A New Code Based Signature Scheme without Trapdoors

Zhe Li [*] Chaoping Xing[†] Sze Ling Yeo[‡]

October 9, 2020

Abstract

We present a signature scheme for Hamming metric random linear codes via the Schnorr-Lyubashevsky framework that employs the rejection sampling on appropriate probability distributions instead of using trapdoors. Such an approach has been widely believed to be more challenging for linear codes as compared to lattices with Gaussian distributions. We prove that our signature scheme achieves EUF-CMA security under the assumption of the decoding one out of many problem or achieves strong EUF-CMA security under the assumption of the codeword finding problem under relaxed parameters. We provide an instantiation of the signature scheme based on Ring-LPN instances as well as quasi-cyclic codes and present some concrete parameters. In addition, a proof of concept implementation of the scheme is provided. We compare our scheme with previous unsuccessful similar attempts and provide a rigorous security analysis of our scheme.

Our construction primarily relies on an efficient rejection sampling lemma for binary linear codes with respect to suitably defined variants of the binomial distribution. Essentially, the rejection sampling lemma indicates that adding a small weight vector to a large weight vector has no significant effect on the distribution of the large weight vector. Concretely, we prove that if the large weight is at least the square of the small weight and the large weight vector admits binomial distribution, the sum distribution of the two vectors can be efficiently adjusted to a binomial distribution via the rejection step and independent from the small weight vector. As a result, our scheme outputs a signature distribution that is independent of the secret key.

Compared to two existing code based signature schemes, namely Durandal and Wave, the security of our scheme is reduced to full-fledged hard coding problems i.e., codeword finding problem and syndrome decoding problem for random linear codes. By contrast, the security of the Durandal and Wave schemes is reduced to newly introduced product spaces subspaces indistinguishability problem and the indistinguishability of generalized $(U, U+V)$ codes problem, respectively. We believe that building our scheme upon the more mature hard coding problems provides stronger confidence to the security of our signature scheme.

1 Introduction

In the recent Round 3 results of NIST post-quantum cryptosystems [NIS20], the Classic McEliece [BCL+17] code based cryptosystem is selected as one of the finalists, and BIKE [ABB+18] and HQC [MAB+19] are selected as two of the alternate candidates. In contrast, no code based signature scheme appears in the NIST Round 3 lists. In terms of encryption schemes, code

[*] School of Physical and Mathematical Sciences, Nanyang Technological University. Email: `lzonline01@gmail.com`
[†] School of Physical and Mathematical Sciences, Nanyang Technological University. Email: `xingcp@ntu.edu.sg`
[‡] School of Physical and Mathematical Sciences, Nanyang Technological University. Email: `yeoszeling@gmail.com`

based cryptography has provided acceptable options, whose security are reduced to syndrome decoding problems for quasi-cyclic Hamming codes or ideal rank-metric codes. However, with regards to signature schemes, the situation is completely different and designing secure code-based signature schemes from hard problems on random linear codes is always a goal in the coding community.

In general, there are two main methods to design signature schemes, including the hash-and-sign framework and the Fiat-Shamir transformation for identification protocols. The hash-and-sign framework was proposed in Diffie and Hellman's revolutionary paper [DH76]. The framework works as follows. Let f be a one-way trapdoor function, namely, it is hard to invert f without the trapdoor, and with the trapdoor, the preimage of f can be computed in polynomial time. To sign a given message, the signing algorithm hashes the message to an element in the range of f, inverts f using its trapdoor, and finally outputs the preimage of the hash value of the message as the signature of the message. For lattice and code based cryptography, the signing procedure involves running a decoding algorithm to find a closest vector for a given syndrome. Several schemes have been proposed in the literature, for instance, CFS [CFS01], RankSign [GRSZ14], Wave [DST19] in code based cryptography, and GGH [GGH97], NTRUSign [HPS01], GPV [GPV08] in lattice based cryptography. For CFS, RankSign, and Wave, the security of the scheme is built on the assumption that the underlying structural codes used are indistinguishable from random linear codes.

An alternative method to build signature scheme is to construct a three-move identification protocol and then convert the identification scheme to signature scheme via the Fiat-Shamir transformation [FS86, AABN02]. A well-known scheme following this approach is the Schnorr [Sch91] signature based on the hardness of discrete logarithm problems. The identification protocol consists of three steps. In the first step, the prover commits to a secret value and sends the commitment to the verifier. In the second step, the verifier binds the commitment and the message to a challenge, and sends the challenge to the prover. In the third step, the prover uses the secret value and challenge to hide the secret key to produce a response, and sends the response to the verifier. In the last step, the verifier checks that the response is consistent with the public key, commitment and the challenge. The Fiat-Shamir transformation replaces the second step by a cryptographic hash function, i.e, the challenge is binded to the public key and commitment via a hash function. Thus the prover is able to locally perform each step of the protocol and the protocol is converted to non-interactive. For number theoretic assumptions, the prover readily picks a random value to hide the secret key. In lattice and code based cryptography, it is easy to forge a random vector in the commitment and response for lattice and coding problems, and thus the secret vector to commit needs to be restricted to a subset of vectors, typically vectors which are *small* under the given metric. The challenge lies in efficiently hiding the secret key without leakage. Lyubashevsky [Lyu12, DDLL13] solves this problem for lattice problems in a sequence of papers via the *rejection sampling* technique. Different from traditional identification protocol, Lyubashevsky uses rejection sampling to alter the response distribution to decouple the dependence of response on the secret key. In what follows, we refer to the identification protocol with rejection sampling as Schnorr-Lyubashevsky framework. There are several attempts to adapt the Schnorr-Lyubashevsky framework to coding problems, for instance RaCoSS [FRX$^+$17], RaCoSS-R [RMF$^+$18], Persichetti's proposal [Per18], Durandal [ABG$^+$19]. So far, none of the code based schemes proves that the achieved signature distribution is independent from the secret key and reveals nothing on the secret key as achieved by the lattice-based signature counterparts.

1.1 Related Work

For provable secure signature schemes in the random oracle model, the goal is to prove that the output signature distribution is independent of the secret key. Once this goal is achieved, the challenger can output valid signatures on requesting via programming the random oracle in the security game and the forger's ability to forge a valid signature is converted to the ability to output a solution to the underlying hard problem.

In code based cryptography, there are two famous hard problems, namely the syndrome decoding problem and the codeword finding problem. For parameters (n, k, w, q), given a parity-check matrix $H \in \mathbb{F}_q^{(n-k)\times n}$ and a syndrome $\mathbf{s} \in \mathbb{F}_q^{n-k}$, the syndrome decoding problem asks to find a vector $\mathbf{e} \in \mathbb{F}_q^n$ such that $\mathrm{wt}(\mathbf{e}) \leq w$ and $H\mathbf{e} = \mathbf{s}$. Taking $\mathbf{s} = \mathbf{0}$, the problem becomes the codeword finding problem. Both these two problems are proven to be NP-hard [BMvT78, Var97]. Code based cryptography is based on the conjecture that the two problems are hard for random linear codes. For instance, the intractability assumption for random linear codes is used in two of the NIST Round 3 alternate candidates, namely, BIKE and HQC. In general, the easy range for the weight w of the two problems is $[\frac{q-1}{q}(n-k), \frac{(q-1)n}{q} + \frac{k}{q}]$. Throughout this paper, we only consider the case $q = 2$, namely, binary linear codes.

The first proposed code based signature scheme is the CFS scheme [CFS01]. The scheme relies on the decoding capability of high rate Goppa codes. Via choosing high rate Goppa codes, a non-negligible fraction of the syndromes can be decoded to the nearest codeword. The security of the scheme builds on the assumption that the chosen Goppa codes are indistinguishable from random linear codes. However, for high rate Goppa codes, a distinguisher was proposed [FGO$^+$13]. In 2014, a signature scheme with the name RankSign in the rank metric setting was proposed [GRSZ14]. The security of RankSign also builds on the assumption that the special codes are indistinguishable from random linear rank metric codes. Later, a structural key-recovery attack was reported in [DT18]. Via adapting the identification protocol to code based cryptography, the Random Code-based Signature Scheme(RaCoSS) scheme was submitted to NIST [FRX$^+$17]. The RaCoSS scheme builds upon random linear codes. However, the scheme was attacked [BHLP17] two days after the submission. Subsequently, the scheme was patched [RMF$^+$18] and attacked [Xag18] again. The main problem of RaCoSS and the patched version is that the weight of valid signatures is large. In particular, the weight range of valid signatures intersects with the easy range of the syndrome decoding problem. Thus, an adversary can directly forge a valid signature for any message without the secret key. In 2018, Persichetti [Per18] adapted the Lyubashevsky scheme to random quasi-cyclic Hamming metric codes. Persichetti's proposal claimed the security of one-time signature. In Persichetti's scheme, the weight of a valid signature is below the GV bound of the code. Thus, it does not suffer from the weakness of RaCoSS. Two subsequent independent works [SBC19, DG20] attacked the scheme. The first attack [DG20] employs the LDPC(MDPC) decoding algorithm to recover the secret key via viewing the challenge vector as a LDPC code and the signature vector as a syndrome of LDPC code. As the weight of the signature is below the GV bound, thus the weight of the secret vector in the commitment phase is below the GV bound as well. Concretely, because the challenge vector is sparse, thus the weights of the secret key and the secret vector in the commitment are in the decoding capability of the corresponding LDPC code. In an independent work, [SBC19] recovered the secret key from one signature via a statistical attack. The statistical attack makes full use of the sparsity of the secret commit vector. Thus, different cyclic rotations on the secret key and secret commit vector have no intersection with non-negligible probability. Both of the two attacks owe the insecurity of Persichetti's scheme to the sparsity of the challenge vector. However, in order to achieve a sufficiently small vector, the challenge vector must necessarily be sparse. Such constraints seem to suggest that it is infeasible to construct a signature scheme via the Schnorr-Lyubashevsky framework in the linear codes setting. In 2018, following the Schnorr-Lyubashevsky framework, Anguil et al. [ABG$^+$19] proposed a signature scheme with the name Durandal in the rank metric context. However, the security of Durandal builds on a complicated new problem PSSI$^+$. It is not proven that the signature distribution is independent from the secret key. In Durandal, the signature is rerandomized by an extra vector to hide the secret key. It is pointed out that the rerandomizing strategy is not easy to be employed on codes with respect to the Hamming metric and it is difficult to appropriately hide the secret key for Hamming metric codes. In 2019, the Wave [DST19] signature scheme was proposed, which follows the hash-and-sign framework. The security of Wave builds on the new assumption that generalized $(U, U+V)$ codes are independent from random linear codes. However, the hardness of distinguishing generalized $(U, U+V)$ codes from random linear codes is still unclear. Interestingly, the output signature is proven to be independent from the secret key. Different from traditional code based cryptography, the Wave signature chooses the desired weight of the underlying syndrome decoding problem to lie at the large end.

In summary, the existing secure code based signature schemes build the security on immature intractability assumptions. Thus the problem to construct a signature scheme, whose security relies on mature assumption like the syndrome decoding problem or the codeword finding problem, is a highly nontrivial task.

1.2 Our Results and Techniques

In this paper, the main contribution is to construct a code based signature scheme via adapting the Schnorr-Lyubashevsky framework to Hamming metric linear codes in the random oracle model. The EUF-CMA security of the signature scheme is reduced to the Decoding One Out of Many (DOOM) problem, which is a multi-syndrome variant of the syndrome decoding problem. For relaxed parameters, the *strong* EUF-CMA security of the scheme is reduced to the codeword finding problem. Via a rejection sampling lemma, the signature distribution is proven to be independent of the secret key. To establish the rejection sampling lemma for binary linear codes, we define the *truncated binomial* distribution and the *shifted binomial* distribution. With these variants of the binomial distribution, we establish an efficient rejection sampling lemma. Concretely, we prove that adding a small weight vector to a large weight vector has no significant impact on the distribution of the large weight vector. With the rejection sampling lemma, we construct the signature scheme. To obtain a practical signature scheme, we instantiate the signature scheme via quasi-cyclic codes and provide parameters for classical 80 and 128 bit security. To provide a proof of concept of the instantiation, an implementation of the scheme is given. We show that all possible known attacks on previous unsuccessful attempts pose no threat to our scheme.

We briefly sketch our results and techniques. We first describe the probability distribution involved in the rejection sampling lemma. Let $\mathcal{B}_p^n$ be the binomial distribution over $\mathbb{F}_2^n$ with each entry sampled from a Bernoulli distribution parameterized by a constant p. The density function of $\mathcal{B}_p^n$ is defined as $\mathcal{B}_p^n(t) := \binom{n}{t} p^t (1-p)^{n-t}$. It follows from the Hoeffding bound that the Hamming weight of random vectors equipped with a binomial distribution is around np. Given $0 \leq a < b \leq n$, the *truncated binomial* distribution $\widetilde{\mathcal{B}}_{a,b,p}^n$ is defined as $\widetilde{\mathcal{B}}_{a,b,p}^n(t) := \frac{\mathcal{B}_p^n(t)}{\sum_{j=a}^b \mathcal{B}_p^n(j)}$ for $t \in [a, b]$. The density function of the truncated binomial distribution is proportional to $\mathcal{B}_p^n$. The sum $\sum_{j=a}^b \mathcal{B}_p^n(j)$ is the normalization factor. If the parameters (a, b) are defined to be symmetric around np, we use $\widetilde{\mathcal{B}}_{\xi,p}^n$ to denote the truncated binomial distribution where $\xi = np - a = b - np$. Given a truncated binomial distribution $\widetilde{\mathcal{B}}_{a,b,p}^n$, one is only interested in the vector of weight lying in the range $[a, b]$. Given a vector $\mathbf{v} \in \mathbb{F}_2^n$, the *shifted binomial* distribution is defined as $\mathcal{B}_{\mathbf{v},p}^n(\mathbf{x}) := \mathcal{B}_p^n(\mathbf{x} - \mathbf{v})$, i.e., $\mathbf{x} - \mathbf{v}$ admits the binomial distribution $\mathcal{B}_p^n$. More precisely, the shifted binomial distribution is obtained by adding a given vector $\mathbf{v}$ to a binomial distribution random variable. Let $\mathrm{wt}(\mathbf{v}) = s$, $\mathrm{wt}(\mathbf{x}) = t$. For $s \leq t$, we prove that

$$\mathcal{B}_{\mathbf{v},p}^n(t) := \sum_{j=0}^{s} \binom{s}{j} \binom{n-s}{t-j} p^{s+t-2j} (1-p)^{n-s-t+2j}.$$

Note that in the above binomial distribution variations, the density function is only associated to the Hamming weight of the involved vectors rather than the vector itself. The main result of the binomial distribution is the ratio of the density functions of binomial distribution and shifted binomial distribution. For $\xi = o(np(1-p))$ and $s = o(np(1-p))$ with p being a constant, we prove

$$\frac{\widetilde{\mathcal{B}}^n_{\xi,p}(t)}{\mathcal{B}^n_{\mathbf{v},p}(t)} \approx \mathrm{e}^{\frac{s^2}{n}\frac{(1-2p)^2}{p(1-p)} - \frac{(t-pn)s}{n}} \frac{\xi}{(ns^2p(1-p))^{1/4}}.$$

Thus, if we take $s^2 < n$ and $s\xi < n$, the ratio is close to $\frac{\xi}{(ns^2p(1-p))^{1/4}}$. The proof of the distribution result is provided in Section 3 via a series of approximations to the binomial coefficients. Intuitively, the result implies that if the weight of the shifted vector is small enough compared to the expected weight of a random vector of variables, then adding the vector to the random vector has no significant impact on the distribution of the resulting vector. From the view of the curve of the density function, the result indicates that under the condition $s^2 < n$ and $s\xi < n$, the truncated binomial distribution and shifted binomial distribution have a large overlapped part. Using the above ratio result of the density function of the binomial distribution variants, we are able to prove a rejection sampling lemma, which is essential to our signature scheme.

Rejection sampling is a general method to tune a real probability distribution g to a desired probability distribution f, which was first introduced by von Neumann [vN51]. The algorithmic procedure works as follows. For a random variable v admitting the distribution g, a procedure outputs v with probability $\frac{f(v)}{g(v)M}$, where $M := max_u \frac{f(u)}{g(u)}$ is the normalization factor of the output probability. The procedure is repeated until a sample is output. The output variable of the rejection sampling method admits the probability distribution f follows from the conditional probability formula. In addition, the expected repetition number of the rejection procedure is M. To make the rejection sampling more efficient, we expect the value of M to be as small as possible. The rejection sampling method was used in Lyubashevsky's sequel of work [Lyu12, DDLL13] and the Wave [DST19] scheme to decouple the dependence of the output signatures on secret key. Concretely, the signature is proven to admit a predefined probability distribution characterized by the parameters of the distribution itself and independent from the secret key.

We state our rejection sampling lemma for binary linear codes. Given a subset $V \subseteq \mathbb{F}_2^n$, let $h : V \to \mathbb{R}$ be a probability distribution. Let $M := max_{t\in[np-\xi,np+\xi]} \frac{\widetilde{\mathcal{B}}^n_{\xi,p}(t)}{\mathcal{B}^n_{\mathbf{v},p}(t)}$. Then the output distribution of the following two algorithms are identical.

- Sample $\mathbf{v}$ from h, $\mathbf{z}$ from $\mathcal{B}^n_{\mathbf{v},p}$, output $(\mathbf{z},\mathbf{v})$ with probability $\frac{\widetilde{\mathcal{B}}^n_{\xi,p}(\mathbf{z})}{M\mathcal{B}^n_{\mathbf{v},p}(\mathbf{z})}$.
- Sample $\mathbf{v}$ from h, $\mathbf{z}$ from $\widetilde{\mathcal{B}}^n_{\xi,p}$, output $(\mathbf{z},\mathbf{v})$ with probability $\frac{1}{M}$.

With the rejection sampling lemma, we successfully design a signature scheme for binary linear codes. The signature scheme is presented in Algorithm 1, Algorithm 2 and Algorithm 3. The secret key is a binary matrix $S \in \mathbb{F}_2^{n\times\ell}$, where each entry of S is sampled from a Bernoulli distribution parameterized by σ, where σ is a very small parameter related to the security level of the scheme. A uniformly random sampled matrix $H \in \mathbb{F}_2^{(n-k)\times n}$ and $T = HS \in \mathbb{F}_2^{(n-k)\times\ell}$ form the public key pair (H,T). Let $\mathcal{H}$ be a cryptographic hash function, whose output is a vector of weight w. To sign a message μ, the signing algorithm first samples a vector $\mathbf{e}$ according to the binomial distribution $\mathcal{B}^n_\tau$ with τ being a constant and computes $\mathbf{y} = H\mathbf{e} \in \mathbb{F}_2^{n-k}$. The signing algorithm uses the hash function $\mathcal{H}$ to compute $\mathbf{c} = \mathcal{H}(\mathbf{y},\mu)$. The weight of the output of the hash function is fixed to a number w. The cryptographic hash function is provided in Section 6.4. Next, the signing algorithm computes $\mathbf{z} = S\mathbf{c} + \mathbf{e}$. Note that the value $\mathbf{z}$ admits the distribution $\mathcal{B}^n_{S\mathbf{c},\tau}$ according to the definition of $\mathbf{z}$. Then the signing algorithm outputs $(\mathbf{z},\mathbf{c})$ as a signature of μ with probability $\frac{\widetilde{\mathcal{B}}^n_{\xi,\tau}(\mathbf{z})}{M\mathcal{B}^n_{S\mathbf{c},\tau}(\mathbf{z})}$. Using the rejection sampling lemma, the distribution of the signature vector $\mathbf{z}$ is adjusted to the truncated distribution $\widetilde{\mathcal{B}}^n_{\xi,\tau}$, which is characterized by the distribution parameter tuple (n,τ,ξ). Note that before the rejection sampling step, the signing algorithm checks whether the weight of $\mathbf{z}$ lies in the range $[n\tau-\xi, n\tau+\xi]$. If

$\mathrm{wt}(\mathbf{z}) \notin [n\tau - \xi, n\tau + \xi]$, the signing algorithm restarts by drawing another vector $\mathbf{e}$. Here the rejection happens before applying the rejection sampling lemma. According to the piling-up lemma, the vector $S\mathbf{c}$ admits the distribution $\mathcal{B}_\eta^n$, where $\eta = \frac{1}{2}(1 - (1 - 2\sigma)^w) \approx \sigma w$ for very small σw. From the Hoeffding bound, the weight of $S\mathbf{c}$ is around $n\eta$. To minimize the value M in the rejection sampling lemma, n is set at least $(n\eta)^2$.

The verifying algorithm checks that $\mathrm{wt}(\mathbf{z}) \in [n\tau - \xi, n\tau + \xi]$ and $\mathbf{c} = \mathcal{H}(H\mathbf{z} - T\mathbf{c}, \mu)$. The two conditions hold according to the signing algorithm.

