Robust and Reusable Fuzzy Extractors and their Application to Authentication from Iris Data

Somnath Panja[1], Nikita Tripathi[1], Shaoquan Jiang[2], and Reihaneh Safavi-Naini[1]

[1] University of Calgary, Canada,
somn.math2007@gmail.com,{nikita.tripathi,rei}@ucalgary.ca
[2] University of Windsor, Canada, shaoquan.jiang@gmail.com

Abstract. Fuzzy extractors (FE) are cryptographic primitives that establish a shared secret between two parties who have similar samples of a random source, and can communicate over a public channel. An example for this is that Alice has a stored biometric at a server and wants to have authenticated communication using a new reading of her biometric on her device. Reusability and robustness of FE, respectively, guarantee that security holds when FE is used with multiple samples, and the communication channel is tamperable. Fuzzy extractors have been studied in information theoretic and computational setting.
Contributions of this paper are two-fold. First, we define a *strongly robust and reusable FE* that combines the strongest security requirements of FEs, and give three constructions. Construction 1 has computational security, and Constructions 2 and 3 provide information theoretic (IT) security, in our proposed model. Construction 1 *provides a solution to the open question of Canetti et al. (Eurocrypt 2014)*, by achieving robustness and reusability (post-quantum) security in standard model for their construction. Constructions 2 and 3 offer a new approach to the construction of IT-secure FE. Construction 3 is the first robust and reusable FE with IT-security without assuming random oracle. Our robust FEs use a new IT-secure MAC with security against key-shift attack which is of independent interest. Our constructions are for structured sources which for Construction 1, matches Canetti et al.'s source.
We then use our Construction 1 for biometric authentication using iris data. We use a widely used iris data set to find the system parameters of the construction for the data set, and implement it. We compare our implementation with an implementation of Canetti et al.'s reusable FE on the same data set, showing the cost of post-quantum security without using random oracle, and robustness in standard model.

Keywords: Fuzzy extractors, Reusable and robust fuzzy extractor, Post-quantum security, Biomertic authentication, Iris authentication

1 Introduction

Establishing a shared random key between two parties is a fundamental problem in cryptography. *Fuzzy extractors* [19], establish a shared key in a setting

where Alice and Bob have "close" samples of a random source, where closeness is with respect to some distance metric, and can communicate over a public channel. The random source must have some guaranteed entropy (min-entropy), capturing unpredictability of the samples. Such correlated information source samples naturally occur in sampling of biometric and behavioral data [14,27,25], readings of a physical randomness source such as sunspots [11] and physically unclonable functions (PUFs) [34,29], or through quantum communication and post-processing in quantum key distribution [4]. FEs are particularly attractive because they allow a secret shared key to be established *non-interactively* using a single message from Alice to Bob over a public channel, where security can be information theoretic or computational. An FE consists of a pair of algorithms, (Gen, Rep) and works as follows: Gen takes Alice's sample w and generates a pair (R, P) where R is the secret key and P is some data that will be sent to Bob, who uses the Rep algorithm on their sample w' that is "close" to w, and P, and reproduces the secret key R. *Reliability and security* of fuzzy extractors require the same key to be derived by the two parties with a probability at least $1 - \epsilon$, and the key R to be σ indistinguishable from a uniformly random string of the same length, respectively.

Reusability and Robustness. FE in practice needs two important properties.

Reusability of fuzzy extractors [7] considers security when the same source is used multiple times. For example biometric scans of an individual is used for enrolment with multiple organizations. Reusability security is defined using a game between a challenger and an adversary Eve, where Eve specifies source samples $w^1, w^2, \cdots w^\eta$ in relation to w, and will be given the output of the Gen algorithm on them. Variations of this definition can be grouped with respect to the allowed type of queried samples by Eve (R1), and the output returned to Eve (R2), each further refined as follows.

R1.1: Eve chooses a shift that will be applied to w to obtain w^i [7][3];

R1.2: multiple readings of the source have arbitrary correlation with w [20]. In both cases w^i must be in "close" distance of w.

R2.1: Eve only sees P^i which is the public output of the Gen on w^i;

R2.2: Eve will be given the full output of Gen on w^i, that is (P^i, R^i).

R2.1 (resp. R2.2) is referred to as *outsider* (resp. *insider*) security [7].

Robustness requires that any modification of P be detectable with a high probability. The two flavours of this requirement are the following.

S1: Eve has access to P only [8];

S2: Eve sees (P, R) before generating P' $(\neq P)$. This is also referred to as *post-application* robustness [17].

In the following we use the above labelling to differentiate FE constructions. We note that $R1.2 \geq R1.2$, $R2.2 \geq R2.1$, and $S2 \geq S1$, where "$\geq$" refers to "more demanding security".

Robust and Reusable FE provides robustness when Eve sees the Gen output on multiple samples. The two types S1 and S2 of Eve's access can be straightforwardly extended to all samples.

[3] The definition allows any distortion function while all constructions are for shift function.

Fuzzy Extractor Constructions. An established approach to constructing FE is to use a *secure sketch*, that is output of the *Gen* algorithm, to enable Bob to recover w from w' without loosing "too much" entropy. The two parties then extract a common key R from their shared w. This approach is referred to as *sketch-and-extract* paradigm. Canetti et al. [20] proposed a new approach, *Sample-then-Lock*, and used it to construct *computational reusable FE* for a class of *structured sources* that are called *sources with high-entropy samples*. This construction and Alamélou et al.'s construction [2] are the only FE constructions that satisfy (R1.2, R2.2) type of reusability (strongest). These are however based on *digital lockers*, that have instantiations only based on hash functions when modeled as random oracle, or using non-standard strong vector DDH assumption [36]. These constructions are reusable only. Canetti et al. [20] note that it is easy achieve robustness for the construction in random oracle model [Theorem 1, [8]], but "Achieving reusability and robustness in the standard model is an open question." There are some constructions of robust and reusable FE such as due to Wen et al. [36] and Wen et al. [37], and provide (R1.1, R2.2, S2) type security.
Biometric Authentication and FEs. Biometrics have been widely used for authentication on mobile devices, where the template is in plaintext and stored and processed on a hardware security module. In remote client and server authentication extreme precaution must be used to ensure security of biometric data. FEs can be used for non-interactive remote authentication with provable cryptographic security for biometric data [33]. Provable security is particularly important because biometric data cannot be updated, and a system comprise will have life long effect. Also because of the uniqueness of biometrics, one compromise would affect all systemsthat use different readings of the same biometric data. Simhardi et al. [33] used a widely used iris data set to show that iris data can be modelled as the structured source in [10], and implemented a slightly modified version of Canetti et al.'s construction for non-interactive authentication system with provable security for iris data.

1.1 Our work

Out contributions are in two directions.

First, we construct (i) a computationally secure robust and reusable fuzzy extractor (Construction 1), and (ii) an IT-secure reusable FE (Construction 2), which is later extended to provide robustness (Construction 3). All constructions satisfy the strongest notions of reusability (R1.2, R2.2), and in the case of Construction 1 and 3, robustness (S2).

The Construction 1 is (post-quantum) secure and satisfies (R1.2, R2.2, S.2) in standard model and for the same distributions as in [10], and answers the open question raised by Canetti et al. [10]. Construction 2 provides a novel approach to the construction of IT-secure extractors for a new class of structured sources. Robustness in Constructions 1 and 3 both rely on a new IT-secure MAC algorithm that is secure against key-shift attack and is of independent interest.

Second, we implement the Construction 1 for an iris based authentication system that provides provable cryptographic security for iris data. We evaluate

performance of the system compared to the construction in [33] which implements a slightly modified version of Canetti et al.'s reusable FE. Our results show that our *Gen* and *Rep* algorithms are roughly 4 and 10 times slower (see section 4.2 and Figure 4(b)), respectively, showing the cost of providing security and robustness in standard model. Interestingly the main cost of *Gen* is to remove the random oracle assumption by replacing the HMAC in that was used for the realization of digital lockers, with the LPN-based encryption in our construction. MAC computation requires computing a polynomial over a finite field that can be significantly optimized but since in its current form, it is only a small fraction of running time of *Gen*, we leave such optimization for future. To make our results comparable with Simhardi et al. [33], we closely follow their experimental setting. We use the same widely used iris data set ND-IRIS-0405 [32,6] which is a superset of the NIST Iris Challenge Evaluation Dataset [31], and is used in Sinhadri et al. The data set was obtained through our University License agreement with the University of Notre Dame, with researchers' access permissions granted by the Ethics Board of our University. More details below.

Strongly robust and reusable computational fuzzy extractor Our starting point is the sample-then-lock approach of Canetti et al. for *α-entropy m-sample* sources. In these sources, for an n-bit sample w, random subsamples of size m have min-entropy at least α. This models entropy sources with *low rate* entropy samples that have *additional structure* on subsamples, and model biometric data such as iris. The construction outline and its intuitive security argument are given in Section 3.1. The main building blocks of the construction are (1) a symmetric key encryption algorithm, proposed by Dodis et al. [16], with security under auxiliary-input, assuming hardness of LPN (or the standard Learning Subspace with Noise (LSN) assumption), and (2) a new one-time IT-secure MAC with key-shift security. The encryption system has also an important property called *unique-key-property*, that is instrumental in proving security of our construction, and ensures that with high probability, there is a single key that can decrypt a correctly generated ciphertext (Claim 4.1. [16]).

Achieving robustness is by using a new key-shift secure MAC that ensures Eve cannot exploit linearity of the encryption system to tamper with the MAC key and the tag simultaneously.

Our construction is in Common Reference String (CRS) model. This model is also used in other known constructions of robust and reusable FE [36].

Theorem 1 proves security properties of the FE. The proof of the MAC construction and proof of the theorem are in Appendix A and Appendix B respectively.

Comparison of the construction with existing works is in Table 1.

Information Theoretic FE We use for the first time, *sample-then-lock* for IT-secure FE, for a class of structured sources, called *α-entropy conditional m-sample* sources. These sources require that the min-entropy of a random sample of length m, conditioned on the knowledge of a random sample of length ℓ, is above a threshold α. See Definition 6. Our main observation is that in information theoretic setting, one can use strong extractors to extract uniform

randomness from subsamples of sufficient min-entropy, and perfectly encrypt a randomly chosen key R using One-Time-Pad (OTP). The security proof of FE shows security of composition of multiple encryptions of the same message (R), using the extractor's output on different (correlated) subsamples (due to the large enough min-entropy from the source) using different random seeds.

Construction 3 extends Construction 2 to obtain the first strongly robust and reusable fuzzy extractor, and achieving security without random oracle. Robustness is achieved by appending a tag that is constructed using the IT-secure one-time MAC that was used in Construction 3.

Security theorems of two constructions are Theorem 2 and Theorem 3, respectively, and their comparison with existing works is detailed in Table 1.

(α, m, ℓ)-conditional sources have a more entropy requirement and it is unclear if they would be appropriate model for biometric data. One example of such sources can be constructed by sending a random string that is encoded as polarized photons in two orthogonal basis over a quantum channel, similar to the first step of BB84 protocol. The established correlated samples of Alice and Bob after reconciliation can be modelled as a (α, m, ℓ)-conditional source. Using our information theoretic sample-then-lock FE will give a new approach to key establishment in this setting.

Comparison. Comparison of FE is multifaceted. We introduced the main variations of reusability and robustness, and lablled each. FEs in this paper as well as well as Canetti et al. are for structured sources. Other FEs are have strict requirement on min-entropy of samples. Table 1 compares our fuzzy extractor constructions with other existing fuzzy extractors, showing (i) Construction 2 is the first arbitrary correlation reusable IT-secure FE, (ii) Construction 3 is the first strongly robust and reusable IT-secure FE without using random oracle, and (iii) Construction 1 is the first computationally secure strongly robust and reusable fuzzy extractor for arbitrary correlation samples in standard model.

Implementation. We implement Construction 1 in Python using iris data. Our implementation is open-source, and the url of the source code is given in section 5. This is the first iris based key derivation system that provides security against active adversaries in the standard model. Similar to Sinhadri et al. [33], our implementation provides security level of 32 bits that is available in iris data samples. Although this security level is not sufficient for a stand-alone system, this work can be incorporated into multi-factor authentication system to strengthen security of the system. In our system, error correction and key derivation are performed simultaneously, and so another noiseless factor such as password can be prepended to each subsample (of the sample w) to combine security of the systems, and achieve stronger security.

1.2 Related work

Fuzzy extractors has been widely studied in information theoretic [17,7,9,23] and computational settings [20,38,3,36]. Table 1 summarizes properties of all references that are directly related to our work. We include a more extensive review of more distant work in the full version of the paper.

FE Scheme	Security	Reusability	Correlation	Robuustness	Model
DRS04[19]	IT. Sec.	–	–	–	Std.
Boyen04[7]	IT. Sec.	R2.1	Shift (R1.1)	–	Std.
Boyen04[7]	IT. Sec.	R2.2	Shift (R1.1)	–	RO
BDKOS05[8]	IT. Sec.	–	–	S1	RO
DKK12[17],CDF08[13]	IT. Sec.	–	–	S2	Std.
FMR13[22]	Comp. Sec.	–	–	–	Std
CFP16[10],ABC18 [2]	Comp. Sec.	R2.2	Arbitrary (R1.2)	–	Dig. Loc.
ACEK17[3]	Comp. Sec.	R2.2	Shift (R1.1)	–	Std.
WLH18[38]	Comp. Sec.	R2.2	Shift* (R1.1)	–	Std.
WL18[35]	Comp. Sec.	R2.2	Shift (R1.1)	–	Std.
WL[36],WLG[37]	Comp. Sec.	R2.2	Shift (R1.1)	S2	Std.
Construction 2	IT. Sec.	R2.2	Arbitrary (R1.2)	–	Std.
Construction 3	IT. Sec.	R2.2	Arbitrary (R1.2)	S2	Std.
Construction 1	Comp. Sec.	R2.2	Arbitrary (R1.2)	S2	Std.

Table 1. IT. Sec. and Comp. Sec. denote information-theoretic and computational security, respectively. Columns are as follows: Security: whether the scheme achieves IT.Sec. or Comp. Sec.; Reusability : the scheme is reusable and the type of reusability; Correlation : type of correlation in reusability definition; Robustness :the scheme is robust and its type; Model - security proof model. Std.: standard model, RO: Random Oracle; Dig. Loc.: requires Digital Locker. Shift* is a stronger definition than an adversary just specify a shift δ^j. It also requires that each marginal distributions W^i has high entropy conditioned on the shift between each pair of distributions [10].

