Pseudorandom Correlation Functions from Variable-Density LPN, Revisited

Geoffroy Couteau, Clément Ducros

▶ **To cite this version:**

Geoffroy Couteau, Clément Ducros. Pseudorandom Correlation Functions from Variable-Density LPN, Revisited. IACR 2023 - 26th International Conference on Practice and Theory of Public-Key Cryptography, May 2023, Atlanta, United States. pp.221-250, 10.1007/978-3-031-31371-4_8 . hal-03947831

HAL Id: hal-03947831
https://hal.science/hal-03947831v1

Submitted on 31 Oct 2023

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Distributed under a Creative Commons Attribution 4.0 International License

Pseudorandom Correlation Functions from Variable-Density LPN, Revisited

Geoffroy Couteau[1] ,
Clément Ducros[2]

[1] CNRS, IRIF, Université de Paris
couteau@irif.fr
[2] IRIF, Université de Paris, INRIA
cducros@irif.fr

Abstract. Pseudorandom correlation functions (PCF), introduced in the work of (Boyle *et al.*, FOCS 2020), allow two parties to locally generate, from short correlated keys, a near-unbounded amount of pseudorandom samples from a target correlation. PCF is an extremely appealing primitive in secure computation, where they allow to confine all preprocessing phases of all future computations two parties could want to execute to a single short interaction with low communication and computation, followed solely by offline computations. Beyond introducing the notion, Boyle *et al.* gave a candidate construction, using a new *variable-density* variant of the learning parity with noise (LPN) assumption. Then, to provide support for this new assumption, the authors showed that it provably resists a large class of linear attacks, which captures in particular all known attacks on LPN.

In this work, we revisit the analysis of the VDLPN assumption. We make two key contributions:

- First, we observe that the analysis of Boyle *et al* is purely asymptotic: they do not lead to any concrete and efficient PCF instantiation within the bounds that offer security guarantees. To improve this state of affairs, we combine a new variant of a VDLPN assumption with an entirely new, much tighter security analysis, which we further tighten using extensive computer simulations to optimize parameters. This way, we manage to obtain for the first time a set of *provable* usable parameters (under a simple combinatorial conjecture which is easy to verify experimentally), leading to a concretely efficient PCF resisting all linear tests.
- Second, we identify a flaw in the security analysis of Boyle *et al.*, which invalidates their proof that VDLPN resists linear attacks. Using several new non-trivial arguments, we repair the proof and fully demonstrate that VDLPN resists linear attack; our new analysis is more involved than the original (flawed) analysis.

Our parameters set leads to PCFs with keys around 3MB allowing $\sim$ 500 evaluations per second on one core of a standard laptop for 110 bits of security; these numbers can be improved to 350kB keys and $\sim$ 3950 evaluations/s using a more aggressive all-prefix variant. All numbers are quite tight: only within a factor 3 of the best bounds one could heuristically hope for.

1 Introduction

The generation of secret correlated random string is the cornerstone of secure computation (MPC). Given access to a trusted source of correlated randomness, any n-party functionality can be securely computed with information-theoretic security (against $n-1$ corrupted parties), and with very high concrete efficiency. For example, given $2m$ random oblivious transfers (in a random oblivious transfer, Alice gets two random bits (s_0, s_1), and Bob gets (b, s_b) for a random bit b), two parties can securely compute any boolean circuit C with up to m AND gates, with perfect security, while exchanging only four bits per AND gate.

The simplicity and efficiency of this paradigm is well known, and most modern MPC protocols take advantage of its features by sharing the same high level two-step structure: in the first step, the *preprocessing phase*, the parties interact to distributively and securely generate these correlated randomness. Since this phase is input-independent, it can be carried out ahead of time. Then, in the second step, the *online phase*, the parties "consume" this correlated randomness in a fast, information-theoretic protocol. The core challenge is this approach lies in step 1: designing a secure protocol to distributively generate long correlated random string.

Pseudorandom correlations. Until recently, all state of the art protocols, such as SPDZ [DPSZ12], required $\Omega(s)$ communication to generate s bits of correlated randomness (ignoring terms depending on the security parameter and the number of parties), leading to communication-intensive preprocessing phases. This state of affair changed in a recent and exciting line of work [BCG$^+$17, BCGI18, BCG$^+$19b, BCG$^+$19a, SGRR19, BCG$^+$20b, CRR21] which introduced the notion of *pseudorandom correlation generators* (PCG), a new cryptographic primitive which allows parties to locally generate, from short correlated seeds, long instances of correlated *pseudorandom* strings. These PCGs enable secure computation with *silent preprocessing* where, after a short interaction to generate the short correlated seeds, the parties never need to interact anymore, and locally generate the long correlated strings. The latest results in this area further demonstrated that this primitive could be achieved with very high concrete efficiency, under appropriate LPN-like cryptographic assumptions.

Pseudorandom correlated functions. The aforementioned constructions of PCG, however, share a common limitation: the expansion of the short keys into long pseudorandom correlated strings is a one-time, monolithic procedure. That is, these PCGs are limited to a single generation of an a priori bounded amount of correlated pseudorandomness. If the parties want to possibly use these correlations across many protocols, then they carry the burden of having to either re-do the distributed generation of the short keys each time, or storing a very large amount of correlated randomness for a possibly long duration.

These limitations were overcome in a recent work [BCG$^+$20a], where the authors introduced the notion of pseudorandom correlated *functions* (PCFs). PCFs

are to PCGs what pseudorandom functions are to pseudorandom generators: they allow to generate an arbitrary amount of correlated (pseudo)randomness in an incremental fashion. That is, given two short correlated keys (K_0, K_1), two parties can locally compute an arbitrary number of correlated strings $F_{K_0}(x)$, $F_{K_1}(x)$), which are all indistinguishable from independent random samples from the target correlation. PCFs allow to confine *all future preprocessing phases* of any future MPC protocols that two parties may wish to run to a *one-time* short interaction, followed solely by local computation to generate the preprocessing material in all subsequent computations.

1.1 Constructions of Pseudorandom Correlated Functions

A PCF is an extremely powerful primitive, but also one which is highly non-trivial to construct. A generic construction of PCF under the LWE assumption can be obtained by letting the two parties homomorphically evaluate a well-chosen circuit using a threshold fully-homomorphic encryption scheme [DHRW16, BCG$^+$20a]: the circuit takes as input a PRF key K, and computes pseudorandom instances of the correlation, using the output of the PRF to generate the pseudorandomness used in these correlations. However, this approach falls short of providing a concretely usable solution. To our knowledge, there are currently two competing approaches to construct usable PCFs:

PCFs from variable-density LPN. The work of [BCG$^+$20a] gave a generic construction of PCF, by combining two primitives:

- A function secret sharing scheme (FSS) for a class of circuits $\mathcal{C}$. At a high level, an FSS for $\mathcal{C}$ allows to share any function $f \in \mathcal{C}$ in two functions f_0, f_1 such that each f_i computationally hides f, yet for any input x, it holds that $f_0(x) + f_1(x) = f(x)$.
- One weak pseudorandom function (WPRF) for some class $\mathcal{C}'$ related to $\mathcal{C}$. A WPRF is a PRF where the adversary in the pseudorandomness game is restricted to only querying random inputs.

Previous works [GI14,BGI15,BGI16] have shown how to construct extremely efficient FSS schemes for simple complexity classes, such as multi-point functions (i.e., a function $f_{\alpha,\beta}$ equal to 0 everywhere, except on n specific points $\alpha = (\alpha_1, \cdots, \alpha_n)$, where it takes a fixed value β), from minimal assumptions (namely, the existence of one-way functions). The shares of an n-point function $f_{\alpha,\beta}$ over a domain of size N consist of $n \log N$ PRG seeds, and evaluating f_i on the entire domain requires only N PRG evaluations. Given this, the authors of [BCG$^+$20a] put forward a new WPRF in the (particularly low) complexity class of multi-point functions, which essentially boils down to a WPRF of the form $F_K(x) = F(x \oplus K)$, where F is a depth-two circuit with one bottom layer of high fan-in ANDs, and a single top high fan-in XOR gate. The security of this new candidate relies on the hardness of a new variant of the learning parity with noise (LPN) assumption, called *variable density* LPN assumption; we will

overview this assumption later on. Given this new WPRF and the efficient FSS scheme of [BGI16], the authors of [BCG+20a] obtain a PCF candidate which can handle a wide variety of low-degree correlations, including (but not limited to) oblivious transfer correlations. The authors provide several variants and parameter choices; their most aggressive choices of parameters lead to a reasonably efficient construction, which (based on rough estimations) could generate hundreds to thousands of pseudorandom OT correlations per second on one core of a standard computer.

PCFs from decisional composite residuosity. An alternative approach to building PCFs was recently put forward in [OSY21], using a Paillier-based construction of homomorphic secret sharing. In contrast to [BCG+20a], this work does not need to rely on new assumptions, and instead only requires the well-established decision composite residuosity assumption. However, this alternative construction has several downsides:

- *Expressivity.* The construction of [OSY21] is inherently limited to oblivious transfer correlations. In contrast, the VDLPN-based construction can generate arbitrary low-degree polynomial correlations, such as OLE, (authenticated) Beaver triples, and many more; these alternative correlations are crucial in many secure computation protocols.
- *Post-quantumness.* The DCR assumption can be broken by Shor's algorithm. In contrast, while VDLPN is a new and little studied assumption, there seems to be no reason to believe that it should be quantumly broken, being a relatively natural LPN-style assumption.
- *Efficiency.* Eventually, the construction of [OSY21] requires a few hundred exponentiations in an RSA group for every OT correlation produced. Using standard benchmark for exponentiations in 2048-bit RSA groups on a modern laptop[3], this translates to a cost of the order of one second *for each OT produced*, which is several orders of magnitude less efficient than what the VDLPN-based approach can plausibly provide, for suitable choices of parameters.

Given the above, the VDLPN approach seems to provide the best alternative to obtain efficient and expressive PCFs; however, its reliance on a new assumption calls for a very careful examination of its security. The work of [BCG+20a] provided an initial security analysis, proving a number of important results regarding the resistance of VDLPN against standard attacks. However, this analysis is purely asymptotic, and does not say much about what concrete choices of parameters can be expected to provide a sufficient security level. In addition, a close inspection of their analysis uncovers an important gap in one of the claim, invalidating part of the analysis (we will expand on this later on). Before we detail our contribution, we provide more context on the underlying new assumption and its analysis.

[3] E.g. A laptop equipped with an Intel i5 2540M processor can compute an RSA decryption in 1.4ms of amortized time

1.2 The Variable-Density LPN Assumption

At a high level, the standard LPN assumption with dimension k and number of samples $n > k$ states the following: given a uniformly random matrix $A \xleftarrow{\$} \mathbb{F}_2^{n \times k}$, sample a vector $\mathbf{b}$ as $\mathbf{b} = A \cdot \mathbf{s} + \mathbf{e}$, where $\mathbf{s}$ is a random vector from $\mathbb{F}_2^k$, and $\mathbf{e}$ is a random *sparse* vector (the noise vector) over $\mathbb{F}_2^n$ (the exact distribution of $\mathbf{e}$ depends on the LPN flavor: it follows a Bernoulli distribution for standard LPN, it is uniform over all vectors of a given weight for exact LPN (XLPN), and it is a concatenation of unit vectors for regular LPN). Then, LPN states that it is hard to distinguish $\mathbf{b}$ from a uniformly random vector (put otherwise, it is hard to solve noisy systems of linear equations).

In coding theoretic terms, LPN therefore states that a noisy codeword from a random linear code looks random. LPN admits an equivalent, dual formulation: viewing A as the generating matrix of a linear code of dimension k, let $H \in F_2^{(n-k) \times n}$ be a *parity-check matrix* of A (which satisfies $H \cdot A = 0$; that is, $H^\intercal$ generates the dual of the code generated by A). Then distinguish $\mathbf{b} = A \cdot \mathbf{s} + \mathbf{e}$ from random is equivalent to distinguishing $H \cdot \mathbf{b} = H \cdot \mathbf{e}$ from random – that is, finding whether an undetermined system of linear equation admits a sparse solution. This is also known as the *syndrome decoding* problem.

The (dual) LPN assumption implies a natural construction of pseudorandom generators, which maps (a short description of) $\mathbf{e}$ to $H \cdot \mathbf{e}$. This PRG (and variants thereof) is at the heart of all known construction of pseudorandom *correlation* generators, due to its linear structure which allows to preserve some target correlations. To obtain a pseudorandom correlation *function*, the work of [BCG+20a] faced the following dilemma: intuitively, we would like to extend the PRG that maps (a representation of) $\mathbf{e}$ to $H \cdot \mathbf{e}$ into a PRF, but this means that we need $H \cdot \mathbf{e}$ to be an exponentially long vector whose entry can be generated incrementally (in this view, an input defines a row $\mathbf{h}$ of the matrix H, the key defines $\mathbf{e}$, and the corresponding output is $\mathbf{h}^\intercal \cdot \mathbf{e}$). We need a way to guarantee that $\mathbf{e}$ and the rows of H both admit a short (polynomial size) representation, and that $\mathbf{h}^\intercal \cdot \mathbf{e}$ can be computed in polynomial time. Unfortunately, defining H and $\mathbf{e}$ to be exponentially sparse does not work in general: $H \cdot \mathbf{e}$ would then become sparse as well, and therefore trivial to distinguish from random.

The key observation in [BCG+20a], and the central idea of their design, is that we can circumvent this issue by making H and $\mathbf{e}$ exponentially sparse, but with *variable density*. Concretely, fix a security parameter λ and consider sampling the rows of H as follows: a row $\mathbf{h}$ is divided into λ blocks $(\mathbf{h}_i)_{i \leq \lambda}$ (looking ahead, the maximum number of queries to the PRF will be bounded by a quantity smaller than 2^λ). Each block $\mathbf{h}_i$ is of length $\lambda \cdot 2^i$ and contains exactly λ 1's: this guarantees that the *density* of $\mathbf{h}_i$ is $1/2^i$. More precisely, $\mathbf{h}_i$ is a concatenation of λ length-2^i unit vectors. This means that $\mathbf{h}$ constructed this way is a *variable density* vector, where the density drops by a factor two when going from one block to the next. The noise vector $\mathbf{e}$ is simply sampled as a row vector. Intuitively, the dense portion of the inner products $\mathbf{h}^\intercal \cdot \mathbf{e}$ guarantees that the result will not be sparse (but the corresponding portions being narrow, many linear dependencies appear), while the sparse portions of the $\mathbf{h}^\intercal \cdot \mathbf{e}$ break

linear dependencies (being exponentially wide, though very sparse). The VDLPN assumption states, informally, that this suffices to guarantee indistinguishability from random.