Note that if we instantiate the scheme with random linear codes, the public key size is huge. According to the rejection sampling lemma, the weight of $S\mathbf{c}$ is at most $\sqrt{n}$. Thus, each column of S is much smaller. For sublinear error weight v, the information set decoding has complexity $2^{v \log \frac{n}{n-k}}$. To achieve λ bit security, we have $n \geq \lambda^2$. To keep the code rate a constant, then the public key size is of $O(n^2) = O(\lambda^4)$. For instance $\lambda = 128$, the public key size is at least $128^4 = 2^{28}$ bits. Thus, we instantiate the signature scheme with quasi-cyclic random codes or Ring-LPN instances. In quasi-cyclic codes, the parity-check matrix is represented by a constant number of vectors and thus the public key size is tremendously reduced. The quasi-cyclic codes have been used for many years in code based cryptography to reduce the public key size without compromising the security level significantly. In particular, quasi-cyclic codes are both employed in BIKE and HQC, two of the NIST Round 3 alternate candidates.

In the security proof, the challenger answers the signing queries via programming the random oracle. By replacing the hash queries with the syndromes of a given DOOM instance, the forger is forced to solve one of the syndrome decoding problem of DOOM. For relaxed parameters, the challenger forces the forger to solve the codeword finding problem for the code parity checked by the matrix H via the general forking lemma. The security proof also works for the quasi-cyclic instantiation.

Assume we use the ring $\mathcal{R} := \mathbb{F}_2[x]/(x^n - 1)$ to instantiate the signature scheme. For simplicity, we consider the case for code rate being $1/2$. Then the key generation step becomes $\mathbf{h}\mathbf{s}_1 + \mathbf{s}_2 = \mathbf{t}$, where $\mathbf{h}, \mathbf{s}_1, \mathbf{s}_2, \mathbf{t} \in \mathcal{R}$, and $\mathbf{s}_1$ and $\mathbf{s}_2$ are of very small Hamming weight. The secret key pair is $(\mathbf{s}_1, \mathbf{s}_2)$ and the public key pair is $(\mathbf{h}, \mathbf{t})$. To make it easy to estimate the secret key security, we fix the weight of $(\mathbf{s}_1, \mathbf{s}_2)$ to be $2u$ for some u. Although the quasi-cyclic code structure does not endanger the security level too much, we need to take it into account as the value n is large. For the syndrome decoding problem $(\mathbf{h}, \mathbf{1})\begin{pmatrix}\mathbf{s}_1 \\ \mathbf{s}_2\end{pmatrix} = \mathbf{t}$, any solution to the syndrome $\mathbf{t}x^j$ corresponds to the desired solution of the quasi-cyclic syndrome decoding problem, where $\mathbf{t}x^j$ denotes the result of applying j cyclic rotations to $\mathbf{t}$. Thus the quasi-cyclic code syndrome decoding problem is converted to a DOOM instance with n syndromes. From the result of [Sen11], one has a speedup factor of $\sqrt{n}$. Thus, we will set $2u \geq \log \sqrt{n} + \lambda$. To further reduce the public key size, we employ the codeword finding problem in key generation. Taking $\mathbf{t} = \mathbf{0}$, the key generation procedure becomes $\mathbf{h} = \frac{\mathbf{s}_2}{\mathbf{s}_1}$ and the public key becomes $\mathbf{h}$. Thus the security of key generation is associated to the codeword finding problem. This assumption is also employed in the key generation procedure of NIST Round 3 alternate candidate BIKE [ABB+18]. For codeword finding problem, the DOOM instance provides a speedup factor of n. So we choose $2u \geq \lambda + \log(n)$. In the commitment step, the signing algorithm draws a vector $\mathbf{e} = (\mathbf{e}_1, \mathbf{e}_2)$ from the distribution $\mathcal{B}_\tau^{2n}$ and computes $\mathbf{y} = \mathbf{h}\mathbf{e}_1 + \mathbf{e}_2$. Note that in our given parameters, the ring degree is large. Thus computing $\mathbf{h}\mathbf{e}_1$ over $\mathcal{R}$ is performed at great expense. The multiplication can be optimized by two methods. Readers refer to Section 6.3.2 for details.

Then the signing algorithm computes $\mathbf{c} = \mathcal{H}(\mathbf{y}, \mu)$. Next, the signing algorithm computes $\mathbf{z} := (\mathbf{s}_1, \mathbf{s}_2)\mathbf{c} + (\mathbf{e}_1, \mathbf{e}_2)$. As $\mathbf{c}$ and $(\mathbf{s}_1, \mathbf{s}_2)$ are both sparse vectors, computing the sum of $\mathrm{wt}(\mathbf{c})$ cyclic rotations of $(\mathbf{s}_1, \mathbf{s}_2)$ can finish the costly ring elements multiplication task. Since the $\mathbf{s}$ and $\mathbf{c}$ are both sparse vectors, the weight of $(\mathbf{s}_1, \mathbf{s}_2)\mathbf{c}$ will be $\mathrm{wt}(\mathbf{s}_1, \mathbf{s}_2)\mathrm{wt}(\mathbf{c})$ with high probability. The rejection sampling vector can be precomputed as the rejection sampling lemma is only sensitive to the Hamming weight of involved vectors. There are two rejections in the signing algorithm. One is the rejection sampling step and the other one is to ensure that $\mathrm{wt}(\mathbf{z})$ lies in the range $[n\tau - \xi, n\tau + \xi]$. The vector $\mathbf{c}$ in the signature can be compressed since it is sparse. The verifying algorithm checks that $\mathrm{wt}(\mathbf{z}) \in [n\tau - \xi, n\tau + \xi]$ and $\mathbf{c} = \mathcal{H}(\mathbf{h}\mathbf{z}_1 + \mathbf{z}_2 - \mathbf{t}\mathbf{c}, \mu)$.

Remark 1.1.

- *If we choose $s^2 < n$ and $s\xi < n$, the rejection rate will be very small. But the signature size and public key size are of very large. So we reduce the public key size and signature size at the cost of increasing the rejection rate. We expect future work to provide a better rejection sampling lemma to improve the square condition in the rejection sampling lemma to achieve a smaller rejection rate and code length.*
- *One can choose a smaller code rate to allow a large parameter τ, even though the parameter τ is viewed as a constant in the rejection sampling lemma. However, for practical parameters, larger τ allows a smaller rejection rate. Thus, one can choose a larger τ to reduce the code length to keep an acceptably small rejection rate. But for smaller code rate, the security for secret key is compromised and thus one needs to increase the weight of secret key to keep the desired security level. A rough inspection indicates that choosing a smaller code rate has no significant effect to the code length. But a smaller code rate leads to a small degree of the ring $\mathcal{R}$ and thus makes the multiplication operation over the ring $\mathcal{R}$ more efficient.*

In our instantiation of the signature scheme, we will choose an appropriate range parameter ξ to minimize the whole rejection rate for the signature scheme. Concrete parameters are given in Section 6.7. A proof of concept implementation of the instantiation is provided.

1.3 Comparison with Existing Secure Code based Signature Schemes

Compared to secure code based signature schemes Durandal and Wave, the security of our signature scheme is reduced to well-known decoding problems DOOM for random linear codes or codeword finding problem for random linear codes, whereas the security of Durandal relies on rank support learning problem, the Advanced Product Spaces Subspaces Indistinguishability (PSSI$^+$) problem, and the security of Wave relies on the indistinguishability of generalized $(U, U+V)$ codes from random linear codes and the DOOM problem. The PSSI$^+$ problem and the distinguishing problem for generalized $(U, U+V)$ codes are both new introduced by the Durandal scheme and the Wave scheme, respectively. The hardness of the two problems remains untested and further studies on the hardness of the two problems are needed. We believe that our scheme which builds its security on the well-known codeword finding problem for random linear codes or DOOM problem for random linear codes ensures a more convincing security. In contrast to Durandal, the signature distributions of Wave and our scheme are both proven to be independent from the secret key and thus lead to no leakage on the secret key.

In the computation part, the Durandal scheme has filter space operation, which is expensive. In the Wave, the signing algorithm runs two information set decoding algorithms. Our scheme involves only vector or ring multiplication and addition operations. Even though, the ring element multiplication is costly for our large parameters, the operation can still be optimized.

The disadvantage of our scheme is that the key size and signature size are still large compared to the Durandal and Wave scheme. The large key size and signature size stem from the exponent 2 in the relation $s^2 < n$ in the rejection sampling lemma. Employing a large code length makes the rejection sampling step more efficient. We believe that there is potential for further research to design other probability distributions to improve the rejection sampling results, and thereby reducing the key sizes.

1.4 Rejection Rate

In our construction, we introduce the truncated binomial distribution. If a value $\mathrm{wt}(\mathbf{z})$ leads to a very large rejection rate, then the vector $\mathbf{z}$ is directly rejected via the weight checking condition $\mathrm{wt}(\mathbf{z}) \in [n\tau - \xi, n\tau + \xi]$. This is different from the rejection sampling lemma used in Lyubashevsky's signature scheme. Via choosing appropriate ξ, the value M is bounded by a smaller value. In our instantiation of the signature scheme, we will choose ξ to achieve the smallest whole rejection rate for the signature scheme.

As we need to check $\mathrm{wt}(\mathbf{z})$ to decide whether to reject or admit it into the rejection sampling step, we need to compute the probability that $\mathrm{wt}(\mathbf{z}) \in [n\tau-\xi, n\tau+\xi]$. If the range $[n\tau-\xi, n\tau+\xi]$ is large enough, we can use the Hoeffding bound to estimate the probability of $\mathrm{wt}(\mathbf{z}) \in [n\tau - \xi, n\tau + \xi]$. From the rejection sampling lemma, we need to choose ξ such that $s\xi < n$. Thus we employ a counting formula to compute the probability. In practice, we will use the counting formula to compute the probability that $\mathrm{wt}(\mathbf{z}) \in [n\tau-\xi, n\tau+\xi]$ and the exact rejection sampling probability. For well-chosen parameters, the global rejection rate becomes controllably small.

1.5 Comparison with Unsuccessful Attempts

Compared to RaCoSS and RaCoSS-R, the weight of the signature vector in our scheme belongs to the small end of the hard range of the syndrome decoding problem. Thus, our scheme thwarts the attacks to RaCoSS and RaCoSS-R. Compared to Persichetti's proposal, our instantiation employs a large weight vector in commitment to hide the secret key part in the signature and thus leads to the weight of the signature also being large. Concretely, in Persichetti's proposal, the weight of signature is required to be below the GV bound and thus the weights of secret key and the vector to commit are both below the GV bound. Therefore, for Persichetti's proposal, the secret vector in commitment cannot properly hide the secret key in the signature. It seems the main difference in parameter setting between Persichetti's proposal and our signature scheme is that we choose a large weight vector to commit and produce a large weight signature. Essentially, choosing a large weight vector to commit, using the large weight vector to randomized the secret key and outputting a large weight signature completely change the nature of the signature scheme. First according to the rejection sampling lemma, choosing a large weight vector to commit makes it possible to efficiently tune the output distribution to completely remove the dependence of signature distribution on the secret key. If an adversary is able to run the LDPC decoding attack or the statistical attack, then the adversary is able to obtain the same information from the truncated binomial distribution. Thus, any information the adversary obtains from the signatures is fully independent from the secret key. Note that even one does not care about huge key size and huge rejection rate, choosing the weight of the response vector to be below the GV bound of the code is insecure. This is because for the weight less than GV bound, with high probability the syndrome decoding problem has a unique solution, so that no distribution is involved in the output of the signature scheme. Thus the rejection sampling cannot be leveraged to adjust the distribution of the output signature as the response is inherently decided by the public key, commitment and challenge. On the other hand, the existing attack on the one-time signature exactly verifies this point that an output signature fully reveals the secret key.

Second, large weights of the signature vector and the vector to commit thwart the existing attacks on our scheme. The LDPC decoding attack does not work for our large weight signature scheme. In our signature scheme, the weights of the vector to commit and signature are just a little smaller than the singleton bound of the code. Thus it is far beyond the decoding capability of LDPC. Our scheme is also resistant to the statistical attack. In general, the statistical attack works as follows. Given a signature $(\mathbf{z}, \mathbf{c})$, then $\mathbf{z} = \mathbf{sc}+\mathbf{e}$ and it is $(\mathbf{z}_1, \mathbf{z}_2) = (\mathbf{s}_1\mathbf{c}+\mathbf{e}_1, \mathbf{s}_2\mathbf{c}+\mathbf{e}_2)$. Consider the first element $\mathbf{z}_1 = \mathbf{cs}_1 + \mathbf{e}_1 = (\mathbf{c}, 1)\begin{pmatrix}\mathbf{s}_1\\ \mathbf{e}_1\end{pmatrix} = \sum_{i\in\mathrm{Supp}(\mathbf{c})} x^i\mathbf{s}_1 + \mathbf{e}_1$. Multiplying $\mathbf{z}_1$ by x^{-j} for $j \in \mathrm{Supp}(\mathbf{c})$, one has

$$\mathbf{z}_1^j := \mathbf{z}_1 x^{-j} = \mathbf{s}_1 + \sum_{i\in\mathrm{Supp}(\mathbf{c}), i\neq j} \mathbf{s}_1 x^{i-j} + \mathbf{e}_1 x^{-j}.$$

Here $\mathbf{s}_1x^{i-j}$ and $\mathbf{e}_1x^{-j}$ are cyclic rotations of $\mathbf{s}_1$ and $\mathbf{e}_1$ respectively. Because $\mathrm{wt}(\mathbf{s}_1)$ and $\mathrm{wt}(\mathbf{c})$ are both below the GV bound, the support of $\mathbf{s}_1$ has no intersection with the supports of $\mathbf{s}_1x^{i-j}$ and $\mathbf{e}_1x^{-j}$ with high probability and the expectation of the support intersection number is very small. The attack lifts the polynomial $\mathbf{z}_1^j$ from $\mathbb{F}_2[x]$ to $\mathbb{Z}[x]$ and computes the sum of lifted polynomials. Most of the coefficients of $\mathbf{s}_1$ pile up in the sum of the lifted polynomials as $\mathbf{s}_1$ appears in every lifted polynomials and the expectation the support intersection is very small. With non-negligible probability, this step can directly recover the secret key $\mathbf{s}_1$. Even if this step does not fully recover $\mathbf{s}_1$, the recovered secret key $\mathbf{s}_1^*$ is close to $\mathbf{s}_1$. Namely, $\mathbf{s}_1 - \mathbf{s}_1^*$ is of very small weight. Then running the information set decoding algorithm to fully recover the secret key from public key and $\mathbf{s}_1^*$ takes a very small work factor. In our scheme, for the case of large weight $\mathbf{e}_1$, the support of $\mathbf{s}_1$ has overlap with the support of $\mathbf{e}_1$ with non-negligible probability. For distinct $\mathbf{z}_1^j$, $\mathbf{e}_1x^{-j}$ intersects with $\mathbf{s}_1$ and previous $\mathbf{e}_1x^{-\ell}$ with non-negligible probability. Thus, the sum of lifted polynomials contains many coefficients that pile up from part of supports of $\mathbf{s}_1$ and $\mathbf{e}_1x^{-j}$. One is unable to tell apart which is from the secret key and which is from $\mathbf{e}_1x^{-j}$. For our instantiation of the signature scheme, the complexity to directly recover $\mathbf{s}_1$ from the signature $\mathbf{z}_1$ is beyond the security level. Subtracting recovered $\mathbf{e}_1^*$ from the signature also has no help to recover the secret key as the unknown part of $\mathbf{e}_1 - \mathbf{e}_1^*$ is still of large weight.

In general, the security of our scheme hinges on the proof that the signature distribution is identical to a predefined truncated binomial distribution. Thus, the output signature reveals nothing about the secret key. An adversary is unable to infer the secret key from one signature as in the attacks on Persichetti's proposal. Even given a polynomial number of signatures, an adversary is still unable to learn any information on the secret key from the signatures because the signatures can be viewed as being directly sampled from the truncated binomial distribution. The security of our signature scheme is reduced to the well-known codeword finding problem for random linear codes or DOOM problem for random linear codes.

1.6 Organization

We recall coding problems, some probability bounds, approximations for binomial coefficients in Section 2. In Section 3, variants of binomial distribution are introduced and the main rejection sampling lemma is proved. Section 4 presents the our signature scheme according to the rejection sampling lemma. The security proof of the signature scheme is provided in Section 5. To make the scheme practical, we instantiate the signature scheme on quasi-cyclic code or Ring-LPN instances in Section 6 and concrete parameters are also provided. We compare our scheme with some unsuccessful attempts in Section 7. Finally, we conclude the paper with some possible methods to improve the rejection sampling lemma.

2 Preliminaries

Notations. We use $\log(\cdot)$ to denote the base 2 logarithm function and $\ln(\cdot)$ the natural logarithm function. We use $\mathcal{S}_w^n \subseteq \mathbb{F}_2^n$ to denote the set of all binary vectors of weight w and length n.

2.1 Coding Theory

We review some basic definition on coding theory. For a comprehensive introduction on coding theory and coding problems, we refer the reader to [MS77].

Definition 2.1 (Linear Codes). *Given integers n and k, a binary linear $[n, k]$ code $\mathcal{C}$ is a k-dimensional subspace of $\mathbb{F}_2^n$. Each element of $\mathcal{C}$ is denoted as codeword.*

Definition 2.2 (Generator Matrices and Parity Check Matrices). *Given integers n and k, a matrix $G \in \mathbb{F}_2^{k\times n}$ is a generator matrix of a linear code $\mathcal{C}$ if $\mathcal{C} := \{\mathbf{m}^T G \in \mathbb{F}_2^n : \mathbf{m} \in \mathbb{F}_2^n\}$. A matrix H is a parity check matrix of $\mathcal{C}$ if $\mathcal{C} := \{\mathbf{c} \in \mathbb{F}_2^n : H\mathbf{c} = \mathbf{0}\}$.*

Definition 2.3 (Syndrome). *Given integers n, k, a parity check matrix $H \in \mathbb{F}_2^{(n-k)\times n}$ of a linear code $\mathcal{C}$, and a vector $\mathbf{e} \in \mathbb{F}_2^n$, the syndrome of $\mathbf{e}$ under H is defined as $\mathbf{s} := H\mathbf{e}$.*

For any linear code $\mathcal{C}$, the matrix representation has the drawback of large key size. Thus, researchers propose to use *quasi-cyclic* codes to reduce the public key size without compromising the security level too much.

Before giving the definition of quasi-cyclic codes, we define the ring of polynomials and the circulant matrix. Given integers n, define the ring $\mathcal{R} := \mathbb{F}_2[x]/(x^n - 1)$. Then there is an isomorphism from $\mathbb{F}_2^n$ to $\mathcal{R}$. Namely, every vector of $\mathbb{F}_2^n$ is mapped to a polynomial in $\mathcal{R}$ via the *coefficient embedding*. We use the isomorphism between $\mathbb{F}_2^n$ and $\mathcal{R}$ to define the circulant matrix.

Definition 2.4 (Circulant Matrix). *Given a vector $\mathbf{h} = (h_0, \ldots, h_{n-1}) \in \mathbb{F}_2^n$, the circulant matrix of $\mathbf{h}$ is defined as*

$$rot(\mathbf{h}) := \begin{pmatrix} h_0 & h_1 & \ldots & h_{n-1} \\ h_{n-1} & h_0 & \ldots & h_{n-2} \\ \vdots & \vdots & \ddots & \vdots \\ h_1 & h_2 & \ldots & h_0 \end{pmatrix}.$$

Each row of $rot(\mathbf{h})$ is the cyclic rotation of the previous row when $\mathbf{h}$ is viewed as row vector. From the perspective of $\mathcal{R}$, the i-th row of $rot(\mathbf{h})$ is $x^i h(x)$.

For any two elements $a(x), b(x)$ of $\mathcal{R}$, the product of ring elements is the product of the polynomials modulo the polynomial $x^n - 1$. From the point of view of matrix vector multiplication, $a(x)b(x) = rot(\mathbf{a})\mathbf{b} = rot(\mathbf{b})\mathbf{a}$.

The quasi-cyclic codes was proposed to noticeably reduce the public key size of Stern's protocol and MDPC code based cryptosystem [GG07, MTSB13].

Definition 2.5 (Quasi-cyclic Codes). *Given integers s, n and a vector $\mathbf{v} := (v_0, v_1, \ldots, v_{ns-1}) \in \mathbb{F}_2^{ns}$, it can be viewed as s blocks and each block is of length n. An $[sn, k]$ linear code $\mathcal{C}$ is* quasi-cyclic *of index s if for any codeword $\mathbf{c} = (c_0, \ldots, c_{ns-1}) \in \mathcal{C}$, applying a cyclic rotation to each length n block, the resulting vector is also contained in $\mathcal{C}$.*

Definition 2.6 (Systematic Quasi-cyclic Codes). *Given integers s, n and a set of $s-1$ vectors $\{\mathbf{h}_0, \ldots, \mathbf{h}_{s-2}\}$, a systematic quasi-cyclic code $[sn, n]$ of index s is a quasi-cyclic code parity checked by the following matrix*

$$H := \begin{pmatrix} I_n & 0 & \ldots & 0 & rot(\mathbf{h}_0) \\ 0 & I_n & \ldots & 0 & rot(\mathbf{h}_1) \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \ldots & I_n & rot(\mathbf{h_{s-2}}) \end{pmatrix}.$$

*In particular, the quasi-cyclic code with index s and rate $1/s$ is denoted s-*quasi-cyclic codes.

