6-2021

Lattice-based remote user authentication from reusable fuzzy signature

Yangguang TIAN

Yingjiu LI

Robert H. DENG
Singapore Management University, robertdeng@smu.edu.sg

Binanda SENGUPTA

Guomin YANG

Citation

TIAN, Yangguang; LI, Yingjiu; DENG, Robert H.; SENGUPTA, Binanda; and YANG, Guomin. Lattice-based remote user authentication from reusable fuzzy signature. (2021). *Journal of Computer Security*. 29, (3), 273-298.
Available at: https://ink.library.smu.edu.sg/sis_research/6590

Journal of Computer Security 29 (2021) 273–298
DOI 10.3233/JCS-191370
IOS Press

Lattice-based remote user authentication from reusable fuzzy signature

Yangguang Tian [a,*], Yingjiu Li [b], Robert H. Deng [c], Binanda Sengupta [c] and Guomin Yang [d]

[a] *Cyber Security Engineering Area, Graduate School of Engineering, Osaka University, Japan*
E-mail: sunshine.tian86@gmail.com
[b] *Computer and Information Science, University of Oregon, USA*
E-mail: yingjiul@uoregon.edu
[c] *School of Information Systems, Singapore Management University, Singapore*
E-mails: robertdeng@smu.edu.sg, binujucse3@gmail.com
[d] *School of Computing and Information Technology, University of Wollongong, NSW, Australia*
E-mail: gyang@uow.edu.au

Abstract. In this paper, we introduce a new construction of reusable fuzzy signature based remote user authentication that is secure against quantum computers. We investigate the reusability of fuzzy signature, and we prove that the fuzzy signature schemes provide biometrics reusability (aka. reusable fuzzy signature). We define formal security models for the proposed construction, and we prove that it achieves user authenticity and user privacy. The proposed construction ensures: 1) a user's biometrics can be securely reused in remote user authentication; 2) a third party having access to the communication channel between a user and the authentication server cannot identify the user.

Keywords: Lattice-based cryptography, fuzzy signature, biometrics reusability, user privacy

1. Introduction

Fuzzy signature (FS), also known as signature with fuzzy secret key [22,28], is a new type of digital signature that allows a signature to be generated using biometrics as a signing key, without relying on any additional data (e.g., the helper data as used in fuzzy extractor [9]). FS can further be explored in a reusable setting: reusable fuzzy signature (RFS), which deals with biometrics reuse. This is because a user may generate multiple message-signature pairs under various signing keys, but with noisy versions of the same biometrics. Reusability allows the security of the generated signature used in an application to remain secure even if some of the generated signatures used in other applications are compromised (e.g., in case the signing keys are leaked to attackers). Therefore, biometrics reusability is an essential security property required in RFS, but there is no such security guarantee in [22,28].

The RFS is significantly useful in many real-world applications, such as biometric-based remote user authentication [3,6,7,20,30]. We consider a setting where user Alice wishes to authenticate herself to server Bob using her biometrics remotely. A traditional method is to use digital signature with fuzzy extractor [6,7,20,30]. That is, Alice relies on her enrolled helper data stored somewhere (e.g., server,

*Corresponding author. E-mail: sunshine.tian86@gmail.com.

smart card, and USB token) to derive a signing key for generating a signature on a nonce message, then Bob verifies the message-signature pair under Alice's enrolled verification key. In RFS, Alice's signature additionally includes a new verification key (her signature is generated by a new signing key), and a new helper data, in which the new helper data is derived from the new signing key and her noisy biometrics. Bob verifies the message-signature pair under Alice's new verification key and checks whether the new verification key is linked to Alice's enrolled verification key. This checking process relies on Alice's new and enrolled helper data, and the successful checking requires that Alice's noisy biometrics and her enrolled biometrics are close enough. A crucial point to stress is that Alice's signature generation does not depend on her enrolled helper data, which is the key difference between digital signature with fuzzy extractor [6,7,20,30] and RFS.

The security and privacy of RFS used in remote user authentication are also essential. First, lattice-based cryptography resistance to quantum computers has been extensively studied in the literature [14, 23–25]. We note that remote user authentication that exploits prior fuzzy signatures [22,28] is not secure, as a quantum computer can efficiently solve some hard mathematical problems such as discrete logarithm (DL) problem [27]. So, the DL-based fuzzy signature schemes proposed in [22,28] are not desired in the post-quantum era. Second, user's privacy is compromised if we apply fuzzy signatures [22,28] to remote user authentication. Specifically, multiple message-signature pairs from a same user Alice can be easily linked by a third party eavesdropping the communication channel, since the fuzzy signatures [22,28] are publicly verifiable (i.e., anyone has Alice's enrolled verification key can verify Alice's message-signature pairs). Therefore, the main goal of this work is to design reusable fuzzy signature based remote user authentication (RFS-RUA) that satisfies resistance to quantum computers, biometrics reusability, and privacy guarantee against eavesdroppers.

Technical Challenges. It is a non-trivial task to design a lattice-based RFS for RUA that is secure against quantum computers. We use the lattice-based digital signatures without trapdoors [11,21], because this line of research supports a simple and efficient signing process [21] compared to lattice-based trapdoor signatures [14]. However, in the design of lattice-based signatures without trapdoors, the signature generation is independent of the signing key. This mechanism may contradict to the homomorphic property required in RFS, which means that the difference ("shift") between two signatures is identical to the difference between two signing/verification key pairs [22,28]. Such homomorphic property allows Bob to find the correct difference between Alice's new verification key and her enrolled verification key. Alice's message-signature pair can be confirmed as valid because she is the only party who can produce the correct difference between her new signing key and her enrolled signing key.

Our Contributions. The main contributions of this work are summarized as follows.

- *New Construction.* We propose a new construction of remote user authentication that is built on top of lattice-based digital signatures [11,21], reusable fuzzy extractors from learning with errors (LWE) [2,32], a family of universal hash functions, and lattice-based public key encryptions.
- *Quantum Resistant.* The proposed construction can withstand quantum computers due to the lattice-based cryptographic primitives we used. Its provable security relies on the worst-case intractability of standard lattice problems.
- *Biometrics Reusability.* We formulate the security definition of RFS. We prove that the lattice-based RFS achieves biometrics reusability since the underlying LWE-based fuzzy extractors are reusable. In particular, the construction of RFS is the first cryptographic primitive that supports Hamming distance.

- *Privacy Protection.* We show a user privacy model to prevent the eavesdroppers from identifying any authorized user or linking any authorized user's multiple sessions. We prove that the proposed construction achieves user privacy under standard assumptions.

Overview of Techniques. We now explain our key technical insights. First, we characterize the homomorphic property of lattice signatures without trapdoors [11,21]. We discover that these lattice signatures have the desired homomorphic property, such that the "shift" between two lattice signatures is identical to the "shift" between two signing/verification key pairs. Second, putting all building blocks together, which include lattice signatures [11,21], reusable fuzzy signatures [2,32], and a family of universal hash functions, we can construct a lattice-based RFS that achieves biometrics reusability. Third, we use the lattice-based public-key encryptions to ensure privacy guarantee, where the validity of lattice-based RFS is verified by the authentication server only. Specifically, Alice's identity is encrypted under the public key of Bob. After extracting Alice's encrypted identity, Bob can identify her enrolled verification key and helper data. Then, he verifies Alice's lattice-based RFS under a new verification key, which is derived from the enrolled verification key, the enrolled and new helper data.

1.1. *Related work*

Fuzzy Signatures. The concept of fuzzy signature was firstly introduced by Takahashi et al. [28], which is a signature scheme that inputs fuzzy data such as biometrics as a signing key. Specifically, the generation of a signature does not rely on additional data such as helper data (or sketch). The proposed generic construction is built on top of a signature scheme with homomorphic (see Section 2.4) properties regarding keys and signatures, and a linear sketch. It is proven secure in the standard model. To relax some requirements on the building blocks used in the generic construction of [22], Matsuda et al. proposed a new generic construction using some relaxed building blocks. For example, Waters signature scheme [31] is replaced by Schnorr signature scheme [26].

The input biometrics of linear sketch in [28] is assumed to be uniformly distributed over metric space. To relax such strong assumption on fuzzy data, Matsuda et al. [22] require only high min-entropy on the distribution of biometrics. Specifically, they consider linear sketches as real numbers that include integer and decimal (i.e., secret key and biometrics) parts. If the difference between two linear sketches is less than a threshold t (a positive real number), then a difference algorithm (i.e., DiffRec) can extract the correct difference (may include noise) between two linear sketches.

Yasuda et al. [34] introduced the "recovering attacks" to recover both the secret key and biometrics from linear sketch. They claim that the "integer plus decimal" format is vulnerable to such attacks. In addition, they provided a trivial countermeasure (add small noisy data to the sketch) with informal security analysis. Meanwhile, Takahashi et al. [29] (merged version of [22,28]) also provided the treatment to avoid the "recovering attacks". The remedy is to add a "rounding-down" operation (or truncation) on the decimal part of the real numbers. However, such extra truncation would bring a correctness loss to the proposed constructions in [22,28].

Instead of using real numbers (with "integer plus decimal" format) to represent and process fuzzy data. In this work, we take binary strings over Hamming distance as fuzzy data input such as Iriscode [16]. We notice that constructing fuzzy signatures from Hamming distance is an open problem, which is pointed out by prior works [22,29].

Lattice Signatures. A lattice-based signature scheme with provable security was first constructed by Gentry et al. [14]. Their construction is based on "hash-and-sign" paradigm, and its security relies on the worst-case hardness of standard lattice problems such as small integer solution (SIS). To achieve more

Table 1

The comparison between different functionalities of lattice-based signature (Sig), fuzzy extractor (FE), and fuzzy signature (FS) schemes. Reusability shows whether the biometrics reuse is formally addressed or not. User privacy means user's privacy concern with respect to the eavesdroppers (see Appendix A). N/A means that the scheme did not consider this functionality

Functionality/Scheme	[21]	[13]	[28]	[22]	[2]	[32]	Ours
Sig/FS/FE	Sig	FE	FS	FS	FE	FE	FS
Reusability	N/A	×	N/A	N/A	✓	✓	✓
User Privacy	×	N/A	×	×	N/A	N/A	✓
Lattice Based	✓	✓	×	×	✓	✓	✓
Standard Model	×	✓	✓	×	✓	✓	×

practical and efficient constructions, another line of research on lattice-based digital signatures relies on the Fiat–Shamir heuristic [12] such as [11,21]. Their constructions are Schnorr-like identification protocols whose security is based on SIS or LWE, and the efficiency relies on a rejection sampling (see Section 2.4) rather than the pre-image sampling used in [14]. In this work, we use lattice-based signature schemes [11,21] as our building blocks.

Reusable Fuzzy Extractor. Fuzzy extractor (FE) is one of the promising approaches to construct a biometric-based remote user authentication [6,7,20,30]. Juels and Wattenberg [18] introduced a cryptography primitive called "fuzzy commitment". It is particularly useful for biometric-based remote authentication systems, because its error-correcting technique can correct certain errors within a suitable metric (Hamming distance). Dodis et al. [9] formally introduced the notions of "secure sketches" and "fuzzy extractors". In particular, they provided concrete constructions of secure sketches and fuzzy extractors in three metrics (Hamming distance, set difference, and edit distance), and the constructions are information-theoretically secure.

Boyen [6] introduced an important notion: reusable fuzzy extractor. It states that a user can produce multiple secret/public string pairs using the same biometrics w, i.e., $\{(R_i, P_i)\} \leftarrow \mathsf{Gen}(w)$. Later, Canetti et al. [8] proposed the first reusable FE from some low-entropy distributions. They particularly refined the security of strongly reusable FE, such that each R_i remains secure even if all other secret strings R_j $(j \neq i)$ are revealed.

Canetti et al. [8] proposed the first computational and information-theoretic secure FEs, Fuller et al. [13] constructed them from LWE [25]. In Fuller et al.'s construction, the entropy of the derived cryptographic key is the same as the entropy of the fuzzy biometrics from which the key is derived. However, their computational FE is not reusable. To achieve reusable FE from LWE, Apon et al. [2] provided a generic transformation to convert non-reusable (resp. weak reusable) fuzzy extractors to weak reusable (resp. strong reusable) ones. According to the definition of reusability [6,8], Apon et al. formalized both weak and strong reusability. Recently, Wen and Liu [32] proposed a new reusable and robust FE. Its reusability also follows the strong reusability defined in [2]. In this work, we follow the strong reusability defined in [2,32], where the "shifts" between many runs of $\mathsf{Gen}(w)$ are controlled by the adversary (i.e., the perturbation attacks defined in [6]).

To highlight our distinction, we show the function (feature) difference between our proposed new construction and some existing works in Table 1: it shows that our proposed construction has quantum resistance, biometrics reusability, and user privacy. The new construction is proven secure in the random oracle model. We stress that the proposed RFS can be regarded as a step forward from fuzzy signatures [22,28] and fuzzy extractors [2,32] in the lattice-based setting. Besides, we present the commonly used notations in Table 2.

Table 2

Summary of notations

Notation	Meaning
$(\mathsf{sk}_i, \mathsf{pk}_i)$	User i's key pair
(w, w')	Enrolled biometrics/near-by biometrics
$\mathsf{dist}(w, w')$	Distance between biometrics w and w'
$t \in \mathbb{R}^+$	Threshold value (a positive real number)
(R, P)	Secret string/public string (helper data)
$\mathsf{SS}(w, sk)$	Sketch (part of helper data P) with a secret key

1.2. *Paper organization*

In the next section, we present some preliminaries which will be used in our proposed construction. In Section 3, we present the notion of RFS and prove its reusability. In Section 4, we first present the formal security models to capture the security requirements of the RFS-based remote user authentication, then we show the proposed construction. We present the security analysis in Section 5, and the paper is concluded in Section 6.