Definition 1 (VDLPN assumption, informal). *Sample a matrix H as $H = H_1||\cdots||H_\lambda$ over $\mathbb{F}_2^{N\times\lambda\cdot(2^{\lambda+1}-1)}$ where the rows are independently sampled as described above, and where $N \ll 2^\lambda$ is some bound on the maximum number of queries. Sample a noise vector $\mathbf{e}$ according to the same distribution as the rows of H. Then the VDLPN assumption states that, given H, $H\cdot\mathbf{e}$ is indistinguishable from a random length-N vector.*

VDLPN directly implies a natural construction of WPRF: a random input x (a bitstring of length $\lambda^2\cdot(\lambda+1)/2$) is parsed as λ blocks of length $\lambda\cdot i$, for $i = 1$ to λ, where each block is further parsed as λ sub-blocks of length i each. A length-i string defines a random unit vector of length 2^i (it encodes the position of the nonzero entry in the vector). The concatenation of these unit vectors forms a uniformly random row $\mathbf{h}_x$ for the matrix H. A similar mapping is applied to convert the bitstring K (the WPRF key) into a noise vector $\mathbf{e}_K$. Eventually, observe that the mapping $F_K : x \to \mathbf{h}_x^\intercal \cdot \mathbf{e}_K$ is efficient, because each of $\mathbf{h}_x^\cdot\mathbf{e}_K$ is exponentially sparse: computing their inner product amounts to computing $O(\lambda^2)$ equality tests between the sub-blocks of x and of K. To construct a pseudorandom *correlation* function, the authors of [BCG+20a] build upon the fact that this WPRF can further be written as a XOR of point functions (each point function takes a sub-block of x as input and returns 0 unless it is equal to the corresponding sub-block of K), which makes it *FSS-friendly*.

1.3 Security of VDLPN

Since VDLPN is a variant of the LPN assumption, the natural first step to analyze its security is to look at existing attacks on LPN. There have been, however, a tremendous number of attacks on LPN designed over the years, including attacks such as Gaussian elimination and the BKW algorithm [BKW00, Lyu05, LF06, EKM17] and variants based on covering codes [ZJW16, BV16, BTV16, GJL20], and attacks based on information set decoding techniques [Pra62,Ste88, FS09, BLP11, MMT11, BJMM12, MO15, EKM17, BM18]. This list is far from exhaustive; one could also mention statistical decoding attacks [AJ01, FKI06, Ove06, DAT17], generalized birthday attacks [Wag02, Kir11], linearization attacks [BM97, Saa07], attacks based on finding low weight code vectors [Zic17], and many more. A core observation of [BCG+20a] is that *all* these attacks fit in a common framework, called *linear tests*. Roughly, a linear test is an attack in which the adversary attempts to distinguish $\mathbf{b}$ from a random vector by finding a nonzero linear function L_H (which can depend on H in an arbitrary way) such that $L_H(\mathbf{b})$ is *biased* (i.e., far from uniform in statistical distance) when $\mathbf{b} = H\cdot\mathbf{e}$. Being secure against linear tests is a statistical property, which one can hope to prove unconditionally. To this end, the work of [BCG+20a] put forward the notion of *low bias* WPRF:

Definition 2 (low-bias WPRF, informal). *A family* $\{F_K\}_K$ *of WPRFs has* low bias *up to* N *samples if*

$$\Pr_{x_1,\cdots,x_N}[\mathsf{bias}(\mathcal{D}) \geq \mathsf{negl}(\lambda)] \leq \mathsf{negl}'(\lambda),$$

where $\mathcal{D}$ *is the distribution that samples a random key* K *for the WPRF, and outputs the vector* $(F_K(x_1), \cdots, F_K(x_N))$*. Above, the* N *inputs are sampled from the input space of the WPRF family, and* $\mathsf{negl}, \mathsf{negl}'$ *denote two negligible functions.*

Above, the bias of a distribution $\mathcal{D}$ over $\mathbb{F}_2^N$ is defined as $\max_{\mathbf{u}\neq\mathbf{0}} |1/2 - \Pr_{\mathbf{v}\leftarrow\mathcal{D}}[\mathbf{u}^\intercal \cdot \mathbf{v} = 1]|$; that is, it is the distance from the uniform distribution over $\mathbb{F}_2$ induced by computing $L(\mathbf{v})$ with the "worst possible" nonzero linear function $L : \mathbb{F}_2^N \mapsto \mathbb{F}_2$. One of the core security claims of [BCG+20a] hinges upon the fact that the VDLPN-based WPRF is a low-bias WPRF; in particular, this means that the VDLPN assumption cannot be broken using essentially any of the known attacks on LPN.

Theorem 3 (Resistance to linear tests [BCG+20a], informal). *The WPRF built from the VDLPN assumption has a low bias up to* $N = 2^{O(\lambda)}$ *samples (the functions* $\mathsf{negl}, \mathsf{negl}'$ *are both equal to* $2^{O(-\lambda)}$ *as well).*

To show that the VDLPN assumption is secure, we will only consider resistance against linear tests - and all our proof of security will consists of showing this resistance. A simple variant of the VDLPN assumption achieves smaller input size ($O(\lambda^2)$ instead of $O(\lambda^3)$), but we ignore it in this simplified overview. Note that [BCG+20a] also considers various other attacks, such as algebraic attacks, linear cryptanalysis, and attacks by low depth (AC^0) circuits. These analyses make VDLPN a plausible assumption, from which [BCG+20a] derives several consequences: a pseudorandom correlation function, as we already discussed, but also the first candidate WPRF in the very low complexity class $\mathsf{XOR\text{-}AND}$ (one layer of ANDs followed by a single XOR gate), which indicates that this class is perhaps hard to learn in the uniform PAC model. Furthermore, VDLPN also implies a WPRF secure against XOR related-key attacks, something which was previously known only assuming very strong cryptographic primitives (namely, high degree multilinear maps).

1.4 Our Contributions

We revisit the security analysis of VDLPN against linear tests. Our main motivation is that the analysis in [BCG+20a] is purely asymptotic, and trying to extract concrete parameters within the range where the analysis applies gives terrible performances. Concretely, let D be the number of blocks, w be the number of ones in each block, and N be a bound on the maximum number of queries (in our simplified exposition above, we used $w = D = \lambda$ and $N \ll 2^\lambda$). The authors of [BCG+20a] suggested the following concrete parameters to instantiate VDLPN: set $w = 1.5\lambda$, $D = w/4$, and $N = 2^D$. They conjectured that this

should achieve λ bits of security. However, this choice of parameters is purely heuristic, and described as a challenge to cryptanalysts: it is not backed up by any concrete cryptanalysis.

On the other hand, their analysis guarantees $2^{\Omega(w)}$ bits of security against linear tests whenever $w > \Gamma \cdot D$, for up to 2^D samples, where Γ is a constant from the proof. A quick back-of-the-envelope calculation reveals that Γ in their analysis is of the order of magnitude of 10^5, and it is far from obvious to improve the constants without significantly changing the analysis (while the proof is not tight, a straightforward "tightening" only saves a small factor). This means that, when instantiating the parameters within the range where the proof offers some security guarantees, the security parameters must be of the order of *several million bits long* – of course, this is entirely impractical.

Furthermore, upon revisiting their analysis, we uncovered one mistake (as well as a second, relatively minor mistake), that invalidates their proof of security against linear attacks. Fixing the mistake turns out to be non-trivial, and constitutes an important part of our contribution.

First contribution: a tighter VDLPN. The corrected analysis we present offers even worse concrete bounds than in the original (flawed) proof: the Γ value is of the order of 10^6, leading to a security parameter w in the millions. In other words, there is no realistically usable range of parameters within the bounds handled by the security analysis. Thus, one is left with a plausible assumption with purely asymptotic parameters on the one hand, and some concrete candidate choices of parameters that lead to a reasonably efficient PCF construction, but that are not supported by any security analysis. The goal of this first contribution is to bridge this (huge) gap between *secure in theory* and *usable in practice*. Since the task is highly non-trivial, we attack the problem simultaneously on three angles. Each angle in itself forms an orthogonal contribution to the overall analysis (in the sense that each of the three techniques leads to significant improvements by themselves).

- *An entirely new proof approach.* First, we step back from the original analysis and seek to understand the main source of slackness in the parameters. Then, we develop an alternative, much more direct approach which, in a sense, allows us to exploit the contribution to the bias of every component of the matrix H (while the previous analysis could only take into account the contribution of the "top contributors", for technical reasons). The new approach achieves much tighter bounds.
- *A proof-friendly VDLPN variant.* Second, we allow ourselves to (slightly) *change* the VDLPN assumption. Concretely, our variant is identical to VDLPN, except for the first block H_1: here, we set H_1 to be a uniformly random matrix instead. This choice stems from the fact that in the analysis, we need to use two different arguments to handle the low weight linear tests and the high weight linear tests; sampling H_1 uniformly at random allows to achieve much tighter bounds for the analysis against low weight tests. We observe

that this variant of VDLPN remains FSS-friendly: using this variant does not harm any of the cryptographic applications.
- *Better bounds through simulations.* Eventually, we rely on extensive computer simulations to achieve tighter bounds. Concretely, we need a bound on the expectation of some complex random variable X, which we obtained using a generalized Chernoff inequality in the previous analysis. While this bound suffices for the asymptotic analysis, its looseness severely impacts the bounds. Here, instead, we estimate $\mathbb{E}[X]$ through computer simulations. We empirically observe that the samples from X have very low variance, and derive a tight bound on $\mathbb{E}[X]$ with a very high confidence interval.

Putting everything together, we manage to prove that (our variant of) VDLPN has bias at most 2^{-80} with probability at least $1 - 2^{-80}$, for a value of w as low as $w = 380$ (with $D = 30$, and up to $N = 2^D$ samples – this is just a sample of candidate parameters, we do not have a closed-form formula).[4] This is a tremendous improvement compared to the previous analysis, and gives for the first time a set of parameters which are simultaneously backed by a thorough security analysis, yet are usable in practice. We stress that, in spite of our computer-verified component, our bounds are much better than purely heuristic bounds: they are provable bounds under a simple, concrete combinatorial conjecture, which is easy to verify through computer experiments. In contrasts, even ignoring the flaw in their asymptotic analysis, all "usable parameters" proposed in [BCG+20a] were purely heuristic, based on the intuition that they might be hard to attack and described as challenges for cryptanalysis, but not supported by any analysis whatsoever.

We believe that our work constitutes a strong step in the direction of showing that one can construct secure and concretely efficient pseudorandom correlation functions, an important and intriguing goal.

Second contribution: fixing the original analysis. In essence, the analysis of a central claim in the proof of resistance against linear tests turned out to be incorrect. The claim, on the other hand, remains essentially correct (up to some concrete choice of the constants involved): only its analysis is flawed, it did not lead to attacks. The mistake appears in a bound on the expectation $\mathbb{E}[Z]$ of a random variable Z, of the form $\mathbb{E}[Z] \in [a, b]$, for some values $0 < a < b$. The authors deduced from this bound a bound of the form $\mathbb{E}[|Z - b| \leq b - a$, but this is wrong in general (the error might stem from an application of the Jensen inequality in the wrong direction): intuitively, if the distribution of Z is "anti-concentrated" with respect to its expectation, then the inequality $\mathbb{E}[|Z-b| \leq b-a$ does not follow from $\mathbb{E}[Z] \in [a, b]$[5].

[4] The choice of 80 bits of security is more conservative than it appears: it means that an adversary will have to compute 2^{80} inner products with a length-2^{30} vector to detect a 2^{-80} bias in the output. In terms of bit-security, this corresponds to at least 110 bits of security.

[5] E.g. if Z is 0 with probability $1/2$, and 10 else, then $\mathbb{E}[Z] = 5 \in [4, 5]$, but $\mathbb{E}[|Z-5|] = 5 > 5 - 4$

On the other hand, if Z follows a "nice" distribution, typically a Gaussian-style distribution (or any bell-shape distribution), and if the value b is sufficiently close to $\mathbb{E}[Z]$, then the claim becomes true. A quick simulation reveals that Z indeed appears to exhibit the right structure. Central to our first contribution is a formal proof that the claim holds for Z. Compared to the analysis of [BCG+20a], our new analysis cannot simply bound the expected value of Z: we have to prove strong *tail bounds* on Z, which is significantly more complex, because Z is a sum of *dependent* variables. Our analysis relies on a power-full bound about the balls and bins problem.

We then turn to integrating our new proof of the central claim to the full proof of resistance against linear tests. Along the way, we found (and fixed) another minor mistake in the analysis, which requires changing the concrete choices of constants in the proof. Due to this, and due to some slackness in our new proof of the central claim (which stems from the limitations of the inequality which we use), the general proof ends up failing on some corner cases. Essentially, the analysis studies separately the contribution of each block H_i of the matrix H to the overall bias; the analysis, however, fails whenever i is too small. Nevertheless, we show that the case of very small values of i can be treated separately with two simple arguments, which completes the proof.

We stress that while the repaired proof follows the high level structure of that of [BCG+20a], the core of correction was not straight forward. This security analysis against linear tests is central to the claim that VDLPN is a plausible assumption (since it resists all known attacks against LPN), and therefore provides a plausible candidate to construct powerful objects such as a PCF (for all low-degree correlations), a XOR-RKA secure WPRF, and a family of extremely simple functions (in the XOR-AND class) hard to learn in the uniform PAC model. We also mention that we notified the authors of [BCG+20a] of our findings, and they acknowledged the flaws in the analysis.

1.5 New Cost Estimations for PCFs, and Challenges

Using the parameters from above ($w = 380, D = 30$), we compute the seed size and estimate the evaluation time of the pseudorandom correlation function of [BCG+20a] instantiated using our new VDLPN variant. On top of the VDLPN variant, the construction uses a puncturable pseudorandom function, instantiated with the GGM construction [GGM86]. We set the security parameter of the PRG used in GGM to $\lambda = 128$. With these parameters, we get the following costs:

- Seed size: 2.94MB
- PCF evaluation time: the evaluation cost is (largely) dominated by $\approx 1.81 \cdot 10^5$ calls to a length-doubling pseudorandom generator.

To give a rough runtime estimation, the PRG can be instantiated using two calls to fixed-key AES. According to [MSY21], using the AES-NI instructions of modern CPUs, one byte of AES-128 can be computed in ~ 1.3 cycles. Hence,

computing $3.6 \cdot 10^5$ blocks of 16 bytes requires about $7.5 \cdot 10^6$ cycles. Concretely, using a 3.8GHz processor, this amounts to roughly 500 PCF evaluations per second on a single core (note that the estimation should not be too far off, because the computation requires no random data access, hence cache misses are unlikely). Since all evaluations are fully parallelizable, using c cores increases this number to $500c$ evaluations per second.

The work of [BCG+20a] also suggested an improved *all prefix* variant, which has shorter seeds and better runtimes, using existing efficient constructions of all-prefix function secret sharing. While this construction lacks a security analysis, this is only because it makes the noise vectors $\mathbf{e}_i$ correlated (our analysis fundamentally uses their independence). However, it seems very reasonable to conjecture that this is just an artefact of the analysis, and that the optimized construction provides the same security level. Under the heuristic assumption that the correlated $\mathbf{e}_i$ behave essentially as well as independent $\mathbf{e}_i$ for resistance to linear tests, we can reuse our previous analysis and obtain the following improved bounds for the *all-prefix* PCF: seed size 0.35MB, and PCF evaluation time around 3950 evaluations per second on a single 3.8GHz processor.

