Computers, Materials & Continua
DOI: 10.32604/cmc.2023.033102
Article

Tech Science Press

Robust and Reusable Fuzzy Extractors from Non-Uniform Learning with Errors Problem

Joo Woo[1], Jonghyun Kim[1] and Jong Hwan Park[2,*]

[1]Graduate School of Information Security, Korea University, Seoul, 02841, Korea
[2]Department of Computer Science, Sangmyung University, Seoul, 03016, Korea
*Corresponding Author: Jong Hwan Park. Email: jhpark@smu.ac.kr
Received: 07 June 2022; Accepted: 12 July 2022

Abstract: A fuzzy extractor can extract an almost uniform random string from a noisy source with enough entropy such as biometric data. To reproduce an identical key from repeated readings of biometric data, the fuzzy extractor generates a helper data and a random string from biometric data and uses the helper data to reproduce the random string from the second reading. In 2013, Fuller et al. proposed a computational fuzzy extractor based on the learning with errors problem. Their construction, however, can tolerate a sub-linear fraction of errors and has an inefficient decoding algorithm, which causes the reproducing time to increase significantly. In 2016, Canetti et al. proposed a fuzzy extractor with inputs from low-entropy distributions based on a strong primitive, which is called digital locker. However, their construction necessitates an excessive amount of storage space for the helper data, which is stored in authentication server. Based on these observations, we propose a new efficient computational fuzzy extractor with small size of helper data. Our scheme supports reusability and robustness, which are security notions that must be satisfied in order to use a fuzzy extractor as a secure authentication method in real life. Also, it conceals no information about the biometric data and thanks to the new decoding algorithm can tolerate linear errors. Based on the non-uniform learning with errors problem, we present a formal security proof for the proposed fuzzy extractor. Furthermore, we analyze the performance of our fuzzy extractor scheme and provide parameter sets that meet the security requirements. As a result of our implementation and analysis, we show that our scheme outperforms previous fuzzy extractor schemes in terms of the efficiency of the generation and reproduction algorithms, as well as the size of helper data.

Keywords: Fuzzy extractor; reusability; robustness; biometric authentication; non-uniform learning with errors

1 Introduction

Authentication necessitates the use of a secret derived from some source with sufficient entropy. Because of their uniqueness and usability, biometric information such as fingerprints, iris patterns, and facial features can be a promising candidate, as can physical unclonable functions and quantum sources. Biometric authentication, in particular, makes a user's life much easier because biometric information does not require a user to memorize or securely store anything in order to authenticate. However, two challenges are found to using biometric data. First, because biometric information is immutable, it is difficult to change once it has been leaked to an adversary. As a result, provable cryptographic security is critical for biometric storage systems. Second, whenever biometric data are generated, small errors occur as a result of the various conditions and environments. In other words, the biometric data provide similar but not identical readings at each measurement.

Since Dodis et al. [1] proposed the concept of a fuzzy extractor to address these issues, it has been regarded as one of the candidate solutions for key management using biometric data. A fuzzy extractor can extract the same random string from a noisy source without storing the source itself. A fuzzy extractor is made up of two algorithms (**Gen**, **Rep**). The generation algorithm **Gen** takes as input w (= an initial reading of the noisy source) and outputs an extracted string R and a helper data P. The reproduction algorithm **Rep** will reproduce R from w' (= a second reading of the same source) with the help of P if w and w' are close enough, i.e., the distance between w and w' is small enough. The correctness of fuzzy extractor guarantees that the same string R is reproduced if the fuzzy data are generated from the same source. Furthermore, the security of the fuzzy extractor guarantees that the helper data does not reveal any information about the biometric data.

The majority of the fuzzy extractors are built on the sketch-then-extract paradigm, with a secure sketch and a randomness extractor as building blocks [2–5]. A secure sketch is an information-reconciliation protocol, which can recover the original value w from w' if w and w' are close enough, and then, the random string will be extracted from the recovered w using a randomness extractor. However, secure sketch has been shown to cause a small leakage of biometric data from helper data, resulting in security loss [6]. As a result, a fuzzy extractor based on secure sketch requires high entropy source data. Fuller stated in [6] that constructing secure sketches with computational unpredictability of w given helper data is still an open problem.

Fuller et al. [6] proposed a computational fuzzy extractor based on the learning with errors (LWE) problem that does not use a secure sketch nor a randomness extractor to avoid the negative result for secure sketches. The random string is not extracted from w in Fuller's computational fuzzy extractor scheme; rather, w "encrypts" the random string in a way that can be decrypted with the knowledge of some close w'. Their approach prevents biometric data leakage from the helper data because the helper data hides secret randomness (the random string) using w, rather than extracting it from w. However, their construction is significantly inefficient and can only tolerate sub-linear errors due to the construction of the decoding algorithm during the reproduction phase, which is limited to guessing a subset of error locations and checking if the chosen part contains any errors. However, in order to be widely used in practice, a fuzzy extractor must be efficient in terms of costing time including generation algorithm and reproduction algorithm and be able to tolerate more than a certain level of error. Some biometric data, such as iris codes, exhibit linear errors. Repeated readings of the same iris demonstrate an average error rate of 10%–30% [7] for the iris code, which is proposed to represent human iris features [8].

Moreover, Fuller's construction does not guarantee reusability or robustness. Boyen [2] formalized the concept of reusability, which is the security notion in the case that several pairs of extracted string and related helper data issued from correlated biometric data are revealed to an adversary. For multiple correlated samples $(w, w_1, \ldots, w_t)$ of the same source, reusability guarantees the (pseudo) randomness of R conditioned on $(P, \{P_i, R_i\}_{i=1,\ldots,t})$ where $(P_i, R_i) \leftarrow \mathbf{Rep}(w_i)$. Robustness is the security notion in the case that an adversary modifies the helper data P before P is given to the user. A robust fuzzy extractor has been studied in order to protect against malicious alteration of helper data and to support mutual authentication between a user and a server. Robustness requires that any modification of P by an adversary will be detected. Boyen et al. [9] introduced the concept of robust fuzzy extractor, and Dodis et al. [10] strengthened robustness, which guarantees that the fuzzy extractor will detect any modification of P by an adversary who also has access to R. Both security concepts must be satisfied in order to use a fuzzy extractor as a secure authentication method in real life.

Canetti et al. [11] proposed a fuzzy extractor construction with inputs from low-entropy distributions that relies on a strong cryptographic primitive called digital locker to avoid the negative result for secure sketches as well. Their strategy is to sample many partial strings from a noisy input string and then independently lock one secret key with each partial string. This sample-then-lock method, however, necessitates an excessive amount of storage space for the helper value in order to store the hashed values of multiple subsets of the noisy data. To be widely used in practice, a fuzzy extractor must have a small size of helper data, which is stored in a server for authentication. Cheon et al. [12] later tweaked this scheme to reduce the size of the helper data. However, the computational cost of the reproduction algorithm significantly increased.

1.1 Contribution

We propose a new computational fuzzy extractor in this paper that does not rely on a secure sketch or digital locker. As a result, our scheme is efficient, and thanks to the new decoding algorithm, it can tolerate linear errors. To address the error caused by the difference between the biometric data used for registration and the biometric data used for authentication, we encode the extracted key using two cryptographic primitives: error correction code (ECC) and the EMBLEM encoding method [13]. We eliminated the recursive decoding algorithm from the previous fuzzy extractor [6], which improved the efficiency of our scheme. Unlike Fuller's scheme, the decoding time for our scheme does not increase when the error rates increase. These points result to increased efficiency for the reproduction algorithm and error tolerance for the biometric data. For example, the reproduce algorithm for our scheme including decoding algorithm takes only 0.28 ms for 80-bit security with 30% error tolerance rates, whereas Fuller's reproduce algorithm takes more than 30 s with 1.3% error tolerance rates [14]. Meanwhile, Canetti's scheme [11], which is based on digital locker, requires a large size of helper data in order to store the hashed values of multiple subsets of the biometric data. For example, the helper data of their construction require approximately 33 GB of storage for 128-bit security with 19.5% error tolerance rates [13]. Meanwhile, the helper data for our construction require only 543 bytes for 128-bit security with 20% error tolerance rates. As a result, we demonstrate that our scheme is much more efficient because it has an efficient decoding algorithm compared to Fuller's fuzzy extractor and has much smaller size of helper data compared to Canetti's fuzzy extractor.

In addition, we extend our fuzzy extractor to robust and reusable. Both security concepts must be satisfied in order to use a fuzzy extractor as a secure authentication method in real life. Moreover, we present formal security proof for the proposed fuzzy extractor based on the non-uniform LWE (NLWE) problem [15] as well as concrete bit hardness for our scheme using a LWE-estimator. Furthermore, we provide parameter sets that meet the security requirements and present the performance of our fuzzy extractor's implementation. As a result, in comparison to previous fuzzy extractor schemes, we build the computational fuzzy extractor that supports a much more efficient implementation with a smaller size of helper data, compared to previous fuzzy extractor schemes.

To ensure the security of our fuzzy extractor, we must assume that the biometric data are drawn from a uniform distribution. Also, we are well aware that in reality, many fuzzy sources, including biometric data, do not provide uniform distributions. The purpose of this paper is to inspire researchers to create an efficient computational fuzzy extractor based on our construction.

1.2 Related Works

Dodis et al. introduced the fuzzy extractor in 2004, which generates a cryptographically secure key using the user's biometric data and is based on secure sketch [1].