We note that a second implementation of iris-based authentication that has appeared concurrently with Sinhadri et al. is due to Cheon et al. [12]. In Section 4, we discuss their construction and our reasoning for choosing Sinhadri el al. for direct comparison.

2 Background and Definitions

Notations. We denote random variables (RVs) with upper-case letters (e.g., X), and their realizations with lower-case letters (e.g., x). The probability distribution of an RV X is denoted by P_X. The *min-entropy* $H_\infty(X)$ of RV $X \in \mathcal{X}$ with distribution P_X is $H_\infty(X) = -\log(\max_x(\mathrm{P}_X(x)))$. The *average conditional min-entropy* [19] is defined as, $\tilde{H}_\infty(X|Y) = -\log \mathbb{E}_{y \leftarrow \mathcal{Y}} \max_{x \in \mathcal{X}} \mathrm{P}_{X|Y}(x|y)$. The statistical distance between two random variables X and Y with the same domain $\mathcal{T}$ is given by $\Delta(X,Y) = \frac{1}{2}\sum_{v \in \mathcal{T}} |\Pr[X = v] - \Pr[Y = v]|$. For a (class of) distinguisher D, we write the computational distance between two random variables X and Y with respect to D as $\delta^{\mathsf{D}}(X,Y) = |\mathbb{E}[\mathsf{D}(X)] - \mathbb{E}[\mathsf{D}(Y)]|$, where $\mathbb{E}(\cdot)$ is the expectation. $\mathcal{D}_s$ denotes the class of randomised circuits which output a single bit and have size at most s. We write $a \approx b$ to represent that a and b are computationally indistinguishable. $(x)_{i \cdots j}$ denotes the block from ith bit to jth bit in x. For a sequence $W = W_1, \cdots, W_n$ and an index set $A = \{i_1, \cdots, i_t\}$, $W[A]$ denotes the subsequence $W_{i_1}, \cdots, W_{i_t}$. For vector $m = (m_0, \cdots, m_{n-1})$, $m(x)$ is the polynomial $\sum_{i=0}^{n-1} m_i x^i$.

We use two classes of functions, $SMALL$ and $NEGL$, to define closeness of families of distributions that are indexed by λ, for statistical and computational indistinguishability, respectively [30].

Extractors and hash functions. Extractor are used to obtain high quality randomness from distributions (also called sources) with sufficient entropy.

Definition 1 (Strong (average case) randomness extractor). *$E : \{0,1\}^n \times \{0,1\}^r \to \{0,1\}^\lambda$ is an average $(n, \alpha, \lambda, \epsilon)$-extractor if for any pair of random variables X, A with X a n-bit string and $\tilde{H}_\infty(X|A) \geq \alpha$, it holds that*

$$\Delta(E(X,R), A, R; U, A, R) \leq \epsilon, \tag{1}$$

where R is a purely random r-bit string and U is uniformly random over $\{0,1\}^\lambda$. An extractor E is linear if $E(X_1 + X_2, R) = E(X_1, R) + E(X_2, R)$, for any $X_1, X_2 \in \{0,1\}^n$ and $R \in \{0,1\}^r$.

A in the definition is the side information. If this variable is removed, we have the definition of $(n, \alpha, \lambda, \epsilon)$-*extractor*. By replacing statistical distance with computational distance, we obtain a $(n, \alpha, \lambda, \epsilon)$-*computational extractor*.

Universal hash function families have been wide used as randomness extractor. The extracted randomness is given by Leftover Hash Lemma [24].

A family of functions $h : \mathcal{X} \times \mathcal{S} \to \mathcal{Y}$ is called a *universal hash family* if $\forall x, y \in \mathcal{X}$, $x \neq y : \Pr[h(x,s) = h(y,s)] \leq \frac{1}{|\mathcal{Y}|}$, where the probability is over the uniform choices of s from $\mathcal{S}$.

A family of hash functions $h : \mathcal{X} \times \mathcal{S} \to \mathcal{Y}$ is called a *strong universal hash family* if for all $x, y \in \mathcal{X}$, $x \neq y$, and any $a, b \in \mathcal{Y}$, $Pr[h(x,s) = a \wedge h(y,s) = b] = \frac{1}{|\mathcal{Y}|^2}$, where the probability is over the uniform choices of s from $\mathcal{S}$.

Lemma 1 (Generalized Leftover Hash Lemma [18]). *Let $h : \mathcal{X} \times \mathcal{S} \to \{0,1\}^\ell$ be a universal hash family. Then for any two variables $A \in \mathcal{X}$ and $B \in \mathcal{Y}$, applying h on A can extract a uniform random variable whose length ℓ satisfies the following $\Delta(h(A,S), S, B; U_\ell, S, B) \leq \frac{1}{2}\sqrt{2^{-\tilde{H}_\infty(A|B)} \cdot 2^\ell}$, where S is chosen uniformly from $\mathcal{S}$. In particular, if h is a universal hash family and $\ell \leq \alpha + 2 - 2\log\frac{1}{\epsilon}$, then h is an (n, α, l, ϵ)-extractor, where A is an n-bit string.*

Computational assumption. Our computational fuzzy extractor construction in Section 3.1 uses the hardness of *Learning Parity with Noise (LPN)* that is equivalent to the problem of decoding random linear codes, and for which no efficient quantum algorithm is known. The assumption is stated as follows.

LPN Assumption. For any polynomial $t = poly(n)$ and for any constant $\gamma > 0$, we have $\{A, Ax + e\}_{n\in\mathbb{N}} \approx \{A, U_t\}_{n\in\mathbb{N}}$, where $A \in_R \{0,1\}^{t\times n}, x \in_R \{0,1\}^n, U_t \in_R \{0,1\}^t$ are all uniformly distributed, and $e = (e_1, \cdots, e_t) \in \{0,1\}^t$ is distributed as: $e_i = 0$ with probability γ and $e_i \in_R \{0,1\}$ with probability $(1-\gamma)$.

The LPN assumption has the following very appealing properties: (i) *Efficiency.* The computation of LPN function $Ax + e$ is extremely efficient. (ii) *Closure under composition.* The LPN function is closed under composition i.e. for any polynomial ℓ we have $\{(A^{(i)}, A^{(i)}x + e^{(i)})_{i=1}^\ell\}_{n\in\mathbb{N}} \approx \{(A^{(i)}, U^{(i)})_{i=1}^\ell\}_{n\in\mathbb{N}}$. (iii) *Robustness against bit-leakage.* LPN assumption still holds even if some k bits of x are leaked; however the security parameter decreases from n to $n-k$. In addition, it is also robust against any linear leakage function, but the security parameter decreases from n to $n-k$ if some k bits are leaked.

2.1 Fuzzy extractors

The definition of information theoretic fuzzy extractors is from Dodis et al. [18].

Definition 2 (Fuzzy extractor). *Let $\mathcal{W}$ be a family of probability distributions over $\mathcal{M}$. An $(\mathcal{M}, \mathcal{W}, \xi, t', \epsilon', \sigma)$-fuzzy extractor FE is a pair of randomized procedures "generate" (Gen) and "reproduce" (Rep) that satisfy the followings:*

i. The generate procedure *$Gen : \mathcal{M} \rightarrow \{0,1\}^{\xi} \times \mathcal{P}$, that takes a source sample $w \in \mathcal{M}$, and outputs a key $r \in \{0,1\}^{\xi}$ and a public helper string $p \in \mathcal{P}$.*

ii. The reproduction procedure *$Rep : \mathcal{M} \times \mathcal{P} \rightarrow \{0,1\}^{\xi}$ that takes an element $w' \in \mathcal{M}$ and the helper string $p \in \mathcal{P}$ as input and outputs a key r.*

iii. The ϵ'-correctness *of a fuzzy extractor states that if $dist(w, w') \leq t'$ and $(r, p) \leftarrow Gen(w)$, then $Pr[Rep(w', p) = r] \geq 1 - \epsilon'$, where the probability is over the randomness of (Gen, Rep), and the closeness $dist(w, w')$ is with respect to some distance measure such as Hamming distance.*

iv. The σ-security *property guarantees that for any distribution $W \in \mathcal{W}$, the key R is close to uniform conditioned on R. i.e., if $(R, P) \leftarrow Gen(W)$, then $\Delta((R, P); (U_{\xi}, P)) \leq \sigma$.*

We use the above definition for computational FE also, but replace the statistical distance with computational distance. The resulting definition matches Canetti et al. definition of FE [Definition 1,[10]].

Let s_{sec} be a polynomial in security parameter. An $(\mathcal{M}, \mathcal{W}, \xi, t', \epsilon', \sigma_{sec})$-*computational FE that is s_{sec}-hard* , follows Definition 2 and uses the following in lieu of item (iv):

iv. The *σ_{sec}-security* property guarantees that for any distribution $W \in \mathcal{W}$, if $(R, P) \leftarrow Gen(W)$, then for all adversary D of size at most s_{sec}, we have $\delta^{D}((R, P); (U_{\xi}, P)) \leq \sigma_{sec}$.

Reusability We use the following definition of reusability, that corresponds to reusability in the sense of (R1.2, R2.2), introduced in the introduction.

Definition 3 (Reusable fuzzy extractors [20]). *Let $\mathcal{W}$ be a family of probability distributions over $\mathcal{M}$. Let (Gen, Rep) be an $(\mathcal{M}, \mathcal{W}, \xi, t', \epsilon', \sigma)$-fuzzy extractor. Let $(W^1, ..., W^{\eta})$ be η correlated RV, where $W^j \in \mathcal{W}, \forall j \in \{1, ..., \eta\}$. For any (unbounded) adversary* D*, consider the following for $j = 1, ..., \eta$:*

- ***Sampling:*** *Challenger samples $w^j \leftarrow W^j$ and $u \leftarrow \{0,1\}^{\xi}$*
- ***Generation:*** *Challenger computes $(r^j, p^j) \leftarrow \mathsf{Gen}(w^j)$.*
- ***Distinguishing*** *Advantage of* D *is*

$$Adv_{\mathsf{D}} = \Pr[\mathsf{D}(r^j, \{r^i\}_{i=1,...,\eta, i \neq j}, \{p^i\}_{i=1}^{\eta}) = 1] - \Pr[\mathsf{D}(u, \{r^i\}_{i=1,...,\eta, i \neq j}, \{p^i\}_{i=1}^{\eta}) = 1]$$

An $(\mathcal{M}, \mathcal{W}, \xi, t', \epsilon', \sigma)$-fuzzy extractor is (η, σ_r)-reusable if for all D *and for all $j = 1, ..., \eta$, the advantage is at most $\sigma_r(\lambda)$.*

Boyen et al. [8] first defined robustness for fuzzy extractors in the sense of S1. The definition was strengthened by strength Dodis et al. [17] to S2. We use the following definition that combines the strongest notions of reusability and robustness (R1.2, R2.2, S2).

Definition 4 (Strongly robust of reusable fuzzy extractor). *Let $\mathcal{W}$ be a family of probability distribution over $\mathcal{M}$. Let $(W^1, ..., W^\eta)$ and W' be $(\eta+1)$ correlated random variables, where $W^j \in \mathcal{W}, \forall j \in \{1, ..., \eta\}$ and $W' \in \mathcal{W}$. Let $FE_{srr} = (Gen, Rep)$ be an (η, σ_r)-reusable $(\mathcal{M}, \mathcal{W}, \xi, t', \epsilon', \sigma)$-fuzzy extractor. We say FE_{srr} is $(q_d, \delta_r(\lambda))$-strongly robust if, for any adversary D, it holds that*

$$Adv^{rob}_{FE_{srr},\mathsf{D}} := Pr|Exp^{rob}_{FE_{srr},\mathsf{D}}(1^\lambda) = 1| \leq \delta_r(\lambda)$$

where $Exp^{rob}_{FE_{srr},\mathsf{D}}$ is a game played by the adversary D and the challenger $\mathcal{C}$.

Game $Exp^{rob}_{FE_{srr},\mathsf{D}}(1^\lambda)$: We define the following game for all $j = 1, ..., \eta$:

- ***Sampling:*** *Challenger samples $w^j \leftarrow W^j$ and $w' \leftarrow W'$*
- ***Generation:*** *Challenger computes $(r^j, p^j) \leftarrow \mathsf{Gen}(w^j)$ and returns $(r^j, p^j)_{j=1}^n$ to the adversary D.*
- ***Reproduction oracle queries:*** *Adversary D may adaptively make at most q_d reproduction oracle queries of the form $\tilde{p}^m$, $m \in \{1, \cdots, q_d\}$, to $\mathcal{C}$. Challenger $\mathcal{C}$ runs $Rep(w', \tilde{p}^m)$ and returns its output to D, $\forall m \in \{1, \cdots, q_d\}$.*
- ***Forgery test:*** *D submits its forgery $\tilde{p}$ to $\mathcal{C}$. A wins if $\tilde{p} \notin \{p^1, \cdots p^\eta\}$ and $Rep(w', \tilde{p}) \neq \perp$. The experiment outputs 1 if D wins and 0 otherwise.*

Computationally secure *reusable fuzzy extractor* and *strongly robust reusable* fuzzy extractors are obtained by replacing computationally unbounded adversary with probabilistic polynomial time (PPT) adversary in the above definitions.

3 Constructions

We construct three reusable fuzzy extractors satisfying (R1.2, R2.2) notion of reusability for structured sources, where Constructions 1 and 3 further provide robustness. Construction 1 is computationally secure, while Constructions 2 and 3 provide information-theoretic security.

3.1 Construction 1: Computational fuzzy extractor

The construction is inspired by the sample-then-lock approach of [10] but avoids using digital lockers, and is for (α, m)-*sources* that were introduced in in [10], and is recalled below.