These numbers demonstrate that, already within the range of our provable bounds, PCFs can achieve very promising parameters, with short seeds, and reasonably fast runtimes. Note that we believe that there remains some small gap between our analysis and the "true" security of VDLPN – namely, smaller parameters might plausibly lead to a secure instance (perhaps as small as $w = 120$ and $D = 30$). We view further tightening our analysis as an interesting open question. Since the cost is linear in w, reducing w to 120 would lead to a factor 3 improvement (on seed size and evaluations per second). Nonetheless, our provable parameters appear already quite tight, being at most a factor-3 off compared to the best parameters one could heuristically hope for.

2 Preliminaries

We use bold font for vectors, and capitals for matrices. For vectors $\mathbf{u}, \mathbf{v}$, $\mathsf{HW}(\mathbf{u})$ denotes the Hamming weight of $\mathbf{u}$, $\mathrm{d_H}(\mathbf{u}, \mathbf{v})$denotes the Hamming distance between $\mathbf{u}, \mathbf{v}$. Below, we recall the definition of the bias of a distribution, and some standard technical lemmas.

Definition 4 (Bias of a Distribution). *Given a distribution $\mathcal{D}$ over $\mathbb{F}^n$ and a vector $\mathbf{u} \in \mathbb{F}^n$, the bias of $\mathcal{D}$ with respect to $\mathbf{u}$, denoted $\mathsf{bias}_{\mathbf{u}}(\mathcal{D})$, is equal to*

$$\mathsf{bias}_{\mathbf{u}}(\mathcal{D}) = |\mathbb{E}_{\mathbf{x}\sim\mathcal{D}}[\mathbf{u}^\intercal \cdot \mathbf{x}] - \mathbb{E}_{\mathbf{x}\sim\mathcal{U}_n}[\mathbf{u}^\intercal \cdot \mathbf{x}]| = \left|\mathbb{E}_{\mathbf{x}\sim\mathcal{D}}[\mathbf{u}^\intercal \cdot \mathbf{x}] - \frac{1}{|\mathbb{F}|}\right|,$$

where $\mathcal{U}_n$ denotes the uniform distribution over $\mathbb{F}^n$. The bias of $\mathcal{D}$, denoted $\mathsf{bias}(\mathcal{D})$, is the maximum bias of $\mathcal{D}$ with respect to any nonzero vector $\mathbf{u}$.

Standard Probability Lemmas. Given t distributions $(\mathcal{D}_1, \cdots, \mathcal{D}_t)$ over $\mathbb{F}_2^n$, we denote by $\bigoplus_{i\le t} \mathcal{D}_i$ the distribution obtained by *independently* sampling $\mathbf{v}_i \xleftarrow{\$}$

$\mathcal{D}_i$ for $i = 1$ to t and outputting $\mathbf{v} \leftarrow \mathbf{v}_1 \oplus \cdots \oplus \mathbf{v}_t$. We will use the following bias of the exclusive-or (cf. [Shp09]).

Lemma 5. *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 $\mathsf{bias}(\bigoplus_{i \leq t} \mathcal{D}_i) \leq 2^{t-1} \cdot \prod_{i=1}^{t} \mathsf{bias}(\mathcal{D}_i) \leq \min_{i \leq t} \mathsf{bias}(\mathcal{D}_i)$.*

Let $\mathsf{Ber}_r(\mathbb{F}_2)$ denote the Bernoulli distribution that outputs 1 with probability r, and 0 otherwise. More generally, we denote by $\mathsf{Ber}_r(\mathbb{F})$ the distribution that outputs a uniformly random element of $\mathbb{F}$ with probability r, and 0 otherwise (this does not exactly match our definition of $\mathsf{Ber}(\mathbb{F}_2)$, but the slight discrepancy will not matter in our applications). We will use a standard simple lemma for computing the bias of a XOR of Bernoulli samples:

Lemma 6 (Piling-up lemma). *For any $0 < r < 1/2$ and any integer n, given n random variables $X_1, \cdots, X_n$ i.i.d. to $\mathsf{Ber}_r(\mathbb{F}_2)$, it holds that $\Pr[\bigoplus_{i=1}^{n} X_i = 0] = 1/2 + (1-2r)^n/2$.*

We will also need two concentration bounds. The bounded difference inequality [McD89] is an application of the more general Azuma inequality [Azu67]. 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 $\mathbf{x}, \mathbf{x}' \in [n]^m$ which differ in a single coordinate, it holds that $|\Phi(\mathbf{x}) - \Phi(\mathbf{x}')| \leq d$.

Lemma 7 (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*

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

Eventually we will rely on the Occupancy Bound from [KMPS94], which provides tight bounds for the balls and bins problem.

Lemma 8 (Occupancy Bound). *Let E be the number of empty bins when m balls are placed randomly into n bins, and define $r = m/n$. The expectation of E is given by $\mu = \mathbb{E}[E] = (1 - \frac{1}{n})^m \approx ne^{-r}$. For any $\theta > 0$,*

$$\Pr[|E - \mu| \geq \theta\mu] \leq 2\exp\left(-\frac{\theta^2\mu^2(n - \frac{1}{2})}{n^2 - \mu^2}\right) = \mathcal{B}$$

Note that we can derive the following two equations : $\Pr[E \geq \mu(\theta + 1)] < \mathcal{B}$ *and* $\Pr[E \leq \mu(1 - \theta)] < \mathcal{B}$

2.1 Coding Theory

Definition 9. *Let n be a positive integer, $\mathcal{C}$ is a linear code if $\mathcal{C}$ is a vector subspace of $\mathbb{F}_q^n$. The integer n is called the length of $\mathcal{C}$. The dimension of $\mathcal{C}$ is its dimension as an $\mathbb{F}_q$-vector space. It is denoted by $k = \dim_{\mathbb{F}_q} \mathcal{C}$*

Definition 10. *(Minimum distance of a code) Let $\mathcal{C}$ be a linear code of length n. The minimum distance of $\mathcal{C}$, is the minimum distance $d_{\mathcal{C}}$ between two distinct codewords of $\mathcal{C}$.*

$$d_{\mathcal{C}} = \min_{\mathbf{x},\mathbf{y} \in \mathcal{C}, \mathbf{x} \neq \mathbf{y}} \{d_{\mathrm{H}}(\mathbf{u}, \mathbf{v})(\mathbf{x}, \mathbf{y})\}$$

Learning Parity with Noise. We define the LPN assumption over a ring $\mathcal{R}$ with dimension k, number of samples n, w.r.t. a code generation algorithm $\mathbf{C}$, and a noise distribution $\mathcal{D}$:

Definition 11 (Dual LPN). *Let $\mathcal{D}(\mathcal{R}) = \{\mathcal{D}_{k,n}(\mathcal{R})\}_{k,n\in\mathbb{N}}$ denote a family of efficiently sampleable distributions over a ring $\mathcal{R}$, such that for any $k, n \in \mathbb{N}$, $\mathsf{Im}(\mathcal{D}_{k,n}(\mathcal{R})) \subseteq \mathcal{R}^n$. Let $\mathbf{C}$ be a probabilistic code generation algorithm such that $\mathbf{C}(k, n, \mathcal{R})$ outputs a matrix $H \in \mathcal{R}^{k\times n}$. For dimension $k = k(\lambda)$, number of samples (or block length) $n = n(\lambda)$, and ring $\mathcal{R} = \mathcal{R}(\lambda)$, the (dual) $(\mathcal{D}, \mathbf{C}, \mathcal{R})$-$\mathsf{LPN}(k, n)$ assumption states that*

$$\{(H, \mathbf{b}) \mid H \xleftarrow{\$} \mathbf{C}(k, n, \mathcal{R}), \mathbf{e} \xleftarrow{\$} \mathcal{D}_{k,n}(\mathcal{R}), \mathbf{b} \leftarrow H \cdot \mathbf{s}\}$$
$$\overset{c}{\approx} \{(H, \mathbf{b}) \mid H \xleftarrow{\$} \mathbf{C}(k, n, \mathcal{R}), \mathbf{b} \xleftarrow{\$} \mathcal{R}^n\}.$$

The dual LPN assumption is also called the *syndrome decoding assumption* in the code-based cryptography literature. The dual LPN assumption as written above is equivalent to the *primal* LPN assumption with respect to G (a matrix $G \in \mathcal{R}^{n\times n-k}$ such that $H \cdot G = 0$), which states that $G \cdot \mathbf{s} + \mathbf{e}$ is indistinguishable from random, where $\mathbf{s} \xleftarrow{\$} \mathcal{R}^{n-k}$ and $\mathbf{e} \xleftarrow{\$} \mathcal{D}_{k,n}(\mathcal{R})$; the equivalence follows from the fact that $H(G \cdot \mathbf{s} + \mathbf{e}) = H \cdot \mathbf{e}$.

The standard LPN assumption refers to the case where H is a uniformly random matrix over $\mathbb{F}_2$, and $\mathbf{e}$ is sampled from $\mathsf{Ber}_r(\mathbb{F}_2)$, where r is called the *noise rate*. Other common noise distributions include exact noise (the noise vector $\mathbf{e}$ is a uniformly random weight-rn vector from $\mathbb{F}_2^n$; this is a common choice in concrete LPN-based constructions) and regular noise (the noise vector $\mathbf{e}$ is a concatenation of rn random unit vectors from $\mathbb{F}_2^{1/r}$, widely used in the PCG literature [BCGI18, BCG+19b, BCG+19a]).

2.2 The Variable Density LPN assumption

We recall the *regular VDLPN* assumption from [BCG+20a]; other variants exist. Let λ be a security parameter. We fix three parameters: a *sparsity* parameter $w = w(\lambda)$ (controlling the number of ones per row of a block), a *block* parameter $D = D(\lambda)$ (controlling the number of blocks), and a bound $N = N(\lambda)$ on the number of samples. The reader can think of w, D as being $\Omega(\lambda)$, with $D < w$, and $N = 2^D$ for concreteness. We set $\mathsf{par} \leftarrow (w, D, N)$.

Let $\mathcal{S}_{1,2^i}$ the distribution of unit vector of size 2^i. Let $\mathcal{R}_{w,i}$ be the distribution of random w-regular vectors over $\mathbb{F}_2^{w\cdot 2^i}$, i.e., the concatenation of w vector sampled from $\mathcal{S}_{1,2^i}$). Let $\mathcal{H}^i_{\mathsf{par}}$ denote the distribution over $N \times (w \cdot 2^i)$ matrices over $\mathbb{F}_2$, where each row of the matrix is sampled independently from $\mathcal{R}_{w,i}$, and let $\mathcal{H}_{\mathsf{par}}$ denote the distribution over $\mathbb{F}_2^{N\times 2N\cdot w}$, obtained by sampling $H_i \xleftarrow{\$} \mathcal{H}^i_{\mathsf{par}}$ for $i = 1$ to D and outputting $H = H_1||\cdots||H_D$, where $||$ is the horizontal concatenation. Eventually we denote $\mathcal{N}_{\mathsf{par}}$ the noise distribution obtained by sampling $\mathbf{e_i}^\top$ according $R_{w,i}$ and outputting $\mathbf{e} \leftarrow (\mathbf{e_1} /\!\!/ \cdots /\!\!/ \mathbf{e_D}) \in \mathbb{F}_2^{2N\cdot w}$ where $/\!\!/$ is this time the vertical concatenation. The matrix H_i sampled from $\mathcal{H}^i_{\mathsf{par}}$ is:

$$H_i = \begin{bmatrix} & & \overbrace{\phantom{u^i_{1,w}}}^{2^i \text{ columns}} \\ u^i_{1,1} & \cdots & u^i_{1,w} \\ \vdots & \vdots & \vdots \\ u^i_{N,1} & \cdots & u^i_{N,w} \end{bmatrix}$$

where $(u^i_{k,j})_{1 \leq k \leq N, 1 \leq j \leq w}$ are sampled from the distribution $\mathcal{S}_{1,2^i}$, and are unit vector over $\mathbb{F}_2^{2^i}$. Thus, there is w non-zero coordinates by rows. Eventually, the matrix H sampled from $\mathcal{H}_{\mathsf{par}}$ is a horizontal concatenation of the H_i:

$$H = \underbrace{[H_1 \cdots H_D]}_{w \cdot 2^{D+1} \text{ columns}}$$

The term *variable density* refers to the fact that the density of 1's in each block H_i is $1/2^i$ by construction. For any H sampled from the distribution $\mathcal{H}_{\mathsf{par}}$ let $\mathcal{O}_{\mathsf{par}}(H)$ be the distribution which samples $\mathbf{e} \xleftarrow{\$} \mathcal{N}_{\mathsf{par}}$ and return $H \cdot \mathbf{e}$.

Definition 12 ($\mathsf{rVDLPN}(w, D, N)$). *The regular VDPLN assumption, with parameters $\mathsf{par} = (w, D, N)$, denoted $\mathsf{rVDLPN}(w, D, N)$), states that:*

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

Note that this is exactly the dual LPN assumption where both the matrix and the noise are sampled from a specific distribution variable-density matrices and vectors.

A WPRF candidate from the rVDLPN assumption. Fix parameters $\mathsf{par}(\lambda) = (w(\lambda), D(\lambda), N(\lambda) = 2^{D(\lambda)})$. Recall that a vector from the distribution $\mathcal{N}_{\mathsf{par}}$ is in fact the vertical concatenation of D vectors from $\mathbf{e_i}$, where $\mathbf{e_i}$ is the transpose vector of the vector from the distribution $\mathcal{R}_{w,i}$. Moreover, $\mathcal{R}_{w,i}$ is the concatenation of w unit vector over $\mathbb{F}_2^{2^i}$, where each of them can be generated with i random bits (encoding the index of the nonzero entry). Therefore, sampling a vector $\mathcal{N}_{\mathsf{par}}$ requires exactly $w \cdot \sum_{i=1}^{D} i = w \cdot D(D-1)/2$ random bits; we write $\mathcal{N}_{\mathsf{par}}(r)$ to denote the vector $\mathbf{e}$ sampled from $\mathcal{N}_{\mathsf{par}}$ using randomness r. We describe the WPRF candidate below.

