Correlated Pseudorandom Functions from Variable-Density LPN

Elette Boyle, Geoffroy Couteau, Niv Gilboa, Yuval Ishai, Lisa Kohl, Peter Scholl

Correlated Pseudorandom Functions from Variable-Density LPN

Elette Boyle [*] Geoffroy Couteau [†] Niv Gilboa [‡] Yuval Ishai [§] Lisa Kohl [¶]
Peter Scholl [‖]

November 13, 2020

Abstract

Correlated secret randomness is a useful resource for many cryptographic applications. We initiate the study of *pseudorandom correlation functions* (PCFs) that offer the ability to securely generate virtually unbounded sources of correlated randomness using only local computation. Concretely, a PCF is a keyed function F_k such that for a suitable joint key distribution (k_0, k_1), the outputs $(f_{k_0}(x), f_{k_1}(x))$ are indistinguishable from instances of a given target correlation. An essential security requirement is that indistinguishability hold not only for outsiders, who observe the pairs of outputs, but also for *insiders* who know one of the two keys.

We present efficient constructions of PCFs for a broad class of useful correlations, including oblivious transfer and multiplication triple correlations, from a *variable-density* variant of the Learning Parity with Noise assumption (VDLPN). We also present several cryptographic applications that motivate our efficient PCF constructions.

The VDLPN assumption is independently motivated by two additional applications. First, different flavors of this assumption give rise to weak pseudorandom function candidates in depth-2 $\mathsf{AC}^0[\oplus]$ that can be conjectured to have subexponential security, matching the best known learning algorithms for this class. This is contrasted with the quasipolynomial security of previous (higher-depth) $\mathsf{AC}^0[\oplus]$ candidates. We support our conjectures by proving resilience to several classes of attacks. Second, VDLPN implies simple constructions of pseudorandom generators and weak pseudorandom functions with security against XOR related-key attacks.

[*]IDC Herzliya, eboyle@alum.mit.edu
[†]CNRS, IRIF, Université de Paris, couteau@irif.fr
[‡]Ben-Gurion University, niv.gilboa@gmail.com
[§]Technion, yuvali@cs.technion.ac.il
[¶]Cryptology Group, CWI Amsterdam, lisa.kohl@cwi.nl
[‖]Aarhus University, peter.scholl@cs.au.dk

Contents

1 Introduction

Correlated secret randomness is a ubiquitous resource in cryptography. A one-time pad, namely a pair of identical random keys, enables perfectly secure communication. Kilian [Kil88] and Beaver [Bea91, Bea95] showed that more complex forms of correlated randomness can similarly facilitate *secure multiparty computation* (MPC)—protocols that enable two or more parties to jointly compute a function of secret inputs revealing nothing beyond the output. A useful example is an *oblivious transfer* (OT) correlation, where one party is given two random bits (s_0, s_1) and another party gets (b, s_b) for a random bit b. Cryptographic power stems from the fact that the correlation forms a *non-product distribution* in which neither party can determine the secret of the other party.

Such correlations not only provide feasibility results, but are a central tool in essentially all concretely efficient instantiations of MPC in the setting of a dishonest majority. For example, two honest-but-curious parties can securely evaluate any Boolean circuit with s AND gates (and an arbitrary number of XOR/NOT gates) using $2s$ independent instances of an OT correlation, and communicating $2s$ bits per party. Moreover, the local computation of both parties involves just a small constant number of bit-operations per gate [GMW87, Gol04]. The efficiency and simplicity of MPC protocols based on correlated randomness gives rise to the following common paradigm. In an *offline preprocessing phase*, before the inputs are known, the parties use a dedicated protocol to securely generate correlated randomness. Then, once the inputs are available, the parties use a (typically very efficient) *online protocol* to securely evaluate the target function by consuming the correlated randomness. The main challenge, and the core bottleneck of almost all practically oriented protocols, is to design efficient methods to securely generate correlated randomness, without revealing more information than prescribed by the joint distribution.

A sequence of recent works [BCGI18, BCG+19b, BCG+19a, BCG+20] put forth a new technique for securely generating correlated randomness via *pseudorandom correlation generators* (PCGs). In the two-party case, a PCG provides a means for locally expanding a short correlated pair of seeds to a longer instance of a pseudorandom correlation. PCGs enable secure computation with "silent" preprocessing, where parties use a small amount of communication to securely generate the correlated seeds and then expand them to the target correlation without any interaction. Moreover, recent PCG constructions achieve this for useful correlations and with good concrete efficiency.

However, PCGs come with a major limitation: The expansion of the correlated seeds is an "all-or-nothing" procedure, where the target correlation is produced *all at once* without enabling fast random access to the long output.[1] This is similar to the limitation of a standard pseudorandom generator (PRG), except that existing PCG constructions do not even support the type of (stateful) *incremental* evaluation enabled by standard PRGs in a "stream-cipher mode." This limits the use of PCGs to a monolithic form of silent preprocessing that requires parties to generate and store big amounts of correlated randomness they *might* want to use in the future.

In this work, we initiate the study of the natural desired target: a *pseudorandom correlation function*[2] (PCF), which extends a PCG analogously to the way a *pseudorandom function* (PRF) [GGM86] extends a PRG. Concretely, we seek to design short correlations of keys that can be expanded "on the fly" to a *virtually unbounded* number of pseudorandom correlation instances, and which further enable fast random access to these instances.

A bit more precisely, recall that a PRF is a keyed function $f_k : \{0,1\}^n \to \{0,1\}^{\ell(n)}$, such that

[1] This applies to the so-called "dual" variant of PCG constructions that can achieve an arbitrary polynomial stretch; settling for at most a quadratic stretch, the "primal" variant allows random access to the output [BCGI18, SGRR19].

[2] One could alternatively view a PCF as a pair of *correlated pseudorandom functions*. The term *pseudorandom correlation function* (similarly to a pseudorandom correlation *generator*) refers to a single function f_k that samples from a pseudorandom correlation given suitably correlated keys.

for a secret random key k, the outputs of f_k are computationally indistinguishable from those of a random function. A *strong PRF* (or just PRF) is secure against distinguishers that query the function on arbitrary inputs x, while for a *weak PRF* (WPRF) the distinguisher is only given *samples* $(x_i, f_k(x_i))$ for uniformly random inputs x_i. Generalizing this notion, a (two-party) PCF is a keyed function f_k such that for a suitable joint key distribution (k_0, k_1), the outputs $(f_{k_0}(x_i), f_{k_1}(x_i))$ are indistinguishable from instances of a given target correlation. An essential security requirement is that indistinguishability hold not only for outsiders, who observe the pairs of outputs, but also for *insiders* who know one of the two keys. Here too, one can consider both a strong PCF and a weak PCF. We use the weak notion by default, since it is easier to construct and it suffices for our motivating applications.

Traditional techniques for upgrading a PRG to a PRF in the standard setting, such as the tree-based GGM construction [GGM86], do not apply here. The challenge lies in the requirement of security against insiders. Applying a GGM-style approach would require a PCG that expands its seeds to two instances of its own seed correlation. This seems infeasible for current PCG constructions and calls for a different approach to PCF design.

1.1 Overview of Contributions

In this work, we initiate a systematic study of pseudorandom correlation functions, and investigate related primitives and assumptions. We make the following main contributions.

Definition and generic construction. We start by formally defining (weak and strong) PCFs and put forth a natural general template for constructing them. The construction combines a (weak) PRF f_k with a *function secret sharing* (FSS) [BGI15] scheme for either the PRF class itself or closely related function classes. A (two-party) FSS scheme for a function class $\mathcal{F}$ enables splitting any function $f \in \mathcal{F}$ into two succinctly described functions (f_0, f_1), such that $f = f_0 + f_1$ and each share f_i hides f. We show how our template can be instantiated using known FSS schemes for circuits [DHRW16, BGI+18] together with any PRF. This leads to a general PCF construction for a useful class of "additive" correlations, which includes most of the useful correlations for MPC, under a standard cryptographic assumption, namely the *Learning With Errors* (LWE) assumption [Reg05].

PCFs from VDLPN. While the above LWE-based construction provides a theoretical feasibility result, it has poor asymptotic and concrete efficiency. Moreover, the construction uses the heavy machinery of fully homomorphic encryption and leaves open the possibility of constructing useful PCF instances from other, seemingly weaker, assumptions. We show how to efficiently construct PCFs for a broad class of useful correlations, including oblivious transfer (OT), vector oblivious linear-function evaluation (VOLE) [ADI+17, BCGI18], and "multiplication triple" correlations [Bea91], from a natural *variable-density* variant of the Learning Parity with Noise (LPN) assumption [BFKL94], or VDLPN for short, that we introduce and study in this work. These efficient PCF constructions are motivated by applications to MPC and non-interactive zero knowledge.

Applications of VDLPN. The VDLPN assumption is independently motivated by two additional applications. First, different flavors of this assumption give rise to WPRF candidates in depth-2 $\mathsf{AC}^0[\oplus]$ (concretely, XOR of conjunctions of input variables and their negations) that can be conjectured to have subexponential security, matching the best known learning algorithms for this class [HS07]. This is contrasted with the quasipolynomial security of previous (higher-depth) $\mathsf{AC}^0[\oplus]$ candidates [ABG+14, BR17]. We support the VDLPN assumption and related conjectures by proving resilience to several classes of attacks, including *linear attacks* that use the input samples to find a biased linear combination of the outputs, and *algebraic* attacks that exploit low rational degree. Finally, we observe that VDLPN implies simple constructions of

PRGs and WPRFs with security against XOR related-key attacks. Previous constructions are either heuristic or rely on strong assumptions such as multilinear maps [ABP19].

The remainder of the Introduction is organized as follows. First, in Section 1.2, we present our main candidate as a low-complexity WPRF, and then (Section 1.3) explain how its conjectured security can be viewed as a variable-density variant of the standard LPN assumption. In Section 1.4, we discuss applications of this candidate WPRF to security against XOR related-key attacks. Then, in Section 1.5, we discuss the construction of PCFs, which relies on the "FSS-friendliness" of our WPRF candidate. We conclude by discussing applications and comparison with alternative approaches and related works. A more detailed discussion of the conjectured security of our main WPRF candidate and its variants is deferred to Section 2.

1.2 Our Low-Complexity WPRF Candidate

Motivated by the goals of improving the efficiency of PCFs and diversifying the underlying assumptions, we put forth new WPRF candidates that are "FSS-friendly" in the sense of being compatible with existing PRG-based FSS schemes. Our candidates are in a very low complexity class: the class $\mathsf{XOR} \circ \mathsf{AND}$ of polynomial-size, depth-2 boolean circuits with one layer of AND gates at the bottom and a single XOR gate at the top (both of arbitrary fan-in).[3] We also refer to such a circuit as an *XNF formula* (for XOR Normal Form). This is similar to DNF, except for replacing disjunction (OR) by XOR. We conjecture our candidates to have *subexponential* security in both the key length and the input length. Concretely, our main candidate $f_k : \{0,1\}^n \to \{0,1\}$ is of the form

$$f_k(x) = \bigoplus_{i=1}^{D} \bigoplus_{j=1}^{w} \bigwedge_{h=1}^{i} (x_{i,j,h} \oplus k_{i,j,h}), \tag{1}$$

where w, D can be set to the security parameter λ, and $n = w \cdot D \cdot (D-1)/2$. We conjecture that this candidate is secure against $2^{o(n^{1/3})}$-time distinguishers. The security of this WPRF candidate is based on a natural *variable-density* flavor of the well-studied *learning parity with noise* (LPN) assumption that we will discuss below. A slightly different candidate is plausibly secure against $2^{o(\sqrt{n})}$-time distinguishers, matching the best known learning algorithm for this class [HS07]. (For both variants, restricting the distinguisher to 2^D samples and D to be, say, $n^{o(1)}$, we get plausible security against $2^{o(n/\log n)}$-time distinguishers.) Subexponential security is good enough to support λ bits of security (against 2^λ-bounded adversaries) in $\mathrm{poly}(\lambda)$ time, and is typically the strongest level of security one can hope to obtain from standard cryptographic assumptions. For a more thorough discussion on the security of our main candidate and its variants we refer to Section 2.3.

In contrast to our candidates, previous WPRF candidates in $\mathsf{AC}^0[\oplus]$ (namely, of constant-depth polynomial-size circuits with AND/OR/XOR gates of unbounded fan-in) were restricted to *quasipolynomial* security, which is considered "borderline insecure" and cannot support λ bits of security in $\mathrm{poly}(\lambda)$ time. Thus, our candidates fill a gap in the current landscape of (weak) PRF candidates in low complexity classes. See Section 2 below for related work.

We support the conjectured security of our candidates by analysis that proves their security against several classes of natural attacks, capturing essentially any relevant class of attacks we are aware of. This includes, for instance, *linear* attacks that try to correlate the outputs via an input-dependent linear combination, as well as *algebraic* attacks that exploit a low *rational degree*. The latter yielded a quasipolynomial-time distinguisher for a previous WPRF candidate in $\mathsf{AC}^0[\oplus]$ [ABG+14, BR17].

[3] As is common in the study of constant-depth PRFs, we consider the complexity of mapping the input to the output when the key is fixed, where inputs can be negated without counting towards the depth. The latter is consistent with, for instance, a DNF formula having depth 2. Viewed as a function of both the input and the key, the depth increases to 3.

Assuming our conjectures, the complexity class $\mathsf{AC}^0[\oplus]$ does not admit a $2^{n^{o(1)}}$-time learning algorithm, even under the uniform distribution. In fact, the same holds for the class of XNF formulas that can be alternatively viewed as sparse $\mathbb{F}_2$-polynomials in inputs and their negations. This also implies similar hardness for learning sparse $\mathbb{F}_2$-polynomials (without negations) in the standard PAC model, which allows for arbitrary input distributions. Our analysis and conjectured hardness assumptions complement the $2^{\tilde{O}(\sqrt{n})}$-time PAC learning algorithm from [HS07] for sparse polynomials, which also applies to XNF formulas by changing the input distribution. The conjectured hardness should be contrasted with efficient learning algorithms when *membership queries* are allowed [SS96, Bsh97]. This corresponds to our WPRF candidates not being strong PRFs.

1.3 Variable-Density LPN

The security hypothesis for our main WPRF candidate from Eq. (1) can be cast as a new natural variant of the standard *Learning Parity with Noise* (LPN) assumption [BFKL94]. We explain this below.

In its dual formulation, LPN asserts that for suitably chosen parity-check matrix $H \in \mathbb{F}_2^{N \times M}$ (where $M > N$) and noise vector $e \in \mathbb{F}_2^M$, the distribution (H, He) is pseudorandom, namely it is indistinguishable from (H, r) for a uniform vector $r \in \mathbb{F}_2^N$ chosen independently of H. Here H is typically chosen to be a uniformly random matrix and e an i.i.d. noise vector in which each entry is set to 1 with probability $0 < p < 1/2$. Many other flavors of LPN have been used for different cryptographic applications. For instance, H can be structured [HKL+12, MBD+18] or even of "medium density" [MTSB13]. The noise vector e is sometimes chosen to have a fixed weight or even a *regular* structure [AFS03], where e consists of disjoint blocks and each block contains a single nonzero entry in a random position.

We can interpret our WPRF candidate from Eq. (1) as a *variable density* version of the dual LPN assumption described above in the following way: For each i, the inner term $\bigwedge_{h=1}^{i}(x_{i,j,h} \oplus k_{i,j,h})$ can be rephrased as an inner product of two weight-1 vectors $H_{i,j}, e_{i,j} \in \{0,1\}^{2^i}$.[4] Concatenating the vectors $H_{i,j}$ for all $i \in [D], j \in [w]$ results in one row of the matrix H (corresponding to one input x), and concatenating the vectors $e_{i,j}$ results in the error vector e. Note that the length of each such vector is $M = O(w2^D)$ and each block has twice the size and half the density of the previous block. Given N inputs x for the WPRF, we obtain a matrix H of dimension N times M, such that one output of our original WPRF construction corresponds exactly to one entry of He.

We refer to the conjectured security of this flavor of variable-density LPN as the *VDLPN assumption*. The core idea of VDLPN is that it allows the dimension of H and e to be superpolynomial in the input length n (while each entry of He is polynomial-time computable) without introducing too much structure or biasing the output. As outlined above, this assumption is equivalent to the security of the WPRF candidate in Eq. (1). In Section 2 we further motivate this choice and discuss its provable security against a variety of relevant attacks. We also discuss other flavors of the VDLPN assumption that correspond to alternative WPRF candidates.

1.4 Application to XOR-RKA Security

Independently of the PCF motivation, our new WPRF candidates are motivated by another cryptographic application: achieving security against *related-key attacks* (RKA) with respect to XOR functions. RKA security captures a model in which the attacker is allowed to see several instances of a primitive where the keys are not independent, but instead satisfy a relation of its choice. XOR-RKA security captures the setting where the adversary has access to the primitive with keys $k, k \oplus \Delta_1, k \oplus \Delta_2, \cdots$, for fixed offsets of his choice. The VDLPN assumption that

[4] To see this, one can think of $H_{i,j}$ as the weight-1 vector with non-zero position $(x_{i,j,1}, \ldots, x_{i,j,i})$ (read as an integer between 0 and $2^i - 1$), and $e_{i,j}$ the weight-1 vector with non-zero position $(1 \oplus k_{i,j,1}, \ldots, 1 \oplus k_{i,j,i})$.

implies the security of our main WPRF candidate actually implies the stronger XOR-RKA security for free. This follows from the fact that the WPRF from Eq. (1) can be written in the form $f_k(x) = h(k \oplus x)$, and so tampering with the key bits is equivalent to tampering with the (random) inputs. For details, see Section 6.5.

XOR-RKA is arguably the most natural flavor of RKA security, capturing fault injection attacks where an adversary can induce bit flips in cryptographic hardware and other forms of tampering (see [GLM+04] and references therein). However, it is typically very hard to prove — the only "provable" XOR-RKA secure PRF is based on the very strong cryptographic assumption of multilinear maps [ABP19]. Our main WPRF candidate implies simple PRGs and WPRFs whose XOR-RKA security follows from the VDLPN assumption. These can in turn be used for constructing other types of XOR-RKA secure variants of primitives that can be based on standard WPRF, including identification and authentication schemes, semantically secure encryption schemes [AHI11], and passive XOR-RKA secure strong PRFs [AW14]. Finally, since our WPRF candidate has the form $f_k(x) = h(k \oplus x)$, its security implies that the function h is *correlation-robust* in the sense of [IKNP03].

We stress that these XOR-RKA primitives depend on a new security assumption that is yet to withstand the test of time. This follows previous theory-oriented works that introduce new simple PRF candidates and study their resistance to concrete classes of attacks [MV12, ABG+14, BR17, BIP+18].

1.5 From FSS-friendly WPRF to PCF

We start this section by briefly explaining how to obtain a PCF from a WPRF together with a suitable function secret sharing scheme, and then show why our WPRF candidate is "FSS-friendly."

Generic construction of PCFs. Recall that an FSS scheme for a function class $\mathcal{F}$ allows splitting a function $f \in \mathcal{F}$ *compactly* into two functions (f_0, f_1), such that $f = f_0 + f_1$ and each share f_i individually hides f. One example for a useful correlation is *vector oblivious linear evaluation* (VOLE) over a field $\mathbb{F}$, where one party holds $(a, c_{0,i})$ and the other party holds $(b_i, c_{1,i})$ such that $c_{1,i} = c_{0,i} + a \cdot b_i$, for fixed a. Now, a WPRF class $\mathcal{F}$ over $\mathbb{F}$ together with a FSS[5] for $a \cdot \mathcal{F}$ allows to construct a PCF for VOLE as follows: First an element $a \xleftarrow{\$} \mathbb{F}$ and a key k for the WPRF are sampled at random, and then compact shares (f_0, f_1) of $a \cdot f_k$ are generated via the FSS. Now, a and f_0 are given to one party and k and f_1 to the other party. By the correctness of the FSS, for all x it holds $f_1(x) = -f_0(x) + a \cdot f_k(x)$, and by the security of the FSS neither party learns about the others parties secrets.

A PCF for random OT can be constructed either directly based on the described approach using the techniques of [BCG+19b, BCG+19a], or alternatively, from a WPRF family over $\mathbb{F}_2$ and a corresponding FSS. Both approaches additionally require a correlation-robust hash function (which, as discussed in Section 1.4, does not require any additional assumption in our case). We further show how to construct PCF for *oblivious linear evaluation* and *multiplication triples* given an FSS for both the WPRF class $\mathcal{F}$ and the *square* of $\mathcal{F}$, namely $\mathcal{F}^2 = \{f \cdot f' : f, f' \in \mathcal{F}\}$.

The above constructions only realize our default notion of *weak* PCF, where inputs are chosen at random. This is good enough in applications where a common source of public randomness is available. Moreover, in the random oracle model, one can easily obtain a strong PCF from a weak one by applying the random oracle to the input.

"FSS-friendliness" of our candidates. The above approach can be instantiated from any WPRF together with a "high-end" FSS from LWE. Our WPRF candidates, on the other hand,

[5] Note that for the case of VOLE, and similarly for random OT correlations, one can in fact replace the FSS primitive by the simpler *puncturable pseudorandom function* primitive [KPTZ13, BW13, BGI14].

are designed to be *FSS-friendly*, in the sense that they can be evaluated by lightweight function secret sharing schemes based on the existence of one-way functions. The primitive we use is a *distributed point function* (DPF) [GI14], namely an FSS scheme for the class of *point functions* $\{P_\alpha\}_{\alpha\in\{0,1\}^*}$, where the function $P_\alpha : \{0,1\}^{|\alpha|} \to \mathbb{F}_2$ evaluates to 1 on input α, and to 0 on all other inputs. Concretely efficient DPF schemes from any PRG were given in [BGI15, BGI16b].

Our main WPRF candidate, described in Eq. (1), is "FSS-friendly" as it can be seen as a sum of $w \cdot D$ point functions: Each AND term $\bigwedge_{h=1}^{i}(x_{i,j,h} \oplus k_{i,j,h})$ can be viewed as evaluating a point function P_α with $\alpha = (1 \oplus k_{i,j,1}, \ldots, 1 \oplus k_{i,j,i})$. Here we make crucial use of the fact that the identity of the variables in each term of Eq. (1) is public and only whether each variable is negated or not is secret.

Additionally, we propose an alternative low-complexity WPRF candidate that is even more FSS-friendly than our main candidate:

$$f_k(x) = \bigoplus_{i=1}^{D} \bigoplus_{j=1}^{w} \bigwedge_{h=1}^{i} (x_{j,h} \oplus k_{j,h}). \tag{2}$$

This candidate exploits the fact that in the best known DPF constructions [BGI16b], sharing the point function P_α directly gives a sharing of all $P_{\alpha'}$ for α' a prefix of α. All of our candidates can also be adapted to have outputs over rings instead of $\mathbb{F}_2$, with the same level of FSS-friendliness. This leads to PCF constructions for useful correlations over general rings — see Section 6.3.

When plugging in our concrete candidate WPRFs we obtain further benefits. We construct a PCF for arbitrary, degree-2 correlations — additive secret shares of a random string X, together with shares of $Y = p(X)$, for some multivariate degree-2 polynomial p — with a *universal* setup procedure. The latter means that the PCF key generation is independent of the polynomial p, and later during expansion the parties have PCFs for *any* choice of p. This overcomes the limitation of a previous LPN-based PCG construction from [BCG+19b], which required computing all N^2 monomials.

Compared to the generic LWE-based construction of PCFs, the LPN-style assumptions that underly our specialized constructions seem qualitatively weaker. For instance, they are not known to imply additively homomorphic encryption or even collision-resistant hashing. VDLPN-based PCFs also have attractive efficiency features that beat the generic alternative both asymptotically and concretely (by several orders of magnitude). Finally, they achieve *perfect correctness* whereas LWE-based constructions have negligible error probability that we do not know how to remove. See Section 1.7 for a more detailed comparison.

1.6 Applications of PCFs

PCFs give rise to a number of interesting cryptographic applications, which we briefly outline here, and in more detail in Section 10.

Secure computation with correlated randomness. The most natural use case, as already mentioned, is their ability to produce a practically unlimited amount of correlated randomness for use in secure computation protocols. Similar to the case of PCGs, this application is not entirely immediate — a PCF cannot substitute for an ideal source of correlated randomness in *every* protocol, since knowing a short representation of this randomness can in some cases contradict security. We can show, however, that PCFs *can* be plugged directly into a large class of natural, practical protocols in a secure manner; this holds for any protocol that is secure even when a corrupt party can influence its own correlated randomness. In particular, this property is satisfied by many, standard secure multi-party computation (MPC) protocols in the preprocessing model, so PCFs allow us to transform these protocols to have *reusable preprocessing*. Here, a one-time setup protocol is first performed to distribute the keys for a PCF. After this setup, the PCF can be used for as many instances of the MPC protocol as is needed, without having to re-run the setup.

Two-round MPC. A special class of multi-party computation protocols that have been developed recently [GGHR14, BL18, GS18] is those with just *two rounds* of interaction, the minimum that is possible. The two-round protocol of Garg et al. [GIS18] uses a setup phase for producing a large number of random oblivious transfers, after which the entire protocols makes only black-box use of a pseudorandom generator. Replacing this oblivious transfer setup with PCFs, we obtain a fully reusable preprocessing phase, which after its setup, can be used for any number of two-round MPC protocols.

Non-interactive zero knowledge with fully reusable preprocessing. Non-interactive zero knowledge (NIZK) allows a prover to convince a verifier of the truth of some statement, by just sending a single message. NIZK with preprocessing allows a setup phase with a trusted third party, who generates secret proving and verification keys, which are given to the prover and verifier, respectively. Preprocessing NIZK can have information-theoretic security given OT or VOLE correlations [KMO89, CDI+19]. This motivated the use of a PCG-based approach in this context [BCGI18, BCG+19b, WYKW20]. Using PCFs, we can obtain preprocessing NIZKs with fully reusable preprocessing, where a single setup consisting of PCF keys can be used to prove an arbitrary number of statements. Compared with other recent NIZK constructions in the designated verifier setting [LQR+19], which can be based on LPN, the PCF approach requires a stronger preprocessing setup, but leads to simpler constructions with better concrete efficiency. These constructions do not require any cryptographic operations after expanding the PCF outputs.

Multi-party PCFs. Some of our two-party PCF constructions can be extended to the *multi-party setting*, where m parties can obtain correlated randomness, which remains secure even when up to $m-1$ keys have been corrupted. These extensions are possible by exploiting a *programmability* feature of the two-party PCFs, which means that some portion of the PCF outputs can be reused as outputs in a separate PCF instance. This allows generically constructing multi-party PCFs, in a similar way to previous constructions of multi-party PCGs [BCG+19b].

1.7 Advantages of the VDLPN-Based PCF Construction

Without an FSS-friendly WPRF such as ours, it seems necessary to rely on general-purpose FSS schemes based on "public-key assumptions." These can be constructed from the LWE assumption using special fully homomorphic encryption (FHE) schemes [DHRW16]. One can also combine a WPRF in NC^1 with an FSS for branching programs, where the latter can be based on the Decisional Diffie-Hellman assumption [BGI16a]. However, this approach suffers from an inverse polynomial correctness error and has a high computational cost.

Below, we analyze the properties of LWE-based constructions, and compare these with our approach. Specifically, we will consider an FSS construction based on the more efficient ring-LWE assumption, and instantiate this with an exponential ring-LWE modulus q; this allows to obtain FSS and PCFs with an exponentially small error probability [DHRW16], which is comparable to our constructions with perfect correctness.

Asymptotic efficiency. We first analyze an optimistic variant of a ring-LWE-based PCF, based on a WPRF which can be computed by a circuit of size $\tilde{O}(\lambda)$; note that the only candidates we are aware of satisfying this are our WPRF from VDLPN (in the regime $D = \mathsf{polylog}(\lambda)$), a permutation-based a candidate PRF by Miles and Viola [MV12] and an "asymptotically optimal" candidate from [BIP+18]. Fully homomorphic encryption from the ring-LWE assumption can be carried out with polylogarithmic overhead on top of the cleartext computation [GHS12]. However, if we want an exponentially small correctness error in the FSS then the ciphertext modulus q is required to be exponential in the security parameter; this multiplies the overhead of homomorphic evaluation by $\tilde{O}(\lambda)$, translating to a computational cost of $\tilde{O}(\lambda^2)$ for each PCF evaluation. Regarding storage costs, the PCF key in this construction consists of $\tilde{O}(\lambda)$ ring-LWE

ciphertexts, giving a total key size of $\tilde{O}(\lambda^3)$ bits.[6]

Moreover, the above computational cost only holds for a WPRF with $\tilde{O}(\lambda)$ size circuits. PRF constructions from standard assumptions such as LWE, ring-LWE [BPR12], number-theoretic assumptions [NR04] or a natural generalization of AES [MV12] require circuits of size $\tilde{O}(\lambda^2)$, and result in $\tilde{O}(\lambda^4)$ computation for PCF.

In contrast, our most aggressive candidate has key size $\tilde{O}(\lambda^2)$ bits and computational cost $\tilde{O}(\lambda)$ PRG operations, clearly improving over the alternatives in both complexity measures.

Concrete efficiency. Thanks to its simplicity, our constructions should in practice be more concretely efficient than LWE-based approaches. For example, we estimate that in our PCFs for VOLE or OT, each party's PCF key can be around 120kB, or 2MB based on our most conservative assumption, for a target bound of 2^{30} samples. The parameters were chosen to achieve 100 bits of security against natural linear attacks such as BKW or the learning algorithm of [HS07], and are based on the optimized PCF construction – see Remark 5.6 in Section 5.2 and Section 6.4. For further discussions and concrete conjectures regarding the exact security of our candidate, we refer the reader to Section 9. For comparison, Boyle et al. [BCG+19b] considered building a PCG from ring-LWE-based function secret sharing (or homomorphic secret sharing) and a suitable pseudorandom generator. To obtain a reasonable computation time, the resulting PCG keys were larger than 3GB, and the stretch of the PCG was still sub-quadratic. Since a single FHE ciphertext is typically the order of several megabytes, and a PCF key will need many such ciphertexts, it seems inherent that LWE-based PCFs will suffer similarly in terms of concrete key size and/or computational cost.

Conceptually weaker assumption. LWE is a powerful assumption that implies, amongst other things, the existence of (leveled) fully homomorphic encryption. On the other hand, LPN-type assumptions, even in a low-noise regime such as VDLPN, are not known to imply additively homomorphic encryption or even collision-resistant hashing. Despite recent progress towards the latter [BLVW19, YZW+19], it is still unknown whether we can construct collision-resistant hashing based on the polynomial hardness of LPN. For additively homomorphic encryption, there are negative results showing that any LPN-based construction must make non-black-box use of the underlying ring, which seems to require techniques going beyond existing constructions [AAB15]. Therefore, our constructions show that PCFs, although a powerful primitive, can plausibly be realized under qualitatively weaker assumptions than before.

Perfect correctness. Our constructions satisfy perfect correctness for all parameter choices. With LWE, one can achieve negligible error probability by using a superpolynomial modulus, which requires a strong variant of LWE. We do not know how to obtain perfectly correct HSS under any variant of LWE, and when the modulus is restricted to be polynomial, current constructions have an inverse polynomial error probability.

1.8 Related Work

The study of secure computation with silent preprocessing is a recent but active research area [BCG+17, Sch18, BCGI18, BCG+19b, SGRR19, BCG+19a]. There is a long line of work which studied constructions of low-bias PRGs and PRFs in low complexity classes [NN90, MST03, LRTV09, Shp09, Vio10, MRRR14, LV17, AK19]. In particular, the work of [GV04] gives an ε-biased strong PRF in $\mathsf{AC}^0[\oplus]$; while it only achieves bias $\varepsilon \geq 1/\mathsf{superpoly}(\lambda)$, it was strengthened in [Hea08] to achieve exponentially small bias. Our result is incomparable: we only construct a *weak* low-bias PRF family, but in the much smaller class $\mathsf{XOR} \circ \mathsf{AND}$. Heuristic constructions of PRFs and weak PRFs with provable security against classes of attacks have been studied in several previous work [MV11, ABG+14, BIP+18].

[6] The number of ciphertexts could be reduced using packing techniques [GHS12], but this requires storing additional 'key-switching' material, and would not change the overall key size.

The combination PRFs and FSS (or the dual of homomorphic secret sharing) has been used before in different contexts. In particular, Boyle et al. used it to establish barriers for FSS [BGI15] and to obtain low-communication MPC protocols [BGI16a], and Bartusek et al. [BGMM20] used it to obtain two-round MPC protocols with reusable first round.

2 Technical Overview of our WPRF Candidates

In this section we give a more detailed technical overview of our new WPRF candidates, their design choices, and security analysis. We start with some general background.

The study of low-complexity cryptography has a long and rich history (see e.g. [Kha93, NR97, NRR00, AIK04, IKOS08, AR16, ABG+14, BIP+18] and references therein). Beyond the direct goal of minimizing the complexity of useful cryptographic primitives, this line of work is motivated by its relevance to hardness results and barriers in computational complexity theory and learning theory. Another, more recent motivation stems from the fact that many advanced primitives, such as secure computation, zero-knowledge proofs, fully homomorphic encryption (FHE), and indistinguishability obfuscation can induce their own efficiency metrics that motivate new designs of low-complexity primitives. Some relevant works in this direction include [IKOS08, ARS+15, CCF+16, MJSC16, LT17].

Our work gives yet another example of this kind: we efficiently realize PCFs by relying on WPRFs with a specific "FSS-friendly" structure. To instantiate this framework, we can use a "heavy hammer" approach, by relying on advanced constructions of FSS for all circuits [BGI15, DHRW16]. However, while such an approach is interesting for establishing feasibility (which we do in Section 5), it is unsatisfactory for several reasons. First, it is unlikely to lead to any concretely efficient candidate (in the same way that hybrid FHE can be achieved by combining any FHE scheme with any standard block cipher, but the resulting scheme will be highly inefficient, hence motivating the design of FHE-friendly ciphers [ARS+15, CCF+16, MJSC16]). Second, it requires a strong flavor of "homomorphic cryptography," which implies a severe limitation on the type of assumptions we can realistically hope to rely on and the level of concrete efficiency. Finally, a curious limitation of all known constructions of FSS of circuits is that they cannot achieve perfect correctness.

Therefore, we take the opposite road: rather than starting from advanced FSS for all circuits based of FHE-style assumptions, we ask whether the simplest and most efficient known FSS schemes [BGI16b], which can be based on any one-way function, are already sufficiently powerful to instantiate our framework. Namely, we ask:

Is there a weak PRF in the complexity class captured by known FSS schemes based on one-way functions?

The FSS schemes of [BGI16b] capture point functions (which are equal to 0 everywhere except on a single point) and other classes of functions, interval functions (which take a fixed value for all inputs from an interval, and 0 otherwise), multi-dimensional generalizations of the latter, decision trees with fixed topology, as well as all linear combinations of the above. All these classes can be expressed as sums of point functions applied to different projections of the inputs. The schemes from [BGI16b] achieve better efficiency than that obtained via independent instances of DPF. But from a feasibility point of view, all these functions can be expressed as depth-2 $\mathsf{AC}^0[\oplus]$ circuits.

WPRFs with quasipolynomial security are known to exist in complexity classes as low as AC^0, the class of polysize constant-depth circuits with arbitrary fan-in $\vee, \wedge$ gates, under standard cryptographic assumptions such as factoring and DDH [Kha93, NR97, NRR00] or assuming the existence of random local functions [AR16]. Furthermore, no weak PRF with better than

quasipolynomial security can exist in AC^0 [LMN89]. Strong PRFs with quasipolynomial security are known to exist in $\mathsf{AC}^0[\oplus]$ under standard cryptographic assumptions [Vio13], and quasipolynomial security is the best one can hope for in this class [RR97, CIKK16]. Finally, strong PRFs with exponential security are known in TC^0 [BPR12] and in the "almost constant-depth" variant of $\mathsf{AC}^0[\oplus]$ [YS16] under standard cryptographic assumptions, and heuristic constructions (with provable resistance against some classes of attacks) of strong PRFs in ACC^0 have been proposed in [BIP+18]. Our work proposes conceptually simple WPRF candidates in $\mathsf{AC}^0[\oplus]$ that have depth 2 (the best possible) and are FSS-friendly. See Table 1 for comparison with related work.

	Complexity Class		
Circuit Depth	AC^0	$\mathsf{AC}^0[\oplus]$	$\mathsf{ACC}^{0\dagger}$
Depth 2	Weak PRF [BFKL94] (quasipolynomial*)	**Weak PRF (subexponential*)**‡	Weak PRF [BIP+18] (exponential*)
Depth 3	Weak PRF [AR16] (quasipolynomial)	Weak PRF [ABG+14, BR17] (quasipolynomial*)	Strong PRF [BIP+18] (exponential*)
Depth > 3	Weak PRF [Kha93] (quasipolynomial)	Strong PRF [NR97, Vio13] (quasipolynomial)	–
Negative Results	No weak PRF with better than quasipolynomial security [LMN89]	No strong PRF with better than quasipolynomial security [RR97, CIKK16]	–

* Starred entries refer to (provable or heuristic) security against known classes of attacks, as opposed to security proofs via reductions to well-studied cryptographic assumptions.
† For entries in ACC^0, it suffices to consider $\mathsf{AC}^0[6]$, that is, the class AC^0 augmented with mod_6 gates.
‡ Subexponential security means that there exists $\epsilon > 0$ such that the candidate is secure against all distinguishers of size 2^{n^ϵ}.

Table 1: Comparison of positive and negative results for constant-depth PRFs. When measuring depth, we consider the complexity of mapping the input to the output when the key is fixed, and do not count negations of the input. For each candidate, we denote in parenthesis its conjectured level of security. Different constructions in the same class rely on incomparable assumptions. The entry shown in **bold** is from this work.

A natural approach. A natural approach to building weak pseudorandom functions in a low complexity class, which was the starting point of most prior works [BPR12, ABG+14, YS16, BIP+18], is to start from (variants of) the learning parity with noise assumption. The LPN assumption postulates that the function family $f_{s,B}(x) = \langle s, x\rangle + B(x)$ is a weak PRF, where $s \in \{0,1\}^n$ is a secret random vector, and $B(x)$ is a noise function which associates to each x a random noise coordinate that is biased towards zero. In spite of its low complexity, this weak PRF family is not efficient: the standard formulation of LPN requires entropy for each noise term, which is too much for storing B in the key, unless the number of samples is restricted to some fixed polynomial.

An approach followed in several works [BPR12, AKPW13, ABG+14, BIP+18] tries to replace this B by a low-entropy and relatively simple function. This was partially successful: in [BPR12, AKPW13], it was shown that, over $\mathbb{F}_q$ for a large modulus q (superpolynomial in [BPR12], polynomial in [AKPW13]), replacing $B(x)$ by some appropriate *rounding* of $\langle x, s\rangle$ leads to a weak PRF under the LWE assumption. However, the use of larger fields puts the construction in a higher complexity class, namely, TC^0. On the negative side, it was shown by Akavia et al. [ABG+14] that no choice of rounding function could possibly allow to base the construction directly on LPN (over $\mathbb{F}_2$) by the same approach. A candidate rounding function was suggested in [ABG+14], and supported by some security analysis against classes of attacks; however, it was later broken in quasipolynomial time [BR17]. More recently, a different choice of rounding

function, using mod-3 addition, was suggested in [BIP+18] and conjectured to resist the attack which breaks the candidate of Akavia et al.

2.1 Our Approach – the LPN Viewpoint

In this section we provide an intuition of our approach, this time starting from the primal LPN assumption. The final result of this section will be a more general definition of variable-density LPN which in particular captures our main candidate.

