Improved Alternating-Moduli PRFs and Post-Quantum Signatures

Navid Alamati* Guru-Vamsi Policharla† Srinivasan Raghuraman‡ Peter Rindal§

Abstract

We revisit the alternating-moduli paradigm for constructing symmetric-key primitives with a focus on constructing efficient protocols to evaluate them using secure multi-party computation (MPC). The alternating-moduli paradigm of Boneh, Ishai, Passèlegue, Sahai, and Wu (TCC 2018) enables the construction of various symmetric-key primitives with the common characteristic that the inputs are multiplied by two linear maps over different moduli.

The first contribution focuses on efficient two-party evaluation of alternating-moduli pseudorandom functions (PRFs), effectively building an oblivious PRF. We present a generalized alternating-moduli PRF construction along with methods to lower the communication and computation. We then provide several variants of our protocols with different computation and communication tradeoffs for evaluating the PRF. Most of our protocols are in the hybrid model while one is based on specialized garbling. Our most efficient protocol effectively is about $3\times$ faster and requires $1.3\times$ less communication.

Our next contribution is the efficient evaluation of the one-way function (OWF) $f(x) = \mathbf{B} \cdot_3 (\mathbf{A} \cdot_2 x)$ proposed by Dinur, Goldfeder, Halevi, Ishai, Kelkar, Sharma, and Zaverucha (CRYPTO 21) where $\mathbf{A} \in \mathbb{F}_2^{m \times n}$, $\mathbf{B} \in \mathbb{F}_3^{t \times m}$, and $\cdot_p$ is multiplication mod p. This surprisingly simple OWF can be evaluated within MPC by secret sharing $[\![x]\!]$ over $\mathbb{F}_2$, locally computing $[\![v]\!] = \mathbf{A} \cdot_2 [\![x]\!]$, performing a modulus switching protocol to $\mathbb{F}_3$ shares, followed by locally computing the output shares $[\![y]\!] = \mathbf{B} \cdot_3 [\![v]\!]$.

We design a bespoke MPC-in-the-Head (MPCitH) signature scheme that evaluates the aforementioned OWF, achieving state-of-the-art performance. The resulting signature has a size ranging from 4.0 to 5.5 KB, achieving between 2-3$\times$ reduction compared to the prior work. To the best of our knowledge, this is only $\approx 5\%$ larger than the smallest signature based on symmetric-key primitives, including the latest NIST post-quantum cryptography competition submissions. We also show that our core techniques can be extended to build very small post-quantum ring signatures for rings of small to medium size, which are competitive with state-of-the-art lattice-based schemes. Our techniques are in fact more generally applicable to set membership in MPCitH.

*VISA Research.
†UC Berkeley. Part of the work was done while the author was an intern at VISA Research.
‡VISA Research and MIT.
§VISA Research.

Contents

1 Introduction **3**
- 1.1 Our Contributions . . . 4
- 1.2 Overview . . . 5

2 Preliminaries **7**

3 Symmetric-Key Primitives from Alternating Moduli **9**
- 3.1 Prior Constructions and Their Shortcomings . . . 9
- 3.2 Insecure Plus/XOR Construction . . . 11
- 3.3 Our Constructions . . . 12
- 3.4 Optimizations . . . 13
- 3.5 Security Analysis and Parameter Selection . . . 14
- 3.6 Properties of the AM-OWF . . . 16
  - 3.6.1 Variants of the AM-OWF . . . 17
  - 3.6.2 Combinatorial Analysis . . . 18

4 Protocols and Evaluation **20**
- 4.1 Review of Silent OT/VOLE/OLE . . . 20
- 4.2 Secret-Sharing-Based $\mathbb{F}_3 \to \mathbb{F}_2$ Modulus Conversion . . . 21
- 4.3 Secret-Sharing-Based $(\mathbb{F}_3, \mathbb{F}_2)$-wPRF Protocol . . . 23
- 4.4 OT-Based $(\mathbb{F}_2, \mathbb{F}_3)$-wPRF Protocol . . . 26
- 4.5 Specialized Garbling . . . 26
- 4.6 Protocol Evaluation . . . 27

5 Post-Quantum Signatures and Ring Signatures **30**
- 5.1 Post-Quantum Ring Signatures . . . 35
- 5.2 Size of Ring Signature . . . 36

1 Introduction

We revisit the line of work on building basic cryptographic primitives such as one-way functions (OWFs) [Lev85], pseudorandom generators (PRGs) [BM82, Yao82], and pseudorandom functions (PRFs) [GGM84] that allow efficient evaluation in a distributed setting. Traditionally, instantiations of these basic cryptographic primitives have taken two approaches as described in [ABG+14]. The first is the *reductionist* approach where the security of the construction can be reduced to a "standard" set of well-studied computational assumptions such as the hardness of factoring, discrete logarithm, or learning with errors (LWE). The second is a *direct construction* approach where the construction itself is considered to be the computational assumption. Indeed, many of today's most widely used primitives follow the direct approach (e.g., AES, SHA).

The reductionist approach comes with several benefits—it improves our theoretical understanding of how various computational hard problems are related and allows the cryptographic community to focus its cryptanalytic efforts on a small set of assumptions. With the test of time, newer assumptions become widely accepted and eventually considered standard by the community. Unfortunately, the benefit of this approach often comes at a large cost to efficiency: often resulting in orders of magnitude worse efficiency compared to the direct construction approach (consider the efficiency gap between SHA-256 and discrete-logarithm-based hash functions). Moreover, with the looming threat of quantum computers, many of the existing standard assumptions are at risk of becoming insecure which, in turn, motivates the investigation of new cryptographic assumptions.

The security of the direct or new constructions, while often not reducible to a standard assumption, are often based on a set of principles that are developed and refined as the problem is studied. The canonical example of this is linear and differential cryptanalysis of symmetric-key cryptography. Other examples include early lattice-based constructions such as NTRU [HPS98]. In many cases, these schemes first developed security principles which, in time, were refined into a small set of underlying (standard) assumptions.

An additional benefit of direct constructions is that they allow for an added degree of flexibility to conform to a changing set of requirements. This is particularly true in this work where we are focused on designing and using symmetric-key primitives that are extremely efficient to evaluate in the multi-party setting. Indeed, it seems unlikely that assumptions and constructions designed for a wildly different set of constraints would be ideal for this setting. An added advantage of the direct approach is that this flexibility often allows the construction to lack the algebraic structure that a quantum computer could exploit. In some cases algebraic techniques such as Gröbner bases can still apply. However, this still requires exploiting specific structure in the construction which is not always present. This is contrasted by traditional assumptions such as factoring or discrete logarithm which have a large amount of structure. With the increased use of secure multi-party computation (MPC) and the looming approach of quantum computers, there has been a need to consider a new set of assumptions that meet a new set of requirements. In particular, primitives such as ring signatures [RST01], oblivious pseudorandom functions (OPRFs) [NR97, FIPR05], verifiable random functions (VRFs) [MRV99], blind signatures [Cha82], and more lack efficient constructions that meet one or more of these requirements. This work is focused on using new symmetric-key techniques and novel protocol designs to implement these applications using MPC, or the zero-knowledge compiler known as MPC-in-the-head [IKOS07].

MPC-friendliness. There are two main categories of efficient MPC protocols, based on garbling (round optimal but large communication overhead) and based on linear secret sharing (communication efficient but round complexity grows with the depth of the circuit). Garbling-based approaches typically have prohibitively high overheads making them impractical for most applications as also noted in [DGH+21]. Ideally, we want to design primitives that can be evaluated in just one round trip (the minimum required) using protocols based on linear secret sharing.

There has been a long line of work in this direction, with a focus on modifying existing symmetric-key primitives to make them *MPC-friendly* [ARS+15, GRR+16, DEG+18, AGP+19, DGGK21, GØSW23]. Despite making progress towards MPC-friendliness, these constructions still suffer from a large communication overhead and/or large round complexity [BIP+18]. To understand the reason behind the unsatisfactory progress in constructing MPC-friendly symmetric-key primitives, it helps to understand the high-level strategy underlying the cryptanalysis of symmetric-key primitives. A popular approach is differential cryptanalysis [BS91] for block ciphers which analyses the effect that a change in the input has on the output. By making sufficiently many queries, one may be able to later distinguish the output from a uniformly random string. The *depth* of the function is correlated with the difficulty of building a

distinguisher and low-depth functions are expected to be less secure, conflicting with the goal of MPC-friendliness.

Deep yet shallow. A key observation made by [BIP+18] is that the notions of *depth* required by MPC and security against cryptanalysis are in fact very different. Starting with this observation, they propose the alternating-moduli paradigm which mixes *linear functions* over different moduli. One can build concretely efficient MPC protocols to evaluate such functions, where the number of rounds needed only depends on the number of piecewise linear functions. The same function when expressed as a polynomial over a single modulus has a much higher degree, making it resistant to cryptanalysis. This key observation allowed them to build a depth-2 weak PRF (wPRF) which can be evaluated in just one round trip, given a preprocessing phase. Their construction has mostly resisted initial cryptanalysis, with some (easily fixable) attacks requiring a large number of samples found in [CCKK21]. This work was followed by the work of Dinur *et al.* [DGH+21] who proposed a new OWF, PRG, and a weak PRF based on the same paradigm and showed that one could construct efficient MPC protocols to evaluate them leading to a very efficient round-optimal oblivious pseudorandom function. They also showed that the OWF could be used to build a post-quantum signature scheme with good concrete efficiency albeit still larger than the state-of-the-art using symmetric-key primitives.

Good but not great. While prior works designed efficient protocols based on the OWF/wPRFs that were proposed, they still either fell short of beating the state-of-the-art or suffered from other limitations. Despite being able to design cryptographic primitives from the ground up, it is unsatisfactory that these protocols are not "the best." In particular, the signature scheme from [DGH+21] has a much larger size than the state-of-the-art 4-7 KB. More advanced primitives such as ring signatures and verifiable random functions have remained relatively unexplored. Moreover, the 2PC protocols for evaluating these symmetric-key functions required significant time to generate the required correlated randomness while the main phase of these protocols have more communication than one would have desired.

1.1 Our Contributions

We emphasize that the goal of this work is to revisit the alternating-moduli paradigm and show that when protocols are carefully designed, such that they exploit the structure of alternating-moduli primitives, they can indeed achieve state-of-the-art performance. In light of the limitations discussed above, we make the following contributions.

New candidate wPRF. We investigate bottlenecks in the (weak) PRF candidates and propose a new candidate in Section 3. Our wPRF requires less communication and effectively a third of the number of oblivious transfers (OTs) when evaluated in MPC. While the work of [DGH+21] proposed a protocol with good performance for the main phase, it omitted relatively large cost of generating correlated randomness. The careful design of our new wPRF candidate optimizes the end-to-end cost of the protocols while at the same time achieving better performance in the main phase. Our construction allows one to instantiate it with $O(\lambda)$ amortized evaluation time while prior works [BIP+18, DGH+21] mandated $O(\lambda^2)$ time. While previous works [BIP+18, DGH+21] have primarily focused on using linear secret sharing, and dismissed garbling-based approaches, we show that using specialized garbling schemes leads to competitive protocols that offer an interesting trade-off between the computation and communication.

New cryptanalysis and generalized wPRF candidate. We present new cryptanalysis of different constructions in Section 3. Similar to [BIP+18], we show that the hardness of our wPRFs is connected to the hardness of solving sparse multivariate polynomials over $\mathbb{F}_3$, or in its dual form, the hardness of interpolating sparse multilinear polynomials. This analysis suggests a generalized construction based on solving a system of sparse multilinear polynomial equations over a small finite field and might, with time, form the basis of an underlying cryptographic assumption.

Fastest OPRF and wPRF protocols. In Section 4 we describe our wPRF protocols and report their performance metrics in Section 4.6. When compared to the prior alternating-moduli wPRF of [DGH+21], our implementation is an order of magnitude faster, due in part to the implementation itself and the structure of the new wPRF. Compared to commonly used LowMC construction, we observe that our protocols have far fewer rounds of interaction (2 versus 14 to 88) and are 3 to 20 times faster. Indeed, even compared to DDH-based OPRF protocols, our protocols are an

order of magnitude faster and require only slightly more communication. Concretely, our fastest protocol requires just 2 rounds, 7.7 microseconds, and 100 bytes of communication in the amortized setting.

Small post-quantum signatures from symmetric-key assumptions. We begin Section 5 by revisiting the digital signature scheme from [DGH+21] and build a specialized MPC-in-the-Head (MPCitH) protocol targeting the same OWF, giving us signature sizes ranging from 4.0 to 5.5 KB. To the best of our knowledge, this gives the second smallest signature (only 5% larger than the smallest) based on symmetric-key/MPCitH techniques.[1] This also addresses an implicit open question in [DGH+21] about whether a specialized proof system for the alternating-moduli OWF can lead to better performance. In this process, we also offer additional insights into the security of the alternating-moduli OWF, which serves as a useful guideline when introducing additional structure.

Smallest post-quantum ring signatures. In Section 5.1, we extend our digital signature scheme to a ring signature scheme by introducing a simple yet powerful technique to prove disjunctions of the same relation in MPCitH. Our ring signature grows linearly in the size of the ring, but for small to medium-sized rings, we are concretely smaller than the state-of-the-art. For larger rings, most MPCitH-based signatures, including ours, can be combined with the compiler by Goel *et al.* [GGHAK22] to build ring signatures whose size only grows logarithmically in the ring size.

1.2 Overview

We propose a new weak PRF in the alternating-moduli paradigm [BIP+18]. For $n, m, t = O(\lambda)$, our construction is defined as

$$F(k, x) := \mathbf{B} \cdot_3 (\mathbf{A} \cdot_2 [k \odot_2 x])$$

where $x, k \in \mathbb{F}_2^n$ are the input and key, $\mathbf{A} \in \mathbb{F}_2^{m \times n}$ is a random matrix, $\mathbf{B} \in \mathbb{F}_3^{t \times m}$ is a compressing random matrix, and $\cdot_p, \odot_p$ are multiplication and component-wise multiplication modulo p. This differs from the prior constructions [BIP+18] which can be defined as $F(\mathbf{A}, x) := \mathbf{B}(\mathbf{A}x)$ where the matrix $\mathbf{A}$ is interpreted as the key. We conjecture that the core hardness of these constructions stems from three components.

1. The input and key are non-linearly combined modulo two.
2. Modulo two summations of subsets are taken.
3. Modulus conversion is followed by a public compressing linear map $\mathbf{B}$.

Observe that both our construction and the construction of [BIP+18] follow these three phases. In particular, the matrix-vector multiplication of [BIP+18] can be viewed as performing (1) component-wise multiplication with each row of the matrix, followed by (2) summation. When viewed this way, observe that the summations in (2) are over disjoint sets, i.e., $\sum_j (A_i \odot x)_j$. This is in contrast with our construction where we take random summations over a common combined input vector $(k \odot x)$. In a sense our construction is reusing the hidden variable of $k_i x_i$ many times while [BIP+18] uses it in a single term.

When interpreted in the MPC context, recall that linear operations are essentially free while multiplications require communication.[2] The number of multiplication terms of [BIP+18] is proportional to the size of $\mathbf{A}$, i.e., $O(\lambda^2)$, and as such, $O(\lambda^2)$ communication is required. The work of [DGH+21] gave specialized protocols for the wPRF of [BIP+18] which reduced the communication complexity to $O(\lambda)$ in the amortized setting. In particular, [DGH+21] proposed modifying the key matrix $\mathbf{A}$ to be circulant, i.e., each row is a shift of the previous, which allows one efficiently multiply $\mathbf{K} \cdot x$ for many x with an amortized $O(\lambda)$ communication overhead. However, we show that the process of generating the correlated randomness for this multiplication still requires $O(\lambda^2)$ work with relatively high constants.

In contrast, our construction only performs $n = O(\lambda)$ multiplications in step (1) and therefore can completely sidestep the need for this expensive correlated randomness. Moreover, we show how the parties can use the fact that

[1] We compared against submissions to the most recent NIST call for additional post-quantum digital signature schemes. See Post-Quantum signatures zoo for easy comparison.

[2] That is, given a modulo p secret sharing of $v \in \mathbb{F}_p^n$ and public $\mathbf{M} \in \mathbb{F}_p^{m \times n}$, the parties can compute $\mathbf{M} \cdot_p v$ without any communication. However, multiplication of two secret-shared scalars requires communication.

the key is typically static to generate reuseable correlated randomness (i.e., n OTs) to compute shares of $(k \odot x)$ for an unbounded number of x. The main complexity of our protocols is how shares of $\mathbf{A}(k \odot x)$ are converted from modulo 2 to modulo 3. We show that this can be done with $m = O(\lambda)$ communication and m OTs which can be generated with sublinear communication using pseudoradnom correlation generators [BCG+19b, RRT23]. Compared to [DGH+21], our techniques require less correlated randomness. The final setup of our protocol is for the parties to locally apply the $\mathbb{F}_3$-linear map $\mathbf{B}$ to their $\mathbb{F}_3^m$ secret sharing of $(\mathbf{A} \cdot (k \odot x))$. To improve efficiency, we propose instantiating $\mathbf{A}$ and $\mathbf{B}$ such that multiplication can be done in linear time while maintaining their desired security properties.

We give two additional variants of this protocol. The first switches the order of the moduli. That is, the wPRF is defined as $F(k, x) = \mathbf{B} \cdot_2 (\mathbf{A} \cdot_3 [k \odot_3 x])$. The resulting protocol is conceptually similar. The primary advantage of this protocol is for MPC applications where the output should be in binary secret sharing format [MRR20, BDG+22]. Our second variant is based on a specialized garbling scheme. The main advantage of this protocol is that it does not require any OT correlations when performing modulus conversion. However, it comes at the expense of requiring $O(\lambda^2)$ communication to compute $(k \odot x)$. We believe this construction is interesting when only a few evaluations are performed. For details on our protocols we refer to Section 4. We also implement our MPC protocols and report their performance in Section 4.6. Overall, we observe a 3 to 20 times reduction in running time compared to alternatives such as [DGH+21, ARS+15] while substantially reducing the communication overhead. Indeed, compared to the DDH-based OPRF [Mea86], our construction is an order of magnitude faster with comparable communication.

Beyond proposing the new wPRF and the associated MPC protocols, we adapt existing techniques to determine parameters for the wPRF. Compared to [BIP+18, DGH+21], we observe that the parameters n, m, t must be increased a moderate amount. However, we give techniques to mitigate this increased parameter size, and in some cases even allow for less communication overhead. Our analysis suggests that the most relevant attack on the construction is a reduction to subset sum [DGH+21]. In particular, given an input-output pair $(x, y = F(k, x))$, one can define the intermediate vector $w := \mathbf{A}(k \odot x)$. Viewing $\mathbf{A}$ as a linear code and letting $\mathbf{P}$ be the associated parity check matrix, the adversary has the constrains that $\mathbf{P}w = 0$ and $\mathbf{B}w = y$. One can solve for w using subset sum solvers which have running time $O(2^{0.33m})$. For some of our parameter regimes we show that the adversary must consider multiple samples and present a new extension of subset sum to this setting. We refer to Section 3.5 for details.

Lastly, we build upon the work of [BIP+18] to show that the hardness of our construction essentially boils down to the hardness of solving sparse multivariate polynomials. In particular, we present an equivalent representation of our wPRF and show that a key recovery attack for the construction directly corresponds to solving a system of sparse multilinear equations over $\mathbb{F}_3$. This also leads us to a generalized framework for instantiating alternating-moduli weak PRFs where the distribution of input space can be varied to capture various constructions.

Signatures. Given that the alternating-moduli OWF (AM-OWF) was designed with the efficient evaluation within MPC in mind, i.e., low non-linear depth but high algebraic degree, it is natural to try to build a signature scheme using the MPC-in-the-head paradigm. In particular, the AM-OWF of [DGH+21] is defined as

$$f(x) := \mathbf{B} \cdot_3 (\mathbf{A} \cdot_2 x)$$

for some $n, m, t \in O(\lambda)$ where $x \in \mathbb{F}_2^n$ is the input, $\mathbf{A} \in \mathbb{F}_2^{m \times n}$ is an expanding random matrix, and $\mathbf{B} \in \mathbb{F}_3^{t \times m}$ is a compressing random matrix. The hardness of inverting this function stems from the non-linearity of changing the modulus between multiplying by $\mathbf{A}$ and $\mathbf{B}$. One interpretation of this problem is that $\mathbf{B}e$ is an LPN instance with correlated noise vector $e = \mathbf{A}x$.

To build a signature in the MPCitH paradigm, the verification key is the output $y = f(x) = \mathbf{B}(\mathbf{A}x)$ on some uniformly random input x, and a signature is simply a proof of knowledge of a preimage x. Indeed, this was the exact approach taken in the prior work [DGH+21], where they used the KKW proof system [KKW18] resulting in signatures with sizes ranging from 10.3 to 13.3 KB, which are still 2-3× larger than the best known signatures from symmetric-key assumptions. The issue is that the KKW proof system has a generic way of handling the preprocessing material needed for the MPC, that is oblivious to the function being evaluated. These checks on preprocessed material account for a significant chunk of the overall proof size.

The MPCitH protocol that we design is very close to the KKW proof system, except for the way in which we handle preprocessing checks. Instead of using a cut-and-choose strategy, we use an idea from [CCJ23] that has the prover first commit to the inputs, and (possibly maliciously generated) preprocessing material. The verifier then permutes

the preprocessing material and forces the prover to use this ordering when executing the online phase of the MPC protocol. We then show that for any choice of (incorrect) preprocessing material, a malicious prover who does not know a valid preimage, has a very low probability of producing an accepting proof. Although the high level ideas are borrowed from the prior work, the concrete hard problem that we consider is very different and hence demands a completely separate, non-trivial analysis. Instead of using a ball-and-bins analysis as done in [CCJ23], we view the problem through the lens of error correcting codes, which enables us to give much cleaner bounds in comparison to the prior work which relied on conjectures that were confirmed for their parameters by explicitly computing them via python scripts (Section 3.6.2).

Ring signatures. A ring signature allows a party to sign a message while remaining anonymous amongst a chosen set of (say) ℓ parties. Given a signature scheme, there is a generic way to construct ring signatures by providing a zero-knowledge proof for the statement:

> *"I know a signature σ on the message m that verifies under a public key pk_i for some $i \in [\ell]$."*

The challenge, however, lies in minimizing the concrete overhead introduced on top of a single signature, when trying to prove membership of the public key $\mathsf{pk}_i \in \{\mathsf{pk}_1, \ldots, \mathsf{pk}_\ell\}$. The state-of-the-art in post-quantum ring signatures are lattice-based schemes [LNS21, ESZ22], which build concretely efficient zero-knowledge proofs for set membership adapted to the lattice setting.

The only competing alternative appears to be based on MPCitH [GGHAK22], where the ring signature size grows as $O(\log \ell)$ but they are concretely worse than [LNS21, ESZ22]. This can be attributed to two main factors:

- They use Picnic [CDG+17] as the core signature scheme which is quite large ≈ 42 KB when instantiated with NIST L5 parameters.[3]
- Although the signature size is $O(\ell)$, the concrete constants are still quite high for small to medium-sized rings.

The former issue can be handled easily by replacing Picnic with either the signature scheme we propose or another newer and smaller MPCitH-based signature scheme (see Figure 9). This would yield concretely good signatures that are competitive with lattice-based schemes at large ring sizes. However, we observe that in many practical scenarios, small to medium-sized rings are used. For instance, in the ring signatures protocol used by Monero as part of RingCT, the number of public keys used in the anonymity set was only very recently upgraded from 11 to 16.[4] We close this gap in the literature for small to medium-sized rings using a simple yet powerful idea (Section 5.1). When combined with our signature scheme described above, this yields competitive post-quantum ring signatures for rings of size < 32.

At a high level, our strategy is to interpolate a polynomial $Y(X)$ such that $Y(i) = \mathsf{pk}_i$ for $i \in [\ell]$ and have the prover show that the public key they know $(i, \mathsf{pk}_i, Q(X))$ satisfies $Y(X) - \mathsf{pk}_i = Q(X)(X - i)$ for some degree $\ell - 2$ polynomial $Q(X)$. The verifier then checks that this equation holds at a random point r in the field. With overwhelming probability, we are then guaranteed that the claimed polynomial and public key indeed satisfy the relation above. However, this is not sufficient by itself, as the prover has to now show that $i \in [\ell]$. We obtain this for *free* when the public keys can be interpreted as field elements in $\mathrm{GF}(p^t)$ for some prime p. This is indeed the case for the AM-OWF and other MPCitH signatures based on AES, LowMC, Rain$_4$, AIM [AES01, ARS+15, DKR+22, KHS+23]. Instead of secret sharing i over $\mathrm{GF}(p^t)$, the prover shares it over $\mathrm{GF}(\ell)$, and if ℓ is a power of p, the parties in the MPC, can locally embed their shares in $\mathrm{GF}(p^t)$ by appending 0s. Note that this immediately guarantees that i can be expressed using $\log_p \ell$ *digits* and this also does not leak any information about i as the verifier already knows that any honest prover uses an $i \in \mathrm{GF}(\ell)$.