2. Preliminaries

In this section, we present the complexity assumptions and the underlying techniques, which will be used in our proposed reusable fuzzy signatures RFS.

2.1. *Complexity assumptions*

Definition 2.1 ($\mathrm{SIS}_{q,n,m,d}$ Distribution)**.** Given a random matrix $\mathbb{A} \in \mathbb{Z}_q^{n\times m}$ and a vector $X \in \{-d, \dots, 0, \dots, d\}^m$, output $(\mathbb{A}, \mathbb{A} \cdot X)$, where d denotes the absolute value of random integers.

Definition 2.2 (Decisional SIS [21])**.** Given a pair $(\mathbb{A}, t)$ such that $t = \mathbb{A} \cdot X$, a probabilistic polynomial-time (PPT) adversary aims to decide whether the pair is generated from real $\mathrm{SIS}_{q,n,m,d}$ distribution, or whether it is uniformly generated from random distribution $\mathbb{Z}_q^{n\times m} \times \mathbb{Z}_q^n$.

If $d \gg q^{n/m}$ (high-density), then real $\mathrm{SIS}_{q,n,m,d}$ distribution is statistically close to uniform distribution over $\mathbb{Z}_q^{n\times m} \times \mathbb{Z}_q^n$, and there are many solutions X such that $\mathbb{A} \cdot X = t$. If $d \ll q^{n/m}$ (low-density), then there is only one solution X.

Definition 2.3 (Decisional LWE [25])**.** Given a matrix $\mathbb{A} \in \mathbb{Z}_q^{m\times n}$, a vector $X \in \mathbb{Z}_q^n$, and an arbitrary distribution $\chi \in \mathbb{Z}_q^m$, a PPT adversary aims to distinguish real distribution $(\mathbb{A}, \mathbb{A} \cdot X + \chi)$ from random distribution over $(\mathbb{Z}_q^{m\times n}, \mathbb{Z}_q^m)$. The $\mathrm{DLWE}_{q,n,m,\chi}$ is $(\epsilon, s_{\mathrm{sec}})$-secure if no PPT adversary $\mathcal{D}_{s_{\mathrm{sec}}}$ of size s_{sec} can distinguish the LWE instances from random ones except with probability ϵ, where $s_{\mathrm{sec}} = poly(\lambda)$, and ϵ is a negligible function of the security parameter λ.

Dottling and Muller-Quade [10] showed that one can encode biometrics w as the error term in a LWE problem by splitting it into m blocks. Furthermore, we rely on the result from Akavia et al. [1] to extract the pseudorandom bits, such that $X \in \mathbb{Z}_q^n$ has simultaneously many hardcore bits.

Lemma 2.1. *If* $\mathrm{DLWE}_{q,n-k,m,\chi}$ *is* $(\epsilon, s_{\mathrm{sec}})$ *secure, then*

$$\delta^{\mathcal{D}_{s'_{\mathrm{sec}}}}\big((X_{1,\dots,k}, \mathbb{A}, \mathbb{A}\cdot X+\chi), (U, \mathbb{A}, \mathbb{A}\cdot X+\chi)\big) \leqslant \epsilon,$$

where $U \in \mathbb{Z}_q^k$, $X_{1,\dots,k}$ *denotes the first k coordinates of X, and* $s'_{\mathrm{sec}} \approx s_{\mathrm{sec}} - n^3$.

2.2. *Universal hash function*

Let $\mathbb{H}$ be a universal hash function family whose domain is $\mathbb{Z}_{q^n}$ and whose range is $\mathbb{Z}_q$. Let $\mathbb{Z}_{q^n}$ be a vector space, which consists of n dimensional of finite ring with prime order q. We define an isomorphism $\psi : (\mathbb{Z}_q)^n \rightarrow \mathbb{Z}_{q^n}$ (ψ^{-1} is its inverse), and $n \in \mathbb{N}$. Note that $(\mathbb{Z}_q)^n = \mathbb{Z}_q^n$. A family of universal hash functions is defined as $\mathbb{H} = \{\mathtt{H}_z : \mathbb{Z}_q^n \rightarrow \mathbb{Z}_q | z \in \mathbb{Z}_{q^n}\}$. Specifically, for each invertible element $z \in Z$ in the seed space $Z \in \mathbb{Z}_{q^n}$, define the hash function $\mathtt{H}_z$ as follows: on input $x \in (\mathbb{Z}_q)^n$, $\mathtt{H}_z(x)$ computes $y \leftarrow \psi(x)\cdot z$, where "$\cdot$" denotes the multiplication in the extension field $\mathbb{Z}_{q^n}$. Let $(y_1, \dots, y_n) \leftarrow \psi^{-1}(y)$, and the output of $\mathtt{H}_z(x)$ is $y_1 \in \mathbb{Z}_q$. Since the isomorphism ψ between $\mathbb{Z}_q^n$ and $\mathbb{Z}_q^n$ is applied to the universal hash function family, we can easily get the desired linearity below

$$\forall x, x' \in (\mathbb{Z}_q)^n \text{ and } y_1, y_2 \in \mathbb{Z}_q : \quad y_1\cdot \mathtt{H}_z(x) + y_2\cdot \mathtt{H}_z\big(x'\big) = \mathtt{H}_z\big(y_1\cdot x + y_2\cdot x'\big).$$

Lemma 2.2. *Assume a family of functions* $\{\mathtt{H}_z : \mathbb{Z}_q^n \rightarrow \mathbb{Z}_q\}_{z\in Z}$ *is universal, for any random variable W taking values in* $\mathbb{Z}_q^n$ *and any random variable Y,*

$$\mathbf{SD}\big(\big(U_Z, \underline{\mathtt{H}_z(W)}, Y\big), (U_Z, \underline{U}, Y)\big) \leqslant \frac{1}{2}\sqrt{2^{-\tilde{\mathbf{H}}_\infty(W|Y)}\cdot|\mathbb{Z}_q|},$$

where U_Z *and* U *are uniformly distributed over* $\mathbb{Z}_{q^n}$ *and* $\mathbb{Z}_q$ *respectively. In particular, such universal hash functions are (average-case, strong) extractors with* ϵ*-statistically close to uniform. The detailed description of (average) mini-entropy* $\tilde{\mathbf{H}}_\infty$ *and statistical distance* **SD** *can be found in [9].*

2.3. *Computational fuzzy extractors*

Let $m \geqslant n$, and q be a prime number. The computational FE [13] consists of the following algorithms:

- Gen: The algorithm takes $w \leftarrow \mathcal{M}$, and $\mathbb{A} \in \mathbb{Z}_q^{m\times n}$, $x \in \mathbb{Z}_q^n$ as input, outputs (R, P), where $\mathcal{M}$ is a uniform distribution over $\mathbb{Z}_q^m$, and $(R, P) = [x_{1,\dots,n/2}, (\mathbb{A}, \mathbb{A}\cdot x + w)]$.
- Rep: The algorithm takes as (w', P) input, outputs $R = x_{1,\dots,n/2}$, where $P = (\mathbb{A}, c)$, $b = c - w'$, $x = \mathsf{Decode}_t(\mathbb{A}, b)$, and $\mathsf{dist}(w, w') \leqslant t$.

The correctness of computational FE relies on the $\mathsf{Decode}_t(\mathbb{A}, b)$ algorithm, which is explicitly shown as follows.

(1) Input: $(\mathbb{A}, b = \mathbb{A}\cdot x + w - w')$.
(2) Select $2n$ distinct indices $i_1, \dots, i_{2n} \leftarrow [1, \dots, m]$.
(3) Restrict $\mathbb{A}$, b to rows $i_1, \dots, i_{2n}$; Denote these by $\mathbb{A}_{i_1}, \dots, \mathbb{A}_{i_{2n}}, b_1, \dots, b_{i_{2n}}$.
(4) Find n linearly independent rows of $\mathbb{A}_{i_1}, \dots, \mathbb{A}_{i_{2n}}$ (if no such rows exist, output abort and stop), and restrict $\mathbb{A}_{i_1}, \dots, \mathbb{A}_{i_{2n}}, b_{i_1}, \dots, b_{i_{2n}}$ to n rows. Denote the result by $\mathbb{A}'$, b'.
(5) Compute $x' = \mathbb{A}'^{-1}\cdot b'$.

(6) Output: x' if $b - \mathbb{A} \cdot x'$ has at most t non-zero coordinates; Otherwise, it returns to step 2.

Recall that $\mathbb{A} \in \mathbb{Z}_q^{m\times n}$, $b \in \mathbb{Z}_q^m$, and Decode_t algorithm can correct at most $t = \mathcal{O}(\log n)$ errors (of Hamming distance) in a random linear code. Note that with probability at least $1/poly(\lambda)$, none of the $2n$ rows selected in step 2 have errors (i.e., biometrics w and w' agree on these rows), thus x' is a solution to the linear system. Furthermore, we notice that the sketch from LWE is in the form of $\mathsf{SS}(w, x) = \mathbb{A} \cdot x + w$, and satisfies the linearity defined in [22,28]. That is,

$$\mathsf{SS}\big(w, x + \Delta(x)\big) = \mathbb{A} \cdot \big(x + \Delta(x)\big) + w = (\mathbb{A} \cdot x + w) + \mathbb{A} \cdot \Delta(x),$$

where SS denotes a secure sketch procedure [13], which takes $w \leftarrow \mathcal{M}$ and a value $x \in \mathbb{Z}_q^n$ as input, output a distribution over $\mathbb{Z}_q^m$. The detailed description of fuzzy extractor and secure sketch is referred to Appendix B.

The computational fuzzy extractors (FE) from LWE has an inherent property: "indistinguishability" (IND). Informally, given two sketches (part of helper string P), which are the "encryption" of two independent biometrics, adversary cannot distinguish them without having decryption keys. We formally prove that the computational FE from LWE is secure in the IND model (note that the adversary here is allowed to access the public sketches only). In addition, we discover that both computational FE [13] and its variant reusable FEs [2,32] have such inherent property.

Definition 2.4. The IND experiment between an adversary $\mathcal{A}$ and a simulator $\mathcal{S}$ is defined below.

```
Experiment $\mathsf{Exp}_{FE}^{\mathrm{IND}}(\lambda)$
$b \in \{0, 1\}, w_b \leftarrow \mathcal{M}, \mathcal{Q} = \emptyset$
    $(R_i, P_i) \leftarrow \mathsf{Gen}(w_b), \mathcal{Q} \leftarrow \mathcal{Q} \cup P_i$
    $return\ P_i$
$b' = \mathcal{A}(\mathrm{guess}, c^*), c^* \leftarrow P_b$
$Return\ 1, if\ b' = b \wedge P_b \notin \mathcal{Q}; else, return\ 0.$
```

In the guess stage, $\mathcal{A}$ is given a challenge sketch c^*, which was not previously simulated by $\mathcal{S}$. We define the advantage of $\mathcal{A}$ as

$$\mathtt{Adv}_{\mathcal{A}}^{\mathrm{IND}}(\lambda) = \big|\Pr[\mathcal{S} \to 1] - 1/2\big|.$$

A computational FE from LWE is IND secure if $\mathtt{Adv}_{\mathcal{A}}^{\mathrm{IND}}(\lambda)$ is negligible in λ.

Lemma 2.3. *The computational fuzzy extractors from LWE achieves the IND security if the* $\mathrm{DLWE}_{q,n,m,\chi}$ *assumption is* $(\epsilon, s_{\mathrm{sec}})$ *secure.*

Informally, we can think of the sketch $\mathbb{A} \cdot x + w$ (part of helper string p) as an "encryption" of x that where decryption works from any close w' (i.e., decryption key). We can also think of any two "encryptions" $\mathbb{A}_0 \cdot x_0 + w_0$ and $\mathbb{A}_1 \cdot x_1 + w_1$ are indistinguishable by any third party without having decryption keys (w_0', w_1').

Proof. Assume that there exists a PPT $\mathcal{A}$ breaking the IND security of the computational fuzzy extractors from LWE, then we can construct an algorithm $\mathcal{S}$ to break the decisional LWE ($\mathrm{DLWE}_{q,n,m,\chi}$)

```
Adversary $\mathcal{S}(\mathbb{A}, v)$
$b \in \{0, 1\}, u \overset{R}{\leftarrow} \mathbb{Z}_q^n, w_b \overset{R}{\leftarrow} \mathbb{Z}_q^m$
  $P_0 \leftarrow \mathbb{A} \cdot X + \chi + w_0$; $P_1 \leftarrow \mathbb{A} \cdot (X + u) + \chi + w_1$
$b' = \mathcal{A}(\text{guess}, c^*), c^* \leftarrow P_b$
$If\ b' = b$, $return$ 1; $else$, $return$ 0.
```

Fig. 1. Description of adversary $\mathcal{S}$ for the proof.

assumption. The algorithm $\mathcal{S}$ has almost the same time complexity with $\mathcal{A}$. We first consider the shared public parameter by all users, i.e., $\mathbb{A}_0 = \mathbb{A}_1$. Then, we show the simulation differences when $\mathbb{A}_0 \neq \mathbb{A}_1$.