In our instantiations of the signature schemes, we only consider the case $s = 2$, i.e., 2-quasi-cyclic codes. Then the parity check matrix becomes $H := (I_n, rot(\mathbf{h}))$ for some vector $\mathbf{h}$. For convenience, we will write the parity check matrix as $H := (1, \mathbf{h})$. Then the matrix vector multiplication is performed over the ring $\mathcal{R}$.

Next we present an extremely important bound for linear codes.

Definition 2.7 (Gilbert-Varshamov(GV) bound)**.** *Given a binary linear code $\mathcal{C} := [n, k]$, the Gilbert-Varshamov bound is the largest integer w satisfying*

$$\sum_{i=0}^{w-1} \binom{n}{i} \leq 2^{n-k}.$$

2.2 Hard Problems in Coding Theory

In this section, we present some hard problems in coding theory. The worst cases of the problems are proven to be NP complete. The average cases of the problems are conjectured to be hard.

Definition 2.8 (Syndrome Decoding Problem(SDP))**.** *Let n, k, w be integers. Given a parity check matrix $H \in \mathbb{F}_2^{(n-k)\times n}$ and a syndrome $\mathbf{s} \in \mathbb{F}_2^{n-k}$, the syndrome decoding problem asks to find a vector $\mathbf{e} \in \mathbb{F}_2^n$ such that $H\mathbf{e} = \mathbf{s}$ and $\mathrm{wt}(\mathbf{e}) \leq w$.*

Remark 2.9. *If the weight w is below the GV bound of the code, then the solution to the syndrome decoding problem is unique.*

Definition 2.10 (Codeword Finding Problem(CFP))**.** *Let n, k, w be integers. Given a parity check matrix $H \in \mathbb{F}_2^{(n-k)\times n}$, the codeword finding problem asks to find a vector $\mathbf{c} \in \mathbb{F}_2^n$ such that $H\mathbf{c} = \mathbf{0}$ and $\mathrm{wt}(\mathbf{c}) \leq w$.*

The codeword finding problem can be viewed as an homogeneous version of the syndrome decoding problem. The syndrome decoding problem is proven to be NP-complete [BMvT78] and the codeword finding problem is proven to be NP-complete [Var97].

Remark 2.11.
- *The average case of the syndrome decoding problem can be regarded as the learning parity with noise(LPN) problem with limited number of LPN instances [AIK07].*
- *Assuming the hardness of the codeword finding problem for weight w, the corresponding syndrome function is collision resistant for weight $w/2$.*
- *For a uniform random H, if w is less than GV bound, the syndrome decoding problem has a unique solution with overwhelming probability. If w is greater than the GV bound, the syndrome function is many-to-one.*

We introduce a hard coding problem that is associated to our security proof. The Decoding One Out of Many(DOOM) problem was initially proposed by [JJ02]. Later, we will point out that the DOOM problem provides a speedup for the quasi-cyclic code problems compared to the matrix version of the coding problems.

Definition 2.12 (Decoding One Out of Many(DOOM))**.** *Let n, k, w, t be integers. Given a parity check matrix $H \in \mathbb{F}_2^{(n-k)\times n}$ and a set of t syndromes $\{\mathbf{s}_1, \mathbf{s}_2 \ldots \mathbf{s}_t\}$, the decoding one out of many problem asks to find a vector $\mathbf{e}_i \in \mathbb{F}_2^n$ such that $H\mathbf{e}_i = \mathbf{s}_i$ and $\mathrm{wt}(\mathbf{e}_i) \leq w$ for some i.*

Next we consider the 2-quasi-cyclic version of the above problems when the parity check matrix H is generated by the 2-quasi-cyclic parity check matrix $H = (I_n, rot(\mathbf{h}))$.

Definition 2.13 (2-Quasi-Cyclic Syndrome Decoding Problem(2-QCSDP))**.** *Let n, k, w be integers. Given a vector $\mathbf{h} \in \mathbb{F}_2^n$ and a syndrome vector $\mathbf{s} \in \mathbb{F}_2^n$, the 2-quasi-cyclic syndrome decoding problem asks to find a pair of vectors $(\mathbf{e}_0, \mathbf{e}_1)$ such that $\mathbf{h}\mathbf{e}_0 + \mathbf{e}_1 = \mathbf{s}$ and $\mathrm{wt}(\mathbf{e}_0) = w, \mathrm{wt}(\mathbf{e}_1) = w$.*

Definition 2.14 (2-Quasi-Cyclic Codeword Finding Problem(2-QCCFP))**.** *Let n, k, w be integers. Given a vector $\mathbf{h} \in \mathbb{F}_2^n$, the 2-quasi-cyclic codeword finding problem asks to find a nonzero codeword $\mathbf{c} = (\mathbf{c}_0, \mathbf{c}_1) \in \mathbb{F}_2^{2n}$ such that $\mathbf{h}\mathbf{c}_0 + \mathbf{c}_1 = \mathbf{0}$ and $\mathrm{wt}(\mathbf{c}_0) = w, \mathrm{wt}(\mathbf{c}_1) = w$.*

The decisional version of the 2-QCCFP is of interest. We will assume the decisional 2-QCCFP is hard, i.e., the produced vector $\mathbf{h}$ is pseudorandom. The decisional 2-QCCFP assumption will reduce the public key size by half. It is also used in the NIST submission BIKE [ABB+18].

Definition 2.15 (Decisional 2-QCCFP)**.** *Let n, k, w be integers. Given a vector $\mathbf{h} = \mathbf{c}_1/\mathbf{c}_0 \in \mathbb{F}_2^n$ with $\mathrm{wt}(\mathbf{c}_0) = w, \mathrm{wt}(\mathbf{c}_1) = w$, the* decisional 2-*QCCFP asks to distinguish* $\mathbf{h}$ *from the uniform random variable sampled from $\mathbb{F}_2^n$.*

Remark 2.16. *(i) The average quasi-cyclic syndrome decoding problem can be viewed as the Ring-LPN problem with fixed number of instances.*

(ii) To make the quasi-cyclic coding problem hard, we choose a prime n such that $(x^n-1)/(x-1) \in \mathbb{F}_2^n[x]$ is irreducible to avoid the squaring attack [LJS+ 16] and the reducible polynomial attack [GJL15].

Till now, there are no complexity results on the hardness of the above quasi-cyclic problems. We have no idea whether the problem is NP-complete or not. The average cases of the problems are conjectured to be hard. Generally speaking, the algorithms to solve the quasi-cyclic problems have no significant advantages compared to the algorithms to solve the matrix version of the coding problems. The best algorithm to solve it remains the the algorithm to solve the matrix version of the problems. As each instance of the 2-QCSDP can be transformed to a DOOM problem with $t = n$, Sendrier [Sen11] provided an analysis to adapt the algorithm to solve syndrome decoding problem to DOOM. The DOOM problem provides a polynomial factor speedup to the quasi-cyclic problems. The result indicates that it provides a $\sqrt{n}$ factor speedup to the 2-QCSDP and a n factor speedup to the 2-QCCFP.

For the decisional 2-QCCFP, the state of the art algorithms to solve it remains the algorithm to solve the search version of 2-QCCFP.

2.3 Information Set Decoding

In this section, we present the algorithms to solve the syndrome problem and the latest results on the complexity of the algorithms. Up to now, the best algorithm to solve the syndrome decoding problem is descendant of the information set decoding(ISD) algorithm proposed by Prange [Pra62]. The state of art results [MMT11, BJMM12, MO15] indicate a complexity of $2^{cw(1+o(1))}$, where w is the error weight and c is a constant related to the code rate, error rate and the algorithm. For sublinear $w = o(n)$, Torres and Sendrier [TS16] proved that the constant c is $c = -\log(1-r)$ all variants of the ISD algorithms, where r is the code rate. The result of Torres and Sendrier provides an extremely good estimation to the complexity of the ISD algorithms.

We remark that for a binary linear code $[n, k]$, the easy weight range of the syndrome decoding problem is $[\frac{n-k}{2}, \frac{n+k}{2}]$. In the instantiation of this paper, we use the small weight range $[0, \frac{n-k}{2})$ as the hard range for signature and secret key.

2.4 Signature Schemes

In this section, we recall the definitions of signature schemes and the security model of the signature schemes.

Definition 2.17 (Signature Schemes)**.** *[GMR88] A signature scheme consists of three PPT algorithms $(KeyGen, Sign, Verify)$,*

- *$KeyGen$ produces a public key, secret key pair upon a security level λ.*
- *$Sign$ produces a valid signature upon a message μ and the secret key.*
- *$Verify$ produces $accept$ or $reject$ upon the public key, a message μ and a signature σ of μ.*

$(KeyGen, Sign, Verify)$ is signature scheme if

$$Pr[Verify(\mu, pk, \sigma) = accept] \geq 1 - negl(\lambda).$$

The probability is over the randomness of $KeyGen$, $Sign$ and $Verify$.

Definition 2.18 (Existential Unforgeability under an adaptive Chosen Message Attack(EUF-CMA))**.** *A signature scheme achieves the existential unforgeability under an adaptive chosen message attack if for any PPT forger $\mathcal{F}$, the probability, that $\mathcal{F}$ is able to forge a valid signature for a fresh message μ whose signature is never queried after requesting polynomial number of signatures of its choice, is negligible. The probability is over the randomness of $KeyGen$, $Sign$, $Verify$ and $\mathcal{F}$ itself.*

In the EUF-CMA security model, the forger is limited to be unable to forge a valid signature for a fresh message. A stronger security model *strong EUF-CMA(SUF-CMA)* requires that the forger is unable to produce a different valid signature for any queried message. In the main body part, we prove that our signature scheme achieves EUF-CMA security. When choosing relaxed parameters, our signature scheme is proven to be strong EUF-CMA secure in Appendix.

Note that during the security proof in the random oracle model, a sign query implies a hash query. When we say a forger makes h hash queries and s sign queries, the implied hash queries from sign queries are excluded. Thus the total number of hash queries is $h + s$.

2.5 Rejection Sampling

To argue that the signature does not leak any information on the private key, we use a general *rejection sampling* lemma [Lyu12]. Here the lemma is defined over $\mathbb{F}_2^n$.

Lemma 2.19 (Rejection Sampling)**.** *Given a subset $V \subseteq \mathbb{F}_2^n$, let $f : \mathbb{F}_2^n \to \mathbb{R}$ and $h : V \to \mathbb{R}$ be two probability distributions. Let $g_{\mathbf{v}} : \mathbb{F}_2^n \to \mathbb{R}$ be a probability distribution indexed by $\mathbf{v} \in V$ such that there exists a number $M \in \mathbb{R}$ such that*

$$Pr\left[\frac{f(\mathbf{z})}{g_{\mathbf{v}}(\mathbf{z})} \leq M : \mathbf{z} \leftarrow f\right] \geq 1 - \epsilon.$$

Then the statistical distance of the following two distributions is ϵ/M and the first distribution outputs something with probability $\frac{1-\epsilon}{M}$.

1. *$\mathbf{v} \leftarrow h$, $\mathbf{z} \leftarrow g_{\mathbf{v}}$, output $(\mathbf{v}, \mathbf{z})$ with probability $\frac{f(\mathbf{z})}{Mg_{\mathbf{v}}(\mathbf{z})}$.*
2. *$\mathbf{v} \leftarrow h$, $\mathbf{z} \leftarrow f$, output $(\mathbf{v}, \mathbf{z})$ with probability $1/M$.*

In particular, if $\epsilon = 0$, the two output distributions are identical and the first distribution outputs something with probability $1/M$.

The proof of the rejection sampling lemma follows a standard method. We omit it here.

Remark 2.20. *Notice that the rejection sampling lemma efficiently tailors any real distribution g to a desired distribution f if $M := max_{\mathbf{z}} \frac{f(\mathbf{z})}{g_{\mathbf{v}}(\mathbf{z})}$ is in an acceptable range.*

2.6 Bounds and Approximations

The piling-up lemma is used to compute the sum distribution of Bernoulli distributions.

Lemma 2.21 (Piling-up Lemma)**.** *Given n identical independent Bernoulli random variables $X_1, X_2 \ldots X_n$ with Bernoulli parameter p, the sum of the random variables yields a fresh Bernoulli random variable with parameter $\tau = \frac{1}{2} - \frac{1}{2}(1-2p)^n$.*

The proof of the piling-up lemma follows a mathematical induction.

Lemma 2.22 (Hoeffding Bound)**.** *[MU05, Theorem 4.12] Given n independent random variables $X_1, X_2 \ldots X_n$ satisfying $E[X_i] = \mu$ and $X_i \in [a, b]$, then*

$$Pr\left[\left|\sum_{i\in[n]} X_i - n\mu\right| \geq \Delta\right] \leq 2\mathrm{e}^{-\frac{\Delta^2}{n(b-a)^2}}.$$

From the Hoeffding bound, given a binomial distributed binary vector $\mathbf{v} \to \mathcal{B}_p^n$, where $\mathcal{B}_p^n$ is the binomial distribution defined over $\mathbb{F}_2^n$ with parameter p, the weight of $\mathbf{v}$ is around np.

We present a result on binomial coefficients without proof. The proof of this result can be found in [MU05].

Lemma 2.23. *For integers k and n with $k < n$, $\frac{2^{nH(k/n)}}{\sqrt{n}} \leq \binom{n}{k} \leq 2^{nH(k/n)}$ where $H(x) = -x\log x - (1-x)\log(1-x)$ is the binary entropy function.*

For linear $k = O(n)$, the lower bound $\frac{2^{nH(k/n)}}{\sqrt{n}}$ provides a good estimation for $\binom{n}{k}$. For sublinear $k = o(n)$, the geometric mean value of the lower bound and the upper bound is a good estimation for $\binom{n}{k}$ i.e., $\log\binom{n}{k} \approx nH(\frac{k}{n}) - \frac{\log n}{4}$.

We provide a reasonable lower bound and upper bound for the binary entropy function $H(\cdot)$.

Lemma 2.24. *For very small x, $-x\log x \leq H(x) \leq -x\log x + (1-x)x\log \mathrm{e}$.*

Proof. The lower bound is taking from the maximal part of the sum of the entropy function. The upper bound is from the inequality $1 - x \leq \mathrm{e}^{-x}$ for very small x. □

For very small x, the upper bound provides a good estimation for $H(x)$.

With the estimation for $H(x)$, we straightforwardly obtain the following estimation for $\binom{n}{k}$. The estimation will be used when we prove the rejection sampling result.

Lemma 2.25. *For $k = o(n)$, $\log\binom{n}{k} \approx k\log\frac{n}{k} + k(1-\frac{k}{n})\log\mathrm{e} - \frac{\log n}{4}$.*

The proof is straightforward.

Proof. $\log\binom{n}{k} \approx nH(\frac{k}{n}) - \frac{\log n}{4} \approx k\log\frac{n}{k} + k(1-\frac{k}{n})\log\mathrm{e} - \frac{\log n}{4}$. □

3 Rejection Sampling and Binomial Distribution

In this section, we provide some results on rejection sampling to obtain an aligned binomial distribution from shifted binomial distribution.

Definition 3.1 (Binomial Distribution)**.** *The binomial distribution over $\mathbb{F}_2^n$ with parameter p is defined as $\mathcal{B}_p^n(\mathbf{x}) := \binom{n}{\mathrm{wt}(\mathbf{x})} p^{\mathrm{wt}(\mathbf{x})}(1-p)^{n-\mathrm{wt}(\mathbf{x})}$.*

Throughout this paper, the parameter p is a constant. Given a vector $\mathbf{v} \in \mathbb{F}_2^n$, we define the shifted binomial distribution with the vector $\mathbf{v}$.

Definition 3.2 (Shifted Binomial Distribution)**.** *Given $\mathbf{v} \in \mathbb{F}_2^n$, the shifted binomial distribution with parameter p is defined as $\mathcal{B}_{\mathbf{v},p}^n(\mathbf{x}) := \mathcal{B}_p^n(\mathbf{x} - \mathbf{v})$, i.e., $\mathbf{x} - \mathbf{v}$ admits the binomial distribution $\mathcal{B}_p^n$.*

In the definition, the density function of shifted binomial distribution relies on the Hamming weight of $\mathbf{x} - \mathbf{v}$, which is not easy to determine. We provide the density function of shifted binomial distribution only depending on the Hamming weight of $\mathbf{x}$ and the Hamming weight of $\mathbf{v}$.

Lemma 3.3. *Given $\mathbf{v} \in \mathbb{F}_2^n$ and the parameter p, assume $\mathrm{wt}(\mathbf{v}) = s$ and $\mathrm{wt}(\mathbf{x}) = t$. For $s \leq t$, we have*

$$\mathcal{B}_{\mathbf{v},p}^n(\mathbf{x}) = \sum_{j=0}^{s} \binom{s}{j}\binom{n-s}{t-j} p^{s+t-2j}(1-p)^{n-s-t+2j}.$$

The proof follows a directly counting argument.

Proof. Let $\mathbf{y} = \mathbf{x} - \mathbf{v}$ admit a binomial distribution $\mathcal{B}_p^n$. Suppose the intersection of the supports of $\mathbf{x}$ and $\mathbf{v}$ is of size j. Then $\mathbf{y}$ is of weight $s + t - 2j$. Thus the support of $\mathbf{v}$ contributes exactly $s - j$ ones to $\mathbf{y}$, which happens with probability $\binom{s}{s-j} p^{s-j}(1-p)^j$. Thus, the support of $\mathbf{x}$ contributes exactly $t - j$ ones to $\mathbf{y}$, which happens with probability $\binom{n-s}{t-j} p^{t-j}(1-p)^{n-s-t+j}$. Thus, for a fixed value j, the probability to get $\mathbf{x}$ of weight t is

$$\binom{s}{s-j} p^{s-j}(1-p)^j \binom{n-s}{t-j} p^{t-j}(1-p)^{n-s-t+j} = \binom{s}{j}\binom{n-s}{t-j} p^{s+t-2j}(1-p)^{n-s-t+2j}.$$

Therefore, $\mathcal{B}_{\mathbf{v},p}^n(\mathbf{x}) = \sum_{j=0}^{s} \binom{s}{j}\binom{n-s}{t-j} p^{s+t-2j}(1-p)^{n-s-t+2j}$. □

Next we define the truncated binomial distribution which is polynomially sampled.

Definition 3.4 (Truncated Binomial Distribution)**.** *The truncated binomial distribution over $\mathbb{F}_2^n$ with parameters p, a, b for $a \le \mathrm{wt}(\mathbf{x}) \le b$ is defined as*

$$\widetilde{\mathcal{B}}_{a,b,p}^n(\mathbf{x}) := \frac{\mathcal{B}_p^n(\mathbf{x})}{\sum_{a \le wt(\mathbf{x}) \le b} \mathcal{B}_p^n(\mathbf{x})}.$$

From the definition of truncated binomial distribution, the density function is proportional to the binomial distribution density function and the domain of truncated binomial distribution is limited to a given range $[a, b]$.

Remark 3.5. *For the case that a and b are symmetrically centered around np, we denote the density function of truncated binomial distribution as $\widetilde{\mathcal{B}}_{\kappa,p}^n(\mathbf{x})$, where $\kappa = b - np = np - a$.*

We notice that in the binomial distribution, shifted binomial distribution, and truncated binomial distribution, the probability density function are only sensitive to the Hamming weight of involved vectors as opponent to the vector itself. In what follows, we use $\mathcal{B}_p^n(v)$ to denote $\mathcal{B}_p^n(\mathbf{v})$ with $\mathrm{wt}(\mathbf{v}) = v$.

In the following, we employ a series of lemmas to establish a rejection sampling lemma to tune a shifted binomial distribution to a truncated binomial distribution for well-chosen parameters.

We first compute the ratio of the density function of binomial distribution over shifted binomial distribution.