Reusable fuzzy extractor: In 2004, Boneh et al. defined a reusable fuzzy extractor based on secure sketch and demonstrated that informational theoretic reusability necessitates a significant decrease in security, implying that the security loss is necessary [2]. In 2016, Canetti et al. [11] constructed a reusable fuzzy extractor based on the powerful tool "digital locker," which is a symmetric key cryptographic primitive. The concept of the scheme is sample-then-lock, which hashes the biometric in an error-tolerant manner. In particular, multiple random subsets of the biometric data are hashed, and then, the values are used as a pad for a (extracted) cryptographic key. Following the second measurement, the cryptographic key is determined by hashing multiple random subsets of the biometric data. Correctness follows if it is likely that the first measurement and the second measurement is identical in at least one subset of biometric data. This scheme based on a digital locker requires a large size of helper data to store the hashed values of multiple subsets of the biometric data. The size of helper data of Canetti's construction is too large to use the fuzzy extractor practically and the performance of reproduction algorithm is inefficient. To reduce the size of the helper data, Cheon et al. [12] modified Canetti's scheme in 2018. However, the computational cost of the reproduction algorithm even increased.

In 2017, Apon et al. [16] improved Fuller's construction [6] to satisfy the reusability requirement using either a random oracle or a lattice-based symmetric encryption technique. However, it falls short of overcoming Fuller's construction limitations: logarithmic fraction of error tolerance due to the time-consuming process of reproduction algorithm even with a small number of errors. Wen et al. [3] proposed a reusable fuzzy extractor based on the LWE problem in 2018, and their scheme is resilient to linear fractions of errors. However, their scheme is based on a secure sketch, and this point results in a leakage of biometric information.

Robust fuzzy extractor: In 2004, Boyen et al. [9] introduced the concept of robustness, and they proposed a general method for converting a fuzzy extract to a robust one using a secure sketch in a random oracle model. Dodis et al. [10] extended post-application robustness by requiring that any modification of P by an adversary who also sees R be detected. Since then, many works introduced information-theoretic fuzzy extractors with robustness [17,18].

Robust and reusable fuzzy extractor: In 2018, Wen et al. [5] defined the concept of robust and reusable fuzzy extractor in the common reference string model based on the decisional diffie-hellman (DDH) and the decision linear (DLIN) assumptions in the standard model, which is not post-quantum resistant. Traditional cryptography relies on the hardness of number-theoretic problems like integer factorization and discrete logarithm to ensure its security. With the advent of quantum computers, researchers demonstrated that quantum algorithms can solve problems such as DDH and DLIN in polynomial time. Also, Wen's scheme is based on a secure sketch. Later, Wen et al. [19] proposed two frameworks for developing robust and reusable fuzzy extractors and presented two instantiations in

2019, one of which is an LWE-based one that supports a sub-linear fraction of errors. The second is a DDH-based one that does not support linear fraction of errors and is not quantum resistant. Moreover, their scheme employs a secure sketch, resulting in a leakage of biometric information, a biometric source with high entropy is required.

1.3 Outline

The remainder of this paper is organized as follows: In Section 2, we introduce notation, mathematical problems, and the error correcting mechanism on which our scheme is based on. We briefly introduce Fuller's fuzzy extractor [6] in Section 2 as well. Then, in Section 3, we define fuzzy extractor and their security models including reusability and robustness. In Section 4, we present our reusable fuzzy extractor and demonstrate how to prove its security, such as reusability. In Section 5, we present our reusable fuzzy extractor with robustness and demonstrate how to prove its robustness. We discuss security requirements and concrete parameters for our construction in Section 6. In Section 7, we present the result as well as performance of implementation and we conclude this paper in Section 8.

2 Preliminaries

2.1 Notation

Let $\mathbb{Z}_q$ denote the quotient ring of integers modulo q. Vectors with entries in $\mathbb{Z}_q$ are referred to as column vectors and are denoted by lowercase letters, such as., x, y. Matrices with entries in $\mathbb{Z}_q$ are denoted by bold capital letters, such as $\boldsymbol{A}, \boldsymbol{B}$.

For finite set S, we denote sampling the element s uniformly from S with $s \leftarrow_r U(S)$ or simply $s \leftarrow_r S$. Let $\mathcal{X}$ be a distribution over $\mathbb{Z}$, then we write $x \leftarrow \mathcal{X}$ if x is sampled in accordance with $\mathcal{X}$. Furthermore, we denote sampling each coordinate of a matrix $\boldsymbol{A} \in \mathbb{Z}^{m\times n}$ with distribution $\mathcal{X}$ by $\boldsymbol{A} \leftarrow \mathcal{X}^{m\times n}$ with $m, n \in \mathbb{Z}_{>0}$. For an algorithm $\mathcal{A}$, the value $y \leftarrow \mathcal{A}(x)$ denotes the output of $\mathcal{A}$ on input x; if $\mathcal{A}$ employs randomness, then $\mathcal{A}(x)$ is a random variable. The min-entropy of X is $H_\infty(X) = -\log(\max_x \Pr[X = x])$ where X is a random variable, and x is a particular outcome.

2.2 LWE Problem [20]

The LWE and NLWE problems are defined here. The LWE problem was defined in [20], and the error distribution is typically a discrete Gaussian distribution. The hardness of the decisional LWE problem is guaranteed by the worst-cased hardness of the decisional version of the shortest vector problem (GapSVP).

For integers n, m, q and distributions $\mathcal{D}_s, \mathcal{D}_e$ over $\mathbb{Z}_q$, given an instance $(\boldsymbol{A}, b) \in \mathbb{Z}_q^{m\times n} \times \mathbb{Z}_q^m$, the decisional LWE problem $(\mathbb{Z}_q, n, m, \mathcal{D}_s, D_e)$-LWE is to determine if $s \leftarrow \mathcal{D}_s^n$ exists such that the instance is of the form $(\boldsymbol{A}, b = \boldsymbol{A}s + e)$ with $\boldsymbol{A} \leftarrow_R \mathbb{Z}_q^{n\times m}, e \leftarrow D_e^m$ or the instance is of the form $(\boldsymbol{A}, u)$ sampled from $\mathbb{Z}_q^{m\times n} \times \mathbb{Z}_q^m$ uniformly.

The advantage of an adversary $\mathcal{A}$ in solving decisional LWE problem is defined as follows:

$$Adv_{LWE}^{q,n,m,D_s,D_e}(\mathcal{A}) = |\Pr[\mathcal{A}(\boldsymbol{A}, b) = 1] - \Pr[\mathcal{A}(\boldsymbol{A}, u) = 1]|. \quad (1)$$

2.3 NLWE Problem [15]

For integers n, m, q, an error distribution $\mathcal{X}$ over $\mathbb{Z}_q$, and a distribution η over $\mathbb{Z}_q$ given an instance $(\boldsymbol{A}, b) \in \eta^{m\times n} \times \mathbb{Z}_q^m$, the non-uniform LWE problem $(\mathbb{Z}_q, n, m, \mathcal{X}, \eta)$-NLWE is to determine if $s \leftarrow \mathbb{Z}_q^n$

exists such that the instance is of the form $(\boldsymbol{A}, b = \boldsymbol{A}s + e)$ with $\boldsymbol{A} \leftarrow_R \eta^{m\times n}, e \leftarrow \mathcal{X}^m$ or the instance is of the form $(\boldsymbol{A}, u)$ sampled from $\eta^{m\times n} \times \mathbb{Z}_q^m$ uniformly.

The advantage of an adversary $\mathcal{A}$ in solving a decisional NLWE problem is defined as follows:

$$Adv_{NLWE}^{q,n,m,\mathcal{X},\eta}(\mathcal{A}) = |\Pr[\mathcal{A}(\boldsymbol{A}, b) = 1] - \Pr[\mathcal{A}(\boldsymbol{A}, u) = 1]|. \tag{2}$$

According to reference [15], for some $k \geq 1$, a reduction occurs from $(\mathbb{Z}_q, n, m, \mathcal{D}_s, D_e)$-LWE to $(\mathbb{Z}_q, n, m, \mathcal{X}, \eta)$-NLWE for certain choices of the distribution η. As a result, the NLWE problem is at least as difficult as the standard LWE problem. In the paper [15], researchers demonstrated that NLWE is as difficult as LWE for a distribution η which is coset sampleable, i.e., there are two probabilistic polynomial-time (PPT) algorithms (MatrixGen, SamplePre) such that:

– MatrixGen$(1^k, n, q)$ outputs a matrix $A \in \mathbb{Z}_q^{n\times k}$ and auxiliary data t.
– SamplePre$(u \in \mathbb{Z}_q^n, t)$ outputs a vector $e \in \mathbb{Z}_q^k$ satisfying $Ae = u$.

Furthermore, if u is distributed uniformly in $\mathbb{Z}_q^n$, then the output of SamplePre(u, t) is distributed statistically close to η. We present the following theorem.

Theorem 2.1. ([15]) Let η be a coset-sampleable distribution. Assume there exists a PPT algorithm $\mathcal{A}$ that solves the $(\mathbb{Z}_q, k \cdot n, m, \mathcal{X}, \eta)$-NLWE problem with advantage ϵ. Then, η is an n-coset sampleable distribution, and there exists a PPT algorithm $\mathcal{B}$ that solves the $(\mathbb{Z}_q, n, m, \mathcal{X})$-LWE problem with the same advantage ϵ.

For some distributions η, such as the uniform distribution on {0,1}, a discrete Gaussian distribution, a uniform distribution over a sufficiently large linear subspace of $\mathbb{Z}_q$, reference [15] shows that for proper parameters, NLWE is as hard as LWE. The following corollary shows the relationship between LWE and NLWE problems when η is the uniform distribution on $\{0, 1\}$.