A first (unsuccessful) attempt. Assume a LPN-based weak PRF, written in the primal formulation: $f_{s,B}(x) = \langle s, x\rangle + B(x)$, where $s \in \{0,1\}^n$ and B is a function corresponding to sampling random noise biased towards 0 (i.e. $B(x) = 1$ with some fixed probability ε). What makes the description of the function B inherently inefficient (when aiming for superpolynomial security) is the following simple attack: consider an adversary getting N random samples $(x^{(i)}, f_{s,B}(x^{(i)}))_{i\leq N}$, arranged in a matrix A. Hence, the adversary gets $(A, A\cdot s+e)$, where e is a noise vector such that the i-th entry of e is $B(x^{(i)})$. To recover s (which suffices to distinguish all further samples from random), the adversary can attempt to guess a size-n subset of noise-free equations in the noisy linear system $A\cdot s+e$, and solve it with Gaussian elimination. With noise rate ε (that is, $\varepsilon = \mathcal{HW}(e)/N$), a straightforward calculation shows that the attack succeeds with probability roughly $\exp(-n\varepsilon)$, hence ε cannot be smaller than $1/n$, and the description size of B must be at least N/n. Therefore, the PRF key size is at least $n + N/n \geq \sqrt{N}$, where N is the number of adversarial queries.

However, the above attack scales with the *length* n of the secret vector s, not with its description complexity (and so do all known attacks on LPN). This suggests that we could rely on a variant of LPN with a secret vector s of much smaller description complexity $d(n)$; this way, the key size would scale as $d(n)+N/n$, which can potentially be exponentially smaller than N if $d(n)$ is exponentially smaller than n. Fortunately, there is a well-known candidate to instantiate this approach: one can sample the secret vector s from the *same Bernouilli distribution* as the noise vector e, and the resulting assumption will still be equivalent to the standard LPN assumption (see e.g. [YS16] for a formal statement and proof of this equivalence). Hence, we could set $n \approx N$, $\varepsilon \approx 1/n$, and have the key size grow as $\varepsilon(n)\cdot N$, almost independent of N (up to logarithmic terms, which we ignore in this high level discussion).

Unfortunately, this still does not quite work. The issue is that the *input* x to $f_{s,B}(x) = \langle s, x\rangle + B(x)$ has the same length as s: hence, if we make s superpolynomially large by setting $n \approx N$, the input x becomes superpolynomially large as well. To implement this approach efficiently, we would need the input x to have a short description as well. In other words, what we need is a variant of LPN which states that it is hard to distinguish $A \cdot s + e$ from random where s, e, and the rows of A, all have a short description (exponentially shorter than their length). Unfortunately, simply sampling A from a sparse (e.g.) Bernouilli distribution cannot work (since $A \cdot s + e$ would be sparse and easily distinguished from random).

A way around. Let us take a step back, and look at the two alternatives: we can set s to be short (and dense), hence allowing A to be relatively narrow, but then $A \cdot s + e$ is easily broken by a Gaussian elimination attack if e is sparse and the adversary gets enough samples. Or we can let s be long (and very sparse), but then A must be comparatively large, and if we make it sparse to reduce the description size of its rows, $A\cdot s+e$ is easily distinguished from random. In fact, any intermediate construction that interpolates between these two approaches falls to an appropriate combination of these simple linear attacks.

Our core observation is that we can defeat all such attacks by *simultaneously* implementing the above strategy of making x and s larger and more sparse at many different "levels of sparsity", and mixing the outcomes together. Concretely, fix some parameter D, and for $i = 1$ to D, let $x_i, e_i \in \{0,1\}^{w2^i}$ be random sparse vectors of density $1/2^i$, such that each x_i, e_i is twice larger than x_{i-1}, e_{i-1}. Then, consider the following candidate weak PRF family:

$$f_{e_1,\cdots,e_D}(x_1,\cdots,x_D) = \bigoplus_{i=1}^{D} \langle x_i, e_i \rangle.$$

(Note that when choosing x_i and e_i of regular structure, i.e. such that each length-w block contains exactly one non-zero noise coordinate, this corresponds to our main WPRF candidate of Eq. (1).)

In the above, each term $\langle x_i, e_i \rangle$ will play both the role of a sparse *noise term* for masking the (more dense) previous terms, and the role of a *parity term* for the next terms. That is, the candidate is obtained as follows: start from $A \cdot s + e$ with dense A and a *twice more sparse* e, then replace e by a term of the form $A' \cdot s' + e'$ where s', e', and the rows of A' are twice more sparse than s, e and the rows of A respectively, and iterate the processus. An adversary collecting many samples must therefore distinguish $\bigoplus_{i=1}^{D} H_i \cdot e_i$ from random given $(H_1, \cdots, H_D)$, where each H_i is a random sparse matrix of density $1/2^i$, containing many x_i samples as its rows. The sparser terms (with i close to D) defeat Gaussian attacks which attempt to find linear dependencies between the equations, since the corresponding secret vectors e_i involve a very large number of unknowns. On the other hand, the more dense terms (with i close to 1) guarantee that the output distribution of f will be dense, and not biased towards 0. More interestingly, as our analysis will show, *any* possible linear attack, which attempts to distinguish f from a random function by detecting a bias in its output, falls somewhere inbetween these two extremal types of linear attacks, and is defeated by one of the $H_i \cdot e_i$ terms, for at least one $i \in [1, D]$. Furthermore, since each x_i, e_i has density $1/2^i$ and length $O(2^i)$, they can all be generated from short random strings. This generation process is extremely simple, and (for x_i, e_i of regular structure) will put our candidate weak PRF family into a very low complexity class: the class $\mathsf{XOR} \circ \mathsf{AND}$ of polynomial-size depth-2 circuits consisting of arbitrary negations of the inputs, followed by a layer of AND gates at the bottom and a single XOR gate at the top. This is about the simplest strict subclass of $\mathsf{AC}^0[\oplus]$ which can possibly contain weak PRFs.

VDLPN vs. standard LPN. The assumption underlying the security of the above candidate, which we study in this paper, is that $\bigoplus_{i=1}^{D} H_i \cdot e_i = [H_1 || \cdots || H_D] \cdot (e_1 // \cdots // e_D)$ ($||$ denotes horizontal concatenation, and $//$ vertical concatenation) cannot be distinguished from random. The standard LPN assumption can be formulated in two ways: the *primal* formulation states that $As+e$ is indistinguishable from random, given a random expanding matrix A; the equivalent *dual* formulation states that $H \cdot e$ is indistinguishable from random, given a random *compressing* matrix H. Many variants of this assumption exist, which change the distribution of the rows of H (e.g. to be slightly more sparse [MTSB13] and/or quasi-cyclic [MBD+18]) and of the noise vector e (e.g. to have a regular structure [AFS03]). All these variants follow the general template of postulating the hardness of dual LPN when the rows of H and the noise vector are sampled from some distributions $(\mathcal{D}_{\mathsf{row}}, \mathcal{D}_{\mathsf{noise}})$. However, all known variants rely on pairs of distributions $(\mathcal{D}_{\mathsf{row}}, \mathcal{D}_{\mathsf{noise}})$ which output vectors of fixed density.

In contrast, our weak PRF candidate also builds upon a dual-LPN-style assumption, with a matrix $H = [H_1 || \cdots || H_D]$ and a noise vector $e = (e_1 // \cdots // e_D)$, but where the corresponding distributions $(\mathcal{D}_{\mathsf{row}}, \mathcal{D}_{\mathsf{noise}})$ have both *variable density*: vectors sampled from these distributions are divided in D blocks of increasingly smaller density. This variable density structure is the key to allow simultaneously for exponentially many samples, while maintaining a polynomial-size compressed representation of the exponentially long vectors. We call this assumption the

variable-density learning parity with noise assumption (VDLPN), and initiate its study in this paper.

2.2 Our Approach – the Low-Bias Function Viewpoint

For an alternative perspective, we can view our candidate as a methodology for obtaining a family of functions in a simple complexity class and with *low bias*, meaning that it provably resists a class of linear attacks. Below, we give some background on this approach, as well as some intuition behind why our candidate has low bias.

A large body of work has been dedicated to the construction of pseudorandom *generators* in low complexity classes. A common strategy is to PRGs which *unconditionally* resist restricted class of attacks, and to use them as plausible heuristic candidates for cryptographically secure PRGs. Perhaps the most representative example is the design of ε-*biased* PRGs, which unconditionally resist all linear distinguishers (see e.g. [NN90, MST03, LRTV09, Shp09, Vio10, MRRR14, LV17, AK19] and references therein). For the reader familiar with the literature on ε-biased PRG, we find it useful to provide an alternative interpretation of our approach as a natural approach to design an unconditionally secure low-bias weak pseudorandom function family (which we then conjecture to be also a cryptographically secure weak PRF family).

An ε-biased PRG is a function $G : \{0,1\}^n \mapsto \{0,1\}^m$ which maps a short seed $s \in \{0,1\}^n$ to a longer string $G(s)$, such that no linear function L has advantage more than ε in distinguishing $G(U_n)$ from U_m (where U_n, U_m denote the uniform distributions over $\{0,1\}^n$ and $\{0,1\}^m$ respectively). A standard strategy to build a low-bias PRG, used for example in [MST03, Shp09], is to design two generators, one secure against "light tests" (linear tests of small Hamming weight), and one secure against "heavy tests" (linear tests of large Hamming weight). Then, the PRG G is constructed as

$$G(x, y) = G_{\mathsf{light}}(x) \oplus G_{\mathsf{heavy}}(y);$$

it is relatively easy to show that this gives a low-bias PRG since G inherits the security against all possible linear tests from its components.

Our candidate WPRF can be seen as following a generalization of this approach. We define an (ε, N)-biased WPRF to be a family $\{f_k : \{0,1\}^n \mapsto \{0,1\}\}_k$ of functions such that the restriction of the function $k \to (f_k(x))_{x\in\{0,1\}^n}$ to a random size-N subet of its outputs is an ε-biased pseudorandom generator with very high probability.

The work of Naor and Naor [NN90] shows that to build low-bias sample spaces, it is useful to restrict our attention to tests whose weight belongs to an interval of the form $[2^i, 2^{i+1}]$, because such tests are provably fooled by random sparse vectors of density $1/2^i$. Building upon this observation, we show how to get a low-complexity weak PRF that fools linear tests with weight in $[2^i, 2^{i+1}]$: we let the random inputs to the PRF define the rows of a random matrix H_i of density $1/2^i$. Using a Chernoff-type concentration bound for random variables with limited dependency, we prove that any given test with weight in $[2^i, 2^{i+1}]$ is fooled by a constant fraction c of the columns of H_i. Therefore, the distribution induced by $H_i \cdot e_i$ for a random sparse vector e_i of weight w (i.e., a random subset-sum of columns of H_i) fools any given test with weight in $[2^i, 2^{i+1}]$ with probability at least c^w. We let f_{i,e_i} be the function which, on input x, samples a $1/2^i$-dense row h of H_i using randomness x, and outputs $h^\intercal \cdot e_i$ (by our argument, f_{i,e_i} has low bias against all tests with weight in $[2^i, 2^{i+1}]$). Finally, to get a 2^D-sample WPRF which has low bias with respect to all possible linear tests, we define

$$f_{e_1 \cdots e_D}(x_1, \cdots, x_D) = f_{1,e_1}(x_1) \oplus \cdots \oplus f_{D,e_D}(x_D).$$

Boolean circuit formulation As presented earlier in (1), we can also obtain the equivalent Boolean circuit formulation:

$$f_k(x) = \bigoplus_{i=1}^{D} \bigoplus_{j=1}^{w} \bigwedge_{h=1}^{i} (x_{i,j,h} \oplus k_{i,j,h}), \tag{3}$$

where $x, k \in \{0,1\}^{wD(D-1)/2}$ are parsed as D sequences such that the i-th sequence, $1 \le i \le D$ is made up of i blocks of i bits. To see that this is equivalent to the VDLPN formulation above, observe that each block of i bits defines a binary unit vector of length 2^i and $\bigwedge_{h=1}^{i}(x_{i,j,h} \oplus k_{i,j,h}) = 1$ exactly when the inner product of the associated binary vectors in the key and in the input is 1. This formulation is conceptually simpler, and highlights that for a fixed key k, it corresponds to an XNF formula, namely a depth-2 circuit computing a single XOR of ANDs of literals (inputs or their negations). In the section below, we also mention several variants of the construction based on this boolean circuit perspective.

2.3 On the Security of Our WPRF Candidates

We conjecture that when instantiating our WPRF candidates with $w = D = \lambda$, every $2^{o(\lambda)}$-size adversary can get a distinguishing advantage of at most $2^{-\Omega(\lambda)}$. Since our main candidate has inputs of size $n = O(wD^2) = O(\lambda^3)$, this corresponds to security against $2^{o(n^{1/3})}$-size adversaries. For the variant with inputs of size $n = O(wD) = O(\lambda^2)$, this corresponds to security against $2^{o(\sqrt{n})}$-size adversaries, which is essentially optimal due to the existence of a $2^{\tilde{O}(\sqrt{n})}$-time (and -size) learning algorithm for XNF with arbitrary distributions [HS07]. When setting $D = \mathsf{polylog}(\lambda)$, $w = \lambda$, and restricting the adversary to polynomially many samples, we conjecture quasiexponential security against adversaries of size $2^{o(n/\mathsf{polylog}(n))}$.

Of course, when defining low-complexity cryptographic primitives, one must be especially careful with security analysis. For example, the candidate WPRF construction in $\mathsf{AC}^0[\oplus]$ from [ABG+14], which initially seemed to have plausible near-exponential security, was shown by Bogdanov and Rosen to be broken in quasipolynomial time via an algebraic "relinearization" attack [BR17].

Here we *prove* resistance against this and a wide range of other attacks considered in the literature. In particular, we identify a large variety of attacks on LPN as special cases of *linear distinguishers*, as described above (Section 2.2), and provably rule them out for our main candidate, as well as an intermediate variant that has $O(wD)$ input size (but bigger key size compared to our most "FSS-friendly" candidate). We observe that Gaussian elimination attacks [EKM17], statistical decoding attacks [AJ01, Ove06, FKI06, DAT17, Zic17], information set decoding attacks [Pra62, Ste88, FS09, BLP11, MMT11, BJMM12, MO15], BKW and variants [BKW00, Lyu05] all fall under this umbrella of a linear attack. We also rule out general algebraic attacks, which covers the attack on the Akavia et al. [ABG+14] candidate, and statistical query attacks based on the learning algorithm of Linial, Mansour and Nisan [LMN89], and prove resistance to linear cryptanalysis as formalized by Miles and Viola [MV11].

For all of the classes of attacks above, we prove that no $2^{O(w)}$-size adversary mounting an attack from one of these classes can have advantage more than $2^{-O(w)}$ in distinguishing these candidates from random functions given 2^D samples. Further, we show our construction is (w/D)-wise independent , so resists all AC^0 tests of size $2^{(w/D)^c}$ for some constant c.

Generalization to arbitrary rings. The VDLPN candidate can be generalized to work over larger rings than $\mathbb{Z}_2$, as detailed in Section 6.3. Here, we simply modify the VDLPN distributions by replacing ones with random non-zero elements from a ring R, which gives a candidate that is still FSS-friendly, and can be used for PCFs that output correlations over R. For an appropriate choice of parameters, our proof of resistance to linear attacks also extends to this arithmetic

setting, and we conjecture its security against general attacks for an arbitrary choice of the ring R.

Variants with smaller inputs and smaller keys. Taking the boolean function definition seen in equation (3), we can consider several variants of the construction by making simplifications. For instance, our first variant reduces the input size from $O(wD^2)$ to $O(wD)$ bits by reusing inputs, replacing the variable $x_{i,j,h}$ with $x_{j,h}$. This modified candidate still provably resists linear attacks, and we conjecture it also resists the other attacks we have considered. We can further modify this variant by XORing an additional triangular function to the weak PRF. Since triangular functions have high algebraic immunity, this allows us to prove resistance to algebraic attacks whilst retaining the benefit of the reduced $O(wD)$ input size.

Finally, we present an aggressive variation which reduces the key size from $O(wD^2)$ down to $O(wD)$ bits. Here, the proofs of resistance for linear and algebraic attacks break down, however, we have not found any attacks, and put forward the security of this variant as an interesting direction for future study.

2.4 Organization

The remainder of the paper is organized as follows. After some preliminaries in Section 3, we first give definitions of PCFs in Section 4. We then present generic constructions of PCFs, based on function secret sharing for WPRF in Section 5. Section 6 describes our main WPRF candidate, explains why it is FSS-friendly and how it implies XOR-RKA secure PRGs and WPRFs. Section 7 analyzes its security, and Section 8 describes our variants. In Section 9 we propose concrete parameters based on concrete attacks. Finally, in Section 10 we describe further details of the applications of our PCFs.

3 Preliminaries

3.1 Notation

Given a field $\mathbb{F}$ and a number $p \in [0,1]$, we let $\mathsf{Ber}_p(\mathbb{F})$ denote the Bernoulli distribution over $\mathbb{F}$ which returns a uniformly random element of $\mathbb{F}$ with probability p, and 0 with probability $1-p$. When the field is not specified, we assume $\mathbb{F} = \mathbb{F}_2$ by default. Given a vector $\vec{v}$, we denote by $\mathcal{HW}(\vec{v})$ the number of nonzero entries of $\vec{v}$. Given two integers (m, n) with $0 \leq m \leq n$, we let $\mathcal{S}_{m,n}(\mathbb{F})$ denote the uniform distribution over the set $\{\vec{v} \in \mathbb{F}^n : \mathcal{HW}(\vec{v}) = m\}$. Here again, we assume $\mathbb{F} = \mathbb{F}_2$ when the field is not specified. Given matrices A, B, we denote $A//B$ their vertical concatenation and $A||B$ their horizontal concatenation (assuming the dimensions match).

3.2 Preliminaries on Bias

Definition 3.1 (Bias of a Distribution) *Given a distribution $\mathcal{D}$ over $\mathbb{F}_2^n$ and a vector $\vec{u} \in \mathbb{F}_2^n$, the bias of $\mathcal{D}$ with respect to $\vec{u}$, denoted $\mathsf{bias}_{\vec{u}}(\mathcal{D})$, is equal to*

$$\mathsf{bias}_{\vec{u}}(\mathcal{D}) = \left| \frac{1}{2} - \Pr_{\vec{v} \xleftarrow{\$} \mathcal{D}} [\vec{u}^\intercal \cdot \vec{v} = 1] \right|.$$

The bias of $\mathcal{D}$, denoted $\mathsf{bias}(\mathcal{D})$, is the maximum bias of $\mathcal{D}$ with respect to any nonzero vector $\vec{u}$:

$$\mathsf{bias}(\mathcal{D}) = \max_{\vec{u} \neq 0^n} \mathsf{bias}_{\vec{u}}(\mathcal{D}).$$

We recall a number of useful facts about bias. First, observe that when $\mathcal{D}$ is the uniform distribution over a multiset S, it holds that

$$\mathsf{bias}_{\vec{u}}(\mathcal{D}) = \left| \frac{1}{2} - \frac{1}{|S|} \sum_{\vec{v} \in S} \vec{u}^{\mathsf{T}} \cdot \vec{v} \right|.$$

Given t distributions $(\mathcal{D}_1, \cdots, \mathcal{D}_t)$ over $\mathbb{F}_2^n$, we denote by $\bigoplus_{i \leq t} \mathcal{D}_i$ the distribution obtained by *independently* sampling $\vec{v}_i \xleftarrow{\$} \mathcal{D}_i$ for $i = 1$ to t and outputting $\vec{v} \leftarrow \vec{v}_1 \oplus \cdots \oplus \vec{v}_t$.

We will use the following fact about the bias of the exclusive-or of independent distributions (cf. [Shp09]).

Lemma 3.2 *Let $t \in \mathbb{N}$ be an integer, and let $(\mathcal{D}_1, \cdots, \mathcal{D}_t)$ be t independent distributions over $\mathbb{F}_2^n$. Then*

$$\mathsf{bias}\left(\bigoplus_{i \leq t} \mathcal{D}_i \right) \leq 2^{t-1} \cdot \prod_{i=1}^{t} \mathsf{bias}(\mathcal{D}_i) \leq \min_{i \leq t} \mathsf{bias}(\mathcal{D}_i).$$

3.3 Concentration Bounds

We recall several standard concentration bounds from the literature.

Lemma 3.3 (Bienaymé-Chebyshev Inequality) *Let X be a random variable with finite expected value μ and finite nonzero variance σ^2. Then for any $k > 0$,*

$$\Pr[|X - \mu| \geq k\sigma] \leq \frac{1}{k^2}.$$

Lemma 3.4 (Chernoff Inequality) *Let $n \in \mathbb{N}$ and let $(X_1, \cdots, X_n)$ be independent random variables taking values in $\{0, 1\}$. Let X denote their sum and $\mu \leftarrow \mathbb{E}[X]$. Then for any $\delta \in [0, 1]$,*

$$\Pr[X \geq (1 + \delta)\mu] \leq \exp\left(-\frac{\delta^2 \mu}{3} \right).$$

Bounded Difference Inequality. Eventually, we will need a Chernoff-type concentration bound for dependent random variables. The bounded difference inequality was first proved by McDiarmid in [McD89]. It is an application of the more general Azuma inequality [Azu67] which provides a powerful generalization of the Chernoff-Hoeffding inequality to martingales satisfying some bounded difference condition. Below, we state the bounded difference inequality in a form less general than the one proven in [McD89], which suffices for our purpose. Let $(n, m) \in \mathbb{N}^2$ be two integers. We say that a function $\Phi : [n]^m \mapsto \mathbb{R}$ satisfies the *Lipschitz property with constant d* if for every $\vec{x}, \vec{x}' \in [n]^m$ which differ in a single coordinate, it holds that

$$|\Phi(\vec{x}) - \Phi(\vec{x}')| \leq d.$$

Lemma 3.5 (Bounded Difference Inequality) *Let $\Phi : [n]^m \mapsto \mathbb{R}$ be a function satisfying the Lipschitz property with constant d, and let $(X_1, \cdots, X_m)$ be independent random variables over $[n]$. Then*

$$\Pr[\Phi(X_1, \cdots, X_m) < \mathbb{E}[\Phi(X_1, \cdots, X_m)] - t] \leq \exp\left(-\frac{2t^2}{m \cdot d^2} \right).$$

3.4 Weak Pseudorandom functions

We consider here *weak* PRFs, which relax standard PRFs by only considering distinguishers that get the outputs of the function on uniformly random inputs. We require *subexponential* security by default, namely security against distinguishers of size 2^{n^ϵ} for some $\epsilon > 0$. This is the typical level of security achieved by constructions based on the strongest plausible versions of standard cryptographic assumptions. We formally define this notion below.

Definition 3.6 ((Weak) pseudorandom function [GGM84, NR95]) *Let $\lambda \in \mathbb{N}$ denote a security parameter and $n = n(\lambda)$, $\kappa = \kappa(\lambda)$ be monotonically-increasing and polynomially-bounded input length and key length functions, respectively. A (weak)* pseudorandom function *is syntactically defined by a function family $\mathcal{F} = \{F_\lambda \colon \{0,1\}^\kappa \times \{0,1\}^n \to \{0,1\}\}$, where the output $F_\lambda(K, x)$ can be computed from (K, x) in polynomial time. Since λ and κ are determined by the input length n, we will sometimes write $F_K(x)$ instead of $F_\lambda(K, x)$. For $T = T(\kappa)$ and $\varepsilon = \varepsilon(\kappa)$, we say that $\mathcal{F}$ is a (T, ε)-*secure strong pseudorandom function (PRF)*, if for every $\lambda \in \mathbb{N}$ and every oracle circuit $\mathcal{D}$ of size $T(\kappa)$, it holds*

$$\Pr_k[\mathcal{D}^{f_k(\cdot)} = 1] - \Pr_R[\mathcal{D}^{R(\cdot)} = 1] \leq \varepsilon(\kappa),$$

where $\kappa = \kappa(\lambda)$, $k \xleftarrow{\$} \{0,1\}^\kappa$ is chosen at random, and $R \colon \{0,1\}^n \to \{0,1\}$ is a truly random function. A T-secure PRF is a $(T, 1/T)$-secure PRF.

We say that $\mathcal{F}$ is a (T, ε)-secure weak PRF (WPRF) *or T-secure WPRF if the above holds when $\mathcal{D}$ only gets access to samples $(x_i, f_k(x_i))$, where $x_i \xleftarrow{\$} \{0,1\}^n$ are chosen uniformly and independently. Finally, we say that a (W)PRF $\mathcal{F}$ has* polynomial *security if it is T-secure for every polynomial T, and that it has* subexponential *(resp., quasipolynomial, exponential) security if there exists $c > 0$ such that it is T-secure for $T = 2^{\kappa^c}$ (resp., $T = \kappa^{\log^c \kappa}$, $T = 2^{\kappa c}$). In this work, when we refer to a (weak) PRF without specifying the level of security, we assume subexponential security by default.*

4 Defining Pseudorandom Correlation Functions

In this section we formally define the notion of pseudorandom correlation functions (PCFs), the new primitive studied in this work. At a high-level, a PCF extends the previous notion of a pseudorandom correlation generator (PCG) [BCG+19b] analogously to the way a pseudorandom function (PRF) extends a pseudorandom generator (PRG).

Similarly to the notion of a PCG, a PCF should securely realize some ideal target correlation. A simple example for a useful ideal correlation is a random bit-oblivious-transfer (OT) correlation, defined by a pair of random variables (Y_0, Y_1) such that $Y_0 = (s_0, s_1)$ is uniform over $\{0,1\}^2$ and $Y_1 = (c, s_c)$ for a random bit c. This correlation is "finite" in the sense that it has a fixed output length.

It is often useful to consider not only finite correlations, but also infinite families of finite correlations where the output length can grow with the security parameter λ. In order to define a meaningful notion of PCF for such a family of correlations, we require the correlation to satisfy the following "reverse sampleability" property: There exists an efficient algorithm that, given an output y_σ in the support of Y_σ, reverse-samples the other output $y_{1-\sigma}$ from the right conditional distribution, namely $[Y_{1-\sigma} | Y_\sigma = y_\sigma]$. We formalize this below.

Definition 4.1 (Reverse-sampleable correlation) *Let $1 \leq \ell_0(\lambda), \ell_1(\lambda) \leq \mathsf{poly}(\lambda)$ be output-length functions. Let $\mathcal{Y}$ be a probabilistic algorithm that on input 1^λ returns a pair of outputs $(y_0, y_1) \in \{0,1\}^{\ell_0(\lambda)} \times \{0,1\}^{\ell_1(\lambda)}$, defining a correlation on the outputs.*

We say that $\mathcal{Y}$ defines a reverse-sampleable correlation*, if there exists a probabilistic polynomial time algorithm* RSample *that takes as input $1^\lambda, \sigma \in \{0,1\}$ and $y_\sigma \in \{0,1\}^{\ell_\sigma(\lambda)}$, and outputs*

$\underline{\mathsf{Exp}^{\mathsf{pr}}_{\mathcal{A},N,0}(\lambda)}$:
for $i = 1$ **to** $N(\lambda)$**:**
$x^{(i)} \xleftarrow{\$} \{0,1\}^{n(\lambda)}$
$(y_0^{(i)}, y_1^{(i)}) \leftarrow \mathcal{Y}(1^\lambda)$
$b \leftarrow \mathcal{A}(1^\lambda, (x^{(i)}, y_0^{(i)}, y_1^{(i)})_{i\in[N(\lambda)]})$
return b

$\underline{\mathsf{Exp}^{\mathsf{pr}}_{\mathcal{A},N,1}(\lambda)}$:
$(\mathsf{k}_0, \mathsf{k}_1) \leftarrow \mathsf{PCF.Gen}(1^\lambda)$
for $i = 1$ **to** $N(\lambda)$**:**
$x^{(i)} \xleftarrow{\$} \{0,1\}^{n(\lambda)}$
for $\sigma \in \{0,1\}$**:** $y_\sigma^{(i)} \leftarrow \mathsf{PCF.Eval}(\sigma, \mathsf{k}_\sigma, x^{(i)})$
$b \leftarrow \mathcal{A}(1^\lambda, (x^{(i)}, y_0^{(i)}, y_1^{(i)})_{i\in[N(\lambda)]})$
return b

Figure 1: Pseudorandom $\mathcal{Y}$-correlated outputs of a PCF.

$y_{1-\sigma} \in \{0,1\}^{\ell_{1-\sigma}(\lambda)}$, *such that for all* $\sigma \in \{0,1\}$ *the following distributions are statistically close:*

$$\{(y_0, y_1) \mid (y_0, y_1) \xleftarrow{\$} \mathcal{Y}(1^\lambda)\} \quad \textit{and}$$
$$\{(y_0, y_1) \mid (y_0', y_1') \xleftarrow{\$} \mathcal{Y}(1^\lambda), y_\sigma \leftarrow y_\sigma', y_{1-\sigma} \leftarrow \mathsf{RSample}(1^\lambda, \sigma, y_\sigma)\}$$

We would typically like a PCF for a given correlation, say an OT correlation, to output for each input an independent pair sampled freshly from the correlation. However, we will sometimes be interested in PCFs that also have correlations *across* different inputs. For instance, in a vector-oblivious-linear-evaluation (VOLE) or authenticated multiplication triples correlation there is a secret multiplier that is common to all inputs. To capture this more general case, we introduce a reverse sampleable correlation with setup, which allows all algorithms to depend on a fixed global secret, ensuring consistency across different invocations.

Remark 4.2 *A* reverse-sampleable correlation with setup *consists of an additional algorithm* Setup *that on input* 1^λ *samples a master key* mk*, which the algorithms* $\mathcal{Y}$, $\mathsf{RSample}$ *in Definition 4.1 take as extra input. The reverse sampling property is then required to hold for all* mk *in the image of* Setup*. For a formal definition we refer to Definition D.1 in Appendix B.*

We are now ready to formalize our main notions of PCF. We start with our default notion of *weak* PCF, in which security holds for randomly chosen inputs, and then define the notion of strong PCF. As is often done in the context of PRFs, it will be convenient to consider not a single function defined over $\{0,1\}^*$ but an infinite family of finite functions parameterized by a security parameter λ, where the input length of the λ-th function is polynomial in λ. For simplicity, in the following we focus on correlations without setup, for the full definitions we refer to Appendix B.

Definition 4.3 (Pseudorandom correlation function (PCF)) *Let* $\mathcal{Y}$ *be a reverse-sampleable correlation with output length functions* $\ell_0(\lambda), \ell_1(\lambda)$ *and let* $\lambda \le n(\lambda) \le \mathsf{poly}(\lambda)$ *be an input length function. Let* $(\mathsf{PCF.Gen}, \mathsf{PCF.Eval})$ *be a pair of algorithms with the following syntax:*

- $\mathsf{PCF.Gen}(1^\lambda)$ *is a probabilistic polynomial time algorithm that on input* 1^λ*, outputs a pair of keys* $(\mathsf{k}_0, \mathsf{k}_1)$*; we assume that* λ *can be inferred from the keys.*
- $\mathsf{PCF.Eval}(\sigma, \mathsf{k}_\sigma, x)$ *is a deterministic polynomial-time algorithm that on input* $\sigma \in \{0,1\}$*, key* k_σ *and input value* $x \in \{0,1\}^{n(\lambda)}$*, outputs a value* $y_\sigma \in \{0,1\}^{\ell_\sigma(\lambda)}$*.*[7]

We say $(\mathsf{PCF.Gen}, \mathsf{PCF.Eval})$ *is a (weak)* (N, B, ε)-secure pseudorandom correlation function (PCF) for $\mathcal{Y}$*, if the following conditions hold:*

[7]Note that it would be sufficient for $\mathsf{PCF.Eval}$ to take as input k_σ and x by appending σ to the key k_σ. This corresponds to the view of a PCF as a single keyed function.

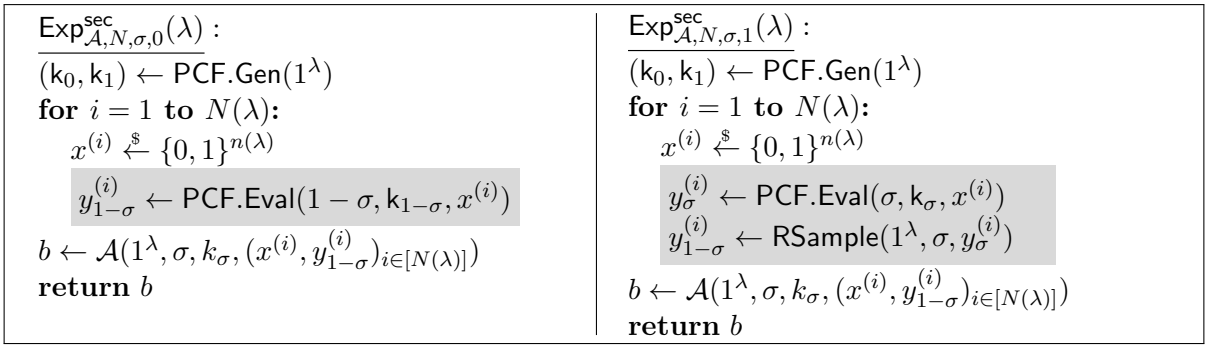

Figure 2: Security of a PCF. Here, RSample is the algorithm for reverse sampling $\mathcal{Y}$ as in Definition 4.1.

- ***Pseudorandom $\mathcal{Y}$-correlated outputs.*** *For every $\sigma \in \{0,1\}$ and non-uniform adversary $\mathcal{A}$ of size $B(\lambda)$, it holds*

$$\left|\Pr[\mathsf{Exp}^{\mathsf{pr}}_{\mathcal{A},N,0}(\lambda)=1]-\Pr[\mathsf{Exp}^{\mathsf{pr}}_{\mathcal{A},N,1}(\lambda)=1]\right| \le \varepsilon(\lambda)$$

  for all sufficiently large λ, where $\mathsf{Exp}^{\mathsf{pr}}_{\mathcal{A},N,b}(\lambda)$ for $b \in \{0,1\}$ is as defined in Figure 1. In particular, the adversary is given access to $N(\lambda)$ samples.

- ***Security.*** *For each $\sigma \in \{0,1\}$ and non-uniform adversary $\mathcal{A}$ of size $B(\lambda)$, it holds*

$$\left|\Pr[\mathsf{Exp}^{\mathsf{sec}}_{\mathcal{A},N,\sigma,0}(\lambda)=1]-\Pr[\mathsf{Exp}^{\mathsf{sec}}_{\mathcal{A},N,\sigma,1}(\lambda)=1]\right| \le \varepsilon(\lambda)$$

  for all sufficiently large λ, where $\mathsf{Exp}^{\mathsf{sec}}_{\mathcal{A},N,\sigma,b}(\lambda)$ for $b \in \{0,1\}$ is as defined in Figure 2 (again, with $N(\lambda)$ samples).

We say that $(\mathsf{PCF.Gen},\mathsf{PCF.Eval})$ *is a* PCF for $\mathcal{Y}$ *if it is a $(p,1/p,p)$-secure PCF for $\mathcal{Y}$ for every polynomial p. If $B=N$, we will write (B,ε)-secure PCF for short.*

Roughly speaking, the above security notion of PCF extends the corresponding security notion for PCGs (see Definitions A.1, A.3 in Appendix A) as follows: $(\mathsf{PCF.Gen},\mathsf{PCF.Eval})$ is a PCF for $\mathcal{Y}$, if for every polynomial $N=N(\lambda)$, and random ensemble of sets $\mathcal{X}=(X_\lambda)_{\lambda\in\mathbb{N}}$, where $X_\lambda=(x_1,\ldots,x_{N(\lambda)}) \subseteq \{0,1\}^{n(\lambda)}$, the pair of algorithms $\mathsf{PCG}=(\mathsf{PCG.Gen},\mathsf{PCG.Expand})$ defined as

- $\mathsf{PCG.Gen}(1^\lambda)$ samples $(\mathsf{k}_0,\mathsf{k}_1) \leftarrow \mathsf{PCF.Gen}(1^\lambda)$.
- $\mathsf{PCG.Expand}(\sigma,\mathsf{k}_\sigma)$ computes $y^{(i)}_\sigma \leftarrow \mathsf{PCF.Eval}(\sigma,\mathsf{k}_\sigma,x^{(i)})$ for all $i \in [N(\lambda)]$ and returns $(y^{(i)}_\sigma)_{i\in[N(\lambda)]}$.

is a PCG for $\mathcal{Y}$ with overwhelming probability over the choice of $\mathcal{X}$.[8] .

We turn to define the strong notion of PCF, where pseudorandomness of outputs and security holds for adversarially chosen queries.

Definition 4.4 (Strong pseudorandom correlation function (sPCF)) *Let $\mathcal{Y}$ and* $(\mathsf{PCF.Gen},\mathsf{PCF.Eval})$ *be as in Definition 4.3. We say that* $(\mathsf{PCF.Gen},\mathsf{PCF.Eval})$ *is an* strong (N,B,ε)-secure PCF (sPCF) for $\mathcal{Y}$ *if the following conditions hold:*

[8] More precisely, PCG is a PCG for the correlation generator $\mathcal{Y}^N$, where $\mathcal{Y}^N$ on input 1^λ returns N independent samples from $\mathcal{Y}$.

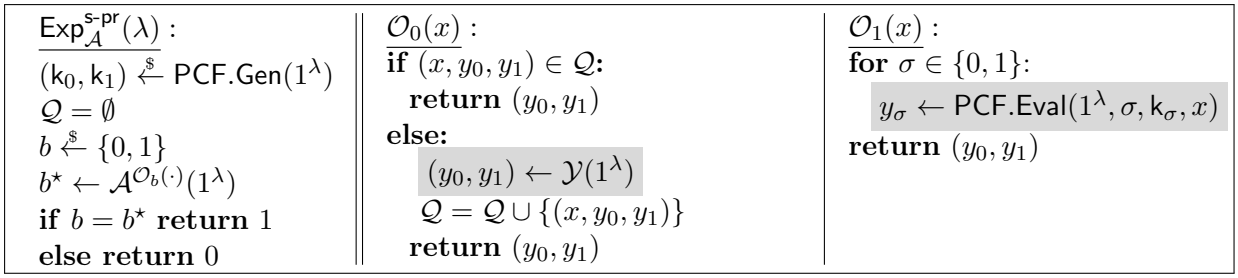

Figure 3: Strong pseudorandom $\mathcal{Y}$-correlated outputs of a PCF.

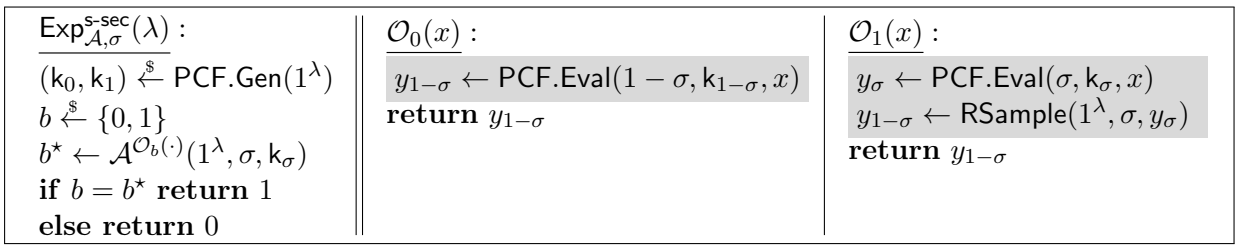

Figure 4: Strong security of a PCF. Here, RSample is the algorithm for reverse sampling $\mathcal{Y}$ according to Definition 4.1.

- ***Strong pseudorandom $\mathcal{Y}$-correlated outputs.*** *For every non-uniform adversary $\mathcal{A}$ of size $B(\lambda)$ asking at most $N(\lambda)$ queries to the oracle $\mathcal{O}_b(\cdot)$, it holds*

$$\left|\Pr[\mathsf{Exp}^{\mathsf{s\text{-}pr}}_{\mathcal{A}}(\lambda) = 1] - \frac{1}{2}\right| \leq \varepsilon(\lambda)$$

for all sufficiently large λ, where $\mathsf{Exp}^{\mathsf{s\text{-}pr}}_{\mathcal{A}}(\lambda)$ is as defined in Figure 3.

- ***Strong security.*** *For all $\sigma \in \{0,1\}$ and non-uniform adversaries $\mathcal{A}$ of size $B(\lambda)$ asking at most $N(\lambda)$ queries to the oracle $\mathcal{O}_b(\cdot)$, it holds*

$$\left|\Pr[\mathsf{Exp}^{\mathsf{s\text{-}sec}}_{\mathcal{A},\sigma}(\lambda) = 1] - \frac{1}{2}\right| \leq \varepsilon(\lambda)$$

for all sufficiently large λ, where $\mathsf{Exp}^{\mathsf{s\text{-}sec}}_{\mathcal{A},\sigma}(\lambda)$ for $b \in \{0,1\}$ is as defined in Figure 4.