Lemma 3.6. *For $\mathbf{v} \in \mathbb{F}_2^n$, assume $\mathrm{wt}(\mathbf{v}) = s$ and $s < t$,*

$$\frac{\mathcal{B}_p^n(t)}{\mathcal{B}_{\mathbf{v},p}^n(t)} = \frac{\binom{n}{s}}{\sum_{j=0}^{s} \binom{t}{j}\binom{n-t}{s-j} p^{s-2j}(1-p)^{2j-s}}.$$

Proof. We have

$$\begin{aligned}\frac{\mathcal{B}_p^n(t)}{\mathcal{B}_{\mathbf{v},p}^n(t)} &= \frac{\binom{n}{t} p^t (1-p)^{n-t}}{\sum_{j=0}^{s} \binom{s}{j}\binom{n-s}{t-j} p^{s+t-2j}(1-p)^{n-s-t+2j}} = \frac{1}{\sum_{j=0}^{s} \frac{\binom{s}{j}\binom{n-s}{t-j}}{\binom{n}{t}} p^{s-2j}(1-p)^{2j-s}} \\ &= \frac{1}{\sum_{j=0}^{s} \frac{\binom{t}{j}\binom{n-t}{s-j}}{\binom{n}{s}} p^{s-2j}(1-p)^{2j-s}} = \frac{\binom{n}{s}}{\sum_{j=0}^{s} \binom{t}{j}\binom{n-t}{s-j} p^{s-2j}(1-p)^{2j-s}},\end{aligned}$$

where the first equality follows the density function of binomial distribution and shifted binomial distribution, the third equality follows

$$\frac{\binom{s}{j}\binom{n-s}{t-j}}{\binom{n}{t}} = \frac{\frac{s!}{j!(s-j)!}\frac{(n-s)!}{(t-j)!(n-s-t+j)!}}{\frac{n!}{t!(n-t)!}} = \frac{s!(n-s)!}{n!}\frac{t!}{j!(t-j)!}\frac{(n-t)!}{(s-j)!(n-s-t+j)!} = \frac{\binom{t}{j}\binom{n-t}{s-j}}{\binom{n}{s}},$$

or the symmetry property of hypergeometric distribution if $\frac{\binom{s}{j}\binom{n-s}{t-j}}{\binom{n}{t}}$ is viewed as the density function of hypergeometric distribution. □

Remark 3.7. *Writing the density function in this form is also beneficial for us to compute the ratio of probabilities for concrete parameters in Section 6 as s is extremely small compared to n and t.*

Define $f(n,t,s,p,j) := \binom{t}{j}\binom{n-t}{s-j}p^{s-2j}(1-p)^{2j-s}$. Assume $f(n,t,s,p,j)$ achieves its maximum value at some $\ell \in \{0\dots s\}$, i.e., $\max_{j=0}^{s} f(n,t,s,p,j) = \binom{t}{\ell}\binom{n-t}{s-\ell}p^{s-2\ell}(1-p)^{2\ell-s}$. Next we try to find ℓ and present a good estimation for $f(n,t,s,p,\ell)$. With the estimation for $f(n,t,s,p,\ell)$, we will prove a good estimation for $\sum_{j=0}^{s} f(n,t,s,p,j)$.

Lemma 3.8. *For $t = pn + \Delta$ with $\Delta = o(pn)$, $s = o(n)$, $f(n,t,s,p,j)$ achieves its maximum value around $s(1-p)$.*

In particular,

$$f(n,t,s,p,\lceil s(1-p)\rfloor) \approx s\log\frac{t}{ps} + (s - \frac{s^2(1-p)^2}{t} - \frac{s^2p^2}{nt} - \frac{(t-pn)s}{(1-p)n})\log \mathrm{e} - \frac{\log(t(n-t))}{4}.$$

In the proof of the lemma, we first prove that the point ℓ is close to $s(1-p)$ and then use the approximation for binomial coefficient to obtain a close approximation for the maximum value.

Proof. A rough inspection indicates that the function f reaches its maximum value at some midst value of $\{0\dots s\}$. Next we try to find the value ℓ. Consider

$$\frac{f(n,t,s,p,j+1)}{f(n,t,s,p,j)} = \frac{\binom{t}{j+1}\binom{n-t}{s-j-1}p^{s-2j-2}(1-p)^{2j-s+2}}{\binom{t}{j}\binom{n-t}{s-j}p^{s-2j}(1-p)^{2j-s}} = (\frac{1-p}{p})^2\frac{(t-j)(s-j)}{(n-t-s+j+1)(j+1)}.$$

Then the above expression can be simplified as

$$\begin{aligned}\frac{f(n,t,s,p,j+1)}{f(n,t,s,p,j)} &= (\frac{1-p}{p})^2\frac{(pn+\Delta-j)(s-j)}{(n-pn-\Delta-s+j+1)(j+1)}\\ &= \frac{1+\frac{\Delta-j}{pn}}{1-\frac{\Delta+s-j-1}{(1-p)n}}\frac{(1-p)(s-j)}{p(j+1)}\\ &\approx \frac{\mathrm{e}^{\frac{\Delta-j}{pn}}}{\mathrm{e}^{-\frac{\Delta+s-j-1}{(1-p)n}}}\frac{(1-p)(s-j)}{p(j+1)} = \mathrm{e}^{\frac{\Delta+ps-j-p}{p(1-p)n}}\frac{(1-p)(s-j)}{p(j+1)}\\ &\approx \frac{(1-p)(s-j)}{p(j+1)}\end{aligned} \tag{1}$$

where the first approximation follows as $\Delta - j$ is sublinear of pn, $\Delta + s - j - 1$ sublinear of $(1-p)n$ and $1 \pm x \approx \mathrm{e}^{\pm x}$ for very small x, and the second approximation follows the same reason. To find the j that reaches the largest value of $f(n,t,s,p,j)$, we let $\frac{f(n,t,s,p,j+1)}{f(n,t,s,p,j)} \approx \frac{(1-p)(s-j)}{p(j+1)} \geq 1$. Then we obtain $j < s(1-p) - p$. It means for $0 \leq j \leq s(1-p) - p$, $f(n,t,s,p,j)$ is a monotonically increasing function and for $s(1-p) \leq j \leq s$, $f(n,t,s,p,j)$ a monotonically decreasing function. Then $f(n,t,s,p,j)$ achieves its maximum value around the point $\ell = \lceil s(1-p)\rfloor$. Next we use the value for ℓ to estimate $f(n,t,s,p,\ell)$.

Then

$$\begin{aligned}
\log f(n,t,s,p,\ell) \approx & \log \binom{t}{s-ps}\binom{n-t}{ps} p^{2ps-s}(1-p)^{s-2ps} \\
\approx & (s-ps)\log\frac{t}{s-ps} + (s-ps)(1-\frac{s-ps}{t})\log \mathrm{e} - \frac{\log t}{4} \\
& + ps\log\frac{n-t}{ps} + ps(1-\frac{ps}{n-t})\log \mathrm{e} - \frac{\log (n-t)}{4} \\
& + (2ps-s)\log p + (s-2ps)\log(1-p) \qquad (2)\\
= & \log(\frac{t}{ps})^s(\frac{p}{1-p})^{ps}(\frac{n-t}{t})^{ps} + (s - \frac{s^2(1-p)^2}{t} - \frac{s^2p^2}{n-t})\log \mathrm{e} - \frac{\log (t(n-t))}{4} \\
= & \log(\frac{t}{ps})^s(\frac{1-\frac{\Delta}{n-pn}}{1+\frac{\Delta}{pn}})^{ps} + (s - \frac{s^2(1-p)^2}{t} - \frac{s^2p^2}{n-t})\log \mathrm{e} - \frac{\log (t(n-t))}{4} \\
\approx & \log(\frac{t}{ps})^s \mathrm{e}^{(-\frac{\Delta}{(1-p)n}-\frac{\Delta}{pn})ps} + (s - \frac{s^2(1-p)^2}{t} - \frac{s^2p^2}{n-t})\log \mathrm{e} - \frac{\log (t(n-t))}{4} \\
& \qquad (3)\\
= & \log(\frac{t}{ps})^s \mathrm{e}^{-\frac{\Delta s}{(1-p)n}} + (s - \frac{s^2(1-p)^2}{t} - \frac{s^2p^2}{n-t})\log \mathrm{e} - \frac{\log (t(n-t))}{4} \\
= & s\log\frac{t}{ps}(s - \frac{s^2(1-p)^2}{t} - \frac{s^2p^2}{nt} - \frac{(t-pn)s}{(1-p)n})\log \mathrm{e} - \frac{\log (t(n-t))}{4},
\end{aligned}$$

where Equation 2 follows Lemma 2.25 for the approximation of binomial coefficients and Equation 3 follows that Δ is sublinear of $np(1-p)$ and $1 \pm x \approx e^{\pm x}$ for very small x. □

Remark 3.9. *In the proof of the above lemma, we use $\Delta + s$ is sublinear of $p(1-p)n$.*

In the process of searching for the value ℓ, we find out that the value of the function $f(n,t,s,p,j)$ variates smoothly when j is extremely close to ℓ. Combining with the above estimation for $f(n,t,s,p,\lceil s(1-p)\rfloor)$, we explicitly approximate the value $\sum_{j=0}^{s} f(n,t,s,p,j)$ in the following lemma.

Lemma 3.10. *Let $t = pn + \Delta$ with $\Delta = o(pn)$ and $s = o(n)$. For $\delta = o(p(1-p)s)$,*

$$\sum_{j=0}^{s} f(n,t,s,p,j) \approx 2\delta f(n,t,s,p,\lceil s(1-p)\rfloor).$$

In particular, taking $\delta = \sqrt{p(1-p)s}$, then

$$\sum_{j=0}^{s} f(n,t,s,p,j) \approx 2\sqrt{p(1-p)s} f(n,t,s,p,\lceil s(1-p)\rfloor).$$

We find the boundary point that the value of the $f(n,t,s,p,j)$ sharply variates around $s(1-p)$ and use the smoothly variating function values to approximate $\sum_{j=0}^{s} f(n,t,s,p,j)$.

Proof. Assume $j = s - ps - \delta$. From the proof of Lemma 3.8, we have

$$\frac{f(j+1)}{f(j)} \approx \frac{s-j}{j+1}\frac{1-p}{p} = \frac{1+\frac{\delta}{ps}}{1-\frac{\delta}{s-ps}} \approx \mathrm{e}^{\frac{\delta}{ps}+\frac{\delta-1}{(1-p)s}} = \mathrm{e}^{\frac{\delta-p}{p(1-p)s}} \approx 1,$$

where the approximation is from the assumption $\delta = o(p(1-p)s)$.

In particular, taking $\delta = \sqrt{p(1-p)s}$, we can use 2δ number of $f(n,t,s,p,\lceil (1-p)s\rfloor)$ to estimate $\sum_{j=0}^{s} f(j)$. Thus, $\sum_{j=0}^{s} f(j) \approx 2\sqrt{p(1-p)s} f(n,t,s,p,\lceil (1-p)s\rfloor)$. □

With the estimation for $\sum_{j=0}^{s} f(n,t,s,p,j)$ in hand, we present the ratio of density function of binomial distribution over shifted binomial distribution.

Lemma 3.11. *For $t = pn + \Delta$ with $\Delta = o(pn)$, $s = o(n)$, assume $\mathrm{wt}(\mathbf{v}) = s$. Then we have*

$$\log \frac{\mathcal{B}_p^n(t)}{\mathcal{B}_{\mathbf{v},p}^n(t)} \approx \log \binom{n}{s} - \log f((1-p)s) - \log(2\sqrt{p(1-p)s})$$
$$\approx \log(\mathrm{e}^{\frac{s^2}{n}\frac{(1-2p)^2}{p(1-p)} - \frac{\Delta s}{n}\frac{1}{p(1-p)}}) + \log(\frac{n}{s^2 p(1-p)})^{1/4}) - 1.$$

The proof follows the results of Lemma 3.8 and Lemma 3.10.

Proof.

$$\log \frac{\mathcal{B}_p^n(t)}{\mathcal{B}_{\mathbf{v},p}^n(t)} \approx \log \binom{n}{s} - \log f((1-p)s) - \log(2\sqrt{p(1-p)s}) \tag{4}$$
$$\approx \log(\frac{n}{s})^s + s(1-\frac{s}{n})\log \mathrm{e} - \frac{\log n}{4} - \log(2\sqrt{p(1-p)s})$$
$$- s\log\frac{t}{ps} - (s - \frac{s^2(1-p)^2}{t} - \frac{s^2p^2}{n-t} - \frac{(t-pn)s}{(1-p)n})\log \mathrm{e} + \frac{\log(t(n-t))}{4} \tag{5}$$
$$= \log(\frac{np}{t})^s + \frac{(t-pn)s}{(1-p)n}\log \mathrm{e} + (\frac{s^2(1-p)^2}{t} - \frac{s^2}{n} + \frac{s^2p^2}{n-t})\log \mathrm{e} + \frac{\log(t(n-t)/n)}{4} - \log(2\sqrt{p(1-p)s})$$
$$\approx -\frac{\Delta s}{p(1-p)n}\log \mathrm{e} + \frac{s^2}{n}(\frac{(1-2p)^2 + \frac{(2p-1)(2p^2-2p+1)}{p(1-p)}\frac{\Delta}{n}}{p(1-p)})\log \mathrm{e} + \log(\frac{n}{p(1-p)s^2})^{1/4}\mathrm{e}^{\frac{\Delta}{4np(1-p)}} - 1$$
$$\approx -\frac{\Delta s}{p(1-p)n}\log \mathrm{e} + \frac{s^2}{n}\frac{(1-2p)^2}{p(1-p)}\log \mathrm{e} + \log(\frac{n}{p(1-p)s^2})^{1/4} - 1$$
$$= \log(\mathrm{e}^{\frac{s^2}{n}\frac{(1-2p)^2}{p(1-p)} - \frac{\Delta s}{n}\frac{1}{p(1-p)}}) + \log(\frac{n}{s^2p(1-p)})^{1/4}) - 1,$$

where Equation 4 follows Lemma 3.10 and Equation 5 follows Lemma 3.8. □

Remark 3.12.
- *The result of the above lemma indicates that if $s^2 > n$ or $s\Delta > n$, the exponential part becomes the main term of the ratio. If $s^2 < n$ and $s\Delta < n$, the polynomial term $O((\frac{n}{s^2})^{1/4})$ becomes the main term of the ratio. To make the ratio being polynomial on the input parameters, we will choose s and Δ such that $s^2 < n$ and $s\Delta < n$.*
- *Intuitively, the lemma shows that altering a vector of large expectation weight by a small weight vector has no significant impact on the distribution of the large weight vector.*
- *From the view of the figure of probability density function, the above lemma demonstrates that for $s^2 < n$ and $s\Delta < n$, the binomial distribution and the shifted binomial distribution have adequate overlap.*

In our signature scheme, we employ the truncated binomial distribution. Thus we would like to give an estimation for $\frac{\widetilde{\mathcal{B}}_{\kappa,p}^n(t)}{\mathcal{B}_{\mathbf{v},p}^n(t)} = \frac{\mathcal{B}_p^n(t)}{\mathcal{B}_{\mathbf{v},p}^n(t)\sum_{j=np-\kappa}^{np+\kappa}\binom{n}{j}p^j(1-p)^{n-j}}$. Before giving an approximation to the ratio of the truncated binomial distribution over shifted binomial distribution, we provide an estimation to the denominator of truncated binomial distribution i.e., $\sum_{j=np-\kappa}^{np+\kappa}\binom{n}{j}p^j(1-p)^{n-j}$ for $\widetilde{\mathcal{B}}_{\kappa,p}^n$.

Lemma 3.13. *For $\kappa = o(np(1-p))$, $\sum_{j=pn-\kappa}^{pn+\kappa}\binom{n}{j}p^j(1-p)^{n-j} \approx \frac{2\kappa}{\sqrt{n}}$.*

Define $g_{n,p}(j) := \binom{n}{j}p^j(1-p)^{n-j}$. Similar to the proof of Lemma 3.8, we find the maximal point of $g_{n,p}(j)$ and the boundary point that the value of the function $g_{n,p}(j)$ sharply variates to provide an estimation to $\sum_j g_{n,p}(j)$.

Proof. Consider

$$\frac{g_{n,p}(j+1)}{g_{n,p}(j)} = \frac{\binom{n}{j+1}p^{j+1}(1-p)^{n-j-1}}{\binom{n}{j}p^j(1-p)} = \frac{n-j}{j+1}\frac{p}{1-p}.$$

Let $\frac{g_{n,p}(j+1)}{g_{n,p}(j)} > 1$. Then we obtain $j < np - 1 + p$. It means for $np - \kappa < j < np - 1 + p$, $g_{n,p}(j)$ is monotonically increasing whereas for $np-1+p < j < np+\kappa$, $g_{n,p}(j)$ is monotonically decreasing. Thus, $g_{n,p}(j)$ achieves its maximum value around the point np. We approximate the value for $g_{n,p}(np)$. Then

$$g_{n,p}(np) = \binom{n}{np}p^{np}(1-p)^{n-np} \approx \frac{2^{nH(p)}}{\sqrt{n}}2^{-nH(p)} = \frac{1}{\sqrt{n}}.$$

Next we find the boundary point that $g_{n,p}(j)$ sharply varies. Assume $j = np - \zeta$ for $\zeta = o(np(1-p))$. Then

$$\frac{g_{n,p}(j+1)}{g_{n,p}(j)} = \frac{n-pn+\zeta}{np-\zeta+1}\frac{p}{1-p} = \frac{1+\frac{\zeta}{n(1-p)}}{1-\frac{\zeta-1}{np}} \approx \mathrm{e}^{\frac{\zeta}{n(1-p)}+\frac{\zeta-1}{np}} \approx 1,$$

where the approximation follows $\zeta = o(np(1-p))$. The result indicates that the value of the function $g_{n,p}(j)$ smoothly varies in the range $[np-\kappa, np+\kappa]$. Thus, we use the value $2\kappa g_{n,p}(np)$ to estimate $\sum_{j=pn-\kappa}^{pn+\kappa}\binom{n}{j}p^j(1-p)^{n-j}$. Then, we obtain

$$\sum_{j=pn-\kappa}^{pn+\kappa}\binom{n}{j}p^j(1-p)^{n-j} \approx 2\kappa g_{n,p}(np) \approx \frac{2\kappa}{\sqrt{n}}.$$

□

Combining Lemma 3.11 and Lemma 3.13 yields the main lemma for the ratio of probability distributions of this section.

Lemma 3.14. *For $t \in [np-\kappa, np+\kappa], s = o(np(1-p))$, where $\kappa = o(np(1-p))$, assume $\mathrm{wt}(\mathbf{v}) = s$. Then we have*

$$\frac{\widetilde{\mathcal{B}}^n_{\kappa,p}(t)}{\mathcal{B}^n_{\mathbf{v},p}(t)} \approx \mathrm{e}^{\frac{s^2}{n}\frac{(1-2p)^2}{p(1-p)}-\frac{(t-pn)s}{n}}\frac{\kappa}{(ns^2p(1-p))^{1/4}}.$$

The proof of the lemma is straightforward.

Proof. From the definition of truncated binomial distribution and Lemma 3.11, we have

$$\frac{\widetilde{\mathcal{B}}^n_{\kappa,p}(t)}{\mathcal{B}^n_{\mathbf{v},p}(t)} = \frac{\mathcal{B}^n_p(t)}{\mathcal{B}^n_p(t)\sum_{j=pn-\kappa}^{pn+\kappa}\binom{n}{j}p^j(1-p)^{n-j}} \approx \mathrm{e}^{\frac{s^2}{n}\frac{(1-2p)^2}{p(1-p)}-\frac{(t-pn)s}{n}}\frac{\kappa}{(ns^2p(1-p))^{1/4}}.$$

□

In our signature scheme, the ratio of the density function of truncated binomial distribution over shifted binomial distribution is referred as *rejection rate.* Roughly speaking, the conclusion is that larger n and larger p lead to a smaller rejection rate.

Remark 3.15. *From the result of the above lemma, to make the rejection rate in an acceptable range, the value of n cannot be too small. Theoretically speaking, choosing $s^2 < n$ and $\kappa s < n$ leads to a reasonably small rejection rate. However, in our signature scheme, the public key size and the signature size are decided directly by the magnitude of n, hence we prefer to choose* *a smaller n. In practice, to achieve some claimed security level, the size of s is almost lower bounded by a fixed value and setting $n > s^2$ makes n too large. Thus, we have only the freedom to alter the magnitude of n. In our instantiation of signature scheme, we choose to decrease the magnitude of n to reduce the public key size and the signature size at the cost of amplifying the rejection rate. As a matter of fact, we choose $n = s^\beta$ for some $\beta \in [1.6, 2]$ to obtain a smaller public key size and an acceptable rejection rate. Furthermore, the value of κ is limited to a small value.*

In order to tune a real shifted binomial distribution to the desired truncated binomial distribution, the rejection sampling algorithm employs a rejection vector $\mathbf{r}$ to finish the task.

Lemma 3.16 (Rejection Sampling Lemma for Binary Linear Codes)**.** *Let V be a subset of $\mathbb{F}_2^n$ and $h : V \to \mathbb{R}$ be a probability distribution. Let p be a constant as the binomial distribution parameter and κ be range parameter for the truncated binomial distribution. Define the rejection vector $\mathbf{r}$ as*

$$r(t) := \frac{1}{M}\frac{\widetilde{\mathcal{B}}^n_{\kappa,p}(t)}{\mathcal{B}^n_{\mathbf{v},p}(t)} \text{ for } t \in [np-\kappa, np+\kappa] \text{ where } M = \max_{t\in[np-\kappa,np+\kappa]} \frac{\widetilde{\mathcal{B}}^n_{\kappa,p}(t)}{\mathcal{B}^n_{\mathbf{v},p}(t)}.$$

Then the following two output distributions are identical.