Definition 5 (α-entropy m-samples [10]). *Let $W = W_1, \cdots, W_n$ be a source distributed over $\mathcal{V}^n$. For $\alpha > 0$, $m > 0$, we say that W is a (α, m)-source if $\tilde{H}_\infty(W[A] \mid A) \geq \alpha$, where A is a purely random subset of $[n]$ of cardinality m.*

Construction outline. The *Gen* algorithm uses ℓ random subsamples of W, to construct ℓ "encryptions" of a single random key R that will be sent to Bob. Each subsample is used as the secret key of a symmetric key encryption algorithm. Bob will be able to recover the key R if at least one of the subsamples of W perfectly matches the corresponding subsample of W', in which case Bob is able to correctly decrypt and recover R. We choose system parameters, including the subsample length m and the number of subsamples ℓ, to ensure that with a high probability at least one out of ℓ subsamples of W' and W match. We use LPN as one-time pad for the enryption. The encryption algorithm XORs an error correcting coded (ECC) version of the message with an LPN sample (with secret extracted from the subsample of W), and the decryption algorithm

removes the fixed part of the LPN sample, allowing the message to be recovered through decoding ECC.

To achieve robustness, we use an information theoretic one-time secure MAC algorithm on concatenation of the ℓ encrypted values. The MAC however needs to provide key-shift security to protect against an attacker that is able to tamper (shift) with the key and tag simultaneously.

In the following we first give the construction, and then motivate and describe the MAC algorithm that would be of independent interest.

Algorithm 1: $Gen(W)$: fuzzy extractor generation function. Ext is an average $(m, \alpha, \hat{\nu}, \epsilon)$-computational linear extractor for W using randomness Z. The random matrices $Z^{(i)}$ and subsets A_i (for $i \in \{1, \cdots, \ell\}$) as well as Z are common reference string (sampled using common random source r). H is a collision resistant cryptographc hash function. $|H|$ denotes output bit length of hash function H.

Input : $W = W_1, W_2, \cdots, W_n$ with $W_i \in \{0,1\}$
Output: An extracted key R and a ciphertext $(p_1, \cdots, p_\ell, T)$

1. $R \xleftarrow{\$} \{0,1\}^\xi$, $R_1 \xleftarrow{\$} \{0,1\}^{2\lambda}$
2. **for** $i \leftarrow 1$ **to** ℓ **do**
   - (i) Sample $A_i = \{i_1, \cdots, i_m\}$ from $[n]$ (using r)
   - (ii) Sample $Z^{(i)} \xleftarrow{\$} \{0,1\}^{\nu \times \hat{\nu}}$ (using r), where ν is the length of ECC.
   - (iii) Sample a random error vector $e^{(i)} = (e_1^{(i)}, \cdots, e_\nu^{(i)}) \in \{0,1\}^\nu$, where $e_j^{(i)}$ is i.i.d. according to $\Pr(e_j^{(i)} = 0) = \frac{1+\gamma}{2}$, $\forall j \in \{1, \cdots, \nu\}$
   - (iv) $d_i = Z^{(i)} \cdot \mathsf{Ext}(W[A_i], Z)$
   - (v) Set $p_i = ECC(0^t|R|R_1) + e^{(i)} + d_i$

   end
3. $p = H(p_1| \cdots |p_\ell)$
4. Let $L = \lceil |H|/\lambda \rceil + 4$
5. $T = Eval(p, R_1)$
6. Output key R, ciphertext $(p_1, \cdots, p_\ell, T)$

Algorithm 2: $T \leftarrow Eval(p, R_1)$

1. Encode p to vector $\mathbf{m}$ of length $L-4$ in $GF(2^\lambda)$
2. Parse $R_1 = x|y$ for $x, y \in \{0,1\}^\lambda$
3. Compute $T = x^L + x^2\mathbf{m}(x) + xy$ for $\mathbf{m}(x) = \sum_{i=0}^{L-5} m_i x^i$.
4. Return T

Construction 1 *Let $W = W_1, W_2, ..., W_n$ be a (α, m)-source with $W_i \in \{0,1\}$. Let Ext be an average $(m, \alpha, \hat{\nu}, \epsilon)$-computational linear extractor. Algorithm 1 (resp. Algorithm 2) describes the generation (resp. reproduction) procedure of our extractor, where $ECC(0^t|R|R_1)$ is the codeword for $0^t|R|R_1$.*

CRS Model. In Construction 1, we assume $A_1, \cdots, A_\ell$ and matrices $Z^{(1)}, \cdots, Z^{(\ell)}$ and Z are sampled uniformly over its respective domain and form the common random string (CRS). The sampling is done using the publicly known function with a shared random string r.

In our construction, this CRS assumption is important. Otherwise, given access to *Rep* oracle, the adversary might select some index set A_i with low entropy for $w'[A_i]$ and hence correctly guess $w'[A_i]$ with high probability, and

modifying P to be acceptable by *Rep* algorithm, leading to unequal keys at the sender and receiver. CRS model prevents attacker from choosing a weak sample.

Bit-string Encoding. We use an algorithm that encodes a bit string to elements in $GF(2^\lambda)$. Let ω be a fixed primitive element of $GF(2^\lambda)$. Then, we can encode λ bits $a_0 a_1 \cdots a_{\lambda-1}$ as $\sum_{i=0}^{\lambda-1} a_i \omega^i$. In this way, we can encode any binary string into a vector of field elements in $GF(2^\lambda)$. An incomplete block can be made into a λ bits by appending sufficiently many zeros. *This encoding of bit strings into vectors of field elements will be used in the rest of this paper.*

Algorithm 3: $Rep(W', (p_1, \cdots, p_\ell, T))$: fuzzy extractor reproduction function. The random matrices $Z^{(i)}$, random subsets A_i and randomness Z are common reference string (sampled using common random source r), $\forall i \in \{1, \cdots, \ell\}$. The system parameters $\ell, t, \lambda, \xi, n, m, L, \gamma$ are shared with sender.

Input : W' and ciphertext $(p_1, \cdots, p_\ell, T)$
Output: An extracted key R or $\perp$
1. $p = H(p_1 | \cdots | p_\ell)$
2. **for** $i \leftarrow 1$ **to** ℓ **do**
  (i) $d_i = Z^{(i)}\mathsf{Ext}(W'[A_i], Z)$
  (ii) Decode $(d_i + p_i)$ and let g_i' be the output of the decoder
  (iii) **if** *Hamming weight of* $(d_i + p_i) + ECC(g_i')$ *is more than* $\nu(\frac{1}{2} - \frac{\gamma}{3})$ **then**
    $g_i = \perp$
  else
    $g_i = g_i'$
  end
  (iv) **if** $((g_i \neq \perp) \wedge ((g_i)_{1\ldots t} = 0^t))$ **then**
    (a) Set $\rho = (g_i)_{t+1\ldots\nu}$
    (b) Set $R = (\rho)_{t+1\ldots t+\xi}, R_1 = (\rho)_{\nu-2\lambda+1\ldots\nu}$
    (c) Encode p to vector $\mathbf{m}$ of length $L-4$ in $GF(2^\lambda)$
    (d) $T' = Eval(p, R_1)$
    (e) If $T' = T$, output key R; otherwise, continue
  end
end
3. Output $\perp$

One-time Secure MAC with Key-Shift Security: A direct approach to provide robustness for FE is to use part of the generated key to compute an authentication tag and append it to $P = p \parallel tag$. However if Eve modifies p, the reconstructed MAC key by Bob will be different from the key used for the generation of tag, and so security of the MAC systems will be unclear. Our main observation in our construction to avoid this issue is that the key R is encrypted using "one-time pad" (with an LPN sample as the key) to obtain p. Because of linearity of OTP, any modification of an observed p to $\hat{p}$ will result in a known (to the adversary) shift $\delta = p + \hat{p}$ to the established key $R + \delta$. (We note that δ could depend on p because $\hat{p}$ is computed by attacker who knows tag $p \parallel tag$). This means that it is sufficient for the MAC to provide security against known shifts of the key. In the following, we propose an information-theoretic key-shift secure one-time MAC, and Lemma 2 proves its security.

For key (x, y), the MAC tag function is $T(x, y, \mathbf{m}) = x^L + x^2\mathbf{m}(x) + xy$, where $\mathbf{m}$ is a vector over $GF(2^\lambda)$ of length at most $L-5$ (so $\mathbf{m}(t)$ is a polynomial of degree at most $L-5$; see notations in Section 2).

Verification algorithm for a message and tag pair $m \parallel tag$ is by applying the tag function on m, and comparing the result with the received tag, tag.
The following lemma shows that the tag function has one-time authentication property; see Appendix A for a proof.

Lemma 2. *Let $L = 3$ mod 4 and m be an arbitrary but given vector of length at most $L-5$ over $GF(2^\lambda)$. Let x, y be uniformly random over $GF(2^\lambda)$. Then, given $T = x^L + x^2 m(x) + xy$, it holds*

$$(x+\delta_1)^L + (x+\delta_1)^2 m'(x+\delta_1) + (x+\delta_1)(y+\delta_2) = T' \tag{2}$$

with probability at most $L2^{-\lambda}$, where $T', \delta_1, \delta_2 \in GF(2^\lambda)$ and m' a vector over $GF(2^\lambda)$ of length at most $L-5$ with $m' \neq m$, are arbitrary but all deterministic in T and the probability is over the choices of (x, y).

Security of the Strongly Robustly Reusable Fuzzy Extractor. Theorem 1 proves security of Construction 1 using properties of the LPN-based encryption system, and security of the above MAC, in CRS model. The proof uses a generalized version of LPN assumption stated below.

Generalized LPN Assumption. For any ζ, t polynomial in μ, assume $A^{(i)} \in_R \{0,1\}^{t\times\mu}, U_t^{(i)} \in_R \{0,1\}^t$ are all uniformly distributed and $e^{(i)} = (e_1^{(i)}, \cdots, e_t^{(i)}) \in \{0,1\}^t$ with $e_j^{(i)}$ i.i.d. according to $\Pr(e_j^{(i)} = 0) = \frac{1+\gamma}{2}$, where $\gamma = \frac{1}{poly(\lambda)}$ for polynomial $poly(\lambda)$). The joint distribution of $(x^{(1)}, \cdots, x^{(\zeta)})$ is arbitrary but with the restriction that each $x^{(i)}$ is negligibly close to uniform over $\{0,1\}^\mu$. Then, the generalized LPN assumption states that $(A^{(i)}, A^{(i)}x^{(i)} + e^{(i)})_{i=1}^{\zeta} \approx (A^{(i)}, U_t^{(i)})_{i=1}^{\zeta}$.

We note that LPN assumption can be seen as an extreme case of the generalized LPN where $x^{(1)} = x^{(2)} = \cdots = x^{(\zeta)}$, and does not require $x^{(1)}, \cdots, x^{(\zeta)}$ to have such high dependency, suggesting plausibility of the assumption.

Our construction requires error-correcting code ECC can correct the error with probability $\frac{1}{2} - \frac{\gamma(\lambda)}{2}$. Such codes exist; see [21] for example. Now we give the security result for Construction 1.

Theorem 1 (Strongly robust and reusable Computational FE). *Let $\mathcal{W}$ be a family of (α, m)-sources over $\{0,1\}^n$ for $\alpha = \omega(\log(\lambda))$ and Ext is an average $(m, \alpha, \hat{\nu}, \epsilon)$-computational linear extractor for W using randomness Z, where λ is the security parameter. If adversary makes at most q_e fuzzy generation queries and q_d reproduction queries, then under the generalized LPN assumption, for any $s_{sec} = poly(\lambda)$, there exists some $\delta_r = NEGL(\lambda)$ and $\sigma_{sec} = NEGL(\lambda)$ such that (Gen, Rep) described in Construction 1 is a (q_d, δ_r)-strongly robust, (q_e, σ_{sec})-reusable $(\{0,1\}^n, \mathcal{W}, \xi, t', \epsilon', \sigma_{sec})$-computational fuzzy extractor that is s_{sec}-hard satisfying*
$(1 - (1 - \frac{t'}{n-m})^m)^\ell + \ell \cdot \frac{2^\alpha}{e^\alpha} \cdot 2^{-t} \cdot L \cdot 2^{-\lambda} \leq \epsilon'(\lambda)$ *and*
$\delta_r = \epsilon_{lpn} + q_d 2^{-\lambda} \ell (L+1) + \epsilon_H$, *where $L = \lceil |H|/\lambda \rceil + 4$ and $|H|$ is the output bit length of the hash unction H and ϵ_H is the probability to break the collision-resistancy of H and ϵ_{lpn} is the probability to break generalized LPN assumption (so that the secret in the assumption is ϵ-close to uniform).*

Proof. The proof of this theorem is in Appendix B.

3.2 Construction 2 & 3: IT-secure fuzzy extractors

The constructions in this section are for (α, m, N)**-sources** defined below.

Definition 6 (**(α, m, N)-source**). *Consider source $W = W_1, \cdots, W_n$ over alphabet $\mathcal{Z}$. We say W is a (α, m, N)-**source** if $\tilde{H}_\infty(W[A] \mid W[B], A, B) \geq \alpha$, where A (resp. B) is a purely random subset of $[n]$ of cardinality m (resp. N), where the probability is over randomness of W and index sets A, B.*

Intuitively the requirement is that entropy of a random sample remains high, given a random sample of cardinality up to N. In our construction, $\mathcal{Z} = \{0,1\}$.

Algo $Gen(W)$

1. $R \xleftarrow{\$} \{0,1\}^\xi$
2. **for** $i \leftarrow 1$ **to** ℓ **do**
   (i) Sample $A_i = \{i_1, \cdots, i_m\}$ from $[n]$ purely randomly
   (ii) $d_i = E(W[A_i], Z)$
   (iii) Set $p_i = ((0^t | R) \oplus d_i)$ for $t = \nu - \xi$

   end
3. Output key R and $(p_1 | A_1, \cdots, p_\ell | A_\ell)$

Algo $Rep(W', (p_1 | A_1, \cdots, p_\ell | A_\ell))$

1. **for** $i \leftarrow 1$ **to** ℓ **do**
   (i) $d_i = E(W'[A_i], Z)$

   (ii) **if** $((d_i \oplus p_i)_{1\ldots t} = 0^t)$ **then**
   output $R = (d_i \oplus p_i)_{t+1\ldots\nu}$

   end

   end
2. Output $\perp$

Figure. 1. Construction 2. Extractor randomness Z and parameters $\ell, t, \lambda, \xi, n, m, L$ are shared between sender and receiver; $(\rho)_{i\ldots j}$ denotes the block from the ith bit to jth bit in ρ.

