoding string* d*. The ensembles described by the following experiments must be identically distributed:*

$Rekeyable^R(1^\lambda, C)$		$Rekeyable^I(1^\lambda, C)$	
1 :	*Sample random oracle* $\mathcal{O}$	1 :	*Sample random oracle* $\mathcal{O}$
2 :	$\delta, (\tilde{C}, e, d) \leftarrow Garble^{\mathcal{O}}(1^\lambda, C)$	2 :	$\delta, (\tilde{C}, e, d) \leftarrow Garble^{\mathcal{O}}(1^\lambda, C)$
3 :		3 :	$(\delta', e', d') \leftarrow Rekey(1^\lambda, C, \tilde{C})$
4 :	**return** $(\delta, \tilde{C}, e, d)$	4 :	**return** $(\delta', \tilde{C}, e', d')$

Rekeyability alone is not sufficient to prove adaptive security, and even selective security combined with rekeyability seems insufficient. Intuitively, it is not clear how to obtain rekey privacy (see next), which is stated w.r.t. a fixed GC, from GC privacy, which is probabilistic over sampled GCs. We accordingly define the notion of a garbling scheme that *privately rekeys*:

▶ **Definition 8** (Privately Rekeying). *Let* Π *be a rekeyable garbling scheme (Definition 7).* Π ***privately rekeys*** *if for all circuits* C*, there exists a simulator* $\mathcal{S}$ *s.t. for all* x*, all GCs* $\tilde{C} \in Supp(Garble(1^\lambda, C))$*, all oracles* $\mathcal{O}$*, and all* PPT *adversaries* $\mathcal{A}$*, the following ensembles are statistically close in* λ*:*

[5] For our work we implement the rekey procedure since our proofs rely on properties of a candidate construction; however, it is not technically necessary that Rekey is efficient. In fact, an inefficient Rekey procedure should always exist, since reverse sampling any distribution is always well-formed.

$\mathit{Real}^{\mathcal{A},C,\tilde{C},\mathcal{O},x}(\lambda)$	$\mathit{Ideal}^{\mathcal{A},\mathcal{S},C,\tilde{C},\mathcal{O},x}(\lambda)$
$(\delta, e, d) \leftarrow \mathit{Rekey}(1^\lambda, C, \tilde{C})$	$(\delta, e, d) \leftarrow \mathit{Rekey}(1^\lambda, C, \tilde{C})$
$\tilde{x} \leftarrow \mathit{Encode}(e, x)$	$\tilde{x} \leftarrow \mathit{Encode}(e, \mathbf{0})$
	$d' \leftarrow \mathcal{S}(1^\lambda, C(x), d)$
return $\mathcal{A}^{\mathcal{O}^\delta}(1^\lambda, \tilde{x}, d)$	**return** $\mathcal{A}^{\mathcal{O}^\delta}(1^\lambda, \tilde{x}, d')$

Roughly speaking, the privacy simulator $\mathcal{S}$ forges an output decoding table d' such that the GC correctly evaluates to $C(x)$, even though the input is 0.

4.3 Adaptive Security

We now prove our main theorem, which connects our notions of query hiding and rekeyability with adaptive security:

▶ **Theorem 9** (Adaptive Privacy from Private Rekeying and Query Hiding)**.** *Let* Π *be a privately rekeyable (Definition 8) garbling scheme that is query hiding (Definition 6).* Π *is adaptively private.*

Proof. By construction of a simulator:

$\mathcal{S}^{\mathcal{O}}(1^\lambda, C)$
$(\tilde{C}, e, d) \leftarrow \mathrm{Garble}^{\mathcal{O}}(1^\lambda, C)$
return $\tilde{C}$
// Second call; receive $C(x)$ from caller
$\tilde{x} \leftarrow \mathrm{Encode}(e, \mathbf{0})$
$d' \leftarrow \mathcal{S}_p(1^\lambda, C(x), d)$
return $(\tilde{x}, d')$

Here, $\mathcal{S}_p$ is the privacy simulator $\mathcal{S}$ provided by Definition 8. In the offline phase, $\mathcal{S}$ simply garbles a circuit normally. In the online phase, $\mathcal{S}$ (1) encodes the all-zeros input and (2) uses Π's privacy simulator $\mathcal{S}_p$ to forge an output decoding string d that convincingly decodes to the correct output $C(x)$.

Now, we show that the real-world and ideal-world experiments are indistinguishable. In Figure 1, we restate adaptive privacy's real/ideal experiments, in-lining the definition of our simulator. Figure 1 also specifies six hybrid distributions used to show indistinguishability.

We find most direct to argue by "meeting in the middle". Namely, we proceed starting from the real/ideal ensemble, and at each step, we show the current real/ideal ensemble is indistinguishable from some respective intermediate ensemble. We conclude by showing these two final ensembles are indistinguishable.

Removing offline oracle queries. In our crucial proof step, we argue the following:

$$\{\mathrm{Real}^{\mathcal{A},C}_{\mathrm{prv}}(\lambda)\} \approx \{\mathrm{F}^{\mathcal{A},C}_{\mathrm{prv},R}(\lambda)\} \qquad \text{and} \qquad \{\mathrm{Ideal}^{\mathcal{A},C}_{\mathrm{prv}}(\lambda)\} \approx \{\mathrm{F}^{\mathcal{A},C}_{\mathrm{prv},S}(\lambda)\}$$

In this step we remove from $\mathcal{A}$'s oracle all queries issued by the call to Garble, but only in the offline phase. Jumping ahead, our informal objective is to show that $\mathcal{A}$'s chosen input x must be independent of Garble's random oracle queries.

$\mathrm{Real}^{\mathcal{A},C}_{\mathrm{prv}}(\lambda)$	$\mathrm{Ideal}^{\mathcal{A},C}_{\mathrm{prv}}(\lambda)$
Sample random oracle $\mathcal{O}$	Sample random oracle $\mathcal{O}$
$(\tilde{C}, e, d) \leftarrow \mathrm{Garble}^{\mathcal{O}}(1^\lambda, C)$	$(\tilde{C}, e, d) \leftarrow \mathrm{Garble}^{\mathcal{O}}(1^\lambda, C)$
$x \leftarrow \mathcal{A}^{\mathcal{O}}(1^\lambda, \tilde{C})$	$x \leftarrow \mathcal{A}^{\mathcal{O}}(1^\lambda, \tilde{C})$
$\tilde{x} \leftarrow \mathrm{Encode}(e, x)$	$\tilde{x} \leftarrow \mathrm{Encode}(e, \mathbf{0})$
	$d' \leftarrow \mathcal{S}_p(1^\lambda, C(x), d)$
return $\mathcal{A}^{\mathcal{O}}(1^\lambda, \tilde{x}, d)$	**return** $\mathcal{A}^{\mathcal{O}}(1^\lambda, \tilde{x}, d')$

$\mathrm{F}^{\mathcal{A},C}_{\mathrm{prv},R}(\lambda)$	$\mathrm{F}^{\mathcal{A},C}_{\mathrm{prv},S}(\lambda)$
Sample random oracle $\mathcal{O}$	Sample random oracle $\mathcal{O}$
$\delta, (\tilde{C}, e, d) \leftarrow \mathrm{Garble}^{\mathcal{O}}(1^\lambda, C)$	$\delta, (\tilde{C}, e, d) \leftarrow \mathrm{Garble}^{\mathcal{O}}(1^\lambda, C)$
$x \leftarrow \mathcal{A}^{\mathcal{O}^{-\delta}}(1^\lambda, \tilde{C})$	$x \leftarrow \mathcal{A}^{\mathcal{O}^{-\delta}}(1^\lambda, \tilde{C})$
$\tilde{x} \leftarrow \mathrm{Encode}(e, x)$	$\tilde{x} \leftarrow \mathrm{Encode}(e, \mathbf{0})$
	$d' \leftarrow \mathcal{S}_p(1^\lambda, C(x), d)$
return $\mathcal{A}^{\mathcal{O}}(1^\lambda, \tilde{x}, d)$	**return** $\mathcal{A}^{\mathcal{O}}(1^\lambda, \tilde{x}, d')$

$\mathrm{G}^{\mathcal{A},C}_{\mathrm{prv},R}(\lambda)$	$\mathrm{G}^{\mathcal{A},C}_{\mathrm{prv},S}(\lambda)$
Sample random oracles $\mathcal{O}$ and $\mathcal{O}'$	Sample random oracles $\mathcal{O}$ and $\mathcal{O}'$
$\delta, (\tilde{C}, e, d) \leftarrow \mathrm{Garble}^{\mathcal{O}'}(1^\lambda, C)$	$\delta, (\tilde{C}, e, d) \leftarrow \mathrm{Garble}^{\mathcal{O}'}(1^\lambda, C)$
$x \leftarrow \mathcal{A}^{\mathcal{O}}(1^\lambda, \tilde{C})$	$x \leftarrow \mathcal{A}^{\mathcal{O}}(1^\lambda, \tilde{C})$
$\tilde{x} \leftarrow \mathrm{Encode}(e, x)$	$\tilde{x} \leftarrow \mathrm{Encode}(e, \mathbf{0})$
	$d' \leftarrow \mathcal{S}_p(1^\lambda, C(x), d)$
return $\mathcal{A}^{\mathcal{O}^\delta}(1^\lambda, \tilde{x}, d)$	**return** $\mathcal{A}^{\mathcal{O}^\delta}(1^\lambda, \tilde{x}, d')$

$\mathrm{H}^{\mathcal{A},C}_{\mathrm{prv},R}(\lambda)$	$\mathrm{H}^{\mathcal{A},C}_{\mathrm{prv},S}(\lambda)$
Sample random oracles $\mathcal{O}$ and $\mathcal{O}'$	Sample random oracles $\mathcal{O}$ and $\mathcal{O}'$
$\delta, (\tilde{C}, e, d) \leftarrow \mathrm{Garble}^{\mathcal{O}'}(1^\lambda, C)$	$\delta, (\tilde{C}, e, d) \leftarrow \mathrm{Garble}^{\mathcal{O}'}(1^\lambda, C)$
$x \leftarrow \mathcal{A}^{\mathcal{O}}(1^\lambda, \tilde{C})$	$x \leftarrow \mathcal{A}^{\mathcal{O}}(1^\lambda, \tilde{C})$
$(\delta', e', d') \leftarrow \mathrm{Rekey}(1^\lambda, C, \tilde{C})$	$(\delta', e', d') \leftarrow \mathrm{Rekey}(1^\lambda, C, \tilde{C})$
$\tilde{x} \leftarrow \mathrm{Encode}(e', x)$	$\tilde{x} \leftarrow \mathrm{Encode}(e', \mathbf{0})$
	$d'' \leftarrow \mathcal{S}_p(1^\lambda, C(x), d')$
return $\mathcal{A}^{\mathcal{O}^{\delta'}}(1^\lambda, \tilde{x}, d')$	**return** $\mathcal{A}^{\mathcal{O}^{\delta'}}(1^\lambda, \tilde{x}, d'')$

Figure 1 Real, ideal, and hybrid distributions used in our proof of Theorem 9.

Of course, there is a difference between games Real and $\mathrm{F}_{\mathrm{prv},R}$ (resp. Ideal and $\mathrm{F}_{\mathrm{prv},I}$). In the former game $\mathcal{A}$ is given access to the same oracle twice, and in the latter $\mathcal{A}$ is given access to two oracles that differ by δ. Thus, if $\mathcal{A}$ can guess a query in δ, it can distinguish the two games, since in the first game it will see the oracle respond consistently in the offline and online phases, and in the second game it will see the oracle "disagree with itself".

Query hiding (Definition 6) is precisely what we need to show that $\mathcal{A}$ cannot guess a query in δ. Indeed, an unbounded adversary with only polynomial oracle queries has at most $\mathsf{negl}(\lambda)$ chance to sample a point in δ. As such, $\mathcal{A}$ only has a $\mathsf{negl}(\lambda)$ probability of determining if it was given $\mathcal{O}$ or $\mathcal{O}^{-\delta}$ on line 3.

Rearranging the oracles. In our second step, we perform a "refactoring" of the random oracle queries. In particular, we argue:

$$\{\mathrm{F}^{\mathcal{A},C}_{\mathrm{prv},R}(\lambda)\} \equiv \{\mathrm{G}^{\mathcal{A},C}_{\mathrm{prv},R}(\lambda)\} \qquad \text{and} \qquad \{\mathrm{F}^{\mathcal{A},C}_{\mathrm{prv},S}(\lambda)\} \equiv \{\mathrm{G}^{\mathcal{A},C}_{\mathrm{prv},S}(\lambda)\}$$

Indeed, games F and G are identically distributed, as they are merely a rearrangement of the random oracle queries. The output distribution of $\mathrm{Garble}^{\mathcal{O}}$ in line 2 of hybrid F is independent of the programmed oracle $\mathcal{O}^{-\delta}$ on line 2. As such, we can rewrite lines 2 and 3 to use different oracles altogether to get hybrid G. On line 6, we note that the only part of $\mathcal{O}'$ that $\mathrm{Garble}^{\mathcal{O}'}$ depends on are exactly those queries in δ. As such, the garbler may as well have used $\mathcal{O}^{\delta}$ to garble. After this step, it is clear that $\mathcal{A}$'s chosen input x must be independent of the random oracle $\mathcal{O}'$ used to garble, since $\mathcal{A}$ is not allowed access to $\mathcal{O}'$.