1. *Sample $\mathbf{v}$ from the distribution h, sample $\mathbf{z}$ from the distribution $\mathcal{B}^n_{\mathbf{v},p}$, and output $(\mathbf{z}, \mathbf{v})$ with probability $r(\mathrm{wt}(\mathbf{z}))$.*
2. *Sample $\mathbf{v}$ from the distribution h, sample $\mathbf{z}$ from the distribution $\widetilde{\mathcal{B}}^n_{\kappa,p}$, and output $(\mathbf{z}, \mathbf{v})$ with probability $\frac{1}{M}$.*

The correctness of the rejection sampling lemma follows the general rejection sampling result of Lemma 2.19. The efficiency of the rejection sampling lemma for binary linear codes follows Lemma 3.16. In particular, for $\mathrm{wt}(\mathbf{v})^2 < n$ and $\mathrm{wt}(\mathbf{v})\kappa < n$, the value M is about $O(\frac{\kappa}{(ns^2p(1-p))^{1/4}})$.

In our signature scheme, we use the above rejection sampling lemma for binary linear codes to decouple the dependence of the signature on the secret key part to prove the signature leaks nothing on the secret key.

We hope the variants of binomial distribution introduced in this section have other application to coding theory for instance the complexity of coding problems as the Gaussian distribution for lattices [AR05].

4 Signature Scheme

In this section, we show our signature scheme from hard coding problems. The security of the signature scheme is from the hardness of the syndrome decoding problem. Let λ be the security parameter.

The signature scheme is presented in Algorithm 1, Algorithm 2 and Algorithm 3.

Algorithm 1 KeyGen

```
Require: Parameters $k, n, \ell, \sigma$.
Ensure: $(H, T) \in \mathbb{F}_2^{(n-k)\times n} \times \mathbb{F}_2^{n\times\ell}, S \in \mathbb{F}_2^{n\times\ell}$
1: Uniformly sample $H$ from $\mathbb{F}_2^{(n-k)\times n}$.
2: Sample the matrix $S \in \mathbb{F}_2^{n\times\ell}$ such that every entry of $S$ is sampled from
   the distribution $\mathcal{B}_\sigma$.
3: Let $T = HS \in \mathbb{F}_2^{(n-k)\times\ell}$.
4: Output $(H, T)$ as the public key pair and $S$ as the secret key.
```

Algorithm 2 Signing

```
Require: The public key pair $(H, T)$, the secret key $S$, the message $\mu$, the
    parameter $\tau < 1/2$, the hash function $\mathcal{H}$ and the range parameter $\xi$.
Ensure: A signature $(\mathbf{z}, \mathbf{c})$ of the message $\mu$.
 1: Sample $\mathbf{e}$ from $\mathcal{B}_\tau^n$. Let $\mathbf{y} = H\mathbf{e}$.
 2: $\mathbf{c} = \mathcal{H}(\mathbf{y}, \mu)$.
 3: $\mathbf{z} = S\mathbf{c} + \mathbf{e}$.
 4: If $\mathrm{wt}(\mathbf{z}) \notin [n\tau - \xi, n\tau + \xi]$, then restart.
 5: Output $(\mathbf{z}, \mathbf{c})$ with probability $\frac{\widetilde{\mathcal{B}}_{\xi,\tau}(\mathbf{z})}{M\mathcal{B}_{S\mathbf{c},\tau}(\mathbf{z})}$, where $M = \max_{j \in [n\tau-\xi, n\tau+\xi]} \frac{\widetilde{\mathcal{B}}_{\xi,\tau}(j)}{M\mathcal{B}_{S\mathbf{c},\tau}(j)}$.
```

Algorithm 3 Verifying

```
Require: Public key pair $(H, T)$, the message $\mu$, the signature $(\mathbf{z}, \mathbf{c})$ and the
    range parameter $\xi$.
Ensure: Accept or Reject
 1: If $\mathrm{wt}(\mathbf{z}) \notin [n\tau - \xi, n\tau + \xi]$, then output Reject.
 2: Output Accept if $\mathcal{H}(H\mathbf{z} - T\mathbf{c}, \mu) = \mathbf{c}$, otherwise Reject.
```

The KeyGen algorithm mainly relies on the distribution $\mathcal{B}_\sigma$. We will choose parameters (n, k, σ) such that given (H, T), finding S is 2^λ-hard. More conditions on the parameter σ and n will be given later. Each entry of the secret key S is independently sampled from $\mathcal{B}_\sigma$. Then each column of S is viewed as independently drawn from $\mathcal{B}_\sigma^n$ and each row of S is viewed as independently drawn from $\mathcal{B}_\sigma^l$ as well. Via appropriately choosing the parameter σ, both rows and columns of S are of small weight.

According to the assumption of syndrome decoding problem, the public key pair (H, T) is computational indistinguishable from uniform distribution.

The hash function $\mathcal{H}$ in the signing algorithm plays the role of random oracle. The output of $\mathcal{H}$ is limited to $\mathcal{S}_w^\ell$ such that $\binom{\ell}{w} \geq 2^\lambda$. We provide an implementation of the hash function $\mathcal{H}$ in Section 6.4.

The signing algorithm first samples a vector $\mathbf{e}$ from the binomial distribution $\mathcal{B}_\tau^n$. Here the parameter τ is a constant and far greater than the secret key distribution parameter σ for two reasons. First larger τ makes that $\mathbf{e}$ aptly hides $S\mathbf{c}$. Second larger τ makes the rejection rate for a predefined weight range in the rejection sampling step in an acceptable range. We always take τ as a constant such that given $(H, H\mathbf{e})$, recovering $\mathbf{e}$ remains hard. From the complexity of information set decoding algorithm, if the work factor for recovering one column of S from (H, T) is beyond the security level, the work factor for recovering $\mathbf{e}$ from $(H, H\mathbf{e})$ is beyond the security level since $\sigma < \tau$. From the Hoeffding bound, we have $n\tau - \gamma \leq \mathrm{wt}(\mathbf{e}) \leq n\tau + \gamma$ except with negligible probability $2^{-\lambda}$, where $\gamma = \sqrt{\frac{\lambda n}{2 \log e}}$. Thus, we choose τ such that $n\tau + \gamma \leq (n-k)/2$. Additionally, we choose τ such that $h(\tau) + k > n$. According to the leftover hash lemma, the statistical distance between $(H, H\mathbf{e})$ and the corresponding uniform distribution is $\frac{1}{2^{(nH(\tau)+k-n)/2}}$. This choice of τ results in that the weight of $\mathbf{e}$ and the signature vector is above the GV bound, which turns the rejection sampling operation to be meaningful.

There are two rejections in the signing algorithm. The first rejection happens when $\mathrm{wt}(\mathbf{z})$ is outside the range $[n\tau - \xi, n\tau + \xi]$. To bound the rejection probability, we need to compute the probability that $\mathrm{wt}(\mathbf{z}) \notin [n\tau - \xi, n\tau + \xi]$. The second rejection happens in the rejection sampling procedure. From the rejection sampling lemma, we know it outputs something with probability

$\frac{1}{M}$.

The signing algorithm responses with the vector $\mathbf{z} = S\mathbf{c} + \mathbf{e}$. Then $\mathbf{z}$ is distributed according to the distribution $\mathcal{B}^n_{S\mathbf{c},\tau}$. Next, the signing algorithm uses rejection sampling to adjust output distribution such that the output distribution is identical to $\widetilde{\mathcal{B}}^n_{\xi,\tau}$. The desired distribution $\widetilde{\mathcal{B}}^n_{\xi,\tau}$ enables the signature decoupled from $S\mathbf{c}$. Thus, the output signature will not reveal information on the secret key S.

If the signing algorithm does not output the signature, then the signing algorithm repeats until it outputs a valid signature. Next we estimate the probability that the signing algorithm outputs a signature of each run. We first bound the probability that $\mathrm{wt}(\mathbf{z})$ belongs to the range $[n\tau - \xi, n\tau + \xi]$. From the distribution of the secret key S, the challenge vector $\mathbf{c}$ and the pilling-up lemma, the vector $S\mathbf{c}$ admits the distribution $\mathcal{B}^n_\eta$, where $\eta = \frac{1}{2}(1 - (1 - 2\sigma)^w) \approx \sigma w$ for very small $\sigma \cdot w$. Then $S\mathbf{c} + \mathbf{e}$ admits the distribution $\mathcal{B}^n_\rho$ with $\rho = \eta(1-\tau) + (1-\eta)\tau$. Thus,

$$Pr\left[\mathrm{wt}(\mathbf{z}) > n\tau + \xi\right] = Pr\left[\mathrm{wt}(\mathbf{z}) - n\rho \geq n\tau + \xi - n\rho\right] \leq \mathrm{e}^{-2(n\tau+\xi-n\rho)^2/n}$$

and

$$Pr\left[\mathrm{wt}(\mathbf{z}) < n\tau - \xi\right] = Pr\left[\mathrm{wt}(\mathbf{z}) - n\rho \leq n\tau - \xi - n\rho\right] \leq \mathrm{e}^{-2(n\tau-\xi-n\rho)^2/n}.$$

Then $Pr\left[\mathrm{wt}(\mathbf{z}) > n\tau + \xi \text{ or } \mathrm{wt}(\mathbf{z}) < n\tau - \xi\right] \leq \mathrm{e}^{-2(n\tau+\xi-n\rho)^2/n} + \mathrm{e}^{-2(n\tau-\xi-n\rho)^2/n} \leq 2\mathrm{e}^{-2(n\tau+\xi-n\rho)^2/n}$ as $\rho > \tau$. Next, from the rejection sampling lemma, the last step outputs something with probability $\frac{1}{M}$. Thus, the signing algorithm outputs something with probability at least $(1 - 2\mathrm{e}^{-2(n\tau+\xi-n\rho)^2/n})/M$.

The parameter ξ for the signature is chosen such that the whole rejection rate of the signing algorithm as small as possible. Besides, the $[n\tau-\xi, n\tau+\xi]$ lies in the hard range of the syndrome decoding problem. According to the rejection sampling lemma, to make the rejection rate in an acceptable range, we choose n such that the magnitude of n is greater than the square of $\mathrm{wt}(S\mathbf{c})$ and greater than $\mathrm{wt}(S\mathbf{c}) \cdot \xi$.

We now explain how to compute the rejection vector. Note that in the rejection sampling lemma, the rejection vector is only related to $\mathrm{wt}(S\mathbf{c}), \mathrm{wt}(\mathbf{z})$ and the parameter τ. Thus, the rejection vector is independent from the secret key, the challenge vector and the signature vector, and can be precomputed. Given $(n, \tau, \mathrm{wt}(S\mathbf{c}), \xi)$, we follow Lemma to compute the rejection vector 3.6.

The output signature for a given message μ is the vector pair $(\mathbf{z}, \mathbf{c})$. As the signature is transformed from the sigma protocol via Fiat-Shamir transformation, thus we refer to the vector $\mathbf{y}, \mathbf{c}, \mathbf{z}$ as the commitment vector, the challenge vector and the response vector, respectively. Sometimes, we refer to the vector $\mathbf{z}$ as the *signature vector*. We remark that the vector $\mathbf{c}$ can be compressed as $\mathbf{c}$ is a sparse vector.

The verifying algorithm verifies that the weight of the response vector $\mathbf{z}$ belongs to the range $[n\tau - \xi, n\tau + \xi]$ and $\mathbf{c} = \mathcal{H}(H\mathbf{z} - T\mathbf{c}, \mu)$ holds as $H\mathbf{z} = H(S\mathbf{c} + \mathbf{e}) = T\mathbf{c} + H\mathbf{e}$. Here the verifier is unable to straightforwardly verify that the signature admits the truncated binomial distribution. The fact that the signature admits truncated binomial distribution is used to argue the signature is decoupled from the secret key part.

Remark 4.1. *Our signature scheme is transformed from a zero-knowledge proof. We expect the underlying zero-knowledge protocol of our signature scheme has other applications to LPN or code based cryptography.*

5 Security Proof

In this section, we provide a security proof to our signature scheme in the random oracle model. We prove that if there exists an adversary breaks the existenial unforgeability game, then the forger can be used to solve the DOOM problem.

If we set the parameter τ small enough, then the corresponding syndrome function is collision resistant. For example, if $H\mathbf{e}_1 = H\mathbf{e}_2$ with $\mathbf{e}_1 \leq n\tau + \xi$ and $\mathbf{e}_2 \leq n\tau + \xi$, then $H(\mathbf{e}_1 - \mathbf{e}_2) = 0$ and $\mathrm{wt}(\mathbf{e}_1 - \mathbf{e}_2) \leq 2n\tau + 2\xi$. If we choose the parameter such that $2n\tau + 2\xi < (n-k)/2$, the syndrome function is collision resistant. In our parameter setting, to take the smallest possible parameters for code length and the rejection rate, we will not choose τ such that the syndrome function is collision resistant. Here we provide a security proof of the scheme based on the DOOM problem. The DOOM problem has been studied many years, which was proposed by Johansson and Jönsson in [JJ02]. Moreover, the security proof of the WAVE [DST19] signature scheme builds on the DOOM problem as well. For completeness, we provide a security proof that the *strong* EUF-CMA security of the signature scheme is reduced to the codeword finding problem for relaxed parameters in Appendix A.

Theorem 5.1. *Assume that there is a forger that breaks the EUF-CMA game. Then there exists an algorithm that solves the DOOM problem.*

In particular, if the forger $\mathcal{F}$ succeeds with probability δ, assuming the forger $\mathcal{F}$ makes at most s signature query and h hash query, there exists an algorithm $\mathcal{A}$ that solves the DOOM instance with h syndromes with probability at least $\frac{\delta}{n^{1/4}} - 2^{-O(n)}$.

We prove the security via a series of hybrid games. For the forger $\mathcal{F}$, there are two kinds of queries, including the signing query and the hash query. Via programming the random oracle, the forger is forced to produce a signature corresponding to one of the syndromes of DOOM instance and thus solve the DOOM problem.

Proof. Given an instance $(H, \mathbf{s}_1, \ldots, \mathbf{s}_h)$ of DOOM, one samples a secret key S first and then publish $(H, T = HS)$ as the public key.

Game 1 is the real signature scheme. Assume $\mathcal{H}$ is the real random oracle function and Hash is an encapsulation of $\mathcal{H}$. We explicitly show the Game 1 here.

```
Sign(sk, $\mu$, $\tau$, $\xi$)
  1. Sample $\mathbf{e}$ from $\mathcal{B}_\tau^n$.
  2. $\mathbf{c} = \mathsf{Hash}(H\mathbf{e}, \mu)$.
  3. Let $\mathbf{z} = S\mathbf{c} + \mathbf{e}$.
  4. If $\mathrm{wt}(\mathbf{z}) \notin [n\tau - \xi, n\tau + \xi]$, then restart.
  5. With probability $\frac{\widetilde{\mathcal{B}}_{\xi,\tau}(\mathbf{z})}{M\mathcal{B}_{S\mathbf{c},\tau}(\mathbf{z})}$
  6.     Output $(\mathbf{z}, \mathbf{c})$.

Hash($\mathbf{y}$, $\mu$)
  1. return $\mathcal{H}(\mathbf{y}, \mu)$.
```

Figure 1: Game 1

In Game 2, we use a table to store all the queries. The table is a global variable such that both Sign and Hash have access to it. The views between Game 1 and Game 2 for the forger $\mathcal{F}$ keep unchanged. Thus, the Game 2 will not alter the forging advantage of $\mathcal{F}$.

In Game 3, the Sign procedure is changed to program the random oracle and the Hash procedure is unchanged. In Game 3, the Sign procedure directly program the random oracle without checking whether the value $(H\mathbf{e}, \mu)$ is set or not. From the assumption of $\mathcal{F}$, we know there are at most $h + s$ queries to the random oracle $\mathcal{H}$. We claim that for a programming the oracle query, it collides with one previous query with probability $2^{-O(n)}$. As there are at most $(h + s)$ values of the random oracle been set, by union bound, the programming oracle

```
Sign(sk, $\mu, \tau, \xi$)
  1. Sample $\mathbf{e}$ from $\mathcal{B}_\tau^n$.
  2. $\mathbf{c} = \mathsf{Hash}(H\mathbf{e}, \mu)$.
  3. Let $\mathbf{z} = S\mathbf{c} + \mathbf{e}$.
  4. If $\mathrm{wt}(\mathbf{z}) \notin [n\tau - \xi, n\tau + \xi]$, then restart.
  5. Store $(H\mathbf{e}, \mu, \mathbf{c})$ to the table.
  6. With probability $\frac{\widetilde{\mathcal{B}}_{\xi,\tau}(\mathbf{z})}{M\mathcal{B}_{S\mathbf{c},\tau}(\mathbf{z})}$
  7.     Output $(\mathbf{z}, \mathbf{c})$.

Hash($\mathbf{y}, \mu$)
  1. If table contains $(\mathbf{y}, \mu, \mathbf{c})$ then
  2.     return $\mathbf{c}$
  3. $\mathbf{c} = \mathcal{H}(\mathbf{y}, \mu)$ and store $(\mathbf{y}, \mu, \mathbf{c})$ to the table.
  4. return $\mathbf{c}$.
```

Figure 2: Game 2

```
Sign(sk, $\mu, \tau, \xi$)
  1. Sample $\mathbf{e}$ from $\mathcal{B}_\tau^n$.
  2. Sample $\mathbf{c} \in \mathcal{S}_\omega^n$.
  3. Let $\mathbf{z} = S\mathbf{c} + \mathbf{e}$.
  4. If $\mathrm{wt}(\mathbf{z}) \notin [n\tau - \xi, n\tau + \xi]$, then restart.
  5. With probability $\frac{\widetilde{\mathcal{B}}_{\xi,\tau}(\mathbf{z})}{M\mathcal{B}_{S\mathbf{c},\tau}(\mathbf{z})}$
  6.     Output $(\mathbf{z}, \mathbf{c})$.
  7.     Program $\mathcal{H}(H\mathbf{e}, \mu) = \mathbf{c}$.
  8.     Store $(H\mathbf{e}, \mu, \mathbf{c})$ to the table.

Hash($\mathbf{y}, \mu$)
  1. If table contains $(\mathbf{y}, \mu, \mathbf{c})$ then
  2.     return $\mathbf{c}$
  3. $\mathbf{c} = \mathcal{H}(\mathbf{y}, \mu)$ and store $(\mathbf{y}, \mu, \mathbf{c})$ to the table.
  4. return $\mathbf{c}$.
```

Figure 3: Game 3

query collides with one of the previous queries with probability at most $(h+s)2^{-O(n)}$. Thus, one of the signing programming oracle collides with previous queries with probability at most $s(h+s)2^{-O(n)}$. If there are no collisions, Game 3 provides the same view as Game 2 to the forger $\mathcal{F}$. Thus, the statistical distance between the views of Game 2 and Game 3 is at most $s(h+s)2^{-O(n)}$.

Now we prove the claim. If the programming oracle query collides with one previous query, then $H\mathbf{e}$ equals to the previous query value. According to our parameter setting that $H(\tau)+k/n-1$ is a positive constant, $H\mathbf{e}$ is statistical close to the uniform distribution. Concretely, from the leftover hash lemma, the statistical distance is $2^{-n(H(\tau)+k-n)/2} = 2^{-O(n)}$. For a uniformly sampled vector $\mathbf{y} \in \mathbb{F}_2^{n-k}$, the probability that $\mathbf{y}$ equals to a given value is $2^{-(n-k)} = 2^{-O(n)}$ as k is linear of n. Thus, the programming random oracle query collides with a given query with probability at most $2^{-n(H(\tau)+k-n)/2} + 2^{-(n-k)} = 2^{-O(n)}$.

```
Sign(sk, $\mu, \tau, \xi$)
1. Sample $\mathbf{c} \in \mathcal{S}_\omega^n$.
2. Sample $\mathbf{z}$ from $\mathcal{B}_\tau^n$.
3. If wt($\mathbf{z}$) $\notin [n\tau - \xi, n\tau + \xi]$, then restart.
4. With probability $\frac{1}{M}$
5.     Output $(\mathbf{z}, \mathbf{c})$.
6.     Program $\mathcal{H}(H\mathbf{z} - T\mathbf{c}, \mu) = \mathbf{c}$.
7.     Store $(H\mathbf{z} - T\mathbf{c}, \mu, \mathbf{c})$ to the table.
```

```
Hash($\mathbf{y}, \mu$)
1. If table contains $(\mathbf{y}, \mu, \mathbf{c})$ then
2.     return $\mathbf{c}$
3. $\mathbf{c} = \mathcal{H}(\mathbf{y}, \mu)$ and store $(\mathbf{y}, \mu, \mathbf{c})$ to the table.
4. return $\mathbf{c}$.
```

Figure 4: Game 4

In Game 4, sk is not employed in the Sign procedure. The views to the forger $\mathcal{F}$ between Game 3 and Game 4 are identical by the rejection sampling lemma for binary linear codes, i.e., Lemma 3.16.

In Game 5, the hash query is replaced by the DOOM instance. According to the output distribution of $\mathcal{H}$, the distribution for $\mathcal{H}(\mathbf{y}, \mu)$ and $\mathcal{H}(\mathbf{s}_i, \mu)$ are same. Thus, the forger is unable to tell apart Game 4 from Game 5. The replacing in Game 5 forces the forger to provide a clue to a solution of the DOOM problem.