Corollary 2.1. ([15]) Let η be the uniform distribution on $\{0, 1\}$. Assume there exists a PPT algorithm $\mathcal{A}$ that solves the $(\mathbb{Z}_q, \lceil \log_2 q \rceil \cdot n, m, \mathcal{X}, \eta)$-NLWE problem with advantage ϵ. Then, η is an n-coset sampleable distribution, and there exists a PPT algorithm $\mathcal{B}$ that solves the $(\mathbb{Z}_q, n, m, \mathcal{X})$-LWE problem with the same advantage ϵ.

The relationship between LWE and NLWE problems is easily generalizable, and the proof of [15] can be directly applied to the following corollary, so the proof is omitted in this paper.

Corollary 2.2. Let $\eta = [-L/2, L/2]$. Assume there exists a PPT algorithm $\mathcal{A}$ that solves the $(\mathbb{Z}_q, \lceil \log_L q \rceil \cdot n, m, \mathcal{X}, \eta)$-NLWE problem with advantage ϵ. Then, η is an n-coset sampleable distribution, and there exists a PPT algorithm $\mathcal{B}$ that solves the $(\mathbb{Z}_q, n, m, \mathcal{X})$-LWE problem with the same advantage ϵ.

2.4 ECC

We encode the extracted key with two cryptographic primitives to account for the error caused by the difference between the biometric data used for registration and the biometric data used for authentication: EMBLEM encoding method and ECC.

The basic idea behind ECC is that the sender encodes the message with redundancy bits to allow the receiver to detect and correct a limited number of errors that may occur anywhere in the message without re-transmission. In this paper, we use the $(n_{ECC}, k_{ECC}, t_{ECC})$ Bose-Chaudhuri-Hocquenghem (BCH) code, which used in a variety of applications in digital communications and storage. n_{ECC} is the block length, k_{ECC} is the message length, and t_{ECC} is the hamming weight of error such that $k_{ECC} > t_{ECC}$.

- **ECC.Encode**: It takes as input a message $m \in \{0,1\}^{k_{ECC}}$ and outputs a codeword $c \in \{0,1\}^{n_{ECC}}$
- **ECC.Decode**: It takes as input a codeword $c \in \{0,1\}^{n_{ECC}}$ and outputs a message $m \in \{0,1\}^{k_{ECC}}$
- **Correctness**: For all messages $m \in \{0,1\}^{k_{ECC}}$ and for all errors $e \in \{0,1\}^{n_{ECC}}$ such that hamming weight of e is less than t_{ECC}, it holds that **ECC.Decode** (**Ecc.Encode** $(m) \oplus e) = m$.

2.5 EMBLEM Encoding Method [13]

We use an additional encoding method, EMBLEM encoding, with ECC encoding to tolerate errors caused by differences between the biometric data used for registration and the biometric data used for authentication. EMBLEM is a new multi-bit encoding method used in LWE-based encryption schemes [13]. Because both the public key and the ciphertext in an LWE-based encryption scheme contain errors, an additional error-handling mechanism is required to obtain the correct plaintext during the decryption algorithm. During the EMBLEM encoding, the t_{EMB}-bit message is encoded by concatenating the bit strings $1||0^d$ where d is a pre-defined error tolerance parameter. As long as the size of the error term is less than 2^d, the error cannot affect the message part. As a result, the message part can be extracted in its entirety. That is, EMBLEM decoding is accomplished by simply extracting the most significant t_{EMB}-bit from each coefficient. For a message $m \in \{0,1\}^{t_{\mathrm{EMB}}}$, **EMBLEM.Encode** and **EMBLEM.Decode** work as follows:

- **EMBLEM.Encode**. it proceeds as follows:
    1. Let m_i be the i-th bit of m for $i \in [0, t_{EMB} - 1]$.
    2. Compute $m_i' = \left(m_i| |1| |0^d\right) \in \mathbb{Z}_q$ where $\lfloor \log q \rfloor = d + 2$.
    3. Output $m' = \left(m_1', \ldots, m_{t_{EMB}}'\right) \in \mathbb{Z}_q^{t_{EMB}}$.
- **EMBLEM.Decode**. it proceeds as follows:
    1. Let m_i be the highest bit of m_i'.
    2. Output a t_{EMB}-bit string $m = (m_0|| \ldots ||m_{t_{EMB}-1})$.
- **Correctness**: For all messages $m \in \{0,1\}^{t_{EMB}}$ and for all errors $e \in \mathbb{Z}_q^{t_{EMB}}$ where the infinite norm of e is less than $2^d - 1$, it holds that **EMBLEM.Decode**(**EMBLEM.Encode**$(m) + e) = m$.

2.6 Fuller's Computational Fuzzy Extractor [6]

In 2017, Fuller et al. [6] introduced a computational fuzzy extractor based on the LWE problem. In order to reduce entropy loss, their construction did not use secure sketch. As a result, the scheme can extract a string whose length equals the entropy of the source. Furthermore, the **Gen** procedure takes $w \leftarrow W$, where W is a uniform distribution over $\mathbb{Z}_{\rho q}^m$ for $\rho \in (0,1)$. In other words, they assume that $w \leftarrow W$ where $W = \mathbb{Z}_{\rho q}^m$. Also, due to the decoding algorithm, their construction is inefficient and can only tolerate a sub-linear fraction of errors. In more detail, their construction is comprised of two algorithms, Gen and Rep, as follows:

- **Gen** (w): It proceeds as follows:
    1. Sample $\boldsymbol{A} \in \mathbb{Z}_q^{m \times n}$ and $s \in \mathbb{Z}_q^n$ uniformly.
    2. Set $P = (\boldsymbol{A}, \boldsymbol{A}s + w)$ and $R = s_{1,\ldots,n/2}$ which is the first $n/2$ coordinates of the secret vector $\boldsymbol{s}$.
    3. Output (P, R).
- **Rep**(w', P): It proceeds as follows:
    1. Compute $s' = \mathbf{Decode}_t(\boldsymbol{A}, b - w')$.
    2. Output $R' = s'_{1,\ldots,n/2}$.

- **Decode**$_t$$(\boldsymbol{A}, b = \boldsymbol{A}s + w - w')$: It proceeds as follows:
    1. Find n rows of $\boldsymbol{A}, b$ where n rows of $\boldsymbol{A}$ are linearly independent.
    2. Compute $s' = \left(\boldsymbol{A}'\right)^{-1} b'$
    3. If $b - \boldsymbol{A}s'$ has more than t nonzero coordinates, go to Step (2).
    4. Output s'

The **Decode**$_t$ algorithm proceeds recursively until Step (1) of the algorithm drops all rows containing errors. As a result, the more errors the value b demonstrates, the longer the decoding algorithm takes. To ensure efficient decoding, their design can only tolerate a limited number of errors. Huth et al. [14] present a device-server authentication system with pre-application robustness based on Fuller's construction, as well as the first implementation of a lossless computational fuzzy extractor in which the entropy of the source equals the entropy of the extracted key, in 2017. Furthermore, they provide parameter sets for 80-bit, 128-bit, and 256-bit security. Their fuzzy extractor can correct $t = O(\log n)$ errors as well. With their implementation, their scheme can correct up to $\frac{t}{m} = \frac{10}{768} = 1.3\%$ errors in the lossless setting [14].

3 Definitions

3.1 Computational Fuzzy Extractor

Fuzzy extractor extracts an almost uniformly random string R from a noisy source w. R can be recovered from any w' that is sufficiently close to w using a public helper string P, and R remains computationally close to uniform even when P is given. Two PPT algorithms make up a fuzzy extractor (Gen, Rep). Gen is the generation algorithm which takes as input $w \in W$ and outputs (P, R) with a public helper string P and an extracted string $R \in \{0,1\}^\ell$, and Rep is the reproduction algorithm which takes the public helper string P and $w' \in W$ and outputs an extracted string $R' \in \{0,1\}^\ell$.

It holds the following properties.

- **Correctness.** It holds that $R' = R$ with overwhelming probability for any $w, w' \in W$ such that $\delta = w - w' \in \mathcal{M}$, where $\mathcal{M}$ is an admissible error distribution for all $(P, R) \leftarrow \text{Gen}(w)$ and $R' \leftarrow \text{Rep}(P, w')$.
- **Security (pseudo-randomness).** The extracted string R is pseudo-random conditioned on P if $(R, P) \leftarrow \text{Gen}(w)$, that is $\mathcal{A}((R, P), (U_\ell, P)) \leq \varepsilon$ for any computationally bounded adversary $\mathcal{A}$ and any distribution W on $\mathcal{M}$.

The reusability is the security notion in the case that several pairs of extracted string and related helper data issued from correlated biometric data are revealed to an adversary, which is clearly a much stronger security guarantee. More formally, it is the security of a reissued pair $(R, P) \leftarrow \text{Gen}(w)$, given some pairs $(R_i, P_i) \leftarrow \text{Gen}(w_i)$ for correlated w and w_i. We follow the security model in [10].