4.1 From Weak to Strong PCFs

Assuming access to a random oracle we can transform a weak PCF into a strong PCF as described in the following.

Theorem 4.5 *Let $\mathsf{PCF} = (\mathsf{PCF.Gen}, \mathsf{PCF}, \mathsf{Eval})$ be an (N, B, ε)-secure PCF for correlation $\mathcal{Y}$ with input length $n(\lambda)$, and let $\mathsf{H}\colon \{0,1\}^{n'(\lambda)} \to \{0,1\}^{n(\lambda)}$ be a hash function modeled as a programmable random oracle. Then $\mathsf{PCF}' = (\mathsf{PCF.Gen}', \mathsf{PCF}, \mathsf{Eval}')$ as defined in Figure 5 is a strong (N, B', ε)-secure PCF for correlation $\mathcal{Y}$ with input length $n'(\lambda)$ for any $B'(\lambda) \leq N(\lambda)$.*

Proof. Let $\mathcal{A}$ be an (N, B', ε)-adversary on the strong pseudorandomness of outputs of PCF'. We construct an adversary $\mathcal{B}$ on the pseudorandomness of outputs of PCF as follows: The adversary $\mathcal{B}$ obtains $(x^{(i)}, y_0^{(i)}, y_1^{(i)})_{i \in [N(\lambda)]}$ via its experiment. Without loss of generality we assume that $\mathcal{A}$ always queries the random oracle before asking a query to the experiment oracles. On the i-th query of $\mathcal{A}$ to H, $\mathcal{B}$ replies $x^{(i)}$. Note that the number of oracle queries of $\mathcal{B}$ is upper bounded by its size $B'(\lambda) \leq N(\lambda)$. On an oracle query x, $\mathcal{B}$ returns $(y_0^{(i)}, y_1^{(i)})$, where i is such that $x^{(i)} = x$. Finally, $\mathcal{B}$ forwards the output of $\mathcal{A}$ to its own experiment. Then, the probability of

Transformation from weak to strong PCF

- $\mathsf{PCF.Gen}'(1^\lambda)$: On input 1^λ, return $(\mathsf{k}_0, \mathsf{k}_1) \leftarrow \mathsf{PCF.Gen}(1^\lambda)$.
- $\mathsf{PCF.Eval}'(\sigma, \mathsf{k}_\sigma, x)$: On input $\sigma \in \{0,1\}$, key k_σ and input value $x \in \{0,1\}^{n'(\lambda)}$, return $y_\sigma \leftarrow \mathsf{PCF.Eval}(\sigma, \mathsf{k}_\sigma, \mathsf{H}(x))$.

Figure 5: From weak to strong PCF.

success probability of $\mathcal{B}$ in distinguishing the experiments for $b = 0$ and $b = 1$ corresponds to the advantage of $\mathcal{A}$. Further, the size of $\mathcal{B}$ is upper bounded by $B'(\lambda) \leq N(\lambda)$ and thus upper bounded by $B(\lambda)$ (ignoring constant factors).

We omit the of security of PCF' as it is almost analogous to the above.

□

5 PCFs from Function Secret Sharing for a Weak PRF

In this section, we construct PCFs for various classes of correlations based on function secret sharing. Our constructions can be seen as generalizations and extensions of PCG constructions from previous works [BCGI18,BCG+19b]. These works are based on function secret sharing and the learning parity with noise assumption, whereas here we assume FSS for a class of functions defined by a weak pseudorandom function (WPRF). This gives a generic way to build a PCF, given any suitable function secret sharing scheme for a WPRF.

Function Secret Sharing. A (two-party) function secret sharing scheme [BGI15, BGI16b], or FSS for short, allows splitting a function $f : \{0,1\}^n \to \mathbb{G}$, where $\mathbb{G}$ is an Abelian group, into two functions f_0, f_1 that add up to f but individually hide f. Each share f_σ of f is described by a succinct key that enables its efficient evaluation on an input $x \in \{0,1\}^n$. We assume that f comes from a public function family $\mathcal{F}$, which is an infinite collection of function descriptions together with an efficient evaluation algorithm. We abuse notation and write $f \in \mathcal{F}$ to refer to both the function and its description.

Definition 5.1 (Function Secret Sharing) *A* function secret sharing *(FSS) scheme for function family $\mathcal{F} = \{f : \{0,1\}^n \to \mathbb{G}\}$, where $(\mathbb{G}, +)$ is a an Abelian group, is pair of PPT algorithms $\mathsf{FSS} = (\mathsf{FSS.Gen}, \mathsf{FSS.Eval})$ with the following syntax:*

- $\mathsf{FSS.Gen}(1^\lambda, f)$*, given security parameter λ and description of a function $f \in \mathcal{F}$, where $f : \{0,1\}^n \to \mathbb{G}$, outputs a pair of keys (K_0, K_1); we assume that $\lambda, n, \mathbb{G}$ are determined by each key.*
- $\mathsf{FSS.Eval}(\sigma, K_\sigma, x)$*, given a key K_σ for party $\sigma \in \{0,1\}$, and an input $x \in \{0,1\}^n$, outputs a group element $y_\sigma \in \mathbb{G}$.*

The scheme should satisfy the following requirements:

- **Correctness:** *For any $f \in \mathcal{F}$ with input domain $\{0,1\}^n$ and $x \in \{0,1\}^n$, we have*

$$\Pr\left[\sum_{\sigma\in\{0,1\}} \mathsf{FSS.Eval}(\sigma, K_\sigma, x) = f(x) \;\middle|\; (K_0, K_1) \leftarrow \mathsf{FSS.Gen}(1^\lambda, f)\right] = 1$$

- **Security:** *For any $\sigma \in \{0,1\}$, there exists a PPT simulator* Sim *such that for any sequence $f_\lambda \in \mathcal{F}$ of polynomial-size function descriptions, the distributions*

  $\{(\sigma, f_\lambda, K_\sigma) \mid (K_0, K_1) \leftarrow \mathsf{FSS.Gen}(1^\lambda, f_\lambda)\}$ *and* $\{(\sigma, f_\lambda, K_\sigma) \mid K_\sigma \leftarrow \mathsf{Sim}(1^\lambda, \mathsf{Leak}_\sigma(f_\lambda))\}$

  are computationally indistinguishable.

The default choice for the leakage functions is $\mathsf{Leak}_\sigma(f) = (n, \mathbb{G})$*, namely we allow leaking the input and output domains of* f*.*

Note that previous FSS definitions only consider a single leakage function Leak that applies to both parties. Here we will use asymmetric leakage, with distinct functions $\mathsf{Leak}_0, \mathsf{Leak}_1$, to capture constructions based on puncturable PRFs. When the leakage functions Leak_σ are efficiently invertible, which will always be the case in this work, one could replace the above simulation-based security definition by requiring indistinguishability of any pair of functions that have the same leakage.

Some of our constructions also require the following property, that outputs of the Eval algorithm on random inputs are pseudorandom. In other words, this means that $\mathsf{FSS.Eval}$ is a weak pseudorandom function for a suitable distribution of keys.

Definition 5.2 (FSS with Weak Pseudorandom Outputs) *We say that an FSS scheme* $(\mathsf{FSS.Gen}, \mathsf{FSS.Eval})$ *for* $\mathcal{F}$ *has* weak pseudorandom outputs *if for any* $\sigma \in \{0,1\}$*,* $N = \mathsf{poly}(\lambda)$*, and any sequence of polynomial-size function descriptions* $f_\lambda \in \mathcal{F}$*, with domain* $\{0,1\}^n$ *and range* $\mathbb{G} = \mathbb{G}(\lambda)$*, for* $n = n(\lambda) = \mathsf{poly}(\lambda)$ *and* $\log|\mathbb{G}(\lambda)| = \mathsf{poly}(\lambda)$*, the distribution*

$$\left\{(x^{(i)}, y^{(i)})_{i=1}^N \;\middle|\; (K_0, K_1) \leftarrow \mathsf{FSS.Gen}(1^\lambda, f_\lambda), x^{(i)} \leftarrow \{0,1\}^n, y^{(i)} \leftarrow \mathsf{FSS.Eval}(\sigma, K_\sigma, x^{(i)})\right\}_{\lambda\in\mathbb{N}}$$

is computationally indistinguishable from the uniform distribution on $(\{0,1\}^n \times \mathbb{G})^N$*.*

Boyle et al. [BGI15] showed that FSS schemes for certain classes of 'poly-spanning' function families have pseudorandom outputs. This does not cover the classes of functions we are interested in. However, any FSS scheme can be easily modified to have weak pseudorandom outputs using a weak PRF $\{F_k : \{0,1\}^n \to \mathbb{G}\}_k$, by having each party add or subtract $F_k(x)$ when running $\mathsf{FSS.Eval}$ on input x. Note that our concrete instantiations described in Section 6.4 already have weak pseudorandom outputs, without this modification.

5.1 PCF for Vector Oblivious Linear Evaluation

Vector oblivious linear evaluation, or VOLE, is a correlation over a ring R where each pair of samples can be seen as an additive secret sharing of one component in a scalar-vector product. Concretely, define the ideal VOLE correlation over R by the pair of randomized algorithms $(\mathsf{Setup}, \mathcal{Y})$, where Setup samples a uniformly random $a \in R$, and $\mathcal{Y}(1^\lambda, a)$ outputs a pair $(a, c_0), (b, c_1)$, where b, c_0 are uniform in R, and $c_1 = c_0 + a \cdot b$. The VOLE correlation can be used in secure two-party computation tasks such as secure linear algebra, private keyword search, and computationally efficient zero-knowledge proofs [ADI+17, BCGI18, WYKW20].

In Fig. 6, we give a simple construction of a PCF for VOLE, from any function secret sharing scheme for scalar multiples of a WPRF family. We remark that for our concrete instantiations given later, FSS for this function family can be obtained with a small tweak to the basic construction of FSS for the WPRF. In the construction, one party is given the WPRF key, while the other party gets the secret scalar $a \in R$, and each party additionally gets an FSS share of a multiplied by the WPRF. This allows an output to be computed with one call to $\mathsf{FSS.Eval}$ from each party.

Theorem 5.3 *Let* $R = R(\lambda)$ *be a finite commutative ring. Suppose there exists an FSS scheme for scalar multiples of a family of weak pseudorandom functions* $\mathcal{F} = \{F_K : \{0,1\}^n \to R\}_{K\in\{0,1\}^\lambda}$*. Then, there is a PCF for the VOLE correlation over* R*, given by the construction in Fig. 6.*

Proof. First, note that since we assume the existence of a WPRF, we can assume w.l.o.g. that the scheme $\mathsf{FSS} = (\mathsf{FSS.Gen}, \mathsf{FSS.Eval})$ satisfies the weak pseudorandom outputs property. We prove the PCF security property for the two separate cases in Definition B.2, and then show that the PCF has pseudorandom correlated outputs.

Security with $\sigma = 0$. In the real experiment, $\mathsf{Exp}^{\mathsf{sec}}_{\mathcal{A},\sigma,0}(\lambda)$, the adversary $\mathcal{A}$ is run on input the PCF key $\mathsf{k}_0 = (K_0^{\mathsf{fss}}, a)$, and the N random input/output pairs $(x^{(i)}, (b_i, c_{1,i}))$, where $x^{(i)}$ is uniformly random, $b_i = F_K(x^{(i)})$ and $c_{1,i} = \mathsf{FSS.Eval}(1, K_1^{\mathsf{fss}}, x)$. We show that this is indistinguishable from the ideal experiment, where $(b_i, c_{1,i})$ are instead reverse-sampled using $(a, c_{0,i})$ derived from key k_0. We use the following sequence of hybrid experiments.

In the first hybrid, instead of computing $c_{1,i}$ from $\mathsf{FSS.Eval}$ using key K_1^{fss}, we compute $c_{1,i} = c_{0,i} + ab_i$ in R. Note that by the correctness property of the FSS scheme, this hybrid is statistically close to the real experiment.

Next, we switch the FSS key K_0^{fss} with a simulated key $\widetilde{K_0^{\mathsf{fss}}} \leftarrow \mathsf{Sim}(1^\lambda, n, R)$ using the FSS simulator Sim. The values $c_{0,i}$ are then obtained from the new key, and $c_{1,i}$ computed from this as before. Due to the security property of the FSS, and the fact that $c_{1,i}$ is independent of K_1^{fss}, this is computationally indistinguishable from the previous experiment.

In the final hybrid, we replace the b_i values with uniformly random elements from R, and again compute $c_{1,i} = c_{0,i} + ab_i$ but using the new b_i. This is identical to the ideal experiment, and indistinguishability holds by a standard reduction to the security of the WPRF.

Security with $\sigma = 1$. Security in this case follows immediately from the correctness property of the FSS scheme. We have that in the real experiment, $\mathcal{A}$ gets a PCF key $\mathsf{k}_1 = (K_1^{\mathsf{fss}}, K)$, and the N random input/output pairs $(x^{(i)}, (a, c_{0,i}))$, computed using a random $x^{(i)}$ and $(a, c_{0,i}) = \mathsf{PCF.Eval}(0, \mathsf{k}_0, x^{(i)})$. The only difference between this and the ideal experiment is the way $c_{0,i}$ is computed, and FSS correctness implies that these are statistically close.

Pseudorandom correlated outputs. We now show that the joint distribution of the N output pairs $(a, c_{0,i}), (b_i, c_{1,i})$, on random inputs $x^{(i)}$, is indistinguishable from a set of samples from the VOLE correlation. Starting from the real distribution, we use the following sequence of experiments, which is similar to the hybrids in the security argument when $\sigma = 0$; the only difference here is that we need to rely on the weak pseudorandom outputs property of the FSS, instead of its security property.

We first replace $c_{i,1}$ with $c_{i,0} + ab_i$, where we get statistical indistinguishability because of the FSS correctness property. Next, we replace each $c_{0,i}$ with a random element of R; here, we appeal to the weak pseudorandom outputs property of the FSS scheme, which implies this is computational indistinguishable from the previous experiment. Finally, we can replace b_i with random elements of R, due to the security of the WPRF. □

5.2 PCF for Oblivious Transfer

A 1-out-of-2 string-OT correlation is sampled as a pair (s_0, s_1) and (c, s_c), where s_0, s_1 are the OT sender's random strings in $\{0,1\}^\lambda$, and c is a random choice bit given to the receiver. We construct a PCF for independent instances of string-OT using FSS for a WPRF, together with a correlation-robust hash function [IKNP03]. Note that the security of our WPRF candidates implies a correlation-robust hash function, so instantiating with these candidates does not require the extra assumption.

Definition 5.4 (Correlation robust hash function) *Let $N = \mathsf{poly}(\lambda)$. We say that an efficiently computable function $\mathsf{H} : \{0,1\}^\lambda \to \{0,1\}^\lambda$ is* correlation robust *if the distribution*

PCF for Vector Oblivious Linear Evaluation

Let $\mathcal{F} = \{F_K : \{0,1\}^n \to R\}_{K \in \{0,1\}^\lambda}$ be a weak PRF, and $\mathsf{FSS} = (\mathsf{FSS.Gen}, \mathsf{FSS.Eval})$ an FSS scheme for $\{cF_K\}_{c \in R, K \in \{0,1\}^\lambda}$ with weak pseudorandom outputs.

$\mathsf{PCF.Gen}(1^\lambda)$**:**

1. Sample a WPRF key $K \leftarrow \{0,1\}^\lambda$ and $a \leftarrow R$
2. Sample a pair of FSS keys $(K_0^{\mathsf{fss}}, K_1^{\mathsf{fss}}) \leftarrow \mathsf{FSS.Gen}(1^\lambda, aF_k)$
3. Output the keys $\mathsf{k}_0 = (K_0^{\mathsf{fss}}, a)$ and $\mathsf{k}_1 = (K_1^{\mathsf{fss}}, K)$.

$\mathsf{PCF.Eval}(\sigma, \mathsf{k}_\sigma, x)$**:** On input a random x:

- If $\sigma = 0$:
  1. Let $c_0 = -\mathsf{FSS.Eval}(0, K_0^{\mathsf{fss}}, x)$
  2. Output (a, c_0)
- If $\sigma = 1$:
  1. Let $c_1 = \mathsf{FSS.Eval}(1, K_1^{\mathsf{fss}}, x)$
  2. Let $b = F_K(x)$
  3. Output (b, c_1)

Figure 6: PCF for VOLE over the ring R based on FSS for scalar multiples of a weak PRF

$$\{t_1, \ldots, t_N, \mathsf{H}(t_1 \oplus s), \ldots, \mathsf{H}(t_N \oplus s)\}$$

where s and $t_1, \ldots, t_N$ are independent and uniformly sampled from $\{0,1\}^\lambda$, is computationally indistinguishable from the uniform distribution on $\{0,1\}^{2\lambda N}$.

Our construction, in Fig. 7, proceeds by first creating *correlated OTs*, which are OTs where the sender's messages are all of the form $q_i, q_i \oplus s$ for a single, random s. Then, as in [IKNP03] and later works, the strings are hashed using the public, correlation robust function, which converts them into OTs on random strings. Note that to rely on correlation robustness instead of a stronger notion, we require that each q_i is uniform; since this is not the case in our construction, we mask it with a portion of the random, PCF input.

To construct correlated OTs, the setup phase will first sample a key K for a WPRF with range $\{0,1\}$, and random bits $s_1, \ldots, s_\lambda$. We then create FSS keys for λ functions, each of which is either the WPRF F_K or the zero function, depending on the bit s_j. Evaluating all λ FSS keys on a public input $x^{(i)}$, the sender and receiver obtain respective strings q_i and t_i in $\{0,1\}^\lambda$ such that

$$q_i \oplus t_i = s \cdot F_K(x^{(i)})$$

where $s = (s_1, \ldots, s_\lambda)$, and $\cdot$ computes the AND of every bit in s with $F_k(x^{(i)})$.

As well as the FSS keys, we then give out the key K to the receiver and s to the sender. The receiver's values t_i satisfy $t_i = q_i \oplus c_i \cdot s$, where $c_i = F_K(x^{(i)})$ is its choice bit, which means the values $(q_i, q_i \oplus s)$ known to the sender form a correlated OT as required.

The above outline assumes we have FSS for a WPRF family together with the zero function. However, it suffices to have FSS for the WPRF, since any FSS scheme for a family $\mathcal{F}$ with outputs in $\mathbb{F}_2$ implies FSS for the class $\mathcal{F} \cup \{0\}$, which includes the zero function [BGI15]. This holds because without loss of generality, we can assume that the $\mathsf{FSS.Eval}$ function is symmetric

PCF for Oblivious Transfer

Let $\mathcal{F} = \{F_K : \{0,1\}^n \to \{0,1\}\}_{K\in\{0,1\}^\lambda}$ be a WPRF, and $\mathsf{FSS} = (\mathsf{FSS.Gen}, \mathsf{FSS.Eval})$ an FSS scheme for $\mathcal{F} \cup \{0\}$.

$\mathsf{PCF.Gen}(1^\lambda)$**:**

1. Sample a WPRF key $K \leftarrow \{0,1\}^\lambda$ and a random $s \leftarrow \{0,1\}^\lambda$
2. For $j = 1, \ldots, \lambda$, sample $(K^{\mathsf{fss}}_{0,j}, K^{\mathsf{fss}}_{1,j}) \leftarrow \mathsf{FSS.Gen}(1^\lambda, s_j F_K)$
3. Output the keys $\mathsf{k}_0 = (s, K^{\mathsf{fss}}_{0,1}, \ldots, K^{\mathsf{fss}}_{0,\lambda})$ and $\mathsf{k}_1 = (K, K^{\mathsf{fss}}_{1,1}, \ldots, K^{\mathsf{fss}}_{1,\lambda})$.

$\mathsf{PCF.Eval}(1^\lambda, \sigma, \mathsf{k}_\sigma, x)$**:** On input a random $x \in \{0,1\}^n$, let x' be the first λ bits of x and then:

- If $\sigma = 0$:
  1. Compute $q = (q_1, \ldots, q_\lambda)$, where $q_j = \mathsf{FSS.Eval}(0, K^{\mathsf{fss}}_{0,j}, x)$
  2. Compute $y_0 = \mathsf{H}(x' \oplus q), y_1 = \mathsf{H}(x' \oplus q \oplus s)$
  3. Output (y_0, y_1)
- If $\sigma = 1$:
  1. Compute $c = F_K(x)$
  2. Compute $t = (t_1, \ldots, t_\lambda)$, where $t_j = \mathsf{FSS.Eval}(1, K^{\mathsf{fss}}_{1,j}, x)$
  3. Output (c, z), where $z = \mathsf{H}(x' \oplus t)$

Figure 7: PCF for 1-out-of-2 oblivious transfer based on FSS for a weak PRF

for both keys, and also that each key hides whether it is the key for party 0 or party 1. The zero function is then shared by sampling a pair $(K^{\mathsf{fss}}_0, K^{\mathsf{fss}}_1)$ and giving K^{fss}_0 to both parties.

Theorem 5.5 *Suppose there exists an FSS scheme for a family of weak pseudorandom functions $\mathcal{F} = \{F_k : \{0,1\}^n \to \{0,1\}\}_{k\in\{0,1\}^\lambda}$, and a correlation robust hash function $\mathsf{H} : \{0,1\}^\lambda \to \{0,1\}^\lambda$. Then, there is a PCF for the string-OT correlation, given in Fig. 7.*

Proof. We first prove the security property of the PCF, and then the pseudorandomness property. As in Definition 4.3, we consider $N = \mathsf{poly}(\lambda)$ queries on random inputs $x^{(1)}, \ldots, x^{(N)}$, where we let $(x')^{(i)}$ be the first λ bits of $x^{(i)}$.

Corrupt receiver ($\sigma = 1$). In the real experiment, $\mathsf{Exp}^{\mathsf{sec}}_{\mathcal{A},\sigma,0}(\lambda)$, the adversary $\mathcal{A}$ is given $\mathsf{k}_1 = (K, K^{\mathsf{fss}}_{1,1}, \ldots, K^{\mathsf{fss}}_{1,\lambda})$ and random input/output pairs $(x^{(i)}, (y_{i,0}, y_{i,1}))$. Here, $K^{\mathsf{fss}}_{1,j}$ is an FSS key for the function $s_j F_K$, where $s_j \in \{0,1\}$, and $y_{i,0}, y_{i,1}$ are the OT sender's strings from applying $\mathsf{PCF.Eval}$ on the key k_0. Note that computing $(y_{i,0}, y_{i,1})$ involves first computing a string q_i using $\mathsf{FSS.Eval}$ on the keys $(K^{\mathsf{fss}}_{0,1}, \ldots, K^{\mathsf{fss}}_{0,\lambda})$, and then hashing both q_i and $q_i \oplus s$.

Consider a hybrid experiment where we replace the strings q_i with $q_i = t_i \oplus s \cdot F_K(x^{(i)})$, where t_i comes from $\mathsf{FSS.Eval}$ on the keys $(K^{\mathsf{fss}}_{1,1}, \ldots, K^{\mathsf{fss}}_{1,\lambda})$, and then give $\mathcal{A}$ strings $y_{i,0}, y_{i,1}$ computed with the new q_i. By the correctness of the FSS scheme, this experiment is statistically close to the previous one.

In the next λ experiments, we successively replace each FSS key $K^{\mathsf{fss}}_{1,j}$ given to $\mathcal{A}$, for $j = 1, \ldots, \lambda$, with a simulated key $\mathsf{Sim}(1^\lambda, \{0,1\}^{n+\lambda}, \mathbb{F}_2)$, and recompute each $t_i, q_i, y_{i,0}, y_{i,1}$ with the simulated keys. By the fact that the strings $y_{i,0}, y_{i,1}$ seen by $\mathcal{A}$ are now independent of the other

FSS keys $K^{\mathsf{fss}}_{0,j}$, from the security of the FSS scheme we have that this sequence of hybrids is computationally indistinguishable from the previous one.

In the final experiment, we replace each of the values $y_{i,1-c_i}$ with a uniformly random string. Note that this is identical to the ideal experiment, $\mathsf{Exp}^{\mathsf{sec}}_{\mathcal{A},\sigma,1}(\lambda)$. In the previous experiment, we had $y_{i,1-c_i} = \mathsf{H}((x')^{(i)} \oplus q_i \oplus (1-c_i)s) = \mathsf{H}((x')^{(i)} \oplus t_i \oplus s)$. Since $(x')^{(i)}$, s are independently uniform, and s is independent of t_i and all other values given to $\mathcal{A}$, these are computationally indistinguishable from the random values in $\mathsf{Exp}^{\mathsf{sec}}_{\mathcal{A},\sigma,1}(\lambda)$, by a reduction to the correlation robustness of H.

Corrupt sender ($\sigma = 0$). In the real experiment, the adversary $\mathcal{A}$ is given $\mathsf{k}_0 = (s, K^{\mathsf{fss}}_{0,1}, \ldots, K^{\mathsf{fss}}_{0,\lambda})$ and random input/output pairs $(x^{(i)}, (c_i, z_i))$, where c_i, z_i are the OT receiver's choice bit and string computed from key k_1, for $i = 1, \ldots, N$.

We first define an experiment where z_i is replaced with the y_{i,c_i} value obtained by running Eval on key k_0. This is statistically close to the first experiment, by the FSS correctness property.

Similarly to the case of a corrupt receiver, we then replace the FSS keys $K^{\mathsf{fss}}_{0,j}$, for $j = 1, \ldots, \lambda$, with simulated keys from the FSS simulator, then recompute $(y_{i,0}, y_{i,1})$ using the new keys and let $z_i = y_{i,c_i}$. This is indistinguishable from the first experiment, by the security of the FSS scheme and a hybrid argument.

Finally, we replace $c_1, \ldots, c_N$ with uniformly random bits, and again define $z_i = y_{i,c_i}$. This is computationally indistinguishable to the previous experiment, due to the security of the weak-PRF F_K, since the key K is uniform and independent of all inputs to $\mathcal{A}$, and the inputs $x^{(i)}$ are uniform. This concludes the proof of the security property.

Pseudorandom correlated outputs. Here, we show that the joint distribution of the PCF outputs $(y_{i,0}, y_{i,1}), (c_i, z_i)$ is indistinguishable from a set of independent samples from the OT correlation. First, again by correctness of FSS, we can replace z_i with the value y_{i,c_i} that comes from the sender's key k_0. Then, we can replace the FSS outputs q_i, which are used to compute $y_{i,0}$ and $y_{i,1}$, with uniform strings, because of the weak pseudorandomness property of FSS. Then, as previously, we can replace the FSS keys $K^{\mathsf{fss}}_{0,j}$ with simulated keys, and use these to compute q_i and the outputs $(y_{i,0}, y_{i,1})$, because of the security property of FSS. Now, since q_i are all independent of s, we use the correlation robustness property of H to replace $y_{i,0}, y_{i,1}$ with uniformly random strings. Finally, the choice bits $c_1, \ldots, c_N$ are now computationally indistinguishable from uniform by the security of the weak-PRF, which completes the proof. □

Remark 5.6 *The PCF for OT can be optimized if we are instead given FSS for scalar multiples of the WPRF over $\mathbb{F}_{2^\lambda}$, namely, the class of functions $F'_{K,s}(x) = s \cdot F_K(x)$, for all $s \in \mathbb{F}_{2^\lambda}$. Then, we only need a single pair of FSS keys for this function family, instead of λ sets of keys. This optimization is used in our concrete instantiation of the PCF for OT, in Section 6.4.*

5.3 PCF for Multiplication Triples

A multiplication triple correlation over some ring R is sampled as a pair (a_0, b_0, c_0), (a_1, b_1, c_1), where a_0, b_0, a_1, b_1 are independent and uniform over R, while c_0, c_1 are uniformly random such that $c_0 + c_1 = (a_0 + a_1)(b_0 + b_1)$.

We can obtain a PCF for multiplication triples, given both FSS for a WPRF family $\mathcal{F}$ and its *square*, namely the function family $\mathcal{F}^2 = \{f_1 f_2 : f_1, f_2 \in \mathcal{F}\}$. The construction simply samples two keys $k_1, k_2 \leftarrow \{0,1\}^\lambda$, and gives out FSS keys for the functions F_{k_1}, F_{k_2} and $F_{k_1} F_{k_2}$. To evaluate the PCF, the three FSS keys are evaluated in turn to produce additive secret shares of a, b and $c = ab$ respectively. Similarly to the VOLE correlation in Section 5.1, security of this construction can be proven based on the security of the WPRF, and the weak pseudorandom outputs property of the FSS scheme.

We remark that in this case, our specific WPRF candidate allows going much further than the general construction for multiplication triples. In the coming section, we present a PCF for arbitrary degree-2 correlations based on our WPRF, which comes at the same cost as FSS for the square of the WPRF family.

6 A Candidate FSS-Friendly WPRF

In this section, we describe candidate weak pseudorandom functions to instantiate the framework developed in Section 5. As outlined in the introduction, our candidates rely on *variable-density* variants of the LPN assumption, which we introduce below.

6.1 Variable-Density Learning Parity with Noise

We first formally introduce the VDLPN assumption. Fix a security parameter λ. VDLPN has three parameters: a *sparsity parameter* $w = w(\lambda)$, which corresponds to the number of nonzero coordinates in each secret error vector $\vec{e}_i$ and each row of the public matrix H_i; a *block parameter* $D = D(\lambda)$, which corresponds to the number of blocks H_i, and a *number of samples* $N = N(\lambda)$, which we set to 2^D. For the sake of concreteness, think of w, D as being linear in λ. Given parameters $\mathsf{par} = (w, D, N)$, the assumption can come in three flavors:

- the *standard* VDLPN assumption (denoted simply $\mathsf{VDLPN}(\mathsf{par})$), in which each $\vec{s}_i$ and each row of H_i, of length $w \cdot 2^i$, is sampled independently from $\mathsf{Ber}_{1/2^i}^{w \cdot 2^i}$;
- the *exact* VDLPN assumption (denoted $\mathsf{xVDLPN}(\mathsf{par})$), in which each $\vec{s}_i$ and each row of H_i are sampled uniformly from the set of all length-$w \cdot 2^i$ vectors with *exactly* w nonzero entries;
- the *regular* VDLPN assumption (denoted $\mathsf{rVDLPN}(\mathsf{par})$), in which each $\vec{s}_i$ and each row of H_i are sampled by contatenating w random length-2^i unit vectors (*i.e.*, they are divided into w equal length block with a single random 1 in each block).

The above distinction is analogous to the standard variants of the LPN assumption, which also comes with an exact variant (see e.g. [Pie12]) and a regular variant (see e.g. [AFS03]). These are widely believed to be no less secure than the standard LPN assumption (this can actually be formally proven for the search variant of exact LPN [Pie12]). The main difference is that in our setting, the exact and regular variant refer not only to the structure of the *noise vector*, but also to the structure of the code *parity-check matrix*, since we always assume that the rows of H follow the same distribution as the noise vector. In the following, we will focus mainly on the regular VDLPN assumption by default. This is motivated by the fact that it leads to a simpler and more efficient WPRF candidate than its standard and exact counterparts.

Distributions. Fix parameters $\mathsf{par} = (w, D, N = 2^D)$. Let $\mathcal{R}_{w,i}$ be the distribution of random w-regular vectors over $\mathbb{F}_2^{w \cdot 2^i}$ (that is, a sample from $\mathcal{R}_{w,i}$ is obtained by concatenating w independent samples from $\mathcal{S}_{1,2^i}$). We let $\mathcal{H}_{\mathsf{par}}^i$ denote the distribution over $N \times (w \cdot 2^i)$ matrices over $\mathbb{F}_2$ where each row is sampled independently from $\mathcal{R}_{w,i}$, and $\mathcal{H}_{\mathsf{par}}$ denote the distribution over $\mathbb{F}_2^{N \times 2N}$ obtained by sampling $H_i \xleftarrow{\$} \mathcal{H}_{\mathsf{par}}^i$ for $i = 1$ to D and outputting $H = H_1 || \cdots || H_D$. Eventually, we denote by $\mathcal{N}_{\mathsf{par}}$ the noise distribution obtained by sampling $\vec{e}_i^{\intercal} \xleftarrow{\$} \mathcal{R}_{w,i}$ and outputting $\vec{e} \leftarrow (\vec{e}_1 // \cdots // \vec{e}_D) \in \mathbb{F}_2^{2N}$.

Definition 6.1 ($\mathsf{rVDLPN}(w, D, N)$) *The* regular variable-density learning parity with noise *assumption with sparsity w, D blocks, and number of samples N, denoted* $\mathsf{rVDLPN}(w, D, N)$*, states that*

$$\{(H, \vec{b}) \mid H \xleftarrow{\$} \mathcal{H}_{\mathsf{par}}, \vec{e} \xleftarrow{\$} \mathcal{N}_{\mathsf{par}}, \vec{b} \leftarrow H \cdot \vec{e}\} \approx \{(H, \vec{b}) \mid H \xleftarrow{\$} \mathcal{H}_{\mathsf{par}}, \vec{b} \xleftarrow{\$} \mathbb{F}_2^N\}.$$

For any H in the support of $\mathcal{H}_{\mathsf{par}}$, we denote by $\mathcal{O}_{\mathsf{par}}(H)$ the output distribution, i.e., the distribution over $\mathbb{F}_2^N$ induced by sampling $\vec{e} \xleftarrow{\$} \mathcal{N}_{\mathsf{par}}$ and outputting $H \cdot \vec{e}$.

6.2 A Candidate WPRF in Depth-2 $\mathsf{AC}^0[\oplus]$ from rVDLPN

The above rVDLPN assumption immediately implies a weak PRF candidate. Fix parameters $\mathsf{par}(\lambda) = (w(\lambda), D(\lambda), N(\lambda) = 2^{D(\lambda)})$. To simplify the description of the WPRF family, it is helpful to make explicit the exact number of random bits used to sample from the distribution $\mathcal{N}_{\mathsf{par}}$. Recall that a sample from $\mathcal{N}_{\mathsf{par}}$ is a (vertical) concatenation of D vectors $\vec{e}_i$, where each vector $\vec{e}_i$ is a (transposed) sample from $\mathcal{R}_{w,i}$. In turn, a sample from $\mathcal{R}_{w,i}$ is a concatenation of w random unit vectors over $\mathbb{F}_2^{2^i}$, where each can be sampled using i random bits (indicating the position, between 1 and 2^i, of the nonzero entry of the vector). Therefore, sampling from $\mathcal{N}_{\mathsf{par}}$ requires exactly $w \cdot \sum_{i=1}^{D} i = w \cdot D(D-1)/2$ random bits. Given $r \in \{0,1\}^{w \cdot D(D-1)/2}$, we write $\mathcal{N}_{\mathsf{par}}(r)$ to denote the value sampled from $\mathcal{N}_{\mathsf{par}}$ using the bistring r as the source of random bits. The construction of the weak PRF family is described below.

- Key size: $K \in \{0,1\}^{\kappa(\lambda)}$ with $\kappa(\lambda) = n(\lambda) = w(\lambda) \cdot D(\lambda)(D(\lambda)-1)/2$.
- Input size: $x \in \{0,1\}^{n(\lambda)}$ with $n(\lambda) = w(\lambda) \cdot D(\lambda)(D(\lambda)-1)/2$.
- $F_K(x)$: on input $x \in \{0,1\}^n$, sample $\vec{h}^\intercal \leftarrow \mathcal{N}_{\mathsf{par}}(x)$ and output $\langle \vec{h}, \mathcal{N}_{\mathsf{par}}(K) \rangle$.

Theorem 6.2 *Assume that* rVDLPN(par) *holds. Then the above construction is an N-query weak pseudorandom function family, with input length and key length $n = \kappa = w \cdot D(D-1)/2$. Furthermore, for any fixed choice of key K, the function F_K can be implemented with a depth-2 $\mathsf{AC}^0[\oplus]$ circuit with a layer of AND gates at the bottom, and a single XOR gate at the top.*

Proof. By construction, K is distributed as the noise vector in rVDLPN, and the distribution of $\vec{h}$ is exactly the same as the distribution each rows of H is sampled from in $\mathcal{H}_{\mathsf{par}}$. Hence, distinguishing $(F_K(\vec{h}_1), \cdots, F_K(\vec{h}_N))$ given random $(\vec{h}_1, \cdots, \vec{h}_N)$ and for a random key K is perfectly equivalent to distinguishing $H \cdot \vec{e}$ from random given H, where H (whose rows are the $\vec{h}_i$) is sampled from $\mathcal{H}_{\mathsf{par}}$ and $K = \vec{e}$ is sampled from $\mathcal{N}_{\mathsf{par}}$. It remains to show that F_K can be implemented with a depth-2 $\mathsf{AC}^0[\oplus]$ circuit with a layer of AND gate at the bottom, and a single XOR gate at the top. By construction we have

$$F_K(x) = \langle \mathcal{N}_{\mathsf{par}}(x), \mathcal{N}_{\mathsf{par}}(K) \rangle.$$

Let us decompose x as $(x_{i,j})_{i \le D, j \le w}$ and K as $(K_{i,j})_{i \le D, j \le w}$, with $|x_{i,j}| = |K_{i,j}| = i$. The string $x_{i,j}$ (resp. $K_{i,j}$) correspond to the portion of x (resp. of K) which is used to sample the j'th unit vector from $\mathcal{R}_{w,i}$ when computing $\mathcal{N}_{\mathsf{par}}(x)$ (resp. $\mathcal{N}_{\mathsf{par}}(K)$). That is,

$$\begin{aligned}
\mathcal{N}_{\mathsf{par}}(x) &= \mathcal{R}_{w,1}((x_{1,j})_{j \le w})^\intercal /\!/ \cdots /\!/ \mathcal{R}_{w,D}((x_{D,j})_{j \le w})^\intercal \\
&= (\mathcal{S}_{1,2}(x_{1,1}) || \cdots || \mathcal{S}_{1,2}(x_{1,w}))^\intercal /\!/ \cdots /\!/ (\mathcal{S}_{1,2^D}(x_{D,1}) || \cdots || \mathcal{S}_{1,2^D}(x_{D,w}))^\intercal, \\
\mathcal{N}_{\mathsf{par}}(K) &= \mathcal{R}_{w,1}((K_{1,j})_{j \le w})^\intercal /\!/ \cdots /\!/ \mathcal{R}_{w,D}((K_{D,j})_{j \le w})^\intercal \\
&= (\mathcal{S}_{1,2}(K_{1,1}) || \cdots || \mathcal{S}_{1,2}(K_{1,w}))^\intercal /\!/ \cdots /\!/ (\mathcal{S}_{1,2^D}(K_{D,1}) || \cdots || \mathcal{S}_{1,2^D}(K_{D,w}))^\intercal.
\end{aligned}$$

Therefore,

$$\begin{aligned}
F_K(x) &= \langle \mathcal{N}_{\mathsf{par}}(x), \mathcal{N}_{\mathsf{par}}(K) \rangle \\
&= \bigoplus_{i=1}^{D} \bigoplus_{j=1}^{w} \langle \mathcal{S}_{1,2^i}(x_{i,j}), \mathcal{S}_{1,2^i}(K_{i,j}) \rangle.
\end{aligned}$$

Now, by definition of $\mathcal{S}_{1,2^i}$, the inner product $\langle \mathcal{S}_{1,2^i}(x_{i,j}), \mathcal{S}_{1,2^i}(K_{i,j}) \rangle$ is equal to 1 if and only if $x_{i,j} = K_{i,j}$, if and only iff $x_{i,j,k} \oplus K_{i,j,k} = 0$ for $k = 1$ to i. Hence:

$$F_K(x) = \bigoplus_{i=1}^{D} \bigoplus_{j=1}^{w} \bigwedge_{k=1}^{i} (x_{i,j,k} \oplus K_{i,j,k} \oplus 1).$$

Therefore, for any fixed choice of key K, the function F_K computes a fan-in-$D \cdot w$ XOR of fan-in-i ANDs of terms, where each term is either an input bit or its complement. □

Note that the above characterization gives an alternative description of our candidate WPRF as a function computed by a simple boolean circuit. Omitting the "$\oplus 1$" terms in the formula above does not change the function family (since it simply amounts to flipping all the bits of the PRF key, which does not change its distribution), hence an equivalent formulation of our weak PRF candidate is given by

$$F_K(x) = F(K \oplus x)$$

where the function F is defined as

$$F(x) = \bigoplus_{i=1}^{D} \bigoplus_{j=1}^{w} \bigwedge_{k=1}^{i} x_{i,j,k}.$$

We conjecture that this candidate, which is based on the rVDLPN assumption, achieves exponential security. More precisely, we put forth the following conjecture:

Conjecture 6.3 (Subexponential Security) *There exists constants (C_1, C_2) such that for every large enough security parameter λ, every distinguisher that runs in time $T(\lambda)$ has advantage at most $O(T/2^{\lambda})$ against the $\mathsf{rVDLPN}(C_1 \cdot \lambda, C_2 \cdot \lambda, 2^{C_2 \cdot \lambda})$ assumption.*

In Section 7, we will provide support to this conjecture, by analyzing the security of the rVDLPN assumption against a wide variety of distinguishers (including linear and low-degree polynomial distinguisher, algebraic distinguishers, statistical query algorithms, AC^0 circuits, and linear cryptanalysis attacks).