The algorithm $\mathcal{S}$ uses $\mathcal{A}$ as a subroutine (see Fig. 1, note that v can be either $\mathbb{A} \cdot X + \chi$ or a random distribution). $\mathcal{S}$ first generates another distribution which has the same property and distribution as its own challenge distribution. That is, the computed distribution $(\mathbb{A}, \mathbb{A} \cdot (X + u) + \chi + w)$, where u and w are randomly chosen by $\mathcal{S}$. If $\mathcal{S}$'s challenge is a real distribution, then it is the computed distribution; Otherwise, it is a random distribution over $(\mathbb{Z}_q^{m\times n}, \mathbb{Z}_q^m)$. By using its challenge and the computed distribution, $\mathcal{S}$ can simulate two sketches (P_0, P_1) for $\mathcal{A}$. At the guess stage, $\mathcal{S}$ returns a challenge ciphertext c^* to $\mathcal{A}$ according to the bit b.

We then analyze the behaviour of $\mathcal{S}$ on $\mathsf{Exp}_{\mathcal{S}}^{\text{LWE-REAL}}$ and $\mathsf{Exp}_{\mathcal{S}}^{\text{LWE-RAND}}$ respectively. In the $\mathsf{Exp}_{\mathcal{S}}^{\text{LWE-REAL}}$, the input $(\mathbb{A}, \mathbb{A} \cdot X + \chi + w)$ satisfies the Rep algorithm of FE described in Section 2.3, where w is uniformly distributed over a small interval. Notice that the computed distribution $(\mathbb{A}, \mathbb{A}\cdot(X+u)+\chi+w)$ are valid and they are uniformly and independently distributed over $(\mathbb{Z}_q^{m\times n}, \mathbb{Z}_q^m)$, because $\mathbb{A} \cdot (X + u) + \chi + w = \mathbb{A} \cdot X + \chi + \mathbb{A} \cdot u + w$ and u is a randomly element in $\mathbb{Z}_q^n$. Thus, $\mathcal{S}$ can simulate the proper distribution of two challenge sketches (i.e., $P_0 \leftarrow \mathbb{A} \cdot X + \chi + w_0$ and $P_1 \leftarrow \mathbb{A} \cdot (X + u) + \chi + w_1$), and the challenge ciphertext c^* is distributed exactly like a real sketch which associates with w_b.

$$P_0 \leftarrow \mathbb{A} \cdot X + \chi + w_0. \triangleright \quad \text{if } b = 0$$
$$P_1 \leftarrow \mathbb{A} \cdot (X + u) + \chi + w_1. \triangleright \quad \text{otherwise}$$

Therefore, we have $\mathsf{Exp}_{\mathcal{C}}^{\text{LWE-REAL}}$ below, which includes the experiment with respect to $b = 1$ (i.e., IND-1) and $b = 0$ (i.e., IND-0).

$$\begin{aligned}\Pr\big[\mathsf{Exp}_{\mathcal{S}}^{\text{LWE-REAL}}(\lambda) = 1\big] &= 1/2 \cdot \Pr\big[\mathsf{Exp}_{\mathcal{A}}^{\text{IND-1}}(\lambda) = 1\big] + 1/2 \cdot \big(1 - \Pr\big[\mathsf{Exp}_{\mathcal{A}}^{\text{IND-1}}(\lambda) = 1\big]\big)\\ &= 1/2 + 1/2 \cdot \mathtt{Adv}_{\mathcal{A}}^{\text{IND}}(\lambda).\end{aligned}$$

As for $\mathsf{Exp}_{\mathcal{S}}^{\text{LWE-RAND}}$, the input distributions to $\mathcal{S}$ in Fig. 1 are all uniformly distributed over $(\mathbb{Z}_q^{m\times n}, \mathbb{Z}_q^m)$. Therefore, the corresponding computed distribution above are also uniformly and independently distributed over $(\mathbb{Z}_q^{m\times n}, \mathbb{Z}_q^m)$. In particular, the challenge ciphertext is a random distribution over $(\mathbb{Z}_q^{m\times n}, \mathbb{Z}_q^m)$, and independent of bit b. Hence we have

$$\Pr\big[\mathsf{Exp}_{\mathcal{S}}^{\text{LWE-RAND}}(\lambda) = 1\big] \leqslant 1/2 + 1/2^{\lambda-1}.$$

The last term indicates that the random distribution to $\mathcal{S}$ happen to have the distribution of a real distribution, which is bounded by $1/2^{\lambda-1}$ since $2^{\lambda-1} < q < 2^{\lambda}$. By combing all equations above, we have

$$\begin{aligned}\mathtt{Adv}_{\mathcal{S}}^{\mathrm{LWE}}(\lambda) &= \Pr\left[\mathsf{Exp}_{\mathcal{S}}^{\mathrm{LWE\text{-}REAL}}(\lambda) = 1\right] + \Pr\left[\mathsf{Exp}_{\mathcal{S}}^{\mathrm{LWE\text{-}RAND}}(\lambda) = 1\right] \\ &\geqslant 1/2 \cdot \Pr[\mathsf{Exp}_{\mathcal{A}}^{\mathrm{IND}}(\lambda) - 1/2^{\lambda-1}.\end{aligned}$$

Finally, we analyze the case of $\mathbb{A}_0 \neq \mathbb{A}_1$. $\mathcal{S}$ performs the simulation using the same method described above, except that $\mathcal{A}$ simulates $P_0 \leftarrow (\mathbb{A}_0 = \mathbb{A}, \mathbb{A}_0 \cdot X + \chi + w_0)$; $P_1 \leftarrow (\mathbb{A}_1 = \mathbb{A}_0 \cdot \mathbb{A}^*, \mathbb{A}_1 \cdot (X + u) + \chi + w_1)$, where $\mathbb{A}^* \overset{R}{\leftarrow} \mathbb{Z}_q^{m\times n}$. $\square$

2.4. Lattice-based signatures

Definition 2.5. The continuous Gaussian distribution over $\mathbb{R}^m$ centred at $\mathbf{v}$ with standard deviation σ is defined by the function $\rho_{\mathbf{v},\sigma}^m(x) = (\frac{1}{\sqrt{2\pi\sigma^2}})^m e^{\frac{-\|x-\mathbf{v}\|^2}{2\sigma^2}}$.

When the center $\mathbf{v} = 0$ we write $\rho_\sigma^m(x)$, and the $\|\cdot\|$ denotes the Euclidean norm. The discrete Gaussian distribution over $\mathbb{Z}^m$ is defined as follows.

Definition 2.6. The discrete Gaussian distribution over $\mathbb{Z}^m$ centered at some $\mathbf{v} \in \mathbb{Z}^m$ with standard deviation σ is defined as $D_{\mathbf{v},\sigma}^m(x) = \rho_{\mathbf{v},\sigma}^m(x)/\rho_\sigma^m(\mathbb{Z}^m)$.

A lattice-based digital signature scheme $\Sigma = (\mathsf{Setup}, \mathsf{KG}, \mathsf{Sign}, \mathsf{Verify})$ has homomorphic property, if the following conditions are held. Besides, we provide the description of the standard digital signature in Appendix C.

- *Simple Key Generation.* $pp \leftarrow \mathsf{Setup}(\lambda)$ and $(\mathtt{s}k, \mathtt{p}k) \leftarrow \mathsf{KG}(pp)$, where $\mathtt{p}k$ is derived from $\mathtt{s}k$ via a deterministic algorithm $\mathtt{p}k \leftarrow \mathsf{KG}'(pp, \mathtt{s}k)$.
- *Linearity of Keys.* $\mathtt{p}k' \leftarrow \mathsf{KG}'(pp, \mathtt{s}k + \Delta(\mathtt{s}k)) = M_{\mathtt{p}k}(pp, \mathsf{KG}'(pp, \mathtt{s}k), \Delta(\mathtt{s}k))$, where $M_{\mathtt{p}k}$ denotes a deterministic algorithm which takes pp, a public key $\mathtt{p}k$ and a "shifted" value $\Delta(\mathtt{s}k)$, outputs a new public key $\mathtt{p}k'$.
- *Linearity of Signatures.* Two distributions are identical: $\{\sigma' \leftarrow \mathsf{Sign}(pp, \mathtt{s}k + \Delta(\mathtt{s}k), msg)\}$ and $\{\sigma' \leftarrow M_\Sigma(pp, \mathtt{p}k, msg, \sigma, \Delta(\mathtt{s}k))\}$, where $\sigma \leftarrow \mathsf{Sign}(pp, \mathtt{s}k, msg)$ and M_Σ denotes a deterministic algorithm which takes pp, a public key $\mathtt{p}k$, a message-signature pair (msg, σ) and a "shifted" value $\Delta(\mathtt{s}k)$, outputs a new signature σ'.
- *Linearity of Verifications.* We require that $\mathsf{Verify}(pp, M_{\mathtt{p}k}(pp, \mathtt{p}k, \Delta(\mathtt{s}k)), msg, M_\Sigma(pp, \mathtt{p}k,$ $msg, \sigma,$
$\Delta(\mathtt{s}k))) =$ "1″, and $\mathsf{Verify}(pp, \mathtt{p}k, msg, \sigma) =$ "1″.

We show that the lattice-based Schnorr-like signature schemes [11,21] have the homomorphic properties regarding keys and signatures. We first present the simplest version of the lattice-based signature scheme based on SIS [21], then, we show Lemma 2.4 aterwards.

- the singer's signing key is $\mathtt{s}k \leftarrow \mathbb{Z}_q^{m\times k}$, and its verification key is $\mathtt{p}k \leftarrow \mathbb{A} \cdot \mathtt{s}k \mod q$. These exists a hash function $\mathtt{H} : \{0,1\}^* \to \{-d, 0, d\}^k$ and $\mathbb{A} \in \mathbb{Z}_q^{n\times m}$.
- the signer generates a potential message-signature pair $(msg, \sigma) = (msg, c, z)$, where σ includes $c \leftarrow \mathtt{H}(\mathbb{A} \cdot y \mod q, msg)$ and $z \leftarrow \mathtt{s}k \cdot c + y$ (for $y \in \mathcal{D}_\sigma^m$).

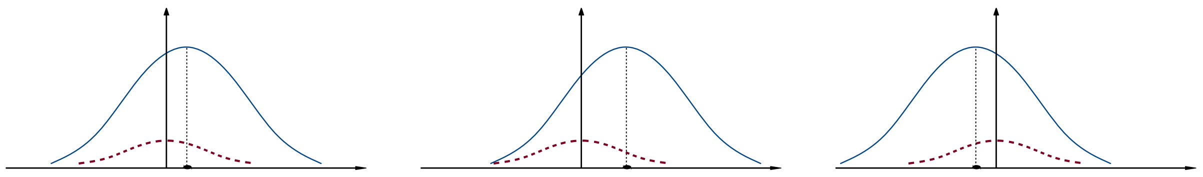
Fig. 2. Rejection sampling with "shifted" Gaussian distribution. In blue is the distribution of z, with $\mathbf{v} = \mathsf{s}k \cdot c$ (left figure) and "shifted" $\mathbf{v}' = (\mathsf{s}k + \Delta(\mathsf{s}k)) \cdot c$ (middle and right figures) over the spaces of all y before rejection sampling. In dashed red is the common target distribution $\mathcal{D}_\sigma^m$.

- the verifier accepts the signature iff $\|z\| \leqslant 2\sigma\sqrt{m}$ and $c = \mathtt{H}(\mathbb{A} \cdot z - \mathsf{p}k \cdot c \mod q, msg)$.

Lemma 2.4. *The lattice-based Schnorr-like signature schemes satisfy the homomorphic property.*

Proof. The key pair is a pair of integer matrixes such that $(\mathsf{s}k, \mathsf{p}k) = (\mathsf{s}k, \mathbb{A} \cdot \mathsf{s}k \mod q)$, where $\mathsf{s}k \in \mathbb{Z}_q^{m\times k}$ and $\mathbb{A} \in \mathbb{Z}_q^{n\times m}$ is a public parameter generated by a trusted party. The "shifted" $\mathsf{p}k' = \mathsf{p}k + \mathbb{A} \cdot \Delta(\mathsf{s}k) = \mathsf{KG}'(\mathbb{A}, \mathsf{s}k + \Delta(\mathsf{s}k))$. Therefore, the condition 1 and 2 are immediate held. As for the condition 3, by definition, $\sigma = (c, z) = (c \leftarrow \mathtt{H}(\mathbb{A} \cdot y \mod q, msg), z \leftarrow \mathsf{s}k \cdot c + y)$. If $\sigma' = \underline{(c', z')}$ is output by $M_\Sigma(pp, \mathsf{p}k, msg, \sigma, \Delta(\mathsf{s}k))$, then it holds that

$$z' \leftarrow \mathsf{s}k \cdot c + y + \underline{\Delta(\mathsf{s}k) \cdot c}, \quad c' = c \leftarrow \mathtt{H}(\mathbb{A} \cdot y \mod q, msg).$$

The output of z (resp. z') depends on the vector $\mathbf{v} = \mathsf{s}k \cdot c$ (resp. $\mathbf{v}' = (\mathsf{s}k + \Delta(\mathsf{s}k)) \cdot c'$) as well as the secret key $\mathsf{s}k$ (resp. $\mathsf{s}k + \Delta(\mathsf{s}k)$). To ensure that the signatures do not leak the secret key, the rejection sampling (Theorem 3.4 [21]) aims to remove such dependence. Informally, rejection sampling is to "re-center" the distribution of z to be a discrete Gaussian distribution centered at "zero" rather than at $\mathbf{v} = \mathsf{s}k \cdot c$. To show the condition 3 is held such that two distributions are identical in the case of "shifted" Gaussian distributions (i.e., $\mathcal{D}_{\mathbf{v}',\sigma}^m$ where $\mathbf{v}' = (\mathsf{s}k + \Delta(\mathsf{s}k)) \cdot c$), we present Fig. 2 in a two-dimensional space for correctness check.

