2-3 OWF and PRF Analysis

1 Schemes and Goals

In the following we will define several cryptosystems that are based on mixing linear operation over $\mathbb{F}_2$ with linear operation over $\mathbb{F}_3$. Each cryptosystem is associated with 3 parameters, n, m, t such that $m > t$, while the relation between n and m depends on the cryptosystem. The input to each cryptosystem consists of a secret over $\mathbb{F}_2^n$ chosen uniformly at random, to which an initial transformation is applied to obtain a vector $w \in \mathbb{F}_2^m$, and a final transformation is applied to w to obtain The output $y \in \mathbb{F}_2^t$ or $y \in \mathbb{F}_3^t$ (depending on the cryptosystem). The transformations involve public matrices or vectors selected uniformly at random (perhaps under some constraints).

2-3 OWF. The parameters n, m, t satisfy $m \geq n$, $m \geq t$. Given a secret $x \in \mathbb{F}_2^n$ and two public matrices $A \in \mathbb{F}_2^{m \times n}$, $B \in \mathbb{F}_3^{t \times m}$:

1. Compute $w = Ax \in \mathbb{F}_2^m$.
2. View w as a vector in $\mathbb{F}_3^m$ and output $y = Bw \in \mathbb{F}_3^t$.

2-3 PRF. The parameters n, m, t satisfy $n \geq m$, $m \geq t$. The secret input is $k \in \mathbb{F}_2^n$. Let $K \in \mathbb{F}_2^{m \times n}$ be a circulant matrix whose rows include all cyclic rotations of k. Given a public input vector $x \in \mathbb{F}_n^2$ and and a public matrix $B \in \mathbb{F}_3^{t \times m}$:

1. Compute $w = Kx \in \mathbb{F}_2^m$
2. View w as a vector in $\mathbb{F}_3^m$ and output $y = Bw \in \mathbb{F}_3^t$.

Compressed LPN-style PRG. The parameters n, m, t satisfy $m \geq n$, $t = 2n$ (the goal is to double the seed length). Given a secret seed $x \in \mathbb{F}_2^n$ and two public matrices $A \in \mathbb{F}_2^{m \times n}$, $B \in \mathbb{F}_2^{t \times m}$:

1. Compute $w = Ax \bmod 2 + ((Ax \bmod 3) \bmod 2) \bmod 2 \in \mathbb{F}_2^m$.
2. Output $y = Bw \in \mathbb{F}_2^t$.

Compressed LPN-style PRF. The parameters n, m, t satisfy $n \geq m$, $m \geq t$. The secret input is $k \in \mathbb{F}_2^n$. Let $K \in \mathbb{F}_2^{m \times n}$ be a circulant matrix whose rows include all cyclic rotations of k. Given a public input vector $x \in \mathbb{F}_2^m$ and a public matrix $B \in \mathbb{F}_2^{t \times m}$:

1. Compute $w = Kx \bmod 2 + ((Kx \bmod 3) \bmod 2) \bmod 2 \in \mathbb{F}_2^m$.
2. Output $y = Bw \in \mathbb{F}_2^t$.

1.1 Choosing Public Inputs

All the cryptosystems we define receive public inputs chosen at random. For example, the 2-3 OWF receives matrices A and B as public inputs. One option is to choose A and B independently per secret input. While an alternative option is to fix one (or even both) of the matrices and reuse them. Generally, this alternative option is more susceptible to multi-target attacks and attacks that are based on self-similarity. Thus, in general, the first option is considered more secure and this is the option we use. For similar reasons we choose the public parameters independently per secret for the PRFs and the PRG we define.

Of course, there are bad choices of the public inputs which could degrade security, and we need to show that these are unlikely to be occur.

Finally, for the PRFs we define, this still leaves open the question of how to select the public inputs x and B per sample computed with a secret key. We chose to select x independently per sample, while fixing B per secret key, as this allows to optimize performance by preprocessing. In terms of security, the choice of fixing B does allow for a wider range of attacks that we need to consider, as we demonstrate in the security analysis.

Restricted public matrices. In order to optimize performance, we select the public matrices for the schemes as random Toeplitz matrices. These matrices define a pairwise independent hash family and are known to satisfy the Gilbert–Varshamov bound for linear codes, which is the main tool we use in the analysis.

Private matrices for PRF constructions. The private matrix K in the PRF constructions is a circulant matrix defined by rotations of a secret $k \in \{0,1\}^n$. While a choice of K with a small rank deficiency does not seem to have a significant impact on the security, some attacks on the schemes (particularly on the 2-3 PRF) may exploit matrices of particularly low rank (as $w = Kx \bmod 2$ resides in a subspace of small dimension).

Thus, if possible, it is preferable to select k such that K is a full rank matrix. Yet, this may require additional communication is distributed protocols. Otherwise, we need to understand the rank distribution of K when k is selected uniformly at random and prove that K has very small rank with negligible probability. For particular choices of n the analysis is simple.

Proposition 1.1. *Let $n = 2^{n'}$ for a positive integer n' and let $K \in \mathbb{F}_2^{n\times n}$ be a circulant matrix selected uniformly at random. Then, for any $a \in \{0,\dots n\}$, $\Pr[rank(K) \leq a] = 2^{-n+a}$.*

Proof. For every vector $u \in \mathbb{F}_2^n$ we associate a polynomial of degree at most $n-1$, $u(x) = \sum_{j=0}^{n-1} u_j x^j \in \mathbb{F}_2[x]$.

Assume that K is formed by rotations of $k \in \mathbb{F}_2^n$. Then for $i \in \{0,\dots,n-1\}$, the i'th column of K is associated with the polynomial $x^i \cdot k(x) \bmod x^n - 1$. Thus, for $u \in \mathbb{F}_2^n$, Ku is given by the coefficients of the polynomial $u(x) \cdot k(x) \bmod x^n - 1$, and $Ku = 0$ if and only if $u(x) \cdot k(x)$ divides $x^n - 1$.

Therefore, there is a bijection between the kernel space of K of the subspace of polynomials $u(x) \in \mathbb{F}_2[x]$ of degree at most $n-1$ that satisfy $u(x) \cdot k(x) \bmod x^n - 1 = 0$. The dimension of this subspace of polynomials is equal to the degree of $\gcd(k(x), x^n - 1)$ over $\mathbb{F}_2[x]$.

Recalling that $n = 2^{n'}$, over $\mathbb{F}_2[x]$ we have $x^{2^{n'}} - 1 = (x-1)^{2^{n'}} = (x-1)^n$. To conclude the proof, $rank(K) \leq a$ if and only if the kernel dimension of K is at least $n - a$. Equivalently, the degree of $\gcd(k(x), (x-1)^n)$ is at least $n - a$, or $k(x)$ divides $(x-1)^{n-a}$. There are exactly 2^a polynomials of degree at most $n - 1$ that divide $(x-1)^{n-a}$ and the probability of selecting one of them is $\frac{2^a}{2^n} = 2^{-n+a}$ as claimed. ■

1.2 Security Goals

For each scheme the goal is to obtain parameter sets n, m, t such that it offers s-bit security, as defined below, while achieving good performance in distributed protocols.

OWF security. Given a OWF scheme $F(\cdot)$ (applied to a secret input, where the public parameters are embedded into F), we define an inversion attack game by choosing $\hat{x} \in \mathbb{F}_2^n$ uniformly at random and giving $\hat{y} = F(\hat{x})$ to the adversary, whose goal to output some $x \in \mathbb{F}_2^n$ such that $F(x) = \hat{y}$. We say that $F(\cdot)$ has s bits of security if no adversary can win the inversion attack game on $F(\cdot)$ with average complexity below 2^{s-1}.