Construction 2 *Let E be an average $(m, \alpha, \nu, \epsilon)$-extractor. Suppose that $W = W_1, W_2, ..., W_n$ be an (α, m, N)-source with W_i over alphabet $\{0,1\}$. The $Gen()$ and $Rep()$ algorithms in figure 1 in describe the fuzzy extractor generation $Gen()$ and reproduction procedure $Rep()$ respectively.*

Security analysis of fuzzy extractor Construction 2 This construction is reusable in the sense of (R1.2, R2.2). Let us start the security analysis with some preparation results. Claim 1 and Lemmas 3-4 are based on quite standard techniques of probabilistic distances. We give proofs in Appendix G for completeness.

The following claim is a well-known fact.

Claim 1. Let X and Y be random variables over $\mathcal{Z}$. Assume that $F : \mathcal{Z} \to \mathcal{V}$ is a deterministic function. Then $\Delta(F(X); F(Y)) \leq \Delta(X; Y)$.

This claim is correct if F is randomized as it holds for each fixed randomness. The following lemma can be proven by triangle inequality and induction on μ.

Lemma 3. *Let W be a (α, m, N)-source of length n over $\{0,1\}$. Let $E : \{0,1\}^m \times \{0,1\}^r \to \{0,1\}^\nu$ be an average $(m, \alpha, \nu, \epsilon)$-randomness extractor. Let Z be a uniform r-bit string and A_i be uniformly random subset of $[n]$ of size m for $i = 1, \cdots, \mu$. Let $d_i = E(W[A_i], Z)$ for $i = 1, \cdots, \mu$. Assume $E(\tilde{U}, Z) = U$, where $\tilde{U}$ is uniformly random over $\{0,1\}^m$ and U is the uniformly distributed over $\{0,1\}^\nu$. If $\mu m < N$, then, $\Delta(\mathbf{d}, Z, \mathbf{A}; U^\mu, Z, \mathbf{A}) \leq \mu\epsilon$, where $\mathbf{A} = (A_1, \cdots, A_\mu)$ and $\mathbf{d} = (d_1, \cdots, d_\mu)$.*

Lemma 4. *Let $W = W_1, \cdots, W_n$ be an (α, m, N)-source of length n over alphabet $\{0,1\}$. Let $E : \{0,1\}^m \times \{0,1\}^r \to \{0,1\}^\nu$ be an average $(m, \alpha, \nu, \epsilon)$-randomness extractor. Let A_i be uniformly random subset of $[n]$ of size m for $i = 1, \cdots, \ell$ and Z is a uniform r-bit string. Let $p_i = E(W[A_i], Z) \oplus 0^t | S$ for $S \leftarrow \{0,1\}^{\nu-t}$, $i = 1, \cdots, \ell$. Then, $\Delta(S, \mathbf{p}, Z, \mathbf{A}; U, \mathbf{p}, Z, \mathbf{A}) \leq 2\ell\epsilon$, where $\mathbf{p} = (p_1, p_2, \cdots, p_\ell), \mathbf{A} = (A_1, \cdots, A_\ell)$, $\ell m < N$ and $U \leftarrow \{0,1\}^{\nu-t}$.*

Theorem 2 (Reusable fuzzy extractor). *Let E be implemented by H, a strong universal hash family. Fix ℓ and let ξ be the length of the extracted key. Then for any $\epsilon'(\lambda)$, $\sigma(\lambda)$ satisfying $\xi \leq \alpha + 2 - 2 \cdot \log(\frac{2\ell}{\sigma}) - t$, (Gen, Rep) described in Construction 2 (i.e. figure 1) is a (η, σ)-reusable $(\mathcal{V}^n, \mathcal{W}, \xi, t', \epsilon', \sigma)$-fuzzy extractor, where $(1 - (1 - \frac{t'}{n-m})^m)^l + l \cdot 2^{-t} \leq \epsilon'(\lambda)$ and $\ell\eta m < N$.*

Proof. The proof of the following theorem is in Appendix D.

Adding robustness to Construction 2 We construct the first strongly robust, reusable, IT-secure FE, and does not use random oracle. The only other reusable FE with IT-security is due to Boyen [7] but does not provide robustness and relies on random oracle. The construction essentially uses the MAC in Section 3.1 to construct and append a tag to the ciphertext of Construction 2, and is in CRS model.

Algo $Gen(W)$

1. $R \xleftarrow{\$} \{0,1\}^\xi$, $R_1 \xleftarrow{\$} \{0,1\}^{2\lambda}$
2. **for** $i \leftarrow 1$ **to** ℓ **do**
   (i) Sample a purely random subset $A_i = \{i_1, \cdots, i_m\}$ from $[n]$ (using r)
   (ii) $d_i = E(W[A_i], Z)$
   (iii) Set $p_i = ((0^t | R | R_1) \oplus d_i)$ for $t = \nu - \xi - 2\lambda$

end

4. Let $L = \lceil \ell\nu/\lambda \rceil + 4$, $p = (p_1, \cdots, p_\ell)$
5. $T = Eval(p, R_1)$
3. Output key R and ciphertext $(p_1, \cdots, p_\ell, T)$

Algo $Rep(W', (p_1, \cdots, p_\ell, T))$

1. **for** $i \leftarrow 1$ **to** ℓ **do**
   (i) $d_i = E(W'[A_i], Z)$
   (ii) **if** $((d_i \oplus p_i)_{1\ldots t} = 0^t)$ **then**
      (a) Set $\rho = (d_i \oplus p_i)_{t+1\ldots\nu}$
      (b) Set $R = (\rho)_{t+1\ldots t+\xi}$, $R_1 = (\rho)_{\nu-2\lambda+1\ldots\nu}$, and
      (c) $T' = Eval((p_1, \cdots, p_\ell), R_1)$
      (d) **If** $T' = T$, **output** key R; **otherwise**, continue
   end

end

2. Output $\perp$

Figure. 2. fuzzy extractor generation (Gen) and reproduction (Rep) function. E is average $(m, \alpha, \nu, \epsilon)-$extractor. The random subsets A_i as well as Z are common reference string (sampled using common random source r), $\forall i \in \{1, \cdots, \ell\}$. The parameters $\ell, t, \lambda, \xi, n, m, L$ shared with sender. The procedure $Eval(\cdot)$ has been defined section 3.1.

Construction 3 *Let E be an average $(m, \alpha, \nu, \epsilon)$-extractor. Let $W = W_1, ..., W_n$ be an (α, m, N)-source, where $W_i \in \{0,1\}$. Figure 2 describes the generation Gen procedure and reproduction Rep procedure of strongly robust and reusable fuzzy extractor. Figure 3(a) and 3(b)(resp. Figure 3(c) and 3(d)) depict a pictorial representation of our fuzzy extractor generation (resp. reproduction) algorithm, where the encoding algorithm for $p_1 | \cdots | p_\ell$ is described in Section 3.1.*

CRS Model. We assume that sender and receiver have access to a common random string (common random string model). This random string is used to sample $A_1, \cdots, A_\ell$ and Z.

The receiver processes public string $(p_1, \cdots, p_\ell, T)$ in $Rep()$. As mentioned before, we use $(\rho)_{i\ldots j}$ to denote the block from the ith bit to jth bit in ρ.

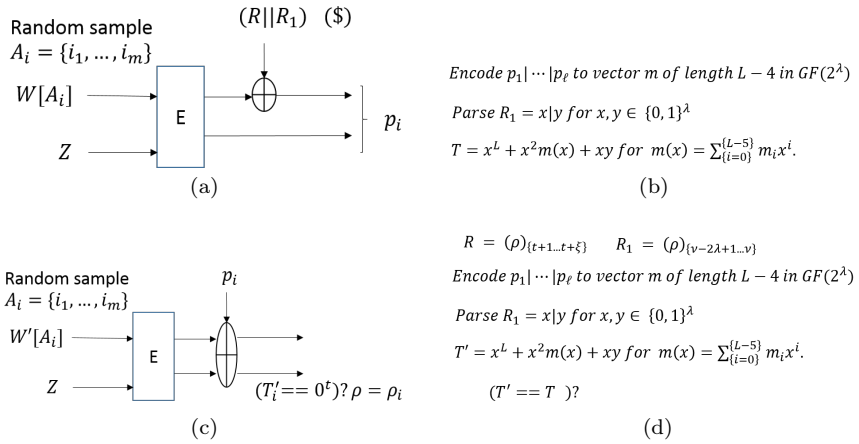


Figure. 3. High level diagram of fuzzy extractor construction 3. Both $Gen(W)$ and $Rep(W')$ procedure are split into two parts. 3(a): The first part of $Gen(W)$. This part iterates ℓ times. 3(b): The second part of $Gen(W)$. This step is executed after completion of ℓ iterations of 3(a). 3(c): The First part of $Rep(W')$. This step is executed at most ℓ times until one match $(T_i' == 0^t)$ is found. 3(d): The second part of $Rep(W')$. This part is executed after one match in 3(c). If verification of $(T' == T)$ fails, the algorithm continues from the part 3(c) again. The upper line of each extractor represents higher order $\xi + 2\lambda$ bits of its output i.e. $(E(.))_{t+1\ldots\nu}$. The lower line of each extractor represents lower order t bits of its output i.e. $(E(.))_{1\ldots t}$. Extracted key is R. Randomness Z is public

Security analysis of fuzzy extractor Construction 3 In this section, we provide security properties of Construction 3.

Lemma 5 is based on quite standard techniques of probabilistic distances. We give its proof in Appendix G for completeness.

Lemma 5. $\Delta(R, \mathbf{p}, Z, \mathbf{A}, T; U, \mathbf{p}, Z, \mathbf{A}, T) \leq 4\ell\epsilon$.

The construction's security property is stated as follows.

Theorem 3 (Robust and reusable FE). *Let $\mathcal{W}$ be a family of (α, m, N)-sources and E be an average $(m, \alpha, \nu, \epsilon)$-randomness linear extractor for source W using randomness Z. Let E be implemented by H, a strong universal hash family. Fix ℓ and let ξ be the length of the extracted key. If adversary makes at most q_e generation queries and q_d reproduction queries, then for any $\delta(\lambda)$, $\epsilon'(\lambda)$, $\sigma(\lambda)$ satisfying $\xi \leq \alpha + 2 - 2 \cdot \log(\frac{4\ell}{\sigma}) - t - 2\lambda$, (Gen, Rep) described in Construction 3 is a (q_d, δ)-strongly robust, (q_e, σ)-reusable $(\mathcal{V}^n, \mathcal{W}, \xi, t', \epsilon', \sigma)$-fuzzy extractor, where $(1 - (1 - \frac{t'}{n-m})^m)^l + l \cdot 2^{-t} \cdot L \cdot 2^{-\lambda} \leq \epsilon'(\lambda)$, and $\delta = (q_d + q_e)\ell\epsilon + q_d 2^{-\lambda}\ell(L+1)$, where $(q_e + q_d)\ell m < N$ and $L = \lceil \ell\nu/\lambda \rceil + 4$.*

Proof. The proof of this theorem is in Appendix E.

4 Implementation of Construction 1 using iris

There are two concurrent work that implemented a modified version of Canetti et al.'s construction. Sinhadri et al. 's modification is to improve storage efficiency,

while Cheon et al. also aims to add robustness to the scheme. However the latter construction is shown [33] to be flawed, and no corrected version has been published. We thus follow the experimental framework of [33] with the goal of evaluating the cost of security in standard model, and providing robustness.

Our implementation is for Construction 1. It is implemented in Python 3 and open sourced, and the open-source code url is given in section 5. The construction requires an error-correcting code. We use Low-Density Parity-Check (LDPC) Code as the underlying error-correcting code. The main modules/libraries that are used in our implementation are as follows: *numpy* and *galois* - used for fast matrix and Galois field operations, respectively; *secrets* - used to generate random bitstrings (using OS randomness); LDPC, – C library for LPDC error-correcting code [1]; OSIRIS – Open Source IRIS software package [28].

Both *Gen* (i.e. Algorithm 1) and *Rep* (i.e. Algorithm 3) algorithms are multi-threaded. In *Gen* algorithm, we need several matrix multiplications. We split those tasks given in lines $(iii) - (v)$ of Algorithm 1 among several cores evenly and run them in parallel. *Rep*, also requires several matrix multiplications and error corrections. We split the workload among several cores evenly, and run the program until either a match is found, or all operations are completed.

4.1 Iris data processing, parameter setting and error correction

The construction parameters can be grouped three groups: 1) *iris data processing parameters* includes n, the size of the iris data w, and m, the subsample size etc.; 2) *system parameters* that includes the number of subsamples ℓ, the reliability parameter $(1-\epsilon)$ and MAC-related parameters etc.; and 3) *error-correcting code-related parameters* that includes the length of message, length of codeword and the dimensions of the generator matrix etc.

We use ND-IRIS-0405 iris dataset that contains iris images of 365 persons, each with multiple images, in total 64964 iris images.

Iris data processing We use OSIRIS [28], an Open Source IRIS software package, that takes an iris image and produces a binary vector of length 32768.

The OSIRIS library has six transforms. We used Transform 5 in our experiments, which is consistent with Simhardi et al.'s [33] work. Their experiments showed that Transform 5 has the lowest error rate for images of the same iris.

Size of barometric sample W. The initial encoding of the iris as a 32768 bit vector is shortened to 15000 bits, after applying "masks" that capture imperfections of the image (e.g. due to eyelashes). Thus, we set $n = 15000$, as the size of the biometric W.

Subsample size and entropy. Simadri et al. used sample $m = 43$, with estimated entropy 32 bits. These values have been obtained through extensive experiments with the same data set and using OSIRIS software. We thus adopt these values in our experiments.

Threshold error rate. Iris matching algorithm compares the Hamming distance between two iris samples with a threshold, to accept or reject a match. This threshold, $T_H = \frac{t'}{n}$, where t' is the Hamming distance between two transformed

iris codes, is commonly set between 20% and 35% [15]. We use $T_H = 28\%$ that is similar to the value used in [33]. We only use T_H to compute an approximate value of ℓ that we discuss now.