- **Reusability.** For any distribution W over metric space $\mathcal{M}$ and any PPT adversary $\mathcal{A}$, its advantage defined below satisfies

$$Adv^{reu}_{FE,\mathcal{A}} := \left|\Pr[Exp^{reu}_{FE,\mathcal{A}}(1) \Rightarrow 1\right] - \left|\Pr[Exp^{reu}_{FE,\mathcal{A}}(0) \Rightarrow 1\right] \leq \varepsilon \quad (3)$$

where $Exp^{reu}_{FE,\mathcal{A}}(\beta)$ describes the reusability experiment conducted by a challenger $\mathcal{C}$ and an adversary $\mathcal{A}$ as follows:

$Exp^{reu}_{FE,\mathcal{A}}(\beta)$:

1. Challenger samples $w \leftarrow W$ and gets $(P, R) \leftarrow \text{Gen}(w)$. If $\beta = 0$, the challenger it returns (P, R) to $\mathcal{A}$; If $\beta = 1$, chooses U_R randomly and returns (P, U_R) .
2. $\mathcal{A}$ may adaptively make at most q_w queries of the following form:
   - $\mathcal{A}$ sends a $\delta_i \in \mathcal{M}$ to $\mathcal{C}$.
   - $\mathcal{C}$ gets $(P_i, R_i) \leftarrow \text{Gen}(w + \delta_i)$ and returns (P_i, R_i) to $\mathcal{A}$.

When $\mathcal{A}$ outputs a guessing bit b', the experiment outputs b'.

The robustness is a security notion that applies when an adversary modifies the helper data P before P is given to the user. The robustness of fuzzy extractor requires that any change of P by an adversary will be detected. Boyen et al. [9] introduced robust fuzzy extractor, and Dodis et al. [10] strengthened robustness to post-application robustness, which guarantees that the fuzzy extractor will detect any modification of P by an adversary who also has an access to R. In 2018, Wen et al. [5] proposed a robust and reusable fuzzy extractor in 2018 and generalized the concept of post-application robustness of reusable fuzzy extractor, allowing the adversary to ask the generation oracle with shift $\delta_i \in \mathcal{M}$ to get $(P_i, R_i) \leftarrow \text{Gen}(w + \delta_i)$. More formally, it is the security of a modified helper data $\hat{P}$, given some pairs $(R, P) \leftarrow \text{Gen}(w)$ and $(R_i, P_i) \leftarrow \text{Gen}(w_i)$ for correlated w and w_i. We follow the security model in [5].

- **Robustness.** For any distribution W over metric space $\mathcal{M}$ and any PPT adversary $\mathcal{A}$, its advantage defined below satisfies

$$Adv^{rob}_{FE,\mathcal{A}} := \left|\Pr[Exp^{rob}_{FE,\mathcal{A}}(1^\lambda) \Rightarrow 1]\right| \le \varepsilon \tag{4}$$

where $Exp^{rob}_{FE,\mathcal{A}}(1^\lambda)$ describes the robustness experiment conducted by a challenger $\mathcal{C}$ and an adversary $\mathcal{A}$ as follows:

$Exp^{rob}_{FE,\mathcal{A}}\left(1^\lambda\right)$:

1. Challenger samples $w \leftarrow W$, gets $(P, R) \leftarrow \text{Gen}(w)$ and returns (P, R) to $\mathcal{A}$.
2. $\mathcal{A}$ may adaptively make at most q_w queries of the following form:
   - $\mathcal{A}$ sends a $\delta_i \in \mathcal{M}$ to $\mathcal{C}$.
   - $\mathcal{C}$ gets $(P_i, R_i) \leftarrow \text{Gen}(w + \delta_i)$ and returns (P_i, R_i) to $\mathcal{A}$.
3. $\mathcal{A}$ submits its forgery $(\hat{P}, \hat{\delta})$ to $\mathcal{C}$.
   - $\mathcal{A}$ wins if $\hat{\delta} \in \mathcal{M}$, $\hat{P}$ is different from P and those P_i and $\text{Rep}\left(\hat{P}, \text{w} + \hat{\delta}\right) \neq \perp$.
   - The experiment outputs 1 if $\mathcal{A}$ wins and 0 otherwise.

4 Computational Fuzzy Extractor with Reusability

The main idea our scheme is that the biometric data $w \leftarrow W$ encrypts the random string R in a way that can be decrypted with the knowledge of some close w'. In this paper, we consider a uniform source $W = \mathbb{Z}_q^n$. The construction first chooses A and e from small range and chooses a random R to be used as the output of the fuzzy extractor. Then we compute the helper data b by using w as the secret key of a symmetric LWE-based encryption scheme, i.e., $b = Aw + e + R$. Based on the LWE problem, the value $Aw + e$ looks like a random value. Therefore, b hides the value Encode (R) securely.

During the reproduction phase, one computes $c = b - Aw' = A(w - w') + e + R$. Since A and e are sampled from small ranges and $w - w'$ is small, $c = R + e'$ where e' is a small. To extract the string R from the helper data c, the random string R is encoded with EMBLEM encoding method and ECC before adding to $Aw + e$ to compute b. Then one can extract R from $c = \text{Encode}(R) + e'$ if e' is small enough.

4.1 Construction

We present a computational reusable fuzzy extractor based on the NLWE problem in this paper as shown in Fig. 1. The following are the specifics of our construction. We assume that $w \leftarrow W$ where $W = \mathbb{Z}_q^n$.

- **Gen**(w): To generate a helper data P and an extracted string R, it proceeds as follows:
    1. Sample $A \in \eta^{m\times n}$ and $e \in \mathcal{X}^m$ uniformly
    2. Sample $R \leftarrow \{0, 1\}^{\ell}$
    3. Compute $b = Aw + e +$ Encode (R) where Encode $(.) =$ **EMBLEM.Encode** (**ECC.Encode**(.))
    4. Set $P = (A, b)$
    5. Output (P, R)

- **Rep** (w', P): To reproduce the string R from the helper data $P = (A, b)$ and w', it proceeds as follows:
    1. Compute $c = b - Aw'$
    2. Compute $R' =$ Decode (c) where Decode $(.) =$ **ECC.Decode (EMBLEM.Decode**(.))

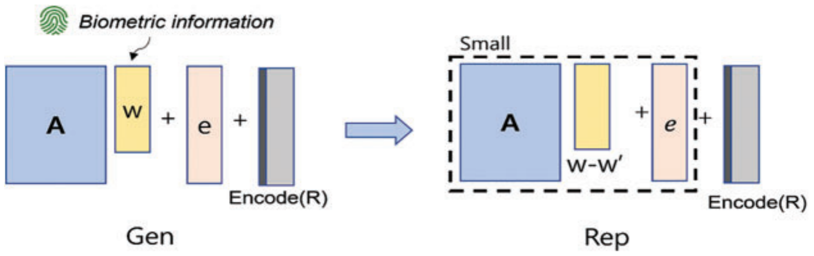

Figure 1: Construction of proposed fuzzy extractor scheme

4.2 Correctness

In this paper, we assume that the admissible error distribution $\delta := w - w'$ exhibits three types: (1) $\delta \in \mathbb{Z}_q^n$ has all zeros components except less than t out of n components are in $\{1, -1\}$, (2) $\delta \in \mathbb{Z}_q^n$ has all zeros components except less than t out of n components are in $\{2, 1, -1, -2\}$, (3) $\delta \in D_\rho$ is sampled from a Gaussian distribution with its standard deviation ρ.

If w, w' are close enough, then we can set $||A(w - w') + e||_\infty \leq 2^d - 1$ where $d = \lfloor \log q \rfloor - 1$ with overwhelming probability through proper parameter setting. If it is the case, then by the correctness of EMBLEM encoding method, **EMBLEM.Decode**(c) = **ECC.Encode** (R) . Now, it is easy to get R with **ECC.Decode**. If $||A(w - w') + e||_\infty \geq 2^d - 1$, then **EMBLEM.Decode**$(c)$ = **ECC.Encode** $(R) \oplus e'$. If the hamming weight of e' is less than t_{ECC} with proper parameter setting, then, by the correctness of ECC, we can get **ECC.Decode(EMBLEM.Decode**(c)) $= R$.

4.3 Security

Theorem 4.1. If the NLWE problem is difficult, our construction presented in Chapter 4.1 is a reusable computational fuzzy extractor.

Proof. We demonstrate the reusability of our construction by defining a sequence of games and proving that the adjacent games are indistinguishable. For each G_i for $i = 0, \ldots, 4$, γ_i is defined as the event that an adversary $\mathcal{A}$ correctly guesses.

Game $\boldsymbol{G_0}$: It is the original game $Exp^{reu}_{FE,\mathcal{A}}(0)$ in which $P = (\boldsymbol{A}, b) \leftarrow \text{Gen}(w)$ where challenger $\mathcal{C}$ samples $\boldsymbol{A} \leftarrow_r \eta^{m\times n}, e \leftarrow_r \mathcal{X}^n, R \leftarrow_r \{0,1\}^{\ell}$, sets $b = \boldsymbol{A}w + e + \text{Encode}(R)$. Finally, it returns (P, R) to $\mathcal{A}$. On receipt of a $\delta_i \in \mathcal{M}$ from $\mathcal{A}$, challenger $\mathcal{C}$ samples $\boldsymbol{A}_i \leftarrow_r \eta^{m\times n}$, $e_i \leftarrow_r \mathcal{X}^n, R_i \leftarrow_r \{0,1\}^{\ell}$, sets $b_i = \boldsymbol{A}_i(w+\delta_i) + e_i + \text{Encode}(R_i)$. In other words, challenger get $P_i = (\boldsymbol{A}_i, b_i) \leftarrow \text{Gen}(w+\delta_i)$. Finally, it returns (P_i, R_i) to $\mathcal{A}$. When $\mathcal{A}$ outputs a guessing bit β', the experiment outputs β'. Clearly, we have $\Pr[\gamma_0] = \Pr[Exp^{reu}_{FE,\mathcal{A}}(0) \Rightarrow 1]$.