PRF security. We refer to both PRFs listed above.

The key $k \in \mathbb{F}_2^n$ that defines the secret matrix K is chosen uniformly at random. Moreover, the public matrix $B \in \mathbb{F}_3^{t \times m}$ (or $B^{(i)} \in \mathbb{F}_2^{t \times m}$) is chosen uniformly at random (or as a random Toeplitz matrix). For a parameter r, an adversary is given 2^r samples $(x^{(1)}, B, y^{(1)}), \ldots, (x^{(r)}, B, y^{(r)})$, where each $x^{(i)} \in \mathbb{F}_2^n$ is chosen independently and uniformly at random.

We will place a restriction of $r \leq 40$, corresponding to a practical limit of 2^{40} on the number of samples available to the adversary.

We consider two types of adversaries. The first type is a distinguisher that attempts to distinguish 2^r samples where each $y^{(i)}$ is generated using the PRF with a fixed k from 2^r samples where each vector $y^{(i)}$ is chosen uniformly at random. The second type attempts to find the secret key given samples generated using the PRF.

We say that the PRF has s bits of security if given (at most) 2^r samples both conditions below hold.

1. The advantage of a distinguishing adversary that runs in time 2^τ is at most $2^{(\tau-s)/2}$.

2. The probability that an adversary that runs in time 2^{τ} find the key is at most $2^{\tau - s}$.

This definition is aligned with the work of Micciancio and Walter [7].

PRG security. The secret seed $x \in \mathbb{F}_2^n$ is chosen uniformly at random, along with the public parameters A, B. The adversary is given a single sample A, B, y. As for PRFs, security is defined against the two types of adversaries and the definition is similar.

1.3 Algebraic Attacks

In algebraic attacks the attacker represents the outputs (on internal variables) of the cipher as multivariate polynomials in the secret key (or preimage), obtaining a system of polynomial equations. The attacker then attempts to the solve the system using techniques such as linearization or applying algorithms for finding a reduced representation of the ideal generated by the polynomials in the form of a Gröbner basis. Such methods are known to be efficient only in particular cases where the polynomials have a special structure, or the polynomials equations are of low degree and the attacker obtains sufficiently many equations to solve the system by linearization.

In our case, the output of the schemes we define mix between the sums $\bmod 2$ and $\bmod 3$. For example, in the 2-3 OWF and PRF constructions each output entry is a sum $\bmod 3$ of entries of w, where each such entry is a sum $\bmod 2$ of the unknown bits of the secret input. Due to the mix between the sums $\bmod 2$ and $\bmod 3$ we conjecture (similarly to [2]) that the output cannot be represented (or well-approximated) by a low degree polynomial over any specific polynomial ring. In particular, it was shown in [2] that the sum mod 3 of ℓ binary-valued variables is a high-degree polynomial over $\mathbb{F}_2$, as long as ℓ is large (e.g., $\ell \approx n$). Crucially, our choice of parameters will ensure that the sums mod 2 and mod 3 are dense and contain many terms. In particular, for the 2-3 OWF and PRF constructions we will make sure that the linear code spanned by the rows of B has large minimal distance, except with very small probability. Overall, we do not expect algebraic attacks to pose a threat to our schemes, and our analysis is mainly based on combinatorial attacks that attempt to recover the secret, or on statistical attacks whose goal is to distinguish the output from random.

2 Security Evaluation of the 2-3 OWF

In this section we analyze the security of the 2-3 OWF. Beforehand, we note that we may assume without loss of generality that in the most efficient construction, the number of expected preimages is (about) 1. Specifically, in our case, we may assume that $n = \log 3 \cdot t$ (up to rounding factors).

Indeed, setting $\log 3 \cdot t > n$ does not reduce the average number of preimages substantially. Consequently, any attack on a scheme with $n = \log 3 \cdot t'$ can be applied to a scheme with $\log 3 \cdot t > n$ by truncating the output to be of length $\log 3 \cdot t'$. Hence a scheme in which $\log 3 \cdot t > n$ does not offer better security than the truncated one. On the other hand, the truncated scheme has shorter output and is generally more efficient. Similarly, if $n > \log 3 \cdot t$, an attacker can fix $n - \log 3 \cdot t$ bits of the secret input to an arbitrary value and try to invert the image of the induced scheme where $n' = \log 3 \cdot t$ (note that on average, such a preimage exists).

2.1 Basic Attacks

We describe several basic attacks and analyze their complexity as a function of n, m, t. First, by exhaustive search, we can invert $\mathrm{OWF}(\cdot)$ in time complexity 2^n or $3^t = 2^{\log 3 \cdot t}$.

Focusing on the value of m, by exhaustive search, we can find x such that $Ax = A\hat{x}$ (which implies that $\mathrm{OWF}(x) = \mathrm{OWF}(\hat{x})$) in time complexity 2^m. A tighter restriction on m is imposed by the following attack: guess $m - t$ bits of $w = Ax$ and solve the linear equation system $\hat{y} = Bw$ over $\mathbb{F}_3$ (which has t equations and variables) to obtain a full suggestion for w. A suggestion for w allows to compute x by solving the linear equation system $Ax = w$ over $\mathbb{F}_2$. This attack has complexity 2^{m-t}. An improved attack is described next.

Enumerating w values. We show how to enumerate over all $w \in \{0,1\}^m$ that satisfy $Bw = \hat{y}$ in time complexity of about $2^{m/2}$ if $m \leq 2\log 3 \cdot t = 2n$, and $2^{m - \log 3 \cdot t} = 2^{m-n}$, otherwise.

Given such an algorithm, we can test each w by solving the equation system $Ax = w$ over $\mathbb{F}_2$, and if a solution exists, we have successfully inverted $\hat{y}$.

Observe that if w and w' do not have a common 1 entry, then $w + w' \bmod 2 = w + w' \bmod 3$ (where the addition is performed entry-wise). Therefore,

$$\begin{aligned} B(w + w' \bmod 2) \bmod 3 &= \\ B(w + w' \bmod 3) \bmod 3 &= \\ (Bw \bmod 3) + (Bw' \bmod 3) \bmod 3&. \end{aligned} \tag{1}$$

We use this observation in the following algorithm, whose complexity as claimed above.

1. Partition the m indices of w into 2 subsets I_1 and $I_2 = [m] \backslash I_1$, each of size $m/2$ bits.
2. For $i \in \{0, 1, \ldots 2^{m/2} - 1\}$, let w_i be the m-bit vector whose value on the $m/2$ indices of I_1 is i, and is 0 on the indices of I_2. For each such i, evaluate $Bw_i \bmod 3 = y_i$ and store the pairs (w_i, y_i) in a table $\mathcal{T}$, sorted by y_i values.
3. For $j \in \{0, 1, \ldots 2^{m/2} - 1\}$, let w'_j be the m-bit vector whose value on the $m/2$ indices of I_2 is j, and is 0 on the indices of I_1. For each such j, evaluate $Bw'_j \bmod 3 = y'_j$ and search $\mathcal{T}$ for the value $\hat{y} - y'_j \bmod 3$. If there exists a match y_i such that $y_i = \hat{y} - y'_j \bmod 3$ (or $y_i + y'_j \bmod 3 = \hat{y}$), recover the value w_i such that $Bw_i \bmod 3 = y_i$ from $\mathcal{T}$ and return $w = w_i + w'_j \bmod 2$.

Note that the expected number of $w \in \{0,1\}^m$ that satisfy $Bw = \hat{y}$ is $2^{m-\log 3 \cdot t}$. Hence, we cannot hope to obtain better complexity than $2^{m-\log 3 \cdot t}$ without exploiting additional constraints on w, imposed by the matrix A. In Section 2.2, we show how this can be done.