2 Preliminaries

We use $[n]$ to denote the set $\{1, 2, \ldots, n\}$. The computational and statistical security parameter is denoted by $\lambda, \sigma \in \mathbb{N}$. A probability is *noticeable* if it is not negligible, and *overwhelming* if it is equal to $1 - \mathsf{negl}(\lambda)$ for some negligible

[3] https://microsoft.github.io/Picnic/
[4] https://github.com/monero-project/research-lab/issues/79.

function $\mathsf{negl}(\lambda)$. For a set $\mathcal{S}$, we write $s \leftarrow \mathcal{S}$ to indicate that s is sampled uniformly at random from $\mathcal{S}$. For a random variable $\mathcal{D}$, we write $d \leftarrow \mathcal{D}$ to indicate that d is sampled according to $\mathcal{D}$. We use $\Delta(X, Y)$ to denote the statistical distance between two random variables X and Y and $\Delta_H(x, y)$ to denote the hamming distance between two vectors $x, y \in \{0,1\}^k$. For two ensembles of random variables $\{\mathcal{D}_{0,\lambda}\}_{\lambda\in\mathbb{N}}$, $\{\mathcal{D}_{1,\lambda}\}_{\lambda\in\mathbb{N}}$, we write $\mathcal{D}_0 \approx_c \mathcal{D}_1$ to indicate that for all PPT $\mathcal{A}$, it holds that

$$\left| \Pr_{d\leftarrow\mathcal{D}_{0,\lambda}}[\mathcal{A}(d) = 1] - \Pr_{d\leftarrow\mathcal{D}_{1,\lambda}}[\mathcal{A}(d) = 1] \right| \leq \frac{1}{2} + \mathsf{negl}(\lambda).$$

We use $[\![x]\!]$ to denote an additive sharing of x and overload this with arithmetic operations such as multiplication and addition to show operations applied on individual shares by the parties in the MPC. We will also write $[\![x]\!]_i$ to denote the share of the i-th party. When useful, we will explicitly state the field $\mathbb{F}_p$ that is being secret shared over as $[\![x]\!]^p$. We use $\langle a, b\rangle$ to denote the inner product of two vectors $a, b \in \mathbb{F}_p^n$ and $\odot$ to denote their Hadamard product.

Here we recall the definition of universal hash families and the leftover hash lemma. Let $H_\infty(X)$ denote the min-entropy of a random variable X and U_n denote the uniform distribution over $\{0,1\}^n$.

Definition 2.1 (Strong Extractors)**.** A function $f : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ is said to be a strong (k, ε) extractor, with seed length d, if for all random variables X on $\{0,1\}^n$, independent of U_d, with $H_\infty(X) \geq k$,

$$\Delta(f(X, U_d), U_m) \leq \varepsilon.$$

Definition 2.2 (Universal Hash Families)**.** A family $\mathcal{H}$ of hash functions of size 2^d from $\{0,1\}^n$ to $\{0,1\}^m$ is said to be *universal* if, for every $x, y \in \{0,1\}^n$ with $x \neq y$,

$$\Pr_{h\in\mathcal{H}}[h(x) = h(y)] \leq 2^{-m}.$$

Definition 2.3 (Leftover Hash Lemma [HILL99])**.** Let X be a random variable with $H_\infty(X) \geq k$, and $\mathcal{H} : \{0,1\}^n \to \{0,1\}^m$ be a universal hash family of size 2^d. If $m = k - 2\log(\varepsilon^{-1})$, then $h(x)$ is a strong (k, ε) extractor, with seed length d and output length m.

Definition 2.4 (q-ary entropy)**.** For any integer $q \geq 2$ and real $x \in [0, 1]$, the *q-ary entropy function* is defined as $H_q(x) = x\log_q(q-1) - x\log_q(x) - (1-x)\log_q(1-x)$.

Definition 2.5 (Volume of Hamming Ball [GRS23])**.** Let $q \geq 2$ and $0 \leq r/n \leq 1 - 1/q$. Then the volume of a Hamming ball of radius r in $\mathbb{F}_q^n$ satisfies $q^{H_q(r/n)n - o(n)} \leq \mathsf{Vol}_q(r, n) \leq q^{H_q(r/n)n}$.

We refer to Honest-Verifier Zero-Knowledge Argument of Knowledge as HVZKAoK and define it below. Given a two-party interactive protocol between PPT algorithms A with input a and B with input b where only B gets an output, we introduce two random variables: $\langle A(a), B(b)\rangle$ denotes the output of the protocol, and $\mathsf{View}(A(a), B(b))$ denotes the transcript of the protocol.

Definition 2.6. An HVZKAoK with soundness error ε for an NP language $\mathcal{L} \subset \{0,1\}^*$ and corresponding relation $\mathcal{R}_\mathcal{L} \subset \{0,1\}^* \times \{0,1\}^*$ is an interactive protocol between a prover P and verifier V that satisfies the following properties:

- **Perfect Completeness.** For every $(x, w) \in \mathcal{R}_\mathcal{L}$, the verifier always accepts the interaction with an honest prover $\Pr[\langle P(x, w), V(x)\rangle = 1] = 1$.
- **ε-Soundness.** For every PPT algorithm $\tilde{\mathsf{P}}$ that satisfies $\Pr[\langle \tilde{P}(x), V(x)\rangle = 1] = \tilde{\varepsilon} > \varepsilon$, there exists an extractor algorithm Ext which, given rewindable black-box access to $\tilde{P}$, outputs a valid witness w' for x in time $\mathsf{poly}(\lambda, 1/(\tilde{\varepsilon} - \varepsilon))$.
- **Honest-Verifier Zero-Knowledge.** An argument of knowledge is (computationally, statistically, perfectly) HVZK if there exists a PPT simulator Sim such that for every $(x, w) \in \mathcal{R}_\mathcal{L}$, $\mathsf{Sim}(x) \equiv \mathsf{View}(\mathsf{P}(x, w), \mathsf{V}(x))$, where $\equiv$ denotes computational, statistical, or perfect indistinguishability between the distributions.

Gap-HVZK. A gap honest-verifier zero-knowledge argument of knowledge [CKY09, CCJ23] with gap $\mathcal{L}'$, where $\mathcal{L}' \supseteq \mathcal{L}$ is an NP language with relation $\mathcal{R}_{\mathcal{L}'}$, is defined as an honest-verifier zero-knowledge argument of knowledge, with the following relaxation of ε-soundness: the extractor Ext is only guaranteed to output a witness w' such that $(x, w') \in \mathcal{L}'$.

Weak Pseudorandom Function (wPRF). A function family $F : \mathcal{K} \times \mathcal{X} \to \mathcal{Y}$ with key space $\mathcal{K}$, input space $\mathcal{X}$, and output space $\mathcal{Y}$ (implicitly parameterized by security parameter λ) is said to be a weak pseudorandom function if for any $q = \mathsf{poly}(\lambda)$ it holds that

$$\{(x_i, F(k, x_i)\}_{i\in[q]} \approx_c \{(x_i, y_i)\}_{i\in[q]}$$

where $k \leftarrow \mathcal{K}, x_i \leftarrow \mathcal{X}$, and $y_i \leftarrow \mathcal{Y}$.

Pseudorandom Function (PRF). A function family $F : \mathcal{K} \times \mathcal{X} \to \mathcal{Y}$ with key space $\mathcal{K}$, input space $\mathcal{X}$, and output space $\mathcal{Y}$ (implicitly parameterized by security parameter λ) is said to be a (strong) pseudorandom function if for any PPT adversary $\mathcal{A}$ we have

$$\left| \Pr_{k\leftarrow\mathcal{K}}[\mathcal{A}^{F(k,\cdot)} = 1] - \Pr_{f\leftarrow\mathcal{F}}[\mathcal{A}^{f(\cdot)} = 1] \right| \leq \mathsf{negl}(\lambda),$$

where $\mathcal{A}^{F(k,\cdot)}$ and $\mathcal{A}^{F(k,\cdot)}$ denote $\mathcal{A}$ has oracle access to $F(k, \cdot)$ and $f(\cdot)$ respectively, and $\mathcal{F}$ denotes the set of all functions from $\mathcal{X}$ to $\mathcal{Y}$.

3 Symmetric-Key Primitives from Alternating Moduli

We now move on to the design, analysis, and implementation of our new weak PRF and the associated MPC protocols. To better understand our new construction we first review closely related MPC-friendly (weak) PRFs.

3.1 Prior Constructions and Their Shortcomings

Low-MC PRP. One of the popular constructions is the Low-MC block cipher [ARS+15] which follows a similar structure as standard block ciphers such as AES. Unlike our constructions, Low-MC is a permutation and therefore faces additional constraints that all operations must be invertible given the key. In addition, Low-MC is a PRP which implies that it is a strong PRF and must be secure for adaptively chosen x, not just random. For an input $x \in \mathbb{F}^n$, Low-MC is computed iteratively using the round function $s_{i+1} := f_i(k_i, s_i)$ for $i \in [r]$ where $s_1 = x$ and the final output is computed as $F(k, x) = s_{r+1}$. Typically, $r \in [14, 2\lambda]$. The round function $f_i(k_i, s_i)$ can be computed in three phases. First, a non-linear and invertible $\mathbb{F}_2$ transformation is applied to the state. In particular, the current state $s_i \in \mathbb{F}_2^n$ is reinterpreted as $g_i \in \mathbb{G}^m$ where $\mathbb{G} = \mathbb{F}_2^3$ and $m = n/3$. Let $\mathbb{P}$ be the permutation group over $\mathbb{G} \to \mathbb{G}$. A constant element $c \in \mathbb{P}$ is applied to each $\mathbb{G}$ element in g_i, i.e., $g'_i := c \cdot g_i$, and g'_i is then reinterpreted as an $\mathbb{F}_2^n$ vector s'_i. Next, a linear and invertible $\mathbb{F}_2$ transformation is applied, $s''_i := \mathbf{A}_i \cdot s'_i$ for a public $\mathbf{A}_i \in \mathbb{F}_2^{n\times n}$. Finally, the round key $k_i \in \mathbb{F}_2^n$ is added to the state which defines the output of the round, i.e., $s_{i+1} := s''_i + k_i$. When implementing Low-MC, computing the action of $c \in \mathbb{P}$ is implemented in MPC as a binary circuit over $\mathbb{F}_2^3$. The authors give a specific value of c which can be implemented as $(\alpha, \beta, \gamma) \to (\alpha \oplus \beta\gamma, \alpha \oplus \beta \oplus \alpha\gamma, \alpha \oplus \beta \oplus \gamma \oplus \alpha\beta)$ using three parallel AND gates and six "free" XOR gates.

As with the hardness of alternating-moduli paradigm, the security of Low-MC stems in part from two operations that are each non-linear with respect to the other, matrix multiplication $\cdot : \mathbb{F}^{n\times n} \times \mathbb{F}^n \to \mathbb{F}^n$ and the permutation group action $\cdot : \mathbb{P} \times \mathbb{G}^m \to \mathbb{G}^m$. Unlike the alternating-moduli paradigm, only a single element $c \in \mathbb{P}$ is applied to the state as opposed to, for example, a matrix $\mathbf{B} \in \mathbb{P}^{m\times m}$. However, multiplying by such a $\mathbf{B}$ appears to necessitate significant work in the MPC setting since it is non-linear over the secret sharing group $\mathbb{G}$. To mitigate the minimal use of operations in $\mathbb{P}$, Low-MC applies many iterations of the round function. The primary shortcoming of Low-MC is the necessity of $r \approx 14$ invocations of the round function, each of which requires 2 rounds of communication, $O(n)$ OT correlations, and $O(n^2)$ work. Ideally, this could be reduced to effectively performing the round function once.

A weak PRF from [ABG+14]. Akavia *et al.* proposed the first weak PRF that follows a similar structure as ours. Their construction defines the key and input to be $k = (b \in \mathbb{F}^n, \mathbf{A} \in \mathbb{F}_2^{n \times n})$ and $x \in \mathbb{F}_2^n$, respectively. First the matrix-vector product $\mathbf{A}x$ is computed followed by computing a public disjunctive normal form (DNF) formula g on the result, i.e., $F(k, x) = g(\mathbf{A}x \oplus b)$. A DNF formula can be computed as a layer of OR gates followed by AND gates, i.e., for some public (t, e, c) such that $t = \mathsf{poly}(n)$ and $e, c \in \{0,1\}^{t \times n}$, the function can be expressed as $g(x) = \bigwedge_{i=1}^{t} \bigvee_{j=1}^{n} e_{i,j}(x_j \oplus c_{i,j})$. Bogdanov and Rosen [BR17] showed that any DNF formula g can be represented as a rational function of degree at most $O(\log n)$, which in turn implies that f can be distinguished in quasi-polynomial time $O(\mathsf{poly}(tn \cdot 2^{\log(t)\log(n)}))$. While it may be possible to instantiate g to have a large enough t to have exponential security in practice, the concrete efficiency is unlikely to be competitive with alternatives.

A weak PRF from [BIP+18] Boneh *et al.* propose a weak PRF with exponential security that can be computed by depth 2 circuits with mixed moduli. In particular, they consider the function $F(\mathbf{K}, x) = g(\mathbf{K} \cdot_2 x)$ where $g(w) = \sum_i w_i \bmod 3$, and $w := \mathbf{K} \cdot_2 x \in \mathbb{F}_2^m$ is a binary vector that is embedded into $\mathbb{F}_3^m$ component-wise in the natural way. As discussed later, [BIP+18] proposes to restrict $\mathbf{K}$ to be a circulant (or toeplitz) matrix. The works of [BIP+18, DGH+21] show that various types of learning algorithms provably cannot learn this function. They additionally show that it can not be approximated by any low-degree polynomial. This result follows from the result of [Raz87, Smo87] showing that MOD_p can not be approximated by any low depth mod q circuit, where p, q are distinct primes. The work of [DGH+21] proceed to give several extensions to their core construction. The first is support for multiple output bits. The function is defined as $F(\mathbf{K}, x) := \mathbf{B} \cdot_3 (\mathbf{K} \cdot_2 x)$ where $\mathbf{K}$ is a square matrix and $\mathbf{B}$ is a compressing matrix. In particular, one can view $\mathbf{B}$ as a generator matrix for a linear code that has high minimum distance. The best-known attacks for this construction attempt to distinguish by detecting a linear bias in the output. However, these attacks scale exponentially in the minimum distance of $\mathbf{B}$ [BIP+18]. We note that the structure of this weak PRF is identical to the aforementioned OWF. Indeed, [DGH+21] based their OWF on the weak PRF construction of [BIP+18] where the key $\mathbf{K}$ is replaced with a public random matrix. The work of [DGH+21] also conjectures that the $\mathbb{F}_2, \mathbb{F}_3$ can be replaced by any distinct prime fields $\mathbb{F}_p, \mathbb{F}_q$. The core hardness of the problem appears unaffected with all known attacks performing equally poorly on larger moduli. However, $\mathbb{F}_2, \mathbb{F}_3$ (and $\mathbb{F}_3, \mathbb{F}_2$) appear to be the most efficient choice due to yielding more efficient modulus conversion protocols. The work of [BIP+18] suggests choosing the key $\mathbf{K} \in \mathbb{F}_p^{m \times n}$ to be a square cirulant matrix, which in turn allows one to express $\mathbf{K}$ in $O(\lambda)$ space and, as we will see below, enables efficient matrix-vector products in the two-party setting [DGH+21]. With some exception that we will discuss later, choosing $\mathbf{K}$ with this distribution does not appear to degrade security for the parameters used by [BIP+18, DGH+21], i.e., $n = m = 2\lambda, t \approx 0.6\lambda$.

For implementing this weak PRF in MPC, [BIP+18] and [DGH+21] have considered two settings: honest-majority three party and semi-honest two party. First the $\mathbb{F}_2$ secret-shared inputs $[\![\mathbf{K}]\!]^2$ and $[\![x]\!]^2$ are multiplied[5] together to obtain $[\![w]\!]^2 := [\![\mathbf{K}]\!]^2 [\![x]\!]^2$. The exact method used to compute this depends on the setting and is discussed below. Once the sharing of $w = \mathbf{K} \cdot x$ is computed, the parties perform a modulus switching protocol where the shares $[\![w]\!]^2$ are converted into $[\![w]\!]^3$. The work of [DGH+21] suggests that one can preprocess a random double sharing $[\![r]\!]^2, [\![r]\!]^3$ for a uniform $r \in \mathbb{F}_2$ using some protocol and to reveal $w' := [\![r]\!]^2 + [\![w]\!]^2$. Using $[\![r]\!]^3$, it is then possible to subtract off r from w' to obtain $[\![w]\!]^3$. The final step of the protocol is to locally compute $[\![f(\mathbf{K}, x)]\!]^3 := \mathbf{B} \cdot_3 [\![w]\!]^3$. The outline above requires two missing steps, efficiently computing a sharing of $\mathbf{K} \cdot x$ and generating random modulus conversion double sharings $[\![r]\!]^2, [\![r]\!]^3$. Next, we discuss how [DGH+21] suggests this can be done.

In the honest-majority three-party setting, the inner product between two vectors can be computed with $O(1)$ communication [AFL+16], which implies that $[\![w]\!]^2 = [\![x]\!]^2 \cdot [\![\mathbf{K}]\!]^2$ can be computed with linear communication overhead, i.e., $O(n + m)$. Similarly, it is possible to have one of the parties generate $[\![r]\!]^2, [\![r]\!]^3$ locally and then only reveal w' to the other parties, i.e., $O(m)$ communication. In the two-party setting the situation is more complicated due to not having an $O(1)$ communication inner product protocol. However, when evaluating the wPRF for a fixed key $\mathbf{K}$ for many inputs, one can amortize this cost. In particular, for bits $b_1, ..., b_q \in \mathbb{F}_2$ and a fixed vector $\Delta \in \mathbb{F}_2^m$, one can use correlated OT protocols, e.g., IKNP [IKNP03], SoftSpoken OT [Roy22], or silent OT [BCG+19a, RRT23] to generate the secret sharings $[\![b_i \Delta]\!]^2$. Moreover, silent OT can achieve this with an amortized cost of q bits of communication when q is sufficiently large. Alternatively, SoftSpoken OT achieves $q \cdot \frac{\lambda}{c}$ communication for any

[5] When the inputs are plaintext, more efficient multiplication can be used.

constant c, e.g., $c = 4$. In particular, this corresponds to a subfield VOLE protocol with $\mathbb{F}_2$ as the subfield and $\mathbb{F}_{2^m}$ as the extension. When $\mathbf{K}$ is circulant, one can set $\Delta := \mathbf{K}_1 \in \mathbb{F}_{2^m}$ and generate sharings of $[\![x_j^{(i)} \cdot \Delta]\!]$ and locally rotate these to obtain $[\![x_j^{(i)} \cdot \mathbf{K}_j]\!]$ and sum them to compute $[\![\mathbf{K}x^{(i)}]\!]$. We note that [DGH+21] only mentions the technique above in passing and does not implement or report on its performance. Instead, they focus on the setting with (free) preprocessed correlated randomness where $[\![\mathbf{K}x]\!]$ can be computed using other methods (which also leverage the fact that $\mathbf{K}$ is circulant). The work of [DGH+21] does not explicitly state how to generate modulus conversion double sharings and again assumes (free) preprocessing for $[\![r]\!]^2, [\![r]\!]^3$. A natural choice would be using OT or the more recent work of [IKNZ23]. Looking forward, we will offer improvements to these technique in Section 4.

A PRF from [BIP+18]. In addition to the weak PRF construction previously discussed, Boneh *et al.* proposed an extension for upgrading the their conjectured weak PRF to a strong/plain PRF. First they show that one can distinguish their weak PRF in the strong/adaptive setting where the adversary can choose the input x. In particular, their adaptive attacks leverage highly correlated inputs such as having small hamming distance. This suggest the use of an error correcting code to ensure that all inputs have high minimum distance. The most efficient solution would be to encode the input using the same modulus as the input to the weak PRF, i.e., $F(\mathbf{A}, x) = \mathbf{B} \cdot_3 (\mathbf{A} \cdot_2 [\mathbf{G} \cdot_2 x])$ where $\mathbf{G}$ is the generator matrix. This could be implemented in MPC with no communication and very little overhead. Unfortunately this approach does not work as this can be viewed as a transformation on only the key, i.e., $F(\mathbf{A}', x) = \mathbf{B} \cdot_3 (\mathbf{A}' \cdot_2 x)$ where $\mathbf{A}' = \mathbf{A} \cdot_2 \mathbf{G}$. Therefore, the same attack applies to this construction. The work of [BIP+18] also shows that the adaptive attacks on their wPRF can be extended to a relatively large class of multiplicative depth-2 circuits. Given this negative result, [BIP+18] turn their attention to depth-3 circuits. They conjecture that performing the linear code over a different modulus does result in strong security and propose a candidate PRF construction. We refer to [BIP+18] for a detailed discussion.

3.2 Insecure Plus/XOR Construction

As a starting point we first introduce a candidate construction that is ultimately determined to be insecure. The aim of this example is to demonstrate how subtle changes to the construction can result in significant security ramifications. Moreover, we believe it is instructive to know what does not work as well as what does. Given the advancements of $f(x) = \mathbf{B} \cdot_3 (\mathbf{A} \cdot x)$ being an OWF, a natural question to ask is whether it possible to use f to construct another, more efficient, weak PRF. From the efficiency perspective, such an f can be evaluated in the plain two-party setting with linear communication due to the only non-linear step consisting of $O(n)$ modulus conversion gates since $\mathbf{A}$ is now public. Arguably the simplest candidate would be

$$F(k, x) = \mathbf{B} \cdot_3 (\mathbf{A} \cdot_2 (k \oplus x)),$$

where k and x are binary vectors and $\oplus$ denotes addition over $\mathbb{F}_2$. To attack the candidate above, it is not hard to see that it suffices to provide an attack for the following single-bit output variant

$$F(k, x) = 1^t \cdot_3 (\mathbf{A} \cdot_2 (k \oplus x))$$

where 1^t denotes the all-one vector of appropriate dimension. We now show that this function is indeed learnable. It will be useful for us to represent $\mathbb{F}_3$ as the set $\{-1, 0, 1\}$ in the natural way. A simple observation is that one can emulate addition over $\mathbb{F}_2$ using multiplication over $\mathbb{F}_3$ by relying on the mapping $\phi(x) = x + 1$ (arithmetic in $\mathbb{F}_3$), which maps $0 \to 1$ and $1 \to -1$. Let $y = F(k, x)$ be an input-output pair from the wPRF candidate above. It follows by inspection that if we use the notation $\bar{x}_j = \phi(x_j)$ and $\bar{k}_j = \phi(k_j)$ for $j \in [n]$, we can write the following

$$y = \sum_{i=1}^{m} \phi\Big(\prod_{j=1}^{n} (\bar{k}_j \bar{x}_j)^{a_{ij}} \Big) = \sum_{i=1}^{m} \phi\Big(\underbrace{\Big(\prod_{j=1}^{n} \bar{k}_j^{a_{ij}} \Big)}_{S_i} \cdot \underbrace{\Big(\prod_{j=1}^{n} \bar{x}_j^{a_{ij}} \Big)}_{T_i \in \{-1,1\}} \Big)$$

where arithmetic operations are done over $\mathbb{F}_3$. Since input is known in the wPRF game, by plugging in the input values we can compute each T_i. Moreover, each S_i is a monomial over $\bar{k} = (\bar{k}_1, \ldots, \bar{k}_n)$, i.e., S_i is simply a subset product

of components of $\overline{k}$ where the subset only depends on $\mathbf{A}$ (and is independent of x). We can now recover the key in two steps. First, given that ϕ is a linear function, after gathering enough samples we can use Gaussian elimination to recover each $S_i \in \{-1, 1\}$ for $i \in [m]$. In the next step, we use S_i to recover the key. To do so, recall that each S_i is a monomial of components of $\overline{k}$. Because multiplication over $\mathbb{F}_3$ (for the set $\{-1, 1\}$) is isomorphic to addition over $\mathbb{F}_2$, it follows that

$$\mathbf{A} \cdot_2 k = \left(\phi^{-1}(S_1), \ldots, \phi^{-1}(S_m)\right) \in \{0, 1\}^m.$$

Since $\mathbf{A}$ is a random expanding matrix we can recover k by Gaussian elimination, as desired. We remark that one can rely on a similar argument to show that the following candidate

$$F(k, x) = \mathbf{B} \cdot_3 (\mathbf{A} \cdot_2 (k \| x))$$

is also insecure, which is obtained by replacing the XOR operation in the plus construction with concatenation (and appropriately modifying the row dimension of the public matrix $\mathbf{A}$).