Rekeying the GC. We use rekeyability (Definition 7) to sample fresh keys associated with the garbled circuit $\tilde{C}$. The following is immediate by rekeyability:

$$\{\mathrm{G}^{\mathcal{A},C}_{\mathrm{prv},R}(\lambda)\} \equiv \{\mathrm{H}^{\mathcal{A},C}_{\mathrm{prv},R}(\lambda)\} \qquad \text{and} \qquad \{\mathrm{G}^{\mathcal{A},C}_{\mathrm{prv},S}(\lambda)\} \equiv \{\mathrm{H}^{\mathcal{A},C}_{\mathrm{prv},S}(\lambda)\}$$

Privacy. Finally, we bridge from the "real world", where we encode x, to the "ideal world", where we encode 0: $\{\mathrm{H}^{\mathcal{A},C}_{\mathrm{prv},R}(\lambda)\} \overset{\mathrm{s}}{\approx} \{\mathrm{H}^{\mathcal{A},C}_{\mathrm{prv},S}(\lambda)\}$. Notice that the outputs of garbling – $\delta, \tilde{C}, e, d$ – are independent of $\mathcal{O}$. Thus, the choice of x is also independent of δ, e, d, except insofar as they are constrained by garbled circuit $\tilde{C}$. We can capture this sentiment by treating $\tilde{C}$, $\mathcal{O}$, and x as if they are universally quantified. This is exactly the setting considered in private rekeying (Definition 8). Applying private rekeying thus discharges the proof. Any query hiding, privately rekeyable garbling scheme is adaptively private. ◀

References

1 Anasuya Acharya, Karen Azari, and Chethan Kamath. On the adaptive security of free-xor-based garbling schemes in the plain model. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 214–244. Springer, 2025. doi:10.1007/978-3-031-91095-1_8.

2 Benny Applebaum, Yuval Ishai, Eyal Kushilevitz, and Brent Waters. Encoding functions with constant online rate or how to compress garbled circuits keys. In Ran Canetti and Juan A. Garay, editors, *CRYPTO 2013, Part II*, volume 8043 of *LNCS*, pages 166–184. Springer, Berlin, Heidelberg, August 2013. doi:10.1007/978-3-642-40084-1_10.

3 Marshall Ball, Tal Malkin, and Mike Rosulek. Garbling gadgets for Boolean and arithmetic circuits. In Edgar R. Weippl, Stefan Katzenbeisser, Christopher Kruegel, Andrew C. Myers, and Shai Halevi, editors, *ACM CCS 2016*, pages 565–577. ACM Press, October 2016. doi:10.1145/2976749.2978410.

4 Cruz Barnum, David Heath, Vladimir Kolesnikov, and Rafail Ostrovsky. Adaptive garbled circuits and garbled RAM from non-programmable random oracles. Cryptology ePrint Archive, Paper 2023/1527, 2023. URL: https://eprint.iacr.org/2023/1527.

5 Mihir Bellare, Viet Tung Hoang, and Phillip Rogaway. Adaptively secure garbling with applications to one-time programs and secure outsourcing. In Xiaoyun Wang and Kazue Sako, editors, *ASIACRYPT 2012*, volume 7658 of *LNCS*, pages 134–153. Springer, Berlin, Heidelberg, December 2012. doi:10.1007/978-3-642-34961-4_10.

6 Mihir Bellare, Viet Tung Hoang, and Phillip Rogaway. Foundations of garbled circuits. In Ting Yu, George Danezis, and Virgil D. Gligor, editors, *ACM CCS 2012*, pages 784–796. ACM Press, October 2012. doi:10.1145/2382196.2382279.

7 Mihir Bellare and Phillip Rogaway. Optimal asymmetric encryption. In Alfredo De Santis, editor, *EUROCRYPT'94*, volume 950 of *LNCS*, pages 92–111. Springer, Berlin, Heidelberg, May 1995. doi:10.1007/BFb0053428.

8 Seung Geol Choi, Jonathan Katz, Ranjit Kumaresan, and Hong-Sheng Zhou. On the security of the "free-XOR" technique. In Ronald Cramer, editor, *TCC 2012*, volume 7194 of *LNCS*, pages 39–53. Springer, Berlin, Heidelberg, March 2012. doi:10.1007/978-3-642-28914-9_3.

9 Marc Fischlin, Anja Lehmann, Thomas Ristenpart, Thomas Shrimpton, Martijn Stam, and Stefano Tessaro. Random oracles with(out) programmability. In Masayuki Abe, editor, *ASIACRYPT 2010*, volume 6477 of *LNCS*, pages 303–320. Springer, Berlin, Heidelberg, December 2010. doi:10.1007/978-3-642-17373-8_18.

10 Eiichiro Fujisaki, Tatsuaki Okamoto, David Pointcheval, and Jacques Stern. RSA-OAEP is secure under the RSA assumption. In Joe Kilian, editor, *CRYPTO 2001*, volume 2139 of *LNCS*, pages 260–274. Springer, Berlin, Heidelberg, August 2001. doi:10.1007/3-540-44647-8_16.

11 Chaya Ganesh, Yashvanth Kondi, Claudio Orlandi, Mahak Pancholi, Akira Takahashi, and Daniel Tschudi. Witness-succinct universally-composable SNARKs. In Carmit Hazay and Martijn Stam, editors, *EUROCRYPT 2023, Part II*, volume 14005 of *LNCS*, pages 315–346. Springer, Cham, April 2023. doi:10.1007/978-3-031-30617-4_11.

12 Sanjam Garg, Steve Lu, and Rafail Ostrovsky. Black-box garbled RAM. In Venkatesan Guruswami, editor, *56th FOCS*, pages 210–229. IEEE Computer Society Press, October 2015. doi:10.1109/FOCS.2015.22.

13 Sanjam Garg, Steve Lu, Rafail Ostrovsky, and Alessandra Scafuro. Garbled RAM from one-way functions. In Rocco A. Servedio and Ronitt Rubinfeld, editors, *47th ACM STOC*, pages 449–458. ACM Press, June 2015. doi:10.1145/2746539.2746593.

14 Sanjam Garg, Rafail Ostrovsky, and Akshayaram Srinivasan. Adaptive garbled RAM from laconic oblivious transfer. In Hovav Shacham and Alexandra Boldyreva, editors, *CRYPTO 2018, Part III*, volume 10993 of *LNCS*, pages 515–544. Springer, Cham, August 2018. doi:10.1007/978-3-319-96878-0_18.

15 Craig Gentry, Shai Halevi, Steve Lu, Rafail Ostrovsky, Mariana Raykova, and Daniel Wichs. Garbled RAM revisited. In Phong Q. Nguyen and Elisabeth Oswald, editors, *EUROCRYPT 2014*, volume 8441 of *LNCS*, pages 405–422. Springer, Berlin, Heidelberg, May 2014. doi:10.1007/978-3-642-55220-5_23.

16 Tianyao Gu, Afonso Tinoco, Sri Harish G. Rajan, and Elaine Shi. PicoGRAM: Practical garbled RAM from decisional diffie-hellman. In *CRYPTO 2025, Part VIII*, LNCS, pages 247–281. Springer, Cham, August 2025. doi:10.1007/978-3-032-01913-4_8.

17 Chun Guo, Jonathan Katz, Xiao Wang, Chenkai Weng, and Yu Yu. Better concrete security for half-gates garbling (in the multi-instance setting). In Daniele Micciancio and Thomas Ristenpart, editors, *CRYPTO 2020, Part II*, volume 12171 of *LNCS*, pages 793–822. Springer, Cham, August 2020. doi:10.1007/978-3-030-56880-1_28.

18 Xiaojie Guo, Kang Yang, Xiao Wang, Yu Yu, and Zheli Liu. Unmodified half-gates is adaptively secure - so is unmodified three-halves. Cryptology ePrint Archive, Report 2023/1528, 2023. URL: https://eprint.iacr.org/2023/1528.

19 David Heath. Efficient arithmetic in garbled circuits. In Marc Joye and Gregor Leander, editors, *EUROCRYPT 2024, Part V*, volume 14655 of *LNCS*, pages 3–31. Springer, Cham, May 2024. doi:10.1007/978-3-031-58740-5_1.

20 David Heath, Vladimir Kolesnikov, and Rafail Ostrovsky. EpiGRAM: Practical garbled RAM. In Orr Dunkelman and Stefan Dziembowski, editors, *EUROCRYPT 2022, Part I*, volume 13275 of *LNCS*, pages 3–33. Springer, Cham, May / June 2022. doi:10.1007/978-3-031-06944-4_1.

21 David Heath, Vladimir Kolesnikov, and Rafail Ostrovsky. Tri-state circuits - A circuit model that captures RAM. In Helena Handschuh and Anna Lysyanskaya, editors, *CRYPTO 2023, Part IV*, volume 14084 of *LNCS*, pages 128–160. Springer, Cham, August 2023. doi:10.1007/978-3-031-38551-3_5.

22 Brett Hemenway, Zahra Jafargholi, Rafail Ostrovsky, Alessandra Scafuro, and Daniel Wichs. Adaptively secure garbled circuits from one-way functions. In Matthew Robshaw and Jonathan Katz, editors, *CRYPTO 2016, Part III*, volume 9816 of *LNCS*, pages 149–178. Springer, Berlin, Heidelberg, August 2016. doi:10.1007/978-3-662-53015-3_6.

23 Zahra Jafargholi and Sabine Oechsner. Adaptive security of practical garbling schemes. In Karthikeyan Bhargavan, Elisabeth Oswald, and Manoj Prabhakaran, editors, *INDOCRYPT 2020*, volume 12578 of *LNCS*, pages 741–762. Springer, Cham, December 2020. doi:10.1007/978-3-030-65277-7_33.

24 Zahra Jafargholi and Daniel Wichs. Adaptive security of Yao's garbled circuits. In Martin Hirt and Adam D. Smith, editors, *TCC 2016-B, Part I*, volume 9985 of *LNCS*, pages 433–458. Springer, Berlin, Heidelberg, October / November 2016. doi:10.1007/978-3-662-53641-4_17.

25 Chethan Kamath, Karen Klein, Krzysztof Pietrzak, and Daniel Wichs. Limits on the adaptive security of Yao's garbling. In Tal Malkin and Chris Peikert, editors, *CRYPTO 2021, Part II*, volume 12826 of *LNCS*, pages 486–515, Virtual Event, August 2021. Springer, Cham. doi:10.1007/978-3-030-84245-1_17.

26 Vladimir Kolesnikov and Thomas Schneider. Improved garbled circuit: Free XOR gates and applications. In Luca Aceto, Ivan Damgård, Leslie Ann Goldberg, Magnús M. Halldórsson, Anna Ingólfsdóttir, and Igor Walukiewicz, editors, *ICALP 2008, Part II*, volume 5126 of *LNCS*, pages 486–498. Springer, Berlin, Heidelberg, July 2008. doi:10.1007/978-3-540-70583-3_40.

27 Huijia Lin, Rafael Pass, and Muthuramakrishnan Venkitasubramaniam. A unified framework for concurrent security: universal composability from stand-alone non-malleability. In Michael Mitzenmacher, editor, *41st ACM STOC*, pages 179–188. ACM Press, May / June 2009. doi:10.1145/1536414.1536441.

28 Yehuda Lindell and Benny Pinkas. A proof of security of Yao's protocol for two-party computation. *Journal of Cryptology*, 22(2):161–188, April 2009. doi:10.1007/s00145-008-9036-8.

29 Steve Lu and Rafail Ostrovsky. How to garble RAM programs. In Thomas Johansson and Phong Q. Nguyen, editors, *EUROCRYPT 2013*, volume 7881 of *LNCS*, pages 719–734. Springer, Berlin, Heidelberg, May 2013. doi:10.1007/978-3-642-38348-9_42.

30 Kathleen Moriarty, Burt Kaliski, Jakob Jonsson, and Andreas Rusch. PKCS #1: RSA Cryptography Specifications Version 2.2. RFC 8017, November 2016. doi:10.17487/RFC8017.

31 Jesper Buus Nielsen. Separating random oracle proofs from complexity theoretic proofs: The non-committing encryption case. In Moti Yung, editor, *CRYPTO 2002*, volume 2442 of *LNCS*, pages 111–126. Springer, Berlin, Heidelberg, August 2002. doi:10.1007/3-540-45708-9_8.

32 NIST. NIST workshop on multi-party threshold schemes 2020, 2020. URL: https://csrc.nist.gov/Events/2020/mpts2020.

33 NIST. NIST workshop on multi-party threshold schemes 2023, 2023. URL: https://csrc.nist.gov/events/2023/mpts2023.

34 Andrew Park, Wei-Kai Lin, and Elaine Shi. NanoGRAM: Garbled RAM with $\widetilde{O}(\log N)$ overhead. In Carmit Hazay and Martijn Stam, editors, *EUROCRYPT 2023, Part I*, volume 14004 of *LNCS*, pages 456–486. Springer, Cham, April 2023. doi:10.1007/978-3-031-30545-0_16.

35 Mike Rosulek and Lawrence Roy. Three halves make a whole? Beating the half-gates lower bound for garbled circuits. In Tal Malkin and Chris Peikert, editors, *CRYPTO 2021, Part I*, volume 12825 of *LNCS*, pages 94–124, Virtual Event, August 2021. Springer, Cham. doi:10.1007/978-3-030-84242-0_5.

36 Dominique Unruh. Random oracles and auxiliary input. In Alfred Menezes, editor, *CRYPTO 2007*, volume 4622 of *LNCS*, pages 205–223. Springer, Berlin, Heidelberg, August 2007. doi:10.1007/978-3-540-74143-5_12.

37 Andrew Chi-Chih Yao. How to generate and exchange secrets (extended abstract). In *27th FOCS*, pages 162–167. IEEE Computer Society Press, October 1986. doi:10.1109/SFCS.1986.25.

38 Samee Zahur, Mike Rosulek, and David Evans. Two halves make a whole - reducing data transfer in garbled circuits using half gates. In Elisabeth Oswald and Marc Fischlin, editors, *EUROCRYPT 2015, Part II*, volume 9057 of *LNCS*, pages 220–250. Springer, Berlin, Heidelberg, April 2015. doi:10.1007/978-3-662-46803-6_8.