Induced schemes. Given the scheme $\mathrm{OWF}(\cdot)$ with a matrix B and output y such that $Bw = y$, and any positive integer r, we can left-multiply both sides by any $r \times t$ matrix C over $\mathbb{F}_3$ to obtain $CBw = Cy$. Note that each row of the matrix CB is a linear combination of the rows of B. Using such a matrix C, we can perform Gaussian elimination on the rows of B.

We denote the resultant induced scheme by $\mathrm{OWF}_C(\cdot)$. Observe that if $\mathrm{OWF}(x) = y$, then $\mathrm{OWF}_C(x) = Cy$. In addition, assuming that the rank of C is $rank(C) \leq t$, we estimate that an arbitrary x that satisfies $F_C(x) = Cy$ also satisfies $F(x) = y$ with probability $3^{-(t-rank(C))}$. We now describe a simple attack that uses an induced scheme with $rank(C) = 1$. Another attack is described in Appendix B.

Low Hamming weight combinations of the rows of $\boldsymbol{B}$. Assume that there is a vector $v \in \mathbb{F}_3^m$ of Hamming weight ℓ in the row space of B, namely, there exists a vector $u \in \mathbb{F}_3^t$ for which $uB = v$. If ℓ is sufficiently small, then we could use the induced scheme $\mathrm{OWF}_u(\cdot)$ to speed up exhaustive search.

Denote the set of ℓ non-zero indices of v by I. Given $\hat{y} = Bw \bmod 3$, we compute the value of $u\hat{y} \bmod 3 = vw \bmod 3$. We can now enumerate the values of the corresponding set I of ℓ bits of w for which $u\hat{y} \bmod 3 = vw \bmod 3$ holds. This set of bits has $\frac{2^\ell}{3}$ possible values. Each such ℓ-bit value gives rise of a system of ℓ linear equations on x, and we exhaustively search its solution space of size $2^{n-\ell}$. Overall, if $\ell \leq n$ the complexity of the attack is $\frac{2^\ell}{3}$, while if $\ell = n+1$, the complexity is $\frac{2^{\ell+1}}{3}$. When $\ell > n+1$, the complexity is higher than 2^n. Thus, we will require that such a vector v of low Hamming weight about n does not exist, except with small probability. This probability is computed in Proposition A.1 in Appendix A.

If more such low Hamming weight vectors are available, then the complexity of the attack may be further reduced, although it seems unlikely to obtain a significant advantage over exhaustive search with this approach.

2.2 Reduction to Subset-Sum

We now show how to reduce the problem of inverting $\mathrm{OWF}(\cdot)$ to solving an instance of subset-sum (for a particular definition of the sum operation) over the space $\{0,1\}^m$.

The reduction. For a vector $w \in \mathbb{F}_2^m$, there is an $(m-n) \times m$ (parity check) matrix P such that there exists $x \in \mathbb{F}_2^n$ for which $Ax = w$ if and only if $Pw = 0$. Assume that $x \in \mathbb{F}_2^n$ satisfies $\mathrm{OWF}(x) = \hat{y}$ and let $w = Ax$. Then, w satisfies the conditions $Pw = 0$ (over $\mathbb{F}_2$) and $Bw = \hat{y}$ (over $\mathbb{F}_3$). We attempt to find such w by a reduction to subset-sum, as detailed below.

Suppose we find a set $J \subseteq [m]$ such that

$$\left(\sum_{j \in J} Pe_j \bmod 2, \sum_{j \in J} Be_j \bmod 3 \right) = (\vec{0}, \hat{y}),$$

where $e_i \in \{0,1\}^m$ be the i'th unit vector. Then, a preimage x such that $\mathrm{OWF}(x) = \hat{y}$ can be computed by solving the linear equation system $Ax = \sum_{j \in J} e_j \bmod 2$.

Thus, we have reduced the problem to subset-sum with m binary variables $(\epsilon_1, \ldots, \epsilon_m) \in \{0,1\}^m$, where we associate $\epsilon_i = 1$ with

$$(Pe_i, Be_i) \in \mathbb{F}_2^{m-n} \times \mathbb{F}_3^t,$$

and define the target as $(\vec{0}, \hat{y}) \in \mathbb{F}_2^{m-n} \times \mathbb{F}_3^t$.

Solving the subset-sum problem. We can now apply the advanced subset-sum algorithm by Howgrave-Graham and Joux [6] and the more recent ones [1, 3], which are based on the *representation technique*. These algorithms were mostly designed to solve subset-sum problems over the integers. Below, we describe the main ideas of these algorithms and explain how to apply them to the special subset-sum problem we consider.

In the subset-sum problem over the integers, we are given a list of m positive integers $(a_1, a_2, \ldots, a_m)$ and another positive integer S such that $S = \sum_{i=1}^{m} \epsilon_i a_i$ for $\epsilon_i \in \{0,1\}$. The goal is to recover the unknown coefficients ϵ_i.

A standard meet-in-the-middle approach for solving the problem has time complexity of about $2^{m/2}$. The representation technique gives an improved algorithm as briefly summarized below.

Assume that a solution to the subset-sum problem is chosen uniformly from $\{0,1\}^m$ and the parameters are set such that the instance has about one solution on average. Effectively, this means that the density of the problem $d = \frac{n}{\log \max(\{a_i\}_{i=1}^m)}$ is set to 1.

The main idea of the basic algorithm of Howgrave-Graham and Joux [6] is to split the problem into two parts by writing $S = \sigma_1 + \sigma_2$, where $\sigma_1 = \sum_{i=1}^{m} \alpha_i a_i$, $\sigma_2 = \sum_{i=1}^{m} \beta_i a_i$ and $(\alpha_i, \beta_i) \in \{(0,0), (0,1), (1,0)\}$. Thus, $\epsilon_i = \alpha_i + \beta_i$ for each i is a solution to the problem.

Note that each coefficient ϵ_i with value 1 can be split into $(0,1)$, or $(1,0)$. Thus, assuming that the solution has Hamming weight[1] of $m/2$ (which occurs with probability $\Omega(1/\sqrt{m})$), it has $2^{m/2}$ different *representations*. Consequently, we may focus of finding only one of these representations by solving two subset-sum problems of Hamming weight $m/4$. Focusing on a single representation of the solution allows to beat the standard meet-in-the-middle approach which requires time $2^{m/2}$.

[1] In general, one may guess the Hamming weight of the solution and repeat the algorithm accordingly a polynomial number of times.

Adaptation of previous subset-sum algorithms. The algorithm of [6] can be easily adapted to our specialized subset-sum problem (although it is not defined over the integers). Moreover the improved algorithm of [1] considers additional representations of the solution by allowing α_i and β_i to also take the value -1 (implying that $\epsilon_i = 0$ can be decomposed into $(\alpha_i, \beta_i) \in \{(0,0),(-1,1),(1,-1)\}$). In our case, we associate $\alpha_i = -1$ with $(P(-e_i), B(-e_i)) = (Pe_i, 2 \cdot Be_i) \in \mathbb{F}_2^{m-n} \times \mathbb{F}_3^t$. Finally, the recent improved algorithm of [3] considers representations over $\{-1,0,1,2\}$ and we can adapt this algorithm to our setting in a similar way.

In terms of complexity, ignoring polynomial factors in m, the attack of [6] runs in time $2^{0.337m}$ and uses $2^{0.256m}$ memory, while the complexity of attack of [3] requires $2^{0.283m}$ time and memory.

2.3 Quantum Attacks

Attackers with access to a quantum computer can improve upon the complexity of some of the attacks described in the classical setting. In the classical setting, exhaustive search and the subset-sum based algorithm are the most relevant attacks that we found of the scheme. This also applies to the post-quantum setting.