6.3 Generalization to Arbitrary Rings

Our WPRF candidate can be naturally generalized to work over an arbitrary ring R, by using the same sparse distributions where the ones are replaced by random nonzero ring elements: given parameters $\mathsf{par} = (w, D, N = 2^D)$, we let $\mathcal{R}_{w,i}(R)$ be the distribution of random w-regular vectors over $R^{w \cdot 2^i}$, $\mathcal{H}^i_{\mathsf{par}}(R)$ denote the distribution over $n \times (w \cdot 2^i)$ matrices over R where each row is sampled independently from $\mathcal{R}_{w,i}(R)$, and $\mathcal{H}_{\mathsf{par}}(R)$ denote the distribution over $R^{N \times 2N}$ obtained by sampling $H_i \xleftarrow{\$} \mathcal{H}^i_{\mathsf{par}}(R)$ for $i = 1$ to D and outputting $H = H_1 || \cdots || H_D$. Finally, we denote by $\mathcal{N}_{\mathsf{par}}(R)$ the noise distribution obtained by sampling $\vec{e}_i^{\intercal} \xleftarrow{\$} \mathcal{R}_{w,i}(R)$ and outputting $\vec{e} \leftarrow (\vec{e}_1 // \cdots // \vec{e}_D) \in R^{2N}$.

Definition 6.4 (R-$\mathsf{rVDLPN}(w, D, N)$) *The* regular variable-density learning parity with noise *assumption over a ring R with sparsity w, D blocks, and number of samples N, denoted by R-$\mathsf{rVDLPN}(w, D, N)$, states that*

$$\{(H, \vec{b}) \mid H \xleftarrow{\$} \mathcal{H}_{\mathsf{par}}(R), \vec{e} \xleftarrow{\$} \mathcal{N}_{\mathsf{par}}(R), \vec{b} \leftarrow H \cdot \vec{e}\} \approx \{(H, \vec{b}) \mid H \xleftarrow{\$} \mathcal{H}_{\mathsf{par}}(R), \vec{b} \xleftarrow{\$} R^N\}.$$

This assumption leads to a candidate WPRF over R via the same construction. The candidate can be written as follows: it receives as input a $w \cdot D(D-1)/2$-bit string $\vec{x}$ together with $w \cdot D$ ring elements $\vec{y}$, and a PRF key has the same form (a $w \cdot D(D-1)/2$-bit string K together with $w \cdot D$ ring elements L). An element of $\{0,1\}$ is interpreted naturally as an element of R

in the formula below (0 is the neutral for addition and 1 the neutral for multiplication). The generalized candidate is defined as

$$F_{K,L}(\vec{x},\vec{y}) = \sum_{i=1}^{D}\sum_{j=1}^{w}(y_{i,j}\cdot L_{i,j})\cdot\prod_{k=1}^{i}(x_{i,j,k}\oplus K_{i,j,k}).$$

Part of our security analysis, most notably the proof of resistance against linear attacks and their natural generalizations, extend to this arithmetic generalization of our candidate if we choose w sufficiently large (in particular, $w > D + \log|R|$). Plugging this generalized WPRF into the VOLE construction of Figure 8 leads to a PCF for VOLE over an arbitrary ring R (see Section 6.4 for more details). We conjecture that this generalization is secure for an arbitrary choice of the ring R, with parameters that only depend on $\log|R|$.

6.4 FSS-Friendliness of our WPRF

Recall that as established in Section 5, we have to show that our weak PRF is FSS-friendly, meaning that there exists a light-weight FSS scheme allowing to share the PRF keys between the parties. In the following we will show that our PRF can indeed be viewed as a sum of *point functions*, which build a function class that is very FSS-friendly, even admitting constructions that only make black-box use of a pseudorandom generator, as elaborated in the following.

Point functions. Let $\mathbb{G}$ be an additive group, $\alpha\in\{0,1\}^{n(\lambda)}$ and $\beta\in\mathbb{G}$. Let $f_{\alpha,\beta}$ be the point function where $f_{\alpha,\beta}(x)$ is zero whenever $x\neq\alpha$, and $f_{\alpha,\beta}(x)=\beta$ if $x=\alpha$. If $\mathbb{G}=(\{0,1\},\oplus)$, we simply write f_α to denote $f_{\alpha,1}$.

Distributed point functions. An FSS scheme for the class of point functions

$$\{f_{\alpha,\beta}:\{0,1\}^{n(\lambda)}\to\mathbb{G}\mid\alpha,\beta\in\{0,1\}^{n(\lambda)}\}$$

is called a *distributed point function (DPF)* [GI14]. In the following, we will consider distributed point functions with variable input space. More precisely, such a DPF consists of a tuple of algorithms $(\mathsf{DPF.Gen},\mathsf{DPF.Eval})$, where $\mathsf{DPF.Gen}$ additionally takes a parameter $n=n(\lambda)$ specifying the input length. Given a PRG $G:\{0,1\}^\lambda\to\{0,1\}^{2\lambda+2}$, we can construct a distributed point function with the following complexities [BGI16b]: Let $m(\lambda)=\lceil\frac{\log|\mathbb{G}|}{\lambda+2}\rceil$, then

- the size of each party's key is at most $n(\lambda)\cdot(\lambda+2)+\lambda+\lceil\log|\mathbb{G}|\rceil$ bits,
- the key generation algorithm Gen invokes G at most $2(n(\lambda)+m(\lambda))$ times,
- the evaluation algorithm Eval invokes G at most $n(\lambda)+m(\lambda)$ times.

Note that the DPF of [BGI16b] has pseudorandom outputs; in fact, this holds generally for every DPF [BGI15].

FSS-friendliness of our construction. Recall that our construction is of the form

$$F_{K,L}(x,y) = \sum_{i=1}^{D}\sum_{j=1}^{w}(y_{i,j}\cdot L_{i,j})\cdot\prod_{k=1}^{i}(x_{i,j,k}\oplus K_{i,j,k})$$

over general ring R. For the following for all $i\in[w],j\in[D]$ let $x_{i,j}=x_{i,j,1}\dots x_{i,j,i}\in\{0,1\}^i$ (and for $K_{i,j}$ accordingly). Then, for all $i\in[w],j\in[D]$ we have $\prod_{k=1}^{i}(x_{i,j,k}\oplus K_{i,j,k})=1$ for the single point $x_{i,j}\in\{0,1\}^i$, for which $x_{i,j}=K_{i,j}$. We can thus rewrite our candidate as

$$F_{K,L}(x) = \sum_{i=1}^{D}\sum_{j=1}^{w}y_{i,j}\cdot f_{K_{i,j},L_{i,j}}(x_{i,j}),$$

PCF for VOLE from regular VDLPN

Let $\mathsf{DPF} = (\mathsf{DPF.Gen}, \mathsf{DPF.Eval})$ be an FSS scheme for point functions with variable input length. Recall that $f_{K,L}\colon \{0,1\}^i \to R$ is the point function with $f_{K,L}(x) = L$ if and only if $x = K$.

$\mathsf{PCF.Gen}(1^\lambda)$**:** On input 1^λ:

1. Sample $a \leftarrow R$.
2. For all $j \in [w]$ and $i \in [D]$:
   - Sample a key $K_{i,j} \leftarrow \{0,1\}^i$.
   - Sample payloads $L_{i,j} \leftarrow R$.
   - Sample a pair of FSS keys $(K^{\mathsf{fss}}_{0,i,j}, K^{\mathsf{fss}}_{1,i,j}) \leftarrow \mathsf{DPF.Gen}(1^\lambda, i, a \cdot f_{K_{i,j},L_{i,j}})$.
3. Set $K := \{K_{i,j}\}_{j\in[w],i\in[D]}$.
4. Output the keys $\mathsf{k}_0 = (\{K^{\mathsf{fss}}_{0,i,j}\}_{j\in[w],i\in[D]}, a)$ and $\mathsf{k}_1 = (\{K^{\mathsf{fss}}_{1,i,j}\}_{j\in[w],i\in[D]}, K)$.

$\mathsf{PCF.Eval}(\sigma, \mathsf{k}_\sigma, (x, y))$**:** On input a random $x \in \{0,1\}^{w\cdot D(D+1)/2}, y \in R^{w\cdot B}$:
Parse x as $x_{i,j} \in \{0,1\}^i$, for $j \in [w], i \in [D]$, and y as $y_{i,j} \in \mathbb{F}$, for $j \in [w], i \in [D]$.

- If $\sigma = 0$:
  1. For all $j \in [w]$ and $i \in [D]$: Let $c_{0,i,j} \leftarrow \mathsf{DPF.Eval}(0, K^{\mathsf{fss}}_{0,i,j}, x_{i,j})$.
  2. Compute $c_0 = \sum_{i=1}^{D} \sum_{j=1}^{w} y_{i,j} \cdot c_{0,i,j}$.
  3. Output (a, c_0).
- If $\sigma = 1$:
  1. For all $j \in [w]$ and $i \in [D]$: Let $c_{1,i,j} \leftarrow \mathsf{DPF.Eval}(1, K^{\mathsf{fss}}_{1,i,j}, x_{i,j})$.
  2. Compute $c_1 = \sum_{i=1}^{D} \sum_{j=1}^{w} y_{i,j} \cdot c_{1,i,j}$.
  3. Let $b = F_K(x)$.
  4. Output (b, c_1).

Figure 8: PCF for vector oblivious linear evaluation over a ring R based on our weak PRF F_K from rVDLPN over R and distributed point function DPF.

where $f_{K_{i,j},L_{i,j}}$ is the point function with $f_{K_{i,j},L_{i,j}}(x_{i,j}) = L_{i,j}$ if and only if $K_{i,j} = x_{i,j}$. Now, we can share $F_{K,L}(x)$ as the sum of $w \cdot D$ distributed point functions, where $f_{K_{i,j},L_{i,j}}$ has input space $\{0,1\}^i$. Let $\kappa = w \cdot D(D+1)/2$. Note that any scalar multiple of a point function is a point function again, and thus DPFs are, in particular, an FSS for $\{a \cdot F_{K,L}\}_{a\in R,K\in\{0,1\}^\kappa,L\in R}$ and $\{F_{K,L}\}_{K\in\{0,1\}^\kappa,L\in R}\cup\{0\}$. Therefore, by the results of Section 5, they are sufficient to obtain a PCF for VOLE, OT and as we show now, even a universal PCF for general constant-degree correlations.

In the following, by $\mathsf{negl}(\lambda)$ we always refer to a negligible function, that is a function that decreases faster than any inverse polynomial: $\mathsf{negl}(\lambda) \le 1/p(\lambda)$ for all polynomials p and all large enough $\lambda \in \mathbb{N}$.

Theorem 6.5 (PCF for VOLE from rVDLPN) *If* $\mathsf{DPF} = (\mathsf{DPF.Gen}, \mathsf{DPF.Eval})$ *is instantiated with the distributed point function from [GI14], then, assuming* $\mathsf{rVDLPN}(w, D, 2^D)$ *over ring* R, $\mathsf{PCF} = (\mathsf{PCF.Gen}, \mathsf{PCF.Eval})$ *as defined in Figure 8 is a* $(2^D, \mathsf{negl}(\lambda))$*-secure PCF for VOLE over* R *with the following complexities:*

PCF for OT from regular VDLPN

Let from $\mathsf{DPF} = (\mathsf{DPF.Gen}, \mathsf{DPF.Eval})$ an FSS scheme for point functions with variable input length. Recall that $f_{K,s}\colon \{0,1\}^i \to \{0,1\}^\lambda$ is the point function with $f_{K,s}(x) = s$ if and only if $x = K$.

$\mathsf{PCF.Gen}(1^\lambda)$**:** On input 1^λ:

1. Sample $s \leftarrow \{0,1\}^\lambda$.
2. For all $j \in [w]$ and $i \in [D]$:
   - Sample a key $K_{i,j} \leftarrow \{0,1\}^i$.
   - Sample a pair of FSS keys $(K^{\mathsf{fss}}_{0,i,j}, K^{\mathsf{fss}}_{1,i,j}) \leftarrow \mathsf{DPF.Gen}(1^\lambda, i, f_{K_{i,j},s})$.
3. Set $K := \{K_{i,j}\}_{j\in[w],i\in[D]}$.
4. Output $\mathsf{k}_0 = (\{K^{\mathsf{fss}}_{0,i,j}\}_{j\in[w],i\in[D]}, s)$ and $\mathsf{k}_1 = (\{K^{\mathsf{fss}}_{1,i,j}\}_{j\in[w],i\in[D]}, K)$.

$\mathsf{PCF.Eval}(\sigma, \mathsf{k}_\sigma, x)$**:** On input a random $x \in \{0,1\}^{w\cdot D(D+1)/2}$:
Let x' be the first λ bits of x. Parse x as $x_{i,j} \in \{0,1\}^i$, for $j \in [w], i \in [D]$.

- If $\sigma = 0$:
  1. For all $j \in [w], i \in [D]$: Let $q_{i,j} \leftarrow \mathsf{DPF.Eval}(0, K^{\mathsf{fss}}_{0,i,j}, x_{i,j})$.
  2. For all $k \in [\lambda]$: Compute $q = \bigoplus_{i=1}^{D} \bigoplus_{j=1}^{w} q_{i,j}$.
  3. Compute $y_0 = \mathsf{H}(x' \oplus q), y_1 = \mathsf{H}(x' \oplus q \oplus s)$.
  4. Output (y_0, y_1).
- If $\sigma = 1$:
  1. Let $c = F_K(x)$.
  2. For all $j \in [w], i \in [D]$: Let $t_{i,j} \leftarrow \mathsf{DPF.Eval}(1, K^{\mathsf{fss}}_{1,i,j}, x_{i,j})$.
  3. Compute $t = \bigoplus_{i=1}^{D} \bigoplus_{j=1}^{w} t_{i,j}$.
  4. Compute $z = \mathsf{H}(x' \oplus t)$.
  5. Output (c, z).

Figure 9: PCF for oblivious transfer based on our weak PRF F_K from rVDLPN and distributed point function DPF.

- *Each party's key is of size* $\mathcal{O}(w \cdot D^2 \cdot \lambda + w \cdot D \cdot \log|R|)$ *bits,*
- *the cost of* $\mathsf{PCF.Gen}$ *and* $\mathsf{PCF.Eval}$ *is dominated by each* $\mathcal{O}(w \cdot D^2 + w \cdot D \cdot \log|R|/\lambda)$ *invocations of a pseudorandom generator.*

Note that in the following, unlike the generic construction for OT from Section 5, we use the optimization mentioned in Remark 5.6, where the DPFs have output in $\{0,1\}^\lambda$ (but the construction is still based on the regular VDLPN assumption over $\mathbb{F}_2$). For $w(\lambda) \in \mathcal{O}(\lambda)$ this gives us the following:

Theorem 6.6 (PCF for OT from rVDLPN) *If* $\mathsf{DPF} = (\mathsf{DPF.Gen}, \mathsf{DPF.Eval})$ *is instantiated with the distributed point function from [GI14] and* H *is a correlation-robust hash function, then, assuming that* $\mathsf{rVDLPN}(\mathcal{O}(\lambda), D, 2^D)$ *holds,* $\mathsf{PCF} = (\mathsf{PCF.Gen}, \mathsf{PCF.Eval})$ *as defined in Figure 9 is a* $(2^D, \mathsf{negl}(\lambda))$*-secure PCF for OT with the following complexities:*

- *Each party's key is of size* $\mathcal{O}(\lambda^2 \cdot D^2)$ *bits,*

- *the cost of* PCF.Gen *and* PCF.Eval *is dominated by each* $\mathcal{O}(\lambda \cdot D^2)$ *invocations of a pseudorandom generator.*

For our final construction, a PCF for general degree-2 correlations, first note that we have:

$$F_K(x) \cdot F_K(y) = \bigoplus_{i,i'=1}^{D} \bigoplus_{j,j'=1}^{w} f_{K_{i,j}}(x_{i,j}) \wedge f_{K_{i',j'}}(y_{i',j'}),$$

where $f_{K_{i,j}}(x_{i,j}) \wedge f_{K_{i',j'}}(y_{i',j'})$ if and only if $K_{i,j} = x_{i,j}$ and $K_{i',j'} = y_{i',j'}$. We can thus rewrite the above as

$$F_K(x) \cdot F_K(y) = \bigoplus_{i,i'=1}^{D} \bigoplus_{j,j'=1}^{w} f_{K_{i,j} \otimes K_{i',j'}}(x_{i,j} \otimes y_{i',j'}).$$

Based on this observations, in Figure 10 we give a PCF for an arbitrary degree-2 correlation. More precisely, after a one-time setup, this universal PCF allows the parties to compute an arbitrary number degree-2 correlations of the form $((X_0, Y_0), (X_1, Y_1))$, where $X = X_0 \oplus X_1 \in \{0,1\}^m$ and $Y_0 + Y_1 = p(X)$ for an arbitrary m-variate polynomial p. Note that the same one-time setup can be used to evaluate many different polynomials. This construction can be generalized to arbitrary degree-d correlation for arbitrary constant d, yielding the following:

Theorem 6.7 (Universal PCF for degree-d correlations from rVDLPN) *If* DPF *is instantiated with the distributed point function from [GI14] and assuming* $\mathsf{rVDLPN}(w, D, 2^D)$ *holds, then* $\mathsf{PCF} = (\mathsf{PCF.Gen}, \mathsf{PCF.Eval})$ *as given in Fig. 10 is a universal* $(2^D, \mathsf{negl}(\lambda))$*-secure PCF for degree-$d$ correlations over* $\mathbb{F}_2$ *with the following complexities:*

- *Each party's key is of size* $\mathcal{O}(\lambda^{d+1} \cdot D^{2d})$ *bits,*
- *the cost of* PCF.Gen *is dominated by* $\mathcal{O}(\lambda^d \cdot D^{2d})$ *invocations of a pseudorandom generator,*
- *the cost of eval* PCF.Eval *of evaluating a degree-d polynomial m-variate polynomial consisting of at most k terms is dominated by* $\mathcal{O}((m+k) \cdot \lambda^d \cdot D^{2d})$ *invocations of a pseudorandom generator.*

6.5 Application: XOR-RKA Secure PRGs and Weak PRFs

Our WPRF candidates give rise to simple PRG and WPRF constructions that are naturally secure against related key attacks for XOR relations. Related key attacks [BK03] are a powerful class of attack, where the adversary is given access to outputs of the cryptographic primitive under several keys which are related, according to some relation known to the adversary. In the model of *XOR-RKA security*, the adversary is given samples under a key K, as well as under several related keys $K \oplus \Delta^{(1)}, K \oplus \Delta^{(2)}, \ldots$, where $\Delta^{(i)}$ are public offsets. This naturally models bit flips that may occur in hardware due to, for instance, fault attacks.

Correlation robust hash function. Recall the notion of a correlation robust hash function (Definition 5.4), a function H such that the distribution of $\mathsf{H}(K \oplus \Delta^{(1)}), \ldots, \mathsf{H}(K \oplus \Delta^{(N)})$, where $\Delta^{(i)}$ are random strings, is computationally indistinguishable from the uniform distribution, even when given the $\Delta^{(i)}$'s. Note that this can also be seen as a form of XOR-RKA secure pseudorandom generator [AHI11].

As seen in Section 6.2, our WPRF candidate can be written as

$$F_K(x) = F(K \oplus x)$$

for some public function F. Assuming that F_K is a secure WPRF for N samples, we immediately get that F is a correlation robust function. As shown in [AHI11], this also implies the existence of

Universal PCF for degree-2 polynomials

Let $\mathsf{DPF} = (\mathsf{DPF.Gen}, \mathsf{DPF.Eval})$ be an FSS scheme for point functions with variable input length. Recall that $f_K \colon \{0,1\}^i \to \{0,1\}$ is the point function with $f_K(x) = 1$ if and only if $x = K$. Note that the following gives a PCF for an arbitary m-variate degree-2 polynomial $p(X_1, \ldots, X_m) = c \oplus \bigoplus_{\alpha=1}^{m} \ell_\alpha X_\alpha \oplus \bigoplus_{\alpha,\beta=1}^{m} q_{\alpha,\beta} X_\alpha X_\beta$ (which does not have to be fixed at the time of key generation).

$\mathsf{PCF.Gen}(1^\lambda)$**:** On input 1^λ:

1. For all $j \in [w]$ and $i \in [D]$:
   - Sample a key $K_{i,j} \leftarrow \{0,1\}^i$.
   - Sample a pair of FSS keys $(K^{\mathsf{fss}}_{0,i,j}, K^{\mathsf{fss}}_{1,i,j}) \leftarrow \mathsf{DPF.Gen}(1^\lambda, i, f_{K_{i,j}})$.
2. For all $j, j' \in [w]$ and $i, i' \in [D]$:
   - Sample a pair of FSS keys $(K^{\mathsf{fss}}_{0,i,j,i',j'}, K^{\mathsf{fss}}_{1,i,j,i',j'}) \leftarrow \mathsf{DPF.Gen}(1^\lambda, i \cdot j, f_{K_{i,j} \otimes K_{i',j'}})$.
3. Set $K^1_\sigma := \{K^{\mathsf{fss}}_{\sigma,i,j}\}_{j \in [w], i \in [D]}$ and $K^2_\sigma := \{K^{\mathsf{fss}}_{\sigma,i,j,i',j'}\}_{j,j' \in [w], i,i' \in [D]}$.
4. Output the keys $\mathsf{k}_0 = (K^1_0, K^2_0)$ and $\mathsf{k}_1 = (K^1_1, K^2_1)$.

$\mathsf{PCF.Eval}(\sigma, \mathsf{k}_\sigma, p, x_1, \ldots, x_m)$**:** On input a random $x_1, \ldots, x_m \in \{0,1\}^{w \cdot D(D+1)/2}$:
Parse x_α as $x_{\alpha,i,j} \in \{0,1\}^i$, for $j \in [w], i \in [D]$ and each $\alpha \in [m]$.

- If $\sigma = 0$:
  1. For all $\alpha \in [m]$:
     - For all $j \in [w]$ and $i \in [D]$: Let $X_{\alpha,0,i,j} \leftarrow \mathsf{DPF.Eval}(0, K^{\mathsf{fss}}_{0,i,j}, x_{\alpha,i,j})$.
     - Compute $X_{\alpha,0} = \bigoplus_{i=1}^{D} \bigoplus_{j=1}^{w} X_{\alpha,0,i,j}$.
  2. For all $\alpha, \beta \in [m]$ with $q_{\alpha,\beta} \neq 0$:
     - For all $j, j' \in [w]$ and $i, i' \in [D]$:
       Let $X_{\alpha,\beta,0,i,j,i',j'} \leftarrow \mathsf{DPF.Eval}(0, K^{\mathsf{fss}}_{0,i,j,i',j'}, x_{\alpha,i,j} \otimes x_{\beta,i',j'})$
     - Compute $X_{\alpha,\beta,0} = \bigoplus_{i,i'=1}^{D} \bigoplus_{j,j'=1}^{w} X_{\alpha,\beta,0,i,j,i',j'}$.
  3. Compute $Y_0 = \sum_{\alpha=1}^{m} \ell_\alpha X_{\alpha,0} \oplus \sum_{\alpha,\beta=1}^{m} q_{\alpha,\beta} X_{\alpha,\beta,0}$.
  4. Output $(\{X_{\alpha,0}\}_{\alpha \in [m]}, Y_0)$.
- If $\sigma = 1$:
  1. For all $\alpha \in [m]$:
     - For all $j \in [w]$ and $i \in [D]$: Let $X_{\alpha,1,i,j} \leftarrow \mathsf{DPF.Eval}(1, K^{\mathsf{fss}}_{1,i,j}, x_{\alpha,i,j})$.
     - Compute $X_{\alpha,1} = \bigoplus_{i=1}^{D} \bigoplus_{j=1}^{w} X_{\alpha,1,i,j}$.
  2. For all $\alpha, \beta \in [m]$ with $q_{\alpha,\beta} \neq 0$:
     - For all $j, j' \in [w]$ and $i, i' \in [D]$:
       Let $X_{\alpha,\beta,1,i,j,i',j'} \leftarrow \mathsf{DPF.Eval}(1, K^{\mathsf{fss}}_{1,i,j,i',j'}, x_{\alpha,i,j} \otimes x_{\beta,i',j'})$
     - Compute $X_{\alpha,\beta,1} = \bigoplus_{i,i'=1}^{D} \bigoplus_{j,j'=1}^{w} X_{\alpha,\beta,1,i,j,i',j'}$.
  3. Compute $Y_1 = \sum_{\alpha=1}^{m} \ell_\alpha X_{\alpha,1} \oplus \sum_{\alpha,\beta=1}^{m} q_{\alpha,\beta} X_{\alpha,\beta,1}$.
  4. Output $(\{X_{\alpha,1}\}_{\alpha \in [m]}, Y_1)$.

Figure 10: Universal PCF for degree-2 polynomials over $\mathbb{F}_2$ based on our weak PRF F_K from rVDLPN and distributed point function DPF.

$\underline{\mathsf{Exp}^{\mathsf{rka\text{-}wprf}}_{\mathcal{A}}(\lambda)}$:
$K \xleftarrow{\$} \{0,1\}^\lambda$
$b \xleftarrow{\$} \{0,1\}$
$b^\star \leftarrow \mathcal{A}^{\mathcal{O}^{\mathsf{rka}}_{K,b}(\cdot)}(1^\lambda)$
if $b = b^\star$
  return 1
else return 0

$\underline{\mathcal{O}^{\mathsf{rka}}_{K,b}(\Delta)}$:
$x \xleftarrow{\$} \{0,1\}^{n(\lambda)}$
if $b = 0$
  $y \xleftarrow{\$} \{0,1\}^\lambda$
else
  $y \leftarrow F_{K \oplus \Delta}(x)$
return (x, y)

Figure 11: XOR-RKA security experiment for a weak PRF

RKA-secure one-time symmetric encryption and RKA-secure deterministic encryption schemes with optimal ciphertext sizes, for XOR relations. Previously, such constructions were only known to exist for less natural, additive relations over $\mathbb{Z}_p$ under a power-DDH assumption [AHI11] or the learning with rounding assumption [AW14], or for XOR relations based on multilinear maps [ABP19].

RKA-Secure WPRF. As well as giving an RKA-secure PRG, our construction is in fact an XOR-RKA secure WPRF. We consider the definition of RKA-secure WPRF by Bellare et al. [BCM11] given below, adapted to the XOR relation. This allows the adversary to choose an arbitrary Δ for each query, which is added to the fixed key K in the real experiment where the adversary learns PRF evaluations.

Definition 6.8 *Let $\{F_K : \{0,1\}^n \to \{0,1\}^\lambda\}_{K \in \{0,1\}^\lambda}$ be a family of efficiently computable functions. We say that F is an N-*XOR-RKA-secure weak PRF *if for every p.p.t. adversary $\mathcal{A}$ making at most $N(\lambda)$ queries to the oracle $\mathcal{O}^{\mathsf{rka}}_{K,b}(\cdot)$, it holds that*

$$\left| \Pr[\mathsf{Exp}^{\mathsf{rka\text{-}wprf}}_{\mathcal{A}}(\lambda) = 1] - \frac{1}{2} \right| \leq \mathsf{negl}(\lambda)$$

where $\mathsf{Exp}^{\mathsf{rka\text{-}wprf}}_{\mathcal{A}}(\lambda)$ is as defined in Figure 11.

Note that since each query samples a fresh random x, allowing the adversary to choose an arbitrary Δ induces exactly the same output distribution as with no offset. Therefore, we immediately obtain the following.

Theorem 6.9 *Suppose the $\mathsf{rVDLPN}(w, D, N)$ assumption holds for some $w(\lambda), D(\lambda), N(\lambda)$. Then, there is an XOR-RKA-secure weak PRF for $N(\lambda)$ queries.*

7 Security Analysis

In this section, we provide a thorough analysis of our candidate weak PRF family against various classes of attacks.

7.1 Resistance Against Linear Tests – Theorem and Corollaries

Our first and main result shows that our candidate weak PRF fools all *linear tests*. Recall the notations of bias from Definition 3.1 and $\mathcal{H}_{\mathsf{par}}, \mathcal{O}_{\mathsf{par}}$ from Section 6.1. More precisely, we prove:

Theorem 7.1 (Low-Bias) *There exist constants (β, μ, ν) such that for any large enough w, any $D \leq \beta \cdot w$, $n \leftarrow 2^D$, $\mathsf{par} \leftarrow (w, D, N)$, it holds that*

$$\Pr_{H \xleftarrow{\$} \mathcal{H}_{\mathsf{par}}} [\mathsf{bias}(\mathcal{O}_{\mathsf{par}}(H)) > \mu^w] \leq \nu^w.$$

In the language of weak PRFs, the above theorem states that, with overwhelming probability (at least $1-\nu^w$) over the choice of at most $N = 2^D$ random inputs $(x^{(1)}, \cdots, x^{(N)})$, any distinguisher that computes a linear function of the entire output string $\vec{y} = (F_K(x^{(1)}), \cdots, F_K(x^{(N)}))$ has advantage at most μ^w. Note that the choice of the linear function can depend *arbitrarily* on $(x^{(1)}, \cdots, x^{(N)})$. We formalize this by putting forth the notion of (ε, δ, N)-biased weak PRF family:

Definition 7.2 ((ε, δ, N)-biased weak PRF family) *A function family*

$$\{F_K : \mathbb{F}_2^{n(\lambda)} \mapsto \mathbb{F}_2\}_{K \in \mathbb{F}_2^{\kappa(\lambda)}}$$

is (ε, δ, N)-biased if for every large enough $\lambda \in \mathbb{N}$, it holds that

$$\Pr_{x^{(1)}, \cdots, x^{(N(\lambda))} \xleftarrow{\$} \mathbb{F}_2^{n(\lambda)}} [\mathsf{bias}(\mathcal{D}_{\lambda,N}(\vec{x})) > \varepsilon(\lambda)] < \delta(\lambda),$$

where $\mathcal{D}_{\lambda,N}(\vec{x}))$ denotes the distribution which samples $K \xleftarrow{\$} \mathbb{F}_2^{\kappa(\lambda)}$ and outputs $\vec{y} = (F_K(x^{(1)}), \cdots, F_K(x^{(N)}))$.

The task of building low-complexity primitives which provably fool all linear tests is a very active area of research, especially in the context of building low-complexity *pseudorandom generators* with superlinear stretch [MST03, Shp09]. Pseudorandom generators that fool all linear tests are called ε-biased pseudorandom generators, and their complexity is well understood: they exist in a complexity class as low as NC^0_5, the class of constant-depth fan-in-2 boolean circuits where each output depends on at most 5 input bits, and cannot exist in NC^0_4 [MST03]. Regarding the complexity of building low-bias strong PRFs, we are only aware of two results [GV04, Hea08] which show that negligible-bias strong PRFs exist in $\mathsf{AC}^0[\oplus]$. The construction is obtained by relatively involved constructions in $\mathsf{AC}^0[\oplus]$ of the various components required in the ε-biased sample space construction of Naor and Naor [NN90]; although it achieves exponentially small bias, it cannot be a candidate exponentially strong PRF, since no such PRF can exist in $\mathsf{AC}^0[\oplus]$ [RR97, CIKK16]. We obtain an incomparable result of independent interest, by showing that a weaker object (an unconditional weak PRF with exponentially small bias) exists in a much smaller complexity class, the class $\mathsf{XOR} \circ \mathsf{AND}$, a subclass of depth-2 $\mathsf{AC}^0[\oplus]$.

Corollary 7.3 *There exist constants (β, μ, ν) such that for any large enough $w(\lambda)$, any $D(\lambda) \leq \beta \cdot w$, there exists a $(\mu^w, \nu^w, 2^D)$-biased weak PRF family in the class $\mathsf{XOR} \circ \mathsf{AND}$ of polynomial-size depth-2 circuits with a layer of AND gates at the bottom, and a single XOR gate at the top.*

Resistance Against Standard Attacks. Our candidate weak PRF relies on a variant of the LPN assumption. A large number of attacks against the LPN assumption have been introduced. The main categories of attacks include Gaussian elimination attacks and its variants [EKM17], statistical decoding attacks [AJ01, Ove06, FKI06, DAT17, Zic17], information set decoding attacks [Pra62, Ste88, FS09, BLP11, MMT11, BJMM12, MO15], and BKW and its variants [BKW00, Lyu05]. Yet another attack against PRF candidates based on LPN-style assumption can be mounted whenever the candidate PRF family satisfies some correlation with a sufficiently low-degree polynomial [ABG+14, BR17].

We will not provide the details of these many attacks in this section. Rather, we confine ourselves to observing that *all the above attacks can be formulated as linear tests in our framework.* Therefore, Theorem 7.1 implies that an adversary running in time $\mathsf{poly}(\lambda)$ and performing any of the above attacks can succeed at distinguishing our candidate's output from random with advantage at most $\mathsf{poly}(\lambda) \cdot \max(\mu^{w(\lambda)}, \nu^{w(\lambda)})$.

Note that this is not a contradiction to the BKW attack [BKW00, Lyu05] of complexity $2^{\mathcal{O}(N/\log N)}$ on Learning Parity with Noise, as the N in the BKW attack corresponds to the size of the secret, which in our case is always at least quadratic in the security parameter.

Resistance Against Polynomial Tests. Recall that our candidate weak PRF, with parameters $\mathsf{par} = (w, D, N)$, is of the form

$$F_K(x) = \bigoplus_{i=1}^{D} \bigoplus_{j=1}^{w} \bigwedge_{k=1}^{i} (x_{i,j,k} \oplus K_{i,j,k}),$$

with input size and key size $n = \kappa = w \cdot D(D-1)/2$. For any divisor d of w, this can be rewritten as

$$\begin{aligned} F_K(x) &= \bigoplus_{i=1}^{D} \bigoplus_{\ell=1}^{d} \bigoplus_{j=0}^{w/d-1} \bigwedge_{k=1}^{i} (x^{(\ell)}_{i,j,k} \oplus K^{(\ell)}_{i,j,k}) \text{ where } x^{(\ell)}_{i,j,k} \leftarrow x_{i,jd+\ell,k}, K^{(\ell)}_{i,j,k} \leftarrow K_{i,jd+\ell,k} \\ &= \bigoplus_{\ell=1}^{d} F_{K^{(\ell)}}(x^{(\ell)}) \end{aligned}$$

where each $K^{(\ell)}, x^{(\ell)}$ belongs to $\{0,1\}^{n/d} = \{0,1\}^{\kappa/d}$. That is, our candidate weak PRF with parameters (w, D, N) can be rewritten as the XOR of d independent instances (on independent keys and inputs) with parameters $(w/d, D, N)$. By a result of Viola [Vio09], the XOR of d low-bias weak PRFs over $\mathbb{F}_2$ fools all degree-d multivariate polynomial tests over $\mathbb{F}_2$, hence we can show that our candidate further resists all *low-degree polynomial tests*. We formalize this observation below.

We say that a distribution $\mathcal{D}$ over $\{0,1\}^N$ has *degree-d polynomial bias at most ε*, and write $\mathsf{polybias}_d(\mathcal{D}) \leq \varepsilon$, if for every degree-$d$ multivariate polynomial $P : \mathbb{F}_2^N \mapsto \mathbb{F}_2$, it holds that

$$\left| \mathop{\mathbb{E}}_{\mathcal{D}} \left[(-1)^{P(\mathcal{D})} \right] - \mathop{\mathbb{E}}_{U_N} \left[(-1)^{P(U_N)} \right] \right| \leq \varepsilon,$$

where U_N is the uniform distribution over $\{0,1\}^N$. Combining our above observation with the result of Viola, we get the following corollary:

Corollary 7.4 (Resistance to Polynomial Tests) *For any $c > 1$, there exist constants (β, μ, ν) (the same as in Theorem 7.1) such that for any large enough w, any divisor d of w, any $D \leq \beta \cdot w/d$, it holds that*

$$\Pr_{H \xleftarrow{\$} \mathcal{H}_{\mathsf{par}}} \left[\mathsf{polybias}_d(\mathcal{O}_{\mathsf{par}}(H)) > 16 \cdot \mu^{w/2^{d-1}} \right] \leq d \cdot \nu^{w/d}.$$

We further note that this result can be directly plugged into the result of [BDVY13]: pseudorandomness against degree-d polynomials implies pseudorandomness against branching programs of width-2 and polynomial length that read d bits of input at a time. Hence, our weak PRF candidate also fools all tests from this class (with probability $16 \cdot t \cdot \mu^{w/2^{d-1}}$ for width-2 branching programs of length t reading d inputs at a time); this captures any degree-d polynomial, but also space-bounded computation with one bit of memory, or d-DNF formula.

7.2 Proof of Resistance Against Linear Tests

In this section, we prove Theorem 7.1. We first set up a few notations: for any $i \leq D$, it follows from the definition of $\mathcal{H}^i_{\mathsf{par}}$ (Section 6.1) that a matrix H_i sampled from $\mathcal{H}^i_{\mathsf{par}}$ can be written as $H_i = H_{i,1} || \cdots || H_{i,w}$, with $H_{i,j} \in \mathbb{F}_2^{N \times 2^i}$, where each row of each matrix $H_{i,j}$ for $j \leq w$ is sampled independently from $\mathcal{S}_{1,2^i}$ (resulting in rows of regular weight w). We let $\mathcal{M}^i_{\mathsf{par}}$ denote the distribution of each $H_{i,j}$ (hence, a sample H_i from $\mathcal{H}^i_{\mathsf{par}}$ is a concatenation of w independent samples from $\mathcal{M}^i_{\mathsf{par}}$).

Definition 7.5 *Given a matrix $M \in \mathbb{F}_2^{N \times 2^i}$, we say that M is* bad *with respect to a vector $\vec{v} \in \mathbb{F}_2^N$ if*

$$\mathcal{HW}\left(\vec{v}^{\intercal} \cdot M\right) \notin \left[\frac{2^i}{5}, \frac{2^{i+2}}{5}\right].$$

Furthermore, given w matrices $(M_1, \cdots, M_w)$ in $\mathbb{F}_2^{N \times 2^i}$, we let $N_{\vec{v}}(M_1, \cdots, M_w)$ denote the number of matrices which are bad against $\vec{v}$ among $M_1 \cdots M_w$.

Equipped with the above definition, we can state the lemma which is at the core of the proof of Theorem 7.1:

Lemma 7.6 *There is a constant C such that for any $1 \leq i \leq D$, and for any vector $\vec{v} \in \mathbb{F}_2^N$ such that $\mathcal{HW}(\vec{v}) \in [2^{i-1}, 2^i]$, it holds that*

$$\Pr_{M_1, \cdots, M_w \xleftarrow{\$} \mathcal{M}^i_{\mathsf{par}}}\left[N_{\vec{v}}(M_1, \cdots, M_w) \geq \frac{w}{2}\right] \leq 2^{-C \cdot 2^i \cdot w}.$$

Proof. The proof of Lemma 7.6 relies on the bounded difference inequality (Lemma 3.5). Fix a vector $\vec{v}$ of weight $\mathcal{HW}(\vec{v}) = \ell \in [2^{i-1}, 2^i]$. It will be helpful to reformulate the event "a sample M from $\mathcal{M}^i_{\mathsf{par}}$ is bad" as a balls and bins problem. Consider a sample M from $\mathcal{M}^i_{\mathsf{par}}$. Recall that by definition of $\mathcal{M}^i_{\mathsf{par}}$, the rows of M are sampled independently from $\mathcal{S}_{1,2^i}$. We can cast the problem as a balls and bins problem as follows: we start with 2^i empty bins, correspondings to the columns of M. Sampling a row of M from $\mathcal{S}_{1,2^i}$ can be viewed as throwing a ball in one of the 2^i bins, picked uniformly at random. Then, for any fixed $\vec{v}$ of weight ℓ, the event

$$\mathcal{HW}\left(\vec{v}^{\intercal} \cdot M\right) \notin I_i \leftarrow \left[\frac{2^i}{5}, \frac{2^{i+2}}{5}\right]$$

is equivalent to the following event: after throwing exactly ℓ balls at random in 2^i bins, the numbers T of bins containing an odd number of balls satisfies $T \notin I_i$. The full experiment can therefore be reformulated as follows: there are 2^i bins, and $\ell \cdot w$ balls are thrown randomly (and sequentially) in these bins, divided into w consecutive phases (ℓ balls are thrown during each phase). Each time ℓ balls have been thrown, we check whether the fraction of bins containing an odd number of balls is between $1/5$ and $4/5$, and empty the bins. At the end of the experiment, we output "failure" if at least $w/2$ of the w checks failed.