3.3 Our Constructions

An alternative view of [BIP+18]. Leading up to our main construction (Definition 3.2), let us first reconsider the weak PRF of Boneh *et al.* [BIP+18] defined as $F(\mathbf{K}, x) := \mathbf{B} \cdot_3 (\mathbf{K} \cdot_2 x)$ where $x \in \mathbb{F}_2^n$ is the input, $\mathbf{K} \in \mathbb{F}_2^{m \times n}$ is the key (typically square and circulant), and $\mathbf{B} \in \mathbb{F}_3^{t \times m}$ is a public compressing matrix. Observe that this function can be computed as $F(\mathbf{K}, x) := \mathbf{B} \cdot_3 [\mathbf{A} \cdot_2 (k \odot_2 x')]$ where for $n' := nm$, the vectors $k', x' \in \mathbb{F}_2^{n'}$ are the appropriately unrolled version of $\mathbf{K}, x$. In particular, let $k'_{im+j} := \mathbf{K}_{i,j}$ and $x' := xx...x$ be m copies of x. Computing the component-wise product of these two vectors provides all of the terms required to compute $\mathbf{K} \cdot_2 x$. All that remains is to perform the summations corresponding to the rows of $\mathbf{K}$. This can be achieved using an appropriately defined public matrix $\mathbf{A} \in \mathbb{F}_2^{m \times nm}$. In particular, $\mathbf{A}$ will be the (compressing) repetition code, or equivalently it will look like a staircase with consecutive runs of n ones in each of the m rows. We argue that this formulation more explicitly describes the construction as it separates the additive and multiplicative steps. An additional benefit of the explicit formulation is that the $n' = nm = O(\lambda^2)$ scalar multiplications between secret vectors k', x' are apparent.

Our weak PRF. Instead of asking if we can reduce the overhead of the Boneh *et al.* construction, it will be more instructive to ask if there are any methods for plausibly *improving* the hardness of the function. The most natural option is to remove all structures from $k', x', \mathbf{A}$. If $\mathbf{A}$ is the repetition code of [BIP+18, DGH+21], then w_j is the sum $\sum_{i \in [n]+(j-1)m} x'_i k'_i = \sum_{i \in [n]} x_i \mathbf{K}_{1,i+j}$. As demonstrated in Section 3.2, the alternating-moduli paradigm becomes insecure if each w_j is not the combination of $O(\lambda)$ multiplications and additions for each w_j. As such, choosing a circulant $\mathbf{K}$ with dimension $O(\lambda)$ in some sense appears to be the most succinct option while remaining secure, i.e., each input bit x_i for $i \in [n]$ is multiplied by each (independent) key bit $\mathbf{K}_{1,j}$, leading to $nm = O(\lambda^2)$ terms where the summations are taken over disjoint sets and $x_i, \mathbf{K}_{1,j}$ each appear once.

However, we observe that this is by no means the only option. Let us remove the constraints on $x', k' \in \mathbb{F}_2^{n'}$ such that they are uniformly distributed and replace $\mathbf{A} \in \mathbb{F}^{m \times n'}$ with a code with high minimum distance, e.g., $\mathbf{A}$ is uniform. Observe that each w_j is now a linear combination of the $x'_i k'_i$ terms. In expectation, each w_j will be the summation of $n' = nm = O(\lambda^2)$ terms. In light of the known cryptanalysis, this construction appears overly conservative in that the w_j terms only need to be a sum of $O(\lambda)$ terms while at the same time not having small linear dependency. We have now defined our new $(\mathbb{F}_2, \mathbb{F}_3)$-wPRF which is a generalization of the Boneh *et al.* wPRF. In particular, we can write the implicit construction above more formally as follows:

Definition 3.1. Let $n, m, t \in \mathbb{N}$, our $(\mathbb{F}_2, \mathbb{F}_3)$-wPRF construction is $F(k, x) := \mathbf{B} \cdot_3 (\mathbf{A} \cdot_2 [k \odot_2 x])$ where $x, k \in \mathbb{F}_2^n$ and $\mathbf{A} \in \mathbb{F}_2^{m \times n}, \mathbf{B} \in \mathbb{F}_3^{t \times m}$ are uniformly distributed.

The weak PRF construction of Boneh *et al.* is then defined by requiring that $x, k, \mathbf{A}$ have repetitive structure along with defining $n = O(\lambda^2)$. However, we will argue that the problem for uniform $x, k, \mathbf{A}$ remains hard even when $n = O(\lambda)$. One interpretation of our result is that prior works perform $O(\lambda^2)$ multiplications followed by multiplication with a repetition code while our construction more efficiently amortizes the multiplications by replacing the repetition code with a high minimum distance code, i.e., our construction more diligently uses the limited number

of multiplications that are available. As with Boneh *et al.* [BIP+18], one can similarly generalize our construction to any prime field.

Definition 3.2 (Generalized weak PRF)**.** Let $n, m, t \in \mathbb{N}$, and let p, q be distinct primes. Our $(\mathbb{F}_p, \mathbb{F}_q)$-wPRF is defined as $F(k, x) := \mathbf{B} \cdot_q (\mathbf{A} \cdot_p [k \odot_p x])$ where $x, k \in \mathbb{F}_p^n$ and $\mathbf{A} \in \mathbb{F}_p^{m \times n}, \mathbf{B} \in \mathbb{F}_q^{t \times m}$ are uniformly distributed.

Following the same analysis as [BIP+18], we conjecture that our depth-2 weak PRF can be compiled into a (strong) PRF by first encoding the input using an error correcting code with a different modulus than the key. In particular, we conjecture that Definition 3.3 is a PRF for appropriately chosen n, m, t, d.

Definition 3.3 (Generalized PRF)**.** Let $n, m, t, d \in \mathbb{N}$, and let p, q be distinct primes. Our $(\mathbb{F}_q, \mathbb{F}_p, \mathbb{F}_q)$-PRF is defined for $x \in \mathbb{F}_q^d$ as $F(k, x) := \mathbf{B} \cdot_q \left(\mathbf{A} \cdot_p [k \odot_p (\mathbf{G} \cdot_q (x, 1))]\right)$ where $k \in \mathbb{F}_p^n$ and $\mathbf{G} \in \mathbb{F}_q^{n \times (d+1)}, \mathbf{A} \in \mathbb{F}_p^{m \times n}, \mathbf{B} \in \mathbb{F}_q^{t \times m}$ are uniformly distributed.

The core intuition behind this construction is that known attacks against our weak PRF and the weak PRFs of [BIP+18] in the adaptive setting heavily rely on PRF evaluations for highly correlated inputs, e.g., having small hamming distance. The work of [BIP+18] proposes a compiler that they define as encoded input. The core idea is to restrict the adversary's choice of inputs to the underlying weak PRF so that highly correlated inputs are not allowed. A natural example of such an encoding is encoding the input using a linear error correcting code $\mathbf{G}$. The most efficient option for doing this is to perform the encoding in $\mathbb{F}_p$. However, the work of [BIP+18] shows that this does not work as one can recast this encoded input function as a new instance of the underlying weak PRF with a different $\mathbf{A}$ matrix. This recasting crucially relies on the fact that $\mathbf{A}$ and $\mathbf{G}$ are over the same modulus and therefore compose into a new matrix $\mathbf{A}'$. Given this observation, they suggest sampling $\mathbf{G}$ over a different modulus and performing an additional round of modulus conversion of $\mathbf{G} \cdot_q (x, 1)$ from $\mathbb{F}_q$ to $\mathbb{F}_p$ before multiplying it with the key. To ensure that the output remains pseudorandom for all-zero input, 1 is appended to x.

3.4 Optimizations

We consider two optimizations with the aim of improved efficiency of our wPRF when evaluated in the two-party setting where the key k is fixed for many inputs $x^{(1)}, ..., x^{(q)}$. Due to the generality of our construction, both can be framed as changing the distribution of the key k or input x. We provide a description of our protocols in Section 4.

Structured input x. For now, let us focus on the $(\mathbb{F}_2, \mathbb{F}_3)$-wPRF where $x^{(i)}, k \in \mathbb{F}_2^n$. We consider the general methods for computing $x^{(i)} \odot k$, i.e., component-wise multiplication.

Reusable key OTs. Given that k is fixed, a natural protocol for computing $[\![k \cdot x^{(i)}]\!]$ is to preprocess a random OT for each bit $k_1, .., k_n$ of the key. These can then later be derandomized using standard techniques to generate the sharing $[\![k \cdot x^{(i)}]\!]$ using only $O(n)$ communication and minimal computation. The advantage of this approach is that no per evaluation OTs are required. When combined with our OT-based modulus switching, we will show that we need only $m = 2\lambda$ OTs per evaluation. Alternatively, when combined with our custom garbled circuit modulus-switching protocols, no per evaluation OTs are needed at all. However, we will see that the disadvantage of this approach is that it requires $n = 4\lambda$ bits of communication, and as a result, this approach will have more communication (but fewer OTs) compared to the protocol of [DGH+21] with $n = 2\lambda$ bits of communication.

Nonreusable input OTs. To bring down the overall communication, we make the observation that the input x can have a smaller "effective" size of just λ bits. First we change the distribution of x such that for some $\hat{x} \in \mathbb{F}^\lambda$, one can express x as s copies of $\hat{x}$, i.e., $x = (\hat{x}||\hat{x}||...||\hat{x})$. We will then be able to efficiently multiply each bit of $\hat{x}_i$ with corresponding s bits of the key, i.e., $k_i, k_{i+n/s}, ...$, using subfield VOLE. In particular, when using silent VOLE in the amortized setting, the communication complexity is essentially independent of s. As a result, the total amortized communication for computing $x \odot k$ with $s = 4$ is just $\lambda = n/s$ bits, a 4 to 2 times reduction depending on the protocol. However, this change implies that $n/s = \lambda$ additional VOLE correlations will be consumed per evaluation. We note that for small s, these silent VOLE correlations can be packed together such that computing λ/s of them

can be computed at the effective computational cost of a single random OT. Finally, as we see later, we note that for [BIP+18, DGH+21], applying similar optimizations results in their schemes being insecure due to certain weaknesses associated with the use of the circulant key $\mathbf{K}$.

Structured key k over $\mathbb{F}_3$. In the case of $(\mathbb{F}_3, \mathbb{F}_2)$-wPRF it will be advantageous for us to restrict the key k which would typically be in $\mathbb{F}_3^n$ to lay in the *binary* subset. The operations of multiplying $\mathbf{A} \cdot_3 (k \odot_3 x)$ will still be performed modulo three. We conjecture that this variant is as secure given that n is increased to compensate for the loss of entropy in the key. When combined with the reusable key OTs optimization above, this enable $x \odot k$ to be computed with half the communication complexity of the original method.

3.5 Security Analysis and Parameter Selection

Here we show that the plausible security of our weak PRF is connected to the hardness of solving sparse multivariate polynomials over $\mathbb{F}_3$. Building on prior works, we proceed to using combinatorial methods to analyze our construction, with a focus on a reduction to subset sum problem.

Polynomial representation. The security of our construction is closely related to the hardness of solving sparse multivariate polynomials. Indeed, in Section 3.2 we presented a weakened variant of our construction that can be broken when viewed as a problem over multivariate polynomials. Recall that for $t = 1$ our construction can be written as $F(k, x) = b^{\mathsf{T}} \cdot_3 (\mathbf{A} \cdot_2 [k \odot_2 x])$. Observe that one can emulate addition over $\mathbb{F}_2$ using multiplication over $\mathbb{F}_3$ by relying on the mapping $\phi(x) = x + 1$ (arithmetic in $\mathbb{F}_3$), which maps $0 \to 1$ and $1 \to -1$ (we also use the notation $\bar{x}_j = \phi(x_j)$ and $\bar{k}_j = \phi(k_j)$ for $j \in [n]$). Then, $F(k, x)$ can be rewritten as

$$\sum_{i=1}^{m} b_i \cdot \phi^{-1}\Big(\prod_{j=1}^{n} \bar{k}_j^{a_{ij} x_j}\Big) = \sum_{i=1}^{m} b_i \cdot \Big(\prod_{j=1}^{n} \bar{k}_j^{a_{ij} x_j} - 1\Big) = \sum_{i=1}^{m} \prod_{j=1}^{n} \bar{k}_j^{a_{ij} x_j} - \underbrace{\sum_{i=1}^{m} b_i}_{P}.$$

The term P is public and hence can be computed by the adversary. Thus, the hardness of our weak PRF construction boils down to the pseudorandomness of the following: $F'(k, x) = \sum_{i=1}^{m} \prod_{j=1}^{n} \bar{k}_j^{a_{ij} x_j}$, where operations are done over $\mathbb{F}_3$. For any input x, the weak PRF output is simply a sparse multilinear polynomial over $\mathbb{F}_3$ (where the polynomial is defined by the input and public parameters), so a key recovery attack for the construction directly corresponds to solving a system of sparse multilinear equations over $\mathbb{F}_3$.

A generalized construction. Building upon the idea above, we now describe a simple framework to instantiate new weak PRFs based on the hardness of solving a system of sparse multilinear equations over a finite field of (small) size. First, we fix a finite field $\mathbb{F}_p$ and two dimensions m and n. Let $\mathbb{F}_p^{*n}$ be the key space and let $\mathcal{D}$ be a distribution over $m \times n$ binary matrices. The construction can be succinctly described as follows: $F(k, \mathbf{X}) = \sum_{i=1}^{m} \prod_{j=1}^{n} k_j^{x_{ij}}$, where all operations are done over $\mathbb{F}_p$. Note that m is the number of terms in the polynomial and the ith term in the polynomial simply corresponds to a subset product over the components of the key k according to the ith row of the input $\mathbf{X}$. For instance, the preceding construction (with public matrix $\mathbf{A}$) is an instantiation of the framework above with the field $\mathbb{F}_3$, and the distribution $\mathcal{D}$ is obtained by sampling $x \leftarrow \{0, 1\}^n$ and setting the ith row of $\mathbf{X}$ as $\mathbf{A}_i \odot x$.

On the duality of polynomial representation. We remark that in the polynomial representation outlined above, one can alternatively put $\bar{x}_j = \phi(x_j)$ in the base and obtain the following representation of F' (a similar representation can be found for F as well): $F'(k, x) = \sum_{i=1}^{m} \prod_{j=1}^{n} \bar{x}_j^{a_{ij} k_j}$. Note that in this representation, a key recovery attack would correspond to *interpolating* sparse multilinear polynomials. While the connection between symmetric-key primitives (based on the alternating-moduli paradigm) and the hardness of interpolating sparse multilinear polynomials has already been observed by [BIP+18], neither of [BIP+18] or [DGH+21] considered the dual problem of solving a system of sparse multilinear polynomial equations for their constructions.

We present several possible parameterizations of our wPRF. The most efficient known attacks [DGH+21] are based on a reduction to subset sum problem [HJ10, BCJ11, BBSS20]. We will review the core reduction which focuses on the AM-OWF. We then discuss how this applies to the wPRF where the key is effectively the AM-OWF input.

Subset sum attack. At a high level, the main idea to attack the AM-OWF can be described in two steps. In the first step, inverting the OWF is reduced to a specialized subset-sum problem. In the second step, modern subset-sum solving algorithms such as [HJ10, BCJ11, BBSS20] are modified in a way that they enable us to solve the resulting specialized subset-sum problem and invert the OWF. We focus on the first step, as the modification step can be done via standard algebraic and combinatorial techniques.

Let $y \in \mathbb{F}_3^t$ be an output of the OWF on an input $x \in \mathbb{F}_2^n$, and let $w = \mathbf{A} \cdot_2 x$ be the intermediary evaluation. Observe that there is an $(m-n) \times m$ parity check matrix $\mathbf{P}$ such that $w = \mathbf{A} \cdot_2 x$ iff w lies in the kernel of $\mathbf{P}$. So, for any intermediary evaluation w we have $\mathbf{P} \cdot_2 w = \mathbf{0}$ and $\mathbf{B} \cdot_3 w = y$. We now aim to find w by a reduction to subset sum. Denoting the i-th unit vector by e_i, note that if we can find a set of indices $I \subseteq [m]$ such that $\left(\sum_{i \in I} \mathbf{P} \cdot_2 e_i, \sum_{i \in I} \mathbf{B} \cdot_3 e_i\right) = (\mathbf{0}, y)$, then we can invert the OWF simply be solving $\mathbf{A} \cdot_2 x = \sum_{i \in I} e_i$. Therefore, we have a reduction to the subset sum problem with the target $\left(\mathbf{0}, y\right) \in \mathbb{F}_2^{m-n} \times \mathbb{F}_3^t$ and m variables $\beta_1, \ldots, \beta_m$, where we associate $\beta_i = 1$ with $\left(\mathbf{P} \cdot_2 e_i, \mathbf{B} \cdot_3 e_i\right)$. We point out that this problem can also be seen as a special case of generalized knapsack problem for the additive group $(\mathbb{F}_2^{m-n} \times \mathbb{F}_3^t, +)$, for which search-to-decision reductions with different parameter setting have been proposed by [MM11].

As mentioned in [DGH+21], a slightly modified version of [HJ10] needs $2^{0.337m}$ time and $2^{0.256m}$ space (ignoring polynomial factors in m) to invert the OWF. In addition, the algorithm of [BBSS20] (with slight modification) runs in $2^{0.283m}$ time and space. Overall, to achieve s-bit security one needs to set $m \geq 3.53s$. Thus, a suggested choice of the parameters (n, m, t) in terms of λ for the AM-OWF would be $(\lambda, 3.53\lambda, \lambda/\log_2 3)$.[6]

One-to-one parameters. We now present a conservative parameter set for our wPRF constructions given the subset sum attack. One can rely on the OWF attack described above to break our $(\mathbb{F}_2, \mathbb{F}_3)$-wPRF. Consider fixing some input-output pair $(x, y = F(k, x))$ and observe that this can be viewed as an instance of the OWF applied to the key k. For each $x_i = 0$, it is clear that k_i does not have any impact on the output of the wPRF. Thus, on average, one needs to double the key/input size to prevent the subset sum attack from recovering k given a single (x, y) sample. We propose the following parameter setting for our $(\mathbb{F}_2, \mathbb{F}_3)$-wPRF to get λ bits of security: $n = 2\lambda, m = 7.06\lambda, t = 2\lambda/\log_2 3$, which is simply doubling the parameters in OWF setting to thwart the subset sum attacks. In particular, both key and input will be 2λ-bit strings. For the $(\mathbb{F}_3, \mathbb{F}_2)$-wPRF we propose $n = 2\lambda/\log_2(3), m = 7.06\lambda/\log_2(3), t = 2\lambda$. Indeed, in the restricted single wPRF sample setting, it is not hard to show that the hardness of the OWF implies the hardness of a key recovery attack. Recall that the parameters of the OWF imply that it is approximately one-to-one. As such, for any given x we should expect there to be one value for $x \odot k$ that is consistent with any given sample. Hence, we denote the parameters above as one-to-one. When generalized beyond the artificial constraint of a single sample, one can view each sample $(x_1, y_1), ..., (x_q, y_q)$ as defining a related OWF instance, where the OWF input k and parameter $\mathbf{A}$ are "subsetted" by x_i.

Many-to-one parameters. Unlike the OWF, recall that given a sample (x, y), non-invertability is not a requirement of a wPRF with respect to arbitrary key k'. Building on this observation we propose to deviate from the OWF parameter regime and consider the setting where there are many consistent $x \odot k$ for any given (x, y). Specifically, we define $n = 4\lambda, m = 2\lambda, t = \lambda/\log_2(3)$ for the $(\mathbb{F}_2, \mathbb{F}_3)$-wPRF and $n = 4\lambda/\log_2(3), m = 2\lambda, t = \lambda$ for the $(\mathbb{F}_3, \mathbb{F}_2)$-wPRF. Focusing on the $(\mathbb{F}_2, \mathbb{F}_3)$-wPRF, observe that for any given sample (x, y), there is a set $Z_{y,x} := \{z \mid y = \mathbf{B} \cdot (\mathbf{A} \cdot z) \wedge z \odot x = z\}$ representing the valid $k \odot x$ preimages. For the parameter regime above, it holds that $|Z| = O(2^\lambda)$. Indeed, consider the $\mathbf{A}', k'$ where $\mathbf{A}'$ consists of the columns of $\mathbf{A}$ indexed by i such that $x_i = 1$ and k' consists of k_i. For the average case of $|x| = 2\lambda$, we have $\mathbf{A}' \in \mathbb{F}_2^{m \times m}$ and therefore the size of Z is precisely the size of $W = \{w \mid y = \mathbf{B} \cdot_3 w \wedge w \in \mathbb{F}_2^m\}$. That is, W is defined by the binary codewords of a random $t \times m$ linear codes (viewing $\mathbf{B}$ as the parity check matrix) and therefore we would expect $|W| \approx (2/3)^m 3^{m-t} > 2^{m-1.6t} > 2^\lambda$. Therefore the adversary has no advantage in recovering k given a single sample.

[6] More aggressive parameter have also been proposed, e.g., $(\lambda, 3.13\lambda, \lambda/\log_2 3)$ [DGH+21].

Consider q samples $(x^{(1)}, y^{(1)}), ..., (x^{(q)}, y^{(q)})$. Let $w^{(i)} := \mathbf{A} \cdot (k \odot x^{(i)})$ and observe that given $\mathbf{A}, \mathbf{B}, x^{(i)}$, one can compute $\mathbf{A}' \in \mathbb{F}_2^{qm \times n}, \mathbf{B}' \in \mathbb{F}_3^{qt \times qm}$ such that $(w^{(1)}, ..., w^{(q)}) = \mathbf{A}' \cdot k$ and $(y^{(1)}, ..., y^{(q)}) = \mathbf{B}' \cdot (w^{(1)}, ..., w^{(q)})$.

Therefore, assuming q is sufficiently large, one can define $\mathbf{P}$ as the parity check matrix of $\mathbf{A}'$ and solve the following subset sum problem $(\sum_{i \in I} \mathbf{P} e_i, \sum_{i \in I} \mathbf{B}' e_i) = (0, y)$ to recover $w = \sum_{i \in I} e_i$ and therefore k. However, the running time for this problem with the best-known subset sum solver [BBSS20] is $O(2^{0.337qm})$. Based on the discussion above, q must be at least 2 and therefore the running time is at least $O(2^{0.337 \cdot 2 \cdot 2\lambda}) = O(2^{1.35\lambda})$.

Parameters for optimizations. In Section 3.4 we propose two alterations to the construction. The first proposes to restrict x to be s copies of $\hat{x} \in \mathbb{F}^\lambda$, i.e., $n = s\lambda$. The main attack that this alteration impacts is the potential improved efficiency for sparse polynomial. As with the prior works, our scheme can be framed as the problem of interpolating sparse polynomials, i.e., $F'(k, x) = \sum_{i=1}^{m} \prod_{j=1}^{n} \bar{x}_j^{A_{ij} k_j}$. The efficiency of these solvers require evaluating the polynomial at specific values, e.g., roots of unity. Therefore, one must ensure that sufficiently few of such points coincide with the random weak PRF inputs, $x^{(1)}, ..., x^{(q)}$. Given that at most $q = 2^{40}$ queries are made, we conjecture that such techniques remain exponential time. Indeed, [BIP+18] shows that if such attacks are effective, then learning with rounding (LWR) [BPR12] for similar parameters is broken. Should a large bound on q be desired, one can increase n or decrease s accordingly to maintain the security margin between $2^{n/s}$ and q.

We note that similar optimizations should not be applied to the protocol of [DGH+21] with a circulant key. In particular, when the wPRF is defined as $F(\mathbf{K}, x) = \mathbf{B} \cdot_3 (\mathbf{K} \cdot_2 x)$ and $\mathbf{K}$ is circulant, the scheme suffers from a vulnerability when x is symmetric, i.e., $x = (\hat{x} || \hat{x})$. Due to the symmetry of both x and $\mathbf{K}$, the intermediate value $w = \mathbf{K} \cdot x$ will also be symmetric, i.e., $w = (\hat{w} || \hat{w})$. Given that $\mathbf{B}$ is close to a rate 0.5 matrix, i.e., $m/2 \approx t$, the attacker can efficiently solve for $\hat{w}$ and thereby recover w, rendering the scheme insecure. To prevent this, the work of [DGH+21] requires x to be uniform over $n = 2\lambda$, which results in symmetric x occurring with negligible probability, unlike the case of $n = \lambda$. The second security-relevant optimization we suggested is the restriction of the key $k \in \mathbb{F}_3^n$ for the $(\mathbb{F}_3, \mathbb{F}_2)$-wPRF to lie in the binary subset. As we discussed before, this halves the communication complexity of computing $x \odot k$ when implemented using reuseable key OTs technique. We are not aware of any attacks that can take advantage of this distribution change beyond relatively trivial attacks. To mitigate the impact of these, we suggest increasing the key length by $\log_2(3)$, i.e., $n = 4\lambda$.