When applying the rejection sampling (with same parameters pp, message msg and randomness $y \in \mathcal{D}_\sigma^m$), we have $z = z' \leftarrow \Sigma.\mathsf{Sign}(pp, \mathsf{s}k + \Delta(\mathsf{s}k), msg; y) \in \mathcal{D}_\sigma^m$. That is, the target distribution $\mathcal{D}_\sigma^m$ is independent of the "shifted" Gaussian distributions. It is easy to check that the two distributions we considered are identical, which shows that the condition 3 holds. Furthermore, the randomness $y \in \mathcal{D}_\sigma^m$ ensures that the discrete Gaussian distributions centered at "zero" and $\mathbf{v}/\mathbf{v}'$ have sufficient common overlap, hence the condition 4 regarding $\Sigma.\mathsf{Verify}$ is also held. Combing the above conditions together, the lemma is complete. □

3. Reusable fuzzy signature

In this section, we first introduce the construction of reusable fuzzy signature (RFS). Then, we show the formal reusability model of the RFS and its reusability analysis.

3.1. Construction

The RFS is built on top of a family of universal hash functions $\mathtt{H}$, fuzzy extractor $\mathsf{FE} = (\mathsf{Gen}, \mathsf{Rep})$, and digital signature $\Sigma = (\mathsf{KG}, \mathsf{Sign}, \mathsf{Verify})$. In particular, both FE and Σ have linearities so that the correctness of RFS is held. A RFS consists of the following algorithms.

- Setup: The algorithm takes a security parameter λ as input, outputs public parameter pp.
- Fuz-KG: The algorithm takes public parameter pp and biometrics w as input, outputs a key pair $(\mathtt{s}k, \mathtt{p}k) \leftarrow \Sigma.\mathsf{KG}$, and a sketch $P \leftarrow \mathsf{FE.Gen}(\mathtt{s}k, w)$. Let $(\mathtt{p}k, P)$ be a reference.
- Fuz-Sign: The randomized algorithm takes public parameter pp, a message msg and a nearby biometrics w' as inputs, outputs a tuple $(\mathtt{p}k', msg, \sigma, P')$, where $(\mathtt{s}k', \mathtt{p}k') \leftarrow \Sigma.\mathsf{KG}, \sigma \leftarrow \Sigma.\mathsf{Sign}(\mathtt{s}k', msg)$, $P' \leftarrow \mathsf{FE.Gen}(\mathtt{s}k', w')$.
- Fuz-Verify: The deterministic algorithm takes public parameter pp, message-signature pair (msg, σ), sketch P' and reference $(\mathtt{p}k, P)$ as input, outputs "1" if $1 = \Sigma.\mathsf{Verify}(\mathtt{p}k', \sigma, msg)$ and $\mathtt{p}k' \leftarrow M_{\mathtt{pk}}(pp, \mathtt{p}k, \Delta(\mathtt{s}k))$, where $\Delta(\mathtt{s}k) \leftarrow \mathtt{H}(\Delta(P))$ and $\Delta(P) = P' - P$; Otherwise, it outputs "0".

Correctness. The equation $\mathtt{p}k' \leftarrow M_{\mathtt{pk}}(pp, \mathtt{p}k, \Delta(\mathtt{s}k))$ holds if: 1) both the public keys $(\mathtt{p}k, \mathtt{p}k')$ and the sketches (P, P') are linear; and 2) $\mathsf{dist}(w, w') \leqslant t$. First, linearity of keys means that $\mathtt{p}k' = M_{\mathtt{pk}}(pp, \mathtt{p}k, \Delta(\mathtt{s}k))$, where $M_{\mathtt{pk}}$ is a deterministic algorithm (see Section 2.4). Linearity of sketches means that $P' = P + \Delta(P)$. Second, $\Delta(\mathtt{s}k) \leftarrow \mathtt{H}(\Delta(P))$ when $\mathsf{dist}(w, w') \leqslant t$. Specifically, if the biometrics involved in sketches (P, P') satisfy $\mathsf{dist}(w, w') \leqslant t$, the difference between two sketches (i.e., $\Delta(P) = P' - P$) equals to the difference between two secret keys (i.e., $\Delta(\mathtt{s}k) = \mathtt{s}k' - \mathtt{s}k$). Note that $\mathtt{H}$ is a universal hash function that maps each (uniformly chosen) key pair used in a digital signature to a secret string used in a public sketch.

3.2. Reusability analysis

First, we present the reusability model of the RFS. Informally, a reusable fuzzy signature requires that an adversary $\mathcal{A}$ cannot distinguish an extracted secret string R^* used to generate a fuzzy signature from a uniform string. Note that the secret string R^* is extracted from a biometrics w^*. The goal of reusability is to ensure that a biometrics w^* remains secure even if an attacker sees the fuzzy signatures generated by independent executions of Fuz-Sign on a series of inputs related to the biometrics w^*. The formal reusability game between an adversary $\mathcal{A}$ and a simulator $\mathcal{S}$ is defined as follows.

- Setup: $\mathcal{S}$ samples a biometrics $w^* \in \mathcal{M}$, generates a key pair $(\mathtt{s}k, \mathtt{p}k)$, a secret string and public sketch pair (R, P) by running the key generation algorithm $\mathsf{Fuz\text{-}KG}(pp, w^*)$. Then, $\mathcal{S}$ sends a reference $(\mathtt{p}k, P)$ to $\mathcal{A}$. $\mathcal{S}$ also tosses a random coin b which will be used later in the game.
- Training: $\mathcal{A}$ may choose a message msg_i and a shift $\delta_i \in \mathcal{M}$ with $\mathsf{dist}(\delta_i, 0) \leqslant t$, and issue Fuz-Sign queries to $\mathcal{S}$, while $\mathcal{S}$ performs the following operations.

  ∗ sample a new key pair $(\mathtt{s}k_i, \mathtt{p}k_i)$.
  ∗ obtain (R_i, P_i) by running $\mathsf{Fuz\text{-}KG}(pp, w^* + \delta_i)$, where $R_i \leftarrow \mathtt{H}(\mathtt{s}k_i)$.
  ∗ generate a message-signature pair (msg_i, σ_i) using the secret key $\mathtt{s}k_i$ (i.e., $\sigma_i \leftarrow \mathsf{Sign}(\mathtt{s}k_i, msg_i)$).
  ∗ return a fuzzy signature $(\mathtt{p}k_i, msg_i, \sigma_i, P_i)$ to $\mathcal{A}$.

- Challenge: $\mathcal{S}$ chooses a shift δ^* and obtains (R^*, P^*) by running $\mathsf{Fuz\text{-}KG}(pp, w^* + \delta^*)$, where $\delta^* \in \mathcal{M}$ and $\mathsf{dist}(\delta^*, 0) \leqslant t$. If $b = 0$, $\mathcal{S}$ gives R^* to $\mathcal{A}$; Otherwise, $\mathcal{S}$ chooses $U \overset{R}{\leftarrow} \{0, 1\}^{|R^*|}$ and gives U to $\mathcal{A}$. Eventually, $\mathcal{A}$ outputs a bit b', and $\mathcal{A}$ wins if $b' = b$. We define the advantage of an adversary $\mathcal{A}$ in the above game as

$$\mathtt{Adv}_{\mathcal{A}}^{\mathrm{RFS}}(\lambda) = \left|\Pr[\mathcal{S} \rightarrow 1] - 1/2\right|.$$

Second, we review the strong reusability of fuzzy extractors (FE) [2,32]. Informally, an adversary, who is given $\{(R_i, P_i)\}$ generated by independent executions of $\mathsf{Gen}(w^* + \delta_i)$ on a series of inputs related to the biometrics w^*, aims to distinguish an extracted secret string R^* from a random string.

Definition 3.1. Let $\mathsf{FE} = (\mathsf{Gen}, \mathsf{Rep})$ be an $(\mathcal{M}, \lambda, t, \epsilon)$-fuzzy extractor for class of distributions $\mathcal{M}$. The FE is ϵ-strongly reusable if for any $w \in \mathcal{M}$, any PPT adversary succeeds with probability at most $1/2 + \epsilon$ in the following experiment between an adversary $\mathcal{A}$ and a simulator $\mathcal{S}$.

(1) $\mathcal{S}$ samples $w^* \in \mathcal{M}$, and obtains (R^*, P^*) by running algorithm $\mathsf{Gen}(w^*)$. The value P^* is given to $\mathcal{A}$.
(2) $\mathcal{A}$ may adaptively make queries of the following form:
   - $\mathcal{A}$ outputs a shift $\delta_i \in \mathcal{M}$ with $\mathsf{dist}(\delta_i, 0) \leqslant t$.
   - $\mathcal{S}$ obtains (R_i, P_i) by running $\mathsf{Gen}(w^* + \delta_i)$, and returns them to $\mathcal{A}$.
(3) $\mathcal{S}$ tosses a random coin b. If $b = 0$, $\mathcal{S}$ gives R^* to $\mathcal{A}$; Otherwise, $\mathcal{S}$ chooses $U \overset{R}{\leftarrow} \{0,1\}^{|R^*|}$ and gives U to $\mathcal{A}$.
(4) $\mathcal{A}$ outputs a bit b', and $\mathcal{A}$ wins if $b' = b$.

Third, we reduce the reusability of RFS to the strong reusability of FE as follows.

Theorem 3.1. *The fuzzy signature scheme is reusable if the underlying fuzzy extractor is strongly reusable.*

Proof. Let $\mathcal{S}$ denote an adversary against the strongly reusability as defined above, who is given a public sketch P^* and a FE generation oracle $\mathcal{O}^{\mathsf{Gen}(w^*)}$, aims to distinguish a real secret string R^* from a random string U. $\mathcal{S}$ simulates the reusability game for $\mathcal{A}$ as follows.

- $\mathcal{S}$ obtains a string pair (R, P) by invoking his oracle $\mathcal{O}^{\mathsf{Gen}(w^*)}$. $\mathcal{A}$ is given a reference $(\mathtt{p}k, P)$, where $\mathtt{p}k \leftarrow \mathsf{KG}(pp, \mathtt{s}k)$, $\mathtt{s}k \leftarrow \mathtt{H}(R)$.
- If $\mathcal{A}$ issues a $\mathsf{Fuz\text{-}Sign}$ query in the form of (msg_i, δ_i) to $\mathcal{S}$, where $\delta_i \in \mathcal{M}$ and $\mathsf{dist}(\delta_i, 0) \leqslant t$, then $\mathcal{S}$ performs the following operations.
   * obtain the values (R_i, P_i) by invoking his oracle $\mathcal{O}^{\mathsf{Gen}(w^*)}$ on input δ_i (i.e., $\mathsf{Gen}(w^* + \delta_i)$);
   * compute a secret key $\mathtt{s}k_i \leftarrow \mathtt{H}(R_i)$, and generate its corresponding public key $\mathtt{p}k_i \leftarrow \mathsf{KG}(pp, \mathtt{s}k_i)$;
   * generate a message-signature pair (msg_i, σ_i) using the secret key $\mathtt{s}k_i$ (i.e., $\sigma_i \leftarrow \mathsf{Sign}(\mathtt{s}k_i, msg_i)$);
   * return a fuzzy signature $(\mathtt{p}k_i, msg_i, \sigma_i, P_i)$ to $\mathcal{A}$.
- $\mathcal{S}$ obtains a secret string R_b^* from his challenger, and sends (R_b^*, P^*) to $\mathcal{A}$. Finally, $\mathcal{S}$ outputs whatever $\mathcal{A}$ outputs. If $\mathcal{A}$ guesses the random bit correctly, then $\mathcal{S}$ breaks the strong reusability of FE. □

4. Proposed construction

In this section, we first present the security models (user authenticity and user privacy) for our proposed remote user authentication construction based on RFS (RFS-RUA). Then, we show the proposed construction and its security analysis.

States. We define a system entity set $\mathcal{U}$ with N users and M servers. We say an instance oracle Π_{ID}^{l} (e.g., session l of user ID) may be used or unused, and a user ID has unlimited number of instances called oracles. The oracle is considered as unused if it has never been initialized. Each unused oracle Π_{ID}^{l} can be initialized with a secret key $\mathtt{s}k$. The oracle is initialized as soon as it becomes part of a protocol execution. After the initialization the oracle is marked as used and turns into the stand-by state where it waits for an invocation to execute a protocol operation. Upon receiving such invocation the oracle Π_{ID}^{l} learns its partner id $\mathsf{pid}_{\mathrm{ID}}^{l}$ and turns into a processing state where it sends, receives and processes messages according to the description of the protocol. During that stage, the internal state information $state_{\mathrm{ID}}^{l}$ is maintained by the oracle. The oracle Π_{ID}^{l} remains in the processing state until it collects enough information to finalize the user authentication procedure. As soon as the user authentication is accomplished, Π_{ID}^{l} accepts and terminates the protocol execution in the sense that it would not send or receive further messages. If the protocol execution fails, then Π_{ID}^{l} terminates without having accepted.

We denote the l-th session established by a server as $\Pi_{\mathbb{S}}^{l}$ and identities of all the users recognized by $\Pi_{\mathbb{S}}^{l}$ during the execution of that session by partner identifier $\mathsf{pid}_{\mathbb{S}}^{l}$. We define $\mathsf{sid}_{\mathbb{S}}^{l}$ as the unique session identifier belonging to the session l established by a server $\mathbb{S}$. Specifically, we have $\mathsf{sid}_{\mathbb{S}}^{l} = \{msg_i\}_{i=1}^{n}$, where $msg_i \in \{0,1\}^*$ is the message transcript transmitted between users and servers.