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

Theorem 13 ([BCG+20a]). *Suppose that $\mathsf{rVDLPN}(\mathsf{par})$ holds. Then the above construction is an N-query WPRF, with input length and key length equal at $w \cdot D(D-1)/2$.*

2.3 Pseudorandom Correlation Functions

Pseudorandom correlation functions, introduced in [BCG+20a], allow to locally generate, from a pair of short correlated keys, an arbitrary polynomial amount of pseudorandom correlations, in an incremental way. A fundamental application of PCF is to secure computation in the preprocessing model: two parties can distributively generate PCF keys, and later use them every time they wish to engage in a secure computation protocol, to generate locally (without any interaction) all preprocessing material required for the protocol. Therefore, PCFs allow to confine *all* future preprocessing phases of all secure computation protocols two parties could want to execute, to a single, one time generation of short correlated keys, followed solely by local computations. Slightly more formally, a PCF is a pair $(\mathsf{PCF.Setup}, \mathsf{PCF.Eval})$ where $\mathsf{PCF.Setup}$ generates short correlated keys $(\mathsf{k}_0, \mathsf{k}_1)$, and $\mathsf{PCF.Eval}(\sigma, \mathsf{k}_\sigma, x)$ outputs a value y_σ such that for any input x, given $\mathsf{k}_{1-\sigma}$, the value y_σ is indistinguishable from a random value sampled conditioned on satisfying the target correlation with $\mathsf{PCF.Eval}(\sigma, \mathsf{k}_{1-\sigma}, x)$ (for $\sigma = 0, 1$). Due to lack of space, we defer the formal definition of PCF to Appendix A of the Supplementary Material.

2.4 Pseudorandom Correlation Function from VDLPN

A construction of PCF from VDLPN follows from the general template established in [BCG+20a], which combines a WPRF in a suitably low complexity class with a function secret sharing scheme for a related class. Instantiating this general template with the VDLPN-based WPRF and the FSS scheme of [BGI16], one gets a PCF for a general class of constant degree polynomial additive correlations. For the sake of concreteness, though, we focus here on PCFs for the random oblivious transfer (OT) correlation, one of the most fundamental and useful correlation in secure computation. A random OT correlation is a pair $(y_0, y_1) \in \{0,1\}^2 \times \{0,1\}^2$, where $y_0 = (u, v)$ for two random bits u, v, and $y_1 = (b, u \cdot (1-b) \oplus v \cdot b)$ for a random bit b.

It is known that, to generate n pseudorandom OT correlations, it suffices to generate the following simpler correlation: Alice gets a (pseudo)random pair of length-n vectors $(\mathbf{u}, \mathbf{v})$, where $\mathbf{u} \xleftarrow{\$} \mathbb{F}_2^n$ and $\mathbf{v} \in \mathbb{F}_{2^\lambda}^n$, and Bob gets $x \xleftarrow{\$} \mathbb{F}_{2^\lambda}$ and $\mathbf{w} \leftarrow x \cdot \mathbf{u} + \mathbf{v}$. This correlation (known as the *subfied vector-OLE* correlation) can be locally converted by Alice and Bob into n pseudorandom OT correlations using a correlation-robust hash function; see [BCG+19b] for details. Therefore, we focus on building a PCF for the subfield VOLE correlation. Unlike the general case, this does not require the full power of function secret sharing: it suffices to rely on a simpler primitive, namely, a puncturable pseudorandom function.

Puncturable pseudorandom functions. A *puncturable pseudorandom function* (PPRF) is a PRF F such that given an input x, and a PRF key k, one can generate a *punctured* key, denoted $k\{x\}$, which allows evaluating F at every point except for x, and does not reveal any information about the

value $F.\mathsf{Eval}(k, x)$. PPRFs have been introduced in [KPTZ13, BW13, BGI14]. Formally, a t-puncturable pseudorandom function (PPRF) with key space $\mathcal{K}$, domain $\mathcal{X}$, and range $\mathcal{Y}$, is a pseudorandom function F with an additional punctured key space $\mathcal{K}_p$ and three probabilistic polynomial-time algorithms $(F.\mathsf{KeyGen}, F.\mathsf{Puncture}, F.\mathsf{Eval})$ such that

- $F.\mathsf{KeyGen}(1^\lambda)$ outputs a random key $K \in \mathcal{K}$,
- $F.\mathsf{Puncture}(K, S)$, on input a ley $K \in \mathcal{K}$, and a subset $S \subset \mathcal{X}$ of size (at most) t, outputs a punctured key $K\{S\} \in \mathcal{K}_p$,
- $F.\mathsf{Eval}(K\{S\}, x)$, on input a key $K\{S\}$ punctured at all points in S, and a point x, outputs $F(K, x)$ if $x \notin S$, and $\perp$ otherwise.

The (static) security of a t-PPRF states is captured by the following game: the adversary $\mathcal{A}$ sends a size-t subset S of inputs. The challenger generates a key K, a punctured key $K\{S\}$, and a random bit b. He sends $K\{S\}$ to $\mathcal{A}$, together with either the values $F_K(x) = F.\mathsf{Eval}(K\{\emptyset\}, x)$ for all $x \in S$ if $b = 0$, or t random bits if $b = 1$. The PPRF is secure if any adversary has negligible advantage over the random guess for finding b in this game. A t-PPRF can be constructed from any one-way function, using the GGM construction [GGM86].

A PCF for SVOLE from VDLPN and a PPRF. We briefly sketch the construction, and refer to [BCG+20a] for a formal analysis. Fix VDLPN parameters $\mathsf{par} = (w, D, N)$ and set $t \leftarrow D \cdot w$. Let F be a t-PPRF with range $\mathbb{F}_{2^\lambda}$.

- $\mathsf{PCF.Setup}(1^\lambda)$: sample $r \xleftarrow{\$} \{0,1\}^{t(D-1)/2}$ and set $\mathbf{e} \leftarrow \mathcal{N}_{\mathsf{par}}(r)$. Let $S \subseteq [w \cdot (2^{D+1} - 1)]$ be the size-t subset of nonzero entries of $\mathbf{e}$. Sample $K \leftarrow F.\mathsf{KeyGen}(1^\lambda)$ and set $K\{S\} \leftarrow F.\mathsf{Puncture}(K, S)$. Sample $x \xleftarrow{\$} \mathbb{F}_{2^\lambda}$ and let $(K_y)_{y \in S} \leftarrow (F_K(i) - x)_{i \in S}$. Set $\mathsf{k}_0 \leftarrow (K, x)$ and $\mathsf{k}_1 \leftarrow (r, K\{S\}, (K_i)_{i \in S})$.
- $\mathsf{PCF.Eval}(\sigma, \mathsf{k}_\sigma, z)$: parse z as a row $\mathbf{h}_z$ of the VDLPN matrix H (i.e., set $\mathbf{h}_z^\intercal \leftarrow \mathcal{N}(z)$). Let $S_z \subseteq [w \cdot (2^{D+1} - 1)]$ denote the index of the 1's in $\mathbf{h}_z$. If $\sigma = 0$, output x and $w = \sum_{i \in S_z} F_K(i)$. If $\sigma = 1$, output $u = h_z^\intercal \cdot \mathbf{e}$ and $v = \sum_{i \in S_z \setminus S} F.\mathsf{Eval}(K\{S\}, i) + \sum_{i \in S_z \cap S} K_i$.

For correctness, observe that for every $i \in S_z \setminus S$, $F_K(i) - F.\mathsf{Eval}(K\{S\}, i) = 0$, and for every i in $S_z \cap S$, $F_K(i) - K_i = x$. Since S_z denotes the 1 entries in $\mathbf{h}_z$ and S denotes the 1 entries in $\mathbf{e}$, we have $w - v = \sum_{i \in S_z} F_K(i) - \sum_{i \in S_z \setminus S} F.\mathsf{Eval}(K\{S\}, i) - \sum_{i \in S_z \cap S} K_i = x \cdot (\mathbf{h}_z^\intercal \cdot \mathbf{e}) = x \cdot u$; the pseudorandomness of u follows from the fact that $z \mapsto h_z^\intercal \cdot \mathbf{e})$ is a WPRF under the VDLPN assumption, and that of w follows from the pseudorandomness of the PPRF.

2.5 Outline of the Original Proof of Resistance against Linear Test

We provide here an overview of the original security analysis in [BCG+20a], resistance against linear attacks. The two claims for which the analysis was

flawed are the Equation 1 and the Lemma 17 . We explain the errors and provide a correction in the Section 4.

As outlined in the introduction, the goal of this analysis is to show that the VDLPN assumption cannot be broken by any *linear test*, which captures in particular all known attacks against LPN. This is formalized in the following theorem:

Theorem 14. *[Resistance against linear tests] There exist constants (Γ, μ, ν), such that for any large enough w, any $\Gamma \cdot D \leq w, N \leftarrow 2^D, \mathsf{par} \leftarrow (w, D, N)$, it holds that*

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

This theorem states that with high 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 $\mathbf{y} = (F_K(x^{(1)}, \cdots, F_K(x^{(N)}))$ has an advantage of at most μ^w in distinguishing the string from uniform. Note that the choice of the linear function can depend arbitrarily on $(x^{(1)}, \cdots, x^{(N)})$.

To bound the bias of $\mathcal{O}_{\mathsf{par}}(H)$, the authors look at the sub-matrices of H, and introduce a notion of *good* and *bad* matrices:

Definition 15. *Given a matrix $M \in \mathbb{F}_2^{N \times 2^i}$, M is judged* bad *with respect to a vector $\mathbf{v} \in \mathbb{F}_2^N$ if*

$$\mathsf{HW}(\mathbf{v}^\top \cdot M) \notin \left[\frac{2^i}{5}, \frac{2^{i+2}}{5}\right].$$

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

A matrix is bad with respect to a vector $\mathbf{v}$ if the bias it induces against the test vector $\mathbf{v}$ is large. The goal of the proof is to guarantee that, with high probability, at least half of the matrices are good. This is stated in the following lemma.

Lemma 16. *There is a constant C, such that for any $1 \leq i \leq D$, and for any vector $\mathbf{v} \in \mathbb{F}_2^N$ such that $\mathsf{HW}(\mathbf{v}) \in [2^{i-1}, 2^i]$, it holds that*

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

The above lemma shows that for any fixed vector $\mathbf{v}$ of weight close to 2^i, the distribution induced by $\mathcal{H}^i_{\mathsf{par}}$ has a low bias against $\mathbf{v}$. The probability that this holds is so high that it remains overwhelming even after a union bound over *all* vectors $\mathbf{v}$ of weight in $[2^{i-1}, 2^i]$. Hence, this implies that in the output $H \cdot \mathbf{e} = \bigoplus_i H_i \cdot \mathbf{e}_i$, each component $H_i \cdot \mathbf{e}_i$ will guarantee low-bias against all vectors in this window of weight; the XOR of these independent samples will inherit the low-bias of all its components, and therefore resist all linear tests.

Bounding the number of bad matrices. In [BCG+20a], the authors reformulate the event that a matrix M is *bad* as a balls and bins problem. Let $M \xleftarrow{\$} \mathcal{H}^i_{\mathsf{par}}$. Recall that by definition of $\mathcal{H}^i_{\mathsf{par}}$, the rows of M are generated independently from $\mathcal{S}_{1,2^i}$. We start with 2^i empty bins, each bin corresponding to a column of M. Sampling a row of M according to the $\mathcal{S}_{1,2^i}$ distribution amounts to throwing a ball randomly into one of the 2^i bins. For a vector $\mathbf{v}$ of weight $l \in [2^{i-1}, 2^i]$, the event $\mathsf{HW}(\mathbf{v}^\top \cdot M) \notin \left[\frac{2^i}{5}, \frac{2^{i+2}}{5}\right] = I_i$ is equivalent to the following event: after randomly throwing l balls into 2^i bins, the number T of bins that contain an odd number of balls satisfies $T \notin I_i$. We have therefore the following experiment: take 2^i bins and throw $l \cdot w$ balls into the bins in w consecutive phase. Each time that l balls have been thrown, we check that the proportion of the number of bins that contains an odd number of balls is between $1/5$ and $4/5$, and clear out the bins. At the end, we return *failure* if more than $w/2$ of the w checks have failed. To bound the probability of returning a failure, define the following *cost function* $\Phi(X_{1,1}, \cdots, X_{l,w}) = \sum_{k=1}^{w} \left(2^{i-1} - \left|\mathsf{HW}\left(\bigoplus_{j=1}^{l} X_{j,k}\right) - 2^{i-1}\right|\right)$, where each $X_{j,k}, 1 \leq j \leq l, 1 \leq k \leq w$, is the random variable corresponding to the bin in which the j-th balls of the k-th phase was thrown (seen as a length-2^i unit vector with a 1 at the bin position). The $X_{j,k}$ are independent. Bounding the number of bad matrices, the authors claimed, amounts to bounding Φ. Indeed:

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

Afterwards, it suffices to bound Φ to conclude. The claim is that the following bound holds:

$$\Pr\left[\Phi(X_{1,1}, \cdots, X_{l,w}) < \frac{w \cdot 2^i}{10}\right] \leq 2^{-C \cdot 2^i \cdot w}. \tag{2}$$

The choice of Φ is of course not arbitrary: Φ is a well-behaved function, in the sense that it is 2-Lipschitz – i.e., changing any single input to Φ can only change its output by at most 2. Fortunately, strong concentration bounds are known on the probability that Lipschitz functions deviate too much from their mean. It therefore only remains to apply such a bound (which is here the McDiarmid inequality, a variant of the Azuma inequality), to get an estimate of the mean of Φ. This bound is stated in the following lemma:

Lemma 17.

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

Given the proof of Lemma 17, the McDiarmid inequality provides a bound on Φ, which translates to a bound on $N_{\mathbf{v}}$ by Equation 1. A union bound over all vectors of weight between $[2^{i-1}, 2^i]$ allows to conclude:

$$\Pr_{M_1,\cdots,M_w \xleftarrow{\$} \mathcal{H}^i_{\mathsf{par}}}\left[\exists \mathbf{v} \in \mathcal{S}_{i,N}, N_{\mathbf{v}}(M_1,\cdots,M_w) \geq \frac{w}{2}\right] \leq 2^{D\cdot 2^i}\cdot 2^{-C\cdot w\cdot 2^i} \leq 2^{-a\cdot w},$$

with $a = \frac{C}{2} > D$. The proof ends with a last union bound over all matrices H_i, for $1 \leq i \leq D$.

Some notations. In the following, we will denote by $X_{j,k}$ indicates the bin into which the j-th ball of the k-th phase is thrown ($X_{j,k}$ is a unit vector). Given a test vector $\mathbf{v} \in \mathbb{F}_2^N$ of weight $\mathsf{HW}(\mathbf{v}) = l$, we define $R_{i,l,k} = \mathsf{HW}(\mathbf{v}^\top \cdot M) = \mathsf{HW}\left(\bigoplus_{j=1}^{l} X_{j,k}\right)$. That is, $R_{i,l,k}$ it is the number of bins that contains an odd number of 1 in the k-th phase; we usually write it $R_{l,k}$ when i is clear from the context. We further define $Z_{i,l,k}$ as $Z_{i,l,k} = |2^{i-1} - R_{l,k}|$ (also usually written $Z_{l,k}$). Eventually, we denote by $S_{i,N}$ the set of vectors $\mathbf{v} \in \mathbb{F}_2^N$ with $\mathsf{HW}(\mathbf{v}) \in [2^{i-1}, 2^i]$.