Assume the forger $\mathcal{F}$ forges a signature $(\mathbf{z}, \mathbf{c})$ for a message μ. Then we have $\mathcal{H}(H\mathbf{z} - T\mathbf{c}, \mu) = \mathbf{c}$. If the Hash function is never queried, then $\mathcal{F}$ can forge such a signature with probability $1/|\mathcal{S}_w^n|$. According to the parameter setting, then the probability is $2^{-O(\lambda)}$.

Note that in the EUF-CMA model, if the forged signature for a message μ is output, then the Sign query is never requested for μ. Thus, with probability $1-1/|\mathcal{S}_w^n|$, the vector $\mathbf{c}$ must be one of the query result from Hash. It means there exists $\mathbf{s}_i$ and μ' such that $\mathcal{H}(H\mathbf{z} - T\mathbf{c}, \mu) = \mathcal{H}(\mathbf{s}_i, \mu')$. If $\mu \neq \mu'$ or $H\mathbf{z} - T\mathbf{c} \neq \mathbf{s}_i$, then the forger outputs a preimage of $\mathbf{c}$. It happens with negligible probability. Thus with overwhelming probability $\mu = \mu'$ and $H\mathbf{z} - T\mathbf{c} = \mathbf{s}_i$. It is $H(\mathbf{z} - S\mathbf{c}) = \mathbf{s}_i$. Thus, $\mathbf{z} - S\mathbf{c}$ is a preimage of $\mathbf{s}_i$. Next we provide of bound to the weight of $\mathbf{z} - S\mathbf{c}$. From the distribution of S and pilling-up lemma, we have the vector $S\mathbf{c}$ admits the distribution $\mathcal{B}_\eta^n$, where $\eta = \frac{1}{2}(1-(1-2\sigma)^w) \approx \sigma w$ for very small σw. The probability that adding $S\mathbf{c}$ to $\mathbf{z}$ does

```
Sign(sk, $\mu, \tau, \xi$)
1. Sample $\mathbf{c} \in \mathcal{S}_\omega^n$.
2. Sample $\mathbf{z}$ from $\mathcal{B}_\tau^n$.
3. If $\mathrm{wt}(\mathbf{z}) \notin [n\tau - \xi, n\tau + \xi]$, then restart.
4. With probability $\frac{1}{M}$
5.     Output $(\mathbf{z}, \mathbf{c})$.
6.     Program $\mathcal{H}(H\mathbf{z} - T\mathbf{c}, \mu) = \mathbf{c}$.
7.     Store $(H\mathbf{z} - T\mathbf{c}, \mu, \mathbf{c})$ to the table.
```

```
Hash($\mathbf{y}, \mu$)
1. If table contains $(\mathbf{y}, \mu, \mathbf{c})$ then
2.     return $\mathbf{c}$
3. Choose the first unused $\mathbf{s}_i$ and compute $\mathbf{c} = \mathcal{H}(\mathbf{s}_i, \mu)$.
4. Store $(\mathbf{y}, \mu, \mathbf{c})$ to the table.
5. Return $\mathbf{c}$.
```

Figure 5: Game 5

not increase the weight of $\mathbf{z}$ is

$$\sum_{\mathrm{wt}(S\mathbf{c})/2 \le j \le \mathrm{wt}(S\mathbf{c})} \binom{\mathrm{wt}(\mathbf{z})}{j} \binom{n - \mathrm{wt}(\mathbf{z})}{\mathrm{wt}(S\mathbf{c}) - j} \eta^{\mathrm{wt}(S\mathbf{c})} (1-\eta)^{n - \mathrm{wt}(S\mathbf{c})} \le \binom{n}{\mathrm{wt}(S\mathbf{c})} \eta^{\mathrm{wt}(S\mathbf{c})} (1-\eta)^{n - \mathrm{wt}(S\mathbf{c})}.$$

From the Hoeffding bound, we have that $\mathrm{wt}(S\mathbf{c})$ is around $n\eta$. Thus, the probability for weight value $n\eta$ becomes $\binom{n}{\mathrm{wt}(S\mathbf{c})} \eta^{\mathrm{wt}(S\mathbf{c})}(1-\eta)^{n-\mathrm{wt}(S\mathbf{c})} = \binom{n}{n\eta}\eta^{n\eta}(1-\eta)^{n-n\eta} \approx \frac{1}{n^{1/4}}$. For the case that $\mathrm{wt}(S\mathbf{c}) \neq n\eta$, the probability is close to $\frac{1}{n^{1/4}}$. Thus, we use $\frac{1}{n^{1/4}}$ to approximate the probability that $\mathrm{wt}(\mathbf{z} - S\mathbf{c})$ is below $n\tau + \xi$.

From the series of games, if $\mathcal{F}$ breaks the Game 1 with probability δ, then it breaks the Game 6 with probability at least $\delta - 2^{-O(n)}$. Thus, it can be used to solve the DOOM problem with probability at least $\frac{\delta}{n^{1/4}} - 2^{-O(n)}$.

This completes the proof. □

6 Instantiation and Parameters

In this section, we instantiate the signature scheme based on the Ring-LPN problem or the quasi-cyclic syndrome decoding problem. We first explain why we do not directly instantiate our scheme based on the plain LPN problem or the syndrome decoding problem. The major reason is the huge public key size. Suppose we expect to achieve λ-bit security level. From the rejection sampling lemma, we need to set the code length at least $n \ge \lambda^2$ and the weight of $\mathbf{z}$ to be the magnitude of λ^2. To make the forging problem in the hard range, $n - k$ should be at least twice as the weight of $\mathbf{z}$. Then the magnitude of the public key size is at least λ^4 bits. For instance $\lambda = 128$, the public key size is at least 2^{28} bits. Thus, we choose the quasi-cyclic structure to reduce the public key size without compromising the security level too much.

Notice that instantiating the signature scheme with quasi-cyclic codes only reduces the public key size to an acceptable scale and has no significant effect on the signature size.

We first explicitly show the quasi-cyclic key generation, signing and the verifying algorithms. Next we present the points that we considered for the quasi-cyclic instantiation. The quasi-cyclic instantiation works over the ring $\mathcal{R} := \mathbb{F}_2[x]/(x^n - 1)$. In what follows, we denote n as the ring extension degree over $\mathbb{F}_2$ and dn as the code length for the quasi-cyclic code with index d. We list two key generation algorithms for different code rate as the underlying assumptions are different for code rate $1/2$ and $1/d$ for $d > 2$.

Algorithm 4 KeyGen for Rate 1/2 Quasi-cyclic Code

```
Require: Parameters $n, u$.
Ensure: $\mathbf{h} \in \mathbb{F}_2^n$ or $\mathcal{R}$.
1: Uniformly sample $\mathbf{s}_1, \mathbf{s}_2$ from $\mathcal{S}_u^n$.
2: Let $h(x) = \frac{s_2(x)}{s_1(x)} \in \mathcal{R}$.
3: Output $\mathbf{h}$ as the public key and $(\mathbf{s}_1, \mathbf{s}_2)$ as the secret key pair.
```

Algorithm 5 KeyGen for Rate $1/d$ Quasi-cyclic Code

```
Require: Parameters $n, u, d$.
Ensure: $\mathbf{h} \in \mathcal{R}^{d-1}, \mathbf{t} \in \mathcal{R}^{d-1}$
1: Uniformly sample $\mathbf{h}_1, \ldots, \mathbf{h}_{d-1}$ from $\mathcal{R}$.
2: Uniformly sample $\mathbf{s}_1, \ldots, \mathbf{s}_d$ from $\mathcal{S}_u^n$.
3: Let $\mathbf{t}_j = \mathbf{h}_j \mathbf{s}_j + \mathbf{s}_d$ for $j \in \{1 \ldots d-1\}$.
4: Let $\mathbf{h} := (\mathbf{h}_1 \ldots \mathbf{h}_{d-1})$, $\mathbf{t} := (\mathbf{t}_1 \ldots \mathbf{t}_{d-1})$ and $\mathbf{s} := (\mathbf{s}_1 \ldots \mathbf{s}_d)$.
5: Output $(\mathbf{h}, \mathbf{t})$ as the public key pair and $\mathbf{s}$ as the secret key.
```

Algorithm 6 Quasi-cyclic Signing

```
Require: The public key pair $(\mathbf{h}, \mathbf{t})$, the secret key $\mathbf{s}$, the message $\mu$, the parameter $\tau < 1/2$, the hash function $\mathcal{H}$ and the range parameter $\xi$.
Ensure: A signature $(\mathbf{z}, \mathbf{c})$ for the message $\mu$.
1: Sample $\mathbf{e}_j$ from $\mathcal{B}_\tau^n$ for $j \in \{1 \ldots d\}$. Let $\mathbf{e} := (\mathbf{e}_1 \ldots \mathbf{e}_d)$.
2: Compute $\mathbf{y}_j = \mathbf{h}_j \mathbf{e}_j + \mathbf{e}_d$ for $j \in \{1 \ldots d-1\}$. Let $\mathbf{y} := (\mathbf{y}_1 \ldots \mathbf{y}_{d-1})$.
3: Let $\mathbf{c} = \mathcal{H}(\mathbf{y}, \mu)$.
4: Compute $\mathbf{z} = \mathbf{s} \cdot \mathbf{c} + \mathbf{e}$.
5: If $\mathrm{wt}(\mathbf{z}) \notin [dn\tau - \xi, dn\tau + \xi]$, then restart.
6: Output $(\mathbf{z}, \mathbf{c})$ with probability $\frac{\widetilde{\mathcal{B}}_{\xi,\tau}(\mathbf{z})}{M\mathcal{B}_{\mathbf{s}\cdot\mathbf{c},\tau}(\mathbf{z})}$, where $M = \max_{j\in[dn\tau-\xi,dn\tau+\xi]} \frac{\widetilde{\mathcal{B}}_{\xi,\tau}(j)}{\mathcal{B}_{\mathbf{s}\cdot\mathbf{c},\tau}(j)}$.
```

Algorithm 7 Quasi-cyclic Verifying

```
Require: Public key pair $(\mathbf{h}, \mathbf{t})$, the message $\mu$, the signature $(\mathbf{z}, \mathbf{c})$ and the range parameter $\xi$.
Ensure: Accept or Reject
1: If $\mathrm{wt}(\mathbf{z}) \notin [dn\tau - \xi, dn\tau + \xi]$, then output Reject.
2: Output Accept if $\mathcal{H}(\sum_{i=1}^{d-1}(\mathbf{h}_i \mathbf{z}_i - \mathbf{b}_i \mathbf{c} + \mathbf{s}), \mu) = \mathbf{c}$, otherwise Reject.
```

6.1 The Key Generation Algorithm

There are two key generation procedures for code rate $1/2$ and $1/d$, respectively. For code rate $1/2$, we use the assumption $\mathbf{h} = \frac{\mathbf{s}_1}{\mathbf{s}_2}$ in key generation to reduce the public key size by half. The assumption is also used in BIKE. For code rate $1/d$, we use the syndrome decoding problem for quasi-cyclic code to produce the public key. The quasi-cyclic syndrome decoding problem is also used in HQC.

Suppose we want to achieve the security level λ. To make it easy to estimate the security level of public key, we directly sample the secret key from $\mathcal{S}^{dn}_{du}$ rather than $\mathcal{B}^{dn}_{u/n}$, where u is some predefined weight. If sk is sampled from $\mathcal{B}^{dn}_{u/n}$, then we need to set a little larger u to keep the target security level as the weight of sk lies in some range. If sk is sampled from $\mathcal{S}^{dn}_{du}$ and the code rate is fixed, we can directly estimate the attack complexity via the CaWoF library [Tor17, TS16] as du is sublinear of dn.

As pointed out in Section 2.3, for sublinear error weight ω, the work factor of information set decoding algorithm is $2^{c\omega(1+o(1))}$, where r is the code rate and $c = -\log(1-r)$. According to Section 2.2, for codeword finding problem, the quasi-cyclic structure provides a speedup factor of n and for syndrome decoding problem the speedup factor is $\sqrt{n}$. Thus, for code rate $1/2$, we choose $2u - \log n > \lambda$. For code rate $1/d$, we choose $du \log \frac{d}{d-1} - \log \sqrt{n} > \lambda$.

6.2 The Ring $\mathcal{R}$

According to Remark 2.16(ii) , we choose a prime n such that $(x^n-1)/(x-1) \in \mathbb{F}_2[x]$ is irreducible. The polynomial inversion is involved in the Algorithm 4. Drucker et al. [DGK20] proposed an algorithm to combine the Itoh-Tsuji algorithm to perform polynomial inversion over the ring $\mathbb{F}_2[x]/((x-1)P)$, where $P = (x^n-1)/(x-1) \in \mathbb{F}_2[x]$ is an irreducible polynomial. To make the inversion algorithm more efficient, Drucker et al. recommends choosing n such that the binary representation of $n-2$ is of small Hamming weight.

6.3 Signing Algorithm

6.3.1 Parameters in Signing Algorithm

The parameter τ is a constant only associated to the code rate r. The parameter τ is chosen a little smaller than $(1-r)/2$ to make the syndrome decoding problem for the commitment and for the signature remains hard. The parameter ξ is chosen to achieve the smallest total rejection rate and $[dn\tau - \xi, dn\tau + \xi]$ lies in the hard range of the syndrome decoding problem. According to the rejection sampling lemma, the parameter ξ is chosen such that $\xi \mathrm{wt}(\mathbf{sc}) \leq dn$.

Remark 6.1. *If we choose a smaller code rate, we can obtain a larger value for τ. Thus, we can choose a smaller n to reduce the public key size and signature size without compromising the rejection rate. However for a smaller rate r, the security of the public key is compromised. We need to increase the weight of the secret key to keep the security level. To maintain the acceptable rejection rate, the code length needs to be increased. Therefore, we need to find a tradeoff between the code rate r and the code length. A rough inspection indicates that smaller code rate will not efficiently reduce the code length, whereas a smaller code rate r makes the ring degree n being small, which makes the multiplication over the ring more efficient.*

6.3.2 Multiplication in Signing Algorithm

There are two multiplication operations in the signing algorithm. The first multiplication is to compute the commitment. The second multiplication is to produce the candidate signature vector $\mathbf{z} = \mathbf{sc}+\mathbf{e}$. Because $\mathbf{c}$ is sparse, the multiplication $\mathbf{sc}$ can be optimized to compute a small number of sum of quasi-cyclic rotations of $\mathbf{s}$ since $\mathbf{sc} = \sum_{i \in \mathsf{Supp}(\mathbf{c})} \mathbf{s}x^i$. Thus the sparse vector multiplication over $\mathcal{R}$ is converted to $\mathrm{wt}(\mathbf{c})$ additions over $\mathcal{R}$. For the commitment, although the weight of $\mathrm{wt}(\mathbf{e})$ is small, it is still not a sparse vector. The multiplication is the most time-consuming operation in the signing algorithm. There are two offline methods to optimize the multiplication operation. One method is to sample a large number $\mathbf{e}$, precompute $\mathbf{y} = \mathbf{he}$ and store the pair $(\mathbf{e}, \mathbf{y})$ to a local table. When signing a message, the signing algorithm directly picks an unused $(\mathbf{e}, \mathbf{y})$ from the local table and the signing algorithm takes almost constant time to finish the expensive multiplication. The main drawback of this method is the uncontrollable large table size. Once there is no unused pair in the table, the signing algorithm needs to run the offline algorithm to update the local table. An alternative method is built upon the assumption that there are at most 2^λ messages to be signed by the given secret key. The method first chooses parameter (ℓ, ψ) such that $\binom{\ell}{\ell/2} \geq 2^\lambda$ and $\frac{1}{2}(1-(1-\psi)^{\ell/2}) = \tau$. Next the method samples ℓ number of $\mathbf{e}$ from the distribution $\mathcal{B}_\psi^{dn}$, computes the corresponding syndrome $\mathbf{y} = \mathbf{he}$ and store the pair $(\mathbf{e}, \mathbf{y})$ to a local table. The signing algorithm each time randomly picks $\ell/2$ pairs $(\mathbf{e}, \mathbf{y})$ from the local table and sums up the $\ell/2$ entries from the table to obtain a fresh $(\mathbf{e}', \mathbf{y}')$ pair. The fresh vector $\mathbf{e}'$ admits the distribution $\mathcal{B}_\tau^{dn}$ from the parameter setting. The correctness of the fresh $(\mathbf{e}', \mathbf{y}')$ pair follows the homomorphic property of the syndrome function for linear codes. The advantage of this method is that under a reasonable assumption, the storage cost for the precomputation is fixed and the inefficient polynomial multiplication is converted to $\ell/2$ number of additions of entries of the local table. Additionally, in the precomputation procedure, each vector $\mathbf{e}$ is sampled from $\mathcal{B}_\psi^{dn}$ and thus sparse. Hence the sparse offline multiplication can be optimized to compute the sum of quasi-cyclic rotations of the public key.

6.4 Hash Function

Assume the weight of the output of the hash function is w. To make it hard to forge a preimage for the hash function, we choose w satisfying $\binom{n}{w} \geq 2^\lambda$. For large n, w is very small and thus $\mathbf{c}$ is sparse. As sk is sampled from $\mathcal{S}_{du}^{dn}$, then the distribution of $\mathbf{sc}$ is not easy to describe.

There are two possible instantiations for the hash function $\mathcal{H}$. One method is to map the message μ and the vector $\mathbf{y}$ to a number of less than λ bits via a cryptographic hash function like SHA-512, and then convert the number to a binary vector with length n and weight w via a bijection. This method is very inefficient for large n. An alternative method is to use the cryptographic hash function to map the message μ and the vector $\mathbf{y}$ to $2w$ numbers, where each number is less than n and each number specifies a position of the support of the challenge vector. Mapping $2w$ numbers to a weight w challenge vector works as follows. If the current number is already set in the challenge vector, then it is omitted until the vector achieves the weight w. With non-negligible probability, the $2w$ numbers specify at least w distinct positions of the challenge vector. Such a method can be found in [DDLL13].

6.5 Rejection Vector and Rejection Rate

The rejection vector is used in the rejection sampling step to decide the probability of outputting a candidate signature $(\mathbf{z}, \mathbf{c})$. Given the parameter $(n, \tau, \xi, \mathrm{wt}(\mathbf{sc}))$, the rejection vector can be precomputed as it is independent from the vectors. For concrete parameters, we follow Lemma 3.6 to compute the rejection vector as $\mathrm{wt}(\mathbf{sc})$ is sublinear of code length. From the rejection sampling lemma for binary linear codes, i.e., Lemma 3.16, we know the rejection sampling step outputs something with expected probability $\frac{1}{M}$.

In fact, there are two rejections in the signing algorithm. The first rejection happens when the weight of $\mathbf{z}$ does not belong to $[dn\tau - \xi, dn\tau + \xi]$. The second rejection happens in the rejection sampling step. To compute the rejection rate, we need to compute the probability of $\mathrm{wt}(\mathbf{z}) \in [dn\tau - \xi, dn\tau + \xi]$. Before computing the weight distribution of $\mathbf{z}$, we need to estimate $\mathrm{wt}(\mathbf{sc})$, which is also related to the rejection vector.

It is easy to obtain $\mathrm{wt}(\mathbf{sc}) \leq \mathrm{wt}(\mathbf{s})\mathrm{wt}(\mathbf{c})$. From the quasi-cyclic structure, the vector $\mathbf{sc}$ can be viewed as the sum of $\mathrm{wt}(\mathbf{c})$ vectors, where each is of weight $\mathrm{wt}(\mathbf{c})$. Thus $\mathrm{wt}(\mathbf{sc})$ is bounded by $\mathrm{wt}(\mathbf{s})\mathrm{wt}(\mathbf{c})$. Next we argue that $\mathrm{wt}(\mathbf{sc}) = \mathrm{wt}(\mathbf{s})\mathrm{wt}(\mathbf{c})$ holds with high probability. From the rejection sampling lemma, we need to set $\mathrm{wt}(\mathbf{sc})$ to be sublinear of the code length.

Thus, $\mathbf{s}$ and $\mathbf{c}$ are both extremely sparse vector. Consider the computation $\mathbf{sc} = \sum_{i \in \mathsf{Supp}(\mathbf{c})} \mathbf{s}x^i$. The vector $\mathbf{s}x^i$ can be viewed as the i-th quasi-cyclic rotation of $\mathbf{s}$. Because $\mathbf{s}$ is very sparse, the support of $\mathbf{s}x^i$ and $\mathbf{s}x^j$ for distinct i, j has no intersection with high probability. Thus, $\mathrm{wt}(\mathbf{sc}) = \mathrm{wt}(\mathbf{s})\mathrm{wt}(\mathbf{c})$ holds with high probability. In the precomputation of the rejection vector, we also use the result $\mathrm{wt}(\mathbf{sc}) = \mathrm{wt}(\mathbf{s})\mathrm{wt}(\mathbf{c})$. Next we compute the probability that $\mathrm{wt}(\mathbf{sc}+\mathbf{e}) = t$ for a given $t \in [dn\tau - \xi, dn\tau + \xi]$. Assume $\mathrm{wt}(\mathbf{sc}) = \beta$. Following the proof of Lemma 3.3, we have

$$Pr[\mathrm{wt}(\mathbf{z}) = t] = \sum_{i=0}^{\beta} \binom{\beta}{i} \binom{dn-\beta}{t-\beta+i} \tau^{t-\beta+2i}(1-\tau)^{dn-t+\beta-2i}.$$

The total probability that the signing algorithm outputs something is $\sum_{t=dn\tau-\xi}^{dn\tau+\xi} Pr[\mathrm{wt}(z) = t]r(t)$, where $\mathbf{r}$ is the rejection vector for the rejection sampling step.