Other attacks. Another potential avenue of attack would be utilizing Gröbner basis to solve a system of multilinear polynomial equations. However, in its plain format, the algorithmic cost of such an attack is quite high and it does not seem to impact the security of our wPRF constructions. Both [BIP+18] and [DGH+21] argued that appropriately designed alternating-moduli constructions cannot be approximated by low-degree polynomials. Moreover, they give conjectures that this extends to rational functions as well. Assuming that the equivalent conjecture for our construction holds, it appears implausible that generic algebraic techniques will be effective for our parameters. On the flip side, we are not aware of any other algebraic attack that particularly exploits the extra structure/information provided by the wPRF and hence we suggest setting parameters based on subset sum. Finally, we refer to the analysis in [BIP+18, DGH+21] for additional attacks that leverage some common structure to all of our schemes. For example, one can break the security if $\mathbf{B}$ has small minimum distance, if one is able to efficiently enumerate all w for a fixed y, if one is able to detect bias in the output bits, or if one can leverage the parity of x, y, along with attacks based on LPN, and connections to learning theory [BIP+18, DGH+21]. However, these attacks are not significantly impacted by our changes and are less efficient than the subset sum attack.

3.6 Properties of the AM-OWF

We now focus on proving some useful properties for the AM-OWF of [DGH+21], which will be helpful for our constructions in Section 5. Recall that the AM-OWF is defined as $f(x) = \mathbf{B}(\mathbf{A}x)$ where $\mathbf{A} \in \mathbb{F}_2^{m \times n}$ and $\mathbf{B} \in \mathbb{F}_3^{t \times m}$ are fixed public matrices chosen uniformly at random, and $\mathbf{A}x$ is interpreted as a binary vector in $\mathbb{F}_3^m$. Roughly speaking, our core lemma shows that when the input size is made slightly larger than the output, the statistical distance between the distribution of the output of the AM-OWF on a uniformly random input and the uniform distribution is negligible. At the same time, we show that a uniformly random value from $\mathbb{F}_3^t$ lies in the image of the OWF with overwhelming probability. We will then use this lemma to prove the hardness of various problems closely related to

the AM-OWF that will allow us to reduce the size of our post-quantum signature scheme. We begin by proving that the family of AM-OWFs is universal.

Lemma 3.4. *Let* $\mathcal{H} = \{h_{\mathbf{A},\mathbf{B}} : \{0,1\}^n \to \{0,1\}^{t \log 3} \mid \mathbf{A} \in \mathbb{F}_2^{m \times n}, \mathbf{B} \in \mathbb{F}_3^{t \times m}; \mathrm{rank}(\mathbf{A}) = n\}$ *be a family of hash functions such that* $h_{\mathbf{A},\mathbf{B}}(x) = \mathbf{B} \cdot_3 (\mathbf{A} \cdot_2 x)$*, then* $\mathcal{H}$ *is a universal hash family.*

Proof. Observe that $\mathrm{rank}(\mathbf{A}) = n$ and hence $\mathbf{A}$ is an injective mapping. Define the group G as $G = (\mathbb{F}_3^t, +)$ and observe that $h_{\mathbf{A},\mathbf{B}}(x)$ is basically a subset sum over m group elements $g_1, \ldots, g_m$ where g_i denotes the ith column of $\mathbf{B}$. Let $w = \mathbf{A}x$ and $w' = \mathbf{A}y$. Since $w \neq w'$ (because $x \neq y$ and $\mathbf{A}$ is an injective mapping), there is at least one index i such that $w_i \neq w'_i$. Without loss of generality, let $i = 1$ and $w_i = 0$. Observe that

$$\Pr[h_{\mathbf{A},\mathbf{B}}(x) = h_{\mathbf{A},\mathbf{B}}(y)] = \Pr\left[\sum_{i=1}^{t} w_i \cdot g_i = \sum_{i=1}^{t} w'_i \cdot g_i\right] = \Pr\left[g_1 = \sum_{i=2}^{t} w_i \cdot g_i - \sum_{i=2}^{t} w'_i \cdot g_i\right] = |G|^{-1} = 3^{-t},$$

where the third equality follows from the fact that g_1 is distributed uniformly and independently from others, and the proof is complete. □

By appealing to the leftover hash lemma, we have the following corollary. Intuitively, Corollary 3.5 shows that if we *swap* the output of the AM-OWF with a uniformly random string of the same size, even a computationally unbounded adversary fails to distinguish them with some noticeable probability.

Corollary 3.5. *Let* $h_{\mathbf{A},\mathbf{B}} \leftarrow \mathcal{H}$*, as defined in Lemma 3.4 and* $n \geq t \log 3 + \omega(\log \lambda)$*, then* $h_{\mathbf{A},\mathbf{B}}$ *is a strong* $(n, \mathsf{negl}(\lambda))$ *extractor with output length* $t \log 3$*.*

In addition to being *close* to the uniform distribution, we will now show that the AM-OWF *covers* an overwhelming fraction of the output domain $\mathbb{F}_3^t$.

Lemma 3.6. *For any* $h_{\mathbf{A},\mathbf{B}} \leftarrow \mathcal{H}$*, let* $\mathcal{O}_{h_{\mathbf{A},\mathbf{B}}} = \{h_{\mathbf{A},\mathbf{B}}(x) | x \in \mathbb{F}_2^n\}$ *denote the covering of* $h_{\mathbf{A},\mathbf{B}}$*. Then, with overwhelming probability over the choice of* $h_{\mathbf{A},\mathbf{B}}$*, we have* $|\mathcal{O}_{h_{\mathbf{A},\mathbf{B}}}|/3^t \geq 1 - \mathsf{negl}(\lambda)$*.*

Proof. Suppose this was not true, then we would have $\Delta(h_{\mathbf{A},\mathbf{B}}(U_k), U_{t \log 3}) \geq 1/\mathsf{poly}(\lambda)$, hence contradicting Lemma 3.4. Thus, the output of the AM-OWF must cover an overwhelming fraction of the output domain. □

3.6.1 Variants of the AM-OWF

We next define three variants of the AM-OWF and show that they are at least as hard as inverting the AM-OWF. Recall that in the OWF game, a challenger samples $\mathbf{A} \leftarrow \mathbb{F}_2^{m \times n}$, $\mathbf{B} \leftarrow \mathbb{F}_3^{t \times m}$, and $x \leftarrow \mathbb{F}_2^n$ and computes an instance $(\mathbf{A}, \mathbf{B}, y = \mathbf{B}(\mathbf{A}x))$. The adversary is tasked with finding *any* x such that $\mathbf{B}(\mathbf{A}x) = y$. The advantage of an adversary in this game is the probability with which it outputs such an x.

Systematic form. To improve the efficiency of our protocols using the AM-OWF, we wish to use the AM-OWF in systematic form, i.e., the adversary is given $(\mathbf{A}, \mathbf{B}, x)$ where all entries are sampled uniformly at random except for the bottom-most $n \times n$ entries of $\mathbf{A}$ and the right-most $t \times t$ entries of $\mathbf{B}$, which are replaced with $\mathbf{I}_n$ and $\mathbf{I}_t$ respectively. We now prove that this new game is just as hard the original OWF game.

Lemma 3.7. *Given an adversary* $\mathcal{A}$ *that wins the* systOWF *game with noticeable advantage, there exists an adversary* $\mathcal{A}'$ *that wins the* OWF *game with noticeable advantage.*

Proof. Given an instance from the OWF game $(\mathbf{A}, \mathbf{B}, y)$, let $\mathbf{M}_1, \mathbf{M}_2$ be defined as the matrices such that $\mathbf{A}\mathbf{M}_1 = \begin{bmatrix} \tilde{\mathbf{A}} \\ \mathbf{I}_n \end{bmatrix}$ and $\mathbf{M}_2\mathbf{B} = [\tilde{\mathbf{B}} | \mathbf{I}_t]$. $\mathcal{A}'$ then provides the adversary $\mathcal{A}$ with $(\mathbf{A}\mathbf{M}_1, \mathbf{M}_2\mathbf{B}, \tilde{y} = \mathbf{M}_2 y)$. $\mathbf{M}_1$ and $\mathbf{M}_2$ are in fact the inverses of the bottom-most square and right-most square of $\mathbf{A}$ and $\mathbf{B}$ respectively, which exist with noticeable probability as these matrices are full rank with noticeable probability and and is easy to compute using standard Gaussian elimination. Thus, when $\mathcal{A}$ responds with $\tilde{x}$ such that $\mathbf{M}_2\mathbf{B}(\mathbf{A}\mathbf{M}_1\tilde{x}) = \tilde{y}$, which happens with non-negligible probability, $\mathcal{A}'$ outputs $x = \mathbf{M}_1\tilde{x}$. □

Anti solution. Due to technical reasons in our protocols, we also need to argue that it is difficult to find an *anti* solution. We define another game $(\mathsf{antiOWF})$ which is identical to the OWF game, except the adversary needs to output an x such that $\mathbf{B}(1 \oplus (\mathbf{A}x)) = y$ and now argue that this as hard as the OWF game.

Lemma 3.8. *Given an adversary $\mathcal{A}$ that wins the $\mathsf{antiOWF}$ game with noticeable advantage, there exists an adversary $\mathcal{A}'$ that wins the OWF game with noticeable advantage.*

Proof. Given an instance from the OWF game $(\mathbf{A}, \mathbf{B}, y)$, provide the adversary $\mathcal{A}'$ with $(\mathbf{A}, \mathbf{B}, \tilde{y} = \mathbf{B}\{1\}^m - y)$. With noticeable probability, the new instance is indistinguishable from an honestly sampled instance by Corollary 3.5 and Lemma 3.6, as the distribution of the image is close to uniformly random and the AM-OWF covers a noticeable fraction of $\mathbb{F}_3^t$. Hence, $\mathcal{A}$ outputs x such that $\mathbf{B}(1 \oplus \mathbf{A}x) = \tilde{y}$ with probability at least $\varepsilon\varepsilon'$. $\mathcal{A}'$ then outputs x as the solution to OWF challenge $(\mathbf{A}, \mathbf{B}, y)$. It is easy to see that if $\mathbf{B}(1 \oplus \mathbf{A}x) = \tilde{y}$, then $\mathbf{B}(\mathbf{A}x) = y$ because the flip operation can be emulated in $\mathbb{F}_3$ as $\mathbf{B}(1 - (\mathbf{A}x))$ where $\mathbf{A}x$ is first embedded in $\mathbb{F}_3$. □

Approximate solution. We define a final game $\mathsf{approxOWF}(n, m, t)$, which is identical to the OWF game but with a relaxed requirement on the solution where the adversary is given $(\mathbf{A}, \mathbf{B}, y)$, and must output (x, v, v') such that $v = \mathbf{A}x$, $y = \mathbf{B}v'$, and $\Delta_H(v, v') \leq 1$. In fact, our result holds for any $\Delta_H(v, v') \leq O(1)$.

Lemma 3.9. *Given an adversary $\mathcal{A}$ that wins the $\mathsf{approxOWF}(n, m+1, t)$ game with noticeable advantage, there exists an adversary $\mathcal{A}'$ that wins the OWF game with noticeable advantage, for a slightly smaller $\mathsf{OWF}(n, m, t)$ game.*

Proof. Given an OWF challenge $(\mathbf{A}, \mathbf{B}, y)$, $\mathcal{A}'$ first guesses a random position $j \in [1, m+1]$, hoping this is the position that $\mathcal{A}$ will cause an *edit*. $\mathcal{A}'$ then computes an $\mathsf{approxOWF}(n, m+1, t)$ instance by:

1. inserting a random row $a \leftarrow \mathbb{F}_2^n$ at the j-th position of $\mathbf{A}$ to get $\tilde{\mathbf{A}}$;
2. inserting a random column $b \leftarrow \mathbb{F}_3^t$ at the j-th position of $\mathbf{B}$ to get $\tilde{\mathbf{B}}$; and
3. $\tilde{y} \leftarrow y + b.v'[j]$, where $v'[j]$ is the j-th entry of v' and sampled uniformly at random from $\mathbb{F}_3$ as a guess for the modified entry.

Observe that this new instance always has a solution in the $\mathsf{approxOWF}(n, m+1, t)$ game, and the statistical distance from the uniform distribution over $\mathbb{F}_3^t$ is upper bounded by $1/\mathsf{poly}(\lambda)$ (Corollary 3.5). Thus $\mathcal{A}$ will continue to output a solution with non-negligible probability on this new instance. Suppose $\mathcal{A}'$'s guess was correct for the modified entry and let $\mathcal{A}$'s output be (x, v, v') and $\tilde{v}$ be identical to v, except with the j-th entry deleted. Then it is easy to see that $(x, \tilde{v})$ is a solution to the OWF challenge given to $\mathcal{A}'$. □

3.6.2 Combinatorial Analysis

Similar to [CCJ23], we compute a bound on the success probability of a cheating prover who is free to choose the correlations used in the MPCitH. Although the high-level approach is the same, we use very different techniques, relying on the properties of error correcting codes.

Definition 3.10 (Combinatorial bound - informal)**.** A real $\mathsf{p} \in (0, 1)$ is a *combinatorial bound*, if for every incorrect witness x, and every pair $(r, r') \in \mathbb{F}_2^m \times \mathbb{F}_3^m$, the probability over a random permutation π that x satisfies the following equations:

- $v' = z \odot (1 - \pi(r')) + (1 - z) \odot \pi(r')$ with $z = \pi(r) \oplus v$, and $v = \mathbf{A}x$
- $\mathbf{B}v' = y$

is upper-bounded by p.

We begin by making a few observations. As also noted in [CCJ23], for all possible combinations of r, r', the effect of the share conversion step can be mapped to one of three different actions:

- Identity (copy): $v'_i = v_i$, whenever $r_i = r'_i$.

- Flip (flip): $v'_i = 1 \oplus v_i$, whenever $r'_i \in \{0,1\} \wedge r_i \neq r'_i$.
- Constant 2 (const2): $v'_i = 2$, whenever $r'_i = 2$.

A malicious adversary who does not use a valid witness x will arrive at a v such that $\mathbf{B}v \neq y$. Therefore, the only way to satisfy $\mathbf{B}v' = y$ is to modify enough positions of v, to make it a codeword. Since we permute the randomness used, the actions chosen by the adversary are permuted before being used. Intuitively, the larger the number of *flips*, the lower the probability that a permutation aligns the flips with the positions that need to be modified.

We begin with the observation that $\mathbf{B}$ can be viewed as the parity-check matrix of a random linear code where instead of checking whether the parity is 0, we check that it is y. Therefore, $\mathbf{B}v' = y$ if and only if v' is a codeword. Because $\mathbf{B}$ is a random linear code, it has good distance with high probability. Therefore, the possible values of v' are sufficiently *spread* out. Suppose that the minimum number of flip and const2 operations required to reach a particular word v' from v are δ_1 and δ_2 respectively. Then the number of permutations that map v to v' is given by $(m-\delta_1-\delta_2)!\,\delta_1!\,\delta_2! \leq (m-\delta)!\,\delta!$, where $\delta = \delta_1 + \delta_2$. Over the choice of permutations, the probability that the adversary successfully lands on a particular word is upper bounded by $1/\binom{m}{\delta}$. Thus, if we can upper bound the number of codewords, at a distance δ from v, and take a union bound over all of them, we obtain the adversary's probability of successfully cheating using a configuration with δ modifications.

Few flips $(0 < \delta < d/2)$. If the minimum distance of $\mathbf{B}$ is d, then within any hamming ball of radius $< d/2$, there is at most one codeword. Thus, if the adversary modifies fewer than $d/2$ positions, there is at most one codeword that satisfies $\mathbf{B}v' = y$.

Many flips $(d/2 \leq \delta \leq m)$. This case is more tricky because there are now (potentially) many codewords on the hamming ball of radius δ. It is easy to see that the total number of words on the hamming ball is $\binom{m}{\delta}$. However, any codeword is at least a distance d away from any other codeword. Therefore, for every codeword that we try to fit on the hamming ball, there are some number of words *adjacent* to this codeword that cannot also be a codeword. Formally, this set of adjacent words can be described as $\mathcal{B}_\delta = \{c \mid \Delta_H(v',c) \leq d \wedge \Delta_H(v,c) = \delta\}$. Without loss of generality, we can choose v to be the origin $\{0\}^m$ and $v' = \{0\}^{m-\delta}\|\{1\}^{\delta_1}\|\{2\}^{\delta_2}$, where $\delta_1 + \delta_2 = \delta$. The description of $\mathcal{B}_\delta$ can be simplified as $\mathcal{B}_\delta = \{c \mid \mathsf{wt}(c) = \delta \wedge \mathsf{wt}(c - v') \leq d\}$. We can now compute a lower bound on $|\mathcal{B}_\delta|$ and hence an upper bound on C_δ, the number of codewords at a distance δ. Observe that we enumerate over words in $\mathcal{B}_\delta$ by first choosing an $\alpha \leq \min(d/2, \delta, m-\delta)$ subset of the 0 positions that will be flipped to either 1 or 2 and then restoring the weight to δ by zeroing out an α-sized subset of the $1/2$ positions. In addition, one can *flip* the $1/2$ positions to $2/1$, in each of the above configurations. It follows that

$$|\mathcal{B}_\delta| \geq \sum_{\alpha=0}^{\min(d/2,\delta,m-\delta)} \binom{m-\delta}{\alpha}\binom{\delta}{\alpha} 2^\alpha \left(\sum_{\beta=1}^{\min(d-2\alpha,\delta-\alpha)} \binom{\delta-\alpha}{\beta} \right).$$

Therefore, $C_\delta = \lceil \binom{m}{\delta} / |\mathcal{B}_\delta| \rceil$ is an upper bound on the number of codewords on the hamming ball of radius δ. To finish the argument, we simply take a union bound over all the codewords on the hamming ball to obtain an upper bound on the probability of successfully landing on some codeword, $\mathsf{p}_2 \leq C_\delta / \binom{m}{\delta}$. By explicitly plotting the combinatorial bound for the parameters of the AM-OWF, we can see that $|\mathcal{B}_\delta| > \binom{m}{\delta}$, in the regime where $\delta < d/2$ and $\delta > m - d/2$.[7] In other words, a single code word and its adjacent words are sufficient to cover the entire hamming ball. This is illustrated in Figure 1.

Anti solution. Observe that if the adversary found an x such that $\mathbf{B}(1 \oplus \mathbf{A}x) = y$, then they could chose all operations to be flip,[8] and succeed every single time. However, based on Lemma 3.8, we know that this is as hard as the OWF game itself. Looking ahead, when we build a zero-knowledge proof of knowledge of a preimage of the AM-OWF, we actually only achieve a weakened notion of knowledge soundness that extracts either a preimage or an anti solution. Since we have shown that both of these problems are hard, it suffices to compile our MPCitH protocol to

[7] In fact, there is only one codeword on a hamming ball at distance $\delta > m - d^*$ for some $d^* > d/2$, as seen in Figure 1.

[8] The adversary could also choose all operations to be const2, but with overwhelming probability $\{2\}^m$ is not in the code.

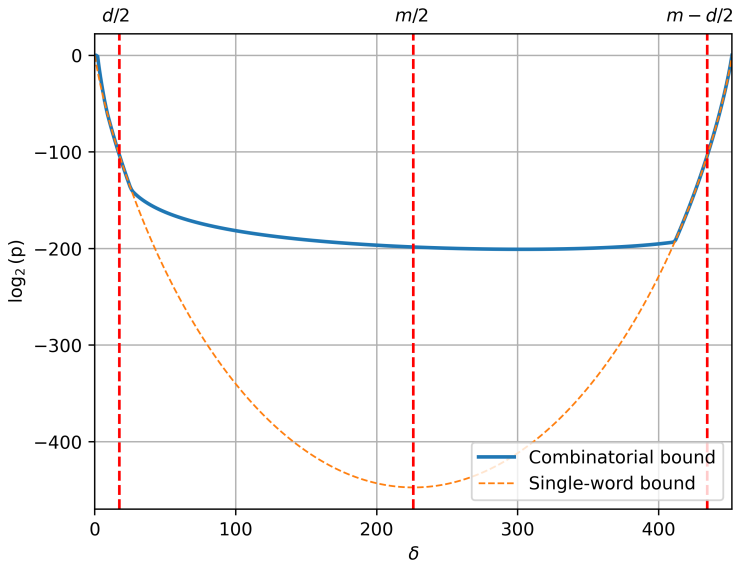

Figure 1: The blue line denotes combinatorial bound for different values of δ for the parameters $(m, d) = (452, 35)$. The dotted orange line is $1/\binom{m}{\delta}$, the probability with which an adversary succeeds in landing on a particular codeword δ hamming distance away. The gap between the two curves highlights the impact of a growing number of codewords on the adversary's chance of success. Importantly, the best strategy is for the adversary to make as few flips as possible (or flip almost all but a few) to avoid detection.

a signature scheme. Similar to [CCJ23], we further relax the soundness notion to an appropriately defined *approximate* notion, where the extractor is able to find an x such that $\mathbf{B}(\mathbf{A}x) = v$, and there exists a v' such that $\mathbf{B}v' = y$ and $\Delta_H(v, v') \leq f$ for an appropriately chosen f. By appealing to Lemma 3.9, we know that for $f = O(1)$, this problem is also hard, thus in each parallel repetition of the MPCitH protocol, the chance of the adversary cheating is smaller, thereby requiring fewer repetitions.

4 Protocols and Evaluation

We focus on the semi-honest two party setting where the input x and key k are each known to each of the two parties, respectively. The structure of our weak PRF lends itself to particularly efficient implementation in this setting. However, we note that other settings such as where x, k are secret shared can easily be implemented using similar techniques. Malicious security can be achieved but we leave an efficient specification of this to future work. Our protocols will assume that the output should be secret shared. In the case of an OPRF, it is straightforward to reveal the result to the party with x.

4.1 Review of Silent OT/VOLE/OLE

To fully appreciate the design decision of our protocols it is important to understand how silent OT and VOLE protocols work [BCG+19a, BCG+19b, RRT23]. At the root of these protocols is the computational hard problem known as syndrome decoding. For a secret random sparse vector $e \in \mathbb{F}_2^\nu$, and a public error correcting code generator matrix $G \in \mathbb{F}_2^{n \times \nu}$, the syndrome decoding assumption state that $(G, G \cdot e)$ is indistinguishable from (G, r) for a uniform r. Note that $n < \nu$, e.g., $\nu = 2n$, and therefore $G \cdot e$ compresses e. Security of this assumption typically holds if G has high minimum distance [RRT23].

The silent OT/VOLE is a two party protocol with a sender and a receiver. In its simplest formulation the goal is for the receiver to hold random vectors $a \in \mathbb{F}_2^n, B \in \mathbb{F}_{2^\lambda}^n$ and the sender to hold scalar $\Delta \in \mathbb{F}_{2^\lambda}$ and a vector

$C \in \mathbb{F}_{2^\lambda}^n$ such that $B + C = a\Delta$. One can view B, C as a secret sharing of $a\Delta$. The protocol proceeds by having the receiver sample a sparse vector $\hat{a} \in \mathbb{F}_2^\nu$ while the sender samples a random $\Delta \in \mathbb{F}_{2^\lambda}$. The parties generate a secret sharing of $\hat{a}\Delta \in \mathbb{F}_{2^\lambda}$ using a punctured PRF technique [BCG+19a]. Leveraging the sparsity of $\hat{a}$, this only requires $O(\lambda \log(n/\lambda))$ communication, where typical parameters have $\nu \approx 2^{20}$. Let $\hat{C}, \hat{B} \in \mathbb{F}_{2^\lambda}^\nu$ denote the shares of $\hat{a}\Delta$. The idea is to then compress these vectors to get $a = G \cdot \hat{a}, B = G \cdot \hat{B}, C = G \cdot \hat{C}$. Since multiplication by G is linear, the correlation $B + C = a\Delta$ still holds. However, by the syndrome decoding assumption, a will be pseudorandom. This protocol can more generally be referred to as $(\mathbb{S}, \mathbb{E})$-subfield VOLE where we have a subfield $\mathbb{S}$ and an extension field $\mathbb{E} = \mathbb{S}^\sigma$ for some σ. The correlation $B = C + a\Delta$ holds for $a \in \mathbb{S}^n$ and $B, C \in \mathbb{E}^n, \Delta \in \mathbb{E}$. The description above is of course for $\mathbb{S} = \mathbb{F}_2$ and $\sigma = \lambda$.