3 Faster PCF from a VDLPN variant

The original proof shows that for an appropriate choice of a constant Γ, if $w \geq \Gamma \cdot D$, then the bias of $\mathcal{O}_{\mathsf{par}}$ is $2^{-\Omega(w)}$ with probability $1 - 2^{-\Omega(w)}$. However, the concrete constants are utterly impractical (the correction of the flaw doesn't help as we will see in section 4). With a quick back-of-the-envelope calculation, to guarantee $D\cdot 2^{-a\cdot w} < 2^{-80}$ we need $w > \frac{85}{a}$ (for $D = 30$). However, the value a in our analysis satisfies $a < \frac{1}{40000}$, leading to a necessary value of $w \approx 10^6$.These parameters are of course completely unusable. Therefore, in its current state, the proof only shows the asymptotic security of the construction in a parameter range which cannot be instantiated; any concrete instantiation is bound to rely only on heuristic parameters instead, not backed up by any security analysis. In this section, we aim at mitigating this unsatisfying situation, and provide a parameter set which is simultaneously usable in practice, and comes with provable security guarantees.

3.1 A Proof Friendly VDLPN Variant for Resistance against linear attack

We put forth a simple tweak of the VDLPN assumption which allows for a much tighter proof of resistance against linear attack, yet enjoys the same applications as the original VDLPN assumption. The tweak is straightforward: recall that in the original construction, the matrix H is sampled as a concatenation of matrices $H_1 \cdots H_D$, where each H_i is a concatenation of w matrices whose rows are unit vectors of length 2^i. In the security analysis, the authors bound the bias of the $H_i \cdot \mathbf{e}_i$ terms against length-$\Theta(2^i)$ attack vectors. However, the bounds from the new correct analysis of section 4 turned out to be much worse for small constant values of i (to the point that the bounds do not suffice anymore for

very small i, and we have to handle them separately). Here, we suggest replacing $H_1||\cdots||H_{i^*-1}$, where i^* is some fixed small constant (we will pick $i^* = 5$ in our concrete instantiation), by a uniformly random matrix R of appropriate dimensions. That is, H is now of the form $H = [R||H_{i^*}||\cdots||H_D]$. As before, the noise distribution will be identical to the row distribution of H. This means that we will have $H \cdot \mathbf{e} = R \cdot \mathbf{e}_r + \sum_{i=i^*}^{D} H_i \cdot \mathbf{e}_i$, where $\mathbf{e}_r$ is a uniformly random vector.

Let t be the width of R. We show that the $R \cdot \mathbf{e}_r$ term guarantees resistance against all low-weight tests. Then, saying that the distribution $\mathcal{D}_R = \{R \cdot \mathbf{e}_r \ : \mathbf{e}_r \xleftarrow{\$} \mathbb{F}_2^t\}$ has zero bias against all vectors of weight below d is equivalent to saying that $\mathcal{D}_R$ is a d-wise independent distribution. It is a well-known fact that this is equivalent to the following: the dual of the code generated by R, which is a random linear code of dimension $2^D - t$, has minimum distance at least d. Fortunately, the minimum distance of random linear codes is well-known. Let S be a random code of dimension $2^D - t$, and codeword length 2^D. Then,

$$\Pr[S \text{ has minimum distance } < d] \leq 2^{-t - H_2(d/2^D)\cdot 2^D},$$

where $H_2(x) = -x\log x - (1-x)\log(1-x)$ is the binary entropy function. Concretely, suppose that we want to perfectly withstand all linear tests of weight at most $d = 15$, with probability at least $1 - 2^{-\lambda}$, given up to $2^D = 2^{30}$ samples. This means we need to pick t such that $t = H_2(15/2^{30}) * 2^{30} + \lambda$; using $\lambda = 128$, this gives $t = 541$. Hence, picking a uniformly random width-541 matrix guarantees that, with probability at least $1 - 2^{-128}$, we only have to worry about any linear test of weight at least $16 = 2^{i^*-1}$. Note that this variant can be used exactly in the same way as the original VDPLN one, as a building block to construct PCF, as long as we can prove its security against linear tests.

3.2 A New Tight Proof Strategy

For the rest of the analysis, we assume that we start with $i \geq i^* = 5$. The adversary chooses an attack vector $\mathbf{v}$ of hamming weight $l \in [2^{i-1}, 2^i]$. We use the following random variable:

$$Z_{i,l,k} = \left| \mathsf{HW}\left(\bigoplus_{j=1}^{l} X_{j,k}^{(i)} \right) - 2^{i-1} \right|.$$

Unlike the original proof (see Section 2.5), this time we aim at a much more direct strategy. Since we ultimately want to bound the probability that the bias of $\mathcal{O}_{\mathsf{par}}(H)$ is too high, we rewrite this bias directly in terms of the above random variable. For a fixed choice of H, let $\mathcal{O}^i_{\mathsf{par}} = \mathcal{O}^i_{\mathsf{par}}(H)$ be the distribution that samples $\mathbf{e}_i$ (a concatenation of w length-2^i unit vectors) and outputs $H_i \cdot \mathbf{e}_i$. Of course, we have $\mathcal{O}_{\mathsf{par}} = \bigoplus_{i \geq i^*} \mathcal{O}^i_{\mathsf{par}} \oplus \mathcal{O}^R_{\mathsf{par}}$ (where $\mathcal{O}^R_{\mathsf{par}}$ denotes the distribution that samples a uniformly random length-t vector $\mathbf{e}_r$ and outputs $R \cdot \mathbf{e}_r$, where t is a parameter which will be fixed afterwards). Furthermore, for any test vector

$\mathbf{v}$, we have $\mathsf{bias}_{\mathbf{v}}(\mathcal{O}_{\mathsf{par}}) \geq \mathsf{bias}_{\mathbf{v}}(\mathcal{O}^i_{\mathsf{par}})$. We therefore focus on bounding the bias against a test vector $\mathbf{v}$ of $\mathcal{O}^i_{\mathsf{par}}$. We have

$$\mathsf{bias}_{\mathbf{v}}(\mathcal{O}^i_{\mathsf{par}}) = \left| \frac{1}{2} - \Pr[(\mathbf{v}^\top \cdot H_i) \cdot \mathbf{e_i} = 1] \right|.$$

$(\mathbf{v}^\top \cdot H_i) \cdot \mathbf{e_i}$ is the XOR of w independent terms $(\mathbf{v}^\top \cdot H_{i,j}) \cdot \mathbf{e_{i,j}}$ where each $\mathbf{e}_{i,j}$ is a length-2^i unit vector. Therefore, we further decompose $\mathcal{O}^i_{\mathsf{par}}$ as the XOR of w distributions $D_1, \cdots, D_w$ (we drop the parameters i, par, and H for now as we consider a fixed choice of them, to lighten the notations). To bound the bias of $\mathcal{O}^i_{\mathsf{par}}$, we must therefore bound

$$\Pr\left[\bigoplus_{k=1}^{w} D_k = 1\right] = \frac{1}{2}\left(1 - \prod_{k=1}^{w}\left(1 - \frac{R_{i,l,k}}{2^{i-1}}\right)\right),$$

where you get the right hand side by applying the piling-up lemma. Hence, we obtain a direct expression of the bias of $\mathcal{O}^i_{\mathsf{par}}$ in terms of the $Z_{i,l,k}$ random variables:

$$\mathsf{bias}_{\mathbf{v}}(\mathcal{O}^i_{\mathsf{par}}) = \frac{1}{2} \cdot \prod_{k=1}^{w} \frac{Z_{i,l,k}}{2^{i-1}}.$$

Fix any bound B. Then by the above,

$$\Pr[\mathsf{bias}_{\mathbf{v}}(\mathcal{O}^i_{\mathsf{par}}) > B] = \Pr\left[\prod_{k=1}^{w} Z_{i,l,k} > 2^{(i-1)w} \times (2B)\right].$$

Now, the key to bounding the right hand side term is the following observation: independently of the exact behavior of the random variables $Z_{i,l,k}$, constrained on the product $\prod_k Z_{i,l,k}$ being at least $2^{(i-1)w} \cdot (2B)$, the *sum* $\sum_k Z_{i,l,k}$ is minimized when all the terms in the product are equal. This implies that whenever $\prod_{k=1}^{w} Z_{i,l,k} > 2^{(i-1)w} \times (2B)$, it necessarily further holds that

$$\sum_{k=1}^{w} Z_{i,l,k} > w \cdot \left(2^{(i-1)w} \times (2B)\right)^{1/w},$$

which allows to upper bound the probability of the bias being too large by

$$\Pr[\mathsf{bias}_{\mathbf{v}}(\mathcal{O}^i_{\mathsf{par}}) > B] \leq \Pr\left[\sum_{k=1}^{w} Z_{i,l,k} > w \cdot 2^{(i-1)} \cdot c\right],$$

where $c = (2B)^{\frac{1}{w}}$. As in the previous proof, we can now re-introduce the function $\Phi\left(X_{1,1}, \cdots, X_{l,w}\right) = 2^{i-1} \cdot w - \sum_{k=1}^{w} Z_{i,l,k}$:

$$\Pr[\Phi < \mathbb{E}[\phi] - t] = \Pr\left[\sum_{k=1}^{w} Z_{i,l,k} > w \cdot (\mathbb{E}[Z_{i,l}] + 2^i \cdot \zeta)\right].$$

With $t = \zeta \cdot w \cdot 2^i$. Let β be a constant such that $\mathbb{E}[Z_{i,l}] \leq \beta \cdot 2^i$ (In the correction of the original proof, we will show that $\beta = 0.44$ works ; we will actually use a tighter constant here). This gives $c = 2(\beta + \zeta)$. As we did before, we can now apply McDiarmid's inequality 7 to get

$$\Pr\left[\sum_{k=1}^{w} Z_{i,l,k} > w \cdot 2^{i-1} \cdot c\right] < \exp\left(-w\frac{2^{2i-1}}{l} \cdot \zeta^2\right),$$

and obtain the bound

$$\Pr\left(\mathsf{bias}_{\mathbf{v}}(\mathcal{O}^i_{\mathsf{par}}) > \frac{1}{2}(2(\beta+\zeta))^w\right) \leq \exp\left(-w\frac{2^{2i-1}}{l} \cdot \zeta^2\right).$$

While this might be obscured by the many variables involved, this last bound is tremendously tighter than what was achieved with the previous proof. In essence, this is because the previous proof relied on Lemma 5 to bound the bias of the XOR of independent distributions, but the latter introduces some exponential slackness in the *number* of distributions involved. To overcome this slackness, the strategy was to only "count" the distributions that contribute the most to the bias, by identifying *good* distributions, showing that, over the choice of H, a sufficient number of distributions will be good, and applying the lemma only to these good distributions. This guarantees that the slackness is compensated by the contribution of each distribution. If, instead, one tries to apply the lemmas to all distribution, the bound obtain is too loose and does not provide any usable guarantee.

Here, we manage to directly account for the contribution to the bias of *all* distributions, by carefully rewriting the bias formula in terms of the $Z_{i,l,k}$ random variables, and by using a standard "optimization trick" to bound the product of the $Z_{i,l,k}$ in terms of their sum. This turns out to be the key to get back to the function which we can bound with known tools (the function Φ, which is Lipschitz), without paying any slackness in the number of distributions involved.

In the following, we will numerically evaluate the constant β (this is an orthogonal optimization: In the correction of the mistake in section 4, we prove that $\beta \leq 0.44$. Using this value for β would already lead to significant improvements, as we will see) and carefully tune the parameters to find out the smallest value of w for which we can achieve 80-bit security against all test vectors simultaneously, fixing the number of samples to a reasonable bound of $N = 2^{30}$.

3.3 Concrete Parameters

In our bound on the bias, the l in the denominator of the probability is one of the key factors for concrete efficiency. Our previous proof used $l \in [2^{i-1}, 2^i]$. In fact, we cannot expect l to be any smaller: $\mathbb{E}[Z_{i,l}]$ measures how, when one throws l balls at random in 2^i bins, the number of bins which end up containing an odd number of balls diverges from the middle value 2^{i-1}. When we throw less than 2^{i-1} balls in total, this number will of course be bounded away from 2^{i-1};

yet, as simulations reveal, it becomes tightly concentrated around 2^{i-1} as soon as l gets larger. We therefore fix $l \in [2^{i-1}, 2^i]$ and empirically estimate $\mathbb{E}[Z_{i,l}]$. Our script is in Appendix D. Table 1 shows the value of β obtained for different choices of $n = 2^i$ and l. For larger values of n and a fixed $l = l(n)$, note that our estimate value for β barely increase (for $l < n$) or decrease (for $l \geq n$).

Table 1. Estimated value of β for different values of n and l, in a confidence interval of 99% (rounded value ± 0.002)

	$n = 32$	$n = 64$	$n = 512$	$n = 1024$	$n = 2048$
$l = \frac{n}{2}$	0.178	0.181	0.184	0.184	0.184
$l = \frac{3 \cdot n}{4}$	0.111	0.111	0.111	0.111	0.112
$l = n$	0.084	0.073	0.067	0.067	0.067

Let us go back to our bound. For a given vector of Hamming weight l,

$$\Pr\left(\mathsf{bias}_{\mathbf{v}}(\mathcal{O}^i_{\mathsf{par}}) > \frac{1}{2}(2(\beta+\zeta))^w\right) \leq \exp\left(-w\frac{2^{2i-1}}{l}\cdot\zeta^2\right),$$

hence, by a union bound over all vectors of Hamming weight l,

$$\Pr\left(\exists \mathbf{v}, \mathsf{HW}(\mathbf{v}) = l, \mathsf{bias}_{\mathbf{v}}(\mathcal{O}^i_{\mathsf{par}}) > \frac{1}{2}(2(\beta+\zeta))^w\right) \leq \binom{N}{l}\exp\left(-w\frac{2^{2i-1}}{l}\cdot\zeta^2\right)$$

From the above inequality, we numerically look for a w such that, for all l such that $\mathsf{HW}(l) \in [2^{i-1}, 2^i]$, $(2\cdot(\beta+\zeta))^w/2 \leq 2^{-80}$ and

$$\binom{N}{l}\cdot\exp\left(-w\frac{2^{2i-1}}{l}\cdot\zeta^2\right) \leq \exp\left(\ln(N)\cdot l - w\cdot\frac{2^{2i-1}}{l}\cdot\zeta^2\right) \leq 2^{-90}.$$

(The 2^{-90} bound is to anticipate the cost of the union bound.) In the following we set the number of samples $N = 2^D = 2^{30}$, which is a realistic value for target applications. To find a suitable w, we calculate the required w for different values of l. Let us first assume that $l = 2^{i-1}$. The second inequality can be rewritten as $\exp\left(2^{i-1}\cdot\left(\ln(2)\cdot 30 - 2\cdot w\cdot\zeta^2\right)\right) \leq 2^{-90}$. If $w\cdot\zeta^2 \geq 12.35$, then the condition is met. Thus, we can now turn to the other inequality to satisfy; we therefore set w to $\zeta^2/12.35$. Using Table 2, we set $\beta = 0.184$ and numerically solve $(2\cdot(0.184+\zeta))^{12.35/\zeta^2} \leq 2^{-80}$ to guarantee that the bias will be lower than 2^{-80}. This gives $\zeta \leq 0.219$ and $w = 12.35/0.219^2 \approx 257$. At the other end of the interval, setting $l = 2^i$, the second inequality becomes $\exp\left(2^i\cdot\left(\ln(2)\cdot 30 - \frac{1}{2}w\cdot\zeta^2\right)\right) \leq 2^{-90}$.