4.1. *Definition*

A RFS-RUA consists of the following algorithms:

- Setup: The algorithm takes a security parameter λ as input, outputs a public parameter pp.
- KeyGen: The algorithm takes public parameter pp as input, outputs key pairs $\{(\mathtt{s}k, \mathtt{p}k)\}$ for users, and key pairs $\{(ssk, spk)\}$ for authentication servers.
- $\mathsf{Enrollment}$. This is a non-interactive protocol between a user and an authentication server over a secure channel. The user enrolls her identity ID, public key $\mathtt{p}k$, and sketch $\mathsf{SS}(w, \mathtt{s}k)$ to the authentication server. The enrolled users become authorized ones after enrollment, and the sketch $\mathsf{SS}(w, \mathtt{s}k)$ means that it derives from biometrics w and secret key $\mathtt{s}k$. We assume a uniform[1] biometrics source $\mathcal{M}$ and $w \in \mathcal{M}$.
- $\mathsf{Authentication}$. This is an interactive protocol between an authorized user and an authentication server over a public channel. The protocol takes public parameter pp, an authorized user's identity ID and biometrics w', and an authentication server's public key spk as input, outputs a new key pair $(\mathtt{s}k', \mathtt{p}k')$, a new sketch $\mathsf{SS}(w', \mathtt{s}k')$ and a message-signature pair (msg, σ). The authentication server accepts the user if: 1) the message-signature pair (msg, σ) is verified as valid under her new public key $\mathtt{p}k'$; and 2) the new public key $\mathtt{p}k'$ is linked to new sketch $\mathsf{SS}(w', \mathtt{s}k')$, and her enrolled public key $\mathtt{p}k$ and sketch $\mathsf{SS}(w, \mathtt{s}k)$. The message msg denotes the transmitted ephemeral data, and the biometrics satisfies $\mathsf{dist}(w', w) \leqslant t$.

4.2. *Security models*

User Authenticity. We define Δ as the set of functions $f : \mathcal{M} \to \mathcal{M}$ satisfies: for any $w \in \mathcal{M}$, $f(w)$ is "close" to w (i.e., $\mathsf{dist}(w, f(w)) \leqslant t$). The perturbations $\delta \in \Delta$ are specified by the adversary and

[1]One may question a uniform source is not practical, we stress that the uniform source can be replaced by a non-uniform source (e.g., symbol-fixing source [19]) while the security of FE is held. We use a uniform source here for presentation simplicity, the case of non-uniform source is discussed in [13,32,33].

applied by the simulator (i.e., challenger). Furthermore, the simulator specifies a class of functions $\Phi = \{\phi\}^{|\mathtt{s}k|}$ whose domain and range are the secret key space of RFS-RUA. In the user authenticity game, an adversary attempts to impersonate an authorized user and authenticate herself to an authentication server. We define a formal authenticity game between a probabilistic polynomial-time (PPT) adversary $\mathcal{A}$ and a simulator $\mathcal{S}$ as follows.

- Setup. $\mathcal{S}$ first generates identities $\{\mathbb{S}_j\}$ and key pairs $\{(ssk_j, sspk_j)\}$ ($j \in [1, M]$) for M servers, and identities $\{\mathrm{ID}_i\}$ ($i \in [1, N]$) for N users in the system. $\mathcal{S}$ also generates user's biometrics information $\{w_i\}$ ($w_i \in \mathcal{M}$), generates a set of public/secret key pairs $\{\mathtt{p}k_i^j, \mathtt{s}k_i^j\}_{j=1}^M$ and their corresponding sketches $\mathsf{SS}(w_i, \mathtt{s}k_i^j)$ for each user i with respect to M servers. Eventually, $\mathcal{S}$ sends all identities, public keys and sketches to $\mathcal{A}$. Let KG be a key generation algorithm which takes public parameter pp and a secret key $\mathtt{s}k$ as input, outputs a public key $\mathtt{p}k$.
- Training. $\mathcal{A}$ can make the following queries in an arbitrary sequence to $\mathcal{S}$.
  - ∗ Send: If $\mathcal{A}$ issues a send query in the form of (ID, l, msg) to simulate a network message for the l-th session of user ID, then $\mathcal{S}$ would simulate the reaction of instance oracle Π_{ID}^l upon receiving message msg, and return to $\mathcal{A}$ the response that Π_{ID}^l would generate; If $\mathcal{A}$ issues a send query in the form of (ID$'$, '$start$'), then $\mathcal{S}$ creates a new instance oracle $\Pi_{\mathrm{ID}'}^l$ and returns to $\mathcal{A}$ the first protocol message.
  - ∗ Biometrics Reveal: If $\mathcal{A}$ issues a biometrics reveal query to user i, then $\mathcal{S}$ returns user i's biometric information w_i to $\mathcal{A}$.
  - ∗ Secret Key Reveal: If $\mathcal{A}$ issues a secret key reveal query to user i with respect to server $\mathbb{S}_j$, then $\mathcal{S}$ returns the enrolled secret key $\mathtt{s}k_i^j$ to $\mathcal{A}$.
  - ∗ Biometrics Shift: If $\mathcal{A}$ issues a biometrics shift query with δ to user i, then $\mathcal{S}$ returns user i's shifted biometric information $\mathsf{SS}(w_i + \delta, \mathtt{s}k_i^j)$ to $\mathcal{A}$.
  - ∗ Secret Key Shift: If $\mathcal{A}$ issues a secret key shift query with ϕ to user i, then $\mathcal{S}$ returns user i's shifted public key $\mathtt{p}k_i' \leftarrow \mathsf{KG}(pp, \phi(\mathtt{s}k_i^j))$ to $\mathcal{A}$.
  - ∗ Server Secret Key Reveal: If $\mathcal{A}$ issues a server secret key reveal query to server $\mathbb{S}_j$, then $\mathcal{S}$ will return server $\mathbb{S}_j$'s secret key ssk_j to $\mathcal{A}$.
- Challenge. $\mathcal{A}$ outputs a message msg and wins the game if all of the following conditions hold.
  1. (1) $\mathbb{S}_j$ accepts user i. It implies $\mathsf{pid}_{\mathbb{S}_j}^s$ and $\mathsf{sid}_{\mathbb{S}_j}^s$ exist.
  2. (2) $\mathcal{A}$ did not issue Biometrics Reveal query to user i.
  3. (3) $\mathcal{A}$ did not issue Secret Key Reveal query to user i with respect to $\mathbb{S}_j$.
  4. (4) $msg \in \mathsf{sid}_{\mathbb{S}_j}^s$, but there exists no instance oracle $\Pi_{\mathrm{ID}_i}^s$ which has sent msg (msg denotes the message transcript from user i).

  $\mathcal{A}$ is allowed to reveal user i's secret keys associate with M-1 servers. We also allow $\mathcal{A}$ to adaptively issue Biometrics Shift queries to challenge user i. We define the advantage of an adversary $\mathcal{A}$ in the above game as

$$\mathtt{Adv}_{\mathcal{A}}^{\text{RFS-RUA}}(\lambda) = \left|\Pr[\mathcal{A}\ wins]\right|.$$

Definition 4.1. A RFS-RUA has user authenticity if for any PPT $\mathcal{A}$, $\mathtt{Adv}_{\mathcal{A}}^{\text{RFS-RUA}}(\lambda)$ is a negligible function of the security parameter λ.

Remark. The perturbation $\delta \in \Delta$ from adversary is used to capture the adaptive chosen perturbation attacks [6]. For example, an adversary may query an oracle to perform extractions and re-generations based on the chosen perturbations of biometrics. The function $\Phi = \{\phi\}^{|\mathtt{s}k|}$ from simulator is used to capture the related key attacks, such that an adversary may "modify" the secret keys [4]. In particular, the simulator should "update" the simulated signatures under the new public keys [22].

User Privacy. Informally, an adversary attempts to identify the authenticated users involved in a RFS-RUA. We define a formal user privacy game between an adversary $\mathcal{A}$ and a simulator $\mathcal{S}$ as follows:

- Setup: $\mathcal{S}$ first generates identities $\{\mathbb{S}_j\}$ and key pairs $\{(ssk_j, spk_j)\}$ ($j \in [1, M]$) for M servers, and identities $\{\mathrm{ID}_i\}$ ($i \in [1, N]$) for N users in the system. $\mathcal{S}$ also generates biometrics information $\{w_i\}$ for N users, generates a set of public/secret key pairs $\{\mathtt{p}k_i^j, \mathtt{s}k_i^j\}_{j=1}^{M}$ and their corresponding sketches $\mathsf{SS}(w_i, \mathtt{s}k_i^j)$ with respect to M servers. Eventually, $\mathcal{S}$ sends all identities, public keys and sketches to $\mathcal{A}$. $\mathcal{S}$ also tosses a random coin b which will be used later in the game.
- Training: $\mathcal{A}$ is allowed to issue Execute queries to $\mathcal{S}$. In addition, $\mathcal{A}$ can issue at most N-2 Biometrics Reveal, N-2 Secret Key Reveal queries, and M-1 Server Secret Key Reveal queries to $\mathcal{S}$. We denote the honest (i.e., uncorrupted) user and server set as $(\mathcal{U}_0', \mathcal{U}_1')$, respectively. The honest user requires that her biometrics as well as her secret key are not corrupted.
  The Execute query [5] allows $\mathcal{A}$ to invoke an honest execution of the protocol and returns a transcript of exchanged messages to $\mathcal{A}$.
- Challenge: $\mathcal{S}$ randomly selects two users $\mathrm{ID}_i, \mathrm{ID}_j \in \mathcal{U}_0'$ as challenge candidates. $\mathcal{S}$ removes them from $\mathcal{U}_0'$ and simulates ID_b^* by either $\mathrm{ID}_b^* = \mathrm{ID}_i$ if $b = 1$ or $\mathrm{ID}_b^* = \mathrm{ID}_j$ if $b = 0$. $\mathcal{S}$ also selects a server $\mathrm{ID}_{\mathbb{S}} \in \mathcal{U}_1'$ at random, and let $\mathrm{ID}_{\mathbb{S}}$ interact with the challenge user ID_b^*. $\mathcal{A}$ can access all the communication transcripts among them.

$$\mathrm{ID}_{\mathbb{S}} \leftrightarrow \mathrm{ID}_b^* = \begin{cases} \mathrm{ID}_i & b = 1 \\ \mathrm{ID}_j & b = 0 \end{cases}$$

  $\mathcal{A}$ is allowed to issue Secret Key Shift queries to challenge candidates. Finally, $\mathcal{A}$ outputs b' as its guess for b. If $b' = b$, then $\mathcal{S}$ outputs 1; Otherwise, $\mathcal{S}$ outputs 0. We define the advantage of an adversary $\mathcal{A}$ in the above game as

$$\mathtt{Adv}_{\mathcal{A}}^{\text{RFS-RUA}}(\lambda) = \left|\Pr[\mathcal{S} \to 1] - 1/2\right|.$$

Definition 4.2. A RFS-RUA has user privacy if for any PPT $\mathcal{A}$, $\mathtt{Adv}_{\mathcal{A}}^{\text{RFS-RUA}}(\lambda)$ is a negligible function of the security parameter λ.

4.3. Proposed construction

The proposed construction consists of the following building blocks.

- A LWE-based computational fuzzy extractor scheme $\mathsf{FE} = (\mathsf{Gen}, \mathsf{Rep})$.
- An existential unforgeability under chosen message attack EUF-CMA secure digital signature scheme $\Sigma = (\mathsf{KG}, \mathsf{Sign}, \mathsf{Verify})$.
- An indistinguishability under chosen plaintext attack IND-CPA secure public key encryption scheme $\mathsf{PKE} = (\mathsf{KG}, \mathsf{Enc}, \mathsf{Dec})$.
- A universal hash function family $\mathbb{H}$.

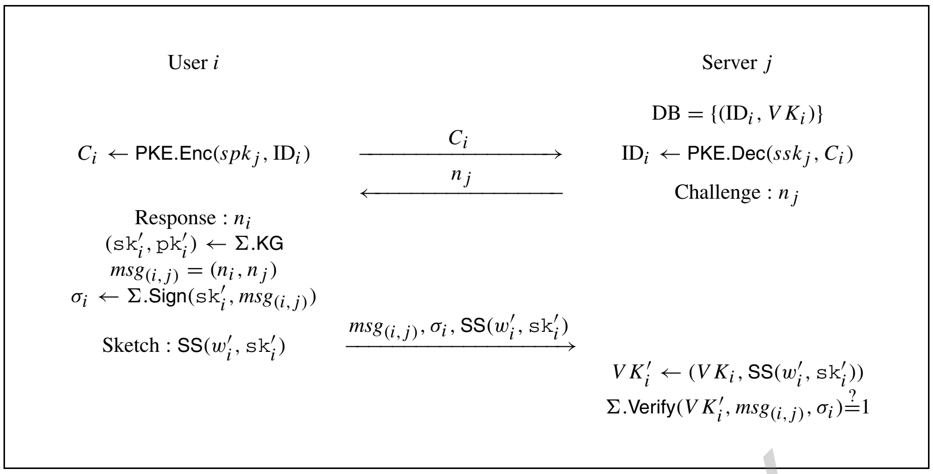
User i
Server j
$\mathrm{DB} = \{(\mathrm{ID}_i, VK_i)\}$
$C_i \leftarrow \mathsf{PKE.Enc}(spk_j, \mathrm{ID}_i)$
C_i
$\mathrm{ID}_i \leftarrow \mathsf{PKE.Dec}(ssk_j, C_i)$
n_j
Challenge : n_j
Response : n_i
$(\mathrm{sk}'_i, \mathrm{pk}'_i) \leftarrow \Sigma.\mathsf{KG}$
$msg_{(i,j)} = (n_i, n_j)$
$\sigma_i \leftarrow \Sigma.\mathsf{Sign}(\mathrm{sk}'_i, msg_{(i,j)})$
Sketch : $\mathsf{SS}(w'_i, \mathrm{sk}'_i)$
$msg_{(i,j)}, \sigma_i, \mathsf{SS}(w'_i, \mathrm{sk}'_i)$
$VK'_i \leftarrow (VK_i, \mathsf{SS}(w'_i, \mathrm{sk}'_i))$
$\Sigma.\mathsf{Verify}(VK'_i, msg_{(i,j)}, \sigma_i) \overset{?}{=} 1$

Fig. 3. Description of Authentication.