System parameters. We set $\xi = 128$, $\lambda = 128$ and $t = 12$ and $\epsilon = 0.5$. Using Theorem 1, we have $(1 - (1 - \frac{t'}{n-m})^m)^\ell + \ell \cdot \frac{2^\alpha}{e^\alpha} \cdot 2^{-t} \cdot (L \cdot 2^{-\lambda} + \epsilon_H) \leq \epsilon'(\lambda)$, which noting that the second term on LHS is negligible, results in $(1-(1-\frac{t'}{n-m})^m)^\ell = \epsilon'(\lambda)$. Now substituting parameters, T_H, m and n, we obtain the number of iteration $\ell = 10^6$ (approximately). We set the error rate for LPN based symmetric key encryption to 11% i.e. $(\frac{1}{2} - \frac{\gamma}{2}) = .11$. This matches the values in [5]. We also experimented with error rate 12%, 12.5% and 13%, but the message recovery rate for these error rates was unacceptable for our use-case.

Error correcting code-related parameters The size of the message $(0^t|R|R_1)$ (Algorithm 1) to be encrypted is 396 bits. We use an LDPC code with length $\nu = 1228$ bits, and message length of 396 bits. Errors were introduced using the `transmit` function provided by the LDPC library [1]. We simulated a binary symmetric channel using `bsc` command with error rate 0.11. More details on LDPC experiments are given in Appendix H.

4.2 Evaluation

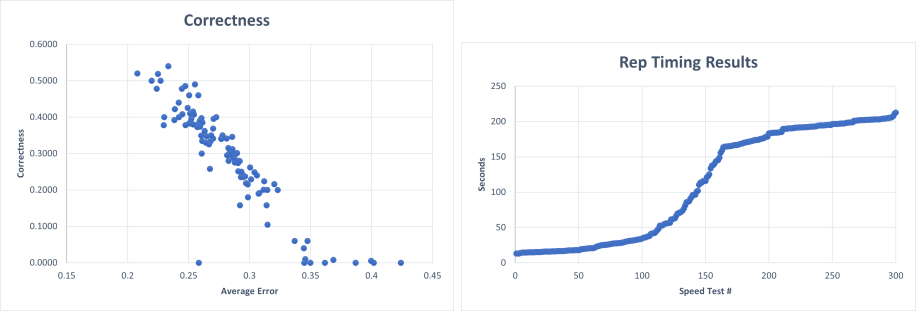
We evaluate the computation time and achievable correctness in practice.

Computation time. The implementation was developed using Python 3.8.9+, and all testing was done on ARC cluster (50 cores, 30GB of RAM) using Python version 3.10.4. The computation time of *Gen* and *Rep* algorithms are given below. Because our matrices and the full ciphertext (10^6 matrices and vectors respectively) could not fit in our available RAM, we had access disk and that slowed the computation. In our results, we exclude the disk access to show the computation time. We measured computation times as follows: randomly pick 5 from a particular class (set of all images of one person's iris) from the ND-IRIS-0405 dataset (300 images in total); run *Gen* on a random image in each class; run *Rep* on every other image in said class. We do not include timings from correctness experiments 4.2, since during conducting them, we had access to less computing power (25 cores per experiment). Our results are as follows.

–*Gen* takes approximately 550 seconds to generate the ciphertexts. (This excludes the time to generate sample positions.)
–*Rep* run time on average is about 111 seconds (excluding disk access). The worst and best times in our experiments were 213 and 13 seconds, respectively.
–*MAC* calculation took 1 second on average, excluding concatenation time of subsample ciphertext.

The execution time distribution for *Rep* is given in Figure 4(b).

Correctness We tested correctness across the ND-IRIS-0405 dataset. The correctness experiments were conducted as follows: *Gen* was run on the first image from a particular class (set of all images of one person's iris), followed by *Rep* on all other images from that class. Correctness rate is the number of successful derivations over the total size of the class. The experiment was run once per class, to keep the computations manageable.



(a) (b)

Figure. 4. (a) Correctness of *Rep* versus the error rate of a person's iris (i.e., class).; (b) Execution times of 300 *Rep* experiments. Times over 300 seconds correspond with failures to recover the key.

We ran the above experiment on 125 classes out of the 356 available in the dataset. We weren't able to complete the remaining 231 experiments due to issues with the ARC cluster. The full set of results will be included in the full paper. Our results are given in Figure 4(a). We observed 50% correctness at around 0.23 average error rate. Compare to Simhardi et al.'s [33] (50% correctness at around 0.32 average error rate), we achieve lower correctness, which can be explained by the additional 0.11 noise introduced by LPN encryption.

5 Concluding remarks

We propose three constructions of FEs, each with important properties in relation to other known constructions, detailed in Table 1, and Construction 1 answering an open question (Canetti et al. 2016). All constructions are secure against an adversary with access to quantum computer. We also implemented Construction 1 to show real-life feasibility of the construction. Although the implementation provides interesting insight into the cost of post-quantum security in standard model, and providing robustness, it can be optimized both by parallelizing computation, and improving memory management. Extending our FEs to tolerate linear error, and construction of IT-secure FE for more general sources are interesting open questions.

Open source code

The source code for the proof-of-concept implementation of the proposed iris key derivation and authentication system based on our Construction 1 can be found at `https://github.com/fuzzy-ext-anon/Comutational-Fuzzy-Extractor`.

References

1. Software for low density parity check (ldpc) codes. https://glizen.com/radford/ldpc.software.html, 2012.
2. Alamélou, Berthier, Cachet, Cauchie, Fuller, Gaborit, and Simhadri. Pseudoentropic isometries: A new framework for fuzzy extractor reusability. In *AsiaCCS 2018*, pages 673–684, 2018.

3. Apon, Cho, Eldefrawy, and Katz. Efficient, reusable fuzzy extractors from LWE. In *CSCML 2017, Proc.*, volume 10332 of *LNCS*, pages 1–18. Springer, 2017.
4. C. H. Bennett, G. Brassard, and J.-M. Robert. Privacy Amplification by Public Discussion. *SIAM J. Comput.*, 17(2):210–229, apr 1988.
5. S. Bogos, F. Tramèr, and S. Vaudenay. On solving lpn using bkw and variants. *Cryptography and Communications*, 8:331–369, 2016.
6. Bowyer and Flynn. The ND-IRIS-0405 iris image dataset. *CoRR*, abs/1606.04853, 2016.
7. Boyen. Reusable cryptographic fuzzy extractors. In *Proceedings of the 11th ACM Conf. on Computer and communications security*, pages 82–91, 2004.
8. Boyen, Dodis, Katz, Ostrovsky, and Smith. Secure remote authentication using biometric data. In *Proc. EUROCRYPT 2005*, pages 147–163. Springer, 2005.
9. Boyen, Mei, and Waters. Direct Chosen Ciphertext Security from Identity-Based Techniques. In *Proc. 12th ACM Conf. - CCS '05*, page 320. ACM, ACM Press, 2005.
10. Canetti, Fuller, Paneth, Reyzin, and Smith. Reusable fuzzy extractors for low-entropy distributions. Cryptol. ePrint Archive, Report 2014/243, 2014. `https://ia.cr/2014/243`.
11. Canetti, Pass, and Shelat. Cryptography from sunspots: How to use an imperfect reference string. In *FOCS'07*, pages 249–259, 2007.
12. Cheon, Jeong, Kim, and Lee. A reusable fuzzy extractor with practical storage size: Modifying canetti et al.'s construction. In *ACISP 23 2018*, LNCS, pages 28–44, 2018.
13. Cramer, Dodis, Fehr, Padró, and Wichs. Detection of algebraic manipulation with applications to robust secret sharing and fuzzy extractors. In *Advances in Cryptol. - EUROCRYPT 2008*.
14. Daugman. How iris recognition works. *IEEE Trans. Circuits Syst. Video Technol.*, 14(1):21–30, 2004.
15. Daugman. Probing the uniqueness and randomness of iriscodes: Results from 200 billion iris pair comparisons. *Proceedings of the IEEE*, 94(11):1927–1935, 2006.
16. Dodis, Kalai, and Lovett. On cryptography with auxiliary input. In *STOC 2009*, pages 621–630. ACM, 2009.
17. Dodis, Kanukurthi, Katz, Reyzin, and D. Smith. Robust fuzzy extractors and authenticated key agreement from close secrets. *IEEE Trans. Inf. Theory*, 58(9):6207–6222, 2012.
18. Dodis, Ostrovsky, Reyzin, and Smith. Fuzzy extractors: How to generate strong keys from biometrics and other noisy data. *SIAM J. Comput.*, 38(1):97–139, 2008.
19. Dodis, Reyzin, and Smith. Fuzzy extractors: How to generate strong keys from biometrics and other noisy data. In *Advances in Cryptol. - EUROCRYPT 2004*, pages 523–540, 2004.
20. C. et al. Reusable fuzzy extractors for low-entropy distributions. In *Annual Int. Conf. Theory Appl. Cryptographic Techniques*, pages 117–146. Springer, 2016.
21. Forney. Concatenated codes. MIT Press, Cambridge, MA, 1966.
22. Fuller, Meng, and Reyzin. Computational fuzzy extractors. In *Advances in Cryptol. - ASIACRYPT 2013*, LNCS, pages 174–193, 2013.
23. Fuller, Reyzin, and Smith. When are fuzzy extractors possible? *IEEE Trans. Inf. Theory*, 2020.
24. Impagliazzo, Levin, and Luby. Pseudo-Random Generation from One-Way Functions. In *STOC '89*, pages 12–24. ACM Press, 1989.
25. Islam, Safavi-Naini, and Kneppers. Scalable behavioral authentication. *IEEE Access*, 9:43458–43473, 2021.
26. T. A. Jr. Iit madras course on ldpc and polar codes for the 5g standard. `https://github.com/tallamjr/iit-madras-5G-standard/tree/master/matlab/base_matrices`, 2020.

27. Karakaya, Alptekin, and İncel. Using behavioral biometric sensors of mobile phones for user authentication. *Procedia Computer Science*, 159:475–484, 2019.

28. Krichen, Mellakh, Salicetti, and Dorizzi. Osiris (open source for iris) reference system, 2017.

29. Majzoobi, Koushanfar, and Potkonjak. Lightweight secure pufs. In *2008 IEEE/ACM Int. Conf. on Computer-Aided Design*, pages 670–673. IEEE, 2008.

30. Pfitzmann and Waidner. A model for asynchronous reactive systems and its application to secure message transmission. In *Proc. 2001 IEEE S&P 2001*, pages 184–200. IEEE, 2000.

31. Phillips, Bowyer, Flynn, Liu, and Scruggs. The iris challenge evaluation 2005. IEEE BTAS 08, 2008.

32. Phillips, Scruggs, O'Toole, Flynn, Bowyer, Schott, and Sharpe. FRVT 2006 and ICE 2006. *IEEE Trans. Pattern Anal. Mach. Intell.*, 32:831–846, 2010.

33. Simhadri, Steel, and Fuller. Cryptographic authentication from the iris. In *ISC 22, 2019*, volume 11723, pages 465–485, 2019.

34. Suh and Devadas. Physical unclonable functions for device authentication and secret key generation. In *2007 44th ACM/IEEE Design Automation Conf.*, pages 9–14, 2007.

35. Wen and Liu. Reusable fuzzy extractor from LWE. In *Inf. Secur. and Priv. - 23rd Australasian Conf., ACISP 2018*, volume 10946 of *LNCS*, pages 13–27. Springer.

36. Wen and Liu. Robustly reusable fuzzy extractor from standard assumptions. In *ASIACRYPT 2018*, volume 11274 of *LNCS*, pages 459–489. Springer.

37. Wen, Liu, and Gu. Generic constructions of robustly reusable fuzzy extractor. In *PKC 2019*, pages 349–378. Springer International Publishing, 2019.

38. Wen, Liu, and Han. Reusable fuzzy extractor from the decisional diffie-hellman assumption. *Des. Codes Cryptogr.*, 86(11):2495–2512, 2018.

A Proof of Lemma 2

We say (m', T) *valid* when Eq. (2) is satisfied. Then, given T, Eq. (2) holds with probability

$$\begin{aligned}
&\mathbb{E}_T\Big(P_{xy}[(m', T') \text{ valid} \mid T]\Big) \\
\leq &\sum_{a \in GF(2^\lambda)} P_{xy}[(m', T') \text{ valid}, T = a] \\
\leq &\sum_a P_{xy}[(m', T') \text{ valid}, T = a, x = 0] \\
&+ \sum_a P_{xy}[(m', T') \text{ valid}, T = a, x \neq 0] \\
\leq &2^{-\lambda} + \sum_a P_{xy}[(m', T') \text{ valid}, T = a, x \neq 0] \qquad (3)
\end{aligned}$$

In the following, we bound $\sum_a P_{xy}[(m', T') \text{ valid}, T = a, x \neq 0]$. We study this for case $\delta_1 = 0$ and case $\delta_1 \neq 0$. We will frequently use the assumption that $T', m', \delta_1, \delta_2$ are determined by T. So given $T = a$, $(T', m', \delta_1, \delta_2)$ are all fixed).