This time, we get $w\cdot\zeta^2 \geq 45.5$ and set $\beta = 0.084$ using Table 2. Solving $(2\cdot(0.084+\zeta))^{\frac{45.5}{\zeta^2}} \leq 2^{-80}$ gives $\zeta = 0.347$, and eventually $w = 45.5/0.347^2 \approx 380$. Generalizing this method, we numerically extrapolate how the value of w evolves when l varies from 2^{i-1} to 2^i. The calculations show that w is monotonously increasing, leading to an overall choice of $w \approx 380$ as a single parameter that

suffices for the entire range of values. This is a major improvement compared to the previous proof. [6] From here, finishing the proof boils down to two union bounds, giving

$$\Pr(\exists \mathbf{v}, \mathsf{HW}(\mathbf{v}) \in [2^{i-1}, 2^i], \mathsf{bias}_{\mathbf{v}} > 2^{-80}) \leq \sum_{l=2^{i-1}}^{2^i} \binom{N}{l} \exp\left(-w\frac{2^{2i-1}}{l} \cdot \zeta^2\right) \leq 2^{i^*-1} \cdot 2^{-90} = 2^{-86},$$

where the last inequality comes from the fact that $2^{i^*} = 2^5$. For $i > i^*$, the bound in the probability decreases (exponentially) faster than the increase of 2^i, and the result remains valid. Eventually, by a union bound on all i, with $D = 30$, $\Pr(\exists \mathbf{v}, \mathsf{HW}(\mathbf{v}) \geq 2^4, \mathsf{bias}_{\mathbf{v}} > 2^{-80}) \leq (D-8) \cdot 2^{-86} < 2^{-80}$.

Note that here, the computation is done for a fixed value $D = 30$. As D increases, w must also increase to achieve the same security. Concretely, the asymptotic analysis shows that D and w are linearly related: there exists a universal constant c such that setting $w = c \cdot D$ suffices to reach a target security level. Unfortunately, the asymptotic analysis gives a very poor value of c. In table 2, we computed the value of w for other values of D, using the same calculation method. We also provide the ratio $c = w/D$, to give an intuition of what the right constant should be (intuitively, since the analysis gets tighter as D increases due to the "border effects" of low values, the value w/D should converge to the "right" asymptotic constant).

Table 2. Security parameter w and the ratio w/D, for different value of D, computed with our method above.

D	w	$c = w/D$
20	293	14.7
25	336	13.4
30	380	12.7
35	421	12
40	461	11.5

Concrete cost estimations for the pseudorandom correlation function obtained by using the VDLPN parameters of our improved analysis are given in the introduction.

[6] The improvement comes from a better estimation of the β parameter on one hand, but also from the better inherent quality of the new proof. In fact, we can consider the same calculation as before, but with $l = 2^7$ and $\beta = 0.44$. This is a non-computer-optimized, provable value of β using $i^* = 7$. With this value, we get $w \approx 13000$, which is already a big gain from the previous method: this is already several orders of magnitudes better than the previous method, though still not practical.

4 Security of VDLPN against Linear Tests, Revisited

As pointed out, the analysis of [BCG+20a], while the proof strategy seems sound and appropriate, contains some errors which invalidate the proof. Fixing the errors turns out to be quite delicate. Below, we elaborate on the two issues; the first is a minor error, which can be fixed relatively easily, at the cost of changing the (arbitrary) choice of constants (in particular, the 1/5 and 1/10 constants): Claim 1 is incorrect as stated; the error in its analysis stems from a reversed inequality. However, a variant of Claim 1 with different constants can be easily shown to hold; this does not change the spirit of the proof, nor its conclusion. The second error is more delicate to fix, and will be the main focus of this Section.

Main error. The main error appears in the proof of Equation 2. The error is in the analysis of Lemma 17. As sketched in the introduction, after calculating an upper bound on the expectation $\mathbb{E}\left[\mathsf{HW}(\bigoplus_{j=1}^{l} X_{j,k})\right]$, the authors deduce a bound on $\mathbb{E}\left[\left|2^{i-1} - \mathsf{HW}(\bigoplus_{j=1}^{l} X_{j,k})\right|\right]$. However, a bound on $\mathbb{E}[Z]$ does not imply a bound on $\mathbb{E}[|Z-b|]$ in general (and typically when Z is "concentrated away" from b). Up to the choice of the constant 1/5 (the proof actually only requires any constant below 1/2), the lemma remains true; however, proving the lemma fundamentally requires characterizing the shape of the random variable $\mathsf{HW}(\bigoplus_{j=1}^{l} X_{j,k})$. This turns out to be non-trivial.

4.1 Repairing the Proof

In this section, we put forward a corrected detailed analysis of the resistance of VDLPN against linear tests. Our proof fixes the two errors in [BCG+20a], at the cost of achieving worse constants, and being more involved. As before, we study individually the bias induced by the H_i components against vectors of weight close to 2^i. However, for now, we only consider large enough values of i, and assume that $n = 2^i \geq 2^7$. We will handle the missing cases separately, in Appendix C of the Supplementary Material.

Definition 18 (δ-Bad Matrices).
Let $M \in \mathbb{F}_2^{N \times 2^i}$. We say that $M \in \mathsf{Bad}_{\delta,\mathbf{v}}$ with respect to a vector $\mathbf{v} \in \mathbb{F}_{2^N}$ if

$$\mathsf{HW}(\mathbf{v}^\top \cdot M) = R_{l,k} \notin \left[\delta \cdot 2^i, (1-\delta) \cdot 2^i\right].$$

Stated in terms of $Z_{l,k}$, this condition rewrites to $Z_{l,k} \in \left[(1/2-\delta) \cdot 2^i, 2^{i-1}\right]$. We let $\mathsf{Good}_{\delta,\mathbf{v}}$ denote the complement of $\mathsf{Bad}_{\delta,\mathbf{v}}$ Given vector $\mathbf{v}$, we also denote $B_{\delta,\mathbf{v}} = \#\mathsf{Bad}_{\delta,\mathbf{v}} = N_{\mathbf{v}}(M_1, \cdots, M_w)$ and $G_{\delta,\mathbf{v}} = \#\mathsf{Good}_{\delta,\mathbf{v}} = w - B_{\delta,\mathbf{v}}$.

The proof. We now prove Theorem 14. Let $\mathcal{O}^i_{\mathsf{par}}(H)$ be the distribution induced by sampling $\mathbf{e}_i$ (as a concatenation of w length-2^i vectors) and outputting $H_i \cdot \mathbf{e}_i$. A sample from $\mathcal{O}^i_{\mathsf{par}}(H)$ can be further decomposed as $\bigoplus_{j \leq w} H_{i,j} \cdot \mathbf{e}_{i,j}$ where the $\mathbf{e}_{i,j}$ are unit vectors. Let D_i denote the distribution of $H_{i,j} \cdot \mathbf{e}_{i,j}$ (these terms are w samples from the same distribution).Let α be a constant. Then,

Lemma 19. *If $B_{\delta,\mathbf{v}} \leq \alpha \cdot w$, then*

$$\mathsf{bias}\left(\bigoplus_{i=1}^{w} D_i\right) \leq \frac{1}{2} \cdot ((1-2\delta)^{(1-\alpha)})^w.$$

Proof. By the piling-up lemma (Lemma 6),

$$\mathsf{bias}\left(\bigoplus_{i=1}^{w} D_i\right) \leq 2^{(1-\alpha)w-1} \cdot \left(\frac{1}{2}-\delta\right)^{(1-\alpha)\cdot w} \leq \frac{1}{2} \cdot ((1-2\delta)^{(1-\alpha)})^w \qquad \square$$

Lemma 19 provides an upper bound of the bias, which depends on the number of good matrices and their quality. We now show that the condition $B_{\delta,\mathbf{v}} \leq \alpha \cdot w$ holds with very high probability:

Lemma 20. *For any $\mathbf{v} \in S_{i,N}$, there is a constant C such that*

$$\Pr\left[B_{\delta,\mathbf{v}} > \alpha \cdot w\right] \leq 2^{-C \cdot 2^i \cdot w}.$$

Proof. As in the original proof, we introduce the function Φ:

$$\begin{aligned}\Phi\left(X_{1,1}, \cdots, X_{l,w}\right) &= \sum_{k=1}^{w}\left(2^{i-1} - \left|\mathsf{HW}\left(\bigoplus_{j=1}^{l} X_{j,k}\right) - 2^{i-1}\right|\right) \\ &= 2^{i-1} \cdot w - \sum_{k=1}^{w} Z_{l,k}.\end{aligned}$$

We want to bound the probability of large bias by a bound on Φ. This is where the first error appeared in the previous proof.

Lemma 21 (Correction of the first error).

$$\Pr\left[B_{\delta,\mathbf{v}} \geq \alpha \cdot w\right] \leq \Pr\left[\Phi(X_{1,1}, \cdots, X_{l,w}) < \gamma \cdot w \cdot 2^i\right],$$

with $\gamma = \frac{1}{2} - \alpha(\frac{1}{2} - \delta)$.

Due to lack of space, the proof of the above lemma will appear in Appendix B.It remains now to find an upper bound on the right hand side probability. As in the original proof, we used the bounded difference inequality. Since Φ is 2-Lipschitz, (this was proved in the original proof),

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

We finally want to prove a lower bound on $\mathbb{E}[\phi(X_{1,1}, \cdots, X_{l,w})]$. Recall that $\Phi\left(X_{1,1}, \cdots, X_{l,w}\right) = 2^{i-1} \cdot w - \sum_{k=1}^{w} Z_{l,k}$, so this reduces to bounding $\mathbb{E}[Z_{l,k}]$. Our main contribution in this analysis is the proof of the following lemma:

Lemma 22 (Correction of the second error). *For all $n \in \mathbb{N}$, there exists $\beta < 1/2$ such that $\mathbb{E}[\mathbb{Z}_{l,k}] < \beta \cdot n$.*

Proof (Sketch). We first provide a high level overview, and due to lack of space, we defer the full proof of Lemma 22 to Appendix B of the Supplementary Material.. The proof consists in finding an upper bound on both $\Pr[R_{l,k} \geq p \cdot n]$ and $\Pr[R_{l,k} \leq (1-p) \cdot n]$ for $p \in [\frac{1}{2}, 1]$ and to use it to find the one on $\mathbb{E}[Z_{l,k}] = \sum_{j=0}^{2^{i-1}-1} \Pr\left(\left|R_{l,k} - 2^{i-1}\right| > j\right)$.

Lemma 23. *Let $n = 2^i > 2^7$, $l \in [2^{i-1}, 2^i]$ and $\mu = (1 - \frac{1}{n})^l$. There exists $0.5 \leq p \leq 1$ such that with $\theta = \frac{pn - l/2}{\mu} - 1$, it holds that*

$$\max\left(\Pr\left[R_{l,k} \geq pn\right], \Pr\left[R_{l,k} \leq (1-p)\, n\right]\right) \leq 2 \exp\left(-\frac{\theta^2 \mu^2 (n - \frac{1}{2})}{n^2 - \mu^2}\right).$$

To prove this lemma, we use the Occupancy Bound for balls and bins from Lemma 8. The occupancy bound is about the proportion of empty bins, but can shrewdly be transformed to bring it back to our specific problem which focuses on parity in bins. This concludes the sketch. □

The end of the proof is the same as in the original proof, up to handling separately the case of small i's. The total number of vectors $\mathbf{v} \in S_{i,N}$ can be bounded by

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

Hence, choosing constant such that $Cw/2 > D$, and setting $a = C/2$, by a union bound, we have

$$\Pr\left[\exists \mathbf{v} \in S_{i,N}, B_{\delta,\mathbf{v}} \geq \alpha \cdot w\right] \leq 2^{D \cdot 2^i} \cdot 2^{-C \cdot 2^i \cdot w} \leq 2^{-a \cdot w}.$$

We eventually use a union bound again on all values of $i \leq D$:

$$\Pr\left[\exists i \leq D, \mathbf{v} \in S_{i,N}, B_{\delta,\mathbf{v}} \geq \alpha \cdot w\right] \leq D \cdot 2^{-a \cdot w}.$$

which, using Lemma 19, rewrites to

$$\Pr\left[\exists i \leq D, \mathbf{v} \in S_{i,N}, \mathsf{bias}_{\mathbf{v}}\left(\bigoplus_{j=1}^{w} D_j\right) \geq \frac{1}{2} \cdot ((1-2\delta)^{(1-\alpha)})^w\right] \leq D \cdot 2^{-a \cdot w}.$$

The argument for small values of i is completely different. In essence, we show that the first block of H, H_1, does already suffice to withstand all even-weight test vectors. Then, with a "brute-force" union bound, we show that the second

block H_2 allows to withstand all tests of *odd* weight, provided that $w = \Omega(2^i \cdot D)$. When i is a constant, this is already captured by the requirement that $w \geq \Gamma \cdot D$ for a suitable constant Γ, which suffices to handle all remaining corner cases. Refer to Appendix C for full details.

Acknowledgements. The first author acknowledges the support of the French Agence Nationale de la Recherche (ANR), under grant ANR-20-CE39-0001 (project SCENE), and of the France 2030 ANR Project ANR-22-PECY-003 SecureCompute. The second author was supported by the DIM RFSI grant LICENCED.

References

AJ01. Abdulrahman Al Jabri. A statistical decoding algorithm for general linear block codes. 2001.

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

BCG+17. Elette Boyle, Geoffroy Couteau, Niv Gilboa, Yuval Ishai, and Michele Orrù. Homomorphic secret sharing: Optimizations and applications. In Bhavani M. Thuraisingham, David Evans, Tal Malkin, and Dongyan Xu, editors, *ACM CCS 2017*, pages 2105–2122. ACM Press, October / November 2017.

BCG+19a. Elette Boyle, Geoffroy Couteau, Niv Gilboa, Yuval Ishai, Lisa Kohl, Peter Rindal, and Peter Scholl. Efficient two-round OT extension and silent non-interactive secure computation. In Lorenzo Cavallaro, Johannes Kinder, XiaoFeng Wang, and Jonathan Katz, editors, *ACM CCS 2019*, pages 291–308. ACM Press, November 2019.

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

BCG+20a. Elette Boyle, Geoffroy Couteau, Niv Gilboa, Yuval Ishai, Lisa Kohl, and Peter Scholl. Correlated pseudorandom functions from variable-density LPN. In *61st FOCS*, pages 1069–1080. IEEE Computer Society Press, November 2020.

BCG+20b. Elette Boyle, Geoffroy Couteau, Niv Gilboa, Yuval Ishai, Lisa Kohl, and Peter Scholl. Efficient pseudorandom correlation generators from ring-LPN. In Daniele Micciancio and Thomas Ristenpart, editors, *CRYPTO 2020, Part II*, volume 12171 of *LNCS*, pages 387–416. Springer, Heidelberg, August 2020.