We use user i and server j to present our proposed construction.

- Setup. Let λ be the security parameter, and let universal hash function family be $\mathbb{H} : \{y = \mathrm{H}_{\mathbb{A}}(x)\}$, which involves a public matrix $\mathbb{A}$.
- KeyGen. User i obtains a key pair $(\mathrm{s}k_i, \mathrm{p}k_i) \leftarrow \Sigma.\mathsf{KG}$ by running the key generation algorithm of Σ. The server j obtains a key pair $(ssk_j, spk_j) \leftarrow \mathsf{PKE.KG}$ by running the key generation algorithm of PKE.
- Enrollment. A user i wants to enroll herself to an authentication server j performs the following steps

  ∗ compute a secret string $s_i \overset{\mathrm{R}}{\leftarrow} \mathrm{H}_{\mathbb{A}}^{-1}(\mathrm{s}k_i)$ from the secret key $\mathrm{s}k_i$, and derive a sketch $\mathsf{SS}(w_i, \mathrm{s}k_i) \leftarrow \mathsf{FE.Gen}(\mathbb{A}, w_i, s_i)$.
  ∗ send a reference tuple (ID_i, VK_i) to server j (note that server j maintains a database DB of all enrolled users, which includes verification keys $\{VK_i\} = \{(\mathrm{p}k_i, \mathsf{SS}(w_i, \mathrm{s}k_i))\}$).

- Authentication. The authentication between user i and server j is shown in Fig. 3.

  ∗ User i generates a ciphertext $C_i \leftarrow \mathsf{PKE.Enc}(spk_j, \mathrm{ID}_i)$ on her identity ID_i under the public key spk_j of server j, and sends it to server j.
  ∗ Server j obtains the identity $\mathrm{ID}_i \leftarrow \mathsf{PKE.Dec}(ssk_j, C_i)$ by running the decryption algorithm of PKE, and sends a challenge nonce n_j to user i.
  ∗ User i performs the following steps

    ∗ obtain a new key pair $(\mathrm{s}k'_i, \mathrm{p}k'_i) \leftarrow \Sigma.\mathsf{KG}$ and derive a new secret string $s'_i \overset{\mathrm{R}}{\leftarrow} \mathrm{H}_{\mathbb{A}}^{-1}(\mathrm{s}k'_i)$ from the new secret key $\mathrm{s}k'_i$.
    ∗ choose a response nonce n_i and generate a message-signature pair $(msg_{(i,j)}, \sigma_i) \leftarrow \Sigma.\mathsf{Sign}(\mathrm{s}k'_i, msg_{(i,j)})$, where $msg_{(i,j)} = (n_i, n_j)$.
    ∗ derive a new sketch $\mathsf{SS}(w'_i, \mathrm{s}k'_i) \leftarrow \mathsf{FE.Gen}(\mathbb{A}, w'_i, s'_i)$, and send $(msg_{(i,j)}, \sigma_i, \mathsf{SS}(w'_i, \mathrm{s}k'_i))$ to server j.

  ∗ Server j performs the following steps

∗ compute a "shift" secret between enrolled sketch and new sketch, where $\Delta(s) \leftarrow \mathsf{SS}(w_i, \mathtt{s}k_i) - \mathsf{SS}(w_i', \mathtt{s}k_i')$ (see correctness below).
∗ compute a "shift" secret key $\Delta(\mathtt{s}k) = \mathtt{H}_{\mathbb{A}}(\Delta(s))$, and derive a new public key $\mathtt{p}k_i'$ from the enrolled public key $\mathtt{p}k_i$ and the "shift" secret key $\Delta(\mathtt{s}k)$ (i.e., $M_{\mathtt{p}k_i}(pp, \mathtt{p}k_i, \Delta(\mathtt{s}k))$).
∗ verify the message-signature pair $(msg_{(i,j)}, \sigma_i)$ under the new public key $\mathtt{p}k_i'$. If the signature passes the verification, it accepts; Otherwise, it aborts.

Instantiations and Correctness. We hereby try to instantiate the underlying cryptographic building blocks which are able to resist quantum computers. First, to instantiate the computational fuzzy extractors from LWE, we rely on the reusable fuzzy extractors proposed by Apon et al. [2] (or the one called robustly reusable fuzzy extractor [32] with a non-uniform source). Second, we can implement the lattice-based digital signatures [11,21] as described in Section 2.4. Third, we could use the passively secure (i.e., IND-CPA) encryptions such as Regev's LWE-based cryptosystem [25] to instantiate the underlying public key encryptions. We notice that the passively secure encryptions will suffice for our defined user privacy model, and we refer to [24] for passively and actively secure cryptosystems which might be alternatively applicable to instantiate our proposed construction.

The public matrix includes $\mathbb{A} = (\mathbb{A}_1 \in \mathbb{Z}_{q^{mk}}, \mathbb{A}_2 \in \mathbb{Z}_q^{mk\times n}, \mathbb{A}_3 \in \mathbb{Z}_q^{m\times n})$ (the transpose of $\mathbb{A}_3$ is $\mathbb{A}_3^\top \in \mathbb{Z}_q^{n\times m}$). We set $m \geqslant c \cdot n$, where c is a positive constant, $n = n(\lambda)$ and $q = q(\lambda) \geqslant 2$ are two integers. Let the universal hash function be $\{\mathtt{H}_{(\mathbb{A}_1,\mathbb{A}_2)} : \mathbb{Z}_q^n \to \mathbb{Z}_q^{m\times k}\}_{\mathbb{A}_1\in Z_1,\mathbb{A}_2\in Z_2}$. For each seed pair $(\mathbb{A}_1, \mathbb{A}_2) \in (\mathbb{Z}_{q^{mk}}, \in \mathbb{Z}_q^{mk\times n})$ and $y \in \mathbb{Z}_q^{m\times k}$, we define "$\mathtt{H}^{-1}_{(\mathbb{A}_1,\mathbb{A}_2)}(y)$" as the set of pre-images of y under $\mathtt{H}_{(\mathbb{A}_1,\mathbb{A}_2)}$. That is, $\mathtt{H}^{-1}_{(\mathbb{A}_1,\mathbb{A}_2)}(y) = \{x \in \mathbb{Z}_q^n : \mathtt{H}_{\mathbb{A}_1,\mathbb{A}_2}(x) = y\}$. In particular, $x \overset{\mathrm{R}}{\leftarrow} \mathtt{H}^{-1}_{(\mathbb{A}_1,\mathbb{A}_2)}(y)$ means that we choose a vector x uniformly at random from set $\mathtt{H}^{-1}_{(\mathbb{A}_1,\mathbb{A}_2)}(y)$, and its size is $c \cdot k$.

The sketch $\mathsf{SS}(w_i, \mathtt{s}k_i)$ is instantiated with a random linear code over a finite field $\mathbb{Z}_q$. That is, $\mathsf{SS}(w_i, \mathtt{s}k_i) \leftarrow \mathbb{A}_3 \cdot s_i + w_i$, where biometrics $w_i \in \mathbb{Z}_q^m$ and secret $s_i \in \mathbb{Z}_q^n$. The correctness of $\Delta(s)$ relies on an algorithm $\mathsf{Decode}_t(\mathbb{A}_3, b)$ [13]. According to the Decode_t algorithm, we have the following equation with respect to the "shift" secret $\Delta(s) = s_i - s_i'$.

$$\begin{aligned}\mathsf{SS}(w_i, \mathtt{s}k_i) - \mathsf{SS}(w_i', \mathtt{s}k_i') &= \mathbb{A}_3 \cdot s_i + w_i - (\mathbb{A}_3 \cdot s_i' + w_i') = \mathbb{A}_3 \cdot (s_i - s_i') + (w_i - w_i') \\ &= \mathbb{A}_3 \cdot \Delta(s) + \delta.\end{aligned}$$

We assume that δ has at most t non-zero coordinates (i.e., at most t of non-zero coordinates can be zeroed out by $w_i - w_i'$), and notice that $\mathbb{A}_3 \cdot \Delta(s)$ is a linear system with m equations and $2n$ unknowns. If $m \geqslant 6n$ (we set $c = 6$ as suggested by [2] in order to ensure the Decode_t works with expected running time), then one can recover $\Delta(s)$ using the Decode_t algorithm with high probability (we refer to [13] for success probability and time complexity of Decode_t). Furthermore, we emphasize that the biometrics w_i, w_i' are computationally secure, because the server j is allowed to learn the "shift" secret $\Delta(s) = s_i - s_i'$ only.

5. Security analysis

In this section, we show the security result of our proposed construction.

Theorem 5.1. *The proposed RFS-RUA achieves user authenticity in the random oracle model if the family of universal hash functions $\mathbb{H} = \{\mathtt{H}_{(z_1,z_2)} : \mathbb{Z}_q^n \to \mathbb{Z}_q^{m\times k}\}_{z_1\in Z_1,z_2\in Z_2}$ is ϵ-statistically secure, the* $\mathrm{DLWE}_{q,n-k,m,\chi}$ *assumption is $(\epsilon, s'_{\mathrm{sec}})$ secure and the digital signature scheme Σ is EUF-CMA secure.*

Proof. We define a sequence of games $\{\mathbb{G}_i\}$ and let $\texttt{Adv}_i^{\text{RFS-RUA}}$ denote the advantage of the adversary $\mathcal{A}$ in game $\mathbb{G}_i$. Assume that $\mathcal{A}$ activates at most $m(\lambda)$ sessions in each game. We highlight the differences between adjacent games by underline.

- $\mathbb{G}_0$: This is the original game for user authenticity security.
- $\mathbb{G}_1$: This game is identical to game $\mathbb{G}_0$ except that the simulator $\mathcal{S}$ will output a random bit if server j accepts user i, but $\mathsf{sid}^l_{\text{ID}_i} \neq \mathsf{sid}^l_{\mathbb{S}_j}$. Since N users involved in this game, we have:

$$\left|\texttt{Adv}_0^{\text{RFS-RUA}} - \texttt{Adv}_1^{\text{RFS-RUA}}\right| \leqslant N \cdot m(\lambda)^2/2^{\lambda}. \tag{1}$$

- $\mathbb{G}_2$: This game is identical to game $\mathbb{G}_1$ except the following difference: $\mathcal{S}$ randomly chooses $g \in [1, m(\lambda)]$ as a guess for the index of the $\mathsf{Challenge}$ session. $\mathcal{S}$ will output a random bit if $\mathcal{A}$'s challenge query does not occur in the g-th session. Therefore we have

$$\texttt{Adv}_1^{\text{RFS-RUA}} = m(\lambda) \cdot \texttt{Adv}_2^{\text{RFS-RUA}}. \tag{2}$$