Partial Derandomization for Leakage-Resilient Shamir's Secret Sharing over Composite Order Fields

S. Venkitesh
Institute for the Theory of Computing, The Stein Faculty of Computer and Information Science, Ben Gurion University of the Negev, Beersheva, Israel

Abstract

We make progress on the question of constructing explicit evaluation places for leakage-resilient Shamir's secret sharing, over composite order fields. Previously, Maji et al. (EUROCRYPT 2024) showed that random evaluation places yield Shamir's secret sharing over the composite order field $\mathbb{F}_{p^d}$ that is statistically secure against physical bit leakage. Later, Nguyen (EUROCRYPT 2025) established a *dichotomy* that linear code-based secret-sharing scheme over the field $\mathbb{F}_{p^d}$ is either statistically secure or completely insecure against such leakage.

Building upon Nguyen's dichotomy, we present a partial derandomization of evaluation places, improving upon the Maji et al. result for a restricted regime of parameters. We replace the random choice of n independent evaluation places by the iterates $x_j = \Phi^j(x_0)$ of a simple fixed rational function Φ, where the initial point $x_0 \in \mathbb{F}_{p^d}^*$ is randomly chosen. The randomness in the evaluation places thus drops from $nd \log p$ bits to $d \log p$ bits. Our construction is valid for the regime $n = O(d/\log_p d)$, and any reconstruction threshold $k \geq 2$; in fact, the scheme attains perfect security (statistical distance exactly zero) against single-block leakage.

Building on Nguyen's dichotomy, our technique is a partial-fraction non-degeneracy argument that exploits the distinct poles of the rational iterates.

2012 ACM Subject Classification Theory of computation → Cryptographic primitives; Security and privacy → Cryptanalysis and other attacks

Keywords and phrases Shamir's secret sharing, leakage resilience, physical bit probing, physical bit leakage, secure evaluation places, rational functions

Digital Object Identifier 10.4230/LIPIcs.ITC.2026.12

Related Version *Full Version*: `https://eprint.iacr.org/2026/1475`

Acknowledgements The author thanks the anonymous reviewers of ITC 2026 for critical feedback, as well as for pointing to [20] and the dichotomy result therein.

1 Introduction

Threshold secret-sharing schemes, such as Shamir's scheme [21], distribute a secret among n parties so that any k of them can reconstruct the secret, while any fewer than k parties learn nothing about it. In the standard corruption model, an adversary obtains the complete shares of some parties and has no information about the remaining shares.

Side-channel attacks have repeatedly circumvented this all-or-nothing assumption. Rather than corrupting entire shares, a side-channel adversary accumulates small amounts of information, like individual bits, power traces, and timing signals, from *all* shares simultaneously (see Ishai, Sahai, and Wagner [10]). The mathematical abstraction of such threats is *independent local leakage*: the adversary applies a bounded function to each share independently and observes only the outputs. The study of locally leakage-resilient secret sharing was initiated by Benhamouda, Degwekar, Ishai, and Rabin [4] and is also implicit in the work of Goyal and Kumar [7].

A particularly natural leakage model, introduced by [10], probes *physical bits* in the memory storing each share. Since elements of a finite field $\mathbb{F}_{p^d}$ are stored as d coordinates over $\mathbb{F}_p$, each represented as a $\lceil \log_2 p \rceil$-bit binary string, a physical bit probe extracts a single bit from this representation. Additive secret sharing is known to be vulnerable to such probes: the parity-of-parities attack of Maji, Nguyen, Paskin-Cherniavsky, Suad, and Wang [15] leaks the least significant bit of each share and distinguishes secrets with advantage $(2/\pi)^n$ over any prime field $\mathbb{F}_p$. Over characteristic-2 fields, this attack can distinguish secrets 0 and 1 with certainty. Shamir's secret sharing inherits these vulnerabilities if its evaluation places are chosen carelessly ([15], and Costes and Stam [5]).

The question, then, is whether Shamir's scheme can be instantiated to resist physical bit probes. Over prime fields, [15] showed that choosing evaluation places uniformly at random yields a leakage-resilient scheme with high probability. Maji, Nguyen, Paskin-Cherniavsky, and Ye [17] extended this randomized construction to composite order fields, particularly fields of characteristic 2, and also gave an exact security classifier for $k = 2$ against single-block leakage. However, no explicit evaluation places were known for $k > 2$ or for the general block-leakage regime over composite fields. Even for $k = 2$, the classifier of [17] certifies individual evaluation-place tuples but does not produce a closed form family. Separately, Hwang, Maji, Nguyen, and Ye [9] initiated the study of explicit constructions over prime fields, providing classifiers and derandomized evaluation places for Mersenne and Fermat primes in the full-threshold regime $n = k$.

In this work, we present a partial derandomization of [17]. Building on the dichotomy of Nguyen [20], which establishes that any linear code-based secret-sharing scheme over the field $\mathbb{F}_{p^d}$ is either statistically secure or completely insecure against physical bit leakage, we replace the random choice of n independent evaluation places by a structured one-parameter family: the evaluation places are $x_j = \Phi^j(x_0)$, where $\Phi(x) := \alpha x/(x+1)$, α is a fixed multiplicative generator of $\mathbb{F}_{p^d}^*$, and $x_0 \in \mathbb{F}_{p^d}^*$ is a randomly chosen *base point*. As it turns out, Φ is a *Möbius transformation*, and the utility of such structured algebraic transformations in defining evaluation places seems unexplored so far. The randomness needed to specify the n evaluation places drops from $nd \log p$ bits to $d \log p$ bits, required to choose the base point x_0. The above mentioned dichotomy argument reduces the security question to a full-rank condition on a test matrix, which we verify for our chosen points by observing a non-degeneracy property that exploits the distinct poles of the iterates.

The dichotomy itself is implicit, for the case $k = 2$, in the linear-algebraic arguments of [17], and [20] later extends this to the case of general k, as well as to general linear code-based secret sharing. We also give an alternative direct derivation of the dichotomy in our setting for completeness, in Appendix A.

1.1 Motivation

The problem of *derandomization of evaluation places* is the central motivation for our work.

The randomized constructions of [15, 17] demonstrate that most evaluation places are secure. In practice, however, they require trusted public randomness, for instance a randomness beacon, to select the places. An adversary who can influence the random seed may steer the construction toward vulnerable evaluation places, unbeknownst to the honest parties. This concern is not merely theoretical; the NIST standardization effort for threshold cryptographic schemes (see Brandão and Peralta [1]), and the practical deployment considerations studied by Faust, Masure, Micheli, Orlt, and Standaert [6] both underscore the need for deterministic, verifiable instantiations.

Ideally, one would like evaluation places that are fixed by the field specification alone, with a proof that they resist physical bit probes. Towards this, [9] initiated the study of classifier and derandomization constructions over prime fields. Their techniques, however, are inherently tied to the structure of prime fields: they exploit square wave orthogonality and 2-adic valuations of rational approximations, both of which rely on the nonlinearity of the bit-extraction map over $\mathbb{F}_p$. Over composite fields $\mathbb{F}_{p^d}$ with $d \geq 2$, the analogous coordinate-extraction map is $\mathbb{F}_p$-linear, and the Fourier-analytic machinery of [9] does not apply. The present work exploits this linearity to make partial progress on the derandomization problem in the composite-field setting; the precise results are stated in Section 1.2.

1.2 Our results

We work over $\mathbb{F} = \mathbb{F}_{p^d}$, where p is any prime and $d \geq 2$. Fix a primitive polynomial $\Pi(t) \in \mathbb{F}_p[t]$ of degree d, and let α be a root of Π in $\mathbb{F}$, so that $\mathrm{ord}(\alpha) = p^d - 1$ and $\mathbb{F} = \mathbb{F}_p[\alpha]$ with canonical basis $\mathcal{B} = \{1, \alpha, \alpha^2, \ldots, \alpha^{d-1}\}$.

Our construction is based on a rational function. Define the *step operator*

$$\Phi(x) = \frac{\alpha x}{x+1},$$

which is a Möbius transformation on the projective line $\mathbb{P}^1(\mathbb{F}) := \mathbb{F} \cup \{\infty\}$ with projective order $p^d - 1$. Our evaluation places are the iterates

$$x_j = \Phi^j(x_0), \quad j = 0, 1, \ldots, n-1,$$

where Φ^j denotes the j-th iterate of Φ under composition, applied to a base point $x_0 \in \mathbb{F}^*$.

The choice of Φ is not arbitrary. The single structural property of the orbit that drives the entire security analysis is that the iterates $\Phi^0, \Phi^1, \ldots, \Phi^{n-1}$, viewed as Möbius transformations on the projective line $\mathbb{P}^1(\mathbb{F})$, have *pairwise distinct finite poles*. This pole-distinctness is what powers the partial-fraction non-degeneracy argument in Section 4. In fact, motivated from the construction of *folded Reed-Solomon codes* [8], which are polynomial codes that can be *list decoded* up to the information theoretic limit, a more obvious and simpler candidate is $\Psi(x) = \alpha x$. Here, every iterate $\Psi^j(x) = \alpha^j x$ is a pure dilation with its only pole at ∞. So the iterates share a single common pole and the partial-fraction argument collapses entirely. We make this contrast precise in Remark 3.6. The role of the $+1$ in the denominator of Φ is therefore not cosmetic; it perturbs each iterate's pole to a distinct finite point of $\mathbb{F}^*$, placing the orbit in sufficiently generic position so that a good bad-set bound becomes possible.

Our main results are as follows.

▶ **Theorem 1.1** (Perfect security against single-block leakage)**.** *Let p be any prime, $d \geq 2$, $k \geq 2$, and $k \leq n \leq d(k-1)$ with $n \leq p^d - 1$. Let α be a multiplicative generator of $\mathbb{F}_{p^d}^*$ and $\Phi(x) = \alpha x/(x+1)$. There exists a set* $\mathrm{Bad} \subset \mathbb{F}_{p^d}^*$ *with*

$$|\mathrm{Bad}| \leq n + n(dp)^n$$

such that for any $x_0 \in \mathbb{F}_{p^d}^* \setminus \mathrm{Bad}$, the evaluation places $\vec{X} = (x_0, \ldots, x_{n-1})$ give a scheme $\mathrm{ShamirSS}(n, k, \vec{X})_{\mathbb{F}_{p^d}}$ *that is* perfectly secure *against all single-$\mathbb{F}_p$-block leakage patterns.*

In particular, the scheme is perfectly secure against all single-physical-bit-per-share leakage (for any p), and against any leakage that applies an arbitrary function $g_j : \mathbb{F}_p \to S_j$ to a single $\mathbb{F}_p$-block of each share, where $S_0, \ldots, S_{n-1}$ are arbitrary nonempty finite sets.

A good base point x_0 exists whenever $d > n(1 + \log_p d) + O(\log_p n)$, giving a practical range of $n = O(d/\log_p d)$ parties.

Table 1 Comparison of our results with the randomized results of [17, 20]. Here $2 \le k \le n$.

Aspect	[17]	[20]	**This work**
Evaluation places	Random in $(\mathbb{F}^*)^n$	Fixed; arbitrary	$\Phi^j(x_0)$ orbit
Multipliers v_i (in shares $v_i P(X_i)$)	All 1	Random in $(\mathbb{F}^*)^n$	All 1
Security guarantee	$\epsilon = 2^{-\Omega(d)}$	$\epsilon = 2^{-\Omega(d)}$	SD $= 0$ for $x_0 \notin$ Bad
Max parties (general k)	$O(dk/\log_p d)$	$O(dk/\log_p d)$	$O(d/\log_p d)$
Multi-block, $M < d$	$\epsilon = 2^{-\Omega(d)}$	$\epsilon = 2^{-\Omega(d)}$	Perfect (fixed pattern)
Random bits in evaluation places	$nd \log p$	0	$d \log p$ (for x_0)
Random bits in multipliers	0	$nd \log p$	0

The security guarantee is *perfect*: the statistical distance is zero, not merely exponentially small. Within the parameter range $n = O(d/\log_p d)$, this is a qualitative strengthening of the statistical security $\epsilon = 2^{-\Omega(d)}$ achieved by the randomized constructions of [17, 20]. The trade-off is on party count: [17, 20] support $n = O(dk/\log_p d)$ parties versus our $n = O(d/\log_p d)$, reflecting the cost of restricting the evaluation places to a one-parameter family.

The construction is a partial derandomization. Once a base point x_0 is fixed, the evaluation places $x_j = \Phi^j(x_0)$ are determined by the field representation alone; the randomness needed to specify the n places thus drops from $nd \log p$ bits (for n independent random places, as in [17]) to $d \log p$ bits needed to fix the single $x_0 \in \mathbb{F}^*$. In the headline regime, $x_0 = \alpha$ is a structured candidate; whether it is good depends on the choice of primitive polynomial Π and is certified by Algorithm 1 (Section 3.3), a sound test whose GOOD verdict guarantees perfect security, with the same worst-case runtime as the classifying algorithm of [20].

For multi-block leakage, we prove the following.

▶ **Theorem 1.2** (Multi-block leakage, improved bound)**.** *Under the hypotheses of Theorem 1.1, fix an admissible multi-block leakage pattern with $M < d - \log_p(2n)$ total $\mathbb{F}_p$-blocks leaked. Then the bad set has size $\le n + n \cdot p^M$, and a good x_0 always exists.*

For universality over all admissible multi-block patterns with $\le M$ blocks total, we get the bound $|\text{Bad}| \le n + n(dpe)^M$, and a good x_0 exists when $M = O(d/\log_p d)$.