BCGI18. Elette Boyle, Geoffroy Couteau, Niv Gilboa, and Yuval Ishai. Compressing vector OLE. In David Lie, Mohammad Mannan, Michael Backes, and XiaoFeng Wang, editors, *ACM CCS 2018*, pages 896–912. ACM Press, October 2018.

BGI14. Elette Boyle, Shafi Goldwasser, and Ioana Ivan. Functional signatures and pseudorandom functions. In Hugo Krawczyk, editor, *PKC 2014*, volume 8383 of *LNCS*, pages 501–519. Springer, Heidelberg, March 2014.

BGI15. Elette Boyle, Niv Gilboa, and Yuval Ishai. Function secret sharing. In Elisabeth Oswald and Marc Fischlin, editors, *EUROCRYPT 2015, Part II*, volume 9057 of *LNCS*, pages 337–367. Springer, Heidelberg, April 2015.

BGI16. Elette Boyle, Niv Gilboa, and Yuval Ishai. Function secret sharing: Improvements and extensions. In Edgar R. Weippl, Stefan Katzenbeisser, Christopher Kruegel, Andrew C. Myers, and Shai Halevi, editors, *ACM CCS 2016*, pages 1292–1303. ACM Press, October 2016.

BJMM12. Anja Becker, Antoine Joux, Alexander May, and Alexander Meurer. Decoding random binary linear codes in $2^{n/20}$: How $1 + 1 = 0$ improves information set decoding. In David Pointcheval and Thomas Johansson, editors, *EUROCRYPT 2012*, volume 7237 of *LNCS*, pages 520–536. Springer, Heidelberg, April 2012.

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

BLP11. Daniel J. Bernstein, Tanja Lange, and Christiane Peters. Smaller decoding exponents: Ball-collision decoding. In Phillip Rogaway, editor, *CRYPTO 2011*, volume 6841 of *LNCS*, pages 743–760. Springer, Heidelberg, August 2011.

BM97. Mihir Bellare and Daniele Micciancio. A new paradigm for collision-free hashing: Incrementality at reduced cost. In Walter Fumy, editor, *EUROCRYPT'97*, volume 1233 of *LNCS*, pages 163–192. Springer, Heidelberg, May 1997.

BM18. Leif Both and Alexander May. Decoding linear codes with high error rate and its impact for LPN security. In Tanja Lange and Rainer Steinwandt, editors, *Post-Quantum Cryptography - 9th International Conference, PQCrypto 2018*, pages 25–46. Springer, Heidelberg, 2018.

BTV16. Sonia Bogos, Florian Tramer, and Serge Vaudenay. On solving lpn using bkw and variants. 2016.

BV16. Sonia Bogos and Serge Vaudenay. Optimization of LPN solving algorithms. In Jung Hee Cheon and Tsuyoshi Takagi, editors, *ASIACRYPT 2016, Part I*, volume 10031 of *LNCS*, pages 703–728. Springer, Heidelberg, December 2016.

BW13. Dan Boneh and Brent Waters. Constrained pseudorandom functions and their applications. In Kazue Sako and Palash Sarkar, editors, *ASIACRYPT 2013, Part II*, volume 8270 of *LNCS*, pages 280–300. Springer, Heidelberg, December 2013.

CRR21. Geoffroy Couteau, Peter Rindal, and Srinivasan Raghuraman. Silver: Silent VOLE and oblivious transfer from hardness of decoding structured LDPC codes. LNCS, pages 502–534. Springer, Heidelberg, 2021.

DAT17. Thomas Debris-Alazard and Jean-Pierre Tillich. Statistical decoding. 2017.

DHRW16. Yevgeniy Dodis, Shai Halevi, Ron D. Rothblum, and Daniel Wichs. Spooky encryption and its applications. In Matthew Robshaw and Jonathan Katz, editors, *CRYPTO 2016, Part III*, volume 9816 of *LNCS*, pages 93–122. Springer, Heidelberg, August 2016.

DPSZ12. Ivan Damgård, Valerio Pastro, Nigel P. Smart, and Sarah Zakarias. Multiparty computation from somewhat homomorphic encryption. In Reihaneh Safavi-Naini and Ran Canetti, editors, *CRYPTO 2012*, volume 7417 of *LNCS*, pages 643–662. Springer, Heidelberg, August 2012.

EKM17. Andre Esser, Robert Kübler, and Alexander May. LPN decoded. In Jonathan Katz and Hovav Shacham, editors, *CRYPTO 2017, Part II*, volume 10402 of *LNCS*, pages 486–514. Springer, Heidelberg, August 2017.

FKI06. Marc PC Fossorier, Kazukuni Kobara, and Hideki Imai. Modeling bit flipping decoding based on nonorthogonal check sums with application to iterative decoding attack of mceliece cryptosystem. 2006.

FS09. Matthieu Finiasz and Nicolas Sendrier. Security bounds for the design of code-based cryptosystems. In Mitsuru Matsui, editor, *ASIACRYPT 2009*, volume 5912 of *LNCS*, pages 88–105. Springer, Heidelberg, December 2009.

GGM86. Oded Goldreich, Shafi Goldwasser, and Silvio Micali. How to construct random functions. *Journal of the ACM*, 33(4):792–807, October 1986.

GI14. Niv Gilboa and Yuval Ishai. Distributed point functions and their applications. In Phong Q. Nguyen and Elisabeth Oswald, editors, *EUROCRYPT 2014*, volume 8441 of *LNCS*, pages 640–658. Springer, Heidelberg, May 2014.

GJL20. Qian Guo, Thomas Johansson, and Carl Löndahl. Solving LPN using covering codes. *Journal of Cryptology*, 33(1):1–33, January 2020.

Kir11. Paul Kirchner. Improved generalized birthday attack. Cryptology ePrint Archive, Report 2011/377, 2011. `https://eprint.iacr.org/2011/377`.

KMPS94. Anil Kamath, Rajeev Motwani, Krishna V. Palem, and Paul G. Spirakis. Tail bounds for occupancy and the satisfiability threshold conjecture. In *35th FOCS*, pages 592–603. IEEE Computer Society Press, November 1994.

KPTZ13. Aggelos Kiayias, Stavros Papadopoulos, Nikos Triandopoulos, and Thomas Zacharias. Delegatable pseudorandom functions and applications. In Ahmad-Reza Sadeghi, Virgil D. Gligor, and Moti Yung, editors, *ACM CCS 2013*, pages 669–684. ACM Press, November 2013.

LF06. Éric Levieil and Pierre-Alain Fouque. An improved LPN algorithm. In Roberto De Prisco and Moti Yung, editors, *SCN 06*, volume 4116 of *LNCS*, pages 348–359. Springer, Heidelberg, September 2006.

Lyu05. Vadim Lyubashevsky. The parity problem in the presence of noise, decoding random linear codes, and the subset sum problem. 2005.

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

MMT11. Alexander May, Alexander Meurer, and Enrico Thomae. Decoding random linear codes in $\tilde{\mathcal{O}}(2^{0.054n})$. In Dong Hoon Lee and Xiaoyun Wang, editors, *ASIACRYPT 2011*, volume 7073 of *LNCS*, pages 107–124. Springer, Heidelberg, December 2011.

MO15. Alexander May and Ilya Ozerov. On computing nearest neighbors with applications to decoding of binary linear codes. In Elisabeth Oswald and Marc Fischlin, editors, *EUROCRYPT 2015, Part I*, volume 9056 of *LNCS*, pages 203–228. Springer, Heidelberg, April 2015.

MSY21. Jean-Pierre Münch, Thomas Schneider, and Hossein Yalame. Vasa: Vector aes instructions for security applications. In *Annual Computer Security Applications Conference*, pages 131–145, 2021.

OSY21. Claudio Orlandi, Peter Scholl, and Sophia Yakoubov. The rise of paillier: Homomorphic secret sharing and public-key silent OT. LNCS, pages 678–708. Springer, Heidelberg, 2021.

Ove06. Raphael Overbeck. Statistical decoding revisited. In Lynn Margaret Batten and Reihaneh Safavi-Naini, editors, *ACISP 06*, volume 4058 of *LNCS*, pages 283–294. Springer, Heidelberg, July 2006.

Pra62. Eugene Prange. The use of information sets in decoding cyclic codes. 1962.

Saa07. Markku-Juhani Olavi Saarinen. Linearization attacks against syndrome based hashes. In K. Srinathan, C. Pandu Rangan, and Moti Yung, editors, *INDOCRYPT 2007*, volume 4859 of *LNCS*, pages 1–9. Springer, Heidelberg, December 2007.

SGRR19. Phillipp Schoppmann, Adrià Gascón, Leonie Reichert, and Mariana Raykova. Distributed vector-OLE: Improved constructions and implementation. In Lorenzo Cavallaro, Johannes Kinder, XiaoFeng Wang, and Jonathan Katz, editors, *ACM CCS 2019*, pages 1055–1072. ACM Press, November 2019.

Shp09. Amir Shpilka. Constructions of low-degree and error-correcting ε-biased generators. 2009.

Ste88. Jacques Stern. A method for finding codewords of small weight. 1988.

Wag02. David Wagner. A generalized birthday problem. In Moti Yung, editor, *CRYPTO 2002*, volume 2442 of *LNCS*, pages 288–303. Springer, Heidelberg, August 2002.

Zic17. Lior Zichron. Locally computable arithmetic pseudorandom generators, 2017.

ZJW16. Bin Zhang, Lin Jiao, and Mingsheng Wang. Faster algorithms for solving LPN. In Marc Fischlin and Jean-Sébastien Coron, editors, *EUROCRYPT 2016, Part I*, volume 9665 of *LNCS*, pages 168–195. Springer, Heidelberg, May 2016.

Supplementary Material

A Preliminaries on Pseudorandom Correlation Functions

The definitions of this section are taken almost verbatim from [BCG+20a].

Definition 24 (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* $\mathsf{RSample}$ *that takes as input $1^\lambda, \sigma \in \{0,1\}$ 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 $\sigma \in \{0,1\}$ the following distributions are statistically close:*

$$\{(y_0, y_1) \mid (y_0, y_1) \xleftarrow{\$} \mathcal{Y}(1^\lambda)\} \quad \textit{and}$$
$$\{(y_0, y_1) \mid (y'_0, y'_1) \xleftarrow{\$} \mathcal{Y}(1^\lambda), y_\sigma \leftarrow y'_\sigma, y_{1-\sigma} \leftarrow \mathsf{RSample}(1^\lambda, \sigma, y_\sigma)\}.$$

It can also be useful to consider correlations *across* different inputs, as e.g. in vector oblivious linear evaluation (VOLE). This is captured by the notion of reverse sampleable correlation with setup, which allows all algorithms to depend on a fixed global secret, ensuring consistency across different invocations; we omit this formal definition for conciseness and refer the reader to [BCG+20a] for more details.

Definition 25 (Pseudorandom correlation function (PCF)). *Let $\mathcal{Y}$ be a reverse-sampleable correlation with output length functions $\ell_0(\lambda), \ell_1(\lambda)$ and let $\lambda \leq n(\lambda) \leq \mathsf{poly}(\lambda)$ be an input length function. Let $(\mathsf{PCF.Setup}, \mathsf{PCF.Eval})$ be a pair of algorithms with the following syntax:*

- $\mathsf{PCF.Setup}(1^\lambda)$ *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 $(\mathsf{PCF.Setup}, \mathsf{PCF.Eval})$ *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*

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

[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.

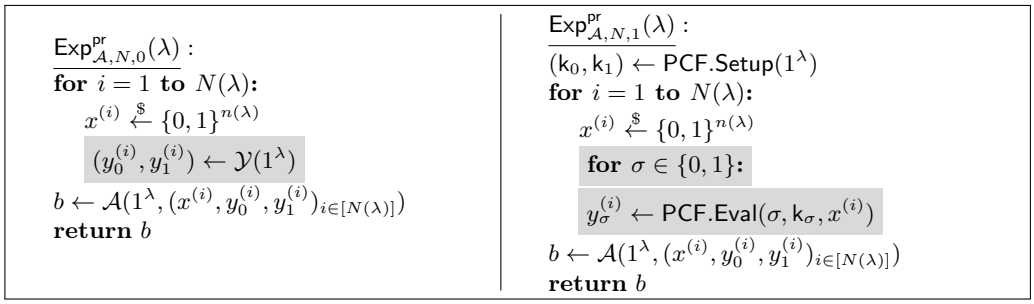

Fig. 1. Pseudorandom $\mathcal{Y}$-correlated outputs of a PCF.

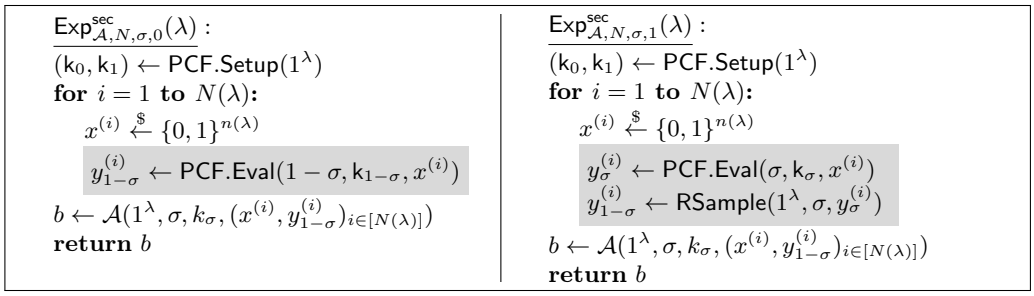

Fig. 2. Security of a PCF. Here, $\mathsf{RSample}$ is the algorithm for reverse sampling $\mathcal{Y}$ as in Definition 24.

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*

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

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 2 (again, with $N(\lambda)$ samples).

We say that $(\mathsf{PCF.Setup}, \mathsf{PCF.Eval})$ *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.*

The above definition captures a notion of *weak* pseudorandom function, where security is only required to hold given random adversarial queries. As for PRFs, one can also strengthen the definition to strong PCFs, which allow arbitrary adversarial queries; a formal definition is given in [BCG+20a]. As shown in [BCG+20a], any weak PCF can be turned into a strong PCF in the random oracle model, by hashing the input before feeding it to the function.

B Missing Proofs

In this section we present the proofs of the lemmas stated in section 4.1. We start with Lemma 21:

Proof. We assume that $B_{\delta,\mathbf{v}} \geq \alpha \cdot w$. This translates to a lower bound of the sum of the $Z_{l,k}$:

$$\sum_{k=1}^{w} Z_{l,k} \geq \alpha \cdot w \cdot \left(\frac{1}{2} - \delta\right) \cdot 2^i,$$

and finally an upper bound on Φ.

$$\Phi\left(X_{1,1}, \cdots, X_{l,w}\right) \leq 2^{i-1} \cdot w - \alpha \cdot w \cdot \left(\frac{1}{2} - \delta\right) \cdot 2^i = 2^i \cdot w \cdot \left(\frac{1}{2} - \alpha\left(\frac{1}{2} - \delta\right)\right).$$

Setting $\gamma = \frac{1}{2} - \alpha(\frac{1}{2} - \delta)$ we get $\Phi(X_{1,1}, \cdots, X_{l,w}) < \gamma \cdot w \cdot 2^i$, which proves that

$$\Pr\left[B_\delta \geq \alpha \cdot w\right] \leq \Pr\left[\Phi(X_{1,1}, \cdots, X_{l,w}) < \gamma \cdot w \cdot 2^i\right].$$

□

We continue with the proof of Lemma 22.