First, it is possible to invert $\mathrm{OWF}(\cdot)$ with Grover's algorithm in time complexity $2^{n/2}$. Second, according to [3], one can solve subset-sum (in m binary variables) on a quantum computer in complexity $2^{0.2156m}$ (ignoring polynomial factors).

2.4 Parameter Selection for the 2-3 OWF

According to the analysis, we determine parameters n, m, t for which we conjecture that $\mathrm{OWF}(\cdot)$ has s bits of security.

First, due to the exhaustive search, we require

$$n \geq s.$$

Second, the most restrictive constraint on m is imposed by the subset-sum algorithm. If we conservatively ignore the hidden polynomial factors and the large memory complexity of the subset-sum algorithm of [3], we need to set

$$0.283m \geq s.$$

Overall, we obtain

$$n = \log 3 \cdot t = s,$$

and

$$m = \tfrac{s}{0.283} \approx 3.53s.$$

We now consider the attack exploiting low Hamming weight combinations of the rows of B, and in particular, Proposition A.1. In our case, we apply the proposition with $\log 3 \cdot t = n$ and $\ell = n$. For $m = 3.53n$, we obtain that the probability of having a vector of Hamming weight at most n is bounded by

$$2 \cdot 2^{m(H(0.283)+2\cdot 0.283-\log 3)} \approx 2 \cdot 2^{-0.16m} \approx 2 \cdot 2^{-0.56s}.$$

For $s \geq 128$, the expression evaluates to (less than) 2^{-70}, so it is unlikely to encounter such an event in practice. Moreover, even if the event occurs, security only regrades by a factor of 3, and by the same proposition the probability that two such vectors are spanned by the rows of B is at most 2^{-140}. Nevertheless, one may increase n (and correspondingly t) by a few bits (at negligible cost) to defeat this attack vector completely.

A more aggressive setting of the parameters may take into account the polynomial factors of [3] (and perhaps its high memory complexity). Unfortunately, the polynomial factors associated with the complexity formulas of the relevant subset-sum algorithms have not been analyzed. For example, if we assume that the polynomial factors are about m^2, and we aim for $s = 128$ bits of security, then we require $m^2 \cdot 2^{0.283m} \geq 2^s = 2^{128}$. Setting $m = 400 = 3.125s$ is sufficient for satisfying the constraint in this setting.

Post-quantum setting. In the post-quantum setting, we have the constraints $n/2 \geq s$ due to Grover's algorithm, and $0.2156m \geq s$ due to the quantum subset-sum algorithm.

Overall, we obtain $n = \log 3 \cdot t = 2s$, and $m = \frac{s}{0.2156} \approx 4.64s$.

3 Security Evaluation of the 2-3 PRF

As for the OWF, we describe several attacks and analyze their complexity as a function of the parameters n, m, t.

Unlike the case of the OWF (where the goal was to find a preimage of a give output), we can choose a small value of t regardless of the other parameters without sacrificing security. In fact, it is clear that a small value of t can only contribute to security, as any attack on a scheme with a small value of t can be applied to a scheme with a larger value of t, simply by ignoring part of the output. Consequently, we may fix t to the smallest value acceptable by the application.

We also note that given a sufficiently large number of samples, we expect that the key k would be uniquely determined by the samples (regardless of the value of t).

3.1 Key Recovery Attacks Exploiting a Few Samples

We describe key recovery attacks that make use of the minimal number of samples required to derive the secret key k.

First, as for the OWF, exhaustive search requires 2^n time. Also, similarly to the OWF, given any sample (x, B, y), we can guess $m - t$ bits of $w = Kx$ and solve the linear equation system $y = Bw \bmod 3$, which then allows to compute a suggestion for k (that can be tested on the remaining samples). This attack has complexity 2^{m-t}.

Furthermore, given a single sample, we can apply the same attack for enumerating w values, described at the beginning of Section 2.2. This attack has complexity which is the maximum between $2^{m/2}$ and $2^{m-\log 3 \cdot t}$.

Reduction to subset-sum. As for the OWF, we can reduce the key recovery problem to the problem of solving subset-sum over the m binary variables of w. However, it is clear that if the algorithm is applied to a single (x, B, y) sample, then its expected complexity cannot drop below $2^{m-\log 3 \cdot t-(m-n)} = 2^{n-\log 3 \cdot t}$, which is the expected number of w values possible given (x, B, y) (the remaining key candidate after analyzing one sample are tested against another sample).

On the other hand, if we try to reduce the complexity by applying the algorithm to more than one sample (e.g., to (x, B, y) and (x', B, y')), then we must take advantage of the dependency between $w = Kx$ and $w' = Kx'$, which are related via linear constraints, imposed by k and by x, x'. However, it is not clear how to model these complex linear constraints in the subset-sum reduction and we were not able to improve the complexity of the single-sample attack.

3.2 Exploiting Multiple Samples

The key recovery attacks described above make use of a minimal number of samples required to derive the secret key. On the other hand, when given more samples, it may be possible to exploit various relations among them to mount distinguishing and even key recovery attacks which we investigate below.

Output bias. We consider a single sample (x, B, y) and analyze the bias of linear combinations of the entries of y over $\mathbb{F}_3$. If any such linear combination has a sufficiently high bias towards some constant, then an attacker can exploit it in a distinguishing attack.

Similarly to the case analyzed for the corresponding OWF, assume there are vectors $v \in \mathbb{F}_3^m$ and $u \in \mathbb{F}_3^t$ such that $uB = v$ and the Hamming weight of v is ℓ. Given $y = Bw \bmod 3$, the attacker computes $uy \bmod 3 = vw \bmod 3$ and thus obtains the value of a linear combination $\bmod 3$ of ℓ entries of $w \in \{0,1\}^m$. Specifically, denoting the set of ℓ non-zero indices of v by I, the attacker computes $\sum_{i \in I} v_i w_i \bmod 3$. We now calculate the bias of sum the mod 3, assuming that w is uniformly distributed in $\mathbb{F}_2^m$.

Each non-zero coefficient of the linear combination v is either 1 and 2. It is easy to prove by induction on ℓ (or by analysis of sums of binomial coefficients) that for any coefficients $v_i \in \{1, 2\}$ where $i \in I$ and for any $a \in \{0, 1, 2\}$,

$$\Pr\left[\sum_{i \in I} v_i w_i \bmod 3 = a\right] \in \{\tfrac{1}{3} \pm \tfrac{1}{2^\ell}, \tfrac{1}{3} \pm \tfrac{2}{2^\ell}\},$$

where the actual probability depends on v, a and ℓ. Thus, the bias of $vw \bmod 3$ is bounded by $\frac{2}{2^\ell}$.

We use Proposition A.1 to deduce that the subspace spanned by the rows of B contains a vector of Hamming weight at most ℓ with probability at most $2 \cdot 2^{m(H(\ell/m)-\log 3)+\ell+\log 3 \cdot t}$.

Our analysis is conservative, as we ignore the work performed by the attacker to find a low Hamming weight vector v spanned by the rows of B. Thus, we will be interested in $\ell \approx s/2$, as we would like to avoid having a linear combination of the output with bias at least $2^{-s/2}$ (except with small probability). Plugging this into the formula above we bound the probability by

$$2 \cdot 2^{m(H(s/2m)-\log 3)+s/2+\log 3 \cdot t}. \tag{2}$$

Conditional output bias. As described above, the bias of expressions of the form $\sum_{i \in I} v_i w_i \bmod 3$, (where $I \subseteq [m]$ is a set of size ℓ, each $v_i \in \{1, 2\}$ is fixed and $w_i \in \{0, 1\}$ are independent and uniformly distributed random variables) is bounded by $2^{-\ell+1}$.