- $\mathbb{G}_3$: This game is identical to game $\mathbb{G}_2$ except that in the g-th session, the k-size pseudorandom bit of encrypted secret in the sketch $\mathsf{SS}(w_i, \mathrm{s}k'_i)$ of user i w.r.t. server j is replaced by a random value. Below we show that the difference between $\mathbb{G}_2$ and $\mathbb{G}_3$ is negligible under the $\text{DLWE}_{q,n-k,m,\chi}$ assumption.
  Let $\mathcal{S}$ denote a distinguisher against the $\text{DLWE}_{q,n-k,m,\chi}$ assumption, who is given a tuple $(\underline{X_{1,\ldots,k}}, \mathbb{A}, \mathbb{A}\cdot X+\chi)$, aims to distinguish the real LWE tuple from a random tuple $(\underline{U}, \mathbb{A}, \mathbb{A}\cdot X+\chi)$ where $U \in_R \mathbb{Z}_q^k$. $\mathcal{S}$ simulates the game for $\mathcal{A}$ as follows.
  - ∗ Setup. $\mathcal{S}$ sets up the game for $\mathcal{A}$ by creating N users and M servers with the corresponding identities. $\mathcal{S}$ randomly selects indexes (i, j) and guesses that the g-th session will happen with regard to user i at server j. $\mathcal{S}$ sets the sketch of user i w.r.t. server j as $\mathsf{SS}(w_i, \mathrm{s}k_b)$ such that $\mathsf{SS}(w_i, \mathrm{s}k_b) = \mathbb{A} \cdot \underline{X_b} + \chi$, where $\mathrm{s}k_b = \mathrm{H}_{(\mathbb{A}_1,\mathbb{A}_2)}(\underline{X_b})$ and $X_b = \underline{X_{1,\ldots,k}}$. $\mathcal{S}$ generates user i's enrolled public/secret key pair $(\mathrm{p}k^l_i, \mathrm{s}k^l_i)$ w.r.t. l servers $(l \neq j)$, and their corresponding sketches $\{\mathsf{SS}(w_i, \mathrm{s}k_b + \mathrm{s}k_l)\}$. In addition, $\mathcal{S}$ honestly generates biometrics for N-1 users, and generates enrolled public/secret key pairs and sketches as $\mathsf{Enrollment}$ specified for N-1 users w.r.t. M servers. Eventually, $\mathcal{S}$ sends all enrolled public keys and references to $\mathcal{A}$. $\mathcal{S}$ sets $\mathbb{A}_3 = \mathbb{A}$, and generates rest public parameters (including matrixes $\mathbb{A}_1, \mathbb{A}_2$) for the system. $\mathcal{S}$ also chooses a random vector from $\mathbb{Z}_q^{n-k}$ to construct $X_b \in \mathbb{Z}_q^n$, we omit it in the following proof for simplicity.
  - ∗ $\mathsf{Training}$. $\mathcal{S}$ answers $\mathcal{A}$'s queries as follows.
    - ∗ If $\mathcal{A}$ issues a Send query in the form of n_j to $\mathcal{S}$, $\mathcal{S}$ chooses a response nonce n_i first, then $\mathcal{S}$ honestly generates the protocol transcript T_i using user i's enrolled secret key $\mathrm{s}k_b$ and sketch $\mathsf{SS}(w_i, \mathrm{s}k_b)$. Specifically, $T_i = (msg_{(i,j)}, \sigma_i, \mathsf{SS}(w_i, \mathrm{s}k'_i))$, where $\sigma_i \leftarrow M_\Sigma(pp, \mathrm{p}k_b, \mathsf{Sign}(\mathrm{s}k_b, msg_{(i,j)}), \Delta(s_i))$, $\mathsf{SS}(w_i, \mathrm{s}k'_i) \leftarrow \mathbb{A} \cdot X_b + \chi + \mathbb{A} \cdot \Delta(s_i)$, where $\mathrm{s}k'_i \leftarrow \mathrm{H}_{(\mathbb{A}_1,\mathbb{A}_2)}(s_b + \Delta(s_i))$, $s_b \overset{R}{\leftarrow} \mathrm{H}^{-1}_{(\mathbb{A}_1,\mathbb{A}_2)}(\mathrm{s}k_b)$ and $\Delta(s_i) \in_R \mathbb{Z}_q^n$ is chosen by $\mathcal{S}$.
      As for user i's g-th session w.r.t. server j, $\mathcal{S}$ first generates $\mathsf{SS}(w_i, \mathrm{s}k'_b) \leftarrow \mathbb{A}\cdot X_b+\chi+\mathbb{A}\cdot\Delta(s)$, and denotes $X'_b = X_b + \Delta(s)$ and $X_b = \underline{U}$; $\mathcal{S}$ then generates a key pair $(\mathrm{s}k'_b, \mathrm{p}k'_b)$ from X'_b, where $\mathrm{s}k'_b = \mathrm{H}_{(\mathbb{A}_1,\mathbb{A}_2)}(X'_b)$ and $\mathrm{p}k'_b = \mathbb{A}^\top \cdot \mathrm{s}k'_b$; eventually, $\mathcal{S}$ honestly generates the message-signature pair according to the protocol specification, and returns the protocol transcript to $\mathcal{A}$.

* If $\mathcal{A}$ issues a Biometrics Reveal query to user i, then $\mathcal{S}$ aborts.
* If $\mathcal{A}$ issues a Secret Key Reveal query to an instance oracle $\Pi^{g}_{\mathrm{ID}_i}$ (g-th session of user i w.r.t. server j), then $\mathcal{S}$ returns new secret key $\mathtt{s}k'_b$ to $\mathcal{A}$.
* If $\mathcal{A}$ issues Biometrics Shift query in the form of δ to $\mathcal{S}$, then $\mathcal{S}$ returns $\mathsf{SS}(w_i + \delta, \mathtt{s}k) = \mathbb{A} \cdot \underline{X_b} + \chi + \delta$ by the linearity of the sketch, where $\mathtt{s}k$ can be either enrolled secret key $\mathtt{s}k_b$ or new secret key $\mathtt{s}k'_b$ that involves at g-th session.
* If $\mathcal{A}$ issues Secret Key Shift query in the form of ϕ to $\mathcal{S}$, then $\mathcal{S}$ returns new public key $\mathtt{p}k'_b \leftarrow \mathsf{KG}(pp, \phi(\mathtt{s}k_b))$. Notice that $\mathcal{A}$ is not allowed to obtain user i's enrolled secret key $\mathtt{s}k_b$.

Note that in the Challenge session of user i w.r.t. server j, if the challenge of $\mathcal{S}$ is $\overline{X_{1,\ldots,k}}$, then the simulation is consistent with $\mathbb{G}_2$; Otherwise, the simulation is consistent with $\mathbb{G}_3$. If the advantage of $\mathcal{A}$ is significantly different in $\mathbb{G}_2$ and $\mathbb{G}_3$, then $\mathcal{S}$ can break the $\mathrm{DLWE}_{q,n-k,m,\chi}$. Since at most N users involved in the system, hence we have

$$\left|\mathtt{Adv}_2^{\text{RFS-RUA}} - \mathtt{Adv}_3^{\text{RFS-RUA}}\right| \leqslant N \cdot \mathtt{Adv}_{\mathcal{S}}^{\mathrm{DLWE}_{q,n-k,m,\chi}}(\lambda). \tag{3}$$

- $\mathbb{G}_4$: This game is identical to game $\mathbb{G}_3$ except that in the g-th session, the enrolled secret key $\mathtt{s}k_i^j$ w.r.t., server j is replaced by a random value. Below we show that the difference between $\mathbb{G}_3$ and $\mathbb{G}_4$ is bounded by a negligible probability.
  Let $\mathcal{S}$ simulate the whole environment honestly according to the protocol specification, and it is easy to see that all the queries made to a user can be simulated perfectly using the user's secret keys and biometrics. In particular, the enrolled secret key of user i w.r.t. server j is $\underline{\mathtt{s}k_i^j}$. In the g-th session of user i w.r.t server j, to answer the Send query from $\mathcal{A}$, $\mathcal{S}$ will simulate the protocol transcript T'_i as follows. $\mathcal{S}$ first simulates the sketch as $\mathsf{SS}(w_i, \mathtt{s}k'_i) \leftarrow \mathbb{A}_3 \cdot (s_i + \Delta(s)) + w_i$, where $\mathtt{s}k'_i \leftarrow \mathtt{H}_{(\mathbb{A}_1,\mathbb{A}_2)}(s_i + \Delta(s))$, $s_i \overset{\mathrm{R}}{\leftarrow} \mathtt{H}^{-1}_{(\mathbb{A}_1,\mathbb{A}_2)}(\underline{u})$, $u \in_R \mathbb{Z}_q^{m\times k}$, and $\Delta(s)$ is randomly chosen by $\mathcal{S}$; $\mathcal{S}$ then generates a key pair $(\mathtt{s}k'_i, \mathtt{p}k'_i)$ from $s_i + \Delta(s)$; eventually, $\mathcal{S}$ honestly generates the message-signature pair using the same method described in previous game $\mathbb{G}_3$.
  We then analyze the statistical distance between two distributions $T'_i = (msg_{(i,j)}, \sigma_i, \mathsf{SS}(w_i, \mathtt{s}k'_i))$ and T_i (of previous game $\mathbb{G}_3$). We notice that the only difference is the simulated value $s_i \overset{\mathrm{R}}{\leftarrow} \mathtt{H}^{-1}_{(\mathbb{A}_1,\mathbb{A}_2)}(\underline{u})$ instead of taking the real enrolled secret key $\mathtt{s}k_i^j$ as input, and according to Lemma 2.2, we have the statistically distance between $\mathtt{s}k_i^j \leftarrow \mathtt{H}_{(\mathbb{A}_1,\mathbb{A}_2)}(s_i)$ and $u \in_R \mathbb{Z}_q^{m\times k}$ with probability no greater than ϵ. Since at most M servers involved in the system, hence we have

$$\left|\mathtt{Adv}_3^{\text{RFS-RUA}} - \mathtt{Adv}_4^{\text{RFS-RUA}}\right| \leqslant M \cdot \mathtt{Adv}_{\mathcal{S}}^{\mathbb{H}}(\lambda). \tag{4}$$

- $\mathbb{G}_5$: This game is identical to game $\mathbb{G}_4$ except that in the g-th session, $\mathcal{S}$ outputs a random bit if **Forge** event happens where $\mathcal{A}$'s Send query includes a valid forgery σ^* while user i's secret key w.r.t server j is not corrupted. Then we have

$$\left|\mathtt{Adv}_4^{\text{RFS-RUA}} - \mathtt{Adv}_5^{\text{RFS-RUA}}\right| \leqslant \Pr[\textbf{Forge}]. \tag{5}$$

Let $\mathcal{F}$ denote a forger against a (lattice-based) signature scheme Σ with EUF-CMA security, who is given a verification key $\mathtt{p}k^*$ and a signing oracle $\mathcal{O}$, and aims to find a forgery σ^*. $\mathcal{S}$ simulates the game for $\mathcal{A}$ as follows.

* Setup. $\mathcal{F}$ sets up the game for $\mathcal{A}$ by creating N users and M servers with the corresponding identities and biometrics. $\mathcal{F}$ sets up the verification key of user i w.r.t. server j as $VK_i^j = (\mathrm{p}k^*, \mathsf{SS}(w_i, \mathrm{s}k_i^j))$, where $\mathsf{SS}(w_i, \mathrm{s}k_i^j) = \mathbb{A}_3 \cdot s_i + w_i, s_i \in_R \mathbb{Z}_q^n$. $\mathcal{F}$ also honestly generates public/secret key pairs and sketches as Enrollment specified for N-1 users and M servers. Eventually, $\mathcal{F}$ sends all enrolled public keys and references to $\mathcal{A}$. Note that $\mathcal{F}$ honestly generates all the public parameters (including matrixes $\mathbb{A}_1, \mathbb{A}_2, \mathbb{A}_3$) in the system. Also note that $\mathcal{A}$ cannot link the simulated sketch $\mathsf{SS}(w_i, \mathrm{s}k_i^j)$ and public key $\mathrm{p}k^*$ since $\mathcal{A}$ is not allowed to access biometrics w_i.
* Training. $\mathcal{F}$ answers $\mathcal{A}$'s queries as follows.
  * If $\mathcal{A}$ issues a Send query in the form of n_j to $\mathcal{F}$, $\mathcal{F}$ chooses a response nonce n_i first, then $\mathcal{F}$ simulates the protocol transcript $T_i = (\sigma_i', \mathsf{SS}(w_i, \mathrm{s}k_i'))$ as follows.
    (1) invoke the signing oracle $\mathcal{O}$ to obtain a message-signature pair $(msg_{(i,j)}, \sigma_i)$, where $msg_{(i,j)} = (n_i, n_j)$;
    (2) generate a sketch $\mathsf{SS}(w_i, \mathrm{s}k_i') \leftarrow \mathsf{SS}(w_i, \mathrm{s}k_i^j) + \mathbb{A}_3 \cdot \Delta(s_i)$, where $\Delta(s_i)$ is randomly chosen by $\mathcal{F}$;
    (3) generate a signature $\sigma_i' \leftarrow M_\Sigma(pp, \mathrm{p}k^*, \sigma_i, \Delta(s_i))$ by using the deterministic algorithm M_Σ described in Section 3;
    (4) returen $(m_{(i,j)}, \sigma_i', \mathsf{SS}(w_i, \mathrm{s}k_i'))$ to $\mathcal{A}$.
  * If $\mathcal{A}$ issues a Secret Key Reveal query to an instance oracle $\Pi_{\mathrm{ID}_i}^i$, then $\mathcal{F}$ returns new secret key $\mathrm{s}k_i' \in_R \mathbb{Z}_q^{m\times k}$ to $\mathcal{A}$. Since $\mathcal{A}$ is not allowed to reveal the enrolled secret key $Dlog(\mathrm{p}k^*)$ (of user i w.r.t. server j), the simulation is perfect.
  * If $\mathcal{A}$ issues Biometrics Shift query in the form of δ to $\mathcal{F}$, then $\mathcal{F}$ returns $\mathsf{SS}(w_i + \delta, \mathrm{s}k_i')$ to $\mathcal{A}$. Notice that $\mathcal{A}$ is not allowed to obtain user i's biometrics w_i.
  * If $\mathcal{A}$ issues Secret Key Shift query in the form of ϕ to $\mathcal{F}$, then $\mathcal{F}$ returns new public key $\mathrm{p}k^{*'} \leftarrow \mathsf{KG}(pp, Dlog(\mathrm{p}k^*), \phi(\Delta(\mathrm{s}k)))$, where $\Delta(\mathrm{s}k)$ is chosen by $\mathcal{A}$.
* When **Forge** event occurs (i.e., $\mathcal{A}$ outputs $(msg^*, \sigma^{*'}, \mathsf{SS}(w_i, \mathrm{s}k^{*'})))$, $\mathcal{F}$ checks whether:
  * the **Forge** event happens at g-th session;
  * the message-signature pair $(msg^*, \sigma^{*'})$ is not previously simulated by $\mathcal{S}$;
  * verifies $\Sigma.\mathsf{Verify}(\mathrm{p}k^{*'}, msg^*, \sigma^{*'}) \overset{?}{=} 1$, where $\mathrm{p}k^{*'} \leftarrow \mathrm{p}k^* + \mathbb{A}_3^\top \cdot \underline{\Delta(\mathrm{s}k^*)}$, $\underline{\Delta(\mathrm{s}k^*)} = \mathrm{H}_{(\mathbb{A}_1,\mathbb{A}_2)}(\Delta(s^*)), \Delta(s^*) \leftarrow \mathsf{SS}(w_i, \mathrm{s}k^{*'}) - \mathsf{SS}(w_i, \mathrm{s}k_i^j)$. Note that $\underline{\Delta(\mathrm{s}k^*)}$ is the correct "shift" between $Dlog(\mathrm{p}k^{*'})$ and $\mathrm{s}k_i^j$.