Game $\boldsymbol{G_1}$: It is the same as G_0, except that $P_i \leftarrow \text{Gen}(w+\delta_i)$, now is changed to $b_i = \boldsymbol{A}_i w + \boldsymbol{A}_i\delta_i + e_i + \text{Encode}(R_i)$ where $\boldsymbol{A}_i \in \eta^{m\times n}$, $R_i \in \{0,1\}^{\ell}$. Clearly, we can get $\Pr[\gamma_0] = \Pr[\gamma_1]$.

Game $\boldsymbol{G_2}$: It is the same as G_1, except that $P_i \leftarrow \text{Gen}(w+\delta_i)$ and $P \leftarrow \text{Gen}(w)$, now is changed to$b_i = U_i + \boldsymbol{A}_i\delta_i + \text{ENC}(R_i)$, $P_i = (\boldsymbol{A}_i, b_i)$, $b = U + \text{Encode}(R)$, and $P = (\boldsymbol{A}, b)$ where $\boldsymbol{A}, \boldsymbol{A}_i \in \eta^{m\times n}$, $U, U_i \in \mathbb{Z}_q^n$, $R, R_i \in \{0,1\}^{\ell}$. As a result, $|\Pr[\gamma_1] - \Pr[\gamma_2]| \le Adv^{n,(q_w+1)m,q,k,\mathcal{X},\eta}_{NLWE}(\mathcal{A})$.

Game $\boldsymbol{G_3}$: It is the same as G_2, except that the coin $\beta = 0$ is replaced to $\beta = 1$, that is, the challenger gives (P, U_R) and (P_i, U_{R_i}) to the adversary. In this game, the helper data P and all P_i for $i = 1, \ldots, q_w$ have no information about w, and U, U_i are uniformly random. As a result, $\Pr[\gamma_2] = \Pr[\gamma_3]$.

Game $\boldsymbol{G_4}$: It is the same as G_3, except that $b_i = U_i + \text{Encode}(R_i)$ and $b = U + \text{Encode}(R)$ now is changed back to $b_i = \boldsymbol{A}_i w + e_i + \text{Encode}(R_i) + \boldsymbol{A}_i\delta_i$ and $b = \boldsymbol{A}w + e + Enc(R)$. We get $|\Pr[\gamma_3] - \Pr[\gamma_4]| \le Adv^{n,(q_w+1)m,q,k,\mathcal{X},\eta}_{NLWE}(\mathcal{A})$.

Game $\boldsymbol{G_5}$: It is the same as G_4, except that $b_i = \boldsymbol{A}_i w + e_i + \text{Encode}(R_i) + \boldsymbol{A}_i\delta_i$ now is changed back to $b_i = \boldsymbol{A}_i(w+\delta_i) + e_i + \text{Encode}(R_i)$. Clearly, we can get $\Pr[\delta_4] = \Pr[\delta_5]$. Furthermore, this game is the original game $Exp^{reu}_{FE,\mathcal{A}}(1)$. As a result, we have $\Pr[\delta_5] = \Pr[Exp^{reu}_{FE,\mathcal{A}}(1) \Rightarrow 1]$. As a result, we get $\left|\Pr[Exp^{reu}_{FE,\mathcal{A}}(1) \Rightarrow 1\right] - \left|\Pr[Exp^{reu}_{FE,\mathcal{A}}(0) \Rightarrow 1\right] \le 2 \cdot Adv^{n,(q_w+1)m,q,k,\mathcal{X},\eta}_{NLWE}(\mathcal{A})$.

5 Extension to Robust and Reusable Computational Fuzzy Extractor

Boyen et al. [9] utilizes a hash function to the fuzzy extractor scheme for robustness by feeding the biometric data into the hash function, i.e., $\sigma = H(w, pub)$ where *pub* is a public data. In reproducing phase, the user who has $w'(= w + \delta)$ can restore w using secure sketch. As a result, the robustness can be validated by a user. Because our scheme is not based on a secure sketch, we cannot use the general method in [9] for robustness. To achieve robustness in our scheme, two keys R and K are generated by a random string r, which is the helper data b is encrypting. Then, R is regarded as an extracted key, and K is regarded as an authentication key for hash function to provide robustness. Despite the fact that the extracted key R is revealed to the adversary through helper data, the adversary cannot forge σ without the authentication key K. In other words, our fuzzy extractor provides post-application robustness, because the only valid user can decrypt the helper data b using w' and obtain the random string r to compute the authentication K.

5.1 Construction

We present a computational reusable fuzzy extractor with robustness based on the construction of computational fuzzy extractor with reusability presented in Chapter 4. The details of our construction are as follows. We assume that $w \leftarrow W$ where $W = \mathbb{Z}_q^n$.

- **Gen** (w): To generate a helper data P and an extracted string R, it proceeds as follows:
  1. Sample $\boldsymbol{A} \in \eta^{m\times n}$ and $e \in \mathcal{X}^n$ uniformly
  2. Sample $r \leftarrow \{0,1\}^{\ell}$
  3. Compute $b = \boldsymbol{A}w + e+$ Encode (r)
  4. Compute $(R, K) = H_1(r, \boldsymbol{A}, b)$ where $R, K \in \{0,1\}^{\ell_1}$
  5. Compute $\sigma = H_2(K, \boldsymbol{A}, b) \in \{0,1\}^{\ell_2}$
  6. Set $P = (\boldsymbol{A}, b, \sigma)$
  7. Output (P, R)

- **Rep** (w', P): To reproduce the string R from the helper data $P = (\boldsymbol{A}, b, \sigma)$ and w', it proceeds as follows:
  1. Compute $c = b - \boldsymbol{A}w'$
  2. Compute $r' = \text{Decode}(c)$
  3. Compute $(R', K') = H_1(r', \boldsymbol{A}, b)$
  4. Compute $\sigma' = H_2(K', \boldsymbol{A}, b)$
  5. Check if $\sigma = \sigma'$ and output R'. Otherwise, output $\perp$.

5.2 Correctness

The correctness of the robust and reusable fuzzy extractor scheme in construction 5.1 follows from the correctness of the underlying reusable fuzzy extractor scheme in construction 4.1.

If w, w' are close enough, then we can set $||\boldsymbol{A}(w - w') + e||_\infty \le 2^d - 1$ where $d = \lfloor \log q \rfloor - 1$ with overwhelming probability with proper parameter settings. If this is the case, then by the correctness of EMBLEM encoding method, **EMBLEM.Decode**(c) = **ECC.Encode** (r). Now, obtaining r using **ECC.Decode** is simple. If $||A(w - w') + e||_\infty \ge 2^d - 1$, then **EMBLEM.Decode**$(c)$ = **ECC.Encode** $(r) \oplus e'$. If the hamming weight of e' is less than t_{ECC} with proper parameter setting, then, by the correctness of ECC, we can get **ECC.Decode(EMBLEM.Decode**$(c)) = r$.

5.3 Security

Theorem 5.1. If the NLWE problem is difficult, our construction presented in Chapter 5.1 is a computational fuzzy extractor with reusability.

The reusability of the fuzzy extractor scheme presented in construction 5.1 is guaranteed in the same way that Theorem 4.1 is guaranteed. We sketch a proof of reusability here. To achieve robustness, our scheme has some changes compared to the scheme in Chapter 4.1. First, b is encrypting a random string r, not an extracted string R directly. Second, two keys R and K are generated by a random string r with a random oracle. Lastly, the helper data are made up of $\boldsymbol{A}, b$ and $\sigma = H_2(K, \boldsymbol{A}, b)$. Despite giving additional information about σ to the adversary, the adversary's advantage does not increase at all because there is no relationship between R and K. Therefore, the rest of the proof of reusability is same to the proof of theorem 4.1. Now, the proof of robustness for fuzzy extractor is presented.

Theorem 5.2. If the NLWE problem is difficult, then our construction in Chapter 5.1 is a computational fuzzy extractor with robustness.

Proof. We prove the robustness of our construction, similar to the proof of reusability in Theorem 4.1, by defining a sequence of games and proving the adjacent games indistinguishable. For each G_i for $i = 0, \ldots, 4$, γ_i is defined as the event that an adversary $\mathcal{A}$ wins.