However, the bias may increase if information about the variables w_i is known. In particular, the case in which $\sum_{i \in I} w_i \bmod 2$ is known was analyzed in [4], where the authors showed that the conditional biases such as

$$\left| \Pr\left[\sum_{i \in I} w_i \bmod 3 = 0 \mid \sum_{i \in I} w_i \bmod 2 = 0 \right] - 1/3 \right|$$

can be as large as about $2^{-0.21\ell}$. While this is still exponentially small in ℓ, it is more significant than the unconditional bias.

The PRF candidate proposed in [2] and analyzed in [4] is similar to the one we analyze for $n = m$, yet the matrix B contains a single row that sums mod 3 all the entries of w. Moreover, since K is a circulant matrix and x has even Hamming weight, then $\sum_{i \in [m]} w_i \bmod 2 = 0$ and the distinguishing attack is applicable. In our case, B is selected at random and the parameters are chosen such that the attacker cannot obtain $\sum_{i \in I} w_i \bmod 3$ for any fixed set I (and particularly $I = [m]$), except with negligible probability. The general distinguishing attack is analyzed below.

Assume that given a sample (x, B, y), there exists a set $I \subseteq [m]$ (that depends of x) such that $\sum_{i \in I} w_i \bmod 2$ is known to the attacker. Moreover, assume that there exists v in the row span of B such that $v_i = 1$ for each $i \in I$ and $v_i = 0$ for each $i \notin I$. Then, the value $\sum_{i \in I} w_i \bmod 3$ can be computed from the output, and the distinguishing advantage is as high as (about) $2^{-0.21\ell}$. Of course, if several samples are available, the distinguishing advantage can increase by accumulating the bias, assuming the above conditions are fulfilled for more than one sample.

An extended variant of the attack may consider a vector $v' = v + u$ in the row span of B. Denoting that Hamming weights of v and u by ℓ_1,ℓ_2, respectively, the conditional bias can be as high as $2^{-0.21\ell_1 - \ell_2 + 1}$. Note that here we conservatively ignore the work required to find such v'.

Given 2^r samples (where $r \leq 40$), we wish to show that the distinguishing advantage is small (except with negligible probability). Indeed, calculation shows that this distinguisher is not stronger that the unconditional distinguisher. Essentially, given a single sample, for any fixed I, the vector v as described above is in the row span of B with minuscule probability 3^{t-m}. In the extended attack, we consider a ball around v, but this ball is much smaller than the one considered in the unconditional distinguisher (as $\ell_2 < s/2$).

Finally, in a more general variant of the attack, the attacker may guess a parity of key bits and calculate the conditional bias over several samples, attempting to amplify it. Note that the desired vector v changes for each sample according to x. By similar calculation, once the attacker has fixed the guess given a sample, almost all additional samples would not allow to amplify the conditional bias, as a good vector is unlikely to be in the row span of B.

Differential cryptanalysis and low Hamming weight samples. As we argue below, the 2-3 PRF seems to be immune to classical statistical differential cryptanalysis.

Assume that the attacker obtains two samples (x, B, y) and $(x + \delta \bmod 2, B, y')$, where $\delta \in \{0,1\}^n$. Denote $w = Kx \bmod 2$ and $w' = K(x+\delta) \bmod 2 = w + K\delta \bmod 2$. Thus, $y' = Bw' \bmod 3 = B(w + K\delta \bmod 2) \bmod 3$. In general, y and y' do not seem to have any statistical relation that holds with sufficiently high probability. Particularly, $w + K\delta \bmod 2 \neq w + K\delta \bmod 3$, except with very small probability.

From an algebraic viewpoint, the attacker can consider the m bits of $w = Kx \bmod 2$ as variables over $\mathbb{F}_2$, but then the algebraic degree of the output over $\mathbb{F}_3$ would be large. The attacker can also consider the m bits of w as variables over $\mathbb{F}_3$. In this case, the attacker obtains t linear equations and t quadratic equations mod 3 (of the form $(w_i)^2 - w_i = 0$) from the sample (x, B, y). On the other hand, the algebraic degree of $w' = K(x+\delta) \bmod 2$ in the variables of w would be large due the dense mod 2 operations, hence it is not clear how to obtain additional low degree equations. In general, such algebraic attacks do not seem more efficient than the attack described above for enumerating w values.

Another scenario which we consider is when the attacker obtains a sample (x, B, y) such that x is of low Hamming weight. In this case each entry of w is a low Hamming weight sum $\bmod 2$ of the bits of k and can thus be described as a low degree polynomial over $\mathbb{F}_3$. Consequently, low degree polynomial equations over $\mathbb{F}_3$ in the secret key can be deduced from the output. Obtaining several such samples may allow the attacker to solve for the key. In general, such low Hamming weight samples are avoided with high probability given the data limit, but we can also place a restriction on the sample generation, forcing it to generate vectors x with minimal Hamming weight (e.g., a lower bound of $n/4$).

Attacks based on self-similarity. There are several simple attacks that take advantage of the fact that B is fixed for all samples generated with the same secret k. A basic attack looks for a collision on the w values for a pair of samples, which can be detected at the output. Given 2^r samples, the advantage of this attack is $2^{2r-rank(K)}$. Given the data complexity bound, we will set the parameters such that the advantage of this attack is negligible (assuming the rank of K is not too small).

Another simple attack is a multi-target attack, where given 2^r samples, the attacker guesses w, computes $Bw \bmod 3$ and compares the result with all given outputs. A match allows to recover a candidate for the secret k. The expected complexity of this attack is 2^{m-r}, but cannot drop below $2^{m-\log 3 \cdot t}$ without exploiting relations between the different w values. In general, given the data limit, the attack less efficient that the attack that enumerates all w values considered above.

Simultaneous sums. We describe a more involved self-similarity attack, which exploits the fixed B value per secret k.

Assume that for some index set I of size at least 3, $\sum_{i \in I} x^{(i)} \bmod 2 = 0$. Then, $\sum_{i \in I} w^{(i)} \bmod 2 = 0$. While it is not clear how this relation influences the output, we extend this initial observation by simultaneously considering sums mod 2 and 3, as follows: assume that there are 4 samples (denoted for simplicity by $\{(x^{(i)}, B, y^{(i)})\}_{i=1}^{4}$) such that for each $j \in [m]$,

$$\sum_{i=1}^{4} w_j^{(i)} = 2.$$

Then, $\sum_{i=0}^{4} x^{(i)} \bmod 2 = 0$ and $\sum_{i=0}^{4} y^{(i)} \bmod 3 = B \cdot \vec{2} \bmod 3$ (where $\vec{2}$ is a vector with m entries whose values are 2). Note that a random 4-tuple of samples satisfies this simultaneous sum constraint with probability $2^{-n-\log 3 \cdot t}$, but the probability that the constraint $\sum_{i=0}^{4} w_j^{(i)} = 2$ is satisfied for every $j \in [m]$ is about

$$\left(\frac{\binom{4}{2}}{16}\right)^{-m} \approx 2^{-1.415m},$$

which is higher than expected if $1.415m < n + \log 3 \cdot t$.

Since the adversary has about 2^{4r} such 4-tuples, the probability that such a simultaneous 4-sum exists is about $2^{4r-1.415m}$. It can be detected in time complexity of about 2^{2r} using a standard matching algorithm. The important constraints for defending against this attack are

$$1.415m > n + \log 3 \cdot t,$$

or

$$2^{4r-1.415m} \tag{3}$$

is negligible, otherwise.