As we mentioned before, one can efficiently multiply many scalars $x^{(1)}, ..., x^{(q)} \in \mathbb{S}$ by a vector $k \in \mathbb{S}^m$ using subfield VOLE. The parties first generate a $(\mathbb{S}, \mathbb{S}^m)$-subfield VOLE correlation of size q where $\Delta = k$. The receiver can then send the difference $x - a$ and have the sender update their correlation such that $B + C = xk$. One can obtain random one-out-of-two OT from $(\mathbb{F}_2, \mathbb{F}_{2^\lambda})$-subfield VOLE. Random OT refers to the correlation where the sender holds random strings $m_{i,0}, m_{i,1}$ while the receiver holds a bit a_i and the string m_{i,a_i}. This can be achieved by defining sender messages $m_{i,0} = H(C_i), m_{1,1} = H(C_i + \Delta)$ where the receiver knows random choice bit a_i and message $m_{i,a_i} = H(B_i)$. This can be generalized to one-out-of-p OTs by performing a $(\mathbb{F}_p, \mathbb{F}_{p_\sigma})$-subfield VOLE and defining $m_{i,j} = H(C_i + j\Delta)$. While possible to instantiate with any p, the implementation will be most efficient when p is a power of two. In this case $GF(p)$ operations can efficiently be implemented. Finally, one can obtain a random binary OLE correlation from a random OT correlation, where the receiver holds $x_0, y_0 \in \mathbb{F}_2$ and the sender holds $x_1, y_1 \in \mathbb{F}_2$ such that $x_0 + x_1 = y_0 y_1$. From a single random OT $(m_0, m_1), (a, m_a)$, the parties can define $y_0 = a, y_1 = \mathsf{lsb}(m_0 + m_1), x_0 = \mathsf{lsb}(m_a), x_1 = \mathsf{lsb}(m_0)$.

4.2 Secret-Sharing-Based $\mathbb{F}_3 \to \mathbb{F}_2$ Modulus Conversion

We now describe how to perform modulus conversion from $[\![w]\!]^3 \in \mathbb{F}_3^m$ to $[\![v]\!]^3 \in \mathbb{F}_2$, where $v = w \bmod 2$. For simplicity, let $m = 1$ and the inputs of the two parties P_0, P_1 are shares (w_0, w_1) such that $w = w_0 +_3 w_1$. The truth table T of $w \bmod 2$ is described in Table 1. P_0 with input w_0 will use the w_0-th row of T, denoted as $T[w_0, :]$, as the

		w_1		
		0	1	2
w_0	0	0	1	0
	1	0	0	1
	2	1	0	0

Table 1: Truth table T of $w_0 \oplus w_1$ where $w_0, w_1 \in \mathbb{F}_3$

sender input in a 1-out-of-3 OT and P_1 will pick up $T[w_0, w_1]$, using w_1 as the receiver input. Looking ahead, we will actually want the parties to obtain shares of $T[w_0, w_1]$. To achieve this, P_0 can simply choose a random $r \in \{0, 1\}$ and use $T[w_0, :] \oplus r$ as its input to the OT. By correctness, P_1 will receive $T[w_0, w_1] \oplus r$. In particular, let us assume the parties hold a 1-out-of-3 random bit OT, where P_0 holds random message bits $m_0, m_1, m_2 \in \mathbb{F}_2$ while P_1 holds a random choice $c \in \mathbb{F}_3$ and the corresponding message m_c. These will be used to mask the truth table T. That is, P_0 will sample mask r and send $t \in \mathbb{F}_3^3$ where

$$\begin{aligned} t_0 &= m_0 + T[w_0, 0] + r, \\ t_1 &= m_1 + T[w_0, 1] + r, \\ t_2 &= m_2 + T[w_0, 2] + r. \end{aligned}$$

P_1 can therefore compute

$$z_1 = t_{u_1} +_3 m_{w_1} = T[w_0, w_1] +_3 r = (w \bmod 2) +_3 r$$

and P_0 can compute $z_0 = r$ and therefore we have $z_0 \oplus z_1 = (w \bmod 2)$.

Reducing the communication. As an optimization, we do not need to send t_0. The idea is that if the receiver wants to learn $T[u_0, 0]$ it can set its share as m_0. That is,

$$\begin{aligned} z_1 &= \text{if } (u_1 = 0) \text{ then } m_0 \text{ else } t_{u_1} +_3 m_{u_1} \\ &= u_{1,0} \cdot_3 t_{u_1} + m_{u_1}. \end{aligned}$$

The sender can now define r appropriately as $r := m_0 + T[w_0, 0]$. In the case of $w_1 = 0$, we have $z_1 = m_0$ and $z_0 = r + T[w_0, 0]$, and therefore we get the right result. The other cases are the same with the randomness of the mask r coming from m_0.

Protocol. Figure 2 presents protocol in the 1-out-of-p OT hybrid setting, where we assume $p \geq 3$. Below, we present how to implement such OTs.

Theorem 4.1. *The protocol $\Pi_{\text{ot-3}\to\text{2-conv}}$ securely realizes the $\mathbb{F}_3 \to \mathbb{F}_2$ modulus conversion functionality in the binary OLE hybrid model.*

Proof. The simulation is to send uniformly random values. The case of a corrupt P_0 is simple, y_1 acts as a one-time pad key. Consider a corrupt P_1. Observe that the t_0, t_1 values are masked by two unknown messages. Since these message values are uniform in the view of P_1, so are the t values. □

We give two conceptual methods for constructing random 1-out-of-3 OTs for single-bit messages. The first is based on standard OT while the second is based on subfield VOLE.

$\Pi_{\text{ot-3}\to\text{2-conv}}$: $\mathbb{F}_3 \to \mathbb{F}_2$ modulus conversion from 1-out-of-p OT

Setup: The parties generate a random 1-out-of-p bit OT with $(m_0, ..., m_{p-1}) \in \mathbb{F}_2^p$ held by P_0 and $(c, m_c) \in \mathbb{F}_p \times \mathbb{F}_2$ held by P_1.

Eval:

1. P_0 inputs $w_0 \in \mathbb{F}_3$ and P_1 inputs $w_1 \in \mathbb{F}_3$.
2. Let $w_{j,0}, w_{j,1} \in \mathbb{F}_2$ be the bit decomposition of w_j, i.e., $w_j = w_{j,0} + 2w_{j,1}$.
3. P_1 sends $d = w_1 - c \bmod p$ to P_0.
4. P_0 computes
   - $v_0 = w_{0,0} \oplus m_d$
   - $t_0 = v_0 \oplus m_{d+1} \oplus w_{0,0} \oplus w_{0,1} \oplus 1$
   - $t_1 = v_0 \oplus m_{d+2} \oplus w_{0,1}$

   and sends (t_0, t_1) to P_1.
5. P_1 compute $v_1 = (w_{1,0} \cdot t_0) \oplus (w_{1,1} \cdot t_1) \oplus x_{1,0} \oplus x_{1,1}$.
6. P_i outputs v_i.

Figure 2: Our protocol for $\mathbb{F}_3 \to \mathbb{F}_2$ modulus conversion from 1-out-of-p OT correlations.

From 1-out-of-2 OT. The generic method for constructing 1-out-of-2^k OTs is to generate k 1-out-of-2 OTs. The random message corresponding to $c \in \mathbb{F}_{2^k}$ will then be the hash of the messages indexed by the bit decomposition of c. However, given that only three messages are used by Figure 2, we present an optimized approach without the need for additional hashing.

We can build a 1-out-of-3 OT for bit messages from binary OLE or equivalently 1-out-of-2 OT. Each 1-out-of-3 OT consumes 2 binary OLEs/OTs. Recall that a binary OLE correlation is defined as a pair of tuples (x_0, y_0) and (x_1, y_1), held by P_0 and P_1 respectively, which satisfy $x_0 \oplus x_1 = y_0 \cdot y_1$. An OLE can be obtained from standard random OT as discussed in Section 4.1. It is possible to partially derandomize an OLE correlation by allowing P_1 to change their y_1 to be a chosen value, y_1^*. The parties will now hold correlation $(x'_0, y'_0, x'_1, y_1^*)$ where x'_0, x'_1, y'_0 are random and y_1^* is chosen by P_1. A random binary OLE correlation can be converted into a chosen one if P_1 sends $d = (y_1 \oplus y_1^*)$, and the parties update their share as $x'_0 = x_0 \oplus y_0 \cdot d, y'_0 = y_0, x'_1 = x_1$. We will derandomize two OLEs using P_1's $u_1 \in \mathbb{F}_3$. This value is represented using two bits $(u_{1,0}, u_{1,1})$ such that $u_1 = u_{1,0} +_3 2 \cdot_3 u_{1,1}$. Let us define the resulting OLEs as $(x_0, x_1, y_0, u_{1,0}), (x'_0, x'_1, y'_0, u_{1,1})$. We will define the random 1-out-of-3 OT messages (in this case single bits) as

$$\begin{aligned} m_0 &= x_0 \qquad \oplus x'_0, \\ m_1 &= x_0 \oplus y_0 \oplus x'_0, \\ m_2 &= x_0 \qquad \oplus x'_0 \oplus y'_0. \end{aligned}$$

From subfield VOLE. We observe that it is possible to further optimize the modulus conversion protocol above by leveraging the capabilities of subfield VOLE. The rational for using binary OLE is that one can efficiently generate this correlation from highly optimized 1-out-of-2 silent OT protocols. However, such protocols can also directly generate 1-out-of-p OTs by first constructing a $(\mathbb{F}_p, \mathbb{F}_{p^\sigma})$ subfield VOLE and then hashing the extension field elements. For our use case, the most natural choice would be to set $p = 3$ and σ such that $2^\lambda \approx p^\sigma$. This would in turn allow the derandomization message to be a single $\mathbb{F}_3$ element, resulting in a total of 3.58 bits of communication instead of 4. However, this comes at a high cost due to the requirement of the silent VOLE protocol using a modulus that is not CPU friendly. One would optimize subfield VOLE using bit decomposition techniques but the computation cost will likely remain high, e.g., 5 times slower by our estimates. As such, we do not think this communication optimization is worth the computation pessimization. We suggest an alternative that essentially halves the computational cost of the already optimized OLE protocol while retaining the same communication overhead of 4 bits. The idea is relatively simple, set $p = 4$ and $\sigma = \lambda/2$. Since $\mathbb{F}_4$ operations naturally map to CPU instructions, one can implement an $(\mathbb{F}_4, \mathbb{F}_{2^\lambda})$-subfield VOLE with essentially the same overhead as $(\mathbb{F}_2, \mathbb{F}_{2^\lambda})$-subfield VOLE used to generate the binary OLEs. By directly using subfield $\mathbb{F}_4$ to perform n modulus conversions, only n subfield VOLE correlations are required instead of $2n$ OTs and therefore the preprocessing cost is halved.

Concurrent work. We would like to point out the concurrent work [IKNZ23] for constructing $\mathbb{F}_3$ to $\mathbb{F}_2$ conversion protocols in the OT hybrid model. Their optimized construction showed that they can generate a modulus conversion preprocessing pair $(\llbracket r \rrbracket^2, \llbracket r \rrbracket^3)$ using an amortized 1.33 OLEs/OTs and 3.08 bits of communication, or at the cost of 18 OLEs/OTs and 2.55 bits. This pair can then be used to convert $\llbracket w \rrbracket^3$ into $\llbracket w \rrbracket^2$ using an additional $1.58 = \log_2(3)$ bits of communication, totaling 4.66 or 4.13 bits per modulus conversion. Our techniques on the other hand require two OLEs/OTs or one $\mathbb{F}_4$ VOLE and only 4 bits of communication. Alternatively, one $\mathbb{F}_3$ VOLE and just 3.58 bits of communication. Although their construction can allow for fewer OLEs, e.g., 1.33 versus 2, the surrounding protocol is more complicated and therefore it is not clear which would be more efficient in practice. Moreover, our $\mathbb{F}_4$ approach is more efficient while sending less data. We leave determining the concrete cost of our communication-optimized $\mathbb{F}_3$-VOLE approach to future work.

4.3 Secret-Sharing-Based $(\mathbb{F}_3, \mathbb{F}_2)$-wPRF Protocol

Given that we aim to have secret-shared output and binary sharings are standard, we begin with our $(\mathbb{F}_3, \mathbb{F}_2)$-wPRF which outputs a vector over $\mathbb{F}_2$. In addition, we will consider the optimized variant where the key is binary. Following the analysis in Section 3.5, we increase the input dimension to $n = 4\lambda$. In particular, we will evaluate

$$F(k, x) = \mathbf{B} \cdot_2 (\mathbf{A} \cdot_3 [k \odot x])$$

where $k \in \mathbb{F}_2^n, x \in \mathbb{F}_3^n, \mathbf{A} \in \mathbb{F}_3^{m \times n}, \mathbf{B} \in \mathbb{F}_2^{m \times t}$.

We consider two primitives for building these protocols, oblivious transfer and garbled circuits. Our first protocol will only make use of the former while the latter will use both. In all cases we choose to optimize the *overall* overhead of the protocols in terms of computational, communication, and round complexity. Where it does not add additional overheads, we will make use of a prepossessing phase. This primarily takes the form of generating oblivious transfer correlations which can later be used in the main protocol.

Reusable key correlations. Our first technique use oblivious transfer to multiply $k \odot x$. This can be achieved by having the party holding k act as the OT receiver with k_i as their choice bit.[9] This choice allows us to perform these OTs ahead of time, once k is fixed but before x is known, and later reuse them for each input x. The OT sender will provide two messages, $(s_i, s_i +_3 x_i)$, where s_i is sampled uniformly at random. Therefore the sender will learn $f_i = s_i +_3 (k_i \cdot_3 x_i)$. Let us interpret s, f as the individual shares of $[\![u]\!]^3$ where $u = k \odot_3 x$. Given this $\mathbb{F}_3^n$ sharing of $k \odot_3 x$, the parties can locally multiply with the public matrix $\mathbf{A} \in \mathbb{F}_3^{m \times n}$, i.e., $[\![w]\!]^3 := \mathbf{A} \cdot_3 [\![u]\!] = \mathbf{A} \cdot_3 [\![k \odot_3 x]\!]$.

Theorem 4.2. *The protocol of Figure 3 evaluates the $(\mathbb{F}_3, \mathbb{F}_2)$-wPRF of Definition 3.2 with semi-honest security in the $\Pi_{\text{ot-3}\to\text{2-conv}}$ hybrid model where P_0 inputs the key k and P_1 inputs x and they receive $[\![F(k,x)]\!]$ as output.*

OT-based $(\mathbb{F}_3, \mathbb{F}_2)$-wPRF protocol with shared output.

Setup:

1. P_0 inputs $k \in \mathbb{F}_2^n$.
2. The parties perform n random OTs where P_0 is OT receiver with choice bit k_i. P_1 receives two random strings $\sigma_{i,0}, \sigma_{i,1} \in \{0,1\}^\lambda$ and P_0 receives σ_{i,k_i}.
3. Let $G_{i,0}, G_{i,1}$ denote stateful PRGs with $\mathbb{F}_3$ output held by P_1 with seeds $\sigma_{i,0}, \sigma_{i,1}$ respectively.
4. Let G'_i denote a stateful PRG with $\mathbb{F}_3$ output held by P_0 with seed σ_{i,k_i}.

Eval:

1. P_1 inputs $x \in \mathbb{F}_3^n$.
2. The parties run the setup for $\Pi_{\text{ot-3}\to\text{2-conv}}$ for m conversions.
3. P_1 computes
   (a) $h_{0,i} \leftarrow G_{i,0}$ for $i \in [n]$
   (b) $h_{1,i} \leftarrow G_{i,1}$ for $i \in [n]$
   (c) $f := x -_3 h_0 -_3 h_1$
   (d) $w_1 := \mathbf{A} \cdot_3 h_0$
4. P_0 computes
   (a) $t_i \leftarrow G'_i$ for $i \in [n]$
   (b) $w_0 := \mathbf{A} \cdot_3 ((k \odot f) +_3 t)$
5. P_0, P_1 invoke $\Pi_{\text{ot-3}\to\text{2-conv}}$ with w_i as the input for P_i. Let v_i be the output for P_i.
6. P_i outputs $\mathbf{B} v_i$.

Figure 3: Our $(\mathbb{F}_3, \mathbb{F}_2)$-wPRF protocol based on OLE with plaintext inputs and secret-shared output.

Proof. Consider a corrupt P_0. Their view consists of the random OT strings $\sigma_{i,k_i} \in \{0,1\}^\lambda, a, c \in \mathbb{F}_2^{2\times m}, f \in \mathbb{F}_3^n$ and their view of $\Pi_{\text{ot-3}\to\text{2-conv}}$. The simulation will send uniformly random strings for each of these. Recall that P_0 knows either s_i or h_i while the other is uniformly distributed due to $\sigma_{i,1\oplus k_i}$ being uniformly distributed. Therefore,

[9] If one does not employ the binary key optimization, this can be generalized to a one-out-of-three OT as used later.

over the random choices of $\sigma_{i,1\oplus k_i}$, f is uniformly distributed and independent of x. Finally, we invoke the simulator for $\Pi_{\text{ot-3}\to\text{2-conv}}$. Now consider a corrupt P_1. Their view consists of the OT strings σ and their view of $\Pi_{\text{ot-3}\to\text{2-conv}}$. The proof is complete by noting that σ and $\Pi_{\text{ot-3}\to\text{2-conv}}$ can be simulated. □

Overheads. Assuming the setup phase is reused, the protocol consumes $2m$ random OT/OLE correlations. When generated using a silent OT protocol [BCG+19a, BCG+19b, RRT23], the amortized cost of the OTs would be $O(\lambda)$ computation and less than one bit of communication. The main phase of the protocol derandomizes the OLEs using two bits of communication and while sending t_2, t_3 (two bits), totaling to a combined $2m + (2m + n)\log_2(3)$ bits. We also note that the recent development of OT protocols with constant overhead by Boyle *et al.* [BCG+23] implies that our share conversion protocol can be evaluated in constant amortized work, independent of the security parameter λ. Moreover, if $\mathbf{A}, \mathbf{B}$ are implemented using linear-time encodable codes, the overall running amortized time of our protocol is $O(\lambda)$ work per evaluation.

OT-based $(\mathbb{F}_2, \mathbb{F}_3)$-wPRF protocol with shared output.

Setup:

1. P_0 inputs $k \in \mathbb{F}_2^n$.
2. The parties perform n random OTs where P_0 is OT receiver with choice bit k_i. P_1 receives two random strings $\sigma_{i,0}, \sigma_{i,1} \in \{0,1\}^\lambda$ and P_0 receives σ_{i,k_i}.
3. Let $G_{i,0}, G_{i,1}$ denote stateful PRGs with $\mathbb{F}_2$ output held by P_1 with seeds $\sigma_{i,0}, \sigma_{i,1}$ respectively.
4. Let G'_i denote a stateful PRG with $\mathbb{F}_2$ output held by P_0 with seed σ_{i,k_i}.

Eval:

1. P_1 inputs $x \in \mathbb{F}_2^n$.
2. The parties preprocess/generate m random OTs where P_0 holds $s \in \mathbb{F}_3^{2\times m}$ and P_1 holds $d \in \mathbb{F}_2^m, s' = (s_{d_1,1}, ..., s_{d_m,m})$.
3. P_1 computes
   (a) $h_{0,i} \leftarrow G_{i,0}$ for $i \in [n]$
   (b) $h_{1,i} \leftarrow G_{i,1}$ for $i \in [n]$
   (c) $f := x \oplus h_0 \oplus h_1$
   (d) $u := \mathbf{A} \cdot_2 h_0$
   (e) $\delta := u \oplus d$

   and sends f, δ to P_0.
4. P_0 computes
   (a) $g_i \leftarrow G'_i$ for $i \in [n]$
   (b) $v := \mathbf{A} \cdot_2 ((k \odot f) \oplus g)$
   (c) $t := v -_3 s_0 -_3 s_1$
   (d) $w := \mathbf{B} \cdot_3 (s_{\delta_1,1}, ..., s_{\delta_m,m})$

   and sends t to P_1.
5. P_1 computes $q := \mathbf{B} \cdot_3 [s' +_3 (t \odot d)]$.
6. P_0 outputs w and P_1 outputs q.

Figure 4: Our $(\mathbb{F}_2, \mathbb{F}_3)$-wPRF protocol based on OLE with plaintext inputs and secret-shared output.

4.4 OT-Based $(\mathbb{F}_2, \mathbb{F}_3)$-wPRF Protocol

We present an OT-based protocol in Figure 4 for our $(\mathbb{F}_2, \mathbb{F}_3)$-wPRF where the inputs are known to the respective parties and the output is shared. Compared to the previous case, this protocol achieves better computational efficiency at the expense of having a mod 3 output domain. However, for applications such as OPRF this has little impact. Conceptually, the protocol works in a similar way. First shares of $x \cdot k$ are computed using (preprocessed) OTs based on the key. The parties additionally generate m random OTs with strings in $\mathbb{F}_3$. These are used to make the $\mathbb{F}_2$ to $\mathbb{F}_3$ modulus conversion. Since the inputs are already binary, each mod gate requires only one OT as opposed to two OLEs in the previous protocol. In the amortized setting, this protocol requires m OTs, two rounds of interaction and $2.6m+n$ bits of communication per evaluation. To convert the secret-shared-output protocol to an OPRF, an additional $1.6t$ bits of communication and zero additional rounds are required.

Theorem 4.3. *The protocol of Figure 4 evaluates the $(\mathbb{F}_2, \mathbb{F}_3)$-wPRF of Definition 3.2 with semi-honest security in the random OLE/OT hybrid model where P_0 inputs the key k and P_1 inputs x and they receive $[\![F(k, x)]\!]$ as output.*

Proof. The simulation of the protocol is to send uniformly random messages. Consider a corrupt P_0. Their view consists of f, δ. The former is uniformly distributed given that either $h_{0,i}$ or $h_{1,i}$ is uniformly distributed. The latter is uniformly distributed due to d being uniform. Now consider a corrupt P_1. Their view consists of t, which is uniformly distributed due to either $s_{0,i}$ or $s_{1,i}$ being uniformly distributed. The simulation of the output distribution trivially follows from the correctness of the protocol. □

4.5 Specialized Garbling

We now describe a specialized garbling scheme [Yao86] for evaluating our wPRF. We present this protocol only for the $(\mathbb{F}_2, \mathbb{F}_3)$-wPRF but note that this can be generalized in a natural way. The core idea is that we can utilize free XOR with specialized unary gates for performing the modulus conversion step. As before, the parties will preprocess OTs using the key as the choice bit. P_1 will act as the garbler with $\Delta \leftarrow \{0,1\}^\lambda$ being the global free-XOR key [KS08]. Instead of garbling labels for x and k, the party P_1 will generate garbled labels $L_i \leftarrow \{0,1\}^\lambda$ where L_i is the zero label for $x_i \cdot k_i$. In particular, the parties will use the preprocessed OTs for k with messages $(L_i, L_i \oplus x_i\Delta)$. The party P_0 will learn $L'_i = L_i \oplus (x_i k_i)\Delta$. The parties P_1, P_0 can multiply these labels with $\mathbf{A}$ to obtain U_i, U'_i respectively. P_1 garbles a unary gate that maps the zero label to a random $r_i \in \mathbb{F}_3$ value while one label is mapped to $r_i +_3 1$. Unlike in traditional garbling, we desire the output of the mod gate to be a single $\mathbb{F}_3$ element and as such the garbled table can be much smaller. In particular, with the use of the point-and-permute technique the garbled table can be a single $\mathbb{F}_3$ element. We will restrict the free-XOR key Δ to have a 1 in its least significant bit. As such, for the least significant bits of U_i, U'_i we have

$$\begin{aligned} u_i &= \mathsf{lsb}(U_i) \oplus \mathsf{lsb}(U'_i) \\ &= \mathsf{lsb}(U_i) \oplus \mathsf{lsb}(U_i \oplus u_i\Delta) \\ &= \mathsf{lsb}(U_i) \oplus \mathsf{lsb}(U_i) \oplus u_i\mathsf{lsb}(\Delta). \end{aligned}$$

We wish to translate this secret sharing into an $\mathbb{F}_3$ sharing with the same underlying value. For now, let us assume that P_0 knows $u_i = 0$. P_1 can define $r_i = -H(U_i)$. If P_0 has $U'_i = U_i$, i.e., $u_i = 0$, then they can also derive r_i which forms a secret sharing of zero. However, if P_0 holds the one label $U'_i = U_i \oplus \Delta$, then they will need assistance to compute $r_i + 1$. This is done by sending a garbled table containing the difference between $r_i + 1$ and $H(U'_i)$. Lastly, it is important that P_0 does not know the underlying value. However, in the explanation above we assume P_0 can conditionally add the difference based on the underlying value. This can be circumvented using the point-and-permute technique. The full protocol is given in Figure 5. The overhead of this protocol is $\lambda n + \log_2(3)m$ bits and requires no OTs after the initial set of OTs for the key. Indeed, this makes conceptually sense due to the garbler not needing OTs for their input, i.e., x. We note that this protocol can easily be extended to the $(\mathbb{F}_3, \mathbb{F}_2)$-wPRF. First the free-XOR can be generalized to work over the $\mathbb{F}_{3^\lambda}$ extension field as opposed to $\mathbb{F}_{2^\lambda}$, see [BMR16]. The unary gates can then consist of two 1-bit values.