Case $\delta_1 = 0$. In this case, $T' - T = x^2(m'(x) - m(x)) + x\delta_2$. Since $m' \neq m$, this is a non-zero polynomial of degree at most $L - 3$ (when fix $T = a$). Hence,

it is satisfied with at most $L-3$ possible x. Further, for any non-zero x and any $a \in GF(2^\lambda)$, there exists a unique y s.t (x,y) results in $T=a$. Thus, given $T=a \wedge x \neq 0$, x is uniformly random over $GF(2^\lambda)-\{0\}$. With these facts in mind, we have

$$\begin{aligned}
&P_{xy}[(m',T') \text{ valid} \mid x\neq 0, T=a, \delta_1=0] \cdot \Pr(x\neq 0, T=a, \delta_1=0) \\
\leq &P_{xy}\big[(x^2(m'(x)-m(x))+x\delta_2 = T'-a) \mid x\neq 0, T=a, \delta_1=0\big] \cdot \Pr(x\neq 0, T=a, \delta_1=0) \\
\leq &P_{xy}\big[(x^2(m'(x)-m(x))+x\delta_2 = T'-a) \mid x\neq 0, T=a\big] \cdot \Pr(x\neq 0, T=a, \delta_1=0) \qquad (4) \\
\leq &\frac{L-3}{2^\lambda-1}\Pr(x\neq 0, T=a, \delta_1=0),
\end{aligned}$$

where Eq. (4) follows from the fact that δ_1 is determined by T. Hence,

$$\sum_a P_{xy}[(m',T') \text{ valid}, T=a, x\neq 0, \delta_1=0] \leq \frac{L-3}{2^\lambda-1}\Pr(x\neq 0, \delta_1=0). \qquad (5)$$

Case $\delta_1 \neq 0$. In this case, $T'-T=\delta_1 x^{L-1}+\delta_1^2 x^{L-2}+Q(x)+\delta_1 y$, where $Q(x)$ is some polynomial of degree at most $L-3$ (when fix $T=a$). Here we use the fact: since $GF(2^\lambda)$ has character is 2 and $L=3 \bmod 4$, it follows that both L and $(L-1)L/2$ are 1 in $GF(2^\lambda)$. Representing y in terms of x and substituting it into T, we have $a=T=\delta_1 x^{L-1}+\mu(x)$ for some polynomial $\mu(x)$ of degree at most $L-3$ (when fix $T=a$). There are at most $L-1$ possible x to satisfy this. Further, when $T=a$ and x are fixed with $x\neq 0$, there is a unique y satisfying $T=a$. Hence, given $T=a$ and the fact $x\neq 0$, we know that x is uniformly random over $GF(2^\lambda)-\{0\}$. Finally, we again remind that $T', m', \delta_1, \delta_2$ are determined by T (hence, given $T=a$, $(T', m', \delta_1, \delta_2)$ are all fixed). With these facts in mind, we have (similar to Case $\delta_1=0$)

$$\begin{aligned}
&P_{xy}[(m',T') \text{ valid} \mid x\neq 0, T=a, \delta_1\neq 0] \cdot \Pr(x\neq 0, T=a, \delta_1\neq 0) \\
\leq &P_{xy}\big[\delta_1 x^{L-1}+\delta_1^2 x^{L-2}+Q(x)+\delta_1 y = T'-a \mid x\neq 0, T=a\big] \cdot \Pr(x\neq 0, T=a, \delta_1=0) \qquad (6) \\
\leq &\frac{L-1}{2^\lambda-1}\Pr(x\neq 0, T=a, \delta_1\neq 0),
\end{aligned}$$

Hence,

$$\sum_a P_{xy}[(m',T') \text{ valid}, T=a, x\neq 0, \delta_1=0] \leq \frac{L-1}{2^\lambda-1}\Pr(x\neq 0, \delta_1\neq 0). \qquad (7)$$

Combining Eq. (5)(7), we have $\sum_a P_{xy}[(m',T') \text{ valid}, T=a, x\neq 0] \leq \frac{L-1}{2^\lambda-1}\Pr(x\neq 0)$. Notice $\Pr(x\neq 0)=\frac{2^\lambda-1}{2^\lambda}$. We know that $\sum_a P_{xy}[(m',T') \text{ valid}, T=a, x\neq 0] \leq \frac{L-1}{2^\lambda}$. Plugging in Eq. (3), we obtain our result. □

B Proof of Theorem 1

To prove that Construction 1 is a computational fuzzy extractor, we need to prove the ϵ'-correctness and σ_{sec}-security of the computational extractor. We first prove the correctness of the algorithm and then argue security.

Reliability (Correctness). We first consider correctness error. That is, when there is no attack, the ciphertext will be accepted with high probability. In our proposed construction, the parameters m and ℓ represents a trade-off between correctness and efficiency. We need to choose a value for ℓ and m. Increasing ℓ improves correctness; however, it decreases efficiency. As mentioned earlier, we assume that the maximum Hamming distance between W and W' is t' i.e. $d(W, W') \leq t'$. For any i,
$Pr[(W'_{j_{i,1}}, W'_{j_{i,2}}, \cdots, W'_{j_{i,m}}) = (W_{j_{i,1}}, W_{j_{i,2}}, \cdots, W_{j_{i,m}})] \geq (1 - t'/(n-m))^m$, where $A_i = \{i_1, \cdots, i_m\} = \{j_{i,1}, j_{i,2}, \cdots, j_{i,m}\}$ and $1 \leq j_{i,1}, j_{i,2}, \cdots, j_{i,m} \leq n$. This is because $Pr[W'_{j_{i,1}} = W_{j_{i,1}}] \geq (1 - t'/n)$ as $j_{i,1}$ is uniform. Conditioned on that event the probability that $W'_{j_{i,2}} = W_{j_{i,2}}$ is at least $1 - t'/(n-1)$. Thus,
$Pr[(W'_{j_{i,1}}, W'_{j_{i,2}}, \cdots, W'_{j_{i,m}}) = (W_{j_{i,1}}, W_{j_{i,2}}, \cdots, W_{j_{i,m}})]$
$\geq (1 - t'/n)(1 - t'/(n-1)) \cdots (1 - t'/(n-m))$
$\geq (1 - t'/(n-m))^m$. Therefore, the probability that no v_i matches at the receiver is at most $(1 - (1 - \frac{t'}{n-m})^m)^\ell$, where $v_i = (W'_{j_{i,1}}, W'_{j_{i,2}}, \cdots, W'_{j_{i,m}})$, and $i \in \{1, 2, \cdots, \ell\}$.

In addition, *Rep* in Algorithm 3 may return an incorrect extracted key due to an error at step (iv) and verification of $T' = T$ at step (e). These errors might occur due to collision that we now explain in detail. Note that steps (i)-(iii) of *Rep* represents the decryption algorithm of the CPA secure symmetric encryption scheme (or the reusable fuzzy extractor) due to Dodis et el. [16] (section 4.1). From the unique-key property of the reusable extractor (i.e. CPA secure symmetric encryption scheme) due to Dodis et el. (Claim 4.1 of [16]), the probability that $(g_i \neq \perp)$ occurs when the generation (i.e. encryption) and reproduction (i.e. decryption) keys are different, is at most $2^\alpha e^{-\alpha}$ (as $\tilde{H}_\infty(W[A_i] \mid A_i) \geq \alpha, \forall i \in \{1, \cdots, \ell\}$). Hence, an error at step (iv) may occur with probability at most $2^\alpha e^{-\alpha} \cdot 2^{-t}$. Indeed, if $\mathsf{Ext}(W[A_i], Z) \neq \mathsf{Ext}(W'[A_i], Z)$, then $Z^{(i)} \cdot (\mathsf{Ext}(W[A_i], Z) - \mathsf{Ext}(W'[A_i], Z))$ is uniformly random independent of $ECC(0^t|R|R_1)$. Hence, if

$$Z^{(i)}(\mathsf{Ext}(W[A_i], Z) - \mathsf{Ext}(W'[A_i], Z)) + e_i + ECC(0^t|R|R_1)$$

can be decoded, the message will be uniformly random and thus has a pattern 0^t with probability at most 2^{-t}. Furthermore, similar to lemma 2 (except $\delta_1|\delta_2 = (d_i \oplus d'_i)_{\nu-2\lambda+1\cdots\nu}$ with d_i from W and d'_i from W'), a collision $T' = T$ occurs with probability $(L \cdot 2^{-\lambda})$. *Rep* may return an incorrect extracted key if there is at least one collision at step (iv) together with a collision $T' = T$ at step (e). This might happen with any of the ℓ executions of both step (iv) and step (e) and hence with probability at most $\ell \cdot \frac{2^\alpha}{e^\alpha} \cdot 2^{-t} \cdot L \cdot 2^{-\lambda}$. Therefore, we need to

set ℓ and m so that

$$(1-(1-\frac{t'}{n-m})^m)^\ell + \ell \cdot \frac{2^\alpha}{e^\alpha} \cdot 2^{-t} \cdot L \cdot 2^{-\lambda} \leq \epsilon'(\lambda) \tag{8}$$

where $\epsilon'(\lambda)$ is the fuzzy extractor's allowable error parameter.

Security. We now prove its security. We would like to show that $\delta^{\mathcal{D}}((R,P);(U_\xi,P)) \leq \sigma_{sec}$, where P is the public string.

At step (i) of Gen in Algorithm 1, we sample a purely random subset $A_i = \{i_1, \cdots, i_m\}$ from $[n]$ of size m. In line (ii), we sample a purely random matrix $Z^{(i)} \in \{0,1\}^{\nu \times \hat{\nu}}$. Furthermore, at step (iii), we sample a random error vector $e^{(i)} = (e_1^{(i)}, \cdots, e_\nu^{(i)}) \in \{0,1\}^\nu$, where $\Pr(e_j^{(i)} = 0) = \frac{1+\gamma}{2}$, $\forall j \in \{1, \cdots, \nu\}$. Note that Z is also a purely random sample.

Since W is an (α, m)-source, we have $\tilde{H}_\infty(W[A_i]|A_i) \geq \alpha$, where A_i is a purely random subset of $[n]$, $\forall i \in \{1, \cdots, \ell\}$.

In line (v), we have $p_i = (ECC(0^t|R|R_1) + e^{(i)} + Z^{(i)}\mathsf{Ext}(W[A_i], Z))$ (as $d_i = Z^{(i)} \cdot \mathsf{Ext}(W[A_i], Z)$). Note that, under the generalized Learning Parity with Noise (LPN) assumption, $(ECC(0^t|R|R_1) + e^{(i)} + Z^{(i)}\mathsf{Ext}(W[A_i], Z))_{i=1}^\ell$ becomes a (CPA) secure symmetric encryption of the message $(0^t|R|R_1)$, where W is the secret key. This is because (1) $\mathsf{Ext}(W[A_i], Z)$ is negligibly close to uniform, and (2) due to generalized LPN assumption, $\{e^{(i)} + Z^{(i)}\mathsf{Ext}(W[A_i], Z)\}_i$ are computationally indistinguishable from jointly uniformly random tuples.

Hence, (Gen, Rep) is a $(\mathcal{V}^n, \mathcal{W}, \xi, t', \epsilon', \sigma_{sec})$-computational fuzzy extractor, where $(1-(1-\frac{t'}{n-m})^m)^\ell + \ell \cdot \frac{2^\alpha}{e^\alpha} \cdot 2^{-t} \cdot L \cdot 2^{-\lambda} \leq \epsilon'(\lambda)$.

Reusability. Following the same argument above except noticing that due to generalized LPN assumption, $\{e^{(i)} + Z^{(i)}\mathsf{Ext}(W[A_i], Z)\}_i$ for all instances are computationally indistinguishable from jointly uniformly random tuples. Thus, after leaking all the instances except the test one, $\{e^{(i)} + Z^{(i)}\mathsf{Ext}(W[A_i], Z)\}_i$ in the test instance are still computational uniformly random and R masked by this tuple is computationally indistinguishable from uniform.

Robustness. The robustness is proven in Appendix C. □

C Robustness Proof for Construction 1

Proof. We need to show that our construction satisfies Definition 4. We show that through q_e generation queries and q_d reproduction queries, the robustness is broken only negligibly. Assume $\mathcal{A}$ is the robustness attacker. Define Γ_0 to be the robustness game where challenger acts normally for each generation and reproduction query. Let $\hat{d}_i = d_i + e_i$. We now revise Γ_0 to Γ_1 so that $\hat{d}_i = Z^{(i)} \cdot \mathsf{Ext}(W[A_i], Z) + e_i$ is replaced by $\hat{d}_i \leftarrow \{0,1\}^\nu$ in generation query or reproduction query (for some $(Z^{(i)}, A_i)$ that does not appear in a previous query; otherwise, use the existing $\hat{d}_i$). Let $\mathbf{Succ}(\Gamma)$ be the success event of $\mathcal{A}$ in game Γ. Then, we consider a distinguisher $\mathcal{D}$ that distinguishes Γ_0 and Γ_1. Upon receiving $\hat{d}_i$ that is either $d_i + e_i = Z^{(i)} \cdot \mathsf{Ext}(W[A_i], Z) + e_i$ or uniformly random, where $Z^{(i)}, A_i, Z$ are sampled using r (where A_i, Z serves as the description of $x_i = \mathsf{Ext}(W[A_i], Z)$), $\mathcal{D}$ simulates Γ_0 with $\mathcal{A}$ against it by providing the latter with public randomness

r, except all $\hat{d}_i$'s are from his challenge tuple. For brevity, we implicitly assume (without a mention) that whenever $\mathcal{A}$ makes a (generation or reproduction) query, both $\mathcal{A}$ and $\mathcal{D}$ knows the underlying parameters $Z^{(i)}, A_i, Z$. Further, to be consistent, whenever $\mathcal{A}$ makes a reproduction query with parameters $Z^{(i)}, A_i, Z$ from the generation query i, $\mathcal{A}$ will simulate the response using the challenge $\hat{d}_i$ corresponding to $Z^{(i)}, A_i, Z$. Finally, if $\mathcal{A}$ succeeds, $\mathcal{D}$ outputs 0; otherwise, it outputs 1. Since $A_1, \cdots, A_\ell$ by challenger or adversary is from public random string, $W[A_i]$ follows the distribution of (α, m)-source W. Hence, by definition of Ext and generalized LPN assumption, we immediately have the following.

Lemma 6. *$|P(\mathbf{Succ}(\Gamma_0)) - P(\mathbf{Succ}(\Gamma_1))| \leq \epsilon_{lpn}$, where ϵ_{lpn} is the advantage of $\mathcal{D}$ in breaking generalized LPN assumption.*

Now we consider the success event $\mathbf{Succ}(\Gamma_1)$. We assume $\mathcal{A}$ will not query $(p_1, \cdots, p_\ell, T)$ from a generation query output $(R, p_1, \cdots, p_\ell, T)$ to the reproduction oracle as $\mathcal{A}$ already knows the answer R. We also assume after outputting the forgery, attacker will terminate (as his success only depends on the validity of the forgery).

We modify Γ_1 to Γ_2 so that $\mathcal{A}$ is provided with $o = W' - W$, where W' is used for verification of the reproduction query and final forgery. It is immediate that $P(\mathbf{Succ}(\Gamma_1)) \leq P(\mathbf{Succ}(\Gamma_2))$. Now we focus on Γ_2. Consider the ith reproduction query $\{Z^{(i)}|A_i|p_i\}_{i=1}^{\ell}|T$. Let bit E_i be the decision bit for the reproduction query (0 for reject and 1 for success). We modify Γ_2 to Γ_3 such that if a reproduction query is valid, than $\mathcal{A}$ succeeds and in addition upon the first $E_i = 1$, then $\mathcal{D}$ terminates the simulation. Obviously, $P(\mathbf{Succ}(\Gamma_2)) \leq P(\mathbf{Succ}(\Gamma_3))$ (as $E_i = 1$ already implies the success of $\mathcal{A}$ and hence there is no need to continue the simulation). Let E_i^* be the event $E_i = 1$ while $E_j = 0$ for $j < i$. So $P(\mathbf{Succ}(\Gamma_3)) \leq \sum_{i=1}^{q_d} P(E_i^*)$. We first bound $P(E_1^*)$.