6.6 The Verifying Algorithm

In the verifying step, the multiplication $\mathbf{hz}$ is performed at great expense. It looks like the great cost is inevitable. The hash function evaluation is very efficient according to Section 6.4. We expect future work to improve our scheme to reduce the code length to make the verifying step more efficient.

6.7 Parameters

We propose parameters for the quasi-cyclic instantiation for classic 80-bit and 128-bit security level. For the given parameters, the public key is generated according to Algorithm 4 and thus the code rate is 1/2. The public key size, signature size and acceptance rate are listed as well.

λ	n	u	w	τ	ξ	$\mathrm{wt}(\mathbf{sc})$	pk size	signature size	acceptance rate
80	66467	49	6	0.23925	70	588	8.12kB	16.24 kB	0.020537
128	248579	75	8	0.24305	135	1200	30.35kB	60.71 kB	0.017391

Recall that in our instantiation the code length, secret key weight and the signature weight range are $2n$,$2u$ and $[2n\tau - \xi, 2n\tau + \xi]$ respectively.

Note that in the given parameters, the code length is not chosen strictly greater than the square of $\mathrm{wt}(\mathbf{sc})$. To achieve a smaller public key size and signature size, we decrease the code length at the cost of increasing the rejection rate.

A proof of concept of the scheme is implemented in [LXY20].

6.8 Security Issues

For code rate 1/2, the decisional 2-QCCFP assumption is involved in the key generation procedure. The same assumption is used in the key generation algorithm of the NIST Round 3 candidate BIKE [ABB+18]. In a different variant of the our signature scheme, the 2-QCSDP assumption is involved in the key generation procedure. The same assumption is used in the key generation of the NIST Round 3 candidate HQC [MAB+19].

In our instantiation, if the key generation step is secure, the commitment step and the signature step are both secure. Because the weight of $\mathbf{e}$ and $\mathbf{z}$ are far greater than the weight of the secret key, the work factor for forging a signature for a given syndrome is beyond the security level.

7 Security Discussion

Our signature scheme is an adaption of the Schnorr-Lyubabshevsky framework for the random linear Hamming metric codes. Adaption of the same framework in coding problem is not completely new. There exist several unsuccessful attempts in literature. We individually compare our scheme with the existing schemes and show the essential difference between our scheme and the unsuccessful attempts, hence strengthen the confidence to the security of our signature scheme. We will review the unsuccessful attempts and the corresponding attacks. Next we will consider the consequences when the attacks are applied to our scheme. We will explain the reasons that our scheme does not suffer from existing attacks in detail.

7.1 RaCoSS Scheme

The first attempt is the scheme Random Code-based Signature Scheme(RaCoSS) [FRX+17], which was submitted to NIST Round 1. The scheme is almost as same as the matrix version of our scheme in Section 4. The main difference between RaCoSS and our scheme lies in the parameter selection and the rejection sampling step. In RaCoSS, the secret key is also sampled from a binomial distribution with small parameters and the commitment secret vector is sampled from a binomial distribution as well. The challenger vector is the output of a cryptographic hash function with small fixed Hamming weight. However, the Hamming weight of the output of the hash function is *too small*, which makes the number of possible output vectors less than the claimed security level. An adversary can directly try to find a preimage to the hash function to attack the scheme. Such an attack was formally formed in [BHLP17]. One can readily increase the Hamming weight of the output vector of the hash function to avoid the attack. In our scheme, the number of possible challenger vectors is beyond the security level.

The main problem of the RaCoSS scheme is that the verifying condition of the weight of the signature vector is too large. The weight range of the verifying condition intersects with the *easy* range of the syndrome decoding problem. Thus, an adversary can directly forge a valid signature for any predefined syndrome. Recall that for a parity check matrix $H \in \mathbb{F}_2^{(n-k)\times n}$, the easy range for the syndrome decoding problem is around $[\frac{n-k}{2}, \frac{n+k}{2}]$. An efficient attack employing this property was first proposed by Bernstein, Hulsing, Lange and Panny [BHLP17] after two days the scheme RaCoSS was submitted to NIST. To form an attack, an adversary firstly samples a valid vector $\mathbf{e}$ and computes $\mathbf{y} = H\mathbf{e}, \mathbf{c} = \mathcal{H}(\mathbf{y}, \mu)$ as an honest signer does. Next the adversary solves the syndrome decoding problem $H\mathbf{z} = T\mathbf{c} + \mathbf{y}$ to find $\mathbf{z}$ with the weight of $\mathbf{z}$ in the acceptable easy range. Thus, $(\mathbf{z}, \mathbf{c})$ is a valid signature for the message μ. As the acceptable weight range intersects with the easy range of the syndrome decoding problem, the adversary can efficiently forge a valid signature vector.

Later, Roy et al. [RMF+18] presented a revised version of RaCoSS with the name RaCoSS-R. In RaCoSS-R, the weight of the signature vector is limited to a different range. Unfortunately, in RaCoSS-R, the acceptable range of the signature vector still has overlap with the easy range of the syndrome decoding problem. Thus, an adversary can still efficiently forge a valid signature. Xagawa [Xag18] described such an attack to RaCoSS-R.

In our scheme, the weight of the signature vector $\mathbf{z}$ is strictly less than $(n-k)/2$. Thus, our scheme does not suffer from an easy range syndrome decoding attack. Moreover, given $H \in \mathbb{F}_2^{(n-k)\times n}$ and $H\mathbf{e} = \mathbf{y}$, one can easily pick out a set of $n-k$ linearly independent columns of H with non-negligible probability as H is uniformly sampled. Without loss of generality, assume the first $n-k$ columns are linearly independent and denote the first $n-k$ columns as a submatrix $\overline{H}$ and the subvector of $\mathbf{e}$ corresponding $\overline{H}$ as $\overline{\mathbf{e}}$. Then one can set the entries of $\mathbf{e}$ as 0 for entries outside $\overline{\mathbf{e}}$ and computes the values of $\overline{\mathbf{e}}$ from $\overline{H}$ and $\mathbf{y}$. Because H is uniformly sampled, then $\overline{\mathbf{e}} = \overline{H}^{-1}\mathbf{y}$ is uniformly distributed as well. As $\mathrm{wt}(\mathbf{e}) = \mathrm{wt}(\overline{\mathbf{e}})$, the expectation weight of $\mathbf{e}$ is $(n-k)/2$. One can use the Hoeffding bound to provide a lower bound to $\mathrm{wt}(\mathbf{e})$

as the entries in $\bar{\mathbf{e}}$ are uniformly distributed. In our scheme, the parameter is chosen such that $\mathrm{wt}(\mathbf{z})$ is less than the possible lower bound of the syndrome decoding method for a claimed security level.

Remark 7.1. *The two security proofs of our signature scheme also work for the RaCoSS scheme. The problem is the security of RaCoSS is reduced to codeword finding problem or syndrome decoding problem where the weight range of the two problems intersects with the easy range of the two problems. The result complies with the aforementioned attack.*

7.2 Persichetti's Proposal

In 2018, Persichetti [Per18] proposed an adaption of the Schnorr-Lyubashevsky framework to the quasi-cyclic codes. The public key is an instance of the quasi-cyclic syndrome decoding problem. The signature is another instance of the quasi-cyclic syndrome decoding problem. The scheme is similar to our quasi-cyclic instantiation in the form sense. The weight of signature is below the GV bound. Thus, an adversary is unable to produce a valid signature without the private key. In Persichetti's proposal, because the weight of the signature vector $\mathbf{z}$ is below the GV bound, the weight of the secret key $\mathbf{s}$ and the weight of the vector $\mathbf{e}$ to commit are both below the GV bound. So the vector $\mathbf{e}$ is unable to properly hide the secret key part in the signature. For this reason, Persichetti claimed that the scheme only achieves the security of one-time signature(OTS).

The main difference in parameter setting between Persichetti's proposal and our scheme is that Persichetti's proposal requires the weight of the signature below the GV bound of the public code whereas our scheme only requires the weight of the signature is constrained in the small hard weight range of the syndrome decoding problem. In addition, our scheme uses the rejection sampling method to decouple the dependence of signature on secret key and to adjust the output signature distribution to a truncated binomial distribution to achieve (strong) EUF-CMA security.

In Persichetti's proposal, the weight of the signature vector is required to be below the GV bound to ensure that the signature vector is uniquely determined by the message, commitment and the challenge vector. Our scheme indicates that this is unnecessary. Even if a given syndrome corresponds to many different valid signatures, without the secret key, a PPT forger is still unable to efficiently produce a valid signature from the complexity of the information set decoding algorithm.

Persichetti mentioned that given a polynomial number of honestly generated signatures, an adversary is able to perform a general statistical attack to recover the secret key. The statistical attack is owed to Jean-Pierre Tillich. Thus, Persichetti only claims that the scheme achieves the security of one-time signatures. As our scheme is proven to achieve the (strong) EUF-CMA security, thus our scheme does not suffer from the statistical attack.

Here we go into the details to explain the reason that our scheme does not suffer from the statistical attack. Recall that the main difference in parameter setting between Persichetti's proposal and our scheme is the weight of the signature vector $\mathbf{s}$ is below the GV bound. TThus the weight of $\mathbf{e}$ is also below the GV bound. Suppose a signature $\mathbf{z} = \mathbf{sc} + \mathbf{e}$. The statistical attack works as follows. Suppose the weight of $\mathbf{c}$ is w. Then the challenge polynomial can be written as $c(x) = \sum_{i \in \mathrm{Supp}(\mathbf{c})} x^i$. For $j \in \mathrm{Supp}(\mathbf{c})$, define

$$\mathbf{z}^j := x^{-j}\mathbf{z} = \mathbf{s} + \sum_{i \in \mathrm{Supp}(\mathbf{c}), i \neq j} \mathbf{s}x^{i-j} + \mathbf{e}x^{-j}.$$

Then $\mathbf{s}x^{i-j}$ and $\mathbf{e}x^{i-j}$ are cyclic rotations of $\mathbf{s}$ and $\mathbf{e}$, respectively. As the weight of $\mathbf{s}$ and $\mathbf{e}$ are both below the GV bound, the supports of $\mathbf{s}$ and $\mathbf{e}x^{-j}$, $\mathbf{s}x^{i-j}$ have no intersection with non-negligible probability. If the support of $\mathbf{s}$ exactly has no overlap with supports of all the $\mathbf{e}x^{-j}, \mathbf{s}x^{i-j}$, the support of $\mathbf{z}^j$ can be viewed as the union of the supports of $\mathbf{s}$ and $\mathbf{e}x^{-j}, \mathbf{s}x^{i-j}$.

Given many valid signatures, an adversary is able to perform the intersection operation on the supports of shifted signatures to recover the secret key with non-negligible probability. Persichetti's paper explains that such an attack is feasible because the weight of the vector $\mathbf{e}$ is too small to appropriately hide the secret key in the signature. In our scheme, the vector $\mathbf{z}$ and the vector $\mathbf{e}$ are both not sparse. For instance, in the given parameters in Section 6, the weight of $\mathbf{z}$ and $\mathbf{e}$ are both beyond a faction of 0.22. Because the secret key $\mathbf{s}$ and the vector $\mathbf{e}$ are independently sampled, the supports of $\mathbf{s}$ and $\mathbf{e}$ have no intersection with negligible probability. Putting it in another words, given a shifted signature vector, some positions of the secret key are erased by distinct $\mathbf{e}x^{-j}$ and an adversary is unable to tell apart that a set position is from the secret key or from $\mathbf{e}x^{-j}$. Taking the intersection of the supports of honestly generated signatures does not reveal the support of the secret key. Note that the weight of $\mathbf{e}$ is far beyond the GV bound has the significant effect to prevent against the statistical attacks. Namely, the vector $\mathbf{e}$ properly hides the secret key in the signature.

From another point of view, in our scheme the output signature distribution is adjusted identical to a truncated binomial distribution, which is characterized by the parameter (n, τ, ξ), and independent from the secret key. Collecting a polynomial number of signatures is equivalent to independently sample from truncated binomial distribution, which does not leak any information on the secret key and makes any statistical attack infeasible to our scheme.

On the other hand, even viewed as one-time signature scheme, Persichetti's proposal is still insecure. There are two key recovery attacks to the one-time signature scheme. Next we go into the details of the two key recovery attacks.

7.2.1 LDPC Decoding Attack to the One-time Signature Scheme

One attack is based on the decoding capability of LDPC codes. The LDPC decoding attack was proposed by Deneuville and Gaborit [DG20]. As Persichetti's proposal only considered the case for rate 1/2, we also describe the attack for code rate 1/2. It is easy to extend the attack to the case for code rate $1/d$. Specially, given a signature $(\mathbf{z}, \mathbf{c})$ of the quasi-cyclic code structure, we have $\mathbf{z} = \mathbf{sc} + \mathbf{e}$. It is $\mathbf{z} = (\mathbf{c}, \mathbf{1}) \begin{pmatrix} \mathbf{s} \\ \mathbf{e} \end{pmatrix}$. In particular, it is $\mathbf{z}_i = \mathbf{s}_i\mathbf{c} + \mathbf{e}_i$ for $i = 1, 2$. Given $(\mathbf{z}_i, \mathbf{c})$, the problem can be viewed as syndrome decoding problem of LDPC code with parity check matrix $(\mathbf{c}, \mathbf{1})$ as the weight of $\mathbf{c}$ is small. Because the weight of $(\mathbf{e}_i, \mathbf{v}_i)$ is below the GV bound of the parity checked code by the public key, an adversary is able to recover the secret key $\mathbf{e}_i$ via running the LDPC decoding algorithm for the code $(\mathbf{c}, \mathbf{1})$. In our scheme, as $\mathbf{e}$ is of large weight and beyond the decoding capability of the LDPC decoding algorithm, thus it will not suffer from such a LDPC decoding attack. For instance in our scheme the weight of $\mathbf{c}$ is at least 6, and the weight of $\mathbf{e}$ is at least $0.22n$. As $6 \times 0.22 \times 2 > 1$, it is far beyond the decoding ability of an LDPC code.

In our scheme, the challenge vector still keeps sparse. We only require the number of possible challenge vectors achieves the desired security level. The sparse challenge vector in our scheme does not cause any information leakage on the secret key. Recall the main difference in parameter setting between our scheme and Persichetti's proposal is that the weight of $\mathbf{e}$ is not required to be below the GV bound. We set the weight of $\mathbf{e}$ to be as large as possible. Thus the vector $\mathbf{e}$ can properly hide the secret key. An adversary is unable to get any information on the secret key from the signature and the challenge vector. Additionally, in our scheme, adapting larger weight signature will reduce the rejection rate as well.

In our scheme, if an adversary collects polynomial number of signatures for instance ℓ signatures, then the adversary has $\{(\mathbf{z}_j, \mathbf{c}_j)\}_{j \in \{1 \ldots \ell\}}$ and each signature satisfies $\mathbf{e}_{ji}\mathbf{c}_j + \mathbf{e}_{ji} = \mathbf{z}_{ji}$ for $j \in \{1 \ldots \ell\}$ and $i \in \{1, 2\}$. Writing the signature together, the adversary has

$$\mathbf{s}_i(\mathbf{c}_1, \ldots, \mathbf{c}_\ell) + (\mathbf{e}_{1i}, \ldots, \mathbf{e}_{\ell i}) = (\mathbf{z}_{1i} \ldots \mathbf{z}_{\ell i}).$$

The weight of $(\mathbf{e}_{1i}, \ldots, \mathbf{e}_{\ell i})$ is still out of the decoding capability of the quasi-cyclic code with $(\mathbf{c}_1, \ldots, \mathbf{c}_\ell)$ being the parity check matrix. Thus, the LDPC decoding does not work for our scheme. Because $\mathbf{s}_1$ is of small weight, one might consider to enumerate all possible $\mathbf{s}_i$ and limit the one that the corresponding $(\mathbf{e}_{1i}, \ldots, \mathbf{e}_{\ell i})$ is of small weight. But the number of possible $\mathbf{s}_i$ is beyond the security level of the scheme. Thus, such an attack is still infeasible.

On the other hand, the signature distribution in our scheme is proven to independent from the secret key. Thus, an adversary is unable to learn any information on the secret key from a polynomial number of signatures.

7.2.2 Statistical Attack to One-time Signature Scheme

Santini, Baldi and Chiaraluce [SBC19] proposed a statistical attack to the one-time signature scheme to recover the secret key. The attack again uses the property that $c(x) = \sum_{i \in \mathrm{Supp}(\mathbf{c})} x^i$. Then

$$\mathbf{z}^j := \mathbf{z}x^{-j} = \mathbf{s} + \sum_{i \in \mathrm{Supp}(\mathbf{c}), i \neq j} \mathbf{s}x^{i-j} + \mathbf{e}x^{-j}.$$

Recall that if there is no intersection between of the supports of $\mathbf{s}$ and $\mathbf{s}x^{i-j}, \mathbf{e}x^{-j}$, the support of $\mathbf{z}^j$ can be viewed as the union of the support of $\mathbf{s}$ and $\mathbf{s}x^{i-j}, \mathbf{e}x^{-j}$. In the attack of [SBC19], $\mathbf{z}^j$ is lifted to a polynomial defined over $\mathbb{Z}[x]$ instead of $\mathbb{F}_2[x]$. The attack computes the sum of all the lifted polynomials. Because $\mathbf{s}$ and $\mathbf{e}$ are both sparse, then the support of $\mathbf{s}$ and $\mathbf{e}$ have no intersection with non-negligible probability. For different j, $\mathbf{e}x^{-j}$ and $\mathbf{s}x^{i-j}$ can be viewed as a random shift of $\mathbf{e}$ and $\mathbf{s}$. Summing up all the lifted polynomials, the support of $\mathbf{s}$ will be accumulated whereas the supports $\mathbf{e}x^{-j}$ and $\mathbf{s}x^{i-j}$ for different j will not be accumulated with non-negligible probability. Thus lifting the polynomial $\mathbf{z}^j$ to the ring $\mathbb{Z}[x]$ and the computing the sum of all the lifted polynomials will lead that every not erased position of $\mathbf{s}$ achieves the peak value in the sum of the lifted polynomials. Then the adversary can use the sum of lifted polynomials to infer the the support of the secret key. Given one signature, Santini et al. demonstrated that the attack can directly recover the secret key with non-negligible probability. If the attack is unable to fully recover the secret key, a large portion of the secret key is recovered and the remaining support of the secret key can be recovered by running the information set decoding(ISD) algorithm. As the secret key $\mathbf{s}$ and $\mathbf{e}$ are both sparse, the number of positions of secret key is erased in $\mathbf{z}^j$ is very small. Thus, the sum of lifted polynomials is able to track most positions of the support of the secret key $\mathbf{s}$. Assume the polynomial recovered from the sum of the lifted polynomials of $\mathbf{z}^j$ is $\mathbf{e}'$. Then the weight of $\mathbf{e} - \mathbf{e}'$ is extremely small. After subtracting the syndrome of $\mathbf{e}'$ from public key, the target vector to be recovered by the ISD algorithm is $\mathbf{e} - \mathbf{e}'$. So the resulting complexity of ISD after subtracting recovered $\mathbf{e}'$ from public key is far less than directly running the ISD algorithm on public key to recover the secret key.

This is not the case for our scheme. In our scheme, $\mathbf{e}$ is not sparse, and it is independently sampled from $\mathbf{s}$. As a matter of fact, one can easily compute that the supports of $\mathbf{s}$ and $\mathbf{e}$ have no intersection with negligible probability and the supports of $\mathbf{e}x^{i-j}$ and $\mathbf{e}x^{i-\ell}$ for distinct j and ℓ have no intersection with negligible probability. Thus, a subset of the support of secret key in $\mathbf{z}^{-j}$ is erased by $\mathbf{e}x^{-j}$. Besides, a subset of the support of $\mathbf{e}x^{-j}$ for distinct j is accumulated in the sum of lifted polynomials as $\mathbf{e}$ is not sparse. Thus, computing the sum of lifted polynomials in our scheme will get a mixing union of supports of $\mathbf{s}$ and $\mathbf{e}x^{-j}$. An adversary is unable to distinguish one element of the support is from $\mathbf{s}$ or from $\mathbf{e}$. The statistical attack provides no advantage for extracting the support of secret key from the signature. In addition, a computation of the successful probability of the above statistical attack demonstrates the probability that it can recover the secret key for our scheme is far beyond the claimed security level.

On the other hand, as pointed out in Section 7.2 and 7.2.1, in our scheme, the signature distribution is tuned to a truncated binomial distribution. Thus a signature produced from our scheme is equivalent to sample from the truncated binomial distribution. Hence, an adversary is unable to learn any information on the secret key for our scheme.