For the proof, we observe that the Vandermonde has nontrivial kernel, and the direct bound from Theorem 5.3 is used; obtaining a polynomial bound for this regime remains open (Section 1.6).

1.3 Comparison with prior work

Table 1 summarizes the comparison with [17, 20], and a few gaps remain. First, there is a factor-k loss in the maximum number of parties for large k; we conjecture this loss is inherent to single-parameter constructions, but leave the question open (Section 1.6). Second, multi-block universality is limited to $M = O(d/\log_p d)$ by the pattern enumeration bottleneck. Closing these gaps, particularly by obtaining a polynomial bad-set bound, is an interesting direction for future work; see Section 1.6.

1.4 Related work

We survey the most relevant prior results.

Leakage-resilient secret sharing

[3, 4] proved that Shamir's scheme is leakage-resilient against arbitrary single-bit local leakage when k/n exceeds a constant threshold. Subsequent works [16, 12, 19] improved this threshold, with the current best being $k \geq 0.69n$. These results hold for *all* evaluation places but require a large reconstruction-to-party ratio. Among other lines of work that bring linear algebraic structure to bear on these problems, Koga and Abe [13] determine tight bounds on the local leakage resilience of the additive (n, n)-threshold scheme via the eigenvalues of circulant matrices.

The complementary regime (small k relative to n) is where the choice of evaluation places becomes critical. [15] initiated the study of this regime over prime fields, and [17] extended it to composite fields. Other constructions of leakage-resilient (non-Shamir) secret sharing, including those based on algebraic geometry codes, appear in [14, 18] and the references therein.

Physical bit probing and the [17] framework

The closest prior randomized construction we improve upon is [17]. Their construction works over $\mathbb{F}_{p^d}$ for any prime p and combines three ingredients: (i) a Fourier-analytic upper bound on statistical distance in terms of dual GRS codewords and Fourier coefficients of block-indicator functions, (ii) a Bézout-type bound on simultaneous zeros of polynomial systems over composite fields [22, 2], and (iii) a generalized Vandermonde determinant analysis.

They also proved an exact security classifier for $k = 2$. Both the dichotomy at $k = 2$ and its proof technique (linear algebra plus the cosets-of-subspaces structure) are the entry point for our test-matrix derivation.

The dichotomy result of [20]

A perfect-or-completely-insecure dichotomy for any linear code-based secret-sharing scheme was given by [20], over a binary extension field against arbitrary physical bit leakage, along with a complete characterization of the insecure leakages via minimal codewords of the dual of the binary image. A remark therein also explicitly extends both results to composite order fields $\mathbb{F}_{p^d}$ and to subfield-coordinate leakage, which is the present setting. The work also gives a classifying algorithm whose worst-case cost matches our verification cost (Proposition 3.12). Beyond the dichotomy, [20] gives a Monte-Carlo construction of GRS-based linear code-based secret sharing with random multipliers and fixed evaluation places, achieving $M \leq (k-1)\lambda/\mathrm{poly}(\log \lambda)$ bits of physical bit leakage tolerance with overwhelming probability over the multiplier choice.

Our contribution is complementary to [20] on the construction side. We give a construction of evaluation places $x_j = \Phi^j(x_0)$ using a suitable rational function Φ, that fixes both the multipliers (all equal to 1, as in standard Shamir's scheme) and the evaluation places to a one-parameter family, with the security question reducing (via the dichotomy of [20]) to a closed form bad-set bound on the base point x_0. The trade-off is on party count: [20] supports $n = O(dk/\log_p d)$ parties, and we support $n = O(d/\log_p d)$ parties.

Explicit constructions over prime fields

[9] constructed efficient classifiers for evaluation places over Mersenne and Fermat primes in the $n = k$ regime, connecting leakage resilience to orthogonality of square wave functions and 2-adic valuations of rational approximations. They provide explicit secure evaluation places for $n = k = 2$ and lift to $n = k > 2$ via a Fourier-analytic theorem. Their techniques are specific to prime fields, where bit extraction is a nonlinear map. Over composite fields, the analogous map is $\mathbb{F}_p$-linear, enabling a fundamentally different (and in several respects simpler) approach.

1.5 Technical overview

The proof proceeds in three phases. We outline each here.

Phase 1: The perfect dichotomy (Section 2)

The key structural observation is that the coordinate extraction map $(\cdot)_i : \mathbb{F} \to \mathbb{F}_p$, defined by $(x)_i = \mathrm{Tr}(x \cdot f_i^*)$ where $\{f_i^*\}$ is the dual basis with respect to the field trace, is $\mathbb{F}_p$-linear. It follows that the leakage map $L_{\vec{i}} : \mathbb{F}^{k-1} \to \mathbb{F}_p^n$ is $\mathbb{F}_p$-linear.

This has a clean consequence. By the structure of cosets of $\mathbb{F}_p$-subspaces, the leakage distribution for secret s is uniform over a coset of $\mathrm{Im}(L_{\vec{i}})$. Two cosets are either identical or disjoint, yielding the dichotomy $\mathrm{SD} \in \{0, 1\}$. Consequently, surjectivity of $L_{\vec{i}}$, equivalently a test matrix $\Theta_{\vec{i}}$ having full column rank n over $\mathbb{F}_p$, is a *sufficient* condition for perfect security against the pattern $\vec{i}$, since it forces every coset to equal $\mathbb{F}_p^n$. (Perfect security against $\vec{i}$ is in fact the weaker condition that the secret's coordinate vector $((s)_{i_j})_j$ lies in $\mathrm{Im}(L_{\vec{i}})$ for every s, which is the minimal codeword characterization of [20, Theorem 6]; see also [17, Theorem 5] for the $k = 2$ case. We use only the sufficient direction.) An alternative derivation of the dichotomy $\mathrm{SD} \in \{0, 1\}$ in our setting is given in Appendix A. The rank criterion is the working tool for the rest of the proof.

Phase 2: Partial-fraction non-degeneracy (Section 4)

The test matrix $\Theta_{\vec{i}}$ encodes the map $\vec{c} \mapsto (\sum_j c_j \eta^{(i_j)} x_j^\ell)_{\ell=1}^{k-1}$. To show full rank, we must prove that no nontrivial $\mathbb{F}_p$-linear combination $G_\ell(x) = \sum_j c_j \eta^{(i_j)} (\Phi^j(x))^\ell$ vanishes identically.

The argument is a partial-fraction analysis. Since the Möbius transforms $\Phi^0, \Phi^1, \ldots, \Phi^{n-1}$ have pairwise distinct poles (Corollary 3.4), the residue of G_ℓ at each pole is determined by a single summand. This pole-distinctness is the unique structural property of the orbit on which the entire argument hinges; Remark 3.6 shows that the naive alternative $\Psi(x) = \alpha x$ fails exactly here. If $G_\ell \equiv 0$, all coefficients must vanish. The resulting zero count $\leq n\ell$ at the smallest power $\ell = 1$ gives $\leq n$ bad base points per coefficient vector, and a union bound over all patterns and coefficients yields the bad set of Theorem 1.1.

Phase 3: Multi-block extension (Section 5)

For multi-block leakage, the kernel condition involves $\mathbb{F}$-valued coefficients $d_j = \sum_r c_{j,r} \eta^{(i_j^{(r)})}$ arising from within-share linear combinations. The admissibility of the leakage pattern ensures that $d_j \neq 0$ whenever the corresponding $c_{j,r}$ are not all zero. Once this is established, the partial-fraction argument extends directly to $\mathbb{F}$-coefficients.

1.6 Discussion and open problems

It is worth recording what our proofs will actually rely on regarding the step operator Φ.

(1) *Rational function structure:* each Φ^j is a degree-1 rational function, which will enable the partial-fraction analysis in Section 4.

(2) *Distinct poles:* $\Phi^0, \dots, \Phi^{n-1}$ have pairwise distinct poles in $\mathbb{P}^1(\mathbb{F})$, which is the hypothesis driving our non-degeneracy lemma.

Our proofs make no appeal to the Ramanujan spectral gap, the expander mixing lemma, character sum bounds, or any equidistribution property. These tools are available for torus-based constructions (Möbius maps on μ_{q+1}) and could potentially be brought to bear on the open problems below.

Several natural questions remain.

▶ **Open Question 1.3** (Universal multi-block for $2 \le k \le n$ and $M = \omega(d/\log_p d)$)**.** *The pattern enumeration bottleneck limits universality to $M = O(d/\log_p d)$. Can this be overcome?*

▶ **Open Question 1.4** (Polynomial-size bad set)**.** *For a fixed admissible pattern, the Vandermonde matrix of evaluation places has nontrivial kernel, and the bad-set bound $\le n \cdot p^M$ from Theorem 5.3 is exponential in M. Can this be improved to a polynomial bound?*

▶ **Open Question 1.5** (Factor-k gap)**.** *The randomized constructions of [17, 20] handle $n = O(dk/\log_p d)$ parties; our single-parameter construction handles $n = O(d/\log_p d)$. Can this be overcome?*

2 Preliminaries

Throughout, p denotes a prime and $d \ge 2$ an integer. We write $\mathbb{F} = \mathbb{F}_{p^d}$ and $\mathbb{F}^* = \mathbb{F} \setminus \{0\}$. The security parameter is $\lambda = d \log_2 p$.

2.1 The field and coordinates

Fix a primitive polynomial $\Pi(t) \in \mathbb{F}_p[t]$ of degree d. Then $\mathbb{F} = \mathbb{F}_p[t]/\Pi(t)$, and a root α of Π satisfies $\mathrm{ord}(\alpha) = p^d - 1$, i.e., α generates $\mathbb{F}^*$. The canonical basis of $\mathbb{F}$ over $\mathbb{F}_p$ is $\mathcal{B} = \{1, \alpha, \alpha^2, \dots, \alpha^{d-1}\}$. Every $x \in \mathbb{F}$ is uniquely written as $x = \sum_{i=0}^{d-1} x_i \alpha^i$ with $x_i \in \mathbb{F}_p$, and we define the i-th $\mathbb{F}_p$-coordinate of x by $(x)_i := x_i$.

▶ **Definition 2.1** (Dual basis and shifting factors)**.** *Let* $\mathrm{Tr} := \mathrm{Tr}_{\mathbb{F}/\mathbb{F}_p}$ *denote the field trace. The* dual basis *of $\mathcal{B}$ with respect to* Tr *is the unique basis $\{f_0^*, f_1^*, \dots, f_{d-1}^*\}$ of $\mathbb{F}$ over $\mathbb{F}_p$ satisfying $\mathrm{Tr}(\alpha^i \cdot f_j^*) = \delta_{ij}$ for all i, j. The* shifting factors *are $\eta^{(i)} := f_i^*/f_0^*$ for $i = 0, 1, \dots, d-1$.*

In particular, $\eta^{(0)} = 1$. The dual basis exists and is unique because the Gram matrix $G_{ij} = \mathrm{Tr}(\alpha^i \cdot \alpha^j)$ is invertible (by non-degeneracy of the trace form).

▶ **Proposition 2.2** (Shift property)**.** *For all $x \in \mathbb{F}$ and all $i \in \{0, \dots, d-1\}$,*

$$(x)_i = (x \cdot \eta^{(i)})_0.$$

Proof. We compute $(x \cdot \eta^{(i)})_0 = \mathrm{Tr}(x \cdot \eta^{(i)} \cdot f_0^*) = \mathrm{Tr}(x \cdot (f_i^*/f_0^*) \cdot f_0^*) = \mathrm{Tr}(x \cdot f_i^*) = (x)_i$. ◀

▶ **Lemma 2.3** ($\mathbb{F}_p$-independence of shifting factors)**.** *The shifting factors $\eta^{(0)}, \eta^{(1)}, \dots, \eta^{(d-1)}$ are $\mathbb{F}_p$-linearly independent. In fact, they form a basis of $\mathbb{F}$ over $\mathbb{F}_p$.*

Proof. The dual basis $\{f_0^*, \dots, f_{d-1}^*\}$ is a basis of $\mathbb{F}$ over $\mathbb{F}_p$. The map $f_i^* \mapsto \eta^{(i)} = f_i^*/f_0^*$ is multiplication by the nonzero scalar $1/f_0^* \in \mathbb{F}^*$, hence an $\mathbb{F}_p$-linear isomorphism of $\mathbb{F}$. ◀

2.2 Shamir's secret sharing and leakage models

▶ **Definition 2.4** (Shamir's scheme). *Let $2 \le k \le n$. Given distinct evaluation places $x_0, \dots, x_{n-1} \in \mathbb{F}^*$, the scheme* $\mathrm{ShamirSS}(n, k, \vec{X})_{\mathbb{F}}$ *shares a secret $s \in \mathbb{F}$ as follows: sample $P(x) = s + a_1 x + \cdots + a_{k-1} x^{k-1}$ with $a_1, \dots, a_{k-1} \in \mathbb{F}$ uniform and independent, and output shares $s_j = P(x_j)$ for $j = 0, \dots, n-1$.*

▶ **Definition 2.5** (Block leakage). *In the* single-block leakage *model, the adversary learns one $\mathbb{F}_p$-coordinate $(s_j)_{i_j} \in \mathbb{F}_p$ from each share s_j, where the block indices $\vec{i} = (i_0, \dots, i_{n-1}) \in \{0, \dots, d-1\}^n$ are adversary-chosen (non-adaptively). For $p = 2$, each $\mathbb{F}_p$-block is a single bit, so block leakage coincides with physical bit leakage.*

▶ **Definition 2.6** (Multi-block leakage). *In the* multi-block leakage *model, the adversary learns $m_j \ge 1$ distinct $\mathbb{F}_p$-coordinates from share j, at positions $i_j^{(1)}, \dots, i_j^{(m_j)} \in \{0, \dots, d-1\}$. The total number of blocks leaked is $M = \sum_{j=0}^{n-1} m_j$. A multi-block pattern is* admissible *if the block positions within each share are distinct.*

▶ **Definition 2.7** (Insecurity). *The insecurity of* $\mathrm{ShamirSS}(n, k, \vec{X})_{\mathbb{F}}$ *against a leakage family $\mathcal{F}$ is*

$$\epsilon_{\mathcal{F}}(\vec{X}) := \max_{f \in \mathcal{F}} \max_{s \in \mathbb{F}^*} \mathrm{SD}(f(\mathrm{Share}(0)),\ f(\mathrm{Share}(s))),$$

where SD *denotes statistical distance. We say the scheme is* perfectly secure *against $\mathcal{F}$ if $\epsilon_{\mathcal{F}}(\vec{X}) = 0$.*

2.3 The perfect dichotomy

We now record the structural foundation that the rest of the analysis builds on: the coordinate map $(\cdot)_i = \mathrm{Tr}(\cdot \cdot f_i^*)$ is $\mathbb{F}_p$-linear, which makes the leakage map linear, forces the leakage distribution to be uniform on a coset of an $\mathbb{F}_p$-subspace, and yields a perfect dichotomy for statistical distance.