Theorem 4.4. *The protocol of Figure 5 evaluates the $(\mathbb{F}_2, \mathbb{F}_3)$-wPRF of Definition 3.2 with semi-honest security in the random oracle and random OT hybrid model where P_0 inputs the key k and P_1 inputs x and they receive $[\![F(k,x)]\!]$ as output. We assume that $H : \{0,1\}^* \to \mathbb{F}_3$ is a random oracle.*

Proof. The simulation of the protocol is to send uniformly random messages. The simulation follows the standard free-XOR garbling argument. Essentially, a corrupt P_0 can only distinguish if they query the random oracle at some input $U'_i \oplus \Delta$. If no such query is made, the simulator can replace all such queries with uniformly random values and then simulation follows immediately. Assuming the adversary makes such a query, then they have essentially guessed Δ. However, this can only happen with negligible probability. □

Garbling-based $(\mathbb{F}_2, \mathbb{F}_3)$-wPRF protocol with shared output.

Setup:

1. P_0 inputs $k \in \mathbb{F}_2^n$.
2. The parties perform n random OTs where P_0 is OT receiver with choice bit k_i. P_1 receives two random strings $\sigma_{i,0}, \sigma_{i,1} \in \{0,1\}^\lambda$ and P_0 receives σ_{i,k_i}.
3. Let $G_{i,0}, G_{i,1}$ denote stateful PRGs with $\mathbb{F}_{2^\lambda}$ output held by P_1 with seeds $\sigma_{i,0}, \sigma_{i,1}$ respectively.
4. Let G'_i denote a stateful PRG with $\mathbb{F}_{2^\lambda}$ output held by P_0 with seed σ_{i,k_i}.

Eval:

1. P_1 inputs $x \in \mathbb{F}_2^n$.
2. P_1 computes
   (a) $\Delta \leftarrow \{0,1\}^{\lambda-1}||1$
   (b) $h_{0,i} \leftarrow G_{i,0}$ for $i \in [n]$
   (c) $h_{1,i} \leftarrow G_{i,1}$ for $i \in [n]$
   (d) $f := x\Delta \oplus h_0 \oplus h_1$
   (e) $U := \mathbf{A} \cdot_2 h_0$
   (f) $r_i := -H(U_i \oplus \mathsf{lsb}(U_i)\Delta) +_3 \mathsf{lsb}(u_i)$ for $i \in [n]$
   (g) $\delta_i := -r_i + \overline{\mathsf{lsb}(U_i)} -_3 H(U_i + \overline{\mathsf{lsb}(U_i)}\Delta)$ for $i \in [n]$
   (h) $q := \mathbf{B} \cdot_3 r$

   and sends f, δ to P_0.
3. P_0 computes
   (a) $g_i \leftarrow G'_i$ for $i \in [n]$
   (b) $U' := \mathbf{A} \cdot_2 ((k \odot f) \oplus g)$
   (c) $v_i := H(U'_i) +_3 \mathsf{lsb}(U'_i)\delta_i$ for $i \in [n]$
   (d) $w := \mathbf{B} \cdot_3 v$
4. P_0 outputs w and P_1 outputs q.

Figure 5: Our $(\mathbb{F}_2, \mathbb{F}_3)$-wPRF protocol based on garbling with plaintext inputs and secret-shared output.

4.6 Protocol Evaluation

We implement our wPRF protocols for the OPRF setting as well as plaintext inputs and secret-shared output. We build on [RR] and intend to open source the implementation. The protocols generate all correlated randomness via silent OT techniques. We compare their performance to the alternating-moduli protocol of [DGH+21] and our GMW-based implementation of LowMC [ARS+15]. Our LowMC implemention employs various optimizations such as transposed

representation, vectorization, and precomputed key schedule. LowMC offers a variety of parameters that give tradeoffs between computational overhead, round complexity, and number of OTs/OLEs that are required. We parameterize it as $(n, m, k, d, r) \in \{(256, 63, 128, 128, 14), (128, 3, 128, 128, 88)\}$. Running times were obtained on a core i7 consumer-grade laptop with 16GB RAM. Each party is executed on a separate thread with network communication being simulated, i.e., minimal communication overhead. Given that our protocol is 2 rounds, one can easily estimate the cost of network communication by dividing the communication by the bandwidth plus network latency.

Protocols with the key OT optimization perform a setup phase where OTs for each bit of the key are performed. This reusable correlated randomness allows the parties to compute shares of $(k \odot x)$. Alternatively, our input-OTs protocol for multiplying $(k \odot x)$ requires more OTs but less communication. We implement two version of our modulus conversion protocol $\Pi_{\text{ot-3}\to\text{2-conv}}$ for the $(\mathbb{F}_3, \mathbb{F}_2)$-wPRF protocol which uses m $\mathbb{F}_4$-VOLE correlations or $2m$ OT/OLE correlations. We refer to each as $\mathbb{F}_4$-VOLE and $\mathbb{F}_2$-OLE respectively. The modulus conversion for $(\mathbb{F}_2, \mathbb{F}_3)$-wPRF requires just m OT/OLE correlations. The final protocol uses garbling techniques to implement modulus conversion and does not require any additional OTs beyond the reusable OTs for the key.

OPRF. In Figure 6 we report the metrics for our protocol when used as an OPRF in comparison to related works. We consider the setting where a *succinct* setup phase is performed. During this phase, the key is known but the evaluation points, i.e., x, are not. For [DGH+21] and our protocols in Figure 6, we separate the preprocessing p and online o metrics as $p + o$.

Scheme	Assumption	r	Communication (bits)	time (μs)
[ADDS21]	R(LWE) & SIS	2	2^{24}	–
[SHB23]	Legendre PRF	$+3$	$+2^{24}$	–
[BKW20]	CSIDH	3	2^{21}	–
[Bas23]	SIDH	2	2^{25}	–
[HHM+24]	CSIDH	$+2$	$+2^{17.4}$	–
[ADDG24]	AM-[BIP+18]	2	$3,821$	$151,000$
[DGH+21]	AM-[DGH+21]	2	$65 + 1,252$	$25.4 + 6.1$
OT - $(\mathbb{F}_2, \mathbb{F}_3)$-wPRF	**AM**	**2**	$\mathbf{38 + 916}$	$\mathbf{7.0 + 0.4}$
$\mathbb{F}_2$-OLE - $(\mathbb{F}_3, \mathbb{F}_2)$-wPRF	AM	2	$38 + 1,173$	$14.6 + 0.4$
$\mathbb{F}_4$-VOLE - $(\mathbb{F}_3, \mathbb{F}_2)$-wPRF	AM	2	$38 + 1,173$	$7.1 + 0.4$
Garbling	AM	3	2^{15}	$0 + 4.0$
[Mea86]	DDH	2	512	121

Figure 6: We summarize comparison of our distributed wPRF protocols against other OPRF protocols. We consider the setting where multiple (adaptive) evaluations for a fixed key are performed. r denotes the round complexity and $+$ denotes that additional rounds and communication are required to set up the protocols. Communication reports the end-to-end amortized communication (including any preprocessing) per evaluation in bits.

The most efficient protocol is our OT-based $(\mathbb{F}_2, \mathbb{F}_3)$-wPRF protocol (Figure 4) with the input-OT optimization. It requires 7.0 microseconds per amortized evaluation and a total of two rounds, including the preprocessing. Excluding the succinct setup phase ($7.0\mu s$ and 38 bits), the amortized communication cost is $916 = \lambda + m + (m + t)\log_2(3)$ bits and $0.4\mu s$ for $(n, m, t) = (4\lambda, 2\lambda, \lambda/\log_2(3))$ and $\lambda = 128$, see Section 3.5. This protocol requires $\lambda + m$ OLEs/OTs per evaluation which form the dominant computational cost, $7.0\mu s$ compared to $0.4\mu s$ of online time. The prior protocol [DGH+21] based on alternating moduli requires an amortized total of $32.9\mu s$ and $1,317$ bits of communication, a $3.7\times$ and $1.4\times$ improvement. Due to lack of implementation, for their online time we use their reported *plaintext* running time of $6.1\mu s$ [DGH+21], however, our implementation technique would likely lower this closer to our $0.4\mu s$. Regardless, the bulk of the running time improvement comes from our protocol requiring fewer OTs. Moreover, our 128 input OTs are of 4 bit strings while [DGH+21] uses 256 OTs of 256 bit strings. When properly implemented, this results in an almost $256/4 = 64\times$ faster OT generation due to multiplying smaller vectors with the syndrome decoding matrix. The lowest communication OPRF protocol is based on DDH [Mea86]. It is extremely communication-efficient, requiring just two curve elements to be sent. However, DDH is insecure in the post-quantum

setting and, unlike ours, does not lend itself to secret-shared output. Moreover, it requires an order of magnitude more time which makes it less attractive when performing many evaluations. However, for a single evaluation DDH [Mea86] remains the best option. We also consider our $(\mathbb{F}_3, \mathbb{F}_2)$-wPRF but observe that the communication overhead is worse than our $(\mathbb{F}_2, \mathbb{F}_3)$-wPRF while it remains competitive in terms of computation when $\mathbb{F}_4$-VOLE is used. The communication overhead is due to requiring disproportionately larger parameters and the mod operation being less efficient. Finally, we consider the garbling-based protocol of Figure 5. This protocol has the interesting advantage that it only requires OT correlations in the setup phase where the key is set. In particular, one OT per bit of the key is performed. All subsequent evaluations of the protocol can be implemented with only calls to the random oracle. The advantage of this is highlighted when only a small number of evaluations are performed. In this regime the sublinear OT protocols have relatively high computational/communication overhead due to the hidden constants. The communication overhead of this protocol is $2^{15} = \lambda n + (m + t)\log_2(3)$.

OPRF with shared output. For this analysis we will continue to assume x is known to one party while k is known to the other. The parties will receive a secret sharing of $F(k, x)$. We report our findings in Figure 7. As above, we divide the running time of the protocol into the time to generate the OT correlations and the online time. As can be seen in Figure 7, the main cost is OT generation, requiring between 15 to 30 times more time than the online phase.

Scheme	Optimization	r	Comm. (bits)	Online (μs)	#OTs	OT time (μs)
LowMC [ARS+15]	–	14	10,584	7.5	5,292	151.4
	–	88	3,168	12.6	1,584	45.0
$(\mathbb{F}_2, \mathbb{F}_3)$-wPRF [DGH+21]	–	2	1,126	6.1	640	25.4
$(\mathbb{F}_2, \mathbb{F}_3)$-**wPRF**	Input OTs	**2**	**790**	**0.4**	**384***	**7.3**
	Key OTs	**2**	**1,174**	**0.4**	**256**	**7.0**
	Key OTs, garbling†	**1**	**32,768**	**4.1**	**0**	**0**
$(\mathbb{F}_3, \mathbb{F}_2)$-**wPRF**	Key OTs, $\mathbb{F}_2$-OLE	**2**	**1,151**	**0.4**	**512**	**14.6**
	Key OTs, $\mathbb{F}_4$-VOLE	**2**	**1,151**	**0.4**	**256**	**7.1**

Figure 7: Comparison of our distributed wPRF protocols with secret-shared output against LowMC [ARS+15] when performing $q = 2^{20}$ concurrent evaluations. r denotes the round complexity excluding a reusable setup. Times reported are amortized per evaluation, and † denotes estimated running time.

As with the OPRF performance numbers, we observe that our $(\mathbb{F}_2, \mathbb{F}_3)$-wPRF with the input-OTs optimization performs the best, requiring a total of $7.97\mu s$ and just $38 + 790$ bits of online communication, where the 38 bits of communication is for the OT generation. The next most efficient protocol is our $(\mathbb{F}_2, \mathbb{F}_3)$-wPRF but with the key-OT optimization which reduces the number of OTs required at the expense of more communication. However, despite requiring 50% fewer OTs, we observe a minimal decrease in running time. This is because the input-OT optimization makes use of a VOLE correlation for short strings, i.e., 4 bits, while OT/OLE requires generation of a VOLE correlation for 128 bit strings. When properly implemented, this essentially translates to a $128/4 = 32\times$ improvement in running time for the matrix multiplication step in the VOLE protocol, which is the main overhead. Thus, we generally suggest only using the input-OT optimization for the $(\mathbb{F}_2, \mathbb{F}_3)$-wPRF. However, a major drawback of the $(\mathbb{F}_2, \mathbb{F}_3)$-wPRF with secret-shared output is that the output shares and values are mod 3. This is highly non-standard and if shared output is desired, more postprocessing would be required. A more natural (and efficient) option is for the wPRF to naively output the desired share format, e.g., $(\mathbb{F}_3, \mathbb{F}_2)$-wPRF. We only implement this protocol with the key-OT optimization. The protocol requires $0.4\mu s$ and between 7 to $14\mu s$ of online and OT generation time, respectively. The protocol requires $1,151$ bits of online communication and 52 bits for OT generation. One could consider using the input-OT optimization to lower the communication overhead but this would require implementing $\mathbb{F}_3$ silent OT/VOLE which has higher computational cost, approximately $5\times$ by our estimates. Compared to [DGH+21], our protocols perform better in communication, running time, and the number of OTs used. When compared to LowMC [ARS+15], our online phase is between 7 and 13 times faster. When profiling both implementations, we observe that almost all of the time is spent in the matrix-vector multiplication routine. Given that [ARS+15] requires many more such multiplications, their

running time should be worse. Additionally, their protocol requires substantially more OTs/OLEs, a factor between 20 and 3 times depending on the parameters used.

5 Post-Quantum Signatures and Ring Signatures

We use the MPC-in-the-head framework [IKOS07] and instantiate it with the alternating-moduli OWF proposed in [DGH+21], which is in turn based on [BIP+18]. Instead of using a generic compiler such as [KKW18], we aim to compute the same circuit but use a bespoke MPC protocol, tailored to the AM-OWF, allowing us to shrink the size by 2-3×, when compared to [DGH+21].

Overview. Our MPC protocol is quite simple and proceeds as follows. The N parties start with an additive sharing of x and can locally compute $\mathbf{A}x$. They then engage in a share conversion procedure to convert shares of $v = \mathbf{A}x$ to shares of $v' \in \mathbb{F}_3^m$. We do this by using preprocessed randomness of the form $(\llbracket r \rrbracket^2, \llbracket r \rrbracket^3)$, where $r \in \mathbb{F}_2^m$. To convert the shares, parties mask shares of x with shares of r in $\mathbb{F}_2$, reconstruct $x+r$, and then compute $\llbracket v' \rrbracket_3 = v \odot \llbracket 1-r \rrbracket^3 + (1-v) \odot \llbracket r \rrbracket^3$. The parties finish the protocol by computing $\mathbf{B}v'$ and reconstructing the output. However, the main difficulty is in *efficiently* compiling this MPC protocol to a publicly verifiable proof of knowledge of a preimage of the AM-OWF. One can of course use generic techniques such as the KKW [KKW18] or ZKB++ [CDG+17] proof systems as done in [DGH+21], and although they produce competitive signature sizes, they are still larger than more recent post-quantum signatures based on symmetric-key assumptions [FJR22,CCJ23,AMGH+23,KZ22,BBdSG+23]. We use a technique similar to [CCJ23] to handle the preprocessing for *free* and design a bespoke MPCitH protocol for proving knowledge of the preimage of the AM-OWF. The high-level idea is as follows. We allow the prover to freely choose (potentially malicious) correlations $(\llbracket r \rrbracket^2, \llbracket r \rrbracket^3)$, but demand that the correlations are permuted using a uniformly random permutation π that is chosen by the verifier.[10] This allows use to completely avoid any checks on the preprocessed correlations, as we are able to bound the probability with which a prover can cheat for the very specific circuit we are interested in. The rest of the protocol proceeds as described above. The full zero-knowledge proof of knowledge protocol is specified in Figure 8. A signature scheme can then be constructed by applying the Fiat-Shamir heuristic, where the random coins provided by the verifier is replaced by a random oracle hash of the protocol transcript and the message being signed.

Our security proof proceeds in a manner very similar to that of [CCJ23]. We first note that Kales and Zaverucha [KZ20] showed that there was an attack on signature schemes using Fiat-Shamir heuristic on 5-round MPC-in-the-Head protocol. The core observation is that a malicious prover can cleverly resample verifier challenges in the second and fourth round such that the cost of finding a forgery is reduced to

$$\mathrm{cost}_{\mathrm{forge}} := \min_{\tau_1,\tau_2:\tau_1+\tau_2=\tau} \frac{1}{\sum_{i=\tau_1}^{\tau} \binom{\tau}{i} \mathsf{p}^i (1-\mathsf{p})^{\tau-i} + N_2^{\tau}} \tag{5.1}$$

where p is the combinatorial bound. We handle this by making the combinatorial bound very small, allowing us to use (approximately) the same number of repetitions as if this attack did not exist. We now define an analogous f-strongly invalid witness and combinatorial bound.

Definition 5.1 (f-strongly invalid witness)**.** We say that $x \in \mathbb{F}_2^n$ is an f-weakly valid witness if there exists $v' \in \mathbb{F}_3^m$ such that $\mathbf{A}x = v$, $\mathbf{B}v' = y$, and either $\Delta_H(v, v') \leq f$ or $\Delta_H(\mathbf{1} \oplus v, v') \leq f$. If x is not an f-weakly valid witness, then it is an f-strongly invalid witness.

Definition 5.2 (Combinatorial bound)**.** A *combinatorial bound* for the zero-knowledge protocol of Figure 8 with parameters (n, t, m) is a real $\mathsf{p} = \mathsf{p}(t, m, f) \in (0, 1)$ such that for *any* f-strongly invalid witness $x \in \mathbb{F}_2^n$, and for any pair of vectors $(r, r') \in \mathbb{F}_2^m \times \mathbb{F}_3^m$,

$$\Pr_{\pi \leftarrow \mathrm{Perm}_m}[\mathbf{B}v' = y \mid v = \mathbf{A}x, v' = \pi(r') + (x \oplus \pi(r)) \odot (1 - 2\pi(r'))] \leq \mathsf{p}(t, m, f),$$

where Perm_m denotes the set of all permutations of $\{1, \ldots, m\}$.

[10] This technique of using shuffled correlations was first observed in [CCJ23].

zkPoK of a Preimage of the AM-OWF

Inputs: The prover and the verifier have matrices $\mathbf{A} \in \mathbb{F}_2^{m \times k}$ and $\mathbf{B} \in \mathbb{F}_3^{K \times m}$ and a vector $y \in \mathbb{F}_3^K$. $\mathbf{B} = [\tilde{\mathbf{B}} | \mathbf{I}_K]$ is in systematic form and the prover knows a vector x such that $\mathbf{B}(\mathbf{A}x) = y$.

Round 1. The prover emulates the preprocessing phase and commits to the inputs:

1. Sample a random seed seed^*.
2. Using seed^* as the root of a depth-$\log N$ GGM-PRF [GGM84], produce leaves $\{(\mathsf{seed}_i, \sigma_i)\}_{i \in [N]}$.
3. For $i \in [N-1]$, expand seed_i to obtain the i-th pseudorandom share of $x \in \mathbb{F}_2^k$, $s \in \mathbb{F}_2^m$, and $t \in \mathbb{F}_3^m$. Set $\mathsf{state}_i = \mathsf{seed}_i$.
4. For the N-th party set:
   - $[\![s]\!]_N$ to be a pseudorandom share.
   - $[\![x]\!]_N = x \oplus_{i=1}^{N-1} [\![x]\!]_i$.
   - $[\![t]\!]_N = s - \sum_{i=1}^{N-1} [\![t]\!]_i \mod 3$.
   - $\mathsf{state}_i = \mathsf{seed}_N || [\![x]\!]_N || [\![t]\!]_N$.
5. Compute and send $h = H(\mathsf{com}_1, \ldots, \mathsf{com}_N)$, where $\mathsf{com}_i = \mathsf{Commit}(\mathsf{seed}_i; \sigma_i)$.

Round 2. The verifier samples a permutation π of m elements and sends it to the verifier.

Round 3. The prover now emulates the execution of the MPC protocol and commits to intermediate wire values.

1. Compute intermediate wire values:
   - $[\![v]\!] = \mathbf{A}[\![x]\!]$.
   - $[\![z]\!] = [\![v]\!] \oplus [\![\pi(s)]\!]$.
   - $z \leftarrow \oplus_{i=1}^N z_i$.
   - $[\![v']\!] = z + (1 - 2z) \odot [\![\pi(t)]\!] \mod 3$.
   - $[\![y]\!] = \mathbf{B}[\![v']\!] \mod 3$.
   - Set $\mathsf{msg}_i = ([\![z]\!]_i, [\![y]\!]_i)$.
2. Compute $h' = H(\mathsf{msg}_1, \ldots, \mathsf{msg}_N)$.
3. Send h' and $z^{(1)}$, which is the first $(m - K)$ entries of z, to the verifier.

Round 4. The verifier chooses a position $i^* \in [N]$ to not open and sends it to the prover.

Round 5. The prover sends $(\mathsf{state}_i, \sigma_i)_{i \neq i^*}$ and com_{i^*}.

Verification. Verifier accepts if all of the below checks pass.

1. For $i \in [N] \setminus i^*$, compute $\mathsf{com}_i = \mathsf{Commit}(\mathsf{state}_i, \sigma_i)$ and recover the i-th party's shares of x, s, and t, using state_i.
2. For all but the i^*-th share, compute
   - $[\![v]\!] = \mathbf{A}[\![x]\!]$.
   - $[\![z]\!] = [\![v]\!] \oplus [\![\pi(s)]\!]$.
   - $[\![v'^{(1)}]\!] = z^{(1)} + (1 - 2z^{(1)}) \odot [\![\pi(t)^{(1)}]\!] \mod 3$.
   - $[\![v'^{(2)}]\!] = y - \tilde{\mathbf{B}}[\![v'^{(1)}]\!] \mod 3$.
   - $z^{(2)} = [\![v'^{(2)}]\!] - [\![\pi(t)^{(2)}]\!]$.
   - $[\![y]\!] = \mathbf{B}[\![v']\!] \mod 3$.
3. Recompute $\mathsf{msg}_{i^*} = (z \oplus_{i \neq i^*} [\![z]\!]_i, y - \sum_{i \neq i^*} [\![y]\!]_i \mod 3)$.

Check if $h = H(\mathsf{com}_1, \ldots, \mathsf{com}_N)$ and $h' = H(\mathsf{msg}_1, \ldots, \mathsf{msg}_N)$.

Figure 8: A 5-round zero-knowledge proof of knowledge of a preimage of the AM-OWF.

We are now ready to state the main theorem of soundness for Figure 8, which we prove subsequently.

Theorem 5.3. *Let* Commit *be a non-interactive commitment scheme and H be a collision-resistant hash function. Let* p *be a combinatorial bound for the protocol in Figure 8. Then the protocol in Figure 8 is a* gap *honest-verifier zero-knowledge argument of knowledge for the relation $\mathcal{R}$ such that $((\mathbf{A}, \mathbf{B}, y), x) \in \mathcal{R}$ iff $\mathbf{B}(\mathbf{A}x) = y$. The* gap *relation $\mathcal{R}'$ is such that $((\mathbf{A}, \mathbf{B}, y), x) \in \mathcal{R}'$ if $\mathbf{B}(\mathbf{A}x) = y$ and x is an f-weakly valid witness. The soundness error of the proof is at most $\varepsilon = \mathsf{p} + 1/N - \mathsf{p}/N$.*

Proof. We need to show that the interactive zero-knowledge proof is complete, honest-verifier zero-knowledge, and that the soundness error is at most ε. Completeness is easy to see from the description of the protocol.