Let this first reproduction query be $C_1 = (A_1', p_1', \cdots, A_\ell', p_\ell', T')$, where $A_1', \cdots, A_\ell'$ are from public randomness r. Hence, $\hat{d}_1, \cdots, \hat{d}_\ell$ for this query are uniformly random (as we consider Γ_2). For simplicity, some generation query has generated ciphertext C_1' using the same sample set $A_1', \cdots, A_\ell'$ (otherwise, the proof will be similar and simpler as it can be regarded as attacker ignores the output from the previous generation query). Now since C_1 is different from the generation query output, it must have $p_i \neq p_i'$ for some i. We can rewrite $p_i' = p_i + (p_i + p_i') = \hat{d}_i + ECC(0^t|R|R_1) + (p_i' + p_i)$. At the receiver side, decoding p_i' using W' will give $(0^t|R|R_1) + DEC(p_i + p_i' + Z_i' \cdot \mathsf{Ext}(o[A_i], Z))$, as Ext is linear, where DEC is the decoding algorithm for ECC. If DEC outputs $\perp$, then the reproduction query will be rejected normally. So we consider the case that DEC output is not $\perp$. Let $\delta_1|\delta_2$ be the last 2λ bits of $DEC(p_i + p_i' + + Z_i' \cdot \mathsf{Ext}(o[A_i], Z))$, which is known to $\mathcal{A}$. Let $R_1 = x|y$. Then, $T = x^L + x^2 m(x) + xy$. If C_1 is verified by the verifier using the key $(x + \delta_1)|(y + \delta_2)$, decoded from p_i' using W', then $T' = (x+\delta_1)^L + (x+\delta_1)^2 m'(x+\delta_1) + (x+\delta_1)(y+\delta_2)$, with $L = \lceil \nu\ell/\lambda \rceil + 4$, where m' is the vector encoded from $H(p_1'|\cdots|p_\ell')$ which is different from m encoded from $H(p_1|\cdots|p_\ell)$ by the collision-resistancy of hash function H. Since now x, y are uniformly random λ bits (and independent of m as $x|y$ in p_i is masked by

one-time pad d_i which is used only in p_i), by Lemma 2, conditional on T, the probability that T' is valid (using the key in p'_i) is at most $2^{-\lambda}(L+1)$. Further, there are at most ℓ possible i. Thus, $P(E_1^*) \leq 2^{-\lambda}\ell(L+1)$.

Now we consider $P(E_k^*)$. To evaluate this, we consider a variant Γ_3' of Γ_3 where the tth reproduction query for $t < k$ is decided as reject (without verifying the tag T in its query). Let Σ be the randomness of Γ_3. Then, if Σ leads to reject for all previous $k-1$ reproduction query, then adversary views in Γ_3' and Γ_3 are identical; otherwise, some **reject** of some tth reproduction query is wrong. But in this case, E_k^* will not occur. It follows $P(E_k^*(\Gamma_3)) = P(E_k^*(\Gamma_3'))$. On the other hand, since the previous $k-1$ reproduction query always results in reject, it can be simulated by $\mathcal{A}$ himself. Hence, $P(E_k^*(\Gamma_3')) = P(E_1^*)$. It follows that $P(\mathbf{Succ}(\Gamma_3)) \leq q_d P(E_1^*) \leq q_d 2^{-\lambda}\ell(L+1)$. From Lemma 6 and $P(\mathbf{Succ}(\Gamma_1)) \leq P(\mathbf{Succ}(\Gamma_2)) \leq P(\mathbf{Succ}(\Gamma_3))$, it follows that $P(\mathbf{Succ}(\Gamma_0)) \leq \epsilon_{lpn} + q_d 2^{-\lambda}\ell(L+1)$. In this probability bound, we assume that no collision for H occurs in the attack, which is violated with probability at most ϵ_H. Hence, our result follows. □

D Proof of Theorem 2

Following Definition 2, we need to proof the ϵ-correctness and σ-security property of the fuzzy extractor, when there is no attack.

Reliability (correctness). Following the same argument as the proof of Theorem 1, we can prove that the probability that no v_i matches at the receiver is at most $(1-(1-\frac{t'}{n-m})^m)^\ell$, where $v_i = (W'_{j_{i,1}}, W'_{j_{i,2}}, \cdots, W'_{j_{i,m}})$, and $i \in \{1, 2, \cdots, \ell\}$.

In addition, *Rep* in figure 1 may return an incorrect extracted key due to an error in step (ii). This error might occur due to a collision that we now explain in detail. A collision at step (ii) may occur with probability $(\frac{1}{2^t})$, assuming E is implemented as a function from a strong universal hash family. Now, *Rep* may return an incorrect extracted key if there is at least one collision at step (ii). This may happen with any of the ℓ executions of step (ii) and hence with probability at most $\ell \cdot 2^{-t}$.

Since the fuzzy extractor's allowable error parameter is $\epsilon'(\lambda)$, we need to choose ℓ and m so that

$$(1-(1-\frac{t'}{n-m})^m)^\ell + \ell \cdot 2^{-t} \leq \epsilon'(\lambda). \tag{9}$$

Security. We now prove that it is an information-theoretic secure fuzzy extractor. We need to prove that $\Delta((R,P),(U_\xi;P)) \leq \sigma(\lambda)$, where P is the public strings.

In step (i) of Gen() algorithm described in figure 1, we sample a random subset $A_i = \{i_1, \cdots, i_m\}$ from $[n]$. In step (ii), E is implemented by H.

Since W is a (α, m, N)-sample source, we have $\tilde{H}_\infty(W[A] \mid W[B], A, B) \geq \alpha$, where A (resp. B) is a purely random subset of $[n]$ of size m (resp. N).

Since $H : \mathcal{X} \times \mathcal{S} \to \{0,1\}^{\nu}$ is a strong universal hash family, from lemma 1, if $\nu \leq \alpha + 2 - 2\log(\frac{2\ell}{\sigma})$, H is an $(m, \alpha, \nu, \frac{\sigma(\lambda)}{2\ell})$-extractor. Now, $\nu = \xi + t$. Thus, if $\xi \leq \alpha + 2 - 2\log(\frac{2\ell}{\sigma(\lambda)}) - t$, H is an $(m, \alpha, \nu, \frac{\sigma(\lambda)}{2\ell})$-extractor.

Since H is an $(m, \alpha, \nu, \frac{\sigma(\lambda)}{2\ell})$-extractor, from lemma 3, 4, if $\xi \leq \alpha + 2 - 2\log(\frac{2\ell}{\sigma(\lambda)}) - t$ and $\ell m < N$, considering $S = R$ in lemma 4, we have

$$\Delta(R, \mathbf{p}, Z, \mathbf{A}; U, \mathbf{p}, Z, \mathbf{A}) \leq \sigma(\lambda). \tag{10}$$

Therefore, (Gen, Rep) is a $(\mathcal{V}^n, \mathcal{W}, \xi, t', \epsilon', \sigma)$-fuzzy extractor, where $(1-(1-\frac{t'}{n-m})^m)^{\ell} + \ell \cdot 2^{-t} \leq \epsilon'(\lambda)$, $\ell m < N$ and $\mathcal{V} \in \{0,1\}$. Furthermore, the length of the extracted key will be
$\xi \leq \alpha + 2 - 2\log(\frac{2\ell}{\sigma(\lambda)}) - t$.

Reusability. In response to a query to Gen oracle (i.e. generation oracle query), the oracle returns a pair of key r^i and ciphertext c^i to the adversary, where $r^i = R^i$ and $c^i = (p_1|A_1, \cdots, p_{\ell}|A_{\ell})^i$ according to the $Gen(W^i)$ procedure described in figure 1. Now since W is a source with (α, m, N)-samples, in each query to generation oracle, $Gen(W^i)$ procedure runs with new samples with conditional entropy α. Hence, the uncertainty about the new samples remains the same before and after η queries to the generation oracle from adversary's perspective. Therefore, the entropy of the new samples remains same. Now proceeding in similar manner as the proof of *security* , we can prove that, if
$\xi \leq \alpha + 2 - 2 \cdot \log(\frac{2\ell}{\sigma}) - t$, then the (Gen, Rep) described in figure 1 is (η, σ)-reusable $(\mathcal{V}^n, \mathcal{W}, \xi, t', \epsilon', \sigma)$-fuzzy extractor, where
$(1 - (1 - \frac{t'}{n-m})^m)^l + l \cdot 2^{-t} \leq \epsilon'(\lambda)$ and $\ell\eta m < N$. □

E Proof of Theorem 3

Correctness. We first consider correctness error. Then we prove the security. Proceeding the same way as the proof of Theorem 1, we can prove that the probability that no v_i matches at the receiver is at most $(1 - (1 - \frac{t'}{n-m})^m)^{\ell}$, where $v_i = (W'_{j_{i,1}}, W'_{j_{i,2}}, \cdots, W'_{j_{i,m}})$, and $i \in \{1, 2, \cdots, \ell\}$.

In addition, *Rep* may be incorrect due to an error in step (ii) and verification of $T' = T$. These errors might occur due to collision. A collision at step (ii) might occur with probability $(\frac{1}{2^t})$, assuming E is implemented as a function from a strong universal hash family. In addition, similar to lemma 2 (except $\delta_1|\delta_2 = (d_i \oplus d'_i)_{\nu-2\lambda+1\cdots\nu}$ with d_i from W and d'_i from W'), a collision $T' = T$ occurs with probability $L2^{-\lambda}$. Now, *Rep* may be incorrect if there is at least one collision at step (ii) together with a collision $T' = T$. This may happen with any of the ℓ iterations and hence with probability at most $\ell \cdot 2^{-t} \cdot L \cdot 2^{-\lambda}$.

Since the fuzzy extractor's allowable error parameter is $\epsilon'(\lambda)$, we need to choose ℓ and m so that

$$(1 - (1 - \frac{t'}{n-m})^m)^{\ell} + \ell \cdot 2^{-t} \cdot L \cdot 2^{-\lambda} \leq \epsilon'(\lambda). \tag{11}$$

Security. We now prove the security. We need to show that $\Delta((R,P),(U_\xi;P)) \leq \sigma(\lambda)$, where P is the public strings. In step (i) of Gen algorithm of Figure 2 , a random subset $A_i = \{i_1, \cdots, i_m\}$ is sampled from $[n]$, and in step (ii), E is implemented by H. Note that W is a (α, m, N)-sample source. Hence, we have $\tilde{H}_\infty(W[A] \mid W[B], A, B) \geq \alpha$, where A (resp. B) is a purely random subset of $[n]$ of size m (resp. N).

Since $H : \mathcal{X} \times \mathcal{S} \to \{0,1\}^\nu$ is a strong universal hash family, from lemma 1, if $\nu \leq \alpha + 2 - 2\log(\frac{4\ell}{\sigma})$, H is an $(m, \alpha, \nu, \frac{\sigma(\lambda)}{4\ell})$-extractor. Now, $\nu = 2\lambda + \xi + t$. Thus, if $\xi \leq \alpha + 2 - 2\log(\frac{4\ell}{\sigma(\lambda)}) - t - 2\lambda$, H is an $(m, \alpha, \nu, \frac{\sigma(\lambda)}{2\ell})$-extractor.

Since H is an $(m, \alpha, \nu, \frac{\sigma(\lambda)}{4\ell}$-extractor, from lemma 3, 4 and 5, if $\xi \leq \alpha + 2 - 2\log(\frac{4\ell}{\sigma(\lambda)}) - t - 2\lambda$ and $\ell m < N$, we have

$$\Delta(R, \mathbf{p}, Z, \mathbf{A}, T; U, \mathbf{p}, Z, \mathbf{A}, T) \leq \sigma(\lambda). \tag{12}$$

Therefore, (Gen, Rep) is a $(\mathcal{V}^n, \mathcal{W}, \xi, t', \epsilon', \sigma)$-fuzzy extractor, where $(1-(1-\frac{t'}{n-m})^m)^\ell + \ell \cdot 2^{-t} \cdot L \cdot 2^{-\lambda} \leq \epsilon'(\lambda)$, $\ell m < N$ and $\mathcal{V} \in \{0,1\}$. Furthermore, the length of the extracted key will be $\xi \leq \alpha + 2 - 2\log(\frac{4\ell}{\sigma(\lambda)}) - t - 2\lambda$.

Reusability. The proof follows similar arguments of *reusability* part of the proof of Theorem 2 and the *security* arguments described above. In response to a query to Gen oracle, the oracle returns a pair of key r^i and ciphertext c^i to the adversary, where $r^i = R^i$ and $c = (p_1, \cdots, p_\ell, T)$ according to the $Gen(W^i)$ procedure described in Figure 2. As W is a source with (α, m, N)-samples, in each Gen oracle query, $Gen(W^i)$ procedure runs with new samples with conditional entropy α. Hence, the uncertainty about the new samples remains the same before and after η Gen oracle (or generation oracle) queries from the adversary's perspective. Therefore, the entropy of each new samples remains same. Now proceeding in similar manner as the proof of Theorem 3, we can prove that, if $\xi \leq \alpha + 2 - 2 \cdot \log(\frac{4 \cdot \ell}{\sigma(\lambda)}) - t - 2\lambda$, then the (Gen, Rep) described in Figure 2 is (η, σ)-reusable $(\mathcal{V}^n, \mathcal{W}, \xi, t', \epsilon', \sigma)$-fuzzy extractor, where $(1-(1-\frac{t'}{n-m})^m)^l + l \cdot 2^{-t} \cdot L \cdot 2^{-\lambda} \leq \epsilon'(\lambda)$ and $\ell\eta m < N$.