We now bound the probability that the experiment fails. For $j = 1$ to ℓ and $k = 1$ to w, let $X_{j,k}$ be the random variable corresponding to the bin in which the k-th ball of the j-th phase landed (note that the $X_{j,k}$ are independent). In order to apply McDiarmid's bounded difference inequality, we need to consider a carefully chosen function of the $X_{j,k}$:

$$\Phi(X_{1,1}, \cdots, X_{\ell,w}) = \sum_{k=1}^{w}\left(2^{i-1} - \left|\mathcal{HW}\left(\bigoplus_{j=1}^{\ell} X_{j,k}\right) - 2^{i-1}\right|\right).$$

Claim 7.7

$$\Pr_{M_1, \cdots, M_w \xleftarrow{\$} \mathcal{M}^i_{\mathsf{par}}}\left[N_{\vec{v}}(M_1, \cdots, M_w) \geq \frac{w}{2}\right] \leq \Pr\left[\Phi(X_{1,1}, \cdots, X_{\ell,w}) < \frac{w \cdot 2^i}{10}\right].$$

Proof. Fix a vector $\vec{v}$ of weight ℓ. Note that sampling a matrix $M \leftarrow \mathcal{M}^i_{\mathsf{par}}$ and computing $\mathcal{HW}(\vec{v}^{\intercal} \cdot M)$ is identical to sampling ℓ random unit vectors $X_1 \cdots X_\ell$ and computing $\mathcal{HW}(\bigoplus_{j=1}^{\ell} X_j)$. Now, suppose that the event $\Phi(X_{1,1}, \cdots, X_{\ell,w}) < w \cdot 2^i/10$ happened. This means that there must exist a subset $K \subseteq [w]$ such that

- For every $k \in K$, $Y_k \leftarrow 2^{i-1} - \left|\mathcal{HW}\left(\bigoplus_{j=1}^{\ell} X_{j,k}\right) - 2^{i-1}\right| < \frac{2^i}{5}$, and
- $|K| \geq \frac{w}{2}$.

Indeed, if this was not the case then at least $w - w/2 = w/2$ of the $k \in [w]$ would satisfy $Y_k \geq 2^i/5$, which would imply $\Phi(X_{1,1}, \cdots, X_{\ell,w}) \geq (w/2) \cdot (2^i/5)$, contradicting the hypothesis. The condition $Y_k < 2^i/5$ rewrites to

$$\left|\mathcal{HW}\left(\bigoplus_{j=1}^{\ell} X_{j,k}\right) - 2^{i-1}\right| > \frac{3 \cdot 2^{i-1}}{5}$$
$$\iff \mathcal{HW}\left(\bigoplus_{j=1}^{\ell} X_{j,k}\right) \notin \left[\frac{2^i}{5}, \frac{2^{i+2}}{5}\right].$$

Therefore, the event $\Phi(X_{1,1}, \cdots, X_{\ell,w}) < w \cdot 2^i/10$ implies in particular that at least $w/2$ of the $k \leq w$ satisfy $\mathcal{HW}\left(\bigoplus_{j=1}^{\ell} X_{j,k}\right) \notin \left[2^i/5, 2^{i+2}/5\right]$, which is identical to the event $N_{\vec{v}}(M_1, \cdots, M_w) \geq w/2$ happening when sampling $M_1 \cdots M_w$ from $\mathcal{M}^i_{\mathsf{par}}$. This concludes the proof of the claim. □

Claim 7.8 *The function Φ is 2-Lipschitz.*

Proof. Consider changing a single input $X_{j,k}$ to Φ. This corresponds to moving a single ball to a different bin during one of the phases. But doing so can only change the number of bins with an odd number of balls at the end of this phase by at most 2, hence the value $\mathcal{HW}(\bigoplus_{j=1}^{\ell} X_{j,k})$ can change by at most 2. Propagating the change, this implies that the value of $\Phi(X_{1,1}, \cdots, X_{\ell,w})$ changes by at most 2. □

Since Φ is 2-Lipschitz, we can apply the bounded difference inequality (Lemma 3.5): for any t, we have

$$\Pr[\Phi(X_{1,1}, \cdots, X_{\ell,w}) < \mathbb{E}[\Phi(X_{1,1}, \cdots, X_{\ell,w})] - t] \leq \exp\left(-\frac{t^2}{2\ell w}\right).$$

It remains to bound $\mathbb{E}[\Phi(X_{1,1}, \cdots, X_{\ell,w})]$.

Claim 7.9

$$\mathbb{E}\left[\Phi\left(X_{1,1}, \cdots, X_{\ell,w}\right)\right] \geq \frac{w \cdot 2^i}{5}.$$

Proof. First, we prove that for any $1 \leq k \leq w$ (i.e., for each phase of ℓ balls thrown),

$$\mathbb{E}\left[\mathcal{HW}\left(\bigoplus_{j=1}^{\ell} X_{j,k}\right)\right] \in \left[\frac{2^i}{5}, \frac{2^i}{2}\right].$$

Let $R_{2^i,\ell} \leftarrow \mathcal{HW}\left(\bigoplus_{j=1}^{\ell} X_{j,k}\right)$. We compute the expected value $\mathbb{E}[R_{2^i,\ell}]$ of $R_{2^i,\ell}$, by induction over ℓ. Clearly, $\mathbb{E}[R_{2^i,0}] = 0$. Furthermore, by linearity of the expectation,

$$\mathbb{E}[R_{2^i,\ell+1}] = (\mathbb{E}[R_{2^i,\ell}] - 1) \cdot \frac{\mathbb{E}[R_{2^i,\ell}]}{2^i} + (\mathbb{E}[R_{2^i,\ell}] + 1) \cdot \left(1 - \frac{\mathbb{E}[R_{2^i,\ell}]}{2^i}\right),$$

since adding a ball in a bin which contains initially an odd number of balls reduces the number of bins with an odd number of balls by 1, while adding a ball in a bin with an even number of balls increases the number by 1. Solving the recursive formula gives

$$\mathbb{E}[R_{2^i,\ell}] = \frac{2^i}{2} \cdot \left(1 - \left(1 - \frac{2}{2^i}\right)^\ell\right) \in \left[\frac{2^i}{5}, \frac{2^i}{2}\right] \text{ for any } \ell \geq 2^{i-1},$$

where the inclusion uses the following standard inequality:

$$\left(1 - \frac{2}{2^i}\right)^\ell \leq \exp\left(-\frac{2\ell}{2^i}\right) < \frac{2}{5} \text{ for any } \ell \geq 2^{i-1}.$$

From there, we get

$$\mathbb{E}\left[\left|\mathcal{HW}\left(\bigoplus_{j=1}^{\ell} X_{j,k}\right) - 2^{i-1}\right|\right] \leq \frac{3}{10} \cdot 2^i$$
$$\implies \mathbb{E}\left[\Phi\left(X_{1,1}, \cdots, X_{\ell,w}\right)\right] \geq \frac{w \cdot 2^i}{5}.$$

□

Summing up, we obtain that for any t,

$$\Pr\left[\Phi(X_{1,1}, \cdots, X_{\ell,w}) < \frac{w \cdot 2^i}{5} - t\right] \leq \exp\left(-\frac{t^2}{2\ell w}\right) \leq \exp\left(-\frac{t^2}{2^{i+1} w}\right),$$

where the second inequality uses the fact that $\ell \leq 2^i$. Plugging in $t = w \cdot 2^i/10$, this gives

$$\Pr\left[\Phi(X_{1,1}, \cdots, X_{\ell,w}) < \frac{w \cdot 2^i}{10}\right] \leq \exp\left(-\frac{w \cdot 2^i}{200}\right) = 2^{-C \cdot w \cdot 2^i}$$

for some appropriate constant C; this concludes the proof of Lemma 7.6. □

Equipped with Lemma 7.6, we complete the proof of Theorem 7.1. Observe that the number of vectors $\vec{v} \in \mathbb{F}_2^N$ satisfying $\mathcal{HW}(\vec{v}) \in [2^{i-1}, 2^i]$ can be bounded by

$$\sum_{\ell=2^{i-1}}^{2^i} \binom{N}{\ell} \leq \sum_{\ell=2^{i-1}}^{2^i} \frac{N^\ell}{\ell!} \leq (2^i - 2^{i-1}) \cdot \frac{N^{2^i}}{(2^{i-1})!} \leq 2^{D \cdot 2^i}.$$

Hence, setting $\beta \leftarrow C/2$, if w is such that $D \leq \beta \cdot w$, we obtain by a straightforward union bound over all vectors $\vec{v}$ of weight in $[2^{i-1}, 2^i]$, denoting $S_{i,N}$ the set of vectors $\vec{v}$ over $\mathbb{F}_2^N$ with $\mathcal{HW}(\vec{v}) \in [2^{i-1}, 2^i]$,

$$\Pr_{M_1, \cdots, M_w \xleftarrow{\$} \mathcal{M}^i_{\mathsf{par}}}\left[\exists \vec{v} \in S_{i,N}, N_{\vec{v}}(M_1, \cdots, M_w) \geq \frac{w}{2}\right] \leq 2^{D \cdot 2^i} \cdot 2^{-C \cdot w \cdot 2^i} \leq 2^{-\alpha \cdot w} \tag{4}$$

for some constant $\alpha = \beta = C/2$. Note that our bounds are quite loose – in particular, the optimal choice of parameters depends on i, while we chose to settle for a worst-case choice of the same parameters for all i. Now, recall that the distribution $\mathcal{H}^i_{\mathsf{par}}$ is exactly the distribution that samples $H_{i,1}, \cdots, H_{i,w}$ independently from $\mathcal{M}^i_{\mathsf{par}}$ and output $H_i \leftarrow H_{i,1} || \cdots || H_{i,w}$. We are now almost ready to conclude: recall that a sample from $\mathcal{O}_{\mathsf{par}}(H)$ with $H = H_1 || \cdots || H_D$ is of the form $H \cdot \vec{e} = \bigoplus_{i=1}^D H_i \cdot \vec{e}_i = \bigoplus_{i=1}^D \bigoplus_{j=1}^w H_{i,j} \cdot \vec{e}_{i,j}$ where each $\vec{e}_i$ is a random sample from $\mathcal{R}_{w,i}$ (transposed), hence each $\vec{e}_{i,j}$ is a random sample from $\mathcal{S}_{1,2^i}$. Now, for any vector $\vec{v} \in \mathbb{F}_2^N$,

$$\vec{v}^\intercal \cdot (H \cdot \vec{e}) = \bigoplus_{i=1}^{D} \bigoplus_{j=1}^{w} (\vec{v}^\intercal \cdot H_{i,j}) \cdot \vec{e}_{i,j}.$$

Returning $\vec{v}^\intercal \cdot H_{i,j} \cdot \vec{e}_{i,j}$ for a vector $\vec{e}_{i,j}$ sampled from $\mathcal{S}_{1,2^i}$ is equivalent to sampling a uniformly random entry of the vector $v^\intercal \cdot H_{i,j}$. Let $\mathcal{D}_{i,j}$ be the distribution which returns $H_{i,j} \cdot \vec{e}_{i,j}$ for a random $\vec{e}_{i,j} \xleftarrow{\$} \mathcal{S}_{1,2^i}$. Observe that the condition $N_{\vec{v}}(H_{i,1}, \cdots, H_{i,w}) < w/2$ can be restated as follows: for at least half of the values $j \in [1, w]$, it holds that

$$\mathsf{bias}_{\vec{v}}(\mathcal{D}_{i,j}) \leq \frac{1}{2} - \frac{1}{5} = \frac{3}{10}.$$

Now, a sample of the form $H_i \cdot \vec{e}_i$ for a random $\vec{e}_i$ from $\mathcal{R}_{w,i}$ is the XOR of independent random samples from each of the $\mathcal{D}_{i,j}$. Therefore, by Lemma 3.2, whenever $N_{\vec{v}}(H_{i,1}, \cdots, H_{i,w}) < w/2$, the bias of $\mathcal{D}_i$ with respect to $\vec{v}$ decrease exponentially with w:

$$\mathsf{bias}_{\vec{v}}(\mathcal{D}_i) \leq \frac{1}{2} \cdot \left(\frac{3}{5}\right)^{w/2}.$$

Plugging this into Equation 4 gives

$$\Pr_{H_i \xleftarrow{\$} \mathcal{H}^i_{\mathsf{par}}} \left[\exists \vec{v} \in S_{i,N}, \mathsf{bias}_{\vec{v}}(\mathcal{D}_i) > \frac{1}{2} \cdot \left(\frac{3}{5}\right)^{w/2} \right] \leq 2^{-\alpha \cdot w}. \tag{5}$$

Therefore, by a union bound again,

$$\Pr_{H \xleftarrow{\$} \mathcal{H}_{\mathsf{par}}} \left[\exists i \in [0, D-1], \exists \vec{v} \in S_{i,N},\ \mathsf{bias}_{\vec{v}}(\mathcal{D}_i) > \frac{1}{2} \cdot \left(\frac{3}{5}\right)^{w/2} \right] \leq D \cdot 2^{-\alpha w}.$$

By the rightmost inequality of Lemma 3.2, for any vector $\vec{v}$, the bias of $\mathcal{O}_{\mathsf{par}}$ with respect to $\vec{v}$ is at most the smallest bias of any of the $\mathcal{D}_i$ with respect to $\vec{v}$ (since a sample from $\mathcal{O}_{\mathsf{par}}$ is the exclusive OR of independent samples from each of the $\mathcal{D}_i$). Now, since the $S_{i,N}$ for $i = 0$ to $D-1$ cover all of $\mathbb{F}_2^N$ (as $N = 2^D$), we get:

$$\Pr_{H \xleftarrow{\$} \mathcal{H}_{\mathsf{par}}} \left[\exists \vec{v} \in \mathbb{F}_2^N,\ \mathsf{bias}_{\vec{v}}(\mathcal{O}_{\mathsf{par}}(H)) > \frac{1}{2} \cdot \left(\frac{3}{5}\right)^{w/2} \right] \leq D \cdot 2^{-\alpha w},$$

which concludes the proof.

Generalization to the Arithmetic Setting. The notion of low-bias weak PRF can be generalized from $\mathbb{F}_2$ to arbitrary groups, in the same way that the notion of low-bias pseudorandom generator extends to arbitrary groups: an ε-biased space over a group $\mathbb{G}$ is a distribution $\mathcal{D}$ such that for any nontrivial character $\chi : \mathbb{G} \mapsto \mathbb{C}$, it holds that $|\mathbb{E}[\chi(\mathcal{D})]| \leq \varepsilon$, see e.g. [LRTV09]. For the special case of $\mathbb{F}^N$ for a general field $\mathbb{F}$, the characters coincide (are in one-to-one correspondance with) linear functions $L : \mathbb{F}^N \mapsto \mathbb{F}$. Therefore, the natural generalization of the notion of ε-bias of a distribution $\mathcal{D}_N$ over $\mathbb{F}^N$ states that for every nonzero vector $\vec{v}$, the distribution induced by sampling $\vec{y} \xleftarrow{\$} \mathcal{D}_N$ and outputting $\langle \vec{v}, \vec{y} \rangle$ is ε-close to the uniform distribution over $\mathbb{F}$.

Most of the proof of resistance against linear test is oblivious to the choice of the underlying field and extends naturally to the arithmetic generalization of our candidate, given in Section 6.3. We sketch here how to adapt the proof to the general case: as before, we will reduce the problem to a balls and bins problem, but where each ball now corresponds to a uniformly random nonzero field element. Notably, remember that we count the number of bins containing exactly an odd number of balls; in our proof, each bin corresponds to a column $\vec{m}_j$ of a matrix M, and a bin contains an odd number of balls when $\vec{v}$ and $\vec{m}_j$ have an odd number of nonzero coordinates at the same position – hence in particular, at least one nonzero coordinate at the same position.

Conditioned on this being the case, since the nonzero entries of $\vec{m}_j$ are uniformly random nonzero elements of $\mathbb{F}$, the scalar product $\langle \vec{v}, \vec{m}_j \rangle$ is a uniformly random nonzero element of $\mathbb{F}$. Then, the rest of the proof proceeds identically. Note that over a large field $\mathbb{F}$, the bound on the number of vectors $\vec{v}$ grows as $2^{(\log|\mathbb{F}|+D)\cdot 2^i}$, hence the condition to be satisfied is that $D + \log|\mathbb{F}| \leq \beta \cdot w$ for some appropriate constant β. Eventually, concluding the proof requires a standard argument to show that a random size-$O(w)$ subset-sum of public random field elements is close to the uniform distribution over $\mathbb{F}$, when w is large enough.

7.3 Resistance Against Algebraic Attacks

While linear attacks capture many attacks from the literature, there is an important and widely studied class of attacks which does not fit into the linear test framework: algebraic attacks. Algebraic attacks have been introduced in [Pat95] and were later extended and abstracted in [Cou01, CM03, Cou03]. More recently, [AL18] formalized the notion of algebraic attacks in the study of random local functions.

Algebraic attacks start with input/ output pairs $(x^{(1)}, F_k(x^{(1)})), \ldots, (x^{(N)}, F_k(x^{(N)}))$ and use these to initialize system of — ideally linear — multivariate equations in the input variables, which can then potentially be solved via Gaussian elimination or by using Gröbner basis [CKPS00, Fau99, Fau02, Cou04].

In its most basic form, an algebraic attack proceeds as follows: given a function $F_K : \{0,1\}^n \mapsto \{0,1\}$ to be inverted (or distinguished from random), it seeks to find low degree multivariate polynomials (g, h) such that

$$F_K \cdot g = h.$$

If polynomials (g, h) of degree at most d are found, then the function F_K can be inverted given $n^{\tilde{O}(d)}$ random samples $(x, F_K(x))$, by solving a linear system in the coefficients of (g, h) given by the equations $F_K(x^{(i)}) \cdot g(x^{(i)}) = h(x^{(i)})$. Note that while the attack proceeds by solving a linear system of equations, the coefficients of this system depend not only on the $x^{(i)}$, but also on the samples $F_K(x^{(i)})$, which prevents the attack from being a linear attack according to our definition (where the coefficients are only allowed to depend on the $x^{(i)}$).

We note that while algebraic attacks are an important and well-studied class of attacks, they had been missed in the security analysis of the candidate weak PRF of Akavia *et al* [ABG+14].[9] It turned out that their candidate can be broken (in quasipolynomial time) using exactly the basic algebraic attack of [CM03] described above (this was observed in [BR17]). The authors of [BIP+18] conjectured their weak PRF candidate to be secure against this algebraic attack, but could not prove it. Below, we formally prove that our candidate weak PRF cannot be broken by the above attack:

Theorem 7.10 *Fix parameters $(w, D, N = 2^D)$. Let $n = \kappa \leftarrow w \cdot D(D-1)/2$, and let F_K be the candidate weak PRF family of Section 6.2. For any $K \in \{0,1\}^\kappa$, the algebraic attack of [CM03] requires a time and number of samples lower bounded by $n^{O(D)} = 2^{O(D \log(D+w))}$.*

Below, we prove Theorem 7.10 by showing that our candidate weak PRF has high *algebraic degree*.

Algebraic Immunity. The algebraic immunity of a boolean function F is defined as the following quantity:

$$\mathsf{AI}(F) = \min_{g \neq 0} \{\deg(g) \mid Fg = 0 \ \vee \ (F \oplus 1)g = 0\}.$$

[9] The analysis of [ABG+14] is also more restricted than ours in that it considers only resistance against a very specific form of linear attacks (correlation with a low degree polynomial), while we prove security against all possible linear attacks.

It is relatively easy to see that the smallest d such that there exist polynomials (g, h) of degree at most d satisfying $F_K \cdot g = h$ necessarily satisfies $d \geq \mathsf{AI}(F_K)$. Therefore, the algebraic immunity of F_K gives a lower bound on the efficiency of the above algebraic attack: attacking a weak PRF family $\{F_K\}_K$ with the above attack requires at least $\min_K \left\{n^{O(\mathsf{AI}(F_k))}\right\}$ samples. We now prove that our candidate weak PRF family has high algebraic immunity, hence cannot be broken by the above algebraic attack (implying Theorem 7.10).

Lemma 7.11 (Algebraic Immunity of rVDLPN) *Fix parameters $(w, D, N = 2^D)$. Let $n = \kappa = w \cdot D(D-1)/2$, and let F_K be the candidate weak PRF family of Section 6.2. For any $K \in \{0,1\}^\kappa$, it holds that*

$$\mathsf{AI}(F_K) \geq D.$$

Proof. The proof is largely taken from [MJSC16]. First, recall that our weak PRF candidate is given by

$$F_K(x) = F(K \oplus x)$$

where the function F is defined as

$$F(x) = \bigoplus_{i=1}^{D} \bigoplus_{j=1}^{w} \bigwedge_{k=1}^{i} x_{i,j,k}.$$

Now, it is easy to see that for any K, the algebraic immunity of F_K is that of F: for any g such that $F_K \cdot g = 0$ (resp. $(1 \oplus F_K) \cdot g = 0$), the function $g_K : x \to g(K \oplus x)$ trivially satisfies $F \cdot g_K = 0$ (resp. $(1 \oplus F) \cdot g_K = 0$), since $x \to K \oplus x$ defines a permutation over the input domain (which is its own inverse). The same holds trivially in the reverse direction as well, hence

$$\forall K \in \{0,1\}^\kappa \colon \mathsf{AI}(F_K) = \mathsf{AI}(F).$$

It remains to bound the algebraic immunity of F. Observe that F is a direct sum of w independent *triangular functions of degree D*, each evaluated on distinct portions of the input, where the triangular function of degree D is defined as

$$T_D(x_1, \cdots, x_{D(D-1)/2}) = x_1 \oplus x_2 x_3 \oplus x_4 x_5 x_6 \oplus \cdots \oplus \bigwedge_{\ell=D(D-1)/2-D}^{D(D-1)/2} x_\ell.$$

By Lemma 3 of [MJSC16], the algebraic immunity of a direct sum of functions (evaluated on independent inputs) is at least the largest algebraic immunity of each of its components, hence

$$\mathsf{AI}(F) \geq \mathsf{AI}(T_D).$$

Now, by Lemma 6 of [MJSC16], the algebraic immunity of the degree-D triangular function is exactly D (the proof is a simple proof by induction, see Appendix C.2 of [MJSC16]). Therefore, we get

$$\forall K \in \{0,1\}^\kappa \colon \mathsf{AI}(F_K) \geq D,$$

which concludes the proof. □

Fast Algebraic Immunity. An improved variant of the above algebraic attack, called *fast algebraic attack*, was introduced by Courtois in [Cou03]. It was shown in [ACG+06] that it suffices, for a candidate F_K to withstand this attack, that F_K has high *fast algebraic immunity* (denote $\mathsf{FAI}(F_K)$), which is defined as

$$\mathsf{FAI}(F) = \min\left\{2\mathsf{AI}(F), \min_{1 \leq \deg g \leq \mathsf{AI}(F)} \{\max\{\deg(g) + \deg(Fg), 3\deg g\}\}\right\}.$$

Then, it is shown in [ACG+06] that the fast algebraic attacks takes time at least $n^{O(\mathsf{FAI}(F))}$ to attack a function F with input size n. By [Car20], the fast algebraic immunity of a function F is lower bounded by $\mathsf{AI} + 1$. This yields the following Lemma.

Lemma 7.12 (Fast Algebraic Immunity of rVDLPN) *Fix parameters $(w, D, 2^D)$. Let $n = \kappa = w \cdot D(D-1)/2$, and let F_K be the candidate weak PRF family of Section 6.2. For any $K \in \{0,1\}^\kappa$, it holds that*

$$\mathsf{FAI}(F_K) \geq D + 1.$$

7.4 Resistance Against Statistical Query Algorithms

Another important approach to ruling out candidate PRFs in low complexity classes is to provide a learning algorithm for the class. This is for example the basis of the approach of Linial, Mansour, and Nisan in [LMN89] which showed that all functions in the class AC^0 can be learned in quasipolynomial time given access to uniformly random samples (hence, there are no weak PRFs with better than quasipolynomial security in AC^0), by showing than any AC^0 function is noticeably correlated with a linear function of polylogarithmically many variables. This attack can be generalized to a class of attacks, called "LMN-style attacks" in [ABG+14], which distinguishes a weak PRF candidate from random by detecting some large enough correlation between F_K and a function Φ belonging to a family $\mathbf{\Phi}$ of functions, where $\mathbf{\Phi}$ has relatively small size. To resist such LMN-style attacks, any candidate PRF must therefore at least not have high correlation with members of any fixed function family of small size.

An even more general class of attacks are those captured by the statistical query model of [Kea98], where random input-output pairs $(x, F_K(x))$ are used to estimate some statistics $\mathbb{E}\,\Phi = \mathbb{E}_x[\Phi(x, F_K(x))]$ for various functions Φ. A function family $\{F_K\}$ is (ε, δ)-pseudorandom against statistical query algorithms from a class of functions $\mathbf{\Phi}$ if

$$\Pr_K\left[\left|\mathop{\mathbb{E}}_x[\Phi(x, F_K(x))] - \mathop{\mathbb{E}}_{x,R}[\Phi(x, R(x))]\right| \leq \varepsilon\right] \geq 1 - \delta.$$

Many known algorithms for learning from random samples operate in this manner. In spite of this generality, the work of [BR17, Section 7.7] proves that to achieve (ε, δ)-pseudorandomness against statistical query algorithms from a class of functions $\mathbf{\Phi}$, it suffices to have low correlation with a fixed family of functions, of size $2|\mathbf{\Phi}|$. Hence, proving resistance against the seemingly less-general LMN-style attacks actually suffices to prove resistance against all statistical query algorithms. More precisely, given the class $\mathbf{\Phi}$, let $\mathbf{\Phi}'$ denote the class of functions $\Phi(\cdot, b)$, for $\Phi \in \mathbf{\Phi}$ and $b \in \{0,1\}$ (of size $2|\mathbf{\Phi}|$).

Lemma 7.13 (Implicit in [BR17]) *For any function family $\{F_K\}_K$ such that*

$$\Pr_{K \xleftarrow{\$} \{0,1\}^\kappa}\left[\exists \Phi \in \mathbf{\Phi}', \Pr_x[F_K(x) = \Phi(x)] > \frac{1}{2} + \varepsilon\right] \leq \delta,$$

then the function family $\{F_K\}$ is (ε, δ)-pseudorandom against statistical query algorithms from the class of functions $\mathbf{\Phi}$.

Below, we prove that our candidate resists all statistical query algorithms. More precisely, we show that for any fixed function family $\mathbf{\Phi}$ of size at most s, with probability at least $1 - s/(2^{w+2}\varepsilon^2)$ over the choice of the PRF key K, the correlation between F_K and any function from $\mathbf{\Phi}$ is at most ε:

Lemma 7.14 *Fix parameters $(w, D, N = 2^D)$. Let $n = \kappa = w \cdot D(D-1)/2$, and let F_K be the candidate weak PRF family of Section 6.2. Let $\mathbf{\Phi} = \{\Phi : \{0,1\}^n \mapsto \{0,1\}\}$ be a collection of functions of size s. Then for any ε,*

$$\Pr_{K \xleftarrow{\$} \{0,1\}^\kappa}\left[\exists \Phi \in \mathbf{\Phi}, \Pr_x[F_K(x) = \Phi(x)] > \frac{1}{2} + \varepsilon\right] \leq \frac{s}{2^{w+2}\varepsilon^2}.$$

By Lemma 7.13, this suffices to guarantee $(\varepsilon, \frac{s}{2^{w+2}\varepsilon^2})$-pseudorandomness against statistical query algorithms from any class of function of size s.

Proof. The proof follows closely the strategy of [ABG+14, BR17], but is slightly more involved because our function family is not pairwise independent. Fix a function $\Phi : \{0,1\}^n \mapsto \{0,1\}$. Define the random variable

$$Z(K) = \Pr_x[F_K(x) = \Phi(x)] = \mathop{\mathbb{E}}_x[\mathbb{1}(F_K(x), \Phi(x))],$$

where $\mathbb{1}(u, v) = 1$ if $u = v$, and 0 otherwise. Then,

$$\begin{aligned}
\mathop{\mathbb{E}}_K[Z(K)] &= \mathop{\mathbb{E}}_K[\mathop{\mathbb{E}}_x[\mathbb{1}(F_K(x), \Phi(x))]] \\
&= \mathop{\mathbb{E}}_x[\mathop{\mathbb{E}}_K[\mathbb{1}(F(K \oplus x), \Phi(x))]] \\
&= \mathop{\mathbb{E}}_x[\mathop{\mathbb{E}}_y[\mathbb{1}(F(y), \Phi(x))]] \\
&= \mathop{\mathbb{E}}_x[\Pr_y(F(y) = \Phi(x))] \\
&= 1/2,
\end{aligned}$$

where the last equality follows from the fact that for any $b \in \{0,1\}$, $\Pr_y[F(y) = b] = 1/2$. The latter is immediate, as F can be written as $F(x) = x_0 \oplus F'(x')$ where $x = x_0||x'$; in other words, $F(x)$ involves a XOR with with one of its variables that never appear in any other monomial. Now, we bound the variance σ^2 of $Z(K)$:

$$\begin{aligned}
\mathop{\mathbb{E}}_K[Z(K)^2] &= \mathop{\mathbb{E}}_K[\mathop{\mathbb{E}}_x[\mathbb{1}(F_K(x), \Phi(x))]^2] \\
&= \mathop{\mathbb{E}}_K[\mathop{\mathbb{E}}_x[\mathbb{1}(F(K \oplus x), \Phi(x))] \cdot \mathop{\mathbb{E}}_y[\mathbb{1}(F(K \oplus y), \Phi(y))]] \\
&= \mathop{\mathbb{E}}_{x,y}[\mathop{\mathbb{E}}_K[\mathbb{1}(F(K \oplus x), \Phi(x)) \cdot \mathbb{1}(F(K \oplus y), \Phi(y))]] \\
&= \sum_{x,y \in \{0,1\}^n} \Pr_K[(F(K \oplus x) = \Phi(x)) \ \wedge \ (F(K \oplus y) = \Phi(y))] \cdot 2^{-2n}.
\end{aligned}$$

From there, the standard approach to conclude [ABG+14, BR17, BIP+18] is to rely on the pairwise independence of F_K to bound $\mathbb{E}_K[\mathbb{1}(F_K(x), \Phi(x)) \cdot \mathbb{1}(F_K(y), \Phi(y))]$. This does not work immediately in our setting, since our candidate weak PRF family is easily seen to *not* be pairwise independent. Instead, we follow a different approach. Recall that our candidate weak PRF is of the form

$$F_K(x) = \bigoplus_{i=1}^{D} \bigoplus_{j=1}^{w} \bigwedge_{k=1}^{i} (x_{i,j,k} \oplus K_{i,j,k}).$$

The idea of the proof will be to look at the degree-2 component only. Whenever two inputs disagree in the part influencing the degree-2 component, the corresponding key component will show up in exactly one of those, allowing to argue independence. As this is the case for "most" inputs, this will be sufficient to prove the lemma. We isolate the degree-two component $\bigoplus_{j=1}^{w}(x_{2,j,1} \oplus K_{2,j,1}) \wedge (x_{2,j,2} \oplus K_{2,j,2})$ from the rest of the above function, and write $x_{2,k} = (x_{2,j,k})_{j \in [w]} \in \{0,1\}^w$ and $K_{2,k} = (K_{2,j,k})_{j \in [w]} \in \{0,1\}^w$ for $k \in \{1,2\}$, and denote by x_R, K_R the remaining bits of x and K (i.e. all $x_{i,j,k}$, $K_{i,j,k}$ for $i \neq 2$), respectively. Hence, F_K can be rewritten as

$$F_K(x) = \langle x_{2,1} \oplus K_{2,1}, x_{2,2} \oplus K_{2,2} \rangle \oplus F'(K_R, x_R),$$

where F' is the function that computes the missing monomials. Now, fix an arbitrary pair of inputs (x, y), decomposed as $((x_{2,1}, x_{2,2}, x_R), (y_{2,1}, y_{2,2}, y_R))$, such that $x_{2,2} \neq y_{2,2}$, and fix

arbitrary $K_{2,2}, K_R$. Observe that

$$\begin{aligned}
&\Pr_{K_{2,1}}[(F(K\oplus x)=\Phi(x)) \ \wedge \ (F(K\oplus y)=\Phi(y))]\\
=&\Pr_{K_{2,1}}[\big(\langle x_{2,1}\oplus K_{2,1}, x_{2,2}\oplus K_{2,2}\rangle = \Phi(x)\oplus F'(K_R,x_R)\big)\\
&\wedge \big(\langle y_{2,1}\oplus K_{2,1}, y_{2,2}\oplus K_{2,2}\rangle = \Phi(y)\oplus F'(K_R,y_R)\big)].
\end{aligned}$$

Now, since $x_{2,2}\neq y_{2,2}$, there is at least one position j such that the jth bits of $x_{2,2}\oplus K_{2,2}$ and of $y_{2,2}\oplus K_{2,2}$ differ; let us fix such a j and assume without loss of generality that the jth bit of $x_{2,2}\oplus K_{2,2}$ is 1. For any bistring s, let us denote $s_{\neq j}$ the string s without its jth bit. Then, we have

$$\langle x_{2,1}\oplus K_{2,1}, x_{2,2}\oplus K_{2,2}\rangle = (K_{2,1})_j \oplus (x_{2,1})_j \oplus \langle (x_{2,1}\oplus K_{2,1})_{\neq j}, (x_{2,2}\oplus K_{2,2})_{\neq j}\rangle$$

while

$$\langle y_{2,1}\oplus K_{2,1}, y_{2,2}\oplus K_{2,2}\rangle = \langle (y_{2,1}\oplus K_{2,1})_{\neq j}, (y_{2,2}\oplus K_{2,2})_{\neq j}\rangle.$$

Therefore, the events $F(K\oplus x)=\Phi(x)$ and $F(K\oplus y)=\Phi(y)$ are truly independent:

$$\begin{aligned}
&\Pr_{K_{2,1}}[(F(K\oplus x)=\Phi(x)) \ \wedge \ (F(K\oplus y)=\Phi(y))]\\
=&\Pr_{(K_{2,1})_j}[\langle x_{2,1}\oplus K_{2,1}, x_{2,2}\oplus K_{2,2}\rangle = \Phi(x)\oplus F'(K_R,x_R)]\\
&\cdot \Pr_{(K_{2,1})_{\neq j}}[\langle y_{2,1}\oplus K_{2,1}, y_{2,2}\oplus K_{2,2}\rangle = \Phi(y)\oplus F'(K_R,y_R)]\\
=&\frac{1}{2}\cdot\frac{1}{2}=\frac{1}{4}.
\end{aligned}$$

Furthermore, when $x_{2,2}=y_{2,2}$, it still holds that

$$\Pr_K[(F(K\oplus x)=\Phi(x)) \ \wedge \ (F(K\oplus y)=\Phi(y))] \leq \Pr_K[(F(K\oplus x)=\Phi(x))] = \frac{1}{2}.$$

Let us partition $\{0,1\}^n\times\{0,1\}^n$ in two disjoint sets, S and $\bar{S}=\{0,1\}^n\times\{0,1\}^n\setminus S$, where S contains all pairs (x,y) which satisfy, when decomposed as $((x_{2,1},x_{2,2},x_R),(y_{2,1},y_{2,2},y_R))$, $x_{2,2}\neq y_{2,2}$. Then, the above imply

$$\begin{aligned}
\mathop{\mathbb{E}}_K[Z(K)_{2,2}] =& \sum_{x,y\in S}\Pr_K[(F(K\oplus x)=\Phi(x)) \ \wedge \ (F(K\oplus y)=\Phi(y))]\cdot 2^{-2n}\\
&+\sum_{x,y\in \bar{S}}\Pr_K[(F(K\oplus x)=\Phi(x)) \ \wedge \ (F(K\oplus y)=\Phi(y))]\cdot 2^{-2n}\\
&\leq |S|\cdot\frac{1}{4}\cdot 2^{-2n}+|\bar{S}|\cdot\frac{1}{2}\cdot 2^{-2n}.
\end{aligned}$$

By construction, $|\bar{S}|=2^{2n-w}$ since $|x_{2,2}|=|y_{2,2}|=w$, and $|S|=2^{2n}-2^{2n-w}$. Therefore,

$$\begin{aligned}
\mathop{\mathbb{E}}_K[Z(K)_{2,2}] &\leq (2^{2n}-2^{2n-w})\cdot\frac{1}{4}\cdot 2^{-2n}+2^{2n-w}\cdot\frac{1}{2}\cdot 2^{-2n}\\
&=(1-2^{-w})\cdot\frac{1}{4}+2^{-w}\cdot\frac{1}{2}\\
&=\frac{1}{4}\cdot(1+2^{-w}).
\end{aligned}$$

Now, by the definition of variance,

$$\begin{aligned}
\sigma_{2,2} &= \mathop{\mathbb{E}}_K[Z(K)_{2,2}] - \mathop{\mathbb{E}}_K[Z(K)]_{2,2}\\
&\leq \frac{1}{4}\cdot(1+2^{-w})-\frac{1}{4}\\
&=\frac{1}{2^{w+2}}.
\end{aligned}$$

To finish the proof, it remains to apply the Bienaymé-Chebyshev inequality (Lemma 3.3), which immediately gives

$$\Pr_{K \xleftarrow{\$} \{0,1\}^\kappa}\left[\exists \Phi \in \mathbf{\Phi}, \Pr_x[F_K(x) = \Phi(x)] > \frac{1}{2} + \varepsilon\right] \leq \frac{1}{2^{w+2}\varepsilon^2},$$

from which the result directly follows by a union bound.

□

7.5 Resistance Against AC^0 Tests

Fix parameters $(w, D, N = 2^D)$. Let $n = \kappa = w \cdot D(D-1)/2$, and let F_K be the candidate weak PRF family of Section 6.2. For any input $x \in \{0,1\}^n$, let R_x denote the random variable distributed as $F_K(x)$ for a random K from $\{0,1\}^\kappa$.

Theorem 7.15 ((w/D)-Wise Independent)

$$\Pr_{x^{(i)} \xleftarrow{\$} \{0,1\}^n}[(R_{x^{(1)}}, \cdots, R_{x^{(n)}}) \text{ are } (w/D)\text{-wise independent }] \geq 1 - \frac{1}{(w/D)!}.$$

Proof. As in the proof of Lemma 7.14, we rely on the fact that for any input x, $F_K(x)$ can be rewritten as

$$F_K(x) = \langle x_{2,1} \oplus K_{2,1}, x_{2,2} \oplus K_{2,2}\rangle \oplus F'(K_R, x_R),$$

where $(x_{2,1}, x_{2,2}, x_R)$ and $(K_{2,1}, K_{2,2}, K_R)$ form appropriate partitions of x and K. Hence, to prove that the $F_K(x^{(i)})$ for $i = 1$ to N are (w/D)-wise independent, it suffices to fix an arbitrary K^R and prove that the random variables $R'_{x^{(i)}}$ induced by $\langle x^{(i)}_{2,1} \oplus K_{2,1}, x^{(i)}_{2,2} \oplus K_{2,2}\rangle$ for random $(K_{2,1}, K_{2,2})$ are (w/D)-wise independent (with high probability over the choice of $x^{(1)}, \cdots, x^{(N)}$). Furthermore, if the random variable $R''_{x^{(i)}}$ induced by $\langle K_{2,1}, x^{(i)}_{2,2}\rangle$ for a random $K_{2,1}$ are (w/D)-wise independent (with high probability over the choice of the $x^{(i)}$), then the $R'_{x^{(i)}}$ are (w/D)-wise independent.