  If all the above conditions hold, $\mathcal{F}$ confirms that it as a successful forgery from $\mathcal{A}$, then $\mathcal{F}$ extracts the forgery via $\sigma^* \leftarrow M_\Sigma(pp, \mathrm{p}k^*, \sigma^{*'}, \underline{\Delta(\mathrm{s}k^*)})$ by using the homomorphic property of Σ (Lemma 3). To this end, $\mathcal{F}$ outputs σ^* as its own forgery; Otherwise, $\mathcal{F}$ aborts the game. Therefore, we have

$$\left|\Pr[\mathbf{Forge}]\right| \leqslant \mathtt{Adv}_{\mathcal{F}}^{\text{EUF-CMA}}(\lambda). \tag{6}$$

Combining the above results together, we have

$$\mathtt{Adv}_{\mathcal{A}}^{\text{RFS-RUA}}(\lambda)$$

$$\leqslant N \cdot m(\lambda)^2/2^\lambda + m(\lambda)\big[N \cdot \mathtt{Adv}_{\mathcal{S}}^{\mathrm{DLWE}_{q,n-k,m,\chi}}(\lambda) + M \cdot \mathtt{Adv}_{\mathcal{S}}^{\mathbb{H}}(\lambda) + \mathtt{Adv}_{\mathcal{F}}^{\text{EUF-CMA}}(\lambda)\big]. \quad \square$$

Theorem 5.2. *The proposed RFS-RUA achieves user privacy in the random oracle model if the decisional $SIS_{q,n,m,d}$ assumption is $(\epsilon, s_{\text{sec}})$ secure, the public key encryption is IND-CPA secure and the computational fuzzy extractor is IND secure.*

Proof. We define a sequence of games $\{\mathbb{G}_i\}$ and let $\mathtt{Adv}_i^{\text{RFS-RUA}}$ denote the advantage of the adversary $\mathcal{A}$ in game $\mathbb{G}_i$. We also highlight the differences between adjacent games by underline.

- $\mathbb{G}_0$: This is the original game for user privacy.
- $\mathbb{G}_1$: This game is identical to game $\mathbb{G}_0$ except that at challenge stage, $\mathcal{S}$ replaces the real identity (message of ciphertext C_i) by random string R. Below we show that the difference between $\mathbb{G}_0$ and $\mathbb{G}_1$ is negligible under the assumption that the public key encryption scheme is IND-CPA secure.
  Let $\mathcal{S}$ denote an attacker who is given a public key $\mathtt{p}k^*$, aims to break the IND-CPA security of the public key encryption scheme. $\mathcal{S}$ simulates the game for $\mathcal{A}$ as follows.

  $*$ Setup. $\mathcal{S}$ sets up the game for $\mathcal{A}$ by creating N users and M servers. $\mathcal{S}$ honestly generates biometrics for N users, generates key pairs with respect to M servers and their corresponding sketches. $\mathcal{S}$ randomly selects index j and guesses the challenge will happen with regard to server j. $\mathcal{S}$ sets the public key of server j as $\mathtt{p}k^*$, and honestly generates key pairs for M-1 servers. It is obvious that $\mathcal{S}$ can easily simulate the protocol execution of N users and M-1 servers except server j. Below we mainly focus on the simulation of server j.

  $*$ Training. If $\mathcal{A}$ issues an Execute query between user i and server j, then $\mathcal{S}$ randomly chooses nonces (n_i, n_j) and new key pairs, and performs the session execution honestly according to the protocol specification.

  $*$ Challenge. Upon receiving challenge candidates $(\mathrm{ID}_0, \mathrm{ID}_1) \in \mathcal{U}_0'$ from $\mathcal{A}$, $\mathcal{S}$ follows the security game to select ID_b^*. Then $\mathcal{S}$ executes the RFS-RUA protocol to generate the protocol transcript. After that, $\mathcal{S}$ generates another random string $\underline{R}$ and sends ID_b and $\underline{R}$ as the challenge messages to its own oracle. After receiving the challenge ciphertext $\underline{C^*}$ from his own challenger, $\mathcal{S}$ replaces the ciphertext generated by ID_b in the first message of the transcript by $\underline{C^*}$. Eventually, $\mathcal{S}$ sends the complete transcript to $\mathcal{A}$.

  Finally, $\mathcal{S}$ outputs whatever $\mathcal{A}$ outputs. If $\mathcal{A}$ guesses the random bit correctly, then $\mathcal{S}$ can break the IND-CPA security of public key encryption scheme. Since at most M servers in the system, and at most $\mathbb{K} = \{0, 1\}^{|\mathrm{ID}|}$ encryptions are executed (because each ciphertext encrypts a single bit of real identity), we have

$$\left|\mathtt{Adv}_0^{\text{RFS-RUA}} - \mathtt{Adv}_1^{\text{RFS-RUA}}\right| \leqslant M \cdot \mathbb{K} \cdot \mathtt{Adv}_{\mathcal{S}}^{\text{IND-CPA}}(\lambda). \tag{7}$$

- $\mathbb{G}_2$: This game is identical to game $\mathbb{G}_1$ except that at challenge stage, $\mathcal{S}$ replaces the sketch $\mathsf{SS}(w_i', \mathtt{s}k_i')$ by a random distribution over $\mathbb{Z}_q^m$. Below we show that the difference between $\mathbb{G}_1$ and $\mathbb{G}_2$ is negligible under the assumption that computational fuzzy extractor is IND secure.
  Let $\mathcal{S}$ denote an attacker who is given a challenge sketch (assuming a common public matrix $\mathbb{A}$ here, and $p \leftarrow \mathbb{A} \cdot X + \chi$), aims to break the IND security of the computational fuzzy extractor. $\mathcal{S}$ simulates the game for $\mathcal{A}$ as follows.

- $*$ Setup. $\mathcal{S}$ sets up the game for $\mathcal{A}$ by creating N users and M servers. $\mathcal{S}$ honestly generates key pairs for M servers. $\mathcal{S}$ randomly selects indexes (i, j) and guesses the challenge will happen with regard to user i and server j. $\mathcal{S}$ sets the enrolled sketch of user i with respect to server j as p_0, and generates the enrolled key pair at random (which is a matrix pair over distribution $(\mathbb{Z}_q^{m\times k}, \mathbb{Z}_q^{n\times k})$). Additionally, $\mathcal{S}$ honestly generates biometrics, key pairs (w.r.t. M servers) and sketches for N-1 users. It is obvious that $\mathcal{S}$ can easily simulate all protocol executions except user i w.r.t server j. Below we mainly focus on the simulation between user i and server j. Note that $\mathbb{A}_3^\top = \underline{\mathbb{A}^\top}$.
- $*$ Training. If $\mathcal{A}$ issues an Execute query, then $\mathcal{S}$ simulates the session execution as follows
  (1) generate a ciphertext C_i on the identity of user i;
  (2) choose nonces n_i, n_j and form a message $msg_i = (n_i, n_j)$;
  (3) choose a new key pair $(\mathtt{s}k_i', \mathtt{p}k_i')$ and generate message-signature pair using the secret key $\mathtt{s}k_i'$;
  (4) compute "shift" $\Delta(\mathtt{s}k_i)$ from (enrolled public key) $\mathtt{p}k_i^j$ and $\mathtt{p}k_i'$;
  (5) simulate a new sketch $\mathsf{SS}(w_i, \mathtt{s}k_i') = \mathtt{p}k_0 + \mathbb{A}^\top \cdot \Delta(\mathtt{s}k_i)$;
  (6) sends the complete transcript to $\mathcal{A}$.
- $*$ Challenge. Upon receiving challenge candidates $(\mathrm{ID}_0, \mathrm{ID}_1) \in \mathcal{U}_0'$ from $\mathcal{A}$, $\mathcal{S}$ follows the security game to select ID_b^* and server j (from honest set $\mathcal{U}_1'$). Then $\mathcal{S}$ executes the RFS-RUA protocol to generate the protocol transcript. After that, $\mathcal{S}$ replaces the simulated sketch associated with ID_b^* in the third message of the transcript by $\underline{C^*}$. Eventually, $\mathcal{S}$ sends the complete transcript to $\mathcal{A}$.

Finally, $\mathcal{S}$ outputs whatever $\mathcal{A}$ outputs. If $\mathcal{A}$ guesses the random bit correctly, then $\mathcal{S}$ can break the IND security of the computational fuzzy extractors. Since at most N users and M servers in the system, we have

$$\left|\mathtt{Adv}_1^{\text{RFS-RUA}} - \mathtt{Adv}_2^{\text{RFS-RUA}}\right| \leqslant N \cdot M \cdot \mathtt{Adv}_{\mathcal{S}}^{\text{IND}}(\lambda). \tag{8}$$

- $\mathbb{G}_3$: This game is identical to game $\mathbb{G}_2$ except that at challenge stage, $\mathcal{S}$ replace the real verification key of digital signature σ_i by a random key over $\mathbb{Z}_q^{n\times k}$. Below we show that the difference between $\mathbb{G}_2$ and $\mathbb{G}_3$ is negligible under the assumption that the decisional $\mathrm{SIS}_{q,n,m,d}$ is hard.
  Let $\mathcal{S}$ denote a decisional SIS problem distinguisher, who is given a pair $(\mathbb{A}^*, t^*)$, aims to decide it whether from a $\mathrm{SIS}_{q,n,m,d}$ distribution or from a random distribution over $\mathbb{Z}_q^{n\times m} \times \mathbb{Z}_q^n$. $\mathcal{S}$ simulates the game for $\mathcal{A}$ as follows.
  - $*$ Setup: $\mathcal{S}$ sets up the game for $\mathcal{A}$ by creating N users and M servers with the corresponding identities and biometrics. $\mathcal{S}$ generates enrolled public/secret key pairs $(\mathtt{p}k_i^j, \mathtt{s}k_i^j)$ and sketches $\mathsf{SS}(w_i, \mathtt{s}k_i^j)$ for N users and M servers. Eventually, $\mathcal{S}$ sends all identities, public keys and sketches to $\mathcal{A}$. In particular, $\mathcal{S}$ sets $\mathbb{A}_3^\top = \underline{\mathbb{A}^{*\top}}$ and honestly generates other public parameters (such as matrixes $\mathbb{A}_1, \mathbb{A}_2$) for the system. Note that $\mathtt{p}k_i^j = \mathbb{A}^{*\top} \cdot \mathtt{s}k_i^j$.
  - $*$ Training: $\mathcal{S}$ honestly simulates the session execution (using user's secret keys and biometrics) according to the protocol specification.
  - $*$ Challenge: Upon receiving challenge candidates $(\mathrm{ID}_0, \mathrm{ID}_1) \in \mathcal{U}_0'$ from the $\mathcal{A}$, $\mathcal{S}$ follows the security game to select ID_b^* and server j. $\mathcal{S}$ simulates the the transcript as follows.
    (1) generate a ciphertext C_i on the identity of user i;
    (2) choose $s' \in_R \mathbb{Z}_q^n$ and compute a new sketch $\mathsf{SS}(w_b, \mathtt{s}k_b') \leftarrow \mathbb{A}^* \cdot s' + w_b$;

(3) choose nonces n_i, n_j and form a message $msg_{(i,j)} = (n_i, n_j)$;
(4) generate a message-signature pair $(msg_{(i,j)}, \sigma_b^*)$ (by design, $\sigma_b^* = (z_b^*, c_b^*)$) using the same method described in the security proof of [21]; Note that the corresponding public (verification) key is $\mathtt{p}k_b^{*'} = \underline{U^*} + \mathbb{A}_3^\top \cdot \Delta(\mathtt{s}k)$, where $\Delta(\mathtt{s}k)$ derives from public sketches.
(5) return $(C_i, msg_{(i,j)}, \underline{\sigma_b^*}, \mathsf{SS}(w_b, \mathtt{s}k'_b))$ as the complete transcript.

Finally, $\mathcal{S}$ outputs whatever $\mathcal{A}$ outputs. Notice that the enrolled public key $\mathtt{p}k_b$ is replaced by the given instance $\underline{U^*}$ in the Challenge stage. Moreover, the simulated message-signature pair $(msg_{(i,j)}, \sigma_b^*)$ can be successfully verified under verification key $\mathtt{p}k_b^{*'}$ (by programming random oracle such that $\mathtt{H}(\mathbb{A}^* \cdot z_b^* - \mathtt{p}k_b^{*'} \cdot c_b^*, msg_{(i,j)}) = c_b^*$). As for the matrix pair $(\mathbb{A}^*, U^*)$, according to the hybrid argument, distinguishing the real public key from a uniform distribution $(\mathbb{A}^*, U^*) \in \mathbb{Z}_q^{n\times m} \times \mathbb{Z}_q^{n\times k}$ is as hard as the $\mathrm{SIS}_{q,n,m,d}$ decisional problem (but with a loss of factor k in the advantage). Therefore, the two verification keys $\mathtt{p}k_b^{*'}$ ($b = [0, 1]$) are statistically indistinguishable w.r.t. high-density SIS or computational indistinguishable w.r.t. low-density SIS. Since at most N users and M servers in the system, we have

$$\left|\mathtt{Adv}_2^{\text{RFS-RUA}} - \mathtt{Adv}_3^{\text{RFS-RUA}}\right| \leqslant N \cdot M \cdot k \cdot \mathtt{Adv}_{\mathcal{S}}^{\mathrm{SIS}_{q,n,m,d}}(\lambda). \tag{9}$$

Combining the above results together, we have

$$\mathtt{Adv}_{\mathcal{A}}^{\text{RFS-RUA}}(\lambda) \leqslant M \cdot \mathbb{K} \cdot \mathtt{Adv}_{\mathcal{S}}^{\text{IND-CPA}}(\lambda) + N \cdot M \cdot \big(\mathtt{Adv}_{\mathcal{S}}^{