The simultaneous 4-sum distinguisher can be easily generalized to a simultaneous d-sum distinguisher for arbitrary d. In general, we look for d-tuples where (for example) for each $j \in [m]$,

$$\sum_{i=1}^{d} w_j^{(i)} = c,$$

for some value c (fixed mod 6) such that $c \bmod 2 = 0$. However, calculation shows that $d = 4$ gives the most efficient distinguisher in our case (with the small data limit).

3.3 Quantum Attacks

The relevant attacks in the quantum setting are similar to ones for the OWF. First, Grover's algorithm, breaks the scheme (recovers the key) is time $2^{n/2}$. Second, the quantum subset-sum algorithm of [3] has complexity $2^{0.2156m}$ (ignoring polynomial factors), but does not drop below $2^{n-\log 3 \cdot t}$.

3.4 Parameter Selection for the 2-3 PRF

According to the analysis, we determine parameters n, m, t for which we conjecture that the 2-3 PRF has s bits of security.

In order to select the parameters, we may first set t to it's minimal possible value (depending on the application). We also assume that t is not too large, and particularly $\log 3 \cdot t \leq s$.

The constraints imposed by the above attacks are as follows. First, due to exhaustive search, we require

$$n \geq s.$$

Second, the subset-sum algorithm imposes the constraint

$$n - \log 3 \cdot t \geq s,$$

(given that $n \geq m$ and $\log 3 \cdot t \leq s$).

We consider two sets of parameter. The first is

$$n = m = s + \log 3 \cdot t.$$

In particular, if $\log 3 \cdot t = s$, then $n = m = 2s$. The second parameter set

$$n = m = 1.25(s + \log 3 \cdot t),$$

and is less aggressive.

Next, we analyze the bias of the output based on (2), assuming that $\log 3 \cdot t = s$. For the first parameter set, we obtain that the probability of having bias of $\frac{2}{2^{s/2}}$ is bounded by

$$2 \cdot 2^{m(H(s/2m) - \log 3) + s/2 + \log 3 \cdot t} = 2 \cdot 2^{2s(H(1/4) - \log 3) + 3s/2} \approx 2 \cdot 2^{-0.049s},$$

which is non-negligible. On the other hand, the consequences of the distinguishing attack given the data limit seem relatively mild, and this parameter set may be considered by applications where performance is critical.

For the second parameter set (assuming $\log 3 \cdot t = s$), the probability of having bias $\frac{2}{2^{s/2}}$ is bounded by

$$2 \cdot 2^{m(H(s/2m) - \log 3) + s/2 + \log 3 \cdot t} = 2 \cdot 2^{2.5s(H(1/5) - \log 3) + 3s/2} \approx 2 \cdot 2^{-0.65s},$$

which we consider negligible.

The advantage of the collision attack is $2^{2r-rank(K)} \leq 2^{80-rank(K)}$. Given that $m = 2s \geq 256$, if we choose n as a power of 2, based on Proposition **??**, the advantage is much smaller than $2^{s/2}$ (except with negligible probability).

Finally, we consider the simultaneous 4-sum distinguisher, and recall that the probability of having such a 4-sum in the output is estimated in (3) as $2^{4r-1.415m} \leq 2^{160-2.83s}$. For a minimal choice of $s = 128$, this probability is smaller than 2^{-200} which is negligible.

4 Security Analysis of Compressed LPN-Style PRG

We analyze the security of the compressed LPN-Style PRG.

If the matrix was a random matrix, then the first step would consist of generating m samples from the alternative PRF construction proposed in [2]. Each sample $w_i = A[i]x \bmod 2 + ((A[i]x \bmod 3) \bmod 2) \bmod 2 \in \mathbb{F}_2^m$ can be viewed as adding noise $((A[i]x \bmod 3) \bmod 2)$ to the inner product $A[i]x \bmod 2$. Given that $A[i]$ is of sufficiently large Hamming weight, then $\Pr_x[(A[i]x \bmod 3) \bmod 2 = 1] \approx 1/3$, which is the magnitude of noise added. The second step consists of a compressing linear transformation B applied to w. The idea is to increase the noise of each sample by mixing it with other samples. This step should defeat standard attacks applied to LPN with a constant noise parameter (such as decoding attacks).

In our case, the matrix A is structured (it is a random Toeplitz matrix), but we were not able to exploit this in an efficient attack.

4.1 Key Recovery Attacks

We begin by considering attacks that attempt to recover the secret key (seed).

Exhaustive search for the key recovery attacks requires time 2^n.

In a different approach for recovering the key, given a sample A, B, y, the attacker enumerates over the subspace of w values that satisfy $Bw = y \bmod 2$. This subspace contains 2^{m-t} vectors. For each such vector, the attacker attempts to recover x given A and w. Thus, given A, w the attacker has m samples generated from the LPN-like construction proposed in [2] as a weak PRF (although we a structured matrix A). Given that m is not too large (i.e., it is a small multiple of n), then the best attack we have on this scheme simply tries to break the LPN instance (which has noise of 1/3), without exploiting the deterministic way in which it is generated. The concrete security of LPN given a small number was analyzed in several publications such as [5], and the complexity of known attack is generally exponential in n. Nevertheless, the attacker is required to solve 2^{m-t} related LPN instances, and perhaps can amortize the complexity. Moreover, the matrix A is structured. Thus, we (conservatively) estimate the total complexity of such attacks by 2^{m-t}.

4.2 Noisy Linear Equations

As noted above, each bit w_i for $i \in [m]$ can be viewed as a noisy linear equation over $\mathbb{F}_2$ with noise of about $1/3$, or bias $2/3 - 1/2 = 1/6$. Our goal is to select the parameter m such that the all linear combinations of the output bits of y have exponentially small bias towards a linear equation in the secret x. We will (heuristically) model each bit w_i as having independent bias of $1/6$.

If the linear subspace spanned by the rows of B contains a vector of Hamming weight ℓ, then by the piling-up lemma, the bias of the corresponding linear combination of the bits of w is

$$2^{\ell-1} \cdot (\tfrac{1}{6})^{\ell} < 2^{-\log 3 \cdot \ell}. \tag{4}$$

Thus, we will require that the rows of B do not span a vector whose Hamming weight is too low. This is similar to the previously analyzed schemes, but the lower bound on the Hamming weight we will enforce for the PRG will be lower. Indeed, the bias above is calculated with respect some linear equation in the unknown secret key bits. Such a bias is generally much less of a security concern compared to a bias towards a constant value (e.g., the bias analyzed for the for the 2-3 PRF construction) which can be used to directly distinguish the output from random using statistical tests. Particularly, an alternative scheme where we change the first transformation to only compute the "noise part" ($w = (Ax \bmod 3) \bmod 2$) would require larger parameters to be secure, as we need a higher lower bound on the Hamming weight ℓ to avoid distinguishing attacks.

For a PRG with s bits of security, we will conservatively require a bias of at most $2^{-0.1s}$. We are not aware of any attack that can exploit such a low bias. Effectively, this means that the minimal distance of B should be at least $s \cdot 0.1/\log 3 < 0.07s$ (except with small probability).

Remark 4.1. In [4] the authors analyze the constructions presented in [2]. In particular, for the alternative PRF construction, given a sample $(a \in \mathbb{F}_2^n, x \cdot a)$, they show that there exists $j \in [n]$ such that

$$|\Pr[a_j = 0 \mid x_j = 0 \text{ and } (a{\cdot}x \bmod 2 + ((a{\cdot}x \bmod 3) \bmod 2) \bmod 2) = 0]| \approx \tfrac{1}{2^{0.21\ell}},$$

where ℓ is the Hamming weight of a. This property was exploited in a distinguishing attack.

Our PRG construction seems to be immune to this type of analysis because the attacker only has access to sufficiently dense linear combinations of (structured) samples of the alternative PRF construction.