Now, it is a well-known results (see for example [NN90]) that N random variables $(\langle K_{2,1}, z^{(i)}\rangle)_{i \leq N}$ will be (w/D)-wise independent if any size-(w/D) subset of the $z^{(i)}$, seen as length-n vectors over $\mathbb{F}_2$, are linearly independent over $\mathbb{F}_2$. Hence, to prove that the $R''_{x^{(i)}}$ are (w/D)-wise independent with high probability over the choice of the $x^{(i)}$, it suffices to prove that any size-(w/D) subset of the vectors $(x^{(i)}_{2,2})_{i \leq N}$ is linearly independent over $\mathbb{F}_2$.

We now bound the probability, over the random choice of $x^{(1)}_{2,2}, \cdots, x^{(N)}_{2,2} \xleftarrow{\$} \mathbb{F}_2^w$, that all (w/D)-tuples of vectors $x^{(i)}_{2,2}$ are linearly independent over $\mathbb{F}_2$. First, let us compute the probability that (w/D) random vectors sampled from $\mathbb{F}_2^w$ are linearly independent: the probability that the first vector is linearly dependent is $1/2^w$ (this is the probability that it is the all-zero vector); the probability that the second vector linearly depends on the first one is $1/2^{w-1}$, and by induction, the probability that the j-th vector linearly depends on the $(j-1)$-th first vectors is $1 - 1/2^{w-j+1}$. Hence, the overall probability that a random (w/D)-tuple of vector is linearly independent is

$$\prod_{j=0}^{w/D-1}\left(1 - \frac{1}{2^{w-j}}\right) \geq \prod_{j=0}^{w/D-1}\left(1 - \frac{1}{2^{w-w/D+1}}\right) = \left(1 - \frac{1}{2^{w-w/D+1}}\right)^{w/D}$$
$$\geq \left(1 - \frac{1}{2^{w(1-1/D)}}\right).$$

Hence, by a straightforward union bound, the probability that there exist any r-tuple of vectors in $(x_{2,2}^{(1)}, \cdots, x_{2,2}^{(N)})$ which are linearly dependent is upper bounded by

$$\binom{N}{w/D} \cdot \frac{1}{2^{w(1-1/D)}} \leq \frac{2^{(w/D)\cdot D - w}}{(w/D)!} = \frac{1}{(w/D)!},$$

which concludes the proof. □

Plugging the result of Braverman [Bra08] into Theorem 7.15, we get:

Corollary 7.16 *Our candidate weak PRF ε-fools depth-d AC^0 circuits of size m, with probability at least $1/(w/D)!$ over the random choice of $N = 2^D$ inputs, for any ε such that*

$$w/D = \left(\log \frac{m}{\varepsilon}\right)^{O(d^2)}.$$

This implies, for example, that our candidate $2^{-w/O(d^2)}$-fools AC^0 circuits of size up to $2^{w/O(d^2)}$ by setting $D \leftarrow w^t$ for any constant $t < 1$. We note that bounded independence also implies security against other classes of circuits beyond AC^0, such as halfspaces [DGJ+09] and degree-2 threshold functions [DKN10].

7.6 Linear Cryptanalysis

Among the standard attacks against PRF, we did not yet investigate linear cryptanalysis [Mat94] and differential cryptanalysis. Both types of attacks have been formalized in [MV11]. Linear cryptanalysis exploits the correlation of F_K with respect to some parity $\vec{v}$ of its input x (seen as a vector over $\mathbb{F}_2^n$). Following the formalism of [MV11], we define the correlation of F_K with respect to a vector $\vec{v}$ as

$$\mathsf{Corr}_{\vec{v}}(F_K) = 2 \cdot \Pr_x[\vec{v}^\intercal \cdot x = F_K(x)] - 1.$$

Then, the attack succeeds given a number of samples proportional to

$$\left(\max_{\vec{v}\neq\vec{0}} \left\{\mathop{\mathbb{E}}_K[\mathsf{Corr}_{\vec{v}}(F_K)^2]\right\}\right)^{-1}.$$

We now bound this quantity. First, observe that

$$\begin{aligned}\mathop{\mathbb{E}}_K[\mathsf{Corr}_{\vec{v}}(F_K)^2] &= \mathop{\mathbb{E}}_K[(2 \cdot \mathop{\mathbb{E}}_x[\mathbb{1}(\vec{v}^\intercal \cdot x, F_K(x))] - 1)^2] \\ &= 4 \cdot \left(\mathop{\mathbb{E}}_K[\mathop{\mathbb{E}}_x[\mathbb{1}(\vec{v}^\intercal \cdot x, F_K(x))]^2] - \mathop{\mathbb{E}}_K[\mathop{\mathbb{E}}_x[\mathbb{1}(\vec{v}^\intercal \cdot x, F_K(x))]]\right) + 1\end{aligned}$$

However, we have already shown in Section 7.4 that for any function Φ (hence in particular for the function $\Phi : x \mapsto \vec{v}^\intercal \cdot x$), it holds that

$$\begin{aligned}\mathop{\mathbb{E}}_K[\mathop{\mathbb{E}}_x[\mathbb{1}(F_K(x), \Phi(x))]] &= \frac{1}{2}, \text{ and} \\ \mathop{\mathbb{E}}_K[\mathop{\mathbb{E}}_x[\mathbb{1}(F_K(x), \Phi(x))]^2] &\leq \frac{1}{4} \cdot (1 + 2^{-w}).\end{aligned}$$

Therefore,

$$\mathop{\mathbb{E}}_K[\mathsf{Corr}_{\vec{v}}(F_K)^2] \leq 4 \cdot \frac{1}{4} \cdot (1 + 2^{-w}) - 4 \cdot \frac{1}{2} + 1 = 2^{-w}.$$

Hence, we get that

$$\left(\max_{\vec{v}\neq\vec{0}}\left\{\mathop{\mathbb{E}}_{K}[\mathsf{Corr}_{\vec{v}}(F_K)^2]\right\}\right)^{-1} \geq 2^w,$$

showing that linear cryptanalysis requires exponentially many samples to succeed against our candidate. Differential cryptanalysis, on the other hand, does not apply to weak PRFs, since it requires "programming" inputs to exploit the existence of values Δ, Δ' such that $\Pr_{x,K}[F_K(x) \oplus F_K(\Delta \oplus x) = \Delta']$ is high. Hence, differential cryptanalysis attacks are only allowed against strong PRFs, where the adversary is allowed to choose the inputs to the PRF. Actually, it is not hard to see our candidate is completely broken by differential cryptanalysis: any Δ prefixed with sufficiently many zeroes satisfy $\Pr_{x,K}[F_K(x) \oplus F_K(\Delta \oplus x) = 0]$ with very high probability.

8 Other Candidate FSS-Friendly WPRFs

In this section, we describe variants of our main candidate weak PRF. These variants are achieved by progressively simplifying our main candidate, and analyzing whether the resulting candidate still resists known attacks. In the discussions of this section, we let w denote a security parameter, and fix parameters $D(w) = O(w)$, and $N(w) = 2^D$. Observe that with these parameters, our main candidate has input size $O(w^3)$, and key size $O(w^3)$. Below, we provide two variants of our main construction which achieve smaller input size and/or key size, and discuss their security.

8.1 First Variant: Reusing Portions of the Input

Our first variant reduces the input length from $w \cdot D(D-1)/2$ to $w \cdot D$. Recall that $F_K(x)$ is of the form $F(K \oplus x)$, where F is a direct sum of w triangular functions T_D, which are evaluated on disjoint subsets of $D(D-1)/2$ bits of the input. The triangular function T_D is defined as

$$T_D(x) = \bigoplus_{i=1}^{D} \bigwedge_{j=1}^{i} x_{j+\sum_{k=1}^{i} k}.$$

In our first variant, the fan-in-i AND terms in the above triangular function are evaluated on length-i prefixes of a single D-bit string x, instead of being evaluated on independent length-i strings $(x_1, \cdots, x_D)$. That is, we replace the triangular function by the following simpler function $T'_D : \{0,1\}^D \mapsto \{0,1\}$:

$$T'_D(x) = \bigoplus_{i=1}^{D} \bigwedge_{j=1}^{i} x_j.$$

Hence, the new candidate weak PRF is given by

$$F_K(x) = \bigoplus_{i=1}^{D} \bigoplus_{j=1}^{w} \bigwedge_{k=1}^{i} (x_{j,k} \oplus K_{i,j,k}).$$

Security. Our first variant has the exact same security guarantees as our main candidate against linear attacks; that is, there exists constant (β, μ, ν) such that for any $w, D(w) \leq \beta \cdot w$, the first variant is a $(\mu^w, \nu^w, 2^D)$-biased weak PRF family. Indeed, the exact same security analysis applies: the proof never relies on the independence between the blocks H_i, but only on the fact that the $\vec{e}_i$ are sampled independently (which is needed to apply Lemma 3.2).

In our main candidate, the use of triangular functions suffices to guarantee resistance against algebraic attack, since the degree-D triangular function has algebraic immunity D. However, it

is easy to see that this does not hold anymore for the simpler function T'_D: we have

$$T'_D(x) = \bigoplus_{i=1}^{D} \bigwedge_{j=1}^{i} x_{j+\sum_{k=1}^{i} k} = x_1 \cdot \left(\bigoplus_{i=1}^{D} \bigwedge_{j=2}^{i} x_{j+\sum_{k=1}^{i} k} \right),$$

hence the degree-1 function $f : x \mapsto 1 + x_1$ is an annihilator of T'_D (i.e., $f \cdot T'_D = 0$), meaning that T'_D has algebraic immunity 1. Yet, we conjecture that the direct sum of w such functions has algebraic immunity $\min(w, D)$. We note that we got confirmation, through personal communication with Pierrick Méaux, that this function indeed has the claimed algebraic immunity; however, this result is not yet published to our knowledge.

Immunizing against other attacks with simple tweaks. We note that simple tweaks to the above function actually suffice to get provable resistance against the other attacks we considered, without changing the asymptotic input length. For example, for algebraic attacks, consider the following tweak of the function:

$$F_{K,K_y}(x, y) = \bigoplus_{i=1}^{D} \bigoplus_{j=1}^{w} \bigwedge_{k=1}^{i} (x_{j,k} \oplus K_{i,j,k}) \oplus T_D(y \oplus K_y).$$

That is, we XOR the first variant with a term $T_D(y \oplus K_y)$. This variant still has input size $O(w^2)$. By Lemma 3.2 (the bias of a direct sum is at least the smallest bias) and Lemma 3 of [MJSC16] (the algebraic immunity of a direct sum is at least the largest algebraic immunity of each of its components), the above variant inherit both the resistance of the first variant to linear attacks, and the high algebraic immunity of triangular functions. This gives us a candidate weak PRF in the class $\mathsf{XOR} \circ \mathsf{AND}$, with input size $O(w^2)$, which provably resists linear attacks and algebraic attacks running in time $2^{O(w)}$.

The same holds for statistical query algorithms or AC^0 attacks: all these variants can easily be enhanced to resist these attacks as well, by XORing them with appropriate functions in $\mathsf{XOR} \circ \mathsf{AND}$ which provably resist these attacks (e.g. XORing with the linear input size function $f_K(x) = \langle x, K \rangle$ suffices to resists statistical query algorithms, since this function is pairwise independent). We conjecture that these tweaks are actually not needed (that is, that this variant can be proven to resist all attacks we considered) and only help simplifying the analysis.

Optimality. A crucial feature of this variant is that it actually achieves optimal (conjectured) security as a function of its input size: it has input size $n = O(w^2)$ (using $D = O(w)$ and resistance at least $2^{O(w)}$ against linear attacks. As we will show in the next section, there actually exists linear attacks which break this candidate (and the first one) in time $2^{\tilde{O}(\sqrt{n})}$, which is exactly $2^{\tilde{O}(w)}$ here. Hence, This candidate has the smallest input size possible to achieve $\tilde{O}(w)$ bits of security.

8.2 Second Variant – Reducing the Key Size

Eventually, we put forth the security of the following candidate as an entirely open question:

"smallest" rVDLPN:

$$F_K(x) = \bigoplus_{i=1}^{D} \bigoplus_{j=1}^{w} \bigwedge_{k=1}^{i} (x_{j,k} \oplus K_{jk}),$$

where $x \in \{0,1\}^{w \cdot D}$ and $K \in \{0,1\}^{w \cdot D}$.

This candidate retains the optimal input size of our previous candidate, but reduces the key size by a factor $(D + 1)/2$. This is particularly interesting for our PCF application, as the

decision tree construction of [BGI16b] gives the FSS for *all* prefix functions (and therefore all products for a fixed j) at only a factor 2 overhead compared to FSS for a single point function. Therefore, we can save a factor of $D/2$ in the FSS key size building on our third candidate assumption.

While this modification makes our proof of resistance against linear attacks break down (since the independence between samples from the various distributions we consider is not guaranteed anymore), we could not find any attack on this candidate, and put forth its conjectured security as an interesting challenge to investigate. We summarize the candidates discussed in this section in Table 2.

			Resistance against attacks			
$F_K(x)$	input size	FSS key size	Linear	Algebraic	Correlation	AC^0
$\bigoplus_{i=1}^{D} \bigoplus_{j=1}^{w} \bigwedge_{k=1}^{i} (x_{i,j,k} \oplus K_{i,j,k})$	$O(w \cdot D^2)$	$O(w^2 \cdot D^2)$	✓	✓	✓	✓
$\bigoplus_{i=1}^{D} \bigoplus_{j=1}^{w} \bigwedge_{k=1}^{i} (x_{j,k} \oplus K_{i,j,k})$	$O(w \cdot D)$	$O(w^2 \cdot D^2)$	✓	✓	✓	✓
$\bigoplus_{i=1}^{D} \bigoplus_{j=1}^{w} \bigwedge_{k=1}^{i} (x_{j,k} \oplus K_{j,k})$	$O(w \cdot D)$	$O(w^2 \cdot D)$	?	?	?	?

Table 2: List of candidate weak PRF families, with $N(w) = 2^{D(w)}$. One can choose $D(w) = \mathsf{polylog}(w)$, if one aims at (quasi)exponential security up to $N = w^{\mathsf{polylog} w}$ samples, or $D(w) = w$ for subexponential security up to $2^{O(w)}$ samples. For the second row, the provable resistance actually holds w.r.t. minor tweaks of the function which do not change its input and key size in the case of algebraic attacks, statistical query algorithms, and AC^0 attacks.

9 Concrete Attacks on our WPRF Candidates

In this section, we complement the asymptotic analysis of our candidate's resistance against families of attacks (see Section 7) by analyzing how our candidates behave against concrete attacks, and putting forth conjectures regarding the exact security of our candidates against families of attacks.

9.1 Concrete Linear Attacks

Most attacks from the LPN literature are linear attacks. To our knowledge, all state-of-the-art attacks (in terms of concrete efficiency) are linear attacks, either from the information set decoding family (for LPN with a linear number of samples) or BKW-style attacks (for LPN with a larger number of samples). In our context, the VDLPN problems with parameters $(w, D, N = 2^D)$ correspond to an LPN-style problem with dimension $(w-1) \cdot 2^D$, and $w \cdot 2^D$ samples. Hence, the dimension is extremely close to the number of samples, and BKW-style attacks seem less likely to be particulary powerful, even using the sample-reduction technique of Lyubashevsky [Lyu05].

A Concrete Conjecture on Security Against Linear Attacks. The proof of resistance against linear attacks from Section 7 is very loose. However, we conjecture that much tighter bounds actually hold. Here, we put forth a simple combinatorial conjecture, which is a strong strenghtening of the loose bound which we prove in Section 7, and which can potentially be experimentally validated through simulated balls-and-bins experiments.

We recall some notation: We fix the number of samples to $N = 2^D$. For $1 \leq i \leq D$, $\mathcal{M}^i_{\mathsf{par}}$ samples a random matrix in $\mathbb{F}_2^{N \times 2^i}$ with row-weight 1. For any vector $\vec{v}$ and fixed i, we say that

a matrix M is *very good against* $\vec{v}$ if $\mathcal{HW}(\vec{v}^{\intercal} \cdot M) \in [2^i/3, 2^{i+1}/3]$. Then, we conjecture that the following holds:

Conjecture 9.1 *Fix $1 \leq i \leq D$, $N = 2^D$, a large enough w, and any vector $\vec{v} \in \mathbb{F}_2^N$ with hamming weight between 2^{i-1} and 2^i. The probability, over the sampling of $M_1, \cdots, M_w$ from $\mathcal{M}^i_{\mathsf{par}}$, that less than $w/2$ of the M_j's are very good against $\vec{v}$ is at most $2^{-w \cdot 2^{i-1}}$.*

The above can be formulated as a balls and bins experiment: consider an experiment where one throws t balls into 2^i bins at random, and declares "success" if the fraction of bins with an odd number of balls at the end of the experiment is between $1/3$ and $2/3$. Then, the above conjecture states that when $t \in [2^{i-1}, 2^i]$, if one repeats this experiment w times, the probability that less than $w/2$ of them are successful is at most $2^{-w \cdot 2^{i-1}}$. We view experimentally validating or disproving this conjecture as an important direction toward understanding the concrete resistance of VDLPN against linear attacks.

Furthermore, if the above conjecture holds, then it implies in particular (by plugging this into the proof of resistance against linear attacks) that with probability at least $2^{-w/2+\log w-2}$ over the sampling of a random VDLPN matrix H with parameters $(w, w/4, 2^{w/4})$, the distribution of samples has bias at most $(1/2) \cdot (2/3)^{w/2}$. Hence, the computational cost of a linear attack against VDLPN parameters, if the conjecture is true, is expected to require about $(N/2) \cdot (2/3)^{w/2}$ boolean operations.

As a concrete example of a choice of parameters according to the above analysis, we suggest the following as a natural target for cryptanalysis: set $w = 120$ and $D = 30$. Is it possible to break VDLPN, with $N = 2^{30}$, using less than 2^{80} boolean operations?

A Restriction-Based Linear Attack. Any learning algorithms for the class of sparse polynomials over $\mathbb{F}_2$ with random samples break the security of the proposed WPRF candidates. For example, our first candidate can be viewed as a wD-sparse polynomial over the variables $z_{i,j,h} = x_{i,j,h} \oplus k_{i,j,h}$, and a learning algorithm with random samples clearly distinguishes between our candidate and a random function. Both Bshouty [Bsh20] and Hellerstein and Servidio [HS07] proposed such algorithms. The first algorithm [Bsh20] works for uniformly chosen samples, and directly learns the polynomial labeling the samples. Since it hasn't been published we provide a description below. The second algorithm [HS07] is a PAC-learning algorithm, i.e. works over any distribution of the samples, and learns the polynomial via intermediate representations of the function as a generalized decision tree and a generalized decision list.

Both algorithms learn sparse polynomials with $\tilde{O}(2^{\sqrt{n}})$ samples and running time and can be viewed in our context as linear attacks. These algorithms show that our second candidate achieves optimal security against linear attacks in an asymptotic sense.

For the restriction algorithm in [Bsh20] use the following notation. Let P be a multivariate polynomial over $\mathbb{F}_2$ with n variables $x = (x_1, \ldots, x_n)$ and k monomials, and let $\sigma = \log^2 kn$. View P as a sum $P(x) = Q(x) + R(x)$, such that $Q(x)$ includes all monomials of degree at most $\sqrt{n}$ and $R(x)$ all the other monomials.

The algorithm begins by choosing a random subset $i_1, \ldots, i_{\sigma\sqrt{n}} \in \{1, \ldots, n\}$, and then analyzing $O(2^{2\sigma\sqrt{n}})$ samples $(y, P(y))$, where $y = (y_1, \ldots, y_n)$ is an assignment to x. Let Y include all the samples that satisfy $y_{i_j} = 0$ for all $j = 1, \ldots, \sigma\sqrt{n}$. With high probability, there are $O(2^{\sigma\sqrt{n}})$ such samples.

Let M be some monomial in R. Partition the $\sigma\sqrt{n}$ indices into σ sequences $I_1, \ldots, I_\sigma$ of $\sqrt{n}$ indices each. Then, the probability that none of the variables in M has an index in I_j is at most $(1 - 1/\sqrt{n})^{\sqrt{n}} \leq e^{-1}$. Therefore, the probability that M does not have an index in $\{I_1, \ldots, I_j\}$ is at most $e^{-\sigma}$ implying that $M(y) \neq 0$ with probability at most $e^{-\sigma}$ for any $y \in Y$. Since R has at most k monomials it follows by union bound that $\Pr[R(y) \neq 0] \leq e^{\log kn}$ which is negligible.

Except with negligible probability, $P(y) = Q(y)$ for all $y \in Y$. Since Q is of degree $\sqrt{n}$ it can be completely learned via linearization given $O(\binom{n}{\sqrt{n}})$ samples. Since with high probability

Y has $O(2^{\sigma\sqrt{n}})$ samples and $2^{\sigma\sqrt{n}} \geq \binom{n}{\sqrt{n}}$, the linearization step can be carried out on samples in Y and $Q(x)$ learned.

We complete the description of the algorithm by noting that since P is sparse, Q is a good approximation for P. That is, $\Pr[M(y) \neq 0] \leq 2^{-\sqrt{n}}$ for a monomial M in R and a uniformly chosen sample y. Therefore, $\Pr[R(y) \neq 0] \leq \frac{k}{2^{\sqrt{n}}}$ for a random y and $P(x) = Q(x)$ except with $\frac{k}{2^{\sqrt{n}}}$ probability.

9.2 Concrete Security Against Other Attacks

To complement the above, we briefly discuss the concrete security of our candidate against other attacks. Regarding algebraic attacks, the natural attack proceeds by using the fact that our candidates are degree-D multivariate polynomials. Therefore, one needs at least n^D samples to linearize the system of equations and distinguish the samples from random (the expected number of samples needed is of the order of $n^D \log(n^D)$). In the first candidate, $n = w \cdot D(D-1)/2$; in the second candidate, $n = w \cdot D$. Therefore, the provable resistance against algebraic attacks (which is proven in the case of the first candidate, and can be proven for a simple modification of the second candidate, see the dicussion in Section 8.1) is actually tight, since it matches exactly what is achieved by the straightforward linearization attack.

Regarding statistical query algorithms, fix some ε. Observe that having $(1/2+\varepsilon)$-correlation with a function from a family of size s with probability $s/(2^{w+2}\varepsilon^2)$ implies an attack which requires about 2^{w+2} function evaluations, and succeeds with some constant advantage: simply set s to $2^{w+2}\varepsilon^2$ and go through all functions from the family. For each function Φ, estimate whether $\Pr_x[F_K(x) = \Phi(x)]$ by sampling about $(1/\varepsilon)^2$ independent x and counting the number of times that $F_K(x) = \Phi(x)$. Since the function family contains a function correlated with F_K, and this correlation is detected with constant probability using $1/\varepsilon^2$ samples, this attack succeeds with constant probability, and requires of the order of 2^w function evaluations.

In particular, in the case of our second candidate where $n = w \cdot D$ (and $D < w$), this implies that every statistical query algorithm also requires at least $2^{\sqrt{n}}$ function evaluations, which is the optimal security one can hope for anyway in light of the restriction attack described above.

9.3 Concrete Efficiency Estimations for PCF for VOLE

In light of the above discussions, instantiating the parameters with $w \approx 1.5 \cdot \lambda$, $D = w/4$, and $N = 2^D$, both our main candidate and the two variants can plausibly achieve λ bits of security. We note that these are very preliminary estimates, which could be off by a large margin and be either too optimistic or too conservative. Further, we note that our arguments and discussions only support the security claims regarding our main candidate and the first variant; for the second variant, we could not provide any proof of resistance against families of attacks. At the same time, however, we could not find any attack that performs significantly better against the second variant compared to the main candidate and the first variant.

Plugging our second variant (modified to work over rings as in Section 6.3) into the construction of PCF for VOLE of Section 6.4, and using the fact that distributed point functions for all prefixes of a point can be generated "all at once" at only a factor-2 cost compared to generating a single DPF (using the decision tree construction of [BGI16b]), leads to a PCF for VOLE over a ring R with $N = 2^D$ samples using key size

$$\approx 2w \cdot (D \cdot \kappa + \log_2 |R|)$$

bits, where κ denotes a seed size for a PRG. Using the more conservative variants of our PCF construction (either the main candidate or the first variant), the above size grows by a factor $D/2$. With $\lambda = 80$ and $\log_2 |R| = 64$, setting κ to 128 (as in an AES-based implementation), this gives the numbers of 120kB and 2MB mentioned in the introduction, for the second variant

Protocol $\Pi^{\mathcal{Y}}_{\mathsf{corr}*}$

Initialization. On input 1^λ proceed as follows:

- The parties call $\mathcal{F}^{\mathsf{PCF.Gen}}$, where $\mathcal{F}^{\mathsf{PCF.Gen}}$ is the ideal functionality which samples keys $(\mathsf{k}_0, \mathsf{k}_1) \xleftarrow{\$} \mathsf{PCF.Gen}(1^\lambda)$ and returns k_σ to P_σ.

Invocation. On input $x \in \{0,1\}^{n(\lambda)}$ proceed as follows:

- P_σ outputs $y_\sigma \leftarrow \mathsf{PCF.Eval}(\sigma, \mathsf{k}_\sigma, x)$.

Figure 12: Protocol for producing correlated randomness based on a strong PCF $(\mathsf{PCF.Gen}, \mathsf{PCF.Eval})$.

and main candidate, respectively. Based on the performance of AES on a single core of a modern CPU, we expect the computational efficiency of the second variant of the PCF can be as high as 10–20 thousand evaluations per second, and this easily scales with multiple cores. We can also obtain a PCF for OT with roughly the same costs, using the binary form of the WPRF and the optimized PCF from Remark 5.6.

10 Applications

In this section we motivate the PCF primitive and our constructions by discussing a number of cryptographic applications. At a high level, the power of PCF stems from enabling two parties — at the cost of a one-time distributed setup — to *locally* generate an effectively unbounded amount of correlated randomness. Correlated randomness is a valuable resource, as in many contexts it gives rise to information-theoretic security and attractive efficiency features. While the information-theoretic security feature is lost when applying a PCF, the efficiency features remain. We discuss several concrete use-cases below.

10.1 Secure Multiparty Computation with Fully Reusable Preprocessing

We first turn our attention to secure multiparty computation (MPC), where two or more parties want to securely evaluate a public function on their private inputs, such that the parties learn the function output and nothing beyond this output. Security is typically defined by considering an external adversary who may corrupt a strict subset of the parties. MPC with no honest majority, where the adversary can corrupt any number of parties, is impossible to realize in general with information-theoretic security. This applies in particular to the two-party case. However, MPC with no honest majority *can* be realized with information-theoretic security given suitable sources of secret correlated randomness, with good concrete efficiency. The most basic notion of correlated randomness in the setting of secure computation are OT correlations, which in fact are sufficient for the secure computation of general functionalities [GMW87, Kil88, IPS08].

In the following we extend the result of [BCG+19b] to PCFs, showing that PCFs can be used as a plug-in replacement to the pre-processing phase in protocols consuming correlated randomness that allow corrupted parties to influence their share of the correlated randomness. This includes most MPC protocols from the literature, and in concretely efficient protocols such as [BDOZ11, DPSZ12, NNOB12, WRK17a, WRK17b]. We first show the applicability of PCFs in the two-party setting, and then consider applications in multi-party setting in Section 10.5.

More formally, a PCF can be used used (as depicted in protocol $\Pi^{\mathcal{Y}}_{\mathsf{corr}*}$, Figure 12) to securely realize a corruptible functionality $\mathcal{F}^{\mathcal{Y}}_{\mathsf{corr}*}$ (see Figure 13) for the ideal generation of correlated randomness.

Functionality $\mathcal{F}^{\mathcal{Y}}_{\mathsf{corr}*}$

Initialization. On input 1^λ, the functionality does as follows:

- Set $\mathcal{Q} = \emptyset$.
- If the correlation requires setup: Sample $\mathsf{mk} \leftarrow \mathsf{Setup}(1^\lambda)$.

Invocation. On input $x \in \{0,1\}^{n(\lambda)}$ proceed as follows:

- Check if $(x, y_0, y_1) \in \mathcal{Q}$.
- Else, if no parties are corrupt, sample $(y_0, y_1) \leftarrow \mathcal{Y}(1^\lambda, \mathsf{mk})$.
- Else, if P_σ is corrupt, wait to receive $y_\sigma \in \{0,1\}^{\ell_\sigma}$ from $\mathcal{A}$. Then, sample $y_{1-\sigma} \leftarrow \mathsf{RSample}(1^\lambda, \mathsf{mk}, \sigma, y_\sigma)$.
- Set $\mathcal{Q} = \mathcal{Q} \cup \{(x, y_0, y_1)\}$ and output y_σ to P_σ.

Figure 13: Corruptible correlated randomness functionality for reverse-sampleable correlation sampled by $\mathcal{Y}$ (with setup algorihm Setup, if required).

Theorem 10.1 *Let* $\mathsf{PCF} = (\mathsf{PCF.Gen}, \mathsf{PCF.Eval})$ *be a strong PCF for the reverse-sampleable correlation sampled by* $\mathcal{Y}$ *(with setup). Then the protocol* $\Pi^{\mathcal{Y}}_{\mathsf{corr}*}$ *securely realizes the* $\mathcal{F}^{\mathcal{Y}}_{\mathsf{corr}*}$ *functionality for two parties against a static, malicious adversary (even if the adversary is given full adaptive control over the choice inputs* $x \in \{0,1\}^{n(\lambda)}$*).*

Proof. Let $\mathcal{A}$ be a static adversary against the protocol $\Pi^{\mathcal{Y}}_{\mathsf{corr}*}$. We construct a simulator Sim, which interacts with $\mathcal{A}$ and $\mathcal{F}^{\mathcal{Y}}_{\mathsf{corr}*}$ to produce a view for $\mathcal{A}$ that is indistinguishable from a real execution of the protocol. When both parties are corrupted, the simulator just runs $\mathcal{A}$ internally and security is straightforward. Similarly, when both parties are honest, simulation is trivial and indistinguishability follows as PCF satisfies pseudorandomness of outputs with adaptive queries (i.e. indistinguishability even holds if the adversary is given full adaptive control over x). Now suppose that only P_σ is corrupted, for $\sigma \in \{0,1\}$. On receiving the input 1^λ, Sim samples a pair of seeds $(\mathsf{k}_0, \mathsf{k}_1) \leftarrow \mathsf{PCF.Gen}(1^\lambda)$ and sends k_σ to $\mathcal{A}$ as its output of $\mathcal{F}^{\mathsf{PCF.Gen}}$. On input x_i, the simulator computes $y_{i,\sigma} \leftarrow \mathsf{PCF.Eval}(\sigma, \mathsf{k}_\sigma, x_i)$ and sends this to $\mathcal{F}^{\mathcal{F}}_{\mathsf{corr}*}$. Let N be the number of inputs to $\mathcal{F}^{\mathcal{F}}_{\mathsf{corr}*}$. Notice that in the ideal execution, the view of the distinguisher consists of the seed k_σ and the honest party's output $(y_{i,1-\sigma})_{i\in[N]}$, which is computed by $\mathcal{F}^{\mathcal{Y}}_{\mathsf{corr}*}$ as $y_{i,1-\sigma} \leftarrow \mathsf{RSample}(1^\lambda, \mathsf{mk}, \sigma, y_{i,\sigma})$. The only difference in the real execution, is that there the i-th honest party's output is computed as $y_{i,1-\sigma} \leftarrow \mathsf{PCF.Eval}(1-\sigma, \mathsf{k}_{1-\sigma}, x_i)$. These two views are computationally indistinguishable, due to the strong security of PCF (even if the adversary has full control over the choice of inputs $x \in \{0,1\}^{n(\lambda)}$ after seeing its key k_σ). □

Note that in order to establish PCFs as a source for correlated randomness, the parties need to securely distribute the PCF keys, which we will briefly elaborate on in the following. But — different to previous approaches for correlated randomness that were either inherently limited to a one-time use or only allowed limited reusability — once the parties share a PCF key pair, they can engage in *arbitrarily* many protocol invocations consuming correlated randomness, without the need to perform any further interactive preprocessing.

Realizing distributed key generation. Recall that the ideal key generation $\mathcal{F}^{\mathsf{PCF.Gen}}$ receives as input 1^λ, sets up a pair of keys $(\mathsf{k}_0, \mathsf{k}_1) \leftarrow \mathsf{PCF.Gen}(1^\lambda)$ and returns k_σ to P_σ.

When restricting the input domain to be polynomially bounded, as for previous constructions

of pseudorandom correlation generators, one could implement $\mathcal{F}^{\mathsf{PCF.Gen}}$ with the protocol for distributed setup of point functions by Doerner and shelat [Ds17] (for semi-honest security). For our PCF for OT with bounded input domain, one could further rely on the optimized protocol of Boyle et al. [BCG+19a] yielding a 2-round setup. They also give a setup protocol with malicious security, but at the cost of a slightly stronger assumption — translated to our setting, this requires to assume that VD-LPN holds even if the adversary is given small leakage on the key (which we expect to be the case when adapting the parameters accordingly).

As the focus of the work is *full reusability* and therefore an exponential input domain, we have to resort to an "off-the-shelf" protocol for setting up the keys, such as [KRRW18]. Note though that this only corresponds to a one-time cost, after which the parties can *locally* produce correlated randomness for all future computations.

Alternatively, if one is willing to settle for security against one corruption, as remarked in [BGI19], this cost can be avoided completely by letting a third party set up the correlated randomness. Note that this is similar to the 3-party framework from [MR18], but in our setting the third party does not have to take part in the actual computation.

10.2 Black-box Two-Round MPC with Fully Reusable Preprocessing

Although the feasibility of MPC protocols has been established more than thirty years ago, there is still a large gap in efficiency between the insecure approach (i.e. each party distributing its private inputs) and the most efficient secure realization. One important measure of complexity is the number of communication rounds between the parties required for the protocol. While the insecure approach gets by with one round, it is easy to see that any secure realization requires at least two rounds. But — even though possible — the construction of secure 2-round protocols for secure computation turns out to be much more challenging than protocols for secure computation with arbitrary round complexity. The first proposal of a two-round secure MPC protocol [GGHR14] was published only in 2014 and still relied on assuming the existence of indistinguishability obfuscation. Since then rapid progress in this area has seen a number of proposed protocols based on seemingly weaker assumptions, such as Learning with Errors [MR17, BP16, PS16] and bilinear maps [GS17, BF01, Jou04], and was ultimately based on the minimal assumption that a 2-round oblivious transfer (OT) protocol exists [BL18, GS18].

However, even latter OT-based protocols are still far from being concretely efficient, which can be explained by their inherent non-black box use of the underlying OT protocol [ABG+20]. Garg et al. [GIS18] showed that this black-box impossibility can be circumvented by giving each pair of parties access to random OT correlations. Given this setup, they presented a 2-round protocol making only a black-box use of a pseudorandom generator, and achieving security against a malicious adversary corrupting an arbitrary number of parties. Plugging in our PCF for setting up the correlated randomness results — at a one-time setup cost — in a *black-box* protocol for two-round MPC with *fully reusable preprocessing*.

10.3 NIZKs with Fully Reusable Preprocessing

Zero-knowledge proofs allow a prover to convince a verifier of a statement without revealing anything beyond this statement. Zero-knowledge proofs, in some sense, can be viewed as a special case of secure computation, where only the prover has a secret input (that is, the witness for the statement being true), and the verifier wants to receive the output (i.e. 1 if the statement is true). While non-interactive zero-knowledge (NIZK) proofs turn out to be impossible to realize in the plain model, they exist when given a common reference string, generated by a trusted third party. There has been great effort in realizing the notion of non-interactive zero knowledge proofs for general NP languages. Remarkably, Peikert and Shiehian [PS19] — following the line of work of [CCRR18, HL18, CCH+19] — gave a construction of NIZKs from plain Learning with Errors. Brakerski et al. [BKM20] recently showed that NIZKs can be constructed based on the hardness

of both the learning parity with noise (LPN) assumption and the existence of trapdoor hash functions. However, we still lack instantiations from Learning Parity with Noise assumption (or any flavors thereof) alone. Further, the results in the recent line of work — instantiating the Fiat–Shamir transform with a suitable correlation intractable hash function family — have a strong non-black-box flavor and are far from being concretely efficient. In order to overcome these limitations and enable further diversification of assumptions several relaxations of NIZKs have been introduced, such as designated verifier NIZKs (DV-NIZKs), where a trusted third party additionally gives a secret verification key to the verifier, or preprocessing NIZKs (PP-NIZKs), in which the trusted party generates — in addition to to the verification key — a secret proving key for the prover.

NIZKs from PCF for OT. Information-theoretic constructions of NIZK are known in the OT-hybrid model [KMO90,PsV06,IKOS07,GIK+15], i.e. given access to OT correlations, where — similar to secure computation — the number of OT correlations to be used scales with the size (and the number) of the theorems to be proven. Building on this line of work, Boyle et al. [BCG+19b] showed that from a PCG for OT, one can build a PP-NIZK supporting a form of *bounded reusability*, where the setup can be used to prove an arbitrary polynomial but *a-priori bounded* number of theorems. Replacing PCGs by PCFs gives rise to a construction of PP-NIZKs with *fully reusable* preprocessing, where after a one-time setup the parties can prove and verify an arbitrary number of statements. The downside of this approach is that in order to compute a NIZK for an NP-relation represented by a Boolean circuit of size s, the online phase has a computational cost that scales superlinear with s, typically multiplied by a statistical security parameter.

NIZKs from PCFs for VOLE or OLE over large fields. It turns out that by replacing OT correlation with OLE or even VOLE over $\mathbb{F}$, one can obtain much more efficient NIZK protocols that apply directly to arithmetic circuits over $\mathbb{F}$ [BCGI18, CDI+19, WYKW20]. The recent VOLE-based protocol from [WYKW20] has competitive concrete efficiency not only for arithmetic circuits but also for Boolean circuits. From an asymptotic point of view, the distinctive feature of this approach to NIZK is that the statement-dependent online phase of generating and verifying a proof has *constant computational overhead* in the arithmetic computation model, compared to evaluating the verification circuit in the clear. That is, the protocol achieves soundness error of $O(1/\mathbb{F})$ while only requiring a small constant number of field operations for each arithmetic gate of the computation being verified. This should be contrasted with zk-SNARGs, which have sublinear communication and verifier computation, but on the other hand are much heavier in terms of prover computation.

Related work. Recently, Lombardi et al. [LQR+19] showed how to construct reusable DV-NIZKs, i.e. where the verifier can reuse the verification key across many verifications, and the stronger notion of malicious DV-NIZKs, based on the generic assumption of public-key encryption together with a weak form of key-dependent secure secret-key encryption. These components can in particular be instantiated from low-noise Learning Parity with Noise. Their result is somewhat orthogonal to ours: While they obtain the stronger notion of reusable DV-NIZKs from a — technically incomparable but "morally" — weaker assumption, the main advantage of constructing PP-NIZKs via the VOLE-hybrid model is that its online phase is very lightweight and does not involve public key cryptography. As discussed above, when evaluating the PCF in a statement-independent offline phase, computing and verifying each VOLE-based proof requires only a small constant-factor overhead compared to evaluating the circuit in the clear.

10.4 Homomorphic Secret Sharing for Constant-Degree Polynomials

Homomorphic secret sharing (HSS) [BGI16a] is the dual notion to function secret sharing, allowing two (or more) parties to locally evaluate a public function on secret-shared inputs. HSS has many applications, including *succinct secure computation* in which communication complexity is smaller than the circuit size. It was observed in [BCG+19b] that a PCG for constant-degree correlations yields a homomorphic secret sharing scheme to evaluate arbitrary constant-degree polynomials. We extend their result as follows:

- The homomorphic secret sharing scheme can build on reusable setup, thereby significantly reducing the cost for sharing an input value.
- The computation time for every evaluation scales in the number of variables and number of terms of the polynomial only (instead of scaling with the number of all possible degree-d terms).