Game G_0: It is the original game $Exp^{rob}_{FE,\mathcal{A}}(1^{\lambda})$ in which $P = (\boldsymbol{A}, b, \sigma) \leftarrow \text{Gen}(w)$ where challenger $\mathcal{C}$ samples $\boldsymbol{A} \leftarrow_r \eta^{m\times n}, e \leftarrow_r \mathcal{X}^n, r \leftarrow_r \{0,1\}^{\ell}$, sets $b = \boldsymbol{A}w + e + \text{Encode}(r)$, and gets $(R, K) = H_1(r, \boldsymbol{A}, b)$ and $\sigma = H_2(K, \boldsymbol{A}, b)$. Finally, it returns (P, R) to $\mathcal{A}$ where $P = (\boldsymbol{A}, b, \sigma)$. On receipt of a $\delta_i \in \mathcal{M}$ from $\mathcal{A}$, challenger $\mathcal{C}$ samples $\boldsymbol{A}_i \leftarrow_r \eta^{m\times n}, e_i \leftarrow_r \mathcal{X}^n, r_i \leftarrow_r \{0,1\}^{\ell}$ sets $b_i = \boldsymbol{A}_i(w + \delta_i) + e_i + \text{Encode}(R_i)$ and gets $(R_i, K_i) = H_1(r_i, \boldsymbol{A}_i, b_i)$ and $\sigma_i = H_2(K_i, \boldsymbol{A}_i, b_i)$. In other words, challenger get $P_i = (\boldsymbol{A}_i, b_i, \sigma_i) \leftarrow \text{Gen}(w + \delta_i)$. Finally, it returns (P_i, R_i) to $\mathcal{A}$. Then, $\mathcal{A}$ submits to $\mathcal{C}$ its forgery $\left(\hat{P}, \hat{\delta}\right)$ with $\hat{P} = \left(\hat{\boldsymbol{A}}, \hat{b}, \hat{\sigma}\right)$. $\mathcal{A}$ wins if dis $\left(\hat{\delta}\right) \le t$, $\hat{P} \ne P$ or any other P_i s and Rep $\left(w + \hat{\delta}, \hat{P}\right) \ne \perp$. Recall that Rep $\left(w + \hat{\delta}, \hat{P}\right) \ne \perp$ if and only if dis $\left(w, w + \hat{\delta}\right) \le t$ and $\hat{\sigma}' = \hat{\sigma}$ holds, where $\hat{\sigma}' = H_2(\hat{K}', \hat{\boldsymbol{A}}, \hat{b})$, $\left(\hat{R}', \hat{K}'\right) = H_1(\hat{r}', \hat{\boldsymbol{A}}, \hat{b})$ and $\hat{r}' = \text{Decode}\left(\hat{b} - \hat{\boldsymbol{A}}\left(w + \hat{\delta}\right)\right)$. The game outputs 1 if $\mathcal{A}$ wins and 0 otherwise. Clearly, we have $\Pr[\gamma_0] = \Pr[Exp^{rob}_{FE,\mathcal{A}}(1^{\lambda}) \Rightarrow 1]$.

Game G_1: It is the same as G_0, except that $P_i \leftarrow \text{Gen}(w + \delta_i)$ now is changed to $b_i = \boldsymbol{A}_i w + \boldsymbol{A}_i \delta_i + e_i + \text{Encode}(r_i)$ where $\boldsymbol{A}_i \in \eta^{m\times n}$, $r_i \in \{0,1\}^{\ell}$. Clearly, we can get $\Pr[\gamma_0] = \Pr[\gamma_1]$.

Game G_2: It is the same as G_1, except that $P_i \leftarrow \text{Gen}(w + \delta_i)$ and $P \leftarrow \text{Gen}(w)$ now is changed to $b = U + \text{Encode}(r)$, $b_i = U_i + \boldsymbol{A}_i\delta_i + \text{Encode}(r_i)$ where $\boldsymbol{A}, \boldsymbol{A}_i \in \eta^{m\times n}$, $U, U_i \in \mathbb{Z}_q^n$, and $r, r_i \in \{0,1\}^{\ell}$. As a result, $|\Pr[\gamma_1] - \Pr[\gamma_2]| \le Adv^{n,(q_W+1)m,q,k,\mathcal{X},\eta}_{NLWE}(\mathcal{A})$.

Game G_3: It is the same as G_2, except that $\sigma = H_2(K, \boldsymbol{A}, b)$ and $\sigma_i = H_2(K_i, \boldsymbol{A}_i, b_i)$ now is changed to $\sigma, \sigma_i \leftarrow_r \{0,1\}^{\ell_2}$. Because H_2 is a random oracle, the adversary cannot tell the difference between Game G_3 and G_4. As a result, we can get $\Pr[\gamma_2] = \Pr[\gamma_3]$.

Game G_4: It is the same as G_3, except that $(R, K) = H_1(r, \boldsymbol{A}, b)$ and $(R_i, K_i) = H_1(r_i, \boldsymbol{A}_i, b_i)$ now is changed to $R, R_i, K, K_i \leftarrow_r \{0,1\}^{\ell_1}$. Because H_1 is a random oracle and b, b_i have no information about r, r_i, the adversary cannot tell the difference between Game G_3 and G_4. As a result, we can get $\Pr[\gamma_3] = \Pr[\gamma_4]$.

In this game, the helper data P and all P_i for $i = 1, \ldots, q_w$ exhibit no information about w. Also, because $\mathcal{A}$ cannot learn any information about w from queries, $\mathcal{A}$ must determine an appropriate value for $\hat{\sigma}$ of the forgery $\left(\hat{P}, \hat{\delta}\right)$ to win this game; i.e., if $\mathcal{A}$ "guesses" that $\hat{K} = H_1(\hat{r}, \hat{\boldsymbol{A}}, \hat{b})$ and $\hat{r} = \text{Decode}(\hat{b} - \hat{\boldsymbol{A}}\hat{w})$ for a particular choice of $(\hat{\boldsymbol{A}}, \hat{b})$, then a "winning" strategy is for $\mathcal{A}$ to obtain $\hat{\sigma} = H_2(\hat{K}, \hat{\boldsymbol{A}}, \hat{b})$ and output $\hat{P} = \left(\hat{\boldsymbol{A}}, \hat{b}, \hat{\sigma}\right)$.

Decode is a deterministic function determined by the inputs. Therefore, when $(\hat{\boldsymbol{A}}, \hat{b})$ are fixed, then $\hat{r} = \text{Decode}(\hat{b} - \hat{\boldsymbol{A}}\hat{w})$ is fixed where $w - \hat{w} \in \mathcal{M}$, and the value $\hat{\sigma} = H_2(\hat{K}, \hat{\boldsymbol{A}}, \hat{b})$is fixed as well because the value $\hat{K}$ is determined by the values $\hat{r}, \hat{\boldsymbol{A}}, \hat{b}$. As a result, if $\mathcal{A}$ ever outputs $\hat{P} = \left(\hat{\boldsymbol{A}}, \hat{b}, \hat{\sigma}\right)$ with $\left(\hat{\boldsymbol{A}}, \hat{b}\right) = (\boldsymbol{A}_i, b_i)$ for some $i \le q_w$, then $\hat{\sigma}$ should be equal to σ, and so, the response is always $\perp$. Thus, we simply assume that $\left(\hat{\boldsymbol{A}}, \hat{b}\right) \ne (\boldsymbol{A}_i, b_i)$ for all $i \le q_w$.

Assume for a moment that there are no collisions in the outputs of any of the adversary's random oracle queries. The probability that the forgery is "successful" is at most the probability that $\mathcal{A}$ asked a query of the form $H_2\left(\hat{K}, \hat{\boldsymbol{A}}, \hat{b}\right)$ for the correct $\hat{K}$ to get the value $\hat{\sigma}$ (where $\left(\hat{\boldsymbol{A}}, \hat{b}\right) \ne (\boldsymbol{A}_i, b_i)$ for all $i \le q_w$

and $\hat{K} = H_1\left(\hat{r}, \hat{A}, \hat{b}\right))$ plus the probability that such a query was not asked, yet $\mathcal{A}$ nevertheless managed to predict the value $H_2\left(\hat{K}, \hat{A}, \hat{b}\right)$. We can easily see that the second case occurs with a probability of no more than $2^{-\ell_2}$.

In the first case, the probability that the adversary $\mathcal{A}$ asked a query of the form $H_2\left(\hat{K}, \hat{A}, \hat{b}\right)$ for the correct $\hat{K}$ is at most the probability that $\mathcal{A}$ asked a query of the form $H_1\left(\hat{r}, \hat{A}, \hat{b}\right)$ for the correct $\hat{r}$ to get the value $\hat{K}$ plus the probability that such a query was not asked, yet $\mathcal{A}$ nevertheless managed to predict the value $H_1\left(\hat{r}, \hat{A}, \hat{b}\right)$. Obviously, the latter case occurs with a probability of not more than $2^{-\ell_1}$. There is at most one $\hat{K}$ such that $H_2\left(\hat{K}, \hat{A}, \hat{b}\right) = \hat{\sigma}$ for any $\hat{\sigma}$ and at most one $\hat{r}$ such that $H_1\left(\hat{r}, \hat{A}, \hat{b}\right) = \hat{K}$, since no collisions are found in these random oracle H_1, H_2 queries by assumption. Thus, the adversary is successful if this $\hat{r}$ equals the correct value Decode $\left(\hat{b} - \hat{A}\hat{w}\right)$. To get the correct value, the adversary either "predicts" the value $\hat{r}$ itself or compute Decode $\left(\hat{b} - \hat{A}\hat{w}\right)$ by "guessing" $\hat{w}$ for which dis $\left(w, \hat{w}\right) \leq t$. By what we just argued, the probability that this occurs is at most $2^{-\ell_1} + 2^{H_\infty\left(\hat{W}|pub\right)}$. Furthermore, by the birthday bound the probability of a collision is at most $\frac{q_{H_1}^2}{2^{2\ell_1}}$ and $\frac{q_{H_2}^2}{2^{\ell_2}}$ for hash oracle H_1 and H_2, respectively [9]. Therefore, we find that $\Pr[\gamma_4] \leq 2^{-\ell_1} + q_{H_1}^2 \cdot 2^{-2\ell_1} + \left(q_{H_2}^2 + 1\right) \cdot 2^{-\ell_2} + 2^{H_\infty\left(\hat{W}|pub\right)}$. By assumptions of the NLWE problem and the biometric data distribution, $2^{H_\infty\left(\hat{W}|pub\right)} \leq 2^{n \cdot \log_2 q}$. In conclusion, $\Pr\left[Exp_{FE,\mathcal{A}}^{rob}\left(1^\lambda\right) \Rightarrow 1\right] \leq Adv_{NLWE}^{n,(q_W+1)m,q,k,\mathcal{X},\eta}\left(\mathcal{A}\right) + 2^{-\ell_1} + q_{H_1}^2 \cdot 2^{-2\ell_1} + \left(q_{H_2}^2 + 1\right) \cdot 2^{-\ell_2} + 2^{n \cdot \log_2 q}$.