▶ **Proposition 2.8** (Leakage map). *For block leakage with indices $\vec{i}$, the leakage vector $\vec{\ell} = ((s_j)_{i_j})_{j=0}^{n-1}$ decomposes as*

$$\ell_j = (s)_{i_j} + \sum_{t=1}^{k-1} \mathrm{Tr}(a_t \cdot x_j^t \cdot f_{i_j}^*),$$

where the map $L_{\vec{i}} : \mathbb{F}^{k-1} \to \mathbb{F}_p^n$ defined by $L_{\vec{i}}(a_1, \dots, a_{k-1})_j = \sum_{t=1}^{k-1} \mathrm{Tr}(a_t \cdot x_j^t \cdot f_{i_j}^*)$ is $\mathbb{F}_p$-linear. (Proof in Appendix A.)

▶ **Proposition 2.9** (Perfect dichotomy). *For block leakage with indices $\vec{i}$ and any two secrets $s, s' \in \mathbb{F}$:*

$$\mathrm{SD}(\vec{\ell}|_s,\ \vec{\ell}|_{s'}) = \begin{cases} 0 & \text{if } \vec{v}(s) - \vec{v}(s') \in \mathrm{Im}(L_{\vec{i}}) \\ 1 & \text{if } \vec{v}(s) - \vec{v}(s') \notin \mathrm{Im}(L_{\vec{i}}) \end{cases}$$

where $\vec{v}(s) = ((s)_{i_0}, \dots, (s)_{i_{n-1}})$. In particular, if $L_{\vec{i}}$ is surjective, then $\mathrm{SD} = 0$ *for all pairs of secrets. (Proof in Appendix A.)*

▶ Remark 2.10 (More general result in previous work). [20, Theorem 6] establishes Proposition 2.9 in much greater generality: for any linear code-based secret-sharing scheme over an extension field $\mathbb{F}_{p^\lambda}$, against arbitrary physical bit leakage. By a remark therein, the result also applies to composite order fields $\mathbb{F}_{p^d}$ and to subfield-coordinate leakage, which is our present setting. The $k = 2$ case is also implicit, via linear algebra, in [17, Theorem 5]. For completeness, proofs of Propositions 2.8 and 2.9 in our setting are deferred to Appendix A; the rest of the paper depends only on the conclusion of Proposition 2.9.

2.4 Surjectivity and the test matrix

▶ **Proposition 2.11** (Dual characterization). *The map $L_{\vec{i}}$ is surjective if and only if the adjoint $L_{\vec{i}}^* : \mathbb{F}_p^n \to \mathbb{F}^{k-1}$ is injective. The adjoint is given by*

$$(L_{\vec{i}}^*(\vec{c}))_\ell = f_0^* \cdot \sum_{j=0}^{n-1} c_j \cdot \eta^{(i_j)} \cdot x_j^\ell, \qquad \ell = 1, \dots, k-1.$$

Since $f_0^* \neq 0$, injectivity of $L_{\vec{i}}^*$ is equivalent to: for all $\vec{c} \in \mathbb{F}_p^n \setminus \{\vec{0}\}$, there exists $\ell \in \{1, \dots, k-1\}$ with $\sum_j c_j \eta^{(i_j)} x_j^\ell \neq 0$.

Proof. The first claim is standard: for an $\mathbb{F}_p$-linear map $L : V \to W$ between finite-dimensional $\mathbb{F}_p$-vector spaces, L is surjective if and only if its adjoint L^* is injective. This follows from $\dim(\mathrm{Im}(L)) = \dim(V) - \dim(\ker(L))$ and $\dim(\ker(L^*)) = \dim(W) - \dim(\mathrm{Im}(L))$, so $\mathrm{Im}(L) = W$ iff $\ker(L^*) = \{0\}$.

For the formula, we compute the adjoint explicitly. The relevant inner products are the standard $\mathbb{F}_p$-inner product $\langle \vec{c}, \vec{y} \rangle_{\mathbb{F}_p} = \sum_j c_j y_j$ on $\mathbb{F}_p^n$, and the trace pairing $\langle \vec{a}, \vec{b} \rangle = \sum_{\ell=1}^{k-1} \mathrm{Tr}(a_\ell b_\ell)$ on $\mathbb{F}^{k-1}$. For any $\vec{a} \in \mathbb{F}^{k-1}$ and $\vec{c} \in \mathbb{F}_p^n$:

$$\begin{aligned} \langle \vec{c},\, L_{\vec{i}}(\vec{a}) \rangle_{\mathbb{F}_p} &= \sum_{j=0}^{n-1} c_j \sum_{t=1}^{k-1} \mathrm{Tr}(a_t x_j^t f_{i_j}^*) \\ &= \sum_{t=1}^{k-1} \mathrm{Tr}\left(a_t \sum_{j=0}^{n-1} c_j x_j^t f_{i_j}^* \right), \end{aligned}$$

where we used $\mathbb{F}_p$-linearity of Tr and the fact that $c_j \in \mathbb{F}_p$ can be moved inside Tr. Now substituting $f_{i_j}^* = \eta^{(i_j)} \cdot f_0^*$:

$$= \sum_{t=1}^{k-1} \mathrm{Tr}\left(a_t \cdot f_0^* \cdot \sum_{j=0}^{n-1} c_j \eta^{(i_j)} x_j^t \right) = \left\langle \vec{a},\, \left(f_0^* \cdot \sum_{j=0}^{n-1} c_j \eta^{(i_j)} x_j^\ell \right)_{\ell=1}^{k-1} \right\rangle.$$

Since this holds for all $\vec{a}$, we conclude $(L_{\vec{i}}^*(\vec{c}))_\ell = f_0^* \cdot \sum_j c_j \eta^{(i_j)} x_j^\ell$.

Finally, since $f_0^* \neq 0$ (it is a dual basis element), $L_{\vec{i}}^*(\vec{c}) = 0$ in $\mathbb{F}^{k-1}$ if and only if $\sum_j c_j \eta^{(i_j)} x_j^\ell = 0$ for every $\ell = 1, \dots, k-1$. ◀

▶ **Definition 2.12** (Test matrix). *For block indices $\vec{i} \in \{0, \dots, d-1\}^n$, the* test matrix *$\Theta_{\vec{i}} \in \mathbb{F}_p^{d(k-1) \times n}$ is the $\mathbb{F}_p$-coordinate representation of the map*

$$\vec{c} \mapsto \left(\sum_{j=0}^{n-1} c_j \eta^{(i_j)} x_j^\ell \right)_{\ell=1}^{k-1} \in \mathbb{F}^{k-1} \cong \mathbb{F}_p^{d(k-1)}.$$

▶ **Corollary 2.13.** *$L_{\vec{i}}$ is surjective if and only if $\mathrm{rank}_{\mathbb{F}_p}(\Theta_{\vec{i}}) = n$. Since $\Theta_{\vec{i}}$ has only $d(k-1)$ rows, this (sufficient) full-rank condition can hold only when $n \leq d(k-1)$, the regime in which we work.*

Proof. By Proposition 2.11, $L_{\vec{i}}$ is surjective if and only if $L_{\vec{i}}^*$ is injective, which holds if and only if the $\mathbb{F}_p$-linear map encoded by $\Theta_{\vec{i}}$ has trivial kernel, i.e., $\mathrm{rank}_{\mathbb{F}_p}(\Theta_{\vec{i}}) = n$. Since $\Theta_{\vec{i}}$ has $d(k-1)$ rows, full column rank requires $n \leq d(k-1)$. ◀

3 The construction

We now define the step operator and establish the properties of the orbit $\{x_j = \Phi^j(x_0)\}$ that will be needed for the security analysis. The main point is that three properties – rational function structure, distinct poles, and a short-period bound – follow from elementary algebra, with no conditions to verify beyond $x_0 \notin \{\alpha - 1\} \cup \mathrm{Poles}(n)$.

3.1 The step operator

▶ **Definition 3.1** (Step operator). *Define* $\Phi : \mathbb{P}^1(\mathbb{F}) \to \mathbb{P}^1(\mathbb{F})$ *by*

$$\Phi(x) = \frac{\alpha x}{x+1},$$

where α *is the generator of* $\mathbb{F}^*$ *from Section 2.1. As a Möbius transformation,* Φ *has matrix* $M_\Phi = \begin{pmatrix} \alpha & 0 \\ 1 & 1 \end{pmatrix}$ *with* $\det(M_\Phi) = \alpha \neq 0$.

▶ **Proposition 3.2** (Projective order). *The projective order of* Φ *equals* $\mathrm{ord}(\alpha) = p^d - 1$.

Proof. The eigenvalues of $M_\Phi = \begin{pmatrix} \alpha & 0 \\ 1 & 1 \end{pmatrix}$ are α and 1 (reading off the diagonal, since M_Φ is lower triangular). Hence M_Φ^j has eigenvalues α^j and 1. Now $\Phi^j = \mathrm{id}$ in $\mathrm{PGL}_2(\mathbb{F})$ if and only if $M_\Phi^j = \lambda I$ for some $\lambda \in \mathbb{F}^*$ (since projective transformations are defined up to scalar multiples of the matrix). The condition $M_\Phi^j = \lambda I$ requires both eigenvalues to be equal, i.e., $\alpha^j = 1$. Conversely, if $\alpha^j = 1$, then both eigenvalues equal 1 and M_Φ^j must be the identity (since M_Φ is diagonalizable over the algebraic closure: its eigenvalues α and 1 are distinct for $d \geq 2$, so M_Φ, and hence M_Φ^j, is diagonalizable). The smallest positive j with $\alpha^j = 1$ is $\mathrm{ord}(\alpha) = p^d - 1$. ◀

▶ **Proposition 3.3** (Explicit formula for iterates). *For* $j \geq 1$,

$$\Phi^j(x) = \frac{\alpha^j x}{C_j x + 1}, \qquad C_j := \frac{\alpha^j - 1}{\alpha - 1} = \sum_{i=0}^{j-1} \alpha^i.$$

For $j = 0$, *we have* $\Phi^0(x) = x$.

Proof. By induction on j. For $j = 1$: $C_1 = 1$ and $\Phi(x) = \alpha x/(x+1)$. For the inductive step:

$$\Phi^{j+1}(x) = \Phi(\Phi^j(x)) = \frac{\alpha \cdot \frac{\alpha^j x}{C_j x + 1}}{\frac{\alpha^j x}{C_j x + 1} + 1} = \frac{\alpha^{j+1} x}{(\alpha^j + C_j)x + 1}.$$

It remains to verify that $\alpha^j + C_j = C_{j+1}$. Indeed, $\alpha^j + (\alpha^j - 1)/(\alpha - 1) = (\alpha^{j+1} - \alpha^j + \alpha^j - 1)/(\alpha - 1) = (\alpha^{j+1} - 1)/(\alpha - 1) = C_{j+1}$. ◀

3.2 Distinct poles, fixed points, and pole avoidance

▶ **Corollary 3.4** (Pairwise distinct poles). *For* $0 \leq j \leq n - 1$ *with* $n \leq p^d - 1$, *the Möbius transforms* $\Phi^0, \Phi^1, \ldots, \Phi^{n-1}$ *have pairwise distinct poles in* $\mathbb{P}^1(\mathbb{F})$. *More precisely, the pole of* $\Phi^0 = \mathrm{id}$ *is* ∞, *and the pole of* Φ^j *for* $j \geq 1$ *is* $-(\alpha - 1)/(\alpha^j - 1)$.

Proof. By Proposition 3.3, the pole of Φ^j for $j \geq 1$ is at $x = -1/C_j = -(\alpha - 1)/(\alpha^j - 1)$. For $1 \leq j < j' \leq n - 1$, these coincide if and only if $\alpha^j = \alpha^{j'}$, i.e., $j \equiv j' \pmod{p^d - 1}$. Since $0 < j < j' < p^d - 1$, this is impossible. The pole of $\Phi^0 = \mathrm{id}$ is ∞, which is distinct from all finite poles. ◀

▶ Remark 3.5 (No involution). The map Φ is never an involution for $d \geq 2$, regardless of characteristic. Indeed, $\Phi^2 = \mathrm{id}$ would require $\alpha^2 = 1$, but $\mathrm{ord}(\alpha) = p^d - 1 \geq p^2 - 1 \geq 3$. In particular, no special treatment is needed for $p = 2$.

▶ Remark 3.6 (A naive dilation fails). Deriving motivation from constructions of polynomial codes with optimal list decoding properties (folded Reed-Solomon codes [8]), the most obvious one-parameter candidate is the pure dilation $\Psi(x) = \alpha x$, whose iterates $\Psi^j(x) = \alpha^j x$ form an orbit $x_j = \alpha^j x_0$ on the multiplicative coset $x_0\langle\alpha\rangle$. We record here that this choice fails the structural property driving our analysis, and consequently does not admit the bad-set bound of Theorem 4.5.