Consider two parties who wish to compute shares of $p(x)$ for some public multivariate polynomial $p(X_1, \ldots, X_m)$ over some ring R, given shares of $x = (x_1, \ldots, x_m) \in R^m$. Given a wPRF-based universal PCF for degree-d correlations (as given in Section 6.4 for the case of $R = \mathbb{F}_2$), we construct a public-key homomorphic secret sharing scheme with reusable setup $\mathsf{HSS} = (\mathsf{HSS.Gen}, \mathsf{HSS.Share}, \mathsf{HSS.Eval})$ for arbitrary degree-d polynomials as follows.

- $\mathsf{HSS.Setup}(1^\lambda)$: Generate PCF keys $(\mathsf{k}_0, \mathsf{k}_1) \leftarrow \mathsf{PCF.Gen}(1^\lambda)$ and let K the corresponding weak PRF key.
- $\mathsf{HSS.Share}(\sigma, K, x)$: Choose a value $\rho \xleftarrow{\$} R$ and output $(\rho, F_K(\rho) + x)$.
- $\mathsf{HSS.Eval}(\sigma, \mathsf{k}_\sigma, s_\sigma, p)$: On input party index $\sigma \in \{0, 1\}$, m-tuple of shares $(\rho_i, y_i)_{i \in [m]}$, and a degree-$d$ multivariate polynomial p, compute a share p'_σ of the polynomial p' satisfying $p'(X) = p(X - R)$, where $R = (F_K(\rho_1), \ldots, F_K(\rho_m))$. Note that the coefficients of p' are public degree $\leq d$ polynomials in R, hence shares of the coefficients can be locally computed via the PCF evaluation algorithm. Output $p'_\sigma(y) = \rho_\sigma(x)$.

Correctness and security follow from the properties of the underlying PCF.

10.5 Programmable PCFs with Applications to MPC with $M \geq 3$ parties

In order for a broader range of applications — that is, for example using PCFs to achieve secure computation in the multi-user setting with online communication linear in the number of users — we require the PCF to additionally satisfy *programmability*, which basically allows the parties to reuse their inputs across different PCF instantiations (compare [BCG+19b]). Note that similar to the basic construction of PCG for OT in [BCG+19b], our standard PCF for OT does not support reusability of inputs, because the sender message pairs are implicitly defined by the receiver's choice bits. But, building on the universal PCF for degree-2 correlations (see Figure 10), we will present a programmable PCF for OT in this section.

In order to define the notion of programmability formally, we first recall the definition of a simple bilinear correlation. For this we consider $\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T$ groups and $e \colon \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ be a bilinear map. Note that the size of the groups will for some applications depend on the security parameters, which will be implicit in the following.

Note that any bilinear correlation is reverse sampleable. This includes most of the commonly used correlations such as OT and OLE. For example, OT can be obtained with $\mathbb{G}_1 = \mathbb{G}_2 = \mathbb{G}_T = (\{0, 1\}, \oplus)$ and $e(x, y) = x \cdot y$ Also, note that two independent bilinear correlations can be locally converted to produce an additively secret-shared instance of the correlation — for example, two OLEs can be locally converted to one multiplication triple.

Definition 10.2 (Simple bilinear correlation) *Let $\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T$ be Abelian groups and $e\colon \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ be a bilinear map. We define $\mathcal{Y}_e$ as the algorithm that samples the* simple bilinear correlation *defined by e. More formally, let $\mathcal{Y}_e$ be the algorithm, that on input 1^λ samples from the following distribution over $(\mathbb{G}_1 \times \mathbb{G}_T) \times (\mathbb{G}_2 \times \mathbb{G}_T)$:*

$$\{((r_0, s_0), (r_1, s_1)) \mid r_0 \leftarrow \mathbb{G}_1, r_1 \leftarrow \mathbb{G}_2, s_0 \leftarrow \mathbb{G}_T, s_1 = e(r_0, r_1) - s_0\}.$$

(Recall that the groups here may depend on λ.)

Definition 10.3 (Simple Bilinear Correlation: M-party) *Given $\mathcal{Y}_e$ specified by $e\colon \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$, we define $\mathcal{Y}_{M,e}$ as the corresponding M-party correlation $\mathcal{Y}_{M,e}$, obtained by sampling from:*

$$\left\{ (a_i, b_i, c_i)_{i\in[M]} \;\middle|\; \begin{array}{c} a_i \xleftarrow{\$} \mathbb{G}_1, b_i \xleftarrow{\$} \mathbb{G}_2 \;\forall i \in [M],\; c_i \xleftarrow{\$} \mathbb{G}_T \;\forall i \in [M-1], \\ c_M = e\left(\sum_{i=1}^{M} a_i, \sum_{i=1}^{M} b_i\right) - \sum_{i=1}^{M-1} c_i \end{array} \right\}$$

Programmability for a PCF means basically, that a party can reuse its input (e.g. the receiver choice bit or the sender messages) across many instances. This is captured in the following definition.

Definition 10.4 (Programmability) *A PCF $\mathsf{PCF} = (\mathsf{PCF.Gen}, \mathsf{PCF.Eval})$ for $\mathcal{Y}_e$ (specified by $e\colon \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$) supports* reusable inputs *if there exists an algorithm $\mathsf{PCF.Gen_p}$ that takes additional random inputs $\rho_0, \rho_1 \in \{0,1\}^\star$ such that:*

- ***Indistinguishability.*** *The distributions*

$$\left\{(\mathsf{k}_0, \mathsf{k}_1) \mid (\mathsf{k}_0, \mathsf{k}_1) \leftarrow \mathsf{PCF.Gen}(1^\lambda)\right\} \text{ and}$$
$$\left\{(\mathsf{k}_0, \mathsf{k}_1) \mid (\rho_0, \rho_1) \leftarrow \$, (\mathsf{k}_0, \mathsf{k}_1) \leftarrow \mathsf{PCF.Gen_p}(1^\lambda; \rho_0, \rho_1)\right\}$$

  are computationally indistinguishable.

- ***Programmability.*** *There exist public efficiently computable functions f_0, f_1 for which*

$$\Pr\left[\begin{array}{l} \rho_0, \rho_1 \leftarrow \$, (\mathsf{k}_0, \mathsf{k}_1) \leftarrow \mathsf{PCF.Gen_p}(1^\lambda; \rho_0, \rho_1) \\ (a, c) \leftarrow \mathsf{PCF.Eval}(0, \mathsf{k}_0, x), \\ (b, d) \leftarrow \mathsf{PCF.Eval}(1, \mathsf{k}_1, x) \end{array} : \begin{array}{l} a = f_0(\rho_0, x) \\ b = f_1(\rho_1, x) \end{array}\right] \geq 1 - \mathsf{negl}(\lambda).$$

- ***Security.*** *For any $\sigma \in \{0,1\}$, the distributions*

$$\left\{(\mathsf{k}_\sigma, (\rho_0, \rho_1)) \mid \;\; (\rho_0, \rho_1) \leftarrow \$, (\mathsf{k}_0, \mathsf{k}_1) \leftarrow \mathsf{PCF.Gen_p}(1^\lambda; \rho_0, \rho_1)\right\} \quad \text{and}$$
$$\left\{(\mathsf{k}_\sigma, (\rho_\sigma, \tilde{\rho})) \mid \;\; (\rho_0, \rho_1, \tilde{\rho}) \leftarrow \$, (\mathsf{k}_0, \mathsf{k}_1) \leftarrow \mathsf{PCF.Gen_p}(1^\lambda; \rho_0, \rho_1),\right\}$$

  are computationally indistinguishable.

  To simplify notation, in the following we will write $\mathsf{PCF.Gen}$ *when referring to* $\mathsf{PCF.Gen_p}$.

Theorem 10.5 (Programmable PCF for OT) *If* DPF *is instantiated with the DPF from [GI14], then, assuming that* $\mathsf{rVDLPN}(\mathcal{O}(\lambda), D, 2^D)$ *holds,* $\mathsf{PCF} = (\mathsf{PCF.Gen}, \mathsf{PCF.Eval})$ *as defined in Figure 14 is an* programmable $(2^D, \mathsf{negl})$*-secure PCF for OT with the following complexities:*

Programmable PCF for OT based on regular VD-LPN

Let $\mathsf{DPF} = (\mathsf{DPF.Gen}, \mathsf{DPF.Eval})$ be an FSS scheme for point functions with variable input length. Recall that $f_{K_{i,j}} : \{0,1\}^i \to \{0,1\}$ is the point function with $f_{K_{i,j}}(x) = 1$ if and only if $x = K_{i,j}$.

$\mathsf{PCF.Gen}(1^\lambda; \rho_0, \rho_1)$**:** On input 1^λ and $\rho_0, \rho_1 \in \{0,1\}^{w \cdot D(D+1)/2}$:

1. Set $K = \rho_0$ and $K' = \rho_1$.
2. Parse K as $K_{i,j} \in \{0,1\}^i$, for $j \in [w], i \in [D]$ (and accordingly for K').
3. For all $j, j' \in [w]$ and $i, i' \in [D]$:
   - Sample FSS keys $K^{\mathsf{fss}}_{0,i,j,i',j'}, K^{\mathsf{fss}}_{1,i,j,i',j'} \leftarrow \mathsf{DPF.Gen}(1^\lambda, i \cdot j, f_{K_{i,j} \otimes K'_{i',j'}})$.
4. For $\sigma \in \{0,1\}$ set $K^{\mathsf{fss}}_\sigma := \{K^{\mathsf{fss}}_{\sigma,i,j,i',j'}\}_{j,j' \in [w], i,i' \in [D]}$.
5. Output the keys $\mathsf{k}_0 = (K, K^{\mathsf{fss}}_0)$ and $\mathsf{k}_1 = (K', K^{\mathsf{fss}}_1)$.

$\mathsf{PCF.Eval}(\sigma, \mathsf{k}_\sigma, x)$**:** On input a random $x \in \{0,1\}^{w \cdot D(D+1)/2}$:
Parse x as $x_{i,j} \in \{0,1\}^i$, for $j \in [w], i \in [D]$.

- If $\sigma = 0$:
  1. Compute $a \leftarrow F_K(x)$.
  2. For $j, j' \in [w]$, $i, i' \in [D]$: $c_{i,j,i',j'} \leftarrow \mathsf{DPF.Eval}(0, K^{\mathsf{fss}}_{0,i,j,i',j'}, x_{i,j} \otimes x_{i',j'})$
  3. Compute $c = \bigoplus_{j,j'=1}^{w} \bigoplus_{i,i'=1}^{D} c_{i,j,i',j'}$.
  4. Output (a, c).
- If $\sigma = 1$:
  1. Compute $b \leftarrow F_{K'}(x)$.
  2. For $j, j' \in [w]$, $i, i' \in [D]$: $d_{i,j,i',j'} \leftarrow \mathsf{DPF.Eval}(1, K^{\mathsf{fss}}_{1,i,j,i',j'}, x_{i,j} \otimes x_{i',j'})$
  3. Compute $d = \bigoplus_{j,j'=1}^{w} \bigoplus_{i,i'=1}^{D} d_{i,j,i',j'}$.
  4. Output (b, d).

Figure 14: Programmable PCF for OT based on regular VD-LPN.

- *Each party's key is of size* $\mathcal{O}(\lambda^3 \cdot D^4)$ *bits,*
- *the cost of* $\mathsf{PCF.Gen}$ *and* $\mathsf{PCF.Eval}$ *is each dominated by* $\mathcal{O}(\lambda^2 \cdot D^4)$ *invocations of a pseudorandom generator.*

Proof. [Sketch] First, note that the output of $\mathsf{PCF.Eval}$ can be transformed into an OT-correlation by setting $s_0 = c$, $s_1 = c \oplus a$, $z = b$ and $s_z = d$, as then have $(s_0 \oplus s_1) \cdot z = a \cdot b = c \oplus d = s_0 \oplus s_z$ as required (by correctness of DPF). Programmibility follows with $f_\sigma(\rho_\sigma, x) = F_{\rho_\sigma}(x)$. Finally, security follows from the security of DPF, similar to the proofs of Theorem 5.3.

□

For a generic construction of a programmable PCF for VOLE from a weak PRF together with a suitable FSS, we refer to Appendix C.

Secure Multi-Party Computation with Linear Communication. Now, we are ready to give a general transformation from any *programmable* PCF for a simple bilinear correlation

Multi-Party PCF for simple bilinear correlation

- $\mathsf{PCF}_M.\mathsf{Gen}(1^\lambda)$:
  1. Sample random $\rho_{0,1},\ldots,\rho_{0,M} \xleftarrow{\$} \{0,1\}^\lambda$, $\rho'_{1,1},\ldots,\rho_{1,M} \xleftarrow{\$} \{0,1\}^\lambda$ as specified by programmability property.
  2. For every $i \neq j \in [M]$: Run $\mathsf{k}_0^{ij}, \mathsf{k}_1^{ij} \leftarrow \mathsf{PCF.Gen}(1^\lambda, \rho_{0,i}, \rho_{1,j})$ and sample PRG keys $K^{ij} \xleftarrow{\$} \{0,1\}^\lambda$
  3. For each $i \in [M]$, output $\mathsf{k}_i = \left(\{K^{ij}\}_{j\neq i}, \{\mathsf{k}_0^{ij}\}_{j\neq i}, \{\mathsf{k}_1^{ji}\}_{j\neq i}\right)$
- $\mathsf{PCF}_M.\mathsf{Eval}(i, \mathsf{k}_i, x)$:
  1. For every $j \neq i$, compute
  $$r_{ij} \leftarrow \mathsf{PRF}_{K^{ij}}(x),$$
  $$(a_{ij}, c_{ij}) \leftarrow \mathsf{PCF.Eval}(0, \mathsf{k}_0^{ij}, x), (b_{ji}, d_{ji}) \leftarrow \mathsf{PCF.Eval}(1, \mathsf{k}_1^{ji}, x)$$
  2. Output $A_i = a_{ij}$, $B_i = b_{ji}$ (same for all j) and $C_i = -\sum_{j\neq i} c_{ij} + \sum_{j\neq i} d_{ji} + \sum_{j\neq i} (-1)^{[i<j]} r_{ij} + e(A_i, B_i)$, where $[i<j] = 1$ if $i<j$ and 0 if $j<i$.

Figure 15: Multi-party PCF for simple bilinear correlation.

to an M-party PCFs for the corresponding M-party correlation. This has implications to the secure computation in the multi-user setting, as it allows to obtain secure M-party computation protocols with total communication complexity $\mathcal{O}(Ms + M^2 \cdot \mathsf{cost}_{\mathsf{PCF.Gen}})$, where s is the circuit size and $\mathsf{cost}_{\mathsf{PCF.Gen}}$ the cost for setting up the seeds of the two-party PCF, which is independent of the circuit size s. In contrast, OT-based MPC protocols have total online communication complexity $\mathcal{O}(M^2 s)$ [GMW87, HOSS18].

Our approach closely follows [BCG+19b], who gave a similar transformation for the context of PCGs. But, due to the fact that PCGs are inherently limited to polynomial stretch, in their case the communication complexity of setup scaled in $\mathcal{O}(M^2 s^\varepsilon)$ for a constant ε. In particular, for their approach an upper bound on the circuit size had to be known at the time of setup. PCFs, on the other hand, allow secure computation with linear communication after a fully-reusable one-time setup. Note that traditional approaches with linear online communication complexity even require an offline communication of $\mathcal{O}(M^2 s)$ [FH96, CDN01, DPSZ12].

In the following we present a generic transformation from any *programmable* PCF for a simple bilinear correlation to a M-party PCF for the corresponding multi-party correlation. For a formal definition of multi-party PCFs, we refer to Appendix D. We omit the proof as it is straightforward.

Theorem 10.6 (Multi-party simple bilinear PCF) *Let* $\mathsf{PCF} = (\mathsf{PCF.Gen}, \mathsf{PCF.Eval})$ *be a PCF with reusable inputs for* $\mathcal{Y}_e$ *(specified by* $e\colon \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$*) with key sizes* $s_0(\lambda), s_1(\lambda)$ *and let* PRF *be a weak PRF. Then the* $\mathsf{PCF}_M = (\mathsf{PCF}_M.\mathsf{Gen}, \mathsf{PCF}_M.\mathsf{Eval})$ *as defined in Figure 15 is a PCF for the corresponding* M*-party correlation* $\mathcal{Y}_{M,e}$ *with the following properties.*

- $\mathsf{PCF}_M.\mathsf{Gen}(1^\lambda)$ *runs* $M(M-1)$ *executions of* PCF.Gen*; each output key* k_i*,* $i \in [M]$*, has size* $(M-1)(s_0(\lambda) + s_1(\lambda) + \lambda)$ *bits.*
- $\mathsf{PCF}_M.\mathsf{Eval}(i, \mathsf{k}_i, x)$ *runs* $2(M-1)$ *executions of* PCF.Eval *and makes* $(M-1)$ *evaluations of* PRF*.*

11 Acknowledgements

We would like to thank Nader Bshouty, Rocco Servedio, Jean-Pierre Tillich, Nicolas Sendrier, and Thomas Debris for useful discussions and pointers, and the anonymous FOCS reviewers for helpful suggestions.

E. Boyle, N. Gilboa, and Y. Ishai supported by ERC Project NTSC (742754). E. Boyle additionally supported by ISF grant 1861/16, AFOSR Award FA9550-17-1-0069, and ERC Project HSS (852952). G. Couteau supported by ERC Project PREP-CRYPTO (724307). N. Gilboa additionally supported by ISF grant 2951/20, ERC grant 876110, and a grant by the BGU Cyber Center. Y. Ishai additionally supported by NSF-BSF grant 2015782, BSF grant 2018393, ISF grant 2774/20, and a grant from the Ministry of Science and Technology, Israel and Department of Science and Technology, Government of India. L. Kohl is funded by NWO Gravitation project QSC. Most research of L. Kohl was done while at Technion, supported by ERC Project NTSC (742754). Research of L. Kohl was done in part while at Karlsruhe Institute of Technology, supported by ERC Project PREP-CRYPTO (724307) and DFG grant HO 4534/2-2. P. Scholl supported by the Danish Independent Research Council under Grant-ID DFF-6108-00169 (FoCC) and an Aarhus University Research Foundation starting grant.

References

[AAB15] B. Applebaum, J. Avron, and C. Brzuska. Arithmetic cryptography: Extended abstract. In *ITCS 2015*, pages 143–151. ACM, 2015.

[ABG+14] A. Akavia, A. Bogdanov, S. Guo, A. Kamath, and A. Rosen. Candidate weak pseudorandom functions in AC^0 *o* MOD_2. In *ITCS 2014*, pages 251–260. ACM, 2014.

[ABG+20] B. Applebaum, Z. Brakerski, S. Garg, Y. Ishai, and A. Srinivasan. Separating two-round secure computation from oblivious transfer. pages 71:1–71:18, 2020.

[ABP19] M. Abdalla, F. Benhamouda, and A. Passelègue. Algebraic XOR-RKA-secure pseudorandom functions from post-zeroizing multilinear maps. LNCS, pages 386–412. Springer, December 2019.

[ACG+06] F. Armknecht, C. Carlet, P. Gaborit, S. Künzli, W. Meier, and O. Ruatta. Efficient computation of algebraic immunity for algebraic and fast algebraic attacks. In *EUROCRYPT 2006*, *LNCS* 4004, pages 147–164. Springer, May / June 2006.

[ADI+17] B. Applebaum, I. Damgård, Y. Ishai, M. Nielsen, and L. Zichron. Secure arithmetic computation with constant computational overhead. LNCS, pages 223–254. Springer, 2017.

[AFS03] D. Augot, M. Finiasz, and N. Sendrier. A fast provably secure cryptographic hash function. Cryptology ePrint Archive, Report 2003/230, 2003. http://eprint.iacr.org/2003/230.

[AHI11] B. Applebaum, D. Harnik, and Y. Ishai. Semantic security under related-key attacks and applications. In *ICS 2011*, pages 45–60. Tsinghua University Press, January 2011.

[AIK04] B. Applebaum, Y. Ishai, and E. Kushilevitz. Cryptography in NC^0. In *45th FOCS*, pages 166–175. IEEE Computer Society Press, October 2004.

[AJ01] A. Al Jabri. A statistical decoding algorithm for general linear block codes. In *IMA International Conference on Cryptography and Coding*, pages 1–8. Springer, 2001.

[AK19] B. Applebaum and E. Kachlon. Sampling graphs without forbidden subgraphs and unbalanced expanders with negligible error. In *FOCS 2019*, pages 171–179, 2019.

[AKPW13] J. Alwen, S. Krenn, K. Pietrzak, and D. Wichs. Learning with rounding, revisited - new reduction, properties and applications. In *CRYPTO 2013, Part I*, *LNCS* 8042, pages 57–74. Springer, August 2013.

[AL18] B. Applebaum and S. Lovett. Algebraic attacks against random local functions and their countermeasures. *SIAM Journal on Computing*, 47(1):52–79, 2018.

[AR16] B. Applebaum and P. Raykov. Fast pseudorandom functions based on expander graphs. LNCS, pages 27–56. Springer, 2016.

[ARS+15] M. R. Albrecht, C. Rechberger, T. Schneider, T. Tiessen, and M. Zohner. Ciphers for MPC and FHE. LNCS, pages 430–454. Springer, 2015.

[AW14] B. Applebaum and E. Widder. Related-key secure pseudorandom functions: The case of additive attacks. Cryptology ePrint Archive, Report 2014/478, 2014. http://eprint.iacr.org/2014/478.

[Azu67] K. Azuma. Weighted sums of certain dependent random variables. *Tohoku Mathematical Journal, Second Series*, 19(3):357–367, 1967.

[BCG+17] E. Boyle, G. Couteau, N. Gilboa, Y. Ishai, and M. Orrù. Homomorphic secret sharing: Optimizations and applications. In *ACM CCS 17*, pages 2105–2122. ACM Press, 2017.

[BCG+19a] E. Boyle, G. Couteau, N. Gilboa, Y. Ishai, L. Kohl, P. Rindal, and P. Scholl. Efficient two-round OT extension and silent non-interactive secure computation. In *ACM CCS 19*, pages 291–308. ACM Press, 2019.

[BCG+19b] E. Boyle, G. Couteau, N. Gilboa, Y. Ishai, L. Kohl, and P. Scholl. Efficient pseudorandom correlation generators: Silent OT extension and more. LNCS, pages 489–518. Springer, 2019.

[BCG+20] E. Boyle, G. Couteau, N. Gilboa, Y. Ishai, L. Kohl, and P. Scholl. Efficient pseudorandom correlation generators from ring-lpn. 2020. To appear on eprint.

[BCGI18] E. Boyle, G. Couteau, N. Gilboa, and Y. Ishai. Compressing vector OLE. In *ACM CCS 18*, pages 896–912. ACM Press, 2018.

[BCM11] M. Bellare, D. Cash, and R. Miller. Cryptography secure against related-key attacks and tampering. In *ASIACRYPT 2011*, *LNCS* 7073, pages 486–503. Springer, December 2011.

[BDOZ11] R. Bendlin, I. Damgård, C. Orlandi, and S. Zakarias. Semi-homomorphic encryption and multiparty computation. In *EUROCRYPT 2011*, *LNCS* 6632, pages 169–188. Springer, May 2011.

[BDVY13] A. Bogdanov, Z. Dvir, E. Verbin, and A. Yehudayoff. Pseudorandomness for width-2 branching programs. *Theory of Computing*, 9(1):283–293, 2013.

[Bea91] D. Beaver. Efficient multiparty protocols using circuit randomization. In *CRYPTO '91*, pages 420–432, 1991.

[Bea95] D. Beaver. Precomputing oblivious transfer. In *CRYPTO'95*, *LNCS* 963, pages 97–109. Springer, August 1995.

[BF01] D. Boneh and M. K. Franklin. Identity-based encryption from the Weil pairing. In *CRYPTO 2001*, *LNCS* 2139, pages 213–229. Springer, August 2001.

[BFKL94] A. Blum, M. L. Furst, M. J. Kearns, and R. J. Lipton. Cryptographic primitives based on hard learning problems. In *CRYPTO'93*, *LNCS* 773, pages 278–291. Springer, August 1994.

[BGI14] E. Boyle, S. Goldwasser, and I. Ivan. Functional signatures and pseudorandom functions. In *PKC 2014*, LNCS, pages 501–519. Springer, 2014.

[BGI15] E. Boyle, N. Gilboa, and Y. Ishai. Function secret sharing. LNCS, pages 337–367. Springer, 2015.

[BGI16a] E. Boyle, N. Gilboa, and Y. Ishai. Breaking the circuit size barrier for secure computation under DDH. In *CRYPTO 2016, Part I*, LNCS, pages 509–539. Springer, August 2016.

[BGI16b] E. Boyle, N. Gilboa, and Y. Ishai. Function secret sharing: Improvements and extensions. In *ACM CCS 16*, pages 1292–1303. ACM Press, 2016.

[BGI+18] E. Boyle, N. Gilboa, Y. Ishai, H. Lin, and S. Tessaro. Foundations of homomorphic secret sharing. pages 21:1–21:21, 2018.

[BGI19] E. Boyle, N. Gilboa, and Y. Ishai. Secure computation with preprocessing via function secret sharing. LNCS, pages 341–371. Springer, 2019.

[BGMM20] J. Bartusek, S. Garg, D. Masny, and P. Mukherjee. Reusable two-round MPC from DDH, 2020. https://eprint.iacr.org/2020/170.

[BIP+18] D. Boneh, Y. Ishai, A. Passelègue, A. Sahai, and D. J. Wu. Exploring crypto dark matter: New simple PRF candidates and their applications. LNCS, pages 699–729. Springer, 2018.

[BJMM12] A. Becker, A. Joux, A. May, and A. Meurer. Decoding random binary linear codes in $2^{n/20}$: How $1 + 1 = 0$ improves information set decoding. In *EUROCRYPT 2012*, *LNCS* 7237, pages 520–536. Springer, April 2012.

[BK03] M. Bellare and T. Kohno. A theoretical treatment of related-key attacks: RKA-PRPs, RKA-PRFs, and applications. In *EUROCRYPT 2003*, *LNCS* 2656, pages 491–506. Springer, May 2003.

[BKM20] Z. Brakerski, V. Koppula, and T. Mour. NIZK from LPN and trapdoor hash via correlation intractability for approximable relations. LNCS, pages 738–767. Springer, 2020.

[BKW00] A. Blum, A. Kalai, and H. Wasserman. Noise-tolerant learning, the parity problem, and the statistical query model. In *32nd ACM STOC*, pages 435–440. ACM Press, May 2000.

[BL18] F. Benhamouda and H. Lin. k-round multiparty computation from k-round oblivious transfer via garbled interactive circuits. LNCS, pages 500–532. Springer, 2018.

[BLP11] D. J. Bernstein, T. Lange, and C. Peters. Smaller decoding exponents: Ball-collision decoding. In *CRYPTO 2011*, *LNCS* 6841, pages 743–760. Springer, August 2011.

[BLVW19] Z. Brakerski, V. Lyubashevsky, V. Vaikuntanathan, and D. Wichs. Worst-case hardness for LPN and cryptographic hashing via code smoothing. LNCS, pages 619–635. Springer, 2019.

[BP16] Z. Brakerski and R. Perlman. Lattice-based fully dynamic multi-key FHE with short ciphertexts. In *CRYPTO 2016, Part I*, LNCS, pages 190–213. Springer, August 2016.

[BPR12] A. Banerjee, C. Peikert, and A. Rosen. Pseudorandom functions and lattices. In *EUROCRYPT 2012*, *LNCS* 7237, pages 719–737. Springer, April 2012.

[BR17] A. Bogdanov and A. Rosen. Pseudorandom functions: Three decades later. Cryptology ePrint Archive, Report 2017/652, 2017. http://eprint.iacr.org/2017/652.

[Bra08] M. Braverman. Polylogarithmic independence fools AC0 circuits. *Journal of the ACM (JACM)*, 57(5):1–10, 2008.

[Bsh97] N. H. Bshouty. On learning multivariate polynomials under the uniform distribution. *Inf. Process. Lett.*, 61(6):303–309, 1997.

[Bsh20] N. H. Bshouty. Pac learning sparse polynomials. Private communication, 2020.

[BW13] D. Boneh and B. Waters. Constrained pseudorandom functions and their applications. In *ASIACRYPT 2013, Part II*, *LNCS* 8270, pages 280–300. Springer, December 2013.

[Car20] C. Carlet. *Boolean Functions for Cryptography and Coding Theory.* Cambridge University Press, 2020.

[CCF+16] A. Canteaut, S. Carpov, C. Fontaine, T. Lepoint, M. Naya-Plasencia, P. Paillier, and R. Sirdey. Stream ciphers: A practical solution for efficient homomorphic-ciphertext compression. In *FSE 2016*, LNCS, pages 313–333. Springer, 2016.

[CCH+19] R. Canetti, Y. Chen, J. Holmgren, A. Lombardi, G. N. Rothblum, R. D. Rothblum, and D. Wichs. Fiat-Shamir: from practice to theory. In *51st ACM STOC*, pages 1082–1090. ACM Press, 2019.

[CCRR18] R. Canetti, Y. Chen, L. Reyzin, and R. D. Rothblum. Fiat-Shamir and correlation intractability from strong KDM-secure encryption. LNCS, pages 91–122. Springer, 2018.

[CDI+19] M. Chase, Y. Dodis, Y. Ishai, D. Kraschewski, T. Liu, R. Ostrovsky, and V. Vaikuntanathan. Reusable non-interactive secure computation. LNCS, pages 462–488. Springer, 2019.

[CDN01] R. Cramer, I. Damgård, and J. B. Nielsen. Multiparty computation from threshold homomorphic encryption. In *Advances in Cryptology - EUROCRYPT 2001, International Conference on the Theory and Application of Cryptographic Techniques, Innsbruck, Austria, May 6-10, 2001, Proceeding*, pages 280–299, 2001.

[CIKK16] M. L. Carmosino, R. Impagliazzo, V. Kabanets, and A. Kolokolova. Learning algorithms from natural proofs. In *CCC 2016*, pages 10:1–10:24, 2016.

[CKPS00] N. Courtois, A. Klimov, J. Patarin, and A. Shamir. Efficient algorithms for solving overdefined systems of multivariate polynomial equations. In *International Conference on the Theory and Applications of Cryptographic Techniques*, pages 392–407. Springer, 2000.

[CM03] N. T. Courtois and W. Meier. Algebraic attacks on stream ciphers with linear feedback. In *International Conference on the Theory and Applications of Cryptographic Techniques*, pages 345–359. Springer, 2003.

[Cou01] N. T. Courtois. The security of hidden field equations (hfe). In *Cryptographers' Track at the RSA Conference*, pages 266–281. Springer, 2001.

[Cou03] N. T. Courtois. Fast algebraic attacks on stream ciphers with linear feedback. In *Annual International Cryptology Conference*, pages 176–194. Springer, 2003.

[Cou04] N. T. Courtois. General principles of algebraic attacks and new design criteria for cipher components. In *International Conference on Advanced Encryption Standard*, pages 67–83. Springer, 2004.

[DAT17] T. Debris-Alazard and J.-P. Tillich. Statistical decoding. In *2017 IEEE International Symposium on Information Theory (ISIT)*, pages 1798–1802. IEEE, 2017.

[DGJ+09] I. Diakonikolas, P. Gopalan, R. Jaiswal, R. A. Servedio, and E. Viola. Bounded independence fools halfspaces. In *50th FOCS*, pages 171–180. IEEE Computer Society Press, October 2009.

[DHRW16] Y. Dodis, S. Halevi, R. D. Rothblum, and D. Wichs. Spooky encryption and its applications. LNCS, pages 93–122. Springer, August 2016.

[DKN10] I. Diakonikolas, D. M. Kane, and J. Nelson. Bounded independence fools degree-2 threshold functions. In *51st FOCS*, pages 11–20. IEEE Computer Society Press, October 2010.

[DPSZ12] I. Damgård, V. Pastro, N. P. Smart, and S. Zakarias. Multiparty computation from somewhat homomorphic encryption. In *CRYPTO 2012*, *LNCS* 7417, pages 643–662. Springer, August 2012.

[Ds17] J. Doerner and a. shelat. Scaling ORAM for secure computation. In *ACM CCS 17*, pages 523–535. ACM Press, 2017.

[EKM17] A. Esser, R. Kübler, and A. May. LPN decoded. LNCS, pages 486–514. Springer, 2017.

[Fau99] J.-C. Faugere. A new efficient algorithm for computing gröbner bases (f4). *Journal of pure and applied algebra*, 139(1-3):61–88, 1999.

[Fau02] J. C. Faugère. A new efficient algorithm for computing gröbner bases without reduction to zero (f 5). In *Proceedings of the 2002 international symposium on Symbolic and algebraic computation*, pages 75–83, 2002.

[FH96] M. K. Franklin and S. Haber. Joint encryption and message-efficient secure computation. *J. Cryptology*, 9(4):217–232, 1996.

[FKI06] M. P. Fossorier, K. Kobara, and H. Imai. Modeling bit flipping decoding based on nonorthogonal check sums with application to iterative decoding attack of mceliece cryptosystem. *IEEE Transactions on Information Theory*, 53(1):402–411, 2006.

[FS09] M. Finiasz and N. Sendrier. Security bounds for the design of code-based cryptosystems. In *ASIACRYPT 2009*, *LNCS* 5912, pages 88–105. Springer, December 2009.

[GGHR14] S. Garg, C. Gentry, S. Halevi, and M. Raykova. Two-round secure MPC from indistinguishability obfuscation. In *TCC 2014*, *LNCS* 8349, pages 74–94. Springer, February 2014.

[GGM84] O. Goldreich, S. Goldwasser, and S. Micali. How to construct random functions (extended abstract). In *25th FOCS*, pages 464–479. IEEE Computer Society Press, October 1984.

[GGM86] O. Goldreich, S. Goldwasser, and S. Micali. How to construct random functions. *J. ACM*, 33(4):792–807, 1986.

[GHS12] C. Gentry, S. Halevi, and N. P. Smart. Fully homomorphic encryption with polylog overhead. In *EUROCRYPT 2012*, *LNCS* 7237, pages 465–482. Springer, April 2012.

[GI14] N. Gilboa and Y. Ishai. Distributed point functions and their applications. In *EUROCRYPT 2014*, LNCS, pages 640–658. Springer, 2014.

[GIK+15] S. Garg, Y. Ishai, E. Kushilevitz, R. Ostrovsky, and A. Sahai. Cryptography with one-way communication. In *CRYPTO 2015, Part II*, LNCS, pages 191–208. Springer, August 2015.

[GIS18] S. Garg, Y. Ishai, and A. Srinivasan. Two-round MPC: Information-theoretic and black-box. LNCS, pages 123–151. Springer, 2018.

[GLM+04] R. Gennaro, A. Lysyanskaya, T. Malkin, S. Micali, and T. Rabin. Algorithmic tamper-proof (ATP) security: Theoretical foundations for security against hardware tampering. In *TCC 2004*, *LNCS* 2951, pages 258–277. Springer, February 2004.

[GMW87] O. Goldreich, S. Micali, and A. Wigderson. How to play any mental game or A completeness theorem for protocols with honest majority. In *19th ACM STOC*, pages 218–229. ACM Press, May 1987.

[Gol04] O. Goldreich. *Foundations of Cryptography: Volume 2, Basic Applications*. Cambridge University Press, New York, NY, USA, 2004.

[GS17] S. Garg and A. Srinivasan. Garbled protocols and two-round MPC from bilinear maps. In *58th FOCS*, pages 588–599. IEEE Computer Society Press, 2017.

[GS18] S. Garg and A. Srinivasan. Two-round multiparty secure computation from minimal assumptions. LNCS, pages 468–499. Springer, 2018.

[GV04] D. Gutfreund and E. Viola. Fooling parity tests with parity gates. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques*, pages 381–392. Springer, 2004.

[Hea08] A. D. Healy. Randomness-efficient sampling within nc. *Computational Complexity*, 17(1):3–37, 2008.

[HKL+12] S. Heyse, E. Kiltz, V. Lyubashevsky, C. Paar, and K. Pietrzak. Lapin: An efficient authentication protocol based on ring-LPN. In *FSE 2012*, *LNCS* 7549, pages 346–365. Springer, March 2012.

[HL18] J. Holmgren and A. Lombardi. Cryptographic hashing from strong one-way functions (or: One-way product functions and their applications). In *59th FOCS*, pages 850–858. IEEE Computer Society Press, 2018.

[HOSS18] C. Hazay, E. Orsini, P. Scholl, and E. Soria-Vazquez. TinyKeys: A new approach to efficient multi-party computation. LNCS, pages 3–33. Springer, 2018.

[HS07] L. Hellerstein and R. A. Servedio. On PAC learning algorithms for rich boolean function classes. *Theor. Comput. Sci.*, 384(1):66–76, 2007.

[IKNP03] Y. Ishai, J. Kilian, K. Nissim, and E. Petrank. Extending oblivious transfers efficiently. In *CRYPTO 2003*, *LNCS* 2729, pages 145–161. Springer, August 2003.

[IKOS07] Y. Ishai, E. Kushilevitz, R. Ostrovsky, and A. Sahai. Zero-knowledge from secure multiparty computation. In *39th ACM STOC*, pages 21–30. ACM Press, June 2007.

[IKOS08] Y. Ishai, E. Kushilevitz, R. Ostrovsky, and A. Sahai. Cryptography with constant computational overhead. In *40th ACM STOC*, pages 433–442. ACM Press, May 2008.

[IPS08] Y. Ishai, M. Prabhakaran, and A. Sahai. Founding cryptography on oblivious transfer - efficiently. In *CRYPTO 2008*, *LNCS* 5157, pages 572–591. Springer, August 2008.

[Jou04] A. Joux. A one round protocol for tripartite Diffie-Hellman. *Journal of Cryptology*, 17(4):263–276, September 2004.

[Kea98] M. Kearns. Efficient noise-tolerant learning from statistical queries. *Journal of the ACM (JACM)*, 45(6):983–1006, 1998.

[Kha93] M. Kharitonov. Cryptographic hardness of distribution-specific learning. In *25th ACM STOC*, pages 372–381. ACM Press, May 1993.

[Kil88] J. Kilian. Founding cryptography on oblivious transfer. In *20th ACM STOC*, pages 20–31. ACM Press, May 1988.

[KMO89] J. Kilian, S. Micali, and R. Ostrovsky. Minimum resource zero-knowledge proofs (extended abstract). In *30th FOCS*, pages 474–479. IEEE Computer Society Press, October / November 1989.

[KMO90] J. Kilian, S. Micali, and R. Ostrovsky. Minimum resource zero-knowledge proofs (extended abstract). In *CRYPTO'89*, *LNCS* 435, pages 545–546. Springer, August 1990.

[KPTZ13] A. Kiayias, S. Papadopoulos, N. Triandopoulos, and T. Zacharias. Delegatable pseudorandom functions and applications. In *ACM CCS 13*, pages 669–684. ACM Press, November 2013.

[KRRW18] J. Katz, S. Ranellucci, M. Rosulek, and X. Wang. Optimizing authenticated garbling for faster secure two-party computation. LNCS, pages 365–391. Springer, 2018.

[LMN89] N. Linial, Y. Mansour, and N. Nisan. Constant depth circuits, Fourier transform, and learnability. In *30th FOCS*, pages 574–579. IEEE Computer Society Press, October / November 1989.

[LQR+19] A. Lombardi, W. Quach, R. D. Rothblum, D. Wichs, and D. J. Wu. New constructions of reusable designated-verifier NIZKs. LNCS, pages 670–700. Springer, 2019.

[LRTV09] S. Lovett, O. Reingold, L. Trevisan, and S. Vadhan. Pseudorandom bit generators that fool modular sums. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques*, pages 615–630. Springer, 2009.

[LT17] H. Lin and S. Tessaro. Indistinguishability obfuscation from trilinear maps and block-wise local PRGs. LNCS, pages 630–660. Springer, 2017.

[LV17] C. H. Lee and E. Viola. Some limitations of the sum of small-bias distributions. *Theory of Computing*, 13(1):1–23, 2017.

[Lyu05] V. Lyubashevsky. The parity problem in the presence of noise, decoding random linear codes, and the subset sum problem. In *Approximation, randomization and combinatorial optimization. Algorithms and techniques*, pages 378–389. Springer, 2005.