6 Concrete Parameter Settings

6.1 System Parameters

Let λ be the security parameter, which is the targeted bit security of the fuzzy extractor. We set the modulus $q = 256$ for all parameter settings in our scheme. The matrix A is sampled from the distribution η in $\mathbb{Z}_q$, where $\eta = [-L/2, L/2]$ and L be the value indicating the range used to sample the matric A, and we set $L = q^{1/2}$ in this paper. To mitigate the heavy cost of Gaussian sampling, we use a centered binomial distribution to simulate a Gaussian distribution for an error distribution, as described in [21]. A centered binomial distribution with the standard deviation of $\sqrt{k/2}$ can be generated as $\Sigma_{i=1}^{k}(b_i - b_i')$, where b_i, b_i' are uniformly random bits, as described in [22].

As previously stated, we use $(n_{ECC}, k_{ECC}, t_{ECC})$ BCH code, where n_{ECC} (the number of NLWE instances) is the block length, k_{ECC} (the length of an extracted key) is the message length, and t_{ECC} is the hamming weight of error such that $n_{ECC} > k_{ECC}$. As in Section 5, we use SHAKE as the hash function.

6.2 Concrete Hardness of the NLWE Problem

We use the method presented by [23], which is a software for estimating the hardness of LWE and a class of LWE-like problems, to estimate the hardness of the NLWE problem on which the security of our fuzzy extractor scheme is based. As previously stated, because the LWE problem can be reduced to the NLWE problem using Theorem 4.1 in Section 4, we can estimate the hardness of the NLWE problem. The LWE-estimator outputs an upper bound on the cost of solving NLWE using currently known attacks, such as the primal lattice attack described in [22,24,25] and the dual lattice attack described in [26,27].

The most important building block in most efficient NLWE attacks is the blockwise Korkine-Zolotarev (BKZ) lattice reduction algorithm [28]. As a result, our estimator calculates the overall hardness against NLWE solvers by estimating the cost of the BKZ algorithm. Several methods exist for calculating the running time of BKZ [22,28,29]. We determined to adopt the BKZ cost model of $(0.292) \cdot \beta$ where β is the BKZ block size. Shortest vector problem (SVP) solver is the foundation of the BKZ algorithm. In terms of the number of SVP oracle calls made by the BKZ algorithm, the core-SVP model [22] assumes that an SVP oracle is required to be called only once in a conservative model. The best-known classical SVP solver runs in time $\approx 2^{0.292 \cdot \beta}$. Tab. 1 shows that our fuzzy extractor under the chosen parameters provides at least λ bit security against NLWE problem.

6.3 *Concrete Parameters*

The helper data are made of $\boldsymbol{A}$ and b where $\boldsymbol{A}$ can be constructed with a 256-bit seed easily. As a result, the helper data are $(256 + m \cdot \log_2 q)/8$ bytes for our reusable fuzzy extractor. Furthermore, for our robust and reusable fuzzy extractor, the helper data consist of $seed_A$, b and σ. Therefore, the helper data are $(256 + m \cdot \log_2 q + \ell)/8$ bytes since the helper data are increased by only $\ell/8$ bytes. As previously stated, in this paper, we assume three cases of the vector $\delta = w - w'$: case (1): $\delta \in \mathbb{Z}_q^n$ has all zeros components except less than t out of n components are in $\{1, -1\}$, case (2): $\delta \in \mathbb{Z}_q^n$ has all zeros components except less than t out of n components are in $\{2, 1, -1, -2\}$, case (3): $\delta \in D_\rho$ is sampled in a Gaussian distribution with its standard deviation ρ. As a result, we provide error tolerance rates based on these scenarios.

Tab. 1 shows three parameter sets that target 80, 128, and 256 bits of security against adversaries, respectively. Fig. 2 presents the false rejection rates (FRRs) of reproduction algorithm when the error rates $(= t/n)$ increase. To get the average FRRs, we repeat our experiment 10,000 times each. For the 80-bit security with 30% error tolerance rates in case (1), we set $n = 160, m = 255, \ell = 87, k = 108$, and we choose BCH $(255, 87, 26)$ as an ECC encoding algorithm. For the 128-bit security with 20% error tolerance rates in case (1), we set $n = 256, m = 511, \ell = 130, k = 84$, and we choose BCH $(511, 130, 55)$ as an ECC encoding algorithm. Finally, for the 256-bit security level with 11% error tolerance rates in case (1), we set $n = 512, m = 1023, \ell = 258, k = 60$, and we choose BCH $(1023, 258, 106)$ as an ECC encoding algorithm. Then, the FRRs for these three parameter sets are all zero. In case (2), the error tolerance rates for 80, 128, and 256-bit security are 12%, 8%, and 5%, respectively. In case (3), the FRRs are all zero until the error is sampled from a Gaussian distribution with a standard deviation of 0.5, 0.35, and 0.25 for the 80, 128, and 256-bit security levels, respectively.

Table 1: Proposed fuzzy extractor parameter sets

Parameter	Set 1	Set 2	Set 3
λ	80	128	256
n	160	256	512
m	255	511	1023
k	108	84	60
$\ell\ (= \ell_1, \ell_2)$	87	130	258

(Continued)

Table 1: Continued

Parameter	Set 1	Set 2	Set 3
Biodata (Bytes)	160	256	512
$T = \frac{t}{n}$ (case 1)	30%	20%	11%
$T = \frac{t}{n}$ (case 2)	12%	8%	5%
Standard deviation ρ (case 3)	0.5	0.35	0.25
Bit hardness	81	129	261

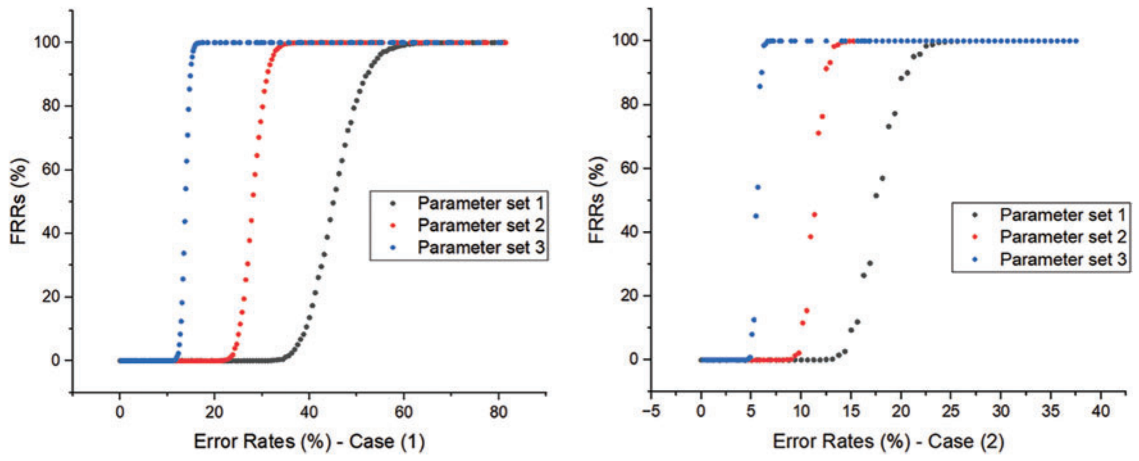

Figure 2: Simulation of reproduction algorithm

7 Results

7.1 Performance Analysis and Comparison

In this section, we describe the performance of our fuzzy extractor scheme. We evaluate the performance of our implementation on a 3.7GHz Intel Core i7-8700 k running Ubuntu 20.04 LTS. Our implementation codes are available to https://github.com/KU-Cryptographic-Protocol-Lab/Fuzzy_Extractor.

Huth et al. [14] improved on Fuller's construction [6] in terms of decoding efficiency. However, their fuzzy extractor is bounded by the same limitations as Fuller's, one of which is an error tolerance. The other one is that the decoding time increases with an increasing error t. As a result, they set a time limit for the decoding attempt to be less than 100 s. As a result, they can correct up to 1.3% error tolerance without false rejection, and the decoding time is more than 30 s on a machine with a 3.2 GHz single core and 8 GB RAM [14].

Canetti et al. [11] proposed a reusable fuzzy extractor based on a cryptographic digital locker in 2016, and their scheme requires a large size of helper data in order to store the hashed values of multiple subsets of the biometric data. For example, the helper data of their construction require approximately 33 GB of storage for 128-bit security with 19.5% error tolerance rates. Chen et al. [12] improved Canetti's construction [11] by reducing the storage space for helper data. However, their storage space for helper data still requires significantly more space than ours. More than 3 MB of storage space is required for 80-bit security with 20% error tolerance. Furthermore, the time complexity of these schemes' reproduction algorithms, which rely on the use of a digital locker to generate helper data, is involved with the computation of a significantly large number of hash values. For example, for 80-bit security with 20% error tolerance, Chen's scheme [12] requires 53.6×10^3 subsets to be hashed, according to [30]. Assuming that the digital locker of these schemes is instantiated with the SHA-256 hash function and that the computation of SHA-256 takes 1 ms, then decoding takes about 1 min. Tab. 2 indicates comparison between our scheme and previous fuzzy extractor schemes.