Honest-Verifier Zero-Knowledge. To prove that the protocol is honest-verifier zero-knowledge we show that the MPC protocol described in Figure 8 is secure against a semi-honest adversary corrupting up to $n - 1$ parties. The simulator works as follows:

- Sample a uniformly random position $i^* \leftarrow [n]$ and a uniformly random permutation $\pi \leftarrow \mathrm{Perm}_m$.
- Carry out Round 1 of Figure 8 honestly, using a uniformly random value for x.
- In Round 3, compute shares of z and y for all $i \neq i^*$ as described. To compute the message sent by the i^*-th party, first sample $z \leftarrow \mathbb{F}_2^m$, and compute $[\![z]\!]_{i^*} = z \oplus_{i \neq i^*} [\![z]\!]_i$ and $[\![y]\!]_{i^*} = y - \sum_{i \neq i^*} [\![y]\!]_i$.

To see that this is a good simulator, note that the distribution of the r, r' and their shares is identical to that of a real execution. Given any $N - 1$ shares of v and r, the distribution of z is uniformly random and hence the distribution of z is also identical to a real execution. Next, the shares of y are fully determined given z and the shares of r, r', and x. Indeed, the i^*-th party's share of y is not consistent with this but the commitment to the i^*-th party's state is never opened. Also note that msg_{i^*} is indeed consistent with the views that are opened. Finally, due to the hiding property of the commitment scheme, the simulation is computationally indistinguishable from a real execution.

Soundness. Let $\tilde{\mathsf{P}}$ be a malicious prover which manages to generate an accepting proof with probability $\tilde{\varepsilon} > \varepsilon$. Then there exists an extractor Ext, which when given black-box access to $\mathcal{A}$ with rewinding capabilities, can extract a witness x such that it is a weakly valid witness (Definition 5.1). Let R denote the randomness used by $\tilde{\mathsf{P}}$ to generate the first round commitment h and let r be a possible realization of R. Let Succ denote the even that $\tilde{\mathsf{P}}$ succeeds in convincing V. By the hypothesis, we have

$$\Pr[\mathsf{Succ}_{\tilde{\mathsf{P}}}] = \tilde{\varepsilon} > \varepsilon = \mathsf{p} + \frac{1}{N} - \frac{\mathsf{p}}{N}.$$

Fix an arbitrary value $\alpha \in \{0, 1\}$ such that $(1 - \alpha)\tilde{\varepsilon} > \varepsilon$, which exists since $\tilde{\varepsilon} > \varepsilon$. We say that a realization r of the prover randomness is good if

$$\Pr[\mathsf{Succ}_{\tilde{\mathsf{P}}} \mid R = r] \geq (1 - \alpha)\tilde{\varepsilon}.$$

Furthermore, by the splitting lemma [PS00], we have $\Pr[R \text{ is good} \mid \mathsf{Succ}_{\tilde{\mathsf{P}}}] \geq \alpha$. Let T_0 be the transcript of a successful execution of the zero-knowledge proof with $\tilde{\mathsf{P}}$, r denote the random coin used by $\tilde{\mathsf{P}}$ in the first round and d_0 denote the fourth-round message of the verifier. If r is *good*, then

$$\Pr[\mathsf{Succ}_{\tilde{\mathsf{P}}} \mid R = r] \geq (1 - \alpha)\tilde{\varepsilon} > \varepsilon > \frac{1}{N},$$

which implies that there necessarily exists a second successful transcript T_1, with a different fourth-round message $d_1 \neq d_0$. As we will demonstrate later, given (T_0, T_1), it is possible to extract a triplet (x, r, r') consistent with both transcripts, where x is a weakly valid witness, and (r, r') is the preprocessing material used by the prover. Let (π_0, d_0) and (π_1, d_1), with $d_0 \neq d_1$ denote the verifier challenges used in successful transcripts T_0 and T_1, respectively. Denote the fifth-round messages in these transcripts by $(\{\mathsf{state}_i, \sigma_i\}_{i \neq d_0}, \mathsf{com}_{d_0})$ and $(\{\mathsf{state}'_i, \sigma'_i\}_{i \neq d_1}, \mathsf{com}_{d_1})$. Suppose $\exists\, i \in [N] \setminus \{d_0, d_1\}$ such that $(\mathsf{state}_i, \sigma_i) \neq (\mathsf{state}'_i, \sigma'_i)$. Then one of the following must be true:

- The committed values are different:

$$\mathsf{com}_i = \mathsf{Commit}(\mathsf{state}_i, \sigma_i) \neq \mathsf{Commit}(\mathsf{state}'_i, \sigma'_i) = \mathsf{com}'_i,$$

  but since both of these transcripts verify, this implies that the prover has found a collision for H.

- The commitments are identical:

$$\mathsf{com}_i = \mathsf{Commit}(\mathsf{state}_i, \sigma_i) = \mathsf{Commit}(\mathsf{state}'_i, \sigma'_i) = \mathsf{com}'_i,$$

  but this violates the binding property of the commitment scheme.

Therefore, it must be the case that $\{\mathsf{state}_i, \sigma_i\}_{i \neq d_0, d_1} = \{\mathsf{state}'_i, \sigma'_i\}_{i \neq d_0, d_1}$. Furthermore, since $d_0 \neq d_1$, they jointly define a unique tuple $(\mathsf{state}_i, \sigma_i)_{i \in [N]}$, from which we can extract the witness x and the preprocessing material (r, r') used by the prover.

Finally, we show that if x is a strongly invalid witness, then $\Pr[\mathsf{Succ}_{\tilde{\mathsf{P}}} \mid R = r] \leq \varepsilon$, contradicting our assumption that r is good. Let us denote $\mathsf{BadPerm}$ the event (defined over a random choice of permutation π, and for the fixed value of (x, r, r')) that $\mathbf{A}x = v$ and $\mathbf{B}v' = y$, where $v' = \pi(r') + (x \oplus \pi(r)) \odot (1 - 2\pi(r'))$. By definition of the combinatorial bound (Definition 5.2), we have $\Pr[\mathsf{BadPerm}] \leq \mathsf{p}$. We can rewrite our desired inequality as

$$\begin{aligned}\Pr[\mathsf{Succ}_{\tilde{P}} \mid R = r] &= \Pr[\mathsf{Succ}_{\tilde{P}} \wedge \mathsf{BadPerm} \mid R = r] + \Pr[\mathsf{Succ}_{\tilde{P}} \wedge \neg\mathsf{BadPerm} \mid R = r] \\ &\leq \mathsf{p} + (1 - \mathsf{p}) \cdot \Pr[\mathsf{Succ}_{\tilde{P}} \mid R = r \wedge \neg\mathsf{BadPerm}].\end{aligned}$$

If we can show that $\Pr[\mathsf{Succ}_{\tilde{P}} \mid R = r \wedge \neg\mathsf{BadPerm}] \leq 1/N$, then we are done. For the sake of contradiction, assume that $\Pr[\mathsf{Succ}_{\tilde{P}} \mid R = r \wedge \neg\mathsf{BadPerm}] > 1/N$. Since $\Pr[\mathsf{Succ}_{\tilde{P}} \mid R = r] \geq \varepsilon$, using the same argument as earlier, given a successful transcript $\tilde{T}_0$ with fourth-round message $\tilde{d}_0$, there must exist a second successful transcript $\tilde{T}_1$ with identical first three rounds but fourth-round message $\tilde{d}_1 \neq \tilde{d}_0$. Moreover, $\tilde{T}_0$ and $\tilde{T}_1$ must be consistent and uniquely define a tuple $(\tilde{\mathsf{state}}_i, \tilde{\sigma}_i)_{i \in [N]}$. Since we condition on the same randomness $R = r$, the $\tilde{h}$ in the second round corresponding to transcripts $(\tilde{T}_0, \tilde{T}_1)$ and (T_0, T_1) must be identical and therefore, by the collision resistance of H, $(\tilde{\mathsf{state}}_i, \tilde{\sigma}_i)_{i \in [N]} = (\mathsf{state}_i, \sigma_i)_{i \in [N]}$. Now, using $\tilde{T}_0$, we can reconstruct the messages sent by all parties using a strategy similar to verifying the transcript to obtain $\{\tilde{\mathsf{msg}}_i^0\}_{i \in [N]}$. Similarly, using $\tilde{T}_1$, we obtain $\{\tilde{\mathsf{msg}}_i^1\}_{i \in [N]}$. Because $\tilde{T}_0$ and $\tilde{T}_1$ share the same first three rounds, using a similar argument as above $\{\tilde{\mathsf{msg}}_i^0\}_{i \in [N] \setminus \{\tilde{d}_0, \tilde{d}_1\}} = \{\tilde{\mathsf{msg}}_i^0\}_{i \in [N] \setminus \{\tilde{d}_0, \tilde{d}_1\}}$. Note that when we recovered $\tilde{\mathsf{msg}}_{\tilde{d}_0}^0$, we set $[\![y]\!]_{\tilde{d}_0} = y - \sum_{i \neq \tilde{d}_0} [\![y]\!]_i$. Because we started with the assumption that x is a strongly invalid witness and conditioned on not using a bad permutation, it must be true that $\tilde{\mathsf{msg}}_{\tilde{d}_0}^0 \neq \tilde{\mathsf{msg}}_{\tilde{d}_0}^1$ as otherwise we have actually found a valid witness. But this also means that we have found two different inputs $\{\tilde{\mathsf{msg}}_i^0\}_{i \in [N]}$ and $\{\tilde{\mathsf{msg}}_i^1\}_{i \in [N]}$ that hash to the same value, contradicting the collision resistance of H. Thus $\Pr[\mathsf{Succ}_{\tilde{P}} \mid R = r \wedge \neg\mathsf{BadPerm}] \leq 1/N$ and hence,

$$\Pr[\mathsf{Succ}_{\tilde{\mathsf{P}}}] \leq \mathsf{p} + (1 - \mathsf{p})\frac{1}{N},$$

when x is a strongly invalid witness. The remaining proof and description of the extractor is identical to that of [FJR22, CCJ23] and we omit it here. □

Proof size. The prover sends the following to the verifier in the proof of knowledge of the preimage of the AM-OWF.

- Commitments h and h' of size 2λ bits each
- The co-path of i^* of size $\lambda \log N$ bits
- com_{i^*} of size 2λ bits
- The N-th shares of x and t of size $(n + m \log 3)$ bits
- z of size $m - t$ bits

where the h, h' commitments across all τ parallel executions can be combined into one commitment of size 2λ each.

$$\text{SIZE} = \underbrace{4\lambda}_{h,h'} + \tau\Big(\underbrace{\lambda(\log N)}_{\text{PPRF key}} + \underbrace{2\lambda}_{\mathsf{com}_{i^*}} + \underbrace{n}_{x_N} + \underbrace{m-t}_{\text{Compressed } z} + \underbrace{m\log 3}_{t_N}\Big).$$

In Figure 9, we report sizes of our signature scheme for the parameter sets listed below. We expect performance similar to [CCJ23] during signing/verification. We also note that because our MPC-in-the-Head is based on additive secret sharing, the hypercube technique from [AMGH+23] can be used to scale up to a larger number of parties while remaining computationally efficient.

Scheme	Size (KB)	Assumption
[HBD+22] (Fast)	16.68	SHA-256
[HBD+22] (Short)	7.67	
[CCJ23] (Fast)	12.5/11.3	RSD/f-almost-RSD over $\mathbb{F}_2$
[CCJ23] (Medium-1)	9.7/8.8	
[CCJ23] (Medium-2)	9.1/8.3	
[CCJ23] (Short)	8.6/7.8	
[AMGH+23] (Fast)	14.4/9.7	SD over $\mathbb{F}_2/\mathbb{F}_{256}$
[AMGH+23] (Short)	9.7/6.9	
[AMGH+23] (Shorter)	7.5/5.5	
[AMGH+23] (Shortest)	6.0/4.5	
[BBdSG+23] (Fast)	6.2/5.6	AES/EM-AES
[BBdSG+23] (Short)	4.9/4.5	
[KZ22] (Fast)	5.8	Rain_4
[KZ22] (Short)	4.4	
[ARZV+23] (Fast)	7.7	MinRank
[ARZV+23] (Short)	5.5	
[ARZV+23] (Shorter)	4.9	
[ARZV+23] (Shortest)	4.4	
[KHS+23] (Fast)	5.8	AIM
[KHS+23] (Short)	4.8	
[KHS+23] (Shorter)	4.1	
[KHS+23] (Shortest)	3.8	
This work (Fast)	**5.5**	**AM-OWF**
This work (Balanced)	**4.6**	
This work (Shortest)	**4.0**	

Figure 9: Comparison of our work against state-of-the-art signature schemes based on symmetric-key assumptions. For the work of [ARZV+23], we report the maximum signature sizes of the hypercube variant. EM-AES refers to AES in Even-Mansour mode.

Parameters. In [DGH+21], the authors propose $(n, m, t) = (\lambda, 3.53\lambda, \lambda/\log 3)$. However, for the relaxed notion of soundness that our zero-knowledge proof achieves, we actually require that it must be hard to find an f-weakly valid witness for the AM-OWF (Definition 5.1). The parameter f is set such that the combinatorial bound is made as low as possible in order to minimize the impact of the attack from [KZ20] on the signature size. As well will show later, this in turn determines the value of n, m, and t. During our search for parameters, we also identified an important metric that must be paid attention to when introducing any form of additional structure in $\mathbf{B}$.

We target 128-bit security with the following parameters, a) **Fast**: $(18, 193)$, b) **Balanced**: $(13, 1723)$, and c) **Short**: $(9, 65536)$ reported as (τ, N), where N is the number of parties and τ is the number of parallel repetitions.

Using Equation 5.1, we set $f = 11$, and now determine n, m, t such that it is still hard to find an f-weakly valid witness. To ensure that the covering lemma (Lemma 3.6) still holds we fix $n = t \log 3 + 10$.

Choosing t and m. We choose t and m such that if we sample a uniformly random value, it is not an f-weakly valid witness with overwhelming probability. The reason we use this strategy is because there does not seem to be a good way to choose inputs such that the output of $\mathbf{A}x$ lies close to the codewords of $\mathbf{B}$. Indeed if there existed a strategy that did better than simply trying at random, then we can extend the approxOWF reduction (Lemma 3.9) to the setting where $\Delta_H(v, v') \leq f$ and break the AM-OWF for the parameters proposed in [DGH+21]. Thus, when introducing any additional structure in $\mathbf{B}$, it must be the case that the codewords must be sufficiently *spread* out at least on average.

Coming back to the choice of t and m, a random linear code approaches the GV bound. Therefore, the number of codewords of $\mathbf{B}$ is approximately $3^m/q^{H_q(d/m)m}$, where d is the minimum distance of $\mathbf{B}$. The fraction of volume occupied by all words within hamming distance f of codewords of $\mathbf{B}$ is therefore given by $q^{H_q(f/m)m - H_q(d/m)m}$, which we set to be smaller than 2^{-128} as this is also the probability with which a random word is within hamming distance f of a codeword. This gives us $t = 135$, $m = 450$, and $n = 224$.

5.1 Post-Quantum Ring Signatures

An efficient proof of knowledge for the AM-OWF also serves as a building block for ring signatures [RST01] and Ring CT [Noe15]. The work of [GGHAK22] introduced generic compilers to lift MPCitH proof for a single NP statement to a disjunction of multiple NP statements, where the additional cost on top of a single MPCitH proof only grows logarithmically in the number of statements. Although the compiler is asymptotically very good, we observe that there is an initial startup cost resulting in a larger than necessary overhead when there are a small number of statements. In this section, we provide concretely more efficient protocols in the *few statements* regime.

Overview. The goal of a ring signature is to convince a verifier that a message was signed by one party out of a set of ℓ parties. Recall that in the case of the AM-OWF-based signature scheme, the public keys lie in $\mathbb{F}_3^t$. A signer needs to produce a proof that they know x_i such that $y_i = \mathbf{B}(\mathbf{A}x_i)$ lies in some set of public keys $(y_1, y_2, \ldots, y_\ell)$. Our strategy is to extend the MPC protocol at the beginning of Section 5 as follows. Recall that at the end of the MPC protocol computing the AM-OWF on input x_i, all parties hold shares of y_i. All that is left to do is to prove that parties hold a secret sharing of some $y_i \in (y_1, \ldots, y_\ell)$. First interpret the public keys $(y_1, y_2, \ldots, y_\ell)$ as elements in the field $\mathrm{GF}(3^t)$ and interpolate a degree-$(\ell - 1)$ polynomial $Y(X)$ such that $Y(i) = y_i$. $Y(X)$ satisfies the following property for all $i \in [\ell]$,

$$Y(X) = (X - i)Q(X) + y_i, \tag{5.2}$$

where $Q(X)$ is a polynomial of degree at most $(\ell - 2)$ and i is interpreted as an element of $\mathrm{GF}(3^t)$ in a natural sense of ternary decomposition. Observe that $Y(X)$ can be computed by the verifier, $Q(X)$ can be computed by the prover and secret shared with the parties in the first round, and the parties hold shares of $Y(i) = y_i$ by the end of round 3.[11] In round 4 of the MPC protocol in Figure 8, we have the verifier send a random point $r \leftarrow \mathrm{GF}(3^t)$ and have the parties evaluate Equation 5.2 at this point. Note that the verifier can actually compute all powers of r, and thus $Y(r)$ can be computed by the verifier. The parties can compute shares of $Q(r)$ and $(r - i)$ through local operations on their shares in *verifiable* manner. Finally, given access to one Beaver triple, the parties can multiply $Q(r)$ and $(r - i)$ to obtain shares of the RHS of Equation 5.2, which can be revealed to ensure that the reconstructed value matches $Y(r)$. We note that the probability with which the verifier samples a bad r such that $Y(X) - y_i \neq Q(X)(X - i)$ but $Y(r) - y_i = Q(r)(r - i)$, is $\leq \ell/|\mathbb{F}_3^t|$ as a polynomial of degree $\ell - 1$ has at most $\ell - 1$ roots.

An astute reader might have observed that we are not done yet, because the prover could have used *any* $i \in \mathrm{GF}(3^t)$ not restricted to $i \in [\ell]$. In fact, a prover could use a public key y^* by choosing i^* to be a root of the polynomial $Y(X) - y^*$. An naive solution is to demand a range proof on i via the ternary analogue of bit decomposition where the prover shows that i can be represented using $\log_3 \ell$ many $\mathbb{F}_3$ elements $(b_0, b_1, \ldots, b_{\log_3 \ell})$. The parties would be provided with secret shares of b_j in $\mathrm{GF}(3^t)$ and they check that a) $i = \sum_{j=1}^{\log_3 \ell} b_j \cdot 3^j$ and b) $b_j \in \{0, 1, 2\}$, which

[11] Because we no longer have the public key in the clear, we cannot exploit the systematic form for compression.

requires two additional multiplications to verify that $b_j(b_j-1)(b_j-2)=0$. Instead, we propose a way to carry out this range check *for free*, as long as ℓ is a power of 3.[12]

Our main observation is that a verifier already knows that $i \in [\ell]$, therefore instead of secret sharing $i \in \mathrm{GF}(3^t)$, we can share it in $\mathrm{GF}(\ell)$ without any loss of privacy. Later, when evaluating Equation 5.2, the parties then locally embed these shares in $\mathrm{GF}(3^t)$ by simply appending $(t - \log_3 \ell)$ zeros. Thereby, adding just $\tau \log \ell$ bits to the final signature size. Although the above construction is simple, it performs surprisingly well (Figure 10) and highlights the advantage of using a flexible framework such as MPCitH. Note that we were forced to use powers of 3 due to the structure of the public keys in the AM-OWF. Instead one could use a OWF with binary output such as [KZ22], which has slightly larger signature sizes, but will still have competitive signature sizes in addition to being able to support powers of 2.

Ring Size:	$\mathbf{2^3}$	$\mathbf{2^6}$	$\mathbf{2^{12}}$	**Assumption**
[LLNW16]	52 MB	94 MB	179 MB	SIS
[TKS+19]	$>$ 124 KB	$>$ 900 KB	61 MB	Ring-SIS
[ESLL19]	41 KB	58 KB	256 KB	M-LWE and M-SIS
[EZS+19]	29 KB	34 KB	148 KB	M-LWE and M-SIS
[GGHAK22]	46 KB	50 KB	59 KB	LowMC
[LNS21]	$<$ 16 KB	$<$ 18 KB	$<$ 19 KB	Ex-M-LWE and M-SIS
Ring Size:	$\mathbf{11}$	$\mathbf{2^5}$	$\mathbf{2^{10}}$	**Assumption**
[ESZ22]	9 KB	11 KB	18 KB	M-LWE and M-SIS
Ring Size:	**9**	**27**	**729**	**Assumption**
This work	**6.2 KB**	**8.9 KB**	**113.0 KB**	**AM-OWF**

Figure 10: Comparison of ring signature sizes for different ring sizes. The sizes of lattice-based ring signatures except for [ESZ22] were obtained from [ESLL19]. The work of [ESZ22] provides benchmarks for different ring sizes, we therefore compare against this work separately. For our work, we use the short parameters (Section 5) and benchmark over the closest powers of 3.

5.2 Size of Ring Signature

The above technique in fact works generically for disjunctions of the same relation in any MPCitH-based proof system, by first interpreting the statement being proved as a field element $\mathrm{GF}(p^t)$ for some prime p. Below, we quantify the *overhead* introduced on top of a single proof.

In the MPCitH proof, the prover now additionally needs to generate Beaver triples as part of the preprocessing material. This can be done using the *sacrifice*-based technique from [DPSZ12, DKL+13, LN17]. The communication can further be optimized by using PRG seeds to generate the shares of Beaver triples. Let $(\llbracket a \rrbracket, \llbracket b \rrbracket, \llbracket c \rrbracket)$ be the Beaver triple such that $a.b = c$, then $\llbracket a \rrbracket, \llbracket b \rrbracket$ of all N parties can be set to be the output of the PRG as the correlation should indeed be uniformly random. Similarly, $\llbracket c \rrbracket$ can also be set to the output of the PRG, but with the share of N-th party $\llbracket c \rrbracket_N = c - \sum_{i=1}^{N-1} \llbracket c \rrbracket$, implying that the additional communication in the MPCitH per beaver triple is just one field element. In our protocol, we only need one Beaver triple per iteration of the MPC proof. Using a sacrifice check for each iteration of the MPCitH requires 2τ Beaver triples in total and therefore, the communication per iteration in the preprocessing phase is two field elements during the check and two additional field elements as the view of parties in the MPC. During the online phase, parties hold shares of the coefficients of the $\ell - 1$ coefficients of $Q(X)$ from Equation 5.2, each of which adds one field element to the communication per iteration of the MPCitH proof. Finally, when parties multiply $Q(r)$ and $(r - i)$, they broadcast two field elements. Thus, the communication overhead of

[12] The ring size can (albeit wastefully) be padded to the next largest power by inserting random keys sampled as the output of a random oracle.

the *online* phase of MPCitH is $(\ell + 1)$ field elements per iteration. The total proof size grows linearly in ℓ and the *incremental* cost of adding a statement to the disjunction is $\approx \tau t \log 3$ bits.

$$\text{SIZE} = |\Pi_{\text{MPCitH}}| + \tau \log 3 (t(\underbrace{\ell - 1}_{Q(X)} + \underbrace{6}_{\text{Multiply}}) + \underbrace{\log_3 \ell}_{i}).$$

Compiling with [GGHAK22]. For rings of small to medium size, our protocol outperforms state-of-the-art ring signatures but for larger rings the sizes can get very big (Figure 10). However, [GGHAK22] devised a concretely efficient compiler to prove set-membership in various MPCitH proofs which uses Merkle trees in a black-box manner to reduce computational and communication overhead. Their sizes are roughly $42 + 1.5 \times \log \ell$ KB for a ring of size ℓ, but the 42 KB is the size of a single signature using LowMC, with the KKW proof system. Since then, there has been tremendous progress in building signatures using MPCitH, bringing sizes down to ≈ 5 KB for a single signature. We expect that using their compiler with any of the KKW-style proof systems from Figure 9 would yield ring signatures that only grow logarithmically in the ring size and are competitive with state-of-the-art lattice-based ring signatures.

Acknowledgements

We thank Omar Alrabiah for helpful discussions on error correcting codes. We thank Zijie Lu for his contributions to the subfield VOLE protocol of libOTe [RR] which allowed us to benchmark our $\mathbb{F}_4$ modulus conversion protocol. We also thank the anonymous reviewers for their feedback.