4.3 Parameter Selection for the Compressed LPN-Style PRG

We determine parameters n, m, t for which we conjecture that the PRG has s bits of security.

Recall the we set $t = 2n$. Exhaustive search implies that $n \geq s$ and we have lower bounded the effort required in key recover by 2^{m-t}. Thus, a reasonable choice of parameters is $n = s$ and $m = 3s, t = 2s$.

For these parameters, we consider the maximal bias of linear combinations according to (4), with $\ell = 0.07s$. By Proposition A.1, the probability of having a vector of Hamming weight more than $0.07s$ in the row span of B is $0.07s \cdot 2^{3s(H(0.07/3)-1)+2s} < 0.07s \cdot 2^{-0.52s}$. We consider this as a negligible probability as the consequences of the this unlikely event are mild.

5 Security Analysis of Compressed LPN-Style PRF

The overall structure of the compressed LPN-Style PRF is similar to the PRG. However, unlike PRG, in the first transformation, the key is part of the matrix and x is a public value. Thus, w can be viewed as m outputs of (another) more structured version of the construction of [2].

In terms of parameters, the main differences are that we have $n \geq m$ and t is not constraint to $2n$ (in fact, we will propose to set it smaller than n). Furthermore, we assume that the attacker obtains 2^r samples for $r \leq 40$ instead of a single sample.

We note that variants of the basic attacks that were analyzed for the 2-3 PRF (such as the collision attack and the multi-target attack) are also applicable to this construction. As in the case of the 2-3 PRF, they do not seem to be a threat for our choice of parameters.

Overall, we have not found any class of attacks that are applicable to this construction, but not to the previous ones. Below we briefly consider the most important attacks that influence the choice of parameters.

Summary of attacks. As for the PRG, we estimate the total complexity of key recovery attacks by 2^{m-t}. Although we have more samples that for the PRF, it is not clear how to exploit them to obtain better complexity.

In addition, similarly to the PRG, we require that each linear combination of the output has bias of at most $2^{-0.1s}$ (toward some linear combination of the key over $\mathbb{F}_2$).

Parameter selection for the Compressed LPN-style PRF. For s-bit security, we set $n = m = 2s$ and $t = s$. By our analysis, exhaustive search requires $2^{m-t} = 2^s$ time.

We consider the maximal bias of linear combinations according to (4), with $\ell = 0.07s$. By Proposition A.1, the probability of having a vector of Hamming weight more than $0.07s$ in the row span of B is $0.07s \cdot 2^{2s(H(0.07/2)-1)+s} < 0.07s \cdot 2^{-0.56s}$, which we consider negligible.

References

1. A. Becker, J. Coron, and A. Joux. Improved Generic Algorithms for Hard Knapsacks. In K. G. Paterson, editor, *Advances in Cryptology - EUROCRYPT 2011 -*

30th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Tallinn, Estonia, May 15-19, 2011. Proceedings, volume 6632 of *Lecture Notes in Computer Science*, pages 364–385. Springer, 2011.

2. D. Boneh, Y. Ishai, A. Passelègue, A. Sahai, and D. J. Wu. Exploring Crypto Dark Matter: - New Simple PRF Candidates and Their Applications. In A. Beimel and S. Dziembowski, editors, *Theory of Cryptography - 16th International Conference, TCC 2018, Panaji, India, November 11-14, 2018, Proceedings, Part II*, volume 11240 of *Lecture Notes in Computer Science*, pages 699–729. Springer, 2018.
3. X. Bonnetain, R. Bricout, A. Schrottenloher, and Y. Shen. Improved Classical and Quantum Algorithms for Subset-Sum. In S. Moriai and H. Wang, editors, *Advances in Cryptology - ASIACRYPT 2020 - 26th International Conference on the Theory and Application of Cryptology and Information Security, Daejeon, South Korea, December 7-11, 2020, Proceedings, Part II*, volume 12492 of *Lecture Notes in Computer Science*, pages 633–666. Springer, 2020.
4. J. H. Cheon, W. Cho, J. H. Kim, and J. Kim. Adventures in Crypto Dark Matter: Attacks, Fixes and Analysis for Weak Pseudorandom Function Candidates. *IACR Cryptol. ePrint Arch.*, 2020:783, 2020.
5. A. Esser, R. Kübler, and A. May. LPN Decoded. In J. Katz and H. Shacham, editors, *Advances in Cryptology - CRYPTO 2017 - 37th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 20-24, 2017, Proceedings, Part II*, volume 10402 of *Lecture Notes in Computer Science*, pages 486–514. Springer, 2017.
6. N. Howgrave-Graham and A. Joux. New Generic Algorithms for Hard Knapsacks. In H. Gilbert, editor, *Advances in Cryptology - EUROCRYPT 2010, 29th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Monaco / French Riviera, May 30 - June 3, 2010. Proceedings*, volume 6110 of *Lecture Notes in Computer Science*, pages 235–256. Springer, 2010.
7. D. Micciancio and M. Walter. On the Bit Security of Cryptographic Primitives. In J. B. Nielsen and V. Rijmen, editors, *Advances in Cryptology - EUROCRYPT 2018 - 37th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Tel Aviv, Israel, April 29 - May 3, 2018 Proceedings, Part I*, volume 10820 of *Lecture Notes in Computer Science*, pages 3–28. Springer, 2018.

A Analysis of Random Linear Codes

We analyze the distance of random linear codes defined by a random matrix or a random Toeplitz matrix. The analysis is based on the probabilistic method and similar to the analysis used to obtain the Gilbert–Varshamov bound.

Proposition A.1. *Let $B \in \mathbb{F}_q^{t \times m}$ be a random matrix, or a random Toeplitz matrix whose rows define a linear code. Then, the minimal distance of (the code defined by) B is at most $\ell < m/2$ with probability at most*

$$f_q(m,t,\ell) \le q^{t-m} \cdot Vol_q(m,\ell),$$

where $Vol_q(m,\ell) = \sum_{i=1}^{\ell} \binom{m}{i} \cdot (q-1)^i$. Moreover, this subspace contains two such linearly independent vectors with probability at most $(f_q(m,t,\ell))^2$.

Finally, let $H(p) = -p \log p - (1-p)\log(1-p)$ *be the binary entropy function. Then*

$$f_2(m,t,\ell) \le \ell \cdot 2^{m(H(\ell/m)-1)+t},$$

and

$$f_3(m,t,\ell) \le 2 \cdot 2^{m(H(\ell/m)-\log 3)+\ell+\log 3 \cdot t}.$$

Proof. The number of non-zero vectors of Hamming weight at most ℓ over $\mathbb{F}_q^m$ is $Vol_q(m,\ell) = \sum_{i=1}^{\ell} \binom{m}{i} \cdot (q-1)^i$.

For $q = 2$, we have $Vol_2(m,\ell) = \sum_{i=1}^{\ell} \binom{m}{i} < \ell \cdot \binom{m}{\ell} \le \ell \cdot 2^{mH(\ell/m)}$, while for $q = 3$, we have $Vol_3(m,\ell) = \sum_{i=1}^{\ell} \binom{m}{i} 2^i < 2 \cdot \binom{m}{\ell} 2^{\ell} \le 2 \cdot 2^{mH(\ell/m)+\ell}$.

Since B is selected uniformly from a pairwise independent hash family (either a random matrix or a random Toeplitz matrix), the probability that each such vector is in the row space is $q^{t-m} - 1 < q^{t-m}$. By a union bound over all vectors of Hamming weight bounded by ℓ, the probability that there exists such a vector in the row space of B is at most

$$f_q(m,t,\ell) \le q^{t-m} \cdot Vol_q(m,\ell).$$

The bound $(f_q(m,t,\ell))^2$ on the probability of having two such linearly independent vectors follows by pairwise independence.