Table 2: Comparison with previous fuzzy extractor schemes

Schemes	Secure sketch	Security assumption	Reusability	Robustness	Decoding time when error rates increases
[3]	O	LWE	O	X	–
[5]	O	DDH/DLIN	O	O	–
[19]	O	DDH/LWE	O	O	–
[6]	X	LWE	X	X	↑
[16]	X	LWE	O	X	↑
[11]	X	X	O	X	↑
[12]	X	X	O	X	↑
Ours	X	NLWE	O	O	–

Conversely, our scheme does not increase the decoding time or storage space for helper data with an increasing error t. Regardless of an increasing error, the reproducing algorithm for our scheme, which includes the decoding algorithm takes 0.28, 0.89, and 3.42 ms, and the storage space requires only 298, 543, and 1,055 bytes for 80, 128, and 256-bit security, respectively, as shown in Tab. 3.

Table 3: Proposed fuzzy extractor performance

Parameter	Set 1	Set 2	Set 3
λ	80	128	256
Biodata (Bytes)	160	256	512
$T = \frac{t}{n}$ (case 1)	30%	20%	11%
Reusable fuzzy extractor (Construction 4.1)			
Helper data (bytes)	287	543	1055
Gen (K cycle)	1455	3862	14135
Rep (K cycle)	1015	3106	12410
Robust and reusable fuzzy extractor (Construction 5.1)			
Helper data (Bytes)	298	560	1088
Gen (K cycle)	1471	3963	14323
Rep (K cycle)	1017	3242	12882

8 Conclusions and Future Work

In this paper, we propose a new computational fuzzy extractor that is more efficient and has small size of helper data. As a result, our scheme is reusable and robust, and it can tolerate linear errors thanks to a new decoding algorithm that employs ECC and the EMBLEM encoding method. These points contribute to increasing the efficiency of the reproduction algorithm and supporting the linear fraction of errors for biometric data. Furthermore, as the error increases, our scheme does not increase the decoding time or storage space for helper data. We present the formal security proof for the proposed fuzzy extractor using the non-uniform LWE problem, as well as the concrete bit hardness for our scheme using the LWE-estimator. To ensure the security of our fuzzy extractor, we must assume that the biometric data are drawn from a uniform distribution, i.e., $w \leftarrow \mathbb{Z}_q^n$, which is an ideal distribution for our scheme. We are well aware that in reality, many fuzzy sources including biometric data, do not provide uniform distributions. The purpose of this paper is to inspire researchers to design an efficient computational fuzzy extractor based on our construction. As future work we want to investigate the hardness of the NLWE problem with non-uniform secret distribution, not just uniform distribution. Also, we want to investigate the feature extraction technique using a deep neural network, which can be of independent interest.

Funding Statement: This work was supported by Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (No. 2022-0-00518, Blockchain privacy preserving techniques based on data encryption).

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

[1] Y. Dodis, L. Reyzin and A. D. Smith, "Fuzzy extractors: How to generate strong keys from biometrics and other noisy data," in *Int. Conf. on the Theory and Applications of Cryptographic Techniques*, Interlaken, Switzerland, pp. 523–540, 2004.
[2] X. Boyen, "Reusable cryptographic fuzzy extractors," in *Proc. of the 11th ACM Conf. on Computer and Communications Security*, Washington, DC, USA, pp. 82–91, 2004.
[3] Y. Wen and S. Liu, "Reusable fuzzy extractor from LWE," in *Australasian Conf. on Information Security and Privacy*, Wollongong, NSW, Australia, pp. 13–27, 2018.
[4] Y. Wen, S. Liu and S. Han, "Reusable fuzzy extractor from the decisional Diffie-Hellman assumption," *Designs, Codes and Cryptography*, vol. 86, no. 11, pp. 2495–2512, 2018.
[5] Y. Wen and S. Liu, "Robustly reusable fuzzy extractor from standard assumptions," in *Int. Conf. on the Theory and Application of Cryptology and Information Security*, Brisbane, QLD, Australia, pp. 459–489, 2018.
[6] B. Fuller, X. Meng and L. Reyzin, "Computational Fuzzy Extractors," in *Int. Conf. on the Theory and Application of Cryptology and Information Security,*, Bengaluru, India, pp. 174–193, 2013.
[7] S. Simhadri, J. Steel and B. Fuller, "Cryptographic authentication from the iris," in *Int. Conf. on Information Security*, New York City, NY, USA, pp. 465–485, 2019.
[8] J. Daugman, "How iris recognition works," in *International Conference on Image Processing*, Rochester, New York, USA, pp. 33–36, 2002.
[9] X. Boyen, Y. Dodis, J. Katz, R. Ostrovsky and A. Smith, "Secure remote authentication using biometric data," in *Annual Int. Conf. on the Theory and Applications of Cryptographic Techniques*, Aarhus, Denmark, pp. 147–163, 2005.
[10] Y. Dodis, J. Katz, L. Reyzin and A. Smith, "Robust fuzzy extractors and authenticated key agreement from close secrets," in *Annual Int. Cryptology Conf.*, Santa Barbara, California, USA, pp. 232–250, 2006.

[11] R. Canetti, B. Fuller, O. Paneth, L. Reyzin and A. Smith, "Reusable fuzzy extractors for low-entropy distributions," *Journal of Cryptology*, vol. 34, no. 1, pp. 1–33, 2021.

[12] J. H. Cheon, J. Jeong, D. Kim and J. Lee, "A reusable fuzzy extractor with practical storage size," in *Australasian Conf. on Information Security and Privacy*, Wollongong, NSW, Australia, pp. 28–44, 2018.

[13] M. Seo, S. Kim, D. H. Lee and J. H. Park, "EMBLEM: (R) LWE-based key encapsulation with a new multi-bit encoding method," *International Journal of Information Security*, vol. 19, no. 4, pp. 383–399, 2020.

[14] C. Huth, D. Becker, J. G. Merchan, P. Duplys and T. Güneysu, "Securing systems with indispensable entropy: LWE-based lossless computational fuzzy extractor for the Internet of Things," *IEEE Access*, vol. 5, pp. 11909–11926, 2017.

[15] D. Boneh, K. Lewi, H. Montgomery and A. Raghunathan, "Key homomorphic PRFs and their applications," in *Annual Cryptology Conf.*, Santa Barbara, CA, USA, pp. 410–428, 2013.

[16] D. Apon, C. Cho, K. Eldefrawy and J. Katz, "Efficient, reusable fuzzy extractors from LWE," in *Int. Conf. on Cyber Security Cryptography and Machine Learning*, Beer-Sheva, Israel, pp. 1–18, 2017.

[17] R. Cramer, Y. Dodis, S. Fehr, C. Padró and D. Wichs, "Detection of algebraic manipulation with applications to robust secret sharing and fuzzy extractors," in *Annual Int.Conf. on the Theory and Applications of Cryptographic Techniques*, Istanbul, Turkey, pp. 471–488, 2008.

[18] B. Kanukurthi and L. Reyzin, "An improved robust fuzzy extractor," in *Int. Conf. on Security and Cryptography for Networks*, Amalfi, Italy, pp. 156–171, 2008.

[19] Y. Wen, S. Liu and G. Dawu, "Generic constructions of robustly reusable fuzzy extractor," in *IACR Int. Workshop on Public Key Cryptography*, Beijing, China, pp. 349–378, 2019.

[20] O. Regev, "On lattices, learning with errors, random linear codes, and cryptography," *Journal of the ACM (JACM)*, vol. 56, no. 6, pp. 1–40, 2009.

[21] X. Lu, Y. Liu, Z. Zhang, D. Jia, H. Xue *et al.,* "LAC: Practical ring-LWE based public-key encryption with byte-level modulus," *Cryptology ePrint Archive*, vol. 1009, pp. 1–36, 2018.

[22] E. Alkim, L. Ducas, T. Pöppelmann and P. Schwabe, "Post-quantum key exchange-A new hope," in *25th USENIX Security Symp. (USENIX Security 16)*, Austin, TX, USA, pp. 327–343, 2016.

[23] M. R. Albrecht, B. R. Curtis, A. Deo, A. Davidson, R. Player *et al.,* "Estimate all the {LWE, NTRU} schemes!," in *Int. Conf. on Security and Cryptography for Networks*, Amalfi, Italy, pp. 351–367, 2018.

[24] M. R. Albrecht, F. Göpfert, F. Virdia and T. Wunderer, "Revisiting the expected cost of solving uSVP and applications to LWE," in *Int. Conf. on the Theory and Application of Cryptology and Information Security*, Hong Kong, China, pp. 297–322, 2017.

[25] S. Bai and S. D. Galbraith, "Lattice decoding attacks on binary LWE," in *Australasian Conf. on Information Security and Privacy*, Wollongong, NSW, Australia, pp. 322–337, 2014.

[26] M. R. Albrecht, "On dual lattice attacks against small-secret LWE and parameter choices in HElib and SEAL," in *Annual Int. Conf. on the Theory and Applications of Cryptographic Techniques*, Paris, France, pp. 103–129, 2017.

[27] D. Micciancio and O. Regev, "Lattice-based cryptography," in: *Post-Quantum Cryptography*, Berlin, Heidelberg: Springer, pp. 147–191, 2009.

[28] Y. Chen and P. Q. Nguyen, "BKZ 2.0: Better lattice security estimates," in *Int. Conf. on the Theory and Application of Cryptology and Information Security*, Seoul, South Korea, pp. 1–20, 2011.

[29] M. R. Albrecht, R. Player and S. Scott, "On the concrete hardness of learning with errors," *Journal of Mathematical Cryptology*, vol. 9, no. 3, pp. 169–203, 2015.

[30] I. Lim, M. Seo, D. H. Lee and J. H. Park, "An improved fuzzy vector signature with reusability," *Applied Sciences*, vol. 10, no. 20, pp. 7141, 2020.