References

[ABG+14] Adi Akavia, Andrej Bogdanov, Siyao Guo, Akshay Kamath, and Alon Rosen. Candidate weak pseudorandom functions in $\mathsf{AC}^0 \circ \mathsf{MOD}_2$. In Moni Naor, editor, *ITCS 2014*, pages 251–260. ACM, January 2014. (Cited on page 3, 10)

[ADDG24] Martin R. Albrecht, Alex Davidson, Amit Deo, and Daniel Gardham. Crypto dark matter on the torus - oblivious prfs from shallow prfs and TFHE. In Marc Joye and Gregor Leander, editors, *EUROCRYPT 2024 , Part VI*, volume 14656 of *LNCS*, pages 447–476. Springer, 2024. (Cited on page 28)

[ADDS21] Martin R. Albrecht, Alex Davidson, Amit Deo, and Nigel P. Smart. Round-optimal verifiable oblivious pseudorandom functions from ideal lattices. In Juan A. Garay, editor, *PKC 2021, Part II*, volume 12711 of *LNCS*, pages 261–289. Springer, 2021. (Cited on page 28)

[AES01] Advanced Encryption Standard (AES). National Institute of Standards and Technology, NIST FIPS PUB 197, U.S. Department of Commerce, November 2001. (Cited on page 7)

[AFL+16] Toshinori Araki, Jun Furukawa, Yehuda Lindell, Ariel Nof, and Kazuma Ohara. High-throughput semi-honest secure three-party computation with an honest majority. In Edgar R. Weippl, Stefan Katzenbeisser, Christopher Kruegel, Andrew C. Myers, and Shai Halevi, editors, *ACM CCS 2016*, pages 805–817. ACM Press, October 2016. (Cited on page 10)

[AGP+19] Martin R. Albrecht, Lorenzo Grassi, Léo Perrin, Sebastian Ramacher, Christian Rechberger, Dragos Rotaru, Arnab Roy, and Markus Schofnegger. Feistel structures for MPC, and more. In Kazue Sako, Steve Schneider, and Peter Y. A. Ryan, editors, *ESORICS 2019, Part II*, volume 11736 of *LNCS*, pages 151–171, September 2019. (Cited on page 3)

[AMGH+23] Carlos Aguilar-Melchor, Nicolas Gama, James Howe, Andreas Hülsing, David Joseph, and Dongze Yue. The return of the sdith. In Carmit Hazay and Martijn Stam, editors, *EUROCRYPT 2023, Part V*, volume 14008 of *LNCS*, pages 564–596. Springer, 2023. (Cited on page 30, 34)

[ARS+15] Martin R. Albrecht, Christian Rechberger, Thomas Schneider, Tyge Tiessen, and Michael Zohner. Ciphers for MPC and FHE. In Elisabeth Oswald and Marc Fischlin, editors, *EUROCRYPT 2015, Part I*, volume 9056 of *LNCS*, pages 430–454, April 2015. (Cited on page 3, 6, 7, 9, 27, 29)

[ARZV+23] Gora Adj, Luis Rivera-Zamarripa, Javier Verbel, Emanuele Bellini, Stefano Barbero, Andre Esser, Carlo Sanna, and Floyd Zweydinger. Mirithh. Technical report, National Institute of Standards and Technology, 2023. https://csrc.nist.gov/csrc/media/Projects/pqc-dig-sig/documents/round-1/spec-files/MiRitH_spec-web.pdf. (Cited on page 34)

[Bas23] Andrea Basso. A post-quantum round-optimal oblivious PRF from isogenies. In Claude Carlet, Kalikinkar Mandal, and Vincent Rijmen, editors, *SAC 2023*, volume 14201 of *LNCS*, pages 147–168. Springer, 2023. (Cited on page 28)

[BBdSG+23] Carsten Baum, Lennart Braun, Cyprien Delpech de Saint Guilhem, Michael Klooß, Emmanuela Orsini, Lawrence Roy, and Peter Scholl. Publicly verifiable zero-knowledge and post-quantum signatures from vole-in-the-head. In Helena Handschuh and Anna Lysyanskaya, editors, *CRYPTO 2023, Part V*, volume 14085 of *LNCS*, pages 581–615. Springer, 2023. (Cited on page 30, 34)

[BBSS20] Xavier Bonnetain, Rémi Bricout, André Schrottenloher, and Yixin Shen. Improved classical and quantum algorithms for subset-sum. In *ASIACRYPT 2020, Part II*, LNCS, pages 633–666, December 2020. (Cited on page 15, 16)

[BCG+19a] 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 Lorenzo Cavallaro, Johannes Kinder, XiaoFeng Wang, and Jonathan Katz, editors, *ACM CCS 2019*, pages 291–308. ACM Press, November 2019. (Cited on page 10, 20, 21, 25)

[BCG+19b] Elette Boyle, Geoffroy Couteau, Niv Gilboa, Yuval Ishai, Lisa Kohl, and Peter Scholl. Efficient pseudorandom correlation generators: Silent OT extension and more. In Alexandra Boldyreva and Daniele Micciancio, editors, *CRYPTO 2019, Part III*, volume 11694 of *LNCS*, pages 489–518, August 2019. (Cited on page 6, 20, 25)

[BCG+23] 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, *EUROCRYPT 2023, Part I*, volume 14004 of *LNCS*, pages 271–302. Springer, 2023. (Cited on page 25)

[BCJ11] Anja Becker, Jean-Sébastien Coron, and Antoine Joux. Improved generic algorithms for hard knapsacks. In Kenneth G. Paterson, editor, *EUROCRYPT 2011*, volume 6632 of *LNCS*, pages 364–385, May 2011. (Cited on page 15)

[BDG+22] Saikrishna Badrinarayanan, Sourav Das, Gayathri Garimella, Srinivasan Raghuraman, and Peter Rindal. Secret-shared joins with multiplicity from aggregation trees. In Heng Yin, Angelos Stavrou, Cas Cremers, and Elaine Shi, editors, *ACM CCS 2022*, pages 209–222. ACM Press, 2022. (Cited on page 6)

[BIP+18] Dan Boneh, Yuval Ishai, Alain Passelègue, Amit Sahai, and David J. Wu. Exploring crypto dark matter: New simple PRF candidates and their applications. In Amos Beimel and Stefan Dziembowski, editors, *TCC 2018, Part II*, volume 11240 of *LNCS*, pages 699–729, November 2018. (Cited on page 3, 4, 5, 6, 10, 11, 12, 13, 14, 16, 28, 30)

[BKW20] Dan Boneh, Dmitry Kogan, and Katharine Woo. Oblivious pseudorandom functions from isogenies. In *ASIACRYPT 2020, Part II*, LNCS, pages 520–550, December 2020. (Cited on page 28)

[BM82] Manuel Blum and Silvio Micali. How to generate cryptographically strong sequences of pseudo random bits. In *23rd FOCS*, pages 112–117. IEEE Computer Society Press, November 1982. (Cited on page 3)

[BMR16] 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. (Cited on page 26)

[BPR12] Abhishek Banerjee, Chris Peikert, and Alon Rosen. Pseudorandom functions and lattices. In David Pointcheval and Thomas Johansson, editors, *EUROCRYPT 2012*, volume 7237 of *LNCS*, pages 719–737, April 2012. (Cited on page 16)

[BR17] Andrej Bogdanov and Alon Rosen. *Pseudorandom Functions: Three Decades Later*, pages 79–158. Springer International Publishing, 2017. (Cited on page 10)

[BS91] Eli Biham and Adi Shamir. Differential cryptanalysis of DES-like cryptosystems. *Journal of Cryptology*, 4(1):3–72, January 1991. (Cited on page 3)

[CCJ23] Eliana Carozza, Geoffroy Couteau, and Antoine Joux. Short signatures from regular syndrome decoding in the head. In Carmit Hazay and Martijn Stam, editors, *EUROCRYPT 2023, Part V*, volume 14008 of *LNCS*, pages 532–563. Springer, 2023. (Cited on page 6, 7, 9, 18, 20, 30, 33, 34)

[CCKK21] Jung Hee Cheon, Wonhee Cho, Jeong Han Kim, and Jiseung Kim. Adventures in crypto dark matter: Attacks and fixes for weak pseudorandom functions. In Juan A. Garay, editor, *PKC 2021, Part II*, volume 12711 of *LNCS*, pages 739–760. Springer, 2021. (Cited on page 4)

[CDG+17] Melissa Chase, David Derler, Steven Goldfeder, Claudio Orlandi, Sebastian Ramacher, Christian Rechberger, Daniel Slamanig, and Greg Zaverucha. Post-quantum zero-knowledge and signatures from symmetric-key primitives. In Bhavani M. Thuraisingham, David Evans, Tal Malkin, and Dongyan Xu, editors, *ACM CCS 2017*, pages 1825–1842. ACM Press, October / November 2017. (Cited on page 7, 30)

[Cha82] David Chaum. Blind signatures for untraceable payments. In David Chaum, Ronald L. Rivest, and Alan T. Sherman, editors, *CRYPTO'82*, pages 199–203. Plenum Press, New York, USA, 1982. (Cited on page 3)

[CKY09] Jan Camenisch, Aggelos Kiayias, and Moti Yung. On the portability of generalized Schnorr proofs. In Antoine Joux, editor, *EUROCRYPT 2009*, volume 5479 of *LNCS*, pages 425–442, April 2009. (Cited on page 9)

[DEG+18] Christoph Dobraunig, Maria Eichlseder, Lorenzo Grassi, Virginie Lallemand, Gregor Leander, Eik List, Florian Mendel, and Christian Rechberger. Rasta: A cipher with low ANDdepth and few ANDs per bit. In Hovav Shacham and Alexandra Boldyreva, editors, *CRYPTO 2018, Part I*, volume 10991 of *LNCS*, pages 662–692, August 2018. (Cited on page 3)

[DGGK21] Christoph Dobraunig, Lorenzo Grassi, Anna Guinet, and Daniël Kuijsters. Ciminion: Symmetric encryption based on toffoli-gates over large finite fields. In Anne Canteaut and François-Xavier Standaert, editors, *EUROCRYPT 2021, Part II*, volume 12697 of *LNCS*, pages 3–34. Springer, 2021. (Cited on page 3)

[DGH+21] Itai Dinur, Steven Goldfeder, Tzipora Halevi, Yuval Ishai, Mahimna Kelkar, Vivek Sharma, and Greg Zaverucha. Mpc-friendly symmetric cryptography from alternating moduli: Candidates, protocols, and applications. In Tal Malkin and Chris Peikert, editors, *CRYPTO 2021, Part IV*, volume 12828 of *LNCS*, pages 517–547. Springer, 2021. (Cited on page 3, 4, 5, 6, 10, 11, 12, 13, 14, 15, 16, 27, 28, 29, 30, 34, 35)

[DKL+13] Ivan Damgård, Marcel Keller, Enrique Larraia, Valerio Pastro, Peter Scholl, and Nigel P. Smart. Practical covertly secure MPC for dishonest majority - or: Breaking the SPDZ limits. In Jason Crampton, Sushil Jajodia, and Keith Mayes, editors, *ESORICS 2013*, volume 8134 of *LNCS*, pages 1–18, September 2013. (Cited on page 36)

[DKR+22] Christoph Dobraunig, Daniel Kales, Christian Rechberger, Markus Schofnegger, and Greg Zaverucha. Shorter signatures based on tailor-made minimalist symmetric-key crypto. In Heng Yin, Angelos Stavrou, Cas Cremers, and Elaine Shi, editors, *ACM CCS 2022*, pages 843–857. ACM Press, 2022. (Cited on page 7)

[DPSZ12] Ivan Damgård, Valerio Pastro, Nigel P. Smart, and Sarah Zakarias. Multiparty computation from somewhat homomorphic encryption. In Reihaneh Safavi-Naini and Ran Canetti, editors, *CRYPTO 2012*, volume 7417 of *LNCS*, pages 643–662. Springer, 2012. (Cited on page 36)

[ESLL19] Muhammed F. Esgin, Ron Steinfeld, Joseph K. Liu, and Dongxi Liu. Lattice-based zero-knowledge proofs: New techniques for shorter and faster constructions and applications. In Alexandra Boldyreva and Daniele Micciancio, editors, *CRYPTO 2019, Part I*, volume 11692 of *LNCS*, pages 115–146, August 2019. (Cited on page 36)

[ESZ22] Muhammed F. Esgin, Ron Steinfeld, and Raymond K. Zhao. Matrict$^+$: More efficient post-quantum private blockchain payments. In *IEEE Symposium on Security and Privacy 2022*, pages 1281–1298. IEEE Computer Society, 2022. (Cited on page 7, 36)

[EZS+19] Muhammed F. Esgin, Raymond K. Zhao, Ron Steinfeld, Joseph K. Liu, and Dongxi Liu. MatRiCT: Efficient, scalable and post-quantum blockchain confidential transactions protocol. In Lorenzo Cavallaro, Johannes Kinder, XiaoFeng Wang, and Jonathan Katz, editors, *ACM CCS 2019*, pages 567–584. ACM Press, November 2019. (Cited on page 36)

[FIPR05] Michael J. Freedman, Yuval Ishai, Benny Pinkas, and Omer Reingold. Keyword search and oblivious pseudorandom functions. In Joe Kilian, editor, *TCC 2005*, volume 3378 of *LNCS*, pages 303–324, February 2005. (Cited on page 3)

[FJR22] Thibauld Feneuil, Antoine Joux, and Matthieu Rivain. Syndrome decoding in the head: Shorter signatures from zero-knowledge proofs. In Yevgeniy Dodis and Thomas Shrimpton, editors, *CRYPTO 2022, Part II*, volume 13508 of *LNCS*, pages 541–572. Springer, 2022. (Cited on page 30, 33)

[GGHAK22] Aarushi Goel, Matthew Green, Mathias Hall-Andersen, and Gabriel Kaptchuk. Efficient set membership proofs using MPC-in-the-head. *PoPETs*, 2022(2):304–324, April 2022. (Cited on page 5, 7, 35, 36, 37)

[GGM84] Oded Goldreich, Shafi Goldwasser, and Silvio Micali. How to construct random functions (extended abstract). In *25th FOCS*, pages 464–479. IEEE Computer Society Press, October 1984. (Cited on page 3, 31)

[GØSW23] Lorenzo Grassi, Morten Øygarden, Markus Schofnegger, and Roman Walch. From farfalle to megafono via ciminion: The PRF hydra for MPC applications. In Carmit Hazay and Martijn Stam, editors, *EUROCRYPT 2023, Part IV*, volume 14007 of *LNCS*, pages 255–286. Springer, 2023. (Cited on page 3)

[GRR+16] Lorenzo Grassi, Christian Rechberger, Dragos Rotaru, Peter Scholl, and Nigel P. Smart. MPC-friendly symmetric key primitives. In Edgar R. Weippl, Stefan Katzenbeisser, Christopher Kruegel, Andrew C. Myers, and Shai Halevi, editors, *ACM CCS 2016*, pages 430–443. ACM Press, October 2016. (Cited on page 3)

[GRS23] Venkatesan Guruswami, Atri Rudra, and Madhu Sudan. Essential coding theory. *Draft available at http://www.cse.buffalo.edu/faculty/atri/courses/coding-theory/book*, 2023. (Cited on page 8)

[HBD+22] Andreas Hulsing, Daniel J. Bernstein, Christoph Dobraunig, Maria Eichlseder, Scott Fluhrer, Stefan-Lukas Gazdag, Panos Kampanakis, Stefan Kolbl, Tanja Lange, Martin M Lauridsen, Florian Mendel, Ruben Niederhagen, Christian Rechberger, Joost Rijneveld, Peter Schwabe, Jean-Philippe Aumasson, Bas Westerbaan, and Ward Beullens. Sphincs+. Technical report, National Institute of Standards and Technology, 2022. https://csrc.nist.gov/Projects/post-quantum-cryptography/selected-algorithms-2022. (Cited on page 34)

[HHM+24] Lena Heimberger, Tobias Hennerbichler, Fredrik Meisingseth, Sebastian Ramacher, and Christian Rechberger. Oprfs from isogenies: Designs and analysis. In Jianying Zhou, Tony Q. S. Quek, Debin Gao, and Alvaro Cardenas, editors, *ASIACCS 2024*. ACM Press, 2024. (Cited on page 28)

[HILL99] Johan Håstad, Russell Impagliazzo, Leonid A. Levin, and Michael Luby. A pseudorandom generator from any one-way function. *SIAM J. Comput.*, 28(4):1364–1396, 1999. (Cited on page 8)

[HJ10] Nick Howgrave-Graham and Antoine Joux. New generic algorithms for hard knapsacks. In Henri Gilbert, editor, *EUROCRYPT 2010*, volume 6110 of *LNCS*, pages 235–256, May / June 2010. (Cited on page 15)

[HPS98] Jeffrey Hoffstein, Jill Pipher, and Joseph H. Silverman. NTRU: A ring-based public key cryptosystem. In Joe Buhler, editor, *ANTS*, volume 1423 of *LNCS*, pages 267–288. Springer, 1998. (Cited on page 3)

[IKNP03] Yuval Ishai, Joe Kilian, Kobbi Nissim, and Erez Petrank. Extending oblivious transfers efficiently. In Dan Boneh, editor, *CRYPTO 2003*, volume 2729 of *LNCS*, pages 145–161, August 2003. (Cited on page 10)

[IKNZ23] Yuval Ishai, Mahimna Kelkar, Varun Narayanan, and Liav Zafar. One-message secure reductions: On the cost of converting correlations. In Helena Handschuh and Anna Lysyanskaya, editors, *CRYPTO 2023, Part I*, volume 14081 of *LNCS*, pages 515–547. Springer, 2023. (Cited on page 11, 23)

[IKOS07] Yuval Ishai, Eyal Kushilevitz, Rafail Ostrovsky, and Amit Sahai. Zero-knowledge from secure multiparty computation. In David S. Johnson and Uriel Feige, editors, *39th ACM STOC*, pages 21–30. ACM Press, June 2007. (Cited on page 3, 30)

[KHS+23] Seongkwang Kim, Jincheol Ha, Mincheol Son, ByeongHak Lee, Dukjae Moon, Joohee Lee, Sangyub Lee, Jihoon Kwon, Jihoon Cho, Hyojin Yoon, and Jooyoung Lee. AIM: symmetric primitive for shorter signatures with stronger security. In Weizhi Meng, Christian Damsgaard Jensen, Cas Cremers, and Engin Kirda, editors, *ACM CCS 2023*, pages 401–415. ACM Press, 2023. (Cited on page 7, 34)

[KKW18] Jonathan Katz, Vladimir Kolesnikov, and Xiao Wang. Improved non-interactive zero knowledge with applications to post-quantum signatures. In David Lie, Mohammad Mannan, Michael Backes, and XiaoFeng Wang, editors, *ACM CCS 2018*, pages 525–537. ACM Press, October 2018. (Cited on page 6, 30)

[KS08] 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, 2008. (Cited on page 26)

[KZ20] Daniel Kales and Greg Zaverucha. An attack on some signature schemes constructed from five-pass identification schemes. In *CANS 20*, LNCS, pages 3–22, 2020. (Cited on page 30, 34)

[KZ22] Daniel Kales and Greg Zaverucha. Efficient lifting for shorter zero-knowledge proofs and post-quantum signatures. Cryptology ePrint Archive, Report 2022/588, 2022. https://eprint.iacr.org/2022/588. (Cited on page 30, 34, 36)

[Lev85] Leonid A. Levin. One-way functions and pseudorandom generators. In *17th ACM STOC*, pages 363–365. ACM Press, May 1985. (Cited on page 3)

[LLNW16] Benoît Libert, San Ling, Khoa Nguyen, and Huaxiong Wang. Zero-knowledge arguments for lattice-based accumulators: Logarithmic-size ring signatures and group signatures without trapdoors. In Marc Fischlin and Jean-Sébastien Coron, editors, *EUROCRYPT 2016, Part II*, volume 9666 of *LNCS*, pages 1–31, May 2016. (Cited on page 36)

[LN17] Yehuda Lindell and Ariel Nof. A framework for constructing fast MPC over arithmetic circuits with malicious adversaries and an honest-majority. In Bhavani M. Thuraisingham, David Evans, Tal Malkin, and Dongyan Xu, editors, *ACM CCS 2017*, pages 259–276. ACM Press, October / November 2017. (Cited on page 36)

[LNS21] Vadim Lyubashevsky, Ngoc Khanh Nguyen, and Gregor Seiler. SMILE: set membership from ideal lattices with applications to ring signatures and confidential transactions. In Tal Malkin and Chris Peikert, editors, *CRYPTO 2021, Part II*, volume 12826 of *LNCS*, pages 611–640. Springer, 2021. (Cited on page 7, 36)

[Mea86] Catherine A. Meadows. A more efficient cryptographic matchmaking protocol for use in the absence of a continuously available third party. In *IEEE Symposium on Security and Privacy 1986*, pages 134–137. IEEE Computer Society, 1986. (Cited on page 6, 28, 29)

[MM11] Daniele Micciancio and Petros Mol. Pseudorandom knapsacks and the sample complexity of LWE search-to-decision reductions. In Phillip Rogaway, editor, *CRYPTO 2011*, volume 6841 of *LNCS*, pages 465–484, August 2011. (Cited on page 15)

[MRR20] Payman Mohassel, Peter Rindal, and Mike Rosulek. Fast database joins and PSI for secret shared data. In *ACM CCS 20*, pages 1271–1287. ACM Press, 2020. (Cited on page 6)

[MRV99] Silvio Micali, Michael O. Rabin, and Salil P. Vadhan. Verifiable random functions. In *40th FOCS*, pages 120–130. IEEE Computer Society Press, October 1999. (Cited on page 3)

[Noe15] Shen Noether. Ring signature confidential transactions for monero. Cryptology ePrint Archive, Report 2015/1098, 2015. https://eprint.iacr.org/2015/1098. (Cited on page 35)

[NR97] Moni Naor and Omer Reingold. Number-theoretic constructions of efficient pseudo-random functions. In *38th FOCS*, pages 458–467. IEEE Computer Society Press, October 1997. (Cited on page 3)

[PS00] David Pointcheval and Jacques Stern. Security arguments for digital signatures and blind signatures. *Journal of Cryptology*, 13(3):361–396, June 2000. (Cited on page 32)

[Raz87] Alexander A Razborov. Lower bounds on the size of bounded depth circuits over a complete basis with logical addition. *Mathematical Notes of the Academy of Sciences of the USSR*, 41(4):333–338, 1987. (Cited on page 10)

[Roy22] Lawrence Roy. Softspokenot: Quieter OT extension from small-field silent VOLE in the minicrypt model. In Yevgeniy Dodis and Thomas Shrimpton, editors, *CRYPTO 2022, Part I*, volume 13507 of *LNCS*, pages 657–687. Springer, 2022. (Cited on page 10)

[RR] Peter Rindal and Lawrence Roy. libOTe: an efficient, portable, and easy to use Oblivious Transfer Library. https://github.com/osu-crypto/libOTe. (Cited on page 27, 37)

[RRT23] Srinivasan Raghuraman, Peter Rindal, and Titouan Tanguy. Expand-convolute codes for pseudorandom correlation generators from LPN. In Helena Handschuh and Anna Lysyanskaya, editors, *CRYPTO 2023, Part IV*, volume 14084 of *LNCS*, pages 602–632. Springer, 2023. (Cited on page 6, 10, 20, 25)

[RST01] Ronald L. Rivest, Adi Shamir, and Yael Tauman. How to leak a secret. In Colin Boyd, editor, *ASIACRYPT 2001*, volume 2248 of *LNCS*, pages 552–565, December 2001. (Cited on page 3, 35)

[SHB23] István András Seres, Máté Horváth, and Péter Burcsi. The legendre pseudorandom function as a multivariate quadratic cryptosystem: security and applications. *Applicable Algebra in Engineering, Communication and Computing*, pages 1–31, 2023. (Cited on page 28)

[Smo87] Roman Smolensky. Algebraic methods in the theory of lower bounds for Boolean circuit complexity. In Alfred Aho, editor, *19th ACM STOC*, pages 77–82. ACM Press, May 1987. (Cited on page 10)

[TKS+19] Wilson Abel Alberto Torres, Veronika Kuchta, Ron Steinfeld, Amin Sakzad, Joseph K. Liu, and Jacob Cheng. Lattice RingCT V2.0 with multiple input and multiple output wallets. In Julian Jang-Jaccard and Fuchun Guo, editors, *ACISP 19*, volume 11547 of *LNCS*, pages 156–175, July 2019. (Cited on page 36)

[Yao82] Andrew Chi-Chih Yao. Theory and applications of trapdoor functions (extended abstract). In *23rd FOCS*, pages 80–91. IEEE Computer Society Press, November 1982. (Cited on page 3)

[Yao86] Andrew Chi-Chih Yao. How to generate and exchange secrets (extended abstract). In *27th FOCS*, pages 162–167. IEEE Computer Society Press, October 1986. (Cited on page 26)