7.3 Summary of the Attacks

From the above description, given one signature produced by our scheme, our scheme is able to withstand existing LDPC decoding attack and the lifted statistical attacks. If an adversary attempts to collect a polynomial number signatures produced by our scheme to attack the scheme, the adversary is able to solve the codeword finding problem or the DOOM problem according to the security proof. In particular, the output signature distribution is tuned to a truncated binomial distribution independent from the secret key part. Thus, an adversary is unable to learn any information on the secret key from polynomial number of signatures.

Both the LDPC decoding attack and the two statistical attacks owe the feasibility of the attacks to the sparsity of the challenge vector $\mathbf{c}$. In the above considerations, we can see that the sparsity of $\mathbf{c}$ is not the essential reason. That the vector $\mathbf{e}$ is sparse is the inherent reason for Persichetti's proposal suffering from various attacks. On the other hand, to keep the signature vector of small weight, the challenge vector has to be sparse in some extent.

Employing larger weight of $\mathbf{e}$ has significant impact in our scheme. First it can securely hide the secret key to resist the statistical attack. Second a LDPC decoding attack is impossible as larger weight of $\mathbf{e}$ makes it beyond the decoding capability of the LDPC code. Third, it can be used together with the rejection sampling lemma to achieve a truncated binomial distribution decoupled from the secret vector.

Remark 7.2. *Even if one does not care huge code length and huge reject rate, applying our rejection sampling lemma to Persichetti's proposal still leads to an insecure scheme. The reason is that for given message, commitment, and challenge vector, requiring the weight of signature vector being below the GV bound uniquely determines the signature vector with overwhelming probability and there is no distribution related to the signature vector. On the other hand, once one signature is output, the above two key recovery attacks can efficiently recover the secret key from the signature.*

8 Conclusions and Future Work

We construct a signature scheme based on the Schnorr-Lyubashevsky framework for random linear Hamming metric code. Employing variants of binomial distributions, we prove an efficient rejection sampling lemma for binary linear codes. Via the rejection sampling strategy, the output signature distribution is adjusted to be independent from the secret key. Thus, an adversary is unable to learn any information on the secret key from polynomial number of signatures. The security of our scheme is reduced on full-fledged hard coding problems. We instantiate the signature scheme based on hard quasi-cyclic coding problems or Ring-LPN instances.

The existing rank-metric alternate of Schnorr-Lyubashevsky framework with the name Durandal [ABG+19] reduces the security of the scheme to a complicated problem Advanced Product Spaces Subspaces Indistinguishability PSSI$^+$, which is a new problem proposed in the Durandal paper. It is not proven that the Durandal signature does not leak information on the secret key. On the other hand, the Wave [DST19] signature scheme follows the hash-and-sign framework. The security of Wave is reduced to the indistinguishability of average generalized $(U, U+V)$ code, which is a new problem proposed in an earlier version of the Wave scheme. We believe that the security of our scheme is reduced to mature hard coding problems provides strong confidence to the security of our signature scheme.

The major drawback of our signature scheme is the large public key size and the large signature size compared to lattice-based alternates submitted to NIST Round 3 Dilithium [DKL+19] and Falcon [FHK+18]. We point out here that the large public key size is related to condition of the rejection sampling lemma. To make the rejection rate in a practically acceptable range, we choose the binomial distribution to be the target distribution to adjust the output signature. We expect future work to improve the rejection sampling lemma proved in this paper. Concretely, the exponent 2 indicating the relation of the weight of shifted vector and the target vector is expected to improved to a smaller value.

An alternate option to reduce the code length is to use q-ary code rather than binary code for a large q for instance $q = 3, 4, 5 \ldots$ The complexity of information set decoding algorithm for q-ary code is $q^{cw+O(1)}$, where w is the weight of secret key, and c is a constant related to the code rate and q. To achieve λ-bit security, choosing larger q leads to smaller Hamming weight of the secret key. Even if one still chooses the code length n to be square of $\mathrm{wt}(\mathbf{s})\mathrm{wt}(\mathbf{c})$ to obtain an acceptable rejection rate, the $\mathrm{wt}(\mathbf{s})$ is reduced, hence the code length is reduced as well. So do the public key size and the signature size. Here is a small issue to notice. If one chooses a larger q, each element of $\mathbb{F}_q$ is of size $\log q$ rather than $\log 2 = 1$. Thus, one needs to find a tradeoff between q and the reduced code length.

For $q \geq 3$, one can additionally use the syndrome decoding problem of large weight as the underlying hard problem. As pointed out in the Wave paper, for binary linear code, the small weight range of and the large weight range of the hard syndrome decoding problem is symmetric. Whereas for $q \geq 3$, it looks like the large weight range of the syndrome decoding problem is harder than the small weight range of the syndrome decoding problem. Maybe one can choose the weight of secret key lying in the large range to reduce the code length. Even for binary linear code, allowing the weight of the signature both in the small weight range and large weight range is possible to reduce the rejection rate. For $q \geq 3$, maybe the rejection rate can be further reduced as there are several methods to convert a small weight vector to a large weight vector.

References

[AABN02] Michel Abdalla, Jee Hea An, Mihir Bellare, and Chanathip Namprempre. From identification to signatures via the fiat-shamir transform: Minimizing assumptions for security and forward-security. In *EUROCRYPT*, volume 2332, pages 418–433. Springer, 2002. 2

[ABB+18] Nicolas Aragon, Paulo Barreto, Slim Bettaieb, Loic Bidoux, Olivier Blazy, Jean-Christophe Deneuville, Phillipe Gaborit, Shay Gueron, Tim Guneysu, Carlos Aguilar Melchor, Rafael Misoczki, Edoardo Persichetti, Nicolas Sendrier, Jean-Pierre Tillich, and Gilles Zémor. BIKE: Bit Flipping Key Encapsulation, Oct 2018. https://bikesuite.org/files/BIKE.pdf. 1, 6, 11, 30

[ABG+19] Nicolas Aragon, Olivier Blazy, Philippe Gaborit, Adrien Hauteville, and Gilles Zémor. Durandal: A rank metric based signature scheme. In *EUROCRYPT*, pages 728–758, 2019. 2, 3, 35

[AIK07] Benny Applebaum, Yuval Ishai, and Eyal Kushilevitz. Cryptography with constant input locality. In Alfred Menezes, editor, *CRYPTO*, Lecture Notes in Computer Science, 2007. 11

[AR05] Dorit Aharonov and Oded Regev. Lattice problems in NP cap conp. *J. ACM*, 52(5):749–765, 2005. 20

[BCL+17] Daniel J. Bernstein, Tung Chou, Tanja Lange, Ingo von Maurich, Rafael Misoczki, Ruben Niederhagen, Edoardo Persichetti, Christiane Peters, Peter Schwabe, Nicolas Sendrier, Jakub Szefer, and Wen Wang. Classic McEliece: conservative code-based cryptography, Nov 2017. https://classic.mceliece.org/nist/mceliece-20171129.pdf. 1

[BHLP17] Daniel J. Bernstein, Andreas Hulsing, Tanja Lange, and Lorenz Panny. Comments on RaCoSS, a submission to NIST's PQC competition. 2017. Available at https://helaas.org/racoss/. 3, 31

[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 *EUROCRYPT*, volume 7237, pages 520–536. Springer, 2012. 12

[BMvT78] Elwyn R. Berlekamp, Robert J. McEliece, and Henk C. A. van Tilborg. On the inherent intractability of certain coding problems (corresp.). *IEEE Trans. Inf. Theory*, 24(3):384–386, 1978. 3, 11

[CFS01] Nicolas T. Courtois, Matthieu Finiasz, and Nicolas Sendrier. How to achieve a mceliece-based digital signature scheme. In *ASIACRYPT*, pages 157–174, 2001. 2, 3

[DDLL13] Léo Ducas, Alain Durmus, Tancrède Lepoint, and Vadim Lyubashevsky. Lattice signatures and bimodal gaussians. In *CRYPTO*, volume 8042, pages 40–56. Springer, 2013. 2, 5, 29

[DG20] Jean-Christophe Deneuville and Philippe Gaborit. Cryptanalysis of a code-based one-time signature. *Designs, Codes and Cryptography*, 88(9):1857–1866, Sep 2020. 3, 33

[DGK20] Nir Drucker, Shay Gueron, and Dusan Kostic. Fast polynomial inversion for post quantum QC-MDPC cryptography. In *Cyber Security Cryptography and Machine Learning - Fourth International Symposium, CSCML 2020, Be'er Sheva, Israel, July 2-3, 2020, Proceedings*, pages 110–127, 2020. 28

[DH76] Whitfield Diffie and Martin E. Hellman. New directions in cryptography. *IEEE Trans. Information Theory*, 22(6):644–654, 1976. 2

[DKL+19] Léo Ducas, Eike Kiltz, Tancrède Lepoint, Vadim Lyubashevsky, Peter Schwabe, Gregor Seiler, and Damien Stehlé. CRYSTALS–Dilithium: Algorithm Specification and Supporting Documentation. Round-2 submission to the NIST PQC project, 2019. 35

[DST19] Thomas Debris-Alazard, Nicolas Sendrier, and Jean-Pierre Tillich. Wave: A new family of trapdoor one-way preimage sampleable functions based on codes. In *ASIACRYPT*, pages 21–51, 2019. 2, 4, 5, 23, 35

[DT18] Thomas Debris-Alazard and Jean-Pierre Tillich. Two attacks on rank metric code-based schemes: Ranksign and an IBE scheme. In *ASIACRYPT*, pages 62–92, 2018. 3

[FGO+13] Jean-Charles Faugère, Valérie Gauthier-Umaña, Ayoub Otmani, Ludovic Perret, and Jean-Pierre Tillich. A distinguisher for high-rate McEliece cryptosystems. *IEEE Trans. Information Theory*, 59(10):6830–6844, 2013. 3

[FHK+18] Pierre-Alain Fouque, Jeffrey Hoffstein, Paul Kirchner, Vadim Lyubashevsky, Thomas Pornin, Thomas Prest, Thomas Ricosset, Gregor Seiler, William Whyte, and Zhenfei Zhang. Falcon: Fast-fourier lattice-based compact signatures over ntru. *Submission to the NIST's post-quantum cryptography standardization process*, 2018. 36

[FRX+17] Kazuhide Fukushima, Partha Sarathi Roy, Rui Xu, Shinsaku Kiyomoto, Kirill Morozov, and Tsuyoshi Takagi. RaCoSS: Random Code-based Signature Scheme. *Submission to NIST post-quantum standardization process*, 2017. Available at https://csrc.nist.gov/projects/post-quantum-cryptography/round-1-submissions. 2, 3, 31

[FS86] Amos Fiat and Adi Shamir. How to prove yourself: Practical solutions to identification and signature problems. In *CRYPTO '86*, volume 263, pages 186–194. Springer, 1986. 2

[GG07] Philippe Gaborit and Marc Girault. Lightweight code-based identification and signature. In *IEEE International Symposium on Information Theory, ISIT 2007, Nice, France, June 24-29, 2007*, pages 191–195, 2007. 10

[GGH97] Oded Goldreich, Shafi Goldwasser, and Shai Halevi. Public-key cryptosystems from lattice reduction problems. In *CRYPTO '97*, pages 112–131, 1997. 2

[GJL15] Qian Guo, Thomas Johansson, and Carl Löndahl. A new algorithm for solving ring-lpn with a reducible polynomial. *IEEE Trans. Inf. Theory*, 61(11):6204–6212, 2015. 12

[GMR88] Shafi Goldwasser, Silvio Micali, and Ronald L. Rivest. A digital signature scheme secure against adaptive chosen-message attacks. *SIAM J. Comput.*, 17(2):281–308, 1988. 12

[GPV08] Craig Gentry, Chris Peikert, and Vinod Vaikuntanathan. Trapdoors for hard lattices and new cryptographic constructions. In *STOC*, pages 197–206, 2008. 2

[GRSZ14] Philippe Gaborit, Olivier Ruatta, Julien Schrek, and Gilles Zémor. New results for rank-based cryptography. In *AFRICACRYPT*, pages 1–12, 2014. RankSign. 2, 3

[HPS01] Jeffrey Hoffstein, Jill Pipher, and Joseph H. Silverman. NSS: an NTRU lattice-based signature scheme. In *EUROCRYPT*, pages 211–228, 2001. 2

[JJ02] Thomas Johansson and Fredrik Jönsson. On the complexity of some cryptographic problems based on the general decoding problem. *IEEE Trans. Inf. Theory*, 48(10):2669–2678, 2002. 11, 23

[LJS$^+$16] Carl Löndahl, Thomas Johansson, Masoumeh Koochak Shooshtari, Mahmoud Ahmadian-Attari, and Mohammad Reza Aref. Squaring attacks on mceliece public-key cryptosystems using quasi-cyclic codes of even dimension. *Des. Codes Cryptogr.*, 80(2):359–377, 2016. 12

[LXY20] Zhe Li, Chaoping Xing, and Sze Ling Yeo. A proof of concept implementation of A New Code Based Signature Scheme without Trapdoors, 2020. `https://github.com/zhli271828/rand_code_sign`. 30

[Lyu12] Vadim Lyubashevsky. Lattice signatures without trapdoors. In *EUROCRYPT*, volume 7237, pages 738–755. Springer, 2012. 2, 5, 13

[MAB$^+$19] Carlos Aguilar Melchor, Nicolas Aragon, Slim Bettaieb, Loic Bidoux, Olivier Blazy, Jean-Christophe Deneuville, Philippe Gaborit, Edoardo Persichetti, and Gilles Zémor. Hamming Quasi-Cyclic (HQC), May 2019. `https://pqc-hqc.org/doc/hqc-specification_2019-04-10.pdf`. 1, 30

[MMT11] Alexander May, Alexander Meurer, and Enrico Thomae. Decoding random linear codes in $\tilde{O}(2^{0.054n})$. In *ASIACRYPT*, volume 7073, pages 107–124. Springer, 2011. 12

[MO15] Alexander May and Ilya Ozerov. On computing nearest neighbors with applications to decoding of binary linear codes. In *EUROCRYPT (1)*, volume 9056, pages 203–228. Springer, 2015. 12

[MS77] Florence Jessie MacWilliams and Neil James Alexander Sloane. *The theory of error correcting codes*, volume 16. Elsevier, 1977. 9

[MTSB13] Rafael Misoczki, Jean-Pierre Tillich, Nicolas Sendrier, and Paulo S. L. M. Barreto. Mdpc-mceliece: New McEliece variants from moderate density parity-check codes. In *ISIT*, pages 2069–2073. IEEE, 2013. 10

[MU05] Michael Mitzenmacher and Eli Upfal. *Probability and Computing: Randomized Algorithms and Probabilistic Analysis*. Cambridge University Press, 2005. 13, 14

[NIS20] Round 3 Submissions, July 2020. https://csrc.nist.gov/projects/post-quantum-cryptography/round-3-submissions. 1

[Per18] Edoardo Persichetti. Efficient one-time signatures from quasi-cyclic codes: A full treatment. *Cryptogr.*, 2(4):30, 2018. 2, 3, 32

[Pra62] Eugene Prange. The use of information sets in decoding cyclic codes. *IRE Trans. Inf. Theory*, 8(5):5–9, 1962. 12

[RMF$^+$18] Partha Sarathi Roy, Kirill Morozov, Kazuhide Fukushima, Shinsaku Kiyomoto, , and Tsuyoshi Takagi. Code-based signature scheme without trapdoors. 2018. RaCoSS-R. See also https://www.ieice.org/ken/paper/20180725L1FF/eng/. 2, 3, 31

[SBC19] Paolo Santini, Marco Baldi, and Franco Chiaraluce. Cryptanalysis of a one-time code-based digital signature scheme. In *IEEE International Symposium on Information Theory, ISIT 2019, Paris, France, July 7-12, 2019*, pages 2594–2598, 2019. 3, 34

[Sch91] Claus-Peter Schnorr. Efficient signature generation by smart cards. *J. Cryptol.*, 4(3):161–174, 1991. 2

[Sen11] Nicolas Sendrier. Decoding one out of many. In *Post-Quantum Cryptography - 4th International Workshop, PQCrypto 2011, Taipei, Taiwan, November 29 - December 2, 2011. Proceedings*, pages 51–67, 2011. 6, 12

[Tor17] Rodolfo Canto Torres. CaWoF, C library for computing asymptotic exponents of generic decoding work factors, Jan 2017. https://gforge.inria.fr/projects/cawof/. 28

[TS16] Rodolfo Canto Torres and Nicolas Sendrier. Analysis of information set decoding for a sub-linear error weight. In *PQCrypto*, volume 9606, pages 144–161. Springer, 2016. 12, 28

[Var97] Alexander Vardy. Algorithmic complexity in coding theory and the minimum distance problem. In *STOC*, pages 92–109, 1997. 3, 11

[vN51] John von Neumann. Various techniques used in connection with random digits. In A. S. Householder, G. E. Forsythe, and H. H. Germond, editors, *Monte Carlo Method*, volume 12 of *National Bureau of Standards Applied Mathematics Series*, chapter 13, pages 36–38. US Government Printing Office, Washington, DC, 1951. 5

[Xag18] Keita Xagawa. Practical Attack on RaCoSS-R. *IACR Cryptol. ePrint Arch.*, 2018:831, 2018. 3, 31

A Strong Security Proof

Here we provide a *strong* security proof for relaxed parameters. If we allow the parameter τ a little smaller, we can prove the security of the signature scheme upon the collision resistant property of syndrome decoding problem and thus the codeword finding problem. Via programming the random oracle, the proof employs the general forking lemma to find a small weight codeword.

Theorem A.1. *Assume that there is a forger that breaks the strong EUF-CMA game. Then there exists an algorithm that solves the codeword finding problem.*

In particular, if the forger $\mathcal{F}$ succeeds with probability δ, assuming the forger $\mathcal{F}$ makes at most s signature query and h hash query, there exists an algorithm $\mathcal{A}$ that solves the codeword *finding problem with probability at least* $\frac{\delta^2}{s+h} - 2^{-O(\lambda)}$*, where the parameter* τ *is chosen such that* $2n\tau + 2\xi + 2n\sigma w + \sqrt{\frac{\lambda n}{2\log \mathrm{e}}} < \frac{n-k}{2}$.

We prove that the forger $\mathcal{F}$ can be converted to an algorithm to find a small preimage for $\mathbf{0}$. In the proof, we simulate the random oracles for the forger.

Proof. Given the matrix $H \in \mathbb{F}_2^{(n-k)\times n}$, one samples a secret key S first and then publish $(H, T = HS)$ as the public key.

For a forger $\mathcal{F}$, it can forges a signature $(\mathbf{z}, \mathbf{c})$ for μ with probability δ. By the general forking lemma, the forger can forge a signature $(\mathbf{z}', \mathbf{c}')$ for μ with probability $\frac{\delta^2}{s+h} - 2^{-O(\lambda)}$ and the two forged signatures share the same message queried to the random oracle. It means $H\mathbf{z} - T\mathbf{c} = H\mathbf{z}' - T\mathbf{c}'$. Replacing T by the secret key, we obtain $H(\mathbf{z} - S\mathbf{c}) = H(\mathbf{z}' - S\mathbf{c}')$. If $\mathbf{z} - \mathbf{z}' - S\mathbf{c} + S\mathbf{c}'$ is nonzero and of small weight, we have a collision for the syndrome function and thus a small weight codeword party checked by the matrix H. First, we give a bound to the weight of $\mathbf{z} - \mathbf{z}' - S\mathbf{c} + S\mathbf{c}'$. From the distribution of S and the pilling-up lemma, the vector $S(\mathbf{c}' - \mathbf{c})$ admits the distribution $\mathcal{B}_\eta^n$, where $\eta = \frac{1}{2}(1 - (1-2\sigma)^{2w}) \approx 2\sigma w$ for very small σw. Then we have $\mathrm{wt}(\mathbf{z} - \mathbf{z}' - S\mathbf{c} + S\mathbf{c}') \leq 2n\tau + 2\xi + 2n\sigma w + \gamma < \frac{n-k}{2}$ but with negligible probability, where $\gamma = \sqrt{\frac{\lambda n}{2\log \mathrm{e}}}$. Next, we compute the probability that $\mathbf{z} - \mathbf{z}' - S\mathbf{c} + S\mathbf{c}'$ is nonzero. Given a vector $\mathbf{z}' - \mathbf{z}$ with weight t lying in $[0, 2n\tau + 2\xi]$, the probability that $S(\mathbf{c} - \mathbf{c}')$ equals to $\mathbf{z} - \mathbf{z}'$ is $\eta^t(1-\eta)^{n-t}$. From the condition of the rejection sampling lemma, we have $(n\eta)^2 < n$, namely, $\eta < \frac{1}{\sqrt{n}}$. For the probability, we have $\eta^t(1-\eta)^{n-t} \leq (1-\eta)^n \approx \mathrm{e}^{-\eta n} = \mathrm{e}^{-\Omega(\sqrt{n})}$, a negligible function of n. Hence, we proved that $\mathbf{z} - \mathbf{z}' - S\mathbf{c} + S\mathbf{c}'$ is a nonzero vector with weight less than $\frac{n-k}{2}$ except with negligible probability. □