Robustness. Robustness proof is given in Appendix F □

F Robustness Proof for Theorem 3

We need to prove our scheme satisfies Definition 4. We show that through q_e generation queries and q_d reproduction queries, the robustness is broken negligibly. Assume $\mathcal{A}$ is the robustness attacker. Upon generation and reproduction queries, challenger acts normally. Denote this game Γ_0. We now revise Γ_0 to Γ_1 so that $d_i = E(W[A_i], Z)$ is replaced by $d_i \leftarrow \{0,1\}^\nu$ in generation query or reproduction query (for some A_i that does not appear in a previous query; otherwise, use the existing d_i). Let $\mathbf{Succ}(\Gamma)$ be the success event of $\mathcal{A}$ in game Γ. Let d_i be taken from $d_i = E(W[A_i], Z)$ or $d_i \leftarrow \{0,1\}^\nu$, for all i. Then, we consider a distinguisher that distinguishes Γ_0 and Γ_1. Upon receiving d_i that is either $d_i = E(W[A_i], Z)$ or uniformly random, challenger prepares the public

sampling randomness r that results in sampling A_i's for all i's in the challenge. Now challenger simulates Γ_0 with $\mathcal{A}$ against it by providing $\mathcal{A}$ with Z and public randomness r, except all d_i's are from his challenge tuple. Finally, if $\mathcal{A}$ succeeds, output 0; otherwise, output 1. The simulated game is a randomized function with input $\mathbf{d}_1, \cdots, \mathbf{d}_{q_d+q_e}$ and a binary output 0 or 1, where $\mathbf{d}_i$ is the vector of $(d_1, \cdots, d_\ell)$ in the generation or a reproduction query (with new A_i's). Denote this function by $G(\mathbf{d})$. Since $A_1, \cdots, A_\ell$ by challenger or adversary is from public random string, $W[A_i]$ follows the distribution of (α, m, N)-source W. By claim 1 and Lemma 3, we immediately have the following.

Lemma 7. $|P(\mathbf{Succ}(\Gamma_0)) - P(\mathbf{Succ}(\Gamma_1))| \leq (q_d + q_e)\ell\epsilon$.

Now we consider the success event $\mathbf{Succ}(\Gamma_1)$. We assume $\mathcal{A}$ will not query the output of generation query to the reproduction oracle as $\mathcal{A}$ already knows the answer. Consider the ith reproduction query $\{A_i|p_i\}_{i=1}^{\ell}|T$. Let bit E_i be the decision bit for the reproduction query (0 for reject and 1 for success). We modify Γ_1 to Γ_2 such that if upon the first $E_i = 1$, then stop the security game. Obviously, $P(\mathbf{Succ}(\Gamma_1)) = P(\mathbf{Succ}(\Gamma_2))$. Let E_i^* be the event $E_i = 1$ while $E_j = 0$ for $j < i$. So $P(\mathbf{Succ}(\Gamma_2)) \leq \sum_{i=1}^{q_d} P(E_i^*)$. We first bound $P(E_1^*)$.

Let this first reproduction query be $C_1 = (A_1', p_1', \cdots, A_\ell', p_\ell', T')$, where $A_1', \cdots, A_\ell'$ are from public randomness r (note: they could be previously sampled in the generation query processing). Hence, $d_1, \cdots, d_\ell$ for this query are uniformly random (as we consider Γ_2). For simplicity, some generation query has generated ciphertext C_1' using the same sample set $A_1', \cdots, A_\ell'$ (otherwise, the proof will be similar and simpler). Assume $d_1, \cdots, d_\ell$ generated $p_1, \cdots, p_\ell, T$ for that generation query. We bound the probability that T' generated by $\mathcal{A}$ is valid. Notice that each generation query uses independent d_i's and hence can be simulated by $\mathcal{A}$. So we can assume that $\mathcal{A}$ only issue one generation query (which uses $A_1', \cdots, A_\ell'$). Denote this by Γ_2^1. Let the hash output be T for the generation query. Since the reproduction query can not use the same ciphertext, assume $p_i' \neq p_i$. Hence, let $x|y = R_1$ in p_i and let $\delta_1|\delta_2$ be the last 2λ bits in $p_i \oplus p_i' \oplus \mathsf{Ext}(o[A_i], A)$, where $o = W' - W$ is assumed to be known to $\mathcal{A}$ (this will only increase the success probability of $\mathcal{A}$). Then, the decrypted tag key from C_1 using W' at verifier will be
$(x + \delta_1)|(y + \delta_2)$. Hence, if T' is valid, we know that $T = x^L + x^2 m(x) + xy$ and $T' = (x+\delta_1)^L + (x+\delta_1)^2 m'(x+\delta_1) + (x+\delta_1)(y+\delta_2)$, with $L = \lceil \nu\ell/\lambda \rceil + 4$. Since now x, y are uniformly random λ bits and are independent of $(A_1|p_1| \cdots |p_\ell|A_\ell)$ due to the one-time pads $d_1, \cdots, d_\ell$) (hence idependent of the encoded vector $\mathbf{m}$), by Lemma 2, conditional on T, the probability that T' is valid is at most $2^{-\lambda}(L+1)$. Further, there are at most ℓ possible i. Thus, $P(E_1^*) \leq 2^{-\lambda}\ell(L+1)$.

Now we consider $P(E_k^*)$. To evaluate this, we consider a variant Γ_2^3 of Γ_2 where the tth reprodution query for $t < k$ is decided as reject (without verifying the tag T in its query). Let Σ be the randomness of Γ_2. Then, if Σ leads to reject for all previous $k-1$ reproduction query, then adversary view in Γ_2^3 and Γ_2 is identical; otherwise, some **reject** of some tth reproduction query is wrong. But in this case, E_k^* will not occur. It follows $P(E_k^*(\Gamma_2)) = P(E_k^*(\Gamma_2^3))$. On the

other hand, since the previous $k-1$ reproduction query always results in reject, it can be simulated by $\mathcal{A}$ himself. Hence, $P(E_k^*(\Gamma_2^3)) = P(E_1^*)$. It follows that $P(\mathbf{Succ}(\Gamma_2)) \leq q_d P(E_1^*) \leq q_d 2^{-\lambda}\ell(L+1)$. From Lemma 7 and $P(\mathbf{Succ}(\Gamma_1)) = P(\mathbf{Succ}(\Gamma_2))$, it follows that $P(\mathbf{Succ}(\Gamma_0)) \leq (q_d+q_e)\ell\epsilon + q_d 2^{-\lambda}\ell(L+1)$. □

G Proofs of Basic Results in Section 3

Proof of Claim 1. By calculation, we have

$$\begin{aligned}
&\Delta(F(X);F(Y))\\
=&\tfrac{1}{2}\textstyle\sum_{v\in\mathcal{V}} |\Pr[F(X)=v]-\Pr[F(Y)=v]|\\
=&\tfrac{1}{2}\textstyle\sum_{v\in\mathcal{V}} |\sum_{u:F(u)=v}(\Pr[X=u]-\Pr[Y=u])|\\
\leq&\tfrac{1}{2}\textstyle\sum_{v\in\mathcal{V}}\sum_{u:F(u)=v} |\Pr[X=u]-\Pr[Y=u]|\\
=&\Delta(X;Y).
\end{aligned}$$

□

Proof of Lemma 3. Use induction. When $\mu=1$, it holds from assumption and Claim 1. If it holds for $\mu=k-1$, consider case $\mu=k$. By assumption, $E(\tilde{U},Z)=U$. Note that $d_i = E(W[A_i],Z)$. We have

$$\begin{aligned}
&\Delta(\mathbf{d},Z,\mathbf{A};U^k,Z,\mathbf{A})\\
\leq&\Delta(d_k,\mathbf{d}^{k-1},Z,\mathbf{A};U,\mathbf{d}^{k-1},Z,\mathbf{A})+\Delta(U,\mathbf{d}^{k-1},Z,\mathbf{A};U,U^{k-1},Z,\mathbf{A})\\
\leq&\Delta(d_k,\{W[A_i]\}_1^{k-1},Z,\mathbf{A};U,\{W[A_i]\}_1^{k-1},Z,\mathbf{A})+\\
&\Delta(\mathbf{d}^{k-1},Z,\mathbf{A};U^{k-1},Z,\mathbf{A})\\
\leq&\epsilon+(k-1)\epsilon=k\epsilon,
\end{aligned}$$

where the 2nd inequality follows from Claim 1; the first part of the last inequality follows from the definition of E and $\tilde{H}_\infty(W[A_k] \mid \{W[A_i]\}_1^{k-1},\mathbf{A}) \geq \alpha$ (from definition of W); the second part follows from the induction assumption with the fact that A_k is independent of the remaining random variable. □

Proof of Lemma 4. Let $d_i = E(W[A_i],Z)$. Denote $d_i = X_i|Y_i$ with X_i the first t bits and Y_i the remaining bits of d_i. Let U_1,U_2 be uniformly random variables in the domain of X_1,Y_1 respectively. Let $\mathbf{U}_1,\mathbf{U}_2$ be ℓ iid copies of U_1,U_2 respectively. For vector $\mathbf{v}$ and variable α, $\mathbf{v}\oplus\alpha$ denotes $(v_1\oplus\alpha,\cdots,v_\ell\oplus\alpha)$. For simplicity, let $C=(Z,\mathbf{A})$.

$$\begin{aligned}
&\Delta(S,\mathbf{p},C;U,\mathbf{p},C)\\
&=\Delta(S,\mathbf{d},C;U,\mathbf{d}',C),\ \text{where } \mathbf{d}'=\mathbf{d}\oplus 0^t|S\oplus 0^t|U\\
&=\Delta(P_{S\mathbf{d}C};P_U P_{\mathbf{d}'C})\\
&=\Delta(P_{\mathbf{d}C};P_{\mathbf{d}'C}),\ \text{(as } S \text{ is ind of } (\mathbf{d},C) \text{ and distributed as } U)\\
&=\Delta(P_{\mathbf{X}\mathbf{Y}C};P_{\mathbf{X},\mathbf{Y}\oplus V,C}),\ (\text{let } V\stackrel{def}{=}U\oplus S))\\
&\leq\Delta(P_{\mathbf{X}\mathbf{Y}C};P_{\mathbf{U}_1\mathbf{U}_2C})+\Delta(P_{\mathbf{U}_1\mathbf{U}_2C};P_{\mathbf{X},\mathbf{Y}\oplus V,C})\\
&\leq\ell\epsilon+\Delta(P_{\mathbf{U}_1\mathbf{U}_2C};P_{\mathbf{X},\mathbf{Y}\oplus V,C}),\ \text{(Lemma 3)}\\
&\leq\ell\epsilon+\Delta(P_{\mathbf{U}_1\mathbf{U}_2CV};P_{\mathbf{X},\mathbf{Y}\oplus V,C,V})
\end{aligned}$$

$$
\begin{aligned}
&= \ell\epsilon + \Delta(P_{\mathbf{U_1},\mathbf{U_2}-V,C,V}; P_{\mathbf{X},\mathbf{Y},C,V}) \\
&= \ell\epsilon + \Delta(P_{\mathbf{U_1}\mathbf{U_2}CV}; P_{\mathbf{XY}CV}) \\
&\quad (\mathbf{U}_2 \text{ and } \mathbf{U}_2 - V \text{ are identical, ind of the remaining variables}) \\
&= \ell\epsilon + \Delta(P_{\mathbf{U_1}\mathbf{U_2}C}; P_{\mathbf{XY}C}),\ (V \text{ is ind of the remaining variables}) \\
&\leq 2\ell\epsilon,\ (\text{Lemma 3})\ \square
\end{aligned}
$$

Proof of Lemma 5. In Lemma 4, $S = R|R_1$ in our scheme. Thus,

$$\Delta(R|R_1, \mathbf{p}, Z, \mathbf{A}; U|U_1, \mathbf{p}, Z, \mathbf{A}) \leq 2\ell\epsilon,$$

where U (resp. U_1) is uniform ξ-bit (resp. 2λ-bit). Let $C = (\mathbf{p}, Z, \mathbf{A})$. By claim 1, let $F(R|R_1, C) = (R, C, T)$. Then, $\Delta(R, C, T; U, C, U') \leq 2\ell\epsilon$, where $U' = tag(U_1, C)$ and $tag(\cdot)$ is the tag algorithm in our scheme used to compute T. Notice that

$$
\begin{aligned}
&\Delta(R, C, T; U, C, T) \\
\leq &\Delta(R, C, T; U, C, U') + \Delta(U, C, U'; U, C, T) \\
\leq &\Delta(R|R_1, C; U|U_1, C) + \Delta(U, C, U'; U, C, T),\ (\text{Claim 1}) \\
\leq &2\ell\epsilon + \Delta(C, U'; C, T),\ (\text{Lemma 4; } U \text{ is ind of } \mathbf{p} \text{ and } T, U') \\
\leq &2\ell\epsilon + \Delta(U_1, C; R_1, C), \quad (\text{Claim 1}) \\
\leq &4\ell\epsilon, \quad (\text{Claim 1 and Lemma 4})
\end{aligned}
$$

This completes our proof. $\square$

H LDPC Experiments

In this section we discuss the experimental results for LDPC code.

H.1 Commands and Parameters

We ran these experiments on the 5G standard matrices[26]. The matrices were converted using a python script `convert.py` (included with this report).

Errors were introduced using the provided `transmit` function. We simulated a binary symmetric channel (with `bsc` command) with error rates $0.11, 0.12, 0.125$, and 0.13. Each round of testing started with creating a random source - 2000 blocks of messages with block size of either 396, 192, 132, or 110. Afterwards, we encoded the messages, introduced errors and attempted to decode. Decoding was done using the probability propagation method with at most 250 iterations. Table 2 presents our results.

H.2 Experimental results for 5G matrices

We ran the experiment for the matrices `NR_1_4_18` `NR_1_4_9`, and `NR_1_1_6` [26]. The parity check matrices are presented in order of success rate.

Name	Code	Noise	Success Rate	Enc Time	Dec Time
NR_1_4_18	(1228, 396)	0.11	2000/2000	0.175 sec	1.5 sec.
		0.13	1989/2000		2 sec.
		0.15	1598/2000		9.72 sec.
NR_1_4_9	(612, 198)	0.12	1989/2000	0.090 sec	0.729 sec.
		0.125	1964/2000		1.019 sec.
		0.13	1906/2000		1.443 sec.
NR_1_1_6	(408, 132)	0.12	1943/2000	0.102 sec	0.683 sec.
		0.125	1888/2000		0.930 sec.
		0.13	1838/2000		1.282 sec.

Table 2. Experimental results for some LDPC code with different noise rate