The functions Ψ^j are affine, so each has its only pole at ∞. Hence all n iterates share a single common pole, and the pole-distinctness hypothesis of Corollary 3.4 fails completely (rather than failing for at most one j). The partial-fraction argument of Section 4 therefore has no traction: the rational function $G_\ell(x) = \sum_j c_j \eta^{(i_j)} (\alpha^j x)^\ell = x^\ell \cdot \sum_j c_j \eta^{(i_j)} \alpha^{j\ell}$ is a monomial in x, and $G_\ell \equiv 0$ is equivalent to the x_0-independent condition $\sum_j c_j \eta^{(i_j)} \alpha^{j\ell} = 0$. The starting point x_0 has dropped out of the rank condition entirely. In other words, the test matrix $\Theta_{\vec{i}}$ is the same matrix for every $x_0 \in \mathbb{F}^*$, so the dilation scheme is either secure for all nonzero x_0 or for none – there is no parameter to optimize and no closed-form bad-set bound to prove. In particular, the central quantitative result $|\mathrm{Bad}| \leq n + n(dp)^n$ of Theorem 4.5 has no analogue for Ψ.

The role of the $+1$ in the denominator of Φ is thus to perturb each iterate's pole away from ∞ to a distinct finite point of $\mathbb{F}^*$ (Corollary 3.4), placing the orbit in generic position relative to the partial-fraction decomposition. This is the essential algebraic feature that distinguishes Φ from its affine cousin and makes the closed-form bad-set bound possible.

▶ **Lemma 3.7** (Fixed points). *For each $1 \leq j < p^d - 1$, the fixed points of Φ^j in $\mathbb{P}^1(\mathbb{F})$ are exactly $\{0, \alpha - 1\}$. In particular,* $\mathrm{ShortPeriod}(n) := \{x_0 \in \mathbb{F}^* : \exists\, 1 \leq j < n,\ \Phi^j(x_0) = x_0\} = \{\alpha - 1\}$.

Proof. Fix $1 \leq j < p^d - 1$, so that $\Phi^j \neq \mathrm{id}$ by Proposition 3.2. The fixed-point equation $\Phi^j(x) = x$ reads $\alpha^j x/(C_j x + 1) = x$. For $x = 0$: $\Phi^j(0) = 0$, so 0 is a fixed point, but $0 \notin \mathbb{F}^*$. For $x \neq 0$: dividing both sides by x gives $\alpha^j/(C_j x + 1) = 1$, hence $x = (\alpha^j - 1)/C_j$. Since $C_j = (\alpha^j - 1)/(\alpha - 1)$, this gives $x = \alpha - 1$. For $x = \infty$: $\Phi^j(\infty) = \alpha^j/C_j \neq \infty$ (since $C_j \neq 0$ for $j < p^d - 1$). The fixed-point set $\{0, \alpha - 1\}$ is independent of j, so restricting to $\mathbb{F}^*$ gives $\mathrm{ShortPeriod}(n) = \{\alpha - 1\}$. ◀

▶ **Definition 3.8** (Pole set). *Define* $\mathrm{Poles}(n) := \{-1/C_j : 1 \leq j \leq n - 1\} \subset \mathbb{F}^*$. *These are pairwise distinct by Corollary 3.4, so* $|\mathrm{Poles}(n)| = n - 1$.

We note that $\mathrm{Poles}(n) \cap \{\alpha - 1\} = \emptyset$*: the equality* $\alpha - 1 = -(\alpha - 1)/(\alpha^j - 1)$ *would require* $\alpha^j = 0$*, which is impossible.*

▶ **Proposition 3.9** (Distinct evaluation places). *For $x_0 \in \mathbb{F}^* \setminus (\{\alpha - 1\} \cup \mathrm{Poles}(n))$ and $n \leq p^d - 1$, the evaluation places $X_0, X_1, \ldots, X_{n-1}$ are pairwise distinct elements of $\mathbb{F}^*$.*

Proof. We verify three properties. (a) *Each $x_j \in \mathbb{F}^*$:* $x_j = 0$ if and only if $x_0 = 0$ (since 0 is a fixed point of Φ), which is excluded by $x_0 \in \mathbb{F}^*$. $x_j = \infty$ if and only if $x_0 = -1/C_j \in \mathrm{Poles}(n)$, which is excluded by hypothesis. (b) *The orbit avoids $\alpha - 1$:* Since $\alpha - 1$ is a fixed point of Φ itself (not just of Φ^j), we have $\Phi^{-1}(\{\alpha - 1\}) = \{\alpha - 1\}$, and hence $x_j = \alpha - 1$ if and only if $x_0 = \alpha - 1$, which is excluded. (c) *Pairwise distinct:* $x_j = X_{j'}$ for $j < j'$ if and only if x_j is a fixed point of $\Phi^{j'-j}$ in $\mathbb{F}^*$. By Lemma 3.7, the only such point is $\alpha - 1$, and by (b), $x_j \neq \alpha - 1$. ◀

Algorithm 1 CLASSIFY(p, d, k, n, x_0): single-block-leakage classifier for the Möbius orbit $x_j = \Phi^j(x_0)$.

Input: prime p, integer $d \geq 2$, reconstruction threshold $k \geq 2$, number of parties $n \leq d(k-1)$, field representation $\mathbb{F} = \mathbb{F}_p[\alpha]$, candidate base point $x_0 \in \mathbb{F}^*$.
Output: GOOD (certifies perfect security), or BAD$(\vec{i})$ where $\vec{i}$ is a pattern at which the sufficient full-rank certificate fails.

if $x_0 \in \{\alpha - 1\} \cup \mathrm{Poles}(n)$ **then**
  return BAD$(\perp)$ `// structural exclusion`
end
Compute $x_j \leftarrow \Phi^j(x_0)$ for $j = 0, \ldots, n-1$
Compute the dual basis $\{f_0^*, \ldots, f_{d-1}^*\}$ and shifting factors $\eta^{(i)} = f_i^*/f_0^*$
for $\vec{i} \in \{0, \ldots, d-1\}^n$ **do**
  Build $\Theta_{\vec{i}} \in \mathbb{F}_p^{d(k-1)\times n}$ as in Definition 2.12
  if $\mathrm{rank}_{\mathbb{F}_p}(\Theta_{\vec{i}}) < n$ **then**
    return BAD$(\vec{i})$
  end
end
return GOOD

▶ Remark 3.10. The total number of structural exclusions is $|\{\alpha - 1\} \cup \mathrm{Poles}(n)| = n$.

▶ Remark 3.11 (Canonical base point). The generator α is a natural canonical candidate for the base point. Two layers of the goodness condition should be distinguished.

Provably: α avoids the structural exclusion set $\{\alpha - 1\} \cup \mathrm{Poles}(n)$. Indeed, $\alpha \neq \alpha - 1$ (since $1 \neq 0$), and $\alpha \in \mathrm{Poles}(n)$ would require $\alpha^{j+1} = 1$ for some $1 \leq j \leq n-1$, which is impossible whenever $n \leq p^d - 2$. Verification is $O(n)$.

Not provably without verification: whether α avoids the security-bad set in the headline single-block regime of Theorem 4.5 requires running the classifier of Algorithm 1.

3.3 Classifier and choice of base point

We now give an explicit classifier that, given a candidate base point x_0, tests the *sufficient* full-rank condition of Corollary 2.13 for the resulting evaluation places $x_j = \Phi^j(x_0)$: for each block-index tuple $\vec{i} \in \{0, \ldots, d-1\}^n$ it checks whether the test matrix $\Theta_{\vec{i}}$ has full column rank n over $\mathbb{F}_p$, returning GOOD only if all of them do. A GOOD verdict *certifies* perfect security against single-block leakage, so the classifier is a *sound* tester. A BAD $(\vec{i})$ verdict reports a pattern at which the full-rank certificate fails; since full rank is sufficient but not necessary (Corollary 2.13), this does not by itself certify insecurity. On the structured Möbius candidates we propose, this sound test is what we use to certify goodness; we do not claim it decides perfect security on arbitrary evaluation places, for which the exact criterion is the containment/minimal-codeword condition of [20, Theorem 6].

▶ **Proposition 3.12** (Classifier complexity). *Algorithm 1 runs in time $O(d^n \cdot n^2 \cdot dk)$ field operations over $\mathbb{F}_p$.*

Proof. The structural-exclusion check is $O(n)$. Computing the orbit $x_0, \ldots, x_{n-1}$ and the shifting factors is $\mathrm{poly}(d, n, \log p)$, dominated by the main loop. The loop iterates d^n times. Each iteration builds a $d(k-1) \times n$ matrix over $\mathbb{F}_p$ (whose entries are $\mathbb{F}_p$-coordinates of $\eta^{(i_j)} x_j^\ell$, each computable in $\mathrm{poly}(d, k, \log p)$ field operations) and computes its rank in $O(n^2 \cdot d(k-1)) = O(n^2 dk)$ field operations. No enumeration over coefficient vectors $\vec{c} \in \mathbb{F}_p^n$ is needed, since the rank test handles all $\vec{c}$ per pattern simultaneously. ◀

Algorithm 1 matches the cost of the classifying algorithm given by Nguyen [20, Fig. 1] on the corresponding inputs. Our contribution in this regard is therefore not a faster verification algorithm; rather, the classifier exists here as a tool to certify the structured Möbius candidates we propose, complementing the closed form bad-set bounds.

4 Single-block security

The goal of this section is to prove that for all but a small number of base points x_0, the test matrix $\Theta_{\vec{i}}$ has full column rank for *every* block-index tuple $\vec{i}$. The main tool is a partial-fraction non-degeneracy lemma, which exploits the fact that the Möbius transforms $\Phi^0, \dots, \Phi^{n-1}$ have pairwise distinct poles.

4.1 Non-degeneracy via partial fractions

▶ **Lemma 4.1** (Non-degeneracy). *Let $\psi_0, \dots, \psi_{n-1}$ be Möbius transforms on $\mathbb{P}^1(\mathbb{F})$ with pairwise distinct poles. Let $\delta_0, \dots, \delta_{n-1} \in \mathbb{F}^*$, $c_0, \dots, c_{n-1} \in \mathbb{F}_p$ with $\vec{c} \neq \vec{0}$, and $\ell \geq 1$. Then*

$$G_\ell(x) := \sum_{j=0}^{n-1} c_j \delta_j (\psi_j(x))^\ell$$

is not identically zero as a rational function on $\mathbb{P}^1(\mathbb{F})$.

Proof. Write $\psi_j(x) = (A_j x + B_j)/(C_j x + D_j)$ with $A_j D_j - B_j C_j \neq 0$. The poles $p_j = -D_j/C_j$ (for $C_j \neq 0$) or $p_j = \infty$ (for $C_j = 0$) are pairwise distinct by hypothesis, so at most one equals ∞.

Case A. All poles are finite ($C_j \neq 0$ for all j). Near the pole p_j, $\psi_j(x)$ has a simple pole with residue $r_j = -(A_j D_j - B_j C_j)/C_j^2 \neq 0$. Consequently, $(\psi_j(x))^\ell$ has a pole of order ℓ at p_j with leading Laurent coefficient $r_j^\ell \neq 0$. Since the poles are pairwise distinct, the leading term of G_ℓ at p_j comes only from the j-th summand: $G_\ell(x) \sim c_j \delta_j r_j^\ell/(x - p_j)^\ell$ as $x \to p_j$. If $G_\ell \equiv 0$, then $c_j \delta_j r_j^\ell = 0$ for all j. Since $\delta_j \neq 0$ and $r_j \neq 0$, this forces $c_j = 0$ for all j, contradicting $\vec{c} \neq \vec{0}$.

Case B. One pole is at ∞ (say $C_0 = 0$, so that ψ_0 is affine; all other $C_j \neq 0$). The residue argument at each finite pole p_j ($j \geq 1$) gives $c_j \delta_j r_j^\ell = 0$, hence $c_j = 0$ for all $j \geq 1$. Then $G_\ell(x) = c_0 \delta_0 (\psi_0(x))^\ell$. Since ψ_0 is nonconstant ($A_0/D_0 \neq 0$ because $\det(M_0) \neq 0$ and $C_0 = 0$), $\delta_0 \neq 0$, and $c_0 \neq 0$ (as $\vec{c} \neq \vec{0}$), this is not identically zero.

At most one pole equals ∞, so Cases A and B are exhaustive. ◀

▶ **Corollary 4.2** (Zero bound). *Under the hypotheses of Lemma 4.1, G_ℓ has at most $n\ell$ zeros in $\mathbb{P}^1(\mathbb{F})$.*

Proof. Write $\psi_j(x) = (A_j x + B_j)/(C_j x + D_j)$. Let $S = \{j : C_j \neq 0\}$ and $T = \{j : C_j = 0\}$, so $|T| \leq 1$ (at most one pole can be ∞). Define the common denominator

$$D(x) = \prod_{j \in S} (C_j x + D_j)^\ell,$$

and set $N(x) = G_\ell(x) \cdot D(x)$. We compute the degree of $N(x)$.

For $j \in S$: the j-th summand of G_ℓ is $c_j \delta_j (A_j x + B_j)^\ell/(C_j x + D_j)^\ell$. After multiplying by $D(x)$, the denominator $(C_j x + D_j)^\ell$ cancels, contributing to $N(x)$ the term $c_j \delta_j (A_j x + B_j)^\ell \prod_{j' \in S,\, j' \neq j} (C_{j'} x + D_{j'})^\ell$, which has degree $\ell + \ell(|S| - 1) = \ell |S|$.