Proof. We first rewrite $\mathbb{E}[Z_{l,k}]$ using the standard fact that $\mathbb{E}[Z] = \sum_j \Pr[Z > j]$:

$$\begin{aligned}\mathbb{E}\left[Z_{l,k}\right] = \mathbb{E}\left[\left|R_{l,k} - 2^{i-1}\right|\right] &= \sum_{j=0}^{2^{i-1}-1} \Pr\left(\left|R_{l,k} - 2^{i-1}\right| > j\right) \\ &= \sum_{j=0}^{2^{i-1}-1} \Pr(R_{l,k} \geq j + 1 + 2^{i-1}) + \sum_{j=0}^{2^{i-1}-1} \Pr(R_{l,k} \leq 2^{i-1} - j - 1).\end{aligned}$$

While we can bound $\Pr\left[R_{l,k} \geq j + 1 + 2^{i-1}\right] + \Pr\left[R_{l,k} \leq 2^{i-1} - j - 1\right]$ by 1 (for every j the two events are disjoint), this only proves that $\mathbb{E}\left[Z_{l,k}\right] \leq 0.5 \cdot n$. Therefore, we are looking for better bounds on these two probabilities. Both bounds come from the Lemma 8; we prove each of them separately below.

Lemma 26. *Let $\frac{1}{2} < p < 1$. Let θ such that $(1-p) \cdot n = \mu \cdot (1-\theta)$. Then,*

$$\Pr[R_{l,k} \geq pn] \leq 2\exp\left(-\frac{\theta^2\mu^2(n - \frac{1}{2})}{n^2 - \mu^2}\right).$$

Proof. Let E be the random variable equal to the number of empty bins. Remark that when $E > x$ then $R_{l,k} < 2^i - x$. Then, it appears necessarily that

$$\Pr[R_{l,k} \geq pn] \leq \Pr[E \leq (1-p)n].$$

We can then use lemma 8, and establish the following

$$\Pr[R_{l,k} \geq pn] \leq 2\exp\left(-\frac{\theta^2\mu^2(n-\frac{1}{2})}{n^2-\mu^2}\right),$$

where we chose θ is chosen such that $(1-p)n = \mu \cdot (1-\theta)$. This proves the first bound we need. □

Let's focus now on the second bound.

Lemma 27. *Let p such that $\frac{1}{2} < p < 1$. Let θ such that $(1-p)\cdot n = n - \mu(\theta+1) - \frac{l}{2}$. Then*

$$\Pr[R_{l,k} \leq (1-p)n] \leq 2\exp\left(-\frac{\theta^2\mu^2(n-\frac{1}{2})}{n^2-\mu^2}\right).$$

Proof. First, we state a simple but interesting result:

Lemma 28. *Let U be the number of bins that contain exactly one ball. Then*

$$U/2 \geq n - E - \frac{l}{2}.$$

Proof. If there are E bins empty, and U bins with one ball only, then the remaining $l-U$ balls are contained in at most $(l-U)/2$ bins, and so

$$E + U + (l-U)/2 \geq \sum_i \text{number of bins containing i balls} = n.$$

and thus, $U/2 \geq n - E - \frac{l}{2}$. □

From there using once again lemma 8

$$\Pr[E \geq \mu(\theta+1)] \leq 2\exp\left(-\frac{\theta^2\mu^2(n-\frac{1}{2})}{n^2-\mu^2}\right)$$

$$\Pr[n - E - \frac{l}{2} \leq n - \mu(\theta+1) - \frac{l}{2}] \leq 2\exp\left(-\frac{\theta^2\mu^2(n-\frac{1}{2})}{n^2-\mu^2}\right).$$

From there we obtain

$$\Pr[U/2 \leq n - \mu(\theta+1) - \frac{l}{2}] \leq 2\exp\left(-\frac{\theta^2\mu^2(n-\frac{1}{2})}{n^2-\mu^2}\right),$$

because $n - E - l/2 \leq U/2$. Finally from $U/2 \leq R_{l,k}$ we get the desired bound

$$\Pr[R_{l,k} \leq n - \mu(\theta+1) - \frac{l}{2}] \leq 2\exp\left(-\frac{\theta^2\mu^2(n-\frac{1}{2})}{n^2-\mu^2}\right)$$

$$\Pr[R_{l,k} \leq (1-p)n] \leq 2\exp\left(-\frac{\theta^2\mu^2(n-\frac{1}{2})}{n^2-\mu^2}\right),$$

by choosing θ such that $(1-p)\cdot n = n - \mu(\theta+1) - \frac{l}{2}$
This proves the second bound we need. □

Now we have upper bounds for $\Pr[R_{l,k} \geq pn]$ and $\Pr[R_{l,k} \leq (1-p)n]$ for $1/2 < p < 1$. It remains to carefully choose *when* to start applying these bounds, i.e., when these bounds become better than the naive $\Pr[R_{l,k} \geq pn] + \Pr[R_{l,k} \leq (1-p)n] \leq 1$.

Thus we want to find the lowest n for which there exists a proportion $1/2 \leq p \leq 1$ such that the following equation stands

$$\Pr[R_{l,k} \geq pn] + \Pr[R_{l,k} \leq (1-p)n] < 1. \tag{3}$$

Remark that we have the same formula for both bounds, but with different values for θ. Let us call them θ_1 and θ_2. For the first one (lemma 26) we we should have θ_1 such that $(1-p)\cdot n = \mu \cdot (1-\theta_1)$; for the second one (lemma 27) should have a θ_2 such that $(1-p)\cdot n = n - \mu(\theta_2+1) - \frac{l}{2}$

We remark that $\theta_1 > \theta_2$, thus the value of n we are looking for is coming the second inequality.

The smallest n such that there exists $1/2 < p < 1$ verifying $\Pr[R_{l,k} \leq (1-p)n] < 1$ is $n = 2^7$. Then, using Table 3 we can conclude that $n = 2^7$ is enough to find a proportion p, as we wanted.

Table 3. Rounded value of $2\exp\left(-\frac{\theta^2\mu^2(n-\frac{1}{2})}{n^2-\mu^2}\right)$; for the two different θ, with n = 128 and l = n.

$1-p$	0.92	0.94	0.96	0.98
$\theta = \theta_1$	5×10^{-5}	2×10^{-6}	3×10^{-7}	4×10^{-8}
$\theta = \theta_2$	1.31	0.90	0.55	0.3

From these calculations we can also show that $\beta < 0.47$, when $n > 2^7$.

At this stage, we have shown that $\mathbb{E}[\phi(X_{1,1}, \cdots, X_{l,w})] \geq 2^i \cdot w \cdot (1/2 - \beta)$, for some positive constant $\beta < 0.5$. Therefore,

$$\Pr\left[\phi(X_{1,1}, \cdots, X_{l,w}) \leq 2^i \cdot w \cdot (1/2-\beta) - t\right] \leq \exp\left(-\frac{t^2}{2lw}\right).$$

Let us define $\zeta = t/(2^i \cdot w)$. Then, the condition $2^i \cdot w \cdot (\frac{1}{2} - \beta) - t = 2^i \cdot w \cdot \gamma$ rewrites to $\gamma = (1/2-\beta) - \zeta = 1/2 - \alpha(1/2-\delta)$. Let us pick a concrete choice of value for $n \geq 2^7$: set $\alpha = 48/49$, $\delta = 1/100$. This gives us a value for $\gamma = 0.02$. As soon as $n \geq 2^7$, we know that $\beta \leq 0.47$, and thus $\zeta \geq 0.01$. Hence, there exists a constant C such that

$$\Pr\left[B_{\delta,\mathbf{v}} \geq \alpha \cdot w\right] \leq 2^{-C \cdot 2^i \cdot w}.$$

This concludes the proof of Lemma 20. □

C Handling the Corner Cases

In the proof, we have made two assumptions: $l \in [2^{i-1}, 2^i]$, and $n = 2^i \geq 2^7$ (recall that l is the Hamming weight of the test vector). Therefore, there is some corner cases that are not covered by the proof, the cases when the adversary attempts an attack with a vector of hamming weight $l \in [1, 63]$.

Case 1: l is odd. We focus on the first submatrix of our matrix H. The matrix has the following shape:

$$H_1 = \begin{bmatrix} & & \overbrace{\phantom{u^1_{1,w}}}^{2 \text{ columns}} \\ u^1_{1,1} & \cdots & u^1_{1,w} \\ \vdots & \vdots & \vdots \\ u^1_{N,1} & \cdots & u^1_{N,w} \end{bmatrix}.$$

where $(u^1_{k,j})_{1 \leq k \leq N, 1 \leq j \leq w}$ are unit vector over F_2^2. Each row uniquely corresponds to a uniform w-bit vector (we say that it *encodes* this vector), where each bit indicates the position of the 1 in the 2-bit vector $u^1_{k,j}$ (0 being left, and 1 being right). We denote the w-bit string encoded by the i-th row by x_i. Recall that $\mathbf{e}_1$ is distributed as a row of H_1. We let K denote the string obtained by flipping all bits in the w-bit string that encodes $\mathbf{e}_1$. Observe that the inner product between the i-th row of H_1 and $\mathbf{e}_1$ is equal to $\bigoplus_{j=1}^{w}(x_{i,j} \oplus K_j)$. Given $\mathbf{v}$, the vector $\mathbf{v} \cdot H_1$ is the XOR of the l rows of H_1 corresponding to nonzero entries in $\mathbf{v}$. Therefore, whenever l is odd, the value $\mathbf{v} \cdot H_1 \cdot \mathbf{e}_1$ is of the form $f(H_1) \oplus (\bigoplus_{j=1}^{w} K_j)$, where f is some appropriate function. That is, value of $\mathbf{v} \cdot H_1 \cdot \mathbf{e}_1$ is masked by a uniformly random bit equal to $(\bigoplus_{j=1}^{w} K_j)$: it is therefore perfectly unbiased.

Case 2: l is even. Let now focus on the second submatrix H_2 of our matrix H. It has the following shape:

$$H_2 = \begin{bmatrix} & & \overbrace{\phantom{u^2_{1,w}}}^{4 \text{ columns}} \\ u^2_{1,1} & \cdots & u^2_{1,w} \\ \vdots & \vdots & \vdots \\ u^2_{N,1} & \cdots & u^2_{N,w} \end{bmatrix}.$$

where $(u^2_{k,j})_{1 \leq k \leq N, 1 \leq j \leq w}$ are unit vector over F_2^4. This time, each row can be seen as encoding a *pair* of w-bits vector: the two bits at position i in both vectors encode together the position of the 1 in the unit length-four vector $u^2_{k,j}$, in binary. For the i-th line, let us denote these vectors as $\mathbf{x_{i,0}}$ and $\mathbf{x_{i,1}}$. For the noise vector $\mathbf{e}_1$, who has the same structure, we also define two vectors $\mathbf{K_0}$ and $\mathbf{K_1}$, as before by flipping all bits of the two vectors encoded by $\mathbf{e}_2$. Now, given a test vector $\mathbf{v}$ with even Hamming weight l, the value $\mathbf{v} \cdot H_2 \cdot \mathbf{e}_2$ is equal to

$\bigoplus_{i\in S}(\mathbf{x_{i,0}} \oplus \mathbf{K_0}) \cdot (\mathbf{x_{i,1}} \oplus \mathbf{K_1})$, where S is the subset of nonzero entries in $\mathbf{v}$. Let $g(\mathbf{x_0}, \mathbf{x_1}) = \bigoplus_{j=1}^{w} x_{i,0,j} \cdot x_{i,1,j}$. With this notation, we can rewrite $\mathbf{v} \cdot H_2 \cdot \mathbf{e}_2$ as

$$\bigoplus_{i\in S} g(\mathbf{x_{i,0}}, \mathbf{x_{i,1}}) \oplus \left\langle \bigoplus_{i\in S} \mathbf{x_{i,0}}, \mathbf{K_1} \right\rangle \oplus \left\langle \bigoplus_{i\in S} \mathbf{x_{i,1}}, \mathbf{K_0} \right\rangle.$$

The leftmost term is independent of the secret noise vector. The above equation implies that if $\bigoplus_{i\in S} \mathbf{x_{i,0}}$ is not the all zero vector, then the above value is additively masked by one of the entries in $\mathbf{K_1}$, which is a uniformly random bit; hence, it is uniformly distributed (the above is a sufficient condition; the same condition with respect to $\mathbf{x_{i,1}}$ and $\mathbf{K_0}$ would also suffice). Furthermore, we can bound the probability that $\bigoplus_{i\in S} \mathbf{x_{i,0}} = \mathbf{0}$: let us call E_0 this event. For any fixed choice of the size-l subset S, the probability that $\bigoplus_{i\in S} \mathbf{x_{i,0}} = \mathbf{0}$ holds over a random choice of the vectors $\mathbf{x_{i,0}}$ is exactly 2^{-w}. By a straightforward union bound over all subsets S of size l,

$$\Pr[E_0] \leq 2^{-w} \cdot \binom{N}{l} \leq 2^{-w} \cdot N^l \leq 2^{l \cdot D - w}.$$

Where $N = 2^D$ is the number of rows in H. Whenever l is a constant (recall that we assume here $l \leq 63$), the above probability is bounded by $2^{-\alpha_0 w}$ as soon as $w > \alpha_1 D$ for an appropriate choice of the constants α_0, α_1 (a quick calculation shows that these constants are *much* better than the ones involved in the case of large l, hence the final constants involved in our theorem remain the same as the constants achieved in the previous proof). This concludes the analysis of the corner cases.

D Script used for simulations

Hereinafter the script written in python used to obtain simulated value of β. The number of simulations is chosen in order to reach a 99% confidence interval.

```
import math
import random
import statistics

n = 2048 # Number of bins
l = 1024# Number of balls
T = 100000# Number of simulations
List_Z = []

for j in range(T):
  #Repeat for each simulation
  Bins = [0 for k in range(n)]
  count_odd = 0
  #We throw the l balls
```

```
    for k in range(l):

      #Choice of the bin among the n
      r = random.randint(0,n-1)
      count_odd += (1-2*Bins[r])
      Bins[r] = (Bins[r] +1) % 2
    List_Z.append(abs(1/2-count_odd/n))
    #Z = n/2 - E[R], and R = count_odd/n.
mean_Z = statistics.mean(List_Z)
# To determine a confidence interval
stdev_Z = statistics.stdev(List_Z)


print("The confidence interval at 99% is
: [{} - {}, {} + {}]".format(
mean_Z,3*stdev_Z/math.sqrt(T),
mean_Z,3*stdev_Z/math.sqrt(T)))
```