[Mat94] M. Matsui. Linear cryptanalysis method for DES cipher. In *EUROCRYPT'93*, *LNCS* 765, pages 386–397. Springer, May 1994.

[MBD+18] C. A. Melchor, O. Blazy, J. Deneuville, P. Gaborit, and G. Zémor. Efficient encryption from random quasi-cyclic codes. *IEEE Trans. Information Theory*, 64(5):3927–3943, 2018.

[McD89] C. McDiarmid. On the method of bounded differences, in "survey in combinatorics,"(j. simons, ed.) london mathematical society lecture notes, vol. 141, 1989.

[MJSC16] P. Méaux, A. Journault, F.-X. Standaert, and C. Carlet. Towards stream ciphers for efficient FHE with low-noise ciphertexts. LNCS, pages 311–343. Springer, 2016.

[MMT11] A. May, A. Meurer, and E. Thomae. Decoding random linear codes in $\tilde{\mathcal{O}}(2^{0.054n})$. In *ASIACRYPT 2011*, *LNCS* 7073, pages 107–124. Springer, December 2011.

[MO15] A. May and I. Ozerov. On computing nearest neighbors with applications to decoding of binary linear codes. LNCS, pages 203–228. Springer, 2015.

[MR17] P. Mohassel and M. Rosulek. Non-interactive secure 2PC in the offline/online and batch settings. LNCS, pages 425–455. Springer, 2017.

[MR18] P. Mohassel and P. Rindal. ABY^3: A mixed protocol framework for machine learning. In *ACM CCS 18*, pages 35–52. ACM Press, 2018.

[MRRR14] R. Meka, O. Reingold, G. N. Rothblum, and R. D. Rothblum. Fast pseudorandomness for independence and load balancing. In *International Colloquium on Automata, Languages, and Programming*, pages 859–870. Springer, 2014.

[MST03] E. Mossel, A. Shpilka, and L. Trevisan. On e-biased generators in NC0. In *44th FOCS*, pages 136–145. IEEE Computer Society Press, October 2003.

[MTSB13] R. Misoczki, J. Tillich, N. Sendrier, and P. S. L. M. Barreto. Mdpc-mceliece: New mceliece variants from moderate density parity-check codes. In *2013 IEEE International Symposium on Information Theory*, pages 2069–2073, 2013.

[MV11] E. Miles and E. Viola. On the complexity of non-adaptively increasing the stretch of pseudorandom generators. In *TCC 2011*, *LNCS* 6597, pages 522–539. Springer, March 2011.

[MV12] E. Miles and E. Viola. Substitution-permutation networks, pseudorandom functions, and natural proofs. In *CRYPTO 2012*, *LNCS* 7417, pages 68–85. Springer, August 2012.

[NN90] J. Naor and M. Naor. Small-bias probability spaces: Efficient constructions and applications. In *22nd ACM STOC*, pages 213–223. ACM Press, May 1990.

[NNOB12] J. B. Nielsen, P. S. Nordholt, C. Orlandi, and S. S. Burra. A new approach to practical active-secure two-party computation. In *CRYPTO 2012*, *LNCS* 7417, pages 681–700. Springer, August 2012.

[NR95] M. Naor and O. Reingold. Synthesizers and their application to the parallel construction of pseudo-random functions. In *36th FOCS*, pages 170–181. IEEE Computer Society Press, October 1995.

[NR97] M. Naor and O. Reingold. Number-theoretic constructions of efficient pseudo-random functions. In *38th FOCS*, pages 458–467. IEEE Computer Society Press, October 1997.

[NR04] M. Naor and O. Reingold. Number-theoretic constructions of efficient pseudo-random functions. *Journal of the ACM*, 51(2):231–262, 2004.

[NRR00] M. Naor, O. Reingold, and A. Rosen. Pseudo-random functions and factoring (extended abstract). In *32nd ACM STOC*, pages 11–20. ACM Press, May 2000.

[Ove06] R. Overbeck. Statistical decoding revisited. In *ACISP 06*, *LNCS* 4058, pages 283–294. Springer, July 2006.

[Pat95] J. Patarin. Cryptanalysis of the matsumoto and imai public key scheme of eurocrypt'88. In *Annual International Cryptology Conference*, pages 248–261. Springer, 1995.

[Pie12] K. Pietrzak. Cryptography from learning parity with noise. In *International Conference on Current Trends in Theory and Practice of Computer Science*, pages 99–114. Springer, 2012.

[Pra62] E. Prange. The use of information sets in decoding cyclic codes. *IRE Transactions on Information Theory*, 8(5):5–9, 1962.

[PS16] C. Peikert and S. Shiehian. Multi-key FHE from LWE, revisited. LNCS, pages 217–238. Springer, 2016.

[PS19] C. Peikert and S. Shiehian. Noninteractive zero knowledge for NP from (plain) learning with errors. LNCS, pages 89–114. Springer, 2019.

[PsV06] R. Pass, a. shelat, and V. Vaikuntanathan. Construction of a non-malleable encryption scheme from any semantically secure one. In *CRYPTO 2006*, *LNCS* 4117, pages 271–289. Springer, August 2006.

[Reg05] O. Regev. On lattices, learning with errors, random linear codes, and cryptography. In *37th ACM STOC*, pages 84–93. ACM Press, May 2005.

[RR97] A. A. Razborov and S. Rudich. Natural proofs. *J. Comput. Syst. Sci.*, 55(1):24–35, 1997.

[Sch18] P. Scholl. Extending oblivious transfer with low communication via key-homomorphic PRFs. LNCS, pages 554–583. Springer, 2018.

[SGRR19] P. Schoppmann, A. Gascón, L. Reichert, and M. Raykova. Distributed vector-OLE: Improved constructions and implementation. In *ACM CCS 19*, pages 1055–1072. ACM Press, 2019.

[Shp09] A. Shpilka. Constructions of low-degree and error-correcting ε-biased generators. *computational complexity*, 18(4):495, 2009.

[SS96] R. E. Schapire and L. Sellie. Learning sparse multivariate polynomials over a field with queries and counterexamples. *J. Comput. Syst. Sci.*, 52(2):201–213, 1996.

[Ste88] J. Stern. A method for finding codewords of small weight. In *International Colloquium on Coding Theory and Applications*, pages 106–113. Springer, 1988.

[Vio09] E. Viola. The sum of d small-bias generators fools polynomials of degree d. *Computational Complexity*, 18(2):209–217, 2009.

[Vio10] E. Viola. The complexity of distributions. In *51st FOCS*, pages 202–211. IEEE Computer Society Press, October 2010.

[Vio13] E. Viola. The communication complexity of addition. In *24th SODA*, pages 632–651. ACM-SIAM, January 2013.

[WRK17a] X. Wang, S. Ranellucci, and J. Katz. Authenticated garbling and efficient maliciously secure two-party computation. In *ACM CCS 17*, pages 21–37. ACM Press, 2017.

[WRK17b] X. Wang, S. Ranellucci, and J. Katz. Global-scale secure multiparty computation. In *ACM CCS 17*, pages 39–56. ACM Press, 2017.

[WYKW20] C. Weng, K. Yang, J. Katz, and X. Wang. Fast, scalable, and communication-efficient zero-knowledge proofs for boolean and arithmetic circuits. *IACR Cryptol. ePrint Arch.*, 2020:925, 2020.

[YS16] Y. Yu and J. P. Steinberger. Pseudorandom functions in almost constant depth from low-noise LPN. LNCS, pages 154–183. Springer, 2016.

[YZW+19] Y. Yu, J. Zhang, J. Weng, C. Guo, and X. Li. Collision resistant hashing from sub-exponential learning parity with noise. LNCS, pages 3–24. Springer, December 2019.

[Zic17] L. Zichron. Locally computable arithmetic pseudorandom generators. Master's thesis, School of Electrical Engineering, Tel Aviv University, 2017.

Appendix

A Pseudorandom Correlation Generators

In the following we recall the definition of a pseudorandom correlation generator from [BCG+19b].

Definition A.1 (Correlation generator) *A PPT algorithm $\mathcal{C}$ is called a* correlation generator*, if $\mathcal{C}$ on input 1^λ outputs a pair of elements in $\{0,1\}^{N(\lambda)} \times \{0,1\}^{N(\lambda)}$, where $N(\lambda) \leq \mathsf{poly}(\lambda)$.*

Definition A.2 (Reverse-sampleable correlation generator) *Let $\mathcal{C}$ be a correlation generator. We say $\mathcal{C}$ is* reverse sampleable *if there exists a PPT algorithm* RSample *that on input 1^λ, $\sigma \in \{0,1\}$ and $R_\sigma \in \{0,1\}^{N(\lambda)}$ outputs $R_{1-\sigma} \in \{0,1\}^{N(\lambda)}$ such that for $\sigma \in \{0,1\}$ the correlation obtained via:*

$$\{(R'_0, R'_1) \,|(R_0, R_1) \leftarrow \mathcal{C}(1^\lambda), R'_\sigma := R_\sigma, R'_{1-\sigma} \leftarrow \mathsf{RSample}(1^\lambda, \sigma, R_\sigma)\}$$

is computationally indistinguishable from $\mathcal{C}(1^\lambda)$.

Definition A.3 (Pseudorandom correlation generator (PCG)) *Let $\mathcal{C}$ be a reverse-sampleable correlation generator. A* pseudorandom correlation generator (PCG) for $\mathcal{C}$ *is a pair of algorithms* $(\mathsf{PCG.Gen}, \mathsf{PCG.Expand})$ *with the following syntax:*

- $\mathsf{PCG.Gen}(1^\lambda)$ *is a PPT algorithm that given a security parameter λ, outputs a pair of seeds $(\mathsf{k}_0, \mathsf{k}_1)$;*
- $\mathsf{PCG.Expand}(\sigma, \mathsf{k}_\sigma)$ *is a polynomial-time algorithm that on input $\sigma \in \{0,1\}$ and a seed k_σ, outputs a bit string $R_\sigma \in \{0,1\}^{N(\lambda)}$.*

The algorithms $(\mathsf{PCG.Gen}, \mathsf{PCG.Expand})$ *should satisfy the following:*

- **Correctness.** *The correlation obtained via:*

  $$\{(R_0, R_1) \,|(\mathsf{k}_0, \mathsf{k}_1) \leftarrow \mathsf{PCG.Gen}(1^\lambda), R_\sigma \leftarrow \mathsf{PCG.Expand}(\sigma, \mathsf{k}_\sigma) \text{ for } \sigma \in \{0,1\}\}$$

  is computationally indistinguishable from $\mathcal{C}(1^\lambda)$.

- **Security.** *For any $\sigma \in \{0,1\}$, the following two distributions are computationally indistinguishable:*

  $$\{(\mathsf{k}_\sigma, R_{1-\sigma}) \mid (\mathsf{k}_0, \mathsf{k}_1) \leftarrow \mathsf{PCG.Gen}(1^\lambda), R_{1-\sigma} \leftarrow \mathsf{PCG.Expand}(1-\sigma, \mathsf{k}_{1-\sigma})\} \text{ and}$$
  $$\{(\mathsf{k}_\sigma, R_{1-\sigma}) \mid (\mathsf{k}_0, \mathsf{k}_1) \leftarrow \mathsf{PCG.Gen}(1^\lambda), R_\sigma \leftarrow \mathsf{PCG.Expand}(\sigma, \mathsf{k}_\sigma), R_{1-\sigma} \leftarrow \mathsf{RSample}(1^\lambda, \sigma, R_\sigma)\}$$

  where RSample *is the reverse sampling algorithm for correlation $\mathcal{C}$.*

$\underline{\mathsf{Exp}^{\mathsf{pr}}_{\mathcal{A},N,0}(\lambda)}$:

$\mathsf{mk} \leftarrow \mathsf{Setup}(1^\lambda)$

for $i = 1$ **to** $N(\lambda)$**:**

$x^{(i)} \xleftarrow{\$} \{0,1\}^{n(\lambda)}$

$(y_0^{(i)}, y_1^{(i)}) \leftarrow \mathcal{Y}(1^\lambda, \mathsf{mk})$

$b \leftarrow \mathcal{A}(1^\lambda, (x^{(i)}, y_0^{(i)}, y_1^{(i)})_{i\in[N(\lambda)]})$

return b

$\underline{\mathsf{Exp}^{\mathsf{pr}}_{\mathcal{A},N,1}(\lambda)}$:

$(\mathsf{k}_0, \mathsf{k}_1) \leftarrow \mathsf{PCF.Gen}(1^\lambda)$

for $i = 1$ **to** $N(\lambda)$**:**

$x^{(i)} \xleftarrow{\$} \{0,1\}^{n(\lambda)}$

for $\sigma \in \{0,1\}$**:** $y_\sigma^{(i)} \leftarrow \mathsf{PCF.Eval}(\sigma, \mathsf{k}_\sigma, x^{(i)})$

$b \leftarrow \mathcal{A}(1^\lambda, (x^{(i)}, y_0^{(i)}, y_1^{(i)})_{i\in[N(\lambda)]})$

return b

Figure 16: Pseudorandom $\mathcal{Y}$-correlated outputs of a PCF.

B Full PCF Definition

In this section we give the full definition of pseudorandom corelation functions for a correlation with setup. Further, we define a non-adaptive PCF, where the adversary is only allowed non-adaptive queries. This definition turns out to be useful, as it is easier to instantiate than a PCF with strong security, but still sufficient for most applications.

Definition B.1 (Reverse-sampleable correlation with setup) *Let $\ell_0(\lambda), \ell_1(\lambda) \leq \mathsf{poly}(\lambda)$ be output length functions. Let $(\mathsf{Setup}, \mathcal{Y})$ be a tuple of probabilistic algorithms, such that Setup on input 1^λ returns a master key mk and $\mathcal{Y}$ on input 1^λ and mk returns a pair of outputs $(y_0, y_1) \in \{0,1\}^{\ell_0(\lambda)} \times \{0,1\}^{\ell_1(\lambda)}$.*

We say that the tuple $(\mathsf{Setup}, \mathcal{Y})$ defines a reverse sampleable correlation with setup*, if there exists a probabilistic polynomial time algorithm $\mathsf{RSample}$ that takes as input 1^λ, mk, $\sigma \in \{0,1\}$ and $y_\sigma \in \{0,1\}^{\ell_\sigma(\lambda)}$, and outputs $y_{1-\sigma} \in \{0,1\}^{\ell_{1-\sigma}(\lambda)}$, such that for all $\mathsf{mk}, \mathsf{mk}'$ in the image of Setup and all $\sigma \in \{0,1\}$ the following distributions are statistically close:*

$$\{(y_0, y_1) \mid (y_0, y_1) \xleftarrow{\$} \mathcal{Y}(1^\lambda, \mathsf{mk})\}$$
$$\{(y_0, y_1) \mid (y_0', y_1') \xleftarrow{\$} \mathcal{Y}(1^\lambda, \mathsf{mk}'), y_\sigma \leftarrow y_\sigma', y_{1-\sigma} \leftarrow \mathsf{RSample}(1^\lambda, \mathsf{mk}, \sigma, y_\sigma)\}$$

Definition B.2 (Pseudorandom correlation function (PCF)) *Let $(\mathsf{Setup}, \mathcal{Y})$ fix a reverse-sampleable correlation with setup which has output length functions $\ell_0(\lambda), \ell_1(\lambda)$, and let $\lambda \leq n(\lambda) \leq \mathsf{poly}(\lambda)$ be an input length function. Let $(\mathsf{PCF.Gen}, \mathsf{PCF.Eval})$ be a pair of algorithms with the following syntax:*

- $\mathsf{PCF.Gen}(1^\lambda)$ *is a probabilistic polynomial time algorithm that on input 1^λ, outputs a pair of keys $(\mathsf{k}_0, \mathsf{k}_1)$;*
- $\mathsf{PCF.Eval}(\sigma, \mathsf{k}_\sigma, x)$ *is a deterministic polynomial-time algorithm that on input $\sigma \in \{0,1\}$, key k_σ and input value $x \in \{0,1\}^{n(\lambda)}$, outputs a value $y_\sigma \in \{0,1\}^{\ell_\sigma(\lambda)}$.*

*We say $(\mathsf{PCF.Gen}, \mathsf{PCF.Eval})$ is a (weak) (N, B, ε)-*secure pseudorandom correlation function (PCF) for $\mathcal{Y}$*, if the following conditions hold:*

- ***Pseudorandom $\mathcal{Y}$-correlated outputs.*** *For every $\sigma \in \{0,1\}$ and non-uniform adversary $\mathcal{A}$ of size $B(\lambda)$, it holds*

$$\left|\Pr[\mathsf{Exp}^{\mathsf{pr}}_{\mathcal{A},N,0}(\lambda) = 1] - \Pr[\mathsf{Exp}^{\mathsf{pr}}_{\mathcal{A},N,1}(\lambda) = 1]\right| \leq \varepsilon(\lambda)$$

for all sufficiently large λ, where $\mathsf{Exp}^{\mathsf{pr}}_{\mathcal{A},N,b}(\lambda)$ for $b \in \{0,1\}$ is as defined in Figure 16. (In particular, where the adversary is given access to $N(\lambda)$ samples.)

$\underline{\mathsf{Exp}^{\mathsf{sec}}_{\mathcal{A},N,\sigma,0}(\lambda)}$:
$(\mathsf{k}_0, \mathsf{k}_1) \leftarrow \mathsf{PCF.Gen}(1^\lambda)$
for $i = 1$ **to** $N(\lambda)$**:**
  $x^{(i)} \xleftarrow{\$} \{0,1\}^{n(\lambda)}$
  $y^{(i)}_{1-\sigma} \leftarrow \mathsf{PCF.Eval}(1-\sigma, \mathsf{k}_{1-\sigma}, x^{(i)})$
$b \leftarrow \mathcal{A}(1^\lambda, \sigma, k_\sigma, (x^{(i)}, y^{(i)}_{1-\sigma})_{i\in[N(\lambda)]})$
return b

$\underline{\mathsf{Exp}^{\mathsf{sec}}_{\mathcal{A},N,\sigma,1}(\lambda)}$:
$(\mathsf{k}_0, \mathsf{k}_1) \leftarrow \mathsf{PCF.Gen}(1^\lambda)$
$\mathsf{mk} \xleftarrow{\$} \mathsf{Setup}(1^\lambda)$
for $i = 1$ **to** $N(\lambda)$**:**
  $x^{(i)} \xleftarrow{\$} \{0,1\}^{n(\lambda)}$
  $y^{(i)}_{\sigma} \leftarrow \mathsf{PCF.Eval}(\sigma, \mathsf{k}_\sigma, x^{(i)})$
  $y^{(i)}_{1-\sigma} \leftarrow \mathsf{RSample}(1^\lambda, \mathsf{mk}, \sigma, y^{(i)}_\sigma)$
$b \leftarrow \mathcal{A}(1^\lambda, \sigma, k_\sigma, (x^{(i)}, y^{(i)}_{1-\sigma})_{i\in[N(\lambda)]})$
return b

Figure 17: Security of a PCF. Here, RSample is the algorithm for reverse sampling $\mathcal{Y}$ as in Definition D.1.

$\underline{\mathsf{Exp}^{\mathsf{s\text{-}pr}}_{\mathcal{A}}(\lambda)}$:
$\mathsf{mk} \leftarrow \mathsf{Setup}(1^\lambda)$
$(\mathsf{k}_0, \mathsf{k}_1) \xleftarrow{\$} \mathsf{PCF.Gen}(1^\lambda)$
$\mathcal{Q} = \emptyset$
$b \xleftarrow{\$} \{0,1\}$
$b^\star \leftarrow \mathcal{A}^{\mathcal{O}_b(\cdot)}(1^\lambda)$
if $b = b^\star$ **return** 1
else return 0

$\underline{\mathcal{O}_0(x)}$:
if $(x, y_0, y_1) \in \mathcal{Q}$**:**
  return (y_0, y_1)
else:
  $(y_0, y_1) \leftarrow \mathcal{Y}(1^\lambda, \mathsf{mk})$
  $\mathcal{Q} = \mathcal{Q} \cup \{(x, y_0, y_1)\}$
  return (y_0, y_1)

$\underline{\mathcal{O}_1(x)}$:
for $\sigma \in \{0,1\}$:
  $y_\sigma \leftarrow \mathsf{PCF.Eval}(1^\lambda, \sigma, \mathsf{k}_\sigma, x)$
return (y_0, y_1)

Figure 18: Strong pseudorandom $\mathcal{Y}$-correlated outputs of a PCF.

- ***Security.*** *For each $\sigma \in \{0,1\}$ and non-uniform adversary $\mathcal{A}$ of size $B(\lambda)$, it holds*

$$\left|\Pr[\mathsf{Exp}^{\mathsf{sec}}_{\mathcal{A},N,\sigma,0}(\lambda) = 1] - \Pr[\mathsf{Exp}^{\mathsf{sec}}_{\mathcal{A},N,\sigma,1}(\lambda) = 1]\right| \leq \varepsilon(\lambda)$$

  for all sufficiently large λ, where $\mathsf{Exp}^{\mathsf{sec}}_{\mathcal{A},N,\sigma,b}(\lambda)$ for $b \in \{0,1\}$ is as defined in Figure 17 (again, with $N(\lambda)$ samples).

Definition B.3 (Strong pseudorandom correlation function (sPCF)) *Let $\mathcal{Y}$ and* (PCF.Gen, PCF.Eval) *be as in Definition B.2. We say that* (PCF.Gen, PCF.Eval) *is a* strong (N, B, ε)-secure PCF (sPCF) for $\mathcal{Y}$ *if the following conditions hold:*

- ***Strong pseudorandom $\mathcal{Y}$-correlated outputs.*** *For every non-uniform adversary $\mathcal{A}$ of size $B(\lambda)$ asking at most $N(\lambda)$ queries to the oracle $\mathcal{O}_b(\cdot)$, it holds*

$$\left|\Pr[\mathsf{Exp}^{\mathsf{s\text{-}pr}}_{\mathcal{A}}(\lambda) = 1] - \frac{1}{2}\right| \leq \varepsilon(\lambda)$$

  for all sufficiently large λ, where $\mathsf{Exp}^{\mathsf{s\text{-}pr}}_{\mathcal{A}}(\lambda)$ is as defined in Figure 18.

- ***Strong security.*** *For all $\sigma \in \{0,1\}$ and non-uniform adversaries $\mathcal{A}$ of size $B(\lambda)$ asking at most $N(\lambda)$ queries to the oracle $\mathcal{O}_b(\cdot)$, it holds*

$$\left|\Pr[\mathsf{Exp}^{\mathsf{s\text{-}sec}}_{\mathcal{A},\sigma}(\lambda) = 1] - \frac{1}{2}\right| \leq \varepsilon(\lambda)$$

  for all sufficiently large λ, where $\mathsf{Exp}^{\mathsf{s\text{-}sec}}_{\mathcal{A},\sigma}(\lambda)$ for $b \in \{0,1\}$ is as defined in Figure 19.

Definition B.4 (Non-adaptive pseudorandom correlation function (naPCF)) *Let $\mathcal{Y}$ and* (PCF.Gen, PCF.Eval) *be as in Definition B.2. We say that* (PCF.Gen, PCF.Eval) *is a* non-adaptive (N, B, ε)-secure PCF for $\mathcal{Y}$ *if the following conditions hold:*

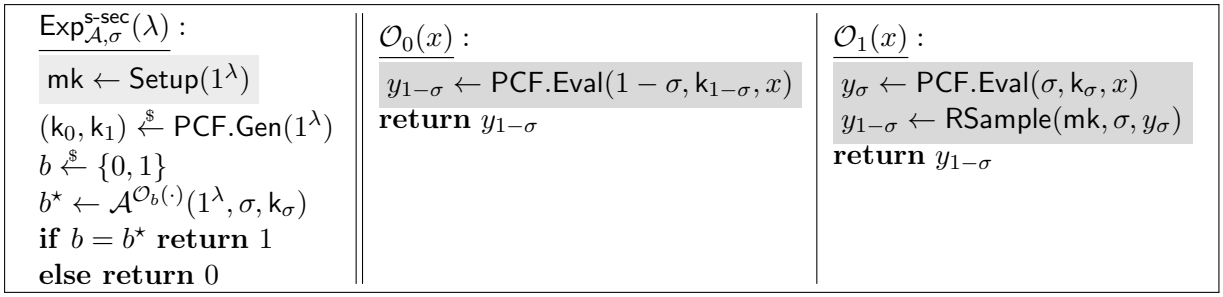

Figure 19: Strong security of a PCF. Here, RSample is the algorithm for reverse sampling $\mathcal{Y}$ according to Definition D.1.

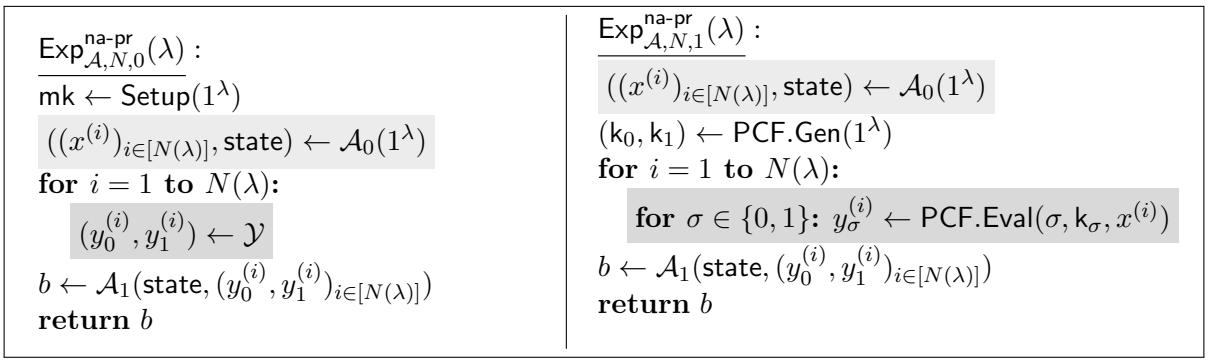

Figure 20: Pseudorandom $\mathcal{Y}$-correlated outputs of a non-adaptive PCF.

- ***Pseudorandom correlated outputs with non-adaptive queries.*** *For $\sigma \in \{0,1\}$ and non-uniform adversary $\mathcal{A} = (\mathcal{A}_0, \mathcal{A}_1)$ of size $B(\lambda)$, it holds*

$$\left|\Pr[\mathsf{Exp}^{\mathsf{na\text{-}pr}}_{\mathcal{A}=(\mathcal{A}_0,\mathcal{A}_1),N,0}(\lambda) = 1] - \Pr[\mathsf{Exp}^{\mathsf{na\text{-}pr}}_{\mathcal{A}=(\mathcal{A}_0,\mathcal{A}_1),N,1}(\lambda) = 1]\right| \leq \varepsilon(\lambda)$$

for all sufficiently large λ, where $\mathsf{Exp}^{\mathsf{s\text{-}pr}}_{\mathcal{A},N,b}(\lambda)$ for $b \in \{0,1\}$ is as defined in Figure 20.

- ***Non-adaptive security.*** *For $\sigma \in \{0,1\}$ and non-uniform adversary $\mathcal{A} = (\mathcal{A}_0, \mathcal{A}_1)$ of size $B(\lambda)$, it holds*

$$\left|\Pr[\mathsf{Exp}^{\mathsf{na\text{-}sec}}_{\mathcal{A}=(\mathcal{A}_0,\mathcal{A}_1),N,\sigma,0}(\lambda) = 1] - \Pr[\mathsf{Exp}^{\mathsf{na\text{-}sec}}_{\mathcal{A}=(\mathcal{A}_0,\mathcal{A}_1),N,\sigma,1}(\lambda) = 1]\right| \leq \varepsilon(\lambda)$$

for all sufficiently large λ, where $\mathsf{Exp}^{\mathsf{na\text{-}sec}}_{\mathcal{A},N,\sigma,b}(\lambda)$ for $b \in \{0,1\}$ is as defined in Figure 21.

C Programmable PCF for VOLE from WPRF and FSS

Lemma C.1 *The PCF construction for VOLE from Figure 22 supports reusable inputs.*

Proof. [Sketch] Indistinguishability is immediate. Programmibility follows with $f_0(\rho_0, x) = \rho_0$ and $f_1(\rho_1, x) = F_{\rho_1}(x)$. Finally, security follows from the security of FSS, similar to the proof of Theorem 5.3.

□

D Multi-Party PCFs

Definition D.1 (Reverse-sampleable multi-user correlation with setup) *Let $\ell_1(\lambda), \ldots, \ell_M(\lambda) \leq \mathsf{poly}(\lambda)$ be output length functions. Let $(\mathsf{Setup}, \mathcal{Y})$ be a tuple of probabilistic algorithms,*

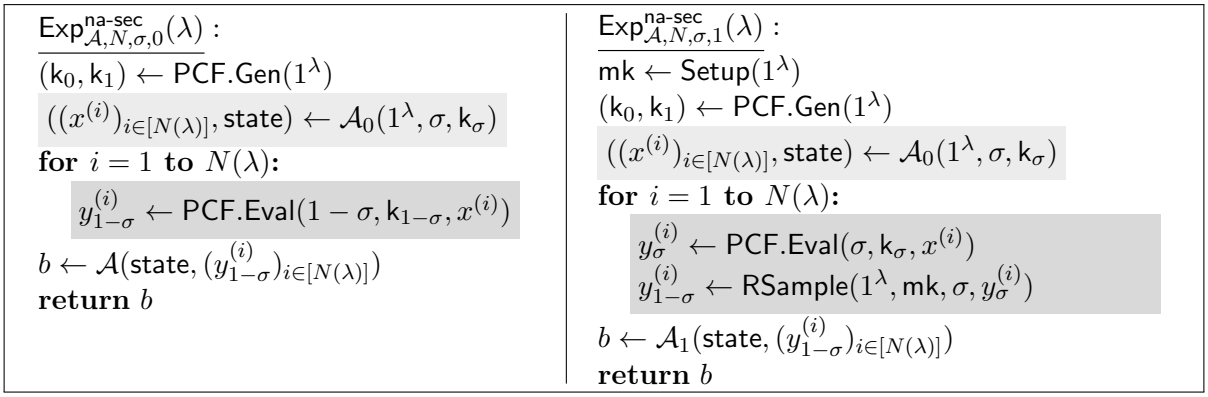

Figure 21: Security of a non-adaptive PCF. Here, RSample is the algorithm for reverse sampling $\mathcal{Y}$ as in Definition D.1.

such that Setup *on input* 1^λ *returns a master key* mk *and* $\mathcal{Y}$ *on input* 1^λ *and* mk *returns outputs* $(y_1,\ldots,y_M) \in \{0,1\}^{\ell_1(\lambda)} \times \cdots \times \{0,1\}^{\ell_M(\lambda)}$.

We say that the tuple $(\mathsf{Setup},\mathcal{Y})$ *defines a* reverse sampleable M-user correlation with setup, *if there exists a probabilistic polynomial time algorithm* RSample *that takes as input* 1^λ, mk, $T \subset \{1,\ldots,M\}$ *and* $(y_i)_{i\in T}$, *and outputs* $(y_j)_{j\notin T}$, *such that for all* mk, mk′ *in the image of* Setup *and all* $\sigma \in \{0,1\}$ *the following distributions are statistically close:*

$$\{(y_1,\ldots,y_M) \,|(y_1,\ldots,y_M) \xleftarrow{\$} \mathcal{Y}(1^\lambda,\mathsf{mk})\}$$
$$\{(y_1,\ldots,y_M) \,|(y'_1,\ldots,y'_M) \xleftarrow{\$} \mathcal{Y}(1^\lambda,\mathsf{mk}), y_j \leftarrow y'_j \text{ for all } j\in T,$$
$$(y_j)_{j\notin T} \leftarrow \mathsf{RSample}(1^\lambda,\mathsf{mk},T,(y_j)_{j\in T})\}$$

Definition D.2 (M-Party pseudorandom correlation function (PCF)) *Let* $(\mathsf{Setup},\mathcal{Y})$ *fix a reverse-sampleable* M*-party correlation with setup which has output length functions* $\ell_1(\lambda),\ldots,\ell_M(\lambda)$, *and let* $\lambda \le n(\lambda) \le \mathsf{poly}(\lambda)$ *be an input length function. Let* $(\mathsf{PCF.Gen},\mathsf{PCF.Eval})$ *be a pair of algorithms with the following syntax:*

- $\mathsf{PCF.Gen}(1^\lambda)$ *is a probabilistic polynomial time algorithm that on input* 1^λ, *outputs a pair of keys* $(\mathsf{k}_1,\ldots,\mathsf{k}_M)$;
- $\mathsf{PCF.Eval}(i,\mathsf{k}_i,x)$ *is a deterministic polynomial-time algorithm that on input* $i\in[M]$, *key* k_i *and input value* $x\in\{0,1\}^{n(\lambda)}$, *outputs a value* $y_i \in \{0,1\}^{\ell_i(\lambda)}$.

We say $(\mathsf{PCF.Gen},\mathsf{PCF.Eval})$ *is a (weak)* (N,B,ε)-secure M-party pseudorandom correlation function (PCF) for $\mathcal{Y}$, *if the following conditions hold:*

- ***Pseudorandom* $\mathcal{Y}$*-correlated outputs.*** *For every* $\sigma\in\{0,1\}$ *and non-uniform adversary* $\mathcal{A}$ *of size* $B(\lambda)$, *it holds*
$$\left|\Pr[\mathsf{Exp}^{\mathsf{M\text{-}pr}}_{\mathcal{A},N,0}(\lambda)=1] - \Pr[\mathsf{Exp}^{\mathsf{M\text{-}pr}}_{\mathcal{A},N,1}(\lambda)=1]\right| \le \varepsilon(\lambda)$$
for all sufficiently large λ, *where* $\mathsf{Exp}^{\mathsf{M\text{-}pr}}_{\mathcal{A},N,b}(\lambda)$ *for* $b\in\{0,1\}$ *is as defined in Figure 23. (In particular, where the adversary is given access to* $N(\lambda)$ *samples.)*
- ***Security.*** *For each* $\sigma\in\{0,1\}, T\subset[M]$ *and non-uniform adversary* $\mathcal{A}$ *of size* $B(\lambda)$, *it holds*
$$\left|\Pr[\mathsf{Exp}^{\mathsf{M\text{-}sec}}_{\mathcal{A},N,\sigma,0,T}(\lambda)=1] - \Pr[\mathsf{Exp}^{\mathsf{M\text{-}sec}}_{\mathcal{A},N,\sigma,1,T}(\lambda)=1]\right| \le \varepsilon(\lambda)$$
for all sufficiently large λ, *where* $\mathsf{Exp}^{\mathsf{M\text{-}sec}}_{\mathcal{A},N,\sigma,b,T}(\lambda)$ *for* $b\in\{0,1\}$ *is as defined in Figure 24 (again, with* $N(\lambda)$ *samples).*

Programmable PCF for VOLE

Let $\mathcal{F} = \{F_k : \{0,1\}^n \to R\}_{k\in\{0,1\}^\lambda}$ be a weak PRF, and $\mathsf{FSS} = (\mathsf{FSS.Gen}, \mathsf{FSS.Eval})$ an FSS scheme for $\{cF_k\}_{c\in R, k\in\{0,1\}^\lambda}$ with weak pseudorandom outputs. Let further $\rho_0 \xleftarrow{\$} \{0,1\}^\lambda$, $\rho_1 \xleftarrow{\$} R$.

$\mathsf{PCF.Gen}(1^\lambda; \rho_0, \rho_1)$**:**

1. Set the weak-PRF key $k \leftarrow \rho_0$ and $a \leftarrow \rho_1$.
2. Sample a pair of FSS keys $K_0^{\mathsf{fss}}, K_1^{\mathsf{fss}} \leftarrow \mathsf{FSS.Gen}(1^\lambda, aF_k)$.
3. Output the keys $\mathsf{k}_0 = (K_0^{\mathsf{fss}}, a)$ and $\mathsf{k}_1 = (K_1^{\mathsf{fss}}, k)$.

$\mathsf{PCF.Eval}(\sigma, \mathsf{k}_\sigma, x)$**:** On input a random x:

- If $\sigma = 0$:
  1. Let $c_0 = -\mathsf{FSS.Eval}(0, K_0^{\mathsf{fss}}, x)$.
  2. Output (a, c_0).
- If $\sigma = 1$:
  1. Let $c_1 = \mathsf{FSS.Eval}(1, K_1^{\mathsf{fss}}, x)$.
  2. Let $b = F_k(x)$.
  3. Output (b, c_1).

Figure 22: Programmable PCF for VOLE over the ring R based on FSS for scalar multiples of a weak PRF.

$\underline{\mathsf{Exp}^{\mathsf{M\text{-}pr}}_{\mathcal{A},N,0}(\lambda)}$:
$\mathsf{mk} \leftarrow \mathsf{Setup}(1^\lambda)$
for $i = 1$ **to** $N(\lambda)$**:**
  $x^{(i)} \xleftarrow{\$} \{0,1\}^{n(\lambda)}$
  $(y_1^{(i)}, \ldots, y_M^{(i)}) \leftarrow \mathcal{Y}(1^\lambda, \mathsf{mk})$
$b \leftarrow \mathcal{A}(1^\lambda, (x^{(i)}, y_1^{(i)}, \ldots, y_M^{(i)})_{i\in[N(\lambda)]})$
return b

$\underline{\mathsf{Exp}^{\mathsf{M\text{-}pr}}_{\mathcal{A},N,1}(\lambda)}$:
$(\mathsf{k}_1, \ldots, \mathsf{k}_M) \leftarrow \mathsf{PCF.Gen}(1^\lambda)$
for $i = 1$ **to** $N(\lambda)$**:**
  $x^{(i)} \xleftarrow{\$} \{0,1\}^{n(\lambda)}$
  $\forall j \in [M]$**:** $y_j^{(i)} \leftarrow \mathsf{PCF.Eval}(j, \mathsf{k}_j, x^{(i)})$
$b \leftarrow \mathcal{A}(1^\lambda, (x^{(i)}, y_1^{(i)}, \ldots, y_M^{(i)})_{i\in[N(\lambda)]})$
return b

Figure 23: Pseudorandom $\mathcal{Y}$-correlated outputs of a PCF.

$\underline{\mathsf{Exp}^{\mathsf{M\text{-}sec}}_{\mathcal{A},N,\sigma,0,T}(\lambda)}$:
$(\mathsf{k}_1, \ldots, \mathsf{k}_M) \leftarrow \mathsf{PCF.Gen}(1^\lambda)$
for $i = 1$ **to** $N(\lambda)$**:**
  $x^{(i)} \xleftarrow{\$} \{0,1\}^{n(\lambda)}$
  $\forall j \notin T : y_j^{(i)} \leftarrow \mathsf{PCF.Eval}(j, \mathsf{k}_j, x^{(i)})$
$b \leftarrow \mathcal{A}(1^\lambda, T, (k_j)_{j\in T}, (x^{(i)}, (y_j^{(i)})_{j\notin T})_{i\in[N]})$
return b

$\underline{\mathsf{Exp}^{\mathsf{M\text{-}sec}}_{\mathcal{A},N,\sigma,1,T}(\lambda)}$:
$(\mathsf{k}_1, \ldots, \mathsf{k}_M) \leftarrow \mathsf{PCF.Gen}(1^\lambda)$
$\mathsf{mk} \xleftarrow{\$} \mathsf{Setup}(1^\lambda)$
for $i = 1$ **to** $N(\lambda)$**:**
  $x^{(i)} \xleftarrow{\$} \{0,1\}^{n(\lambda)}$
  $\forall j \in T$**:** $y_j^{(i)} \leftarrow \mathsf{PCF.Eval}(j, \mathsf{k}_j, x^{(i)})$
  $(y_j^{(i)})_{j\notin T} \leftarrow \mathsf{RSample}(\mathsf{mk}, T, (y_j^{(i)})_{j\in T})$
$b \leftarrow \mathcal{A}(1^\lambda, T, (k_j)_{j\in T}, (x^{(i)}, (y_j^{(i)})_{j\notin T})_{i\in[N(\lambda)]})$
return b

Figure 24: Security of a PCF. Here, $\mathsf{RSample}$ is the algorithm for reverse sampling $\mathcal{Y}$ as in Definition D.1.