For $j \in T$: we have $C_j = 0$, so $\psi_j(x) = (A_j/D_j)x + B_j/D_j$ is affine (with $A_j/D_j \neq 0$). The j-th summand of G_ℓ is $c_j\delta_j((A_j/D_j)x + B_j/D_j)^\ell$, a polynomial of degree ℓ. After multiplying by $D(x) = \prod_{j' \in S}(C_{j'}x + D_{j'})^\ell$, which has degree $\ell|S|$, the contribution is a polynomial of degree $\ell + \ell|S|$.

Since $|S| + |T| = n$: if $T = \emptyset$, then $|S| = n$ and $\deg(N) \leq \ell n$. If $|T| = 1$, then $|S| = n - 1$ and $\deg(N) \leq \max(\ell(n-1),\ \ell + \ell(n-1)) = \ell n$. In both cases, $\deg(N) \leq n\ell$.

Since $G_\ell \not\equiv 0$ by Lemma 4.1, $N \not\equiv 0$, so N has at most $\deg(N) \leq n\ell$ roots (counted with multiplicity) in $\mathbb{F}$. The zeros of G_ℓ in $\mathbb{P}^1(\mathbb{F})$ are a subset of the roots of N: indeed, any root of D that is also a root of N must be a pole of some summand of G_ℓ, hence a pole of G_ℓ (since the poles are distinct, there is no cancellation), not a zero. Therefore G_ℓ has at most $n\ell$ zeros in $\mathbb{P}^1(\mathbb{F})$. ◀

4.2 Full rank of the test matrix

▶ **Theorem 4.3.** *Let p be any prime, $d \geq 2$, and $2 \leq k \leq n \leq d(k-1)$ with $n \leq p^d - 1$. For any $\vec{i} \in \{0, \ldots, d-1\}^n$ and $\vec{c} \in \mathbb{F}_p^n \setminus \{\vec{0}\}$, we have*

$$|\{x_0 \in \mathbb{F}^* : \sum_{j=0}^{n-1} c_j \eta^{(i_j)}(\Phi^j(x_0))^\ell = 0 \ \ \forall\, \ell = 1, \ldots, k-1\}| \leq n.$$

Proof. Apply Lemma 4.1 with $\ell = 1$, $\psi_j = \Phi^j$, and $\delta_j = \eta^{(i_j)}$. By Corollary 3.4, the Φ^j have pairwise distinct poles. The shifting factors $\eta^{(i_j)}$ are nonzero (being ratios of dual basis elements). So $G_1(x) = \sum_j c_j \eta^{(i_j)} \Phi^j(x) \not\equiv 0$, and by Corollary 4.2, G_1 has at most n zeros in $\mathbb{P}^1(\mathbb{F})$. The bad set is contained in $\{x_0 \in \mathbb{F}^* : G_1(x_0) = 0\}$, which has size $\leq n$. ◀

▶ Remark 4.4. The bound $\leq n$ comes from the $\ell = 1$ case alone. The bad set requires $G_\ell = 0$ for *all* ℓ, and the containment in the zero set of G_1 already suffices.

▶ **Theorem 4.5** (Perfect security against single-block leakage)**.** *Let p be any prime, $d \geq 2$, and $2 \leq k \leq n \leq d(k-1)$ with $n \leq p^d - 1$. Define*

$$\text{Bad} := \{\alpha - 1\} \cup \text{Poles}(n) \ \cup \bigcup_{\vec{i} \in \{0,\ldots,d-1\}^n} \ \bigcup_{\vec{c} \in \mathbb{F}_p^n \setminus \{\vec{0}\}} \{x_0 \in \mathbb{F}^* : G_1(x_0; \vec{i}, \vec{c}) = 0\}.$$

Then $|\text{Bad}| \leq n + n(dp)^n$. For any $x_0 \in \mathbb{F}^* \setminus \text{Bad}$:
(a) *$x_0, \ldots, x_{n-1}$ are pairwise distinct elements of $\mathbb{F}^*$.*
(b) *$\text{ShamirSS}(n, k, \vec{X})_{\mathbb{F}}$ is perfectly secure against all single-block leakage.*

Proof. Part (a) follows from Proposition 3.9, since $x_0 \notin \{\alpha - 1\} \cup \text{Poles}(n)$.

For part (b): by the security reduction (Corollary 2.13), we need $\text{rank}_{\mathbb{F}_p}(\Theta_{\vec{i}}) = n$ for every block-index tuple $\vec{i} \in \{0, \ldots, d-1\}^n$. By Proposition 2.11, this fails for $\vec{i}$ if and only if there exists $\vec{c} \in \mathbb{F}_p^n \setminus \{\vec{0}\}$ with $\sum_j c_j \eta^{(i_j)} x_j^\ell = 0$ for all $\ell = 1, \ldots, k-1$. For each such pair $(\vec{i}, \vec{c})$, Theorem 4.3 bounds the set of bad base points by $\leq n$.

We now count. The structural exclusion set $\{\alpha - 1\} \cup \text{Poles}(n)$ has size n (by Remark 3.10). The number of block-index tuples is $|\{0, \ldots, d-1\}^n| = d^n$. The number of nonzero coefficient vectors is $|\mathbb{F}_p^n \setminus \{\vec{0}\}| = p^n - 1$. By Theorem 4.3, each pair contributes at most n bad base points. By the union bound:

$$|\text{Bad}| \leq n + d^n \cdot (p^n - 1) \cdot n \leq n + n \cdot d^n \cdot p^n = n + n(dp)^n.$$

For any $x_0 \in \mathbb{F}^* \setminus \text{Bad}$: by construction, x_0 avoids the structural exclusion set (ensuring part (a)) and, for every $\vec{i}$ and every nonzero $\vec{c}$, there exists ℓ with $G_\ell(x_0; \vec{i}, \vec{c}) \neq 0$. This means $\Theta_{\vec{i}}$ has trivial kernel for every $\vec{i}$, i.e., full column rank n. By Corollary 2.13, $L_{\vec{i}}$ is surjective for all $\vec{i}$, and by Proposition 2.9, SD $= 0$ for all pairs of secrets and all block-leakage patterns. ◀

▶ **Corollary 4.6** (Existence). *A good x_0 exists whenever $p^d - 1 > n + n(dp)^n$, i.e., $d > n(1 + \log_p d) + O(\log_p n)$, giving $n = O(d/\log_p d)$.*

4.3 Physical bit security

▶ **Proposition 4.7** (Block security implies sub-block security). *Let $S_0, \ldots, S_{n-1}$ be arbitrary nonempty finite sets. If $\text{ShamirSS}(n, k, \vec{X})_{\mathbb{F}}$ is perfectly secure against single-block leakage, then it is also perfectly secure against any leakage that applies an arbitrary function $g_j : \mathbb{F}_p \to S_j$ to a single $\mathbb{F}_p$-block of each share.*

Proof. By Theorem 4.5, for every block-index tuple $(i_0, \ldots, i_{n-1}) \in \{0, \ldots, d-1\}^n$, the map $L_{\vec{i}}$ is surjective. By Proposition 2.9, this means the leakage vector $((P(X_0))_{i_0}, \ldots, (P(X_{n-1}))_{i_{n-1}}) \in \mathbb{F}_p^n$ has the same distribution for every secret $s \in \mathbb{F}$: it is uniform over $\mathbb{F}_p^n$.

Now suppose the adversary applies deterministic functions $g_j : \mathbb{F}_p \to S_j$ to obtain $(g_0((P(X_0))_{i_0}), \ldots, g_{n-1}((P(X_{n-1}))_{i_{n-1}}))$. Since each g_j is a fixed function, this post-processed leakage is a deterministic function of $((P(x_j))_{i_j})_{j=0}^{n-1}$. By the data processing inequality for statistical distance, for any two secrets s, s':

$$\text{SD}\big((g_j((P(x_j))_{i_j}))_j \,\big|\, s,\ \ (g_j((P(x_j))_{i_j}))_j \,\big|\, s'\big) \leq \text{SD}\big(((P(x_j))_{i_j})_j \,\big|\, s,\ \ ((P(x_j))_{i_j})_j \,\big|\, s'\big) = 0.$$

Hence perfect security is preserved under arbitrary post-processing. ◀

▶ **Corollary 4.8** (Physical bit security, all characteristics). *For any prime p: if the adversary leaks one physical bit per share (from an arbitrary $\mathbb{F}_p$-block, possibly different blocks for different shares), our evaluation places provide perfect security.*

Proof. A physical bit $\text{bit}_{r_j}((P(x_j))_{i_j})$ is a function $g_j : \mathbb{F}_p \to \{0, 1\}$ applied to a single block. The result follows from Proposition 4.7. ◀

5 Multi-block security

We now extend the analysis to the multi-block setting, where the adversary probes multiple $\mathbb{F}_p$-coordinates from each share.

5.1 Setup

For an admissible multi-block leakage pattern with $M = \sum_{j=0}^{n-1} m_j \leq d(k-1)$ total blocks, the leakage map $L : \mathbb{F}^{k-1} \to \mathbb{F}_p^M$ is $\mathbb{F}_p$-linear. The test matrix $\Theta \in \mathbb{F}_p^{d(k-1)\times M}$ has columns indexed by pairs (share j, block $i_j^{(r)}$).

Full column rank $\text{rank}_{\mathbb{F}_p}(\Theta) = M$ is a *sufficient* condition for perfect security, and is the criterion we establish below; as in the single-block case (Corollary 2.13) it need not be necessary.

The structure of the kernel is as follows. A nonzero vector $(c_{j,r})_{j,r} \in \ker(\Theta)$ groups naturally by shares, producing $d_j = \sum_{r=1}^{m_j} c_{j,r}\eta^{(i_j^{(r)})} \in \mathbb{F}$. By admissibility, the shifting factors within each share are $\mathbb{F}_p$-independent, so $d_j = 0$ if and only if all of the corresponding $c_{j,r}$ vanish. In particular, at least one d_j must be nonzero.

▶ Remark 5.1 (Admissibility is necessary). Since any $m_j \leq d$ distinct blocks from the same share have $\mathbb{F}_p$-independent shifting factors (by Lemma 2.3), any pattern with distinct block positions within each share is admissible.

5.2 Non-degeneracy with field coefficients

▶ **Lemma 5.2.** *Let $\psi_0, \ldots, \psi_{n-1}$ be Möbius transforms on $\mathbb{P}^1(\mathbb{F})$ with pairwise distinct poles. Let $d_0, \ldots, d_{n-1} \in \mathbb{F}$ with at least one $d_j \neq 0$, and $\ell \geq 1$. Then $\sum_j d_j(\psi_j(x))^\ell \not\equiv 0$ as a rational function, and has at most $n\ell$ zeros in $\mathbb{P}^1(\mathbb{F})$.*

Proof. Write $\psi_j(x) = (A_j x + B_j)/(C_j x + D_j)$ with $\det(M_j) = A_j D_j - B_j C_j \neq 0$, and let p_j denote the pole of ψ_j. The poles are pairwise distinct by hypothesis. Let $S = \{j : C_j \neq 0\}$ (finite poles) and $T = \{j : C_j = 0\}$ (pole at ∞), so $|T| \leq 1$.

Case A. All poles are finite ($T = \emptyset$). For each j, the residue of ψ_j at its pole $p_j = -D_j/C_j$ is $r_j = -\det(M_j)/C_j^2 \neq 0$. Near p_j, we have $\psi_j(x) = r_j/(x - p_j) + O(1)$, so $(\psi_j(x))^\ell = r_j^\ell/(x - p_j)^\ell + \text{lower-order terms}$. Since $p_j \neq p_{j'}$ for $j \neq j'$, the function $\sum_j d_j(\psi_j(x))^\ell$ has a Laurent expansion near p_j whose leading term is $d_j r_j^\ell/(x - p_j)^\ell$; all other summands are regular at p_j. If $\sum_j d_j(\psi_j(x))^\ell \equiv 0$, this leading coefficient must vanish: $d_j r_j^\ell = 0$. Since $r_j \neq 0$, we get $d_j = 0$ for every $j \in S$. With $T = \emptyset$, this gives $d_j = 0$ for all j, contradicting the hypothesis.

Case B. One pole is at ∞ (say $j = 0$, so $C_0 = 0$; all other $C_j \neq 0$). The residue argument at each finite pole p_j ($j \geq 1$) gives $d_j r_j^\ell = 0$, hence $d_j = 0$ for all $j \geq 1$. Then $\sum_j d_j(\psi_j(x))^\ell = d_0(\psi_0(x))^\ell$. Since $C_0 = 0$, $\psi_0(x) = (A_0/D_0)x + B_0/D_0$ is a nonconstant affine function ($A_0/D_0 \neq 0$ because $\det(M_0) = A_0 D_0 \neq 0$). Thus $d_0(\psi_0(x))^\ell$ is a nonzero polynomial (since $d_0 \neq 0$ by hypothesis, as all other d_j vanish).