Specifically, for $q = 2$, we obtain $f_2(m,t,\ell) \le \ell \cdot 2^{m(H(\ell/m)-1)+t}$, while for $t = 3$, we obtain $f_3(m,t,\ell) \le 2 \cdot 3^{t-m} \cdot 2^{mH(\ell/m)+\ell} = 2 \cdot 2^{m(H(\ell/m)-\log 3)+\ell+\log 3 \cdot t}$. ■

B Attack on the 2-3 OWF Based on Bilinearity

We describe an attack that in not more efficient than the one based on the subset-sum reduction, but will be relevant to variant of this scheme.

We may select a subset $I \subset [m]$ of $c \le t$ column indices of the matrix B such that these columns are linearly independent. Hence, the columns contain a $c \times c$ sub-matrix of full rank which can be used to nullify the remaining $t - c$ rows (restricted to the columns I) via Gaussian elimination. Therefore, we may define C of dimensions $(t-c) \times t$ such that CB restricted to c columns of I is a zero sub-matrix of dimensions $(t-c) \times c$. For the induced scheme $\mathrm{OWF}_C(\cdot)$, the values of the c indices I of w have no effect on the outcome $Cy = CBw$.

Bilinear sets. Assume that we can find sets $X \subseteq \{0,1\}^n$ and $X' \subseteq \{0,1\}^n$ such that any $(x,x') \in X \times X'$, satisfy the bilinear property $\mathrm{OWF}(x + x' \bmod 2) = \mathrm{OWF}(x) + \mathrm{OWF}(x') \bmod 3$.

If the sets are of size S, then given $\hat{y}$, we show below how to test whether there exists $x^* \in \{x + x' \bmod 2 \mid x \in X \wedge x' \in X'\} = X + X' \bmod 2$ such that $\mathrm{OWF}(x^*) = \hat{y}$. Since the time complexity of this algorithm is about S and the size of the set $X + X' \bmod 2$ is generally about k^2, the algorithm is faster than exhaustive search over the set $X + X' \bmod 2$ by a multiplicative factor of S.

The testing algorithm is as follows.

1. For each $x \in X$ evaluate $\mathrm{OWF}(x) = y$ and store the pairs (x, y) in a table $\mathcal{T}$, sorted by y values.
2. For each $x' \in X'$ evaluate $\mathrm{OWF}(x') = y'$ and search $\mathcal{T}$ for the value $\hat{y} - y' \bmod 3$. If there exists a match y such that $y = \hat{y} - y' \bmod 3$ (or $y + y' \bmod 3 = \hat{y}$), recover the value x such that $\mathrm{OWF}(x) = y$ from $\mathcal{T}$ and return $x + x' \bmod 2$.

The correctness of the algorithm follows directly from the bilinear property.

It remains to describe how to find sets X and X' that satisfy the bilinear property with sizes as large as possible. For this purpose, it will be convenient to work with an induced scheme $\mathrm{OWF}_C(\cdot)$ for an appropriate choice of a $r \times t$ matrix C of rank r such that $r \leq t$. Thus, we construct sets X and X' for $\mathrm{OWF}_C(\cdot)$ and use the above algorithm to find about 3^{t-r} preimages to $C\hat{y}$ for $\mathrm{OWF}_C(\cdot)$. Consequently, we expect at least one of them to also be a preimage of $\hat{y}$ for the original scheme $\mathrm{OWF}(\cdot)$.

Specifically, for a parameter c, we select a subset $I \subset [m]$ of c linearly independent columns of B and use them (as described above) to define a $(t-c) \times t$ matrix C such that the values of the c indices I of w have no effect on the outcome $Cy = CBw$ of $\mathrm{OWF}_C(\cdot)$. As a result, these c indices of w can be ignored and we are only concerned with the remaining $m - c$ indices.

We partition the $m - c$ indices of $[m] \backslash I$ arbitrarily into two sets I_1 and I_2 of equal size $(m-c)/2$. The set X is defined by solving the system $Ax = 0$ on the indices of I_1, while the set X' is defined by solving this system on the indices of I_2. Note that X and X' satisfy the bilinear property (on the indices $[m] \backslash I$ of w that influence the output). The size of each set is $2^{n-(m-c)/2}$, since the dimension of the solution space to the homogenous linear equation system is $n - (m-c)/2$.

Details of the preimage attack. The algorithm of the attack is given below.

1. Repeat until a preimage of $\hat{y}$ for $\mathrm{OWF}(\cdot)$ it found:
   (a) Randomly select a subset $I \subset [m]$ of c linearly independent columns of B and compute the $(t-c) \times t$ matrix C, as described above.
   (b) Randomly partition the indices of $[m] \backslash I$ into two subsets of size $(m-c)/2$ and use them to define the sets X and X' as described above.
   (c) For each $x \in X$ evaluate $\mathrm{OWF}_C(x) = y$ and store the pairs (x, y) in a table $\mathcal{T}$, sorted by y values.
   (d) For each $x' \in X'$ evaluate $\mathrm{OWF}_C(x') = y'$ and search $\mathcal{T}$ for the value $C\hat{y} - y' \bmod 3$. For all matches y such that $y = C\hat{y} - y' \bmod 3$ (or $y + y' \bmod 3 = C\hat{y}$), recover the value x such that $\mathrm{OWF}_C(x) = y$ from $\mathcal{T}$. Test whether $\mathrm{OWF}(x + x' \bmod 2) = \hat{y}$, and return $x + x' \bmod 2$ if equality hold.

Note that the preimages tested by the algorithm do not have any particular property (such has having low Hamming weight), hence we expect the algorithm to succeed for most values of $\hat{y}$ (that have a preimage) after testing $3^{t-r} = 3^c$ preimages (as $r = t - c$).

Time complexity analysis. For each iteration of the main loop, we compute $\mathrm{OWF}_C(\cdot)$ on the sets X and X' in time $2 \cdot 2^{n-(m-c)/2}$. In addition, as the output of $\mathrm{OWF}_C(\cdot)$ is in $\mathbb{F}_3^{t-c}$, the number of candidate preimages that we need to test on $\mathrm{OWF}(\cdot)$ is expected to be

$$2^{n-(m-c)/2} \cdot 2^{n-(m-c)/2} / 3^{t-c} = 2^{2n-m+c\cdot(\log 3+1)-\log 3 \cdot t}.$$

Optimizing the time complexity requires balancing the two terms, namely setting (ignoring the factor of 2 in the first term)

$$n-(m-c)/2 = 2n-m+c\cdot(\log 3+1)-\log 3 \cdot t,$$

or

$$c = \tfrac{-n+m/2+\log 3 \cdot t}{\log 3+1/2}. \tag{5}$$

In each iteration, we consider $2^{2n-(m-c)}$ possible preimages. With a choice of c as above, we actively test $2^{n-(m-c)/2}$ of these that satisfy the matching condition. Thus, we need to repeat the main loop $\frac{3^c}{2^{n-(m-c)/2}}$ times, where each iteration has time complexity of about $4 \cdot 2^{n-(m-c)/2}$. Hence the total time complexity is

$$\begin{aligned} &\tfrac{3^c}{2^{n-(m-c)/2}} \cdot 4 \cdot 2^{n-(m-c)/2} = \\ &\qquad\qquad 4 \cdot 3^c = \\ &2^{2+(-n+m/2+\log 3 \cdot t)\cdot \frac{\log 3}{\log 3+1/2}}. \end{aligned} \tag{6}$$

As we assume that $n = \log 3 \cdot t$, the complexity is

$$2^{2+m\cdot\frac{\log 3}{2\log 3+1}} \approx 2^{\frac{m}{2.63}}.$$