The zero bound $\leq n\ell$ follows by the same degree argument as in Corollary 4.2: multiply $\sum_j d_j(\psi_j(x))^\ell$ by $D(x) = \prod_{j \in S}(C_j x + D_j)^\ell$ to obtain a nonzero polynomial $N(x)$ of degree $\leq n\ell$. ◀

5.3 Multi-block theorems

▶ **Theorem 5.3** (Multi-block, fixed pattern)**.** *Fix an admissible multi-block leakage pattern with $M \leq d(k-1)$ total blocks. The set of $x_0 \in \mathbb{F}^*$ for which the scheme is not perfectly secure has size $\leq n + n \cdot p^M$. A good x_0 exists when $M < d - \log_p(2n)$.*

Proof. A nonzero vector $(c_{j,r})_{j,r} \in \mathbb{F}_p^M \setminus \{\vec{0}\}$ lies in $\ker(\Theta)$ if and only if

$$\sum_{j=0}^{n-1} \sum_{r=1}^{m_j} c_{j,r} \eta^{(i_j^{(r)})} x_j^\ell = 0 \quad \text{for all } \ell = 1, \ldots, k-1.$$

Grouping the inner sum by shares, this becomes $\sum_{j=0}^{n-1} d_j x_j^\ell = 0$ for all ℓ, where $d_j := \sum_{r=1}^{m_j} c_{j,r} \eta^{(i_j^{(r)})} \in \mathbb{F}$. By admissibility, the shifting factors $\eta^{(i_j^{(1)})}, \ldots, \eta^{(i_j^{(m_j)})}$ within share j are $\mathbb{F}_p$-independent (Lemma 2.3), so $d_j = 0$ if and only if $c_{j,r} = 0$ for all r. Since $\vec{c} \neq \vec{0}$, at least one $d_j \neq 0$.

By Lemma 5.2 with $\ell = 1$ and $\psi_j = \Phi^j$: the function $G_1(x) = \sum_j d_j \Phi^j(x)$ is not identically zero and has at most n zeros in $\mathbb{P}^1(\mathbb{F})$. The set of $x_0 \in \mathbb{F}^*$ with $G_1(x_0) = 0$ therefore has size $\leq n$.

There are $p^M - 1$ nonzero coefficient vectors in $\mathbb{F}_p^M$. By the union bound over all such vectors, plus the n structural exclusions:

$$|\text{Bad}| \leq n + (p^M - 1) \cdot n \leq n + n \cdot p^M.$$

A good x_0 exists when $|\mathbb{F}^*| = p^d - 1 > n + n \cdot p^M$, which holds when $p^d > 2n \cdot p^M$, i.e., $M < d - \log_p(2n)$. ◀

▶ **Theorem 5.4** (Universal multi-block). *To achieve perfect security against all admissible patterns with $\leq M$ blocks simultaneously, we get $|\text{Bad}| \leq n + n \cdot (dpe)^M$. A good x_0 exists when $d > M\left(\frac{5}{2} + \log_p d\right) + \log_p(2n)$.*

Proof. We take a union bound over all admissible multi-block patterns with $\leq M$ blocks total. An admissible pattern is specified by choosing, for each share j, a nonempty subset of $\{0, \ldots, d-1\}$ of block positions. The total number of such patterns is at most $\binom{d}{m_0} \cdots \binom{d}{m_{n-1}}$ summed over all $(m_0, \ldots, m_{n-1})$ with $\sum m_j \leq M$. Also note that by definition of admissible patterns, we have $m_0, \ldots, m_{n-1} \geq 1$ and $M \geq n$. This gives us the total number as

$$\sum_{\substack{m_0+\cdots+m_{n-1} \leq M \\ m_0,\ldots,m_{n-1} \geq 1}} \binom{d}{m_0} \cdots \binom{d}{m_{n-1}} \leq \sum_{t=n}^{M} \binom{nd}{t} \leq \left(\frac{nde}{M}\right)^M \leq (de)^M.$$

where the first inequality is by the Cauchy-Vandermonde identity [11, Exercise 1.9], the second inequality is by standard estimate on binomial coefficient [11, Proposition 1.4], and the third inequality is due to $M \geq n$.

For each fixed pattern, Theorem 5.3 gives at most $n \cdot p^M$ security-bad base points (beyond the structural exclusions). Taking the union bound over all $\leq (de)^M$ patterns, we get

$$|\text{Bad}| \leq n + (n \cdot p^M) \cdot (de)^M = n + n(dpe)^M.$$

A good x_0 exists when $p^d - 1 > n + n(dpe)^M$. Taking logarithms base p, we need $d > \log_p(2n) + M(\log_p d + 1 + \log_p e)$, which is satisfied when $d > M\left(\frac{5}{2} + \log_p d\right) + \log_p(2n)$. ◀

References

1. Luís T. A. N. Brand ao and René Peralta. NIST first call for multi-party threshold schemes, 2023. URL: `https://csrc.nist.gov/publications/detail/nistir/8214c/draft`.
2. Mitali Bafna, Madhu Sudan, Santhoshini Velusamy, and Dain Xiang. Elementary analysis of isolated zeroes of a polynomial system, 2021. arXiv Preprint. `doi:10.48550/arXiv.2102.00602`.
3. Fabrice Benhamouda, Akshay Degwekar, Yuval Ishai, and Tal Rabin. On the local leakage resilience of linear secret sharing schemes. In *Advances in Cryptology – CRYPTO 2018*, volume 10991 of *LNCS*, pages 531–561. Springer, 2018. `doi:10.1007/978-3-319-96884-1_18`.
4. Fabrice Benhamouda, Akshay Degwekar, Yuval Ishai, and Tal Rabin. On the local leakage resilience of linear secret sharing schemes. *Journal of Cryptology*, 34(2):10, 2021. `doi:10.1007/s00145-021-09375-2`.
5. Nicolas Costes and Martijn Stam. Redundant code-based masking revisited. *IACR Transactions on Cryptographic Hardware and Embedded Systems (TCHES)*, 2021(1):426–450, 2021. `doi:10.46586/tches.v2021.i1.426-450`.
6. Sebastian Faust, Loïc Masure, Elena Micheli, Maximilian Orlt, and François-Xavier Standaert. Connecting leakage-resilient secret sharing to practice: Scaling trends and physical dependencies of prime field masking. In *Advances in Cryptology – EUROCRYPT 2024*, pages 316–344. Springer, 2024. `doi:10.1007/978-3-031-58737-5_12`.

7 Vipul Goyal and Ashutosh Kumar. Non-malleable secret sharing. In *50th ACM Symposium on Theory of Computing (STOC 2018)*, pages 685–698, 2018. doi:10.1145/3188745.3188872.

8 Venkatesan Guruswami and Atri Rudra. Explicit codes achieving list decoding capacity: Error-correction with optimal redundancy. *IEEE Transactions on Information Theory*, 54(1):135–150, 2008. doi:10.1109/TIT.2007.911222.

9 Jihun Hwang, Hemanta K. Maji, Hai H. Nguyen, and Xiuyu Ye. Leakage-resilience of Shamir's secret sharing: Identifying secure evaluation places. In *6th Conference on Information-Theoretic Cryptography (ITC 2025)*, LIPIcs, 2025. doi:10.4230/LIPIcs.ITC.2025.3.

10 Yuval Ishai, Amit Sahai, and David Wagner. Private circuits: Securing hardware against probing attacks. In *Advances in Cryptology – CRYPTO 2003*, volume 2729 of *LNCS*, pages 463–481. Springer, 2003. doi:10.1007/978-3-540-45146-4_27.

11 Stasys Jukna. *Extremal combinatorics: with applications in computer science*, volume 571. Springer, 2011. doi:10.1007/978-3-642-17364-6.

12 Ohad Klein and Ilan Komargodski. New bounds on the local leakage resilience of Shamir's secret sharing scheme. In *Advances in Cryptology – CRYPTO 2023*, volume 14081 of *LNCS*, pages 139–170. Springer, 2023. doi:10.1007/978-3-031-38557-5_5.

13 Hiroki Koga and Hiroto Abe. New tight bounds on the local leakage resilience of the additive (n, n)-threshold scheme determined by the eigenvalues of circulant matrices. In Goichiro Hanaoka and Bo-Yin Yang, editors, *Advances in Cryptology – ASIACRYPT 2025*, pages 34–66, Singapore, 2026. Springer Nature Singapore. doi:10.1007/978-981-95-5125-5_2.

14 Ashutosh Kumar, Raghu Meka, and Amit Sahai. Leakage-resilient secret sharing against colluding parties. In *2019 IEEE 60th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 636–660, 2019. doi:10.1109/FOCS.2019.00045.

15 Hemanta K. Maji, Hai H. Nguyen, Anat Paskin-Cherniavsky, Tom Suad, and Mingyuan Wang. Leakage-resilience of the Shamir secret-sharing scheme against physical-bit leakages. In *Advances in Cryptology – EUROCRYPT 2021*, volume 12697 of *LNCS*, pages 344–374. Springer, 2021. doi:10.1007/978-3-030-77886-6_12.

16 Hemanta K. Maji, Hai H. Nguyen, Anat Paskin-Cherniavsky, and Mingyuan Wang. Improved bound on the local leakage-resilience of shamir's secret sharing. In *2022 IEEE International Symposium on Information Theory (ISIT)*, pages 2678–2683, 2022. doi:10.1109/ISIT50566.2022.9834695.

17 Hemanta K. Maji, Hai H. Nguyen, Anat Paskin-Cherniavsky, and Xiuyu Ye. Constructing leakage-resilient Shamir's secret sharing: Over composite order fields. In *Advances in Cryptology – EUROCRYPT 2024*, volume 14654 of *LNCS*, pages 286–315. Springer, 2024. doi:10.1007/978-3-031-58737-5_11.

18 Hemanta K. Maji, Anat Paskin-Cherniavsky, Tom Suad, and Mingyuan Wang. Constructing locally leakage-resilient linear secret-sharing schemes. In *Advances in Cryptology – CRYPTO 2021*, volume 12827 of *LNCS*, pages 779–808. Springer, 2021. doi:10.1007/978-3-030-84252-9_26.

19 Hai H. Nguyen. Towards breaking the half-barrier for Shamir's secret sharing scheme. In *Advances in Cryptology – CRYPTO 2024*, 2024. doi:10.1007/978-3-031-68388-6_10.

20 Hai H. Nguyen. Physical-bit leakage resilience of linear code-based secret sharing. In Serge Fehr and Pierre-Alain Fouque, editors, *Advances in Cryptology – EUROCRYPT 2025*, pages 64–93, Cham, 2025. Springer Nature Switzerland. doi:10.1007/978-3-031-91101-9_3.

21 Adi Shamir. How to share a secret. *Communications of the ACM*, 22(11):612–613, 1979. doi:10.1145/359168.359176.

22 Xiaomei Zhao. A note on multiple exponential sums in function fields. *Finite Fields and Their Applications*, 18(1):35–55, 2012. doi:10.1016/j.ffa.2011.06.003.

A Alternative proof of the dichotomy

The perfect-or-completely-insecure dichotomy used throughout this paper (Proposition 2.9) is a special case of the general dichotomy result of [20]. For completeness, we record here a direct derivation in our setting, working through the leakage map $L_{\vec{i}}$ and the cosets-of-subspaces

structure. The proof depends only on $\mathbb{F}_p$-linearity of coordinate extraction and on Shamir's $\mathbb{F}$-linearity in the randomness; it does not use any property of generalized Reed–Solomon codes or of binary images.

Proof of Proposition 2.8. Since $P(x_j) = s + \sum_{t=1}^{k-1} a_t x_j^t$, the $\mathbb{F}_p$-linearity of the coordinate map gives

$$\ell_j = (P(x_j))_{i_j} = (s)_{i_j} + \sum_{t=1}^{k-1} (a_t x_j^t)_{i_j}.$$

By the definition of coordinates via the dual basis (Proposition 2.2 and Definition 2.1), we have $(a_t x_j^t)_{i_j} = \mathrm{Tr}(a_t x_j^t \cdot f_{i_j}^*)$. This gives the decomposition $\ell_j = (s)_{i_j} + \sum_{t=1}^{k-1} \mathrm{Tr}(a_t \cdot x_j^t \cdot f_{i_j}^*)$.

It remains to verify that $L_{\vec{i}}$ is $\mathbb{F}_p$-linear. For any $\lambda \in \mathbb{F}_p$ and $\vec{a}, \vec{b} \in \mathbb{F}^{k-1}$:

$$L_{\vec{i}}(\lambda\vec{a} + \vec{b})_j = \sum_{t=1}^{k-1} \mathrm{Tr}((\lambda a_t + b_t) \cdot x_j^t \cdot f_{i_j}^*) = \lambda \sum_{t=1}^{k-1} \mathrm{Tr}(a_t x_j^t f_{i_j}^*) + \sum_{t=1}^{k-1} \mathrm{Tr}(b_t x_j^t f_{i_j}^*),$$

where we used $\mathrm{Tr}(\lambda \cdot y) = \lambda \cdot \mathrm{Tr}(y)$ for $\lambda \in \mathbb{F}_p$ (since the trace is $\mathbb{F}_p$-linear) and $\mathrm{Tr}(y + z) = \mathrm{Tr}(y) + \mathrm{Tr}(z)$. Thus $L_{\vec{i}}(\lambda\vec{a} + \vec{b}) = \lambda L_{\vec{i}}(\vec{a}) + L_{\vec{i}}(\vec{b})$. ◀

Proof of Proposition 2.9. Since $\vec{a} = (a_1, \ldots, a_{k-1})$ is uniform over $\mathbb{F}^{k-1}$ and $L_{\vec{i}}$ is $\mathbb{F}_p$-linear, the random variable $L_{\vec{i}}(\vec{a})$ is uniform over the subspace $\mathrm{Im}(L_{\vec{i}}) \subseteq \mathbb{F}_p^n$. By Proposition 2.8, the leakage for secret s is $\vec{v}(s) + L_{\vec{i}}(\vec{a})$, which is uniform over the coset $\vec{v}(s) + \mathrm{Im}(L_{\vec{i}})$. Two cosets of an $\mathbb{F}_p$-subspace of $\mathbb{F}_p^n$ are either identical (when the difference of the shifts lies in the subspace, giving $\mathrm{SD} = 0$) or disjoint (giving $\mathrm{SD} = 1$, since uniform distributions on disjoint sets of equal size have statistical distance 1). If $L_{\vec{i}}$ is surjective, then $\mathrm{Im}(L_{\vec{i}}) = \mathbb{F}_p^n$, so every coset equals $\mathbb{F}_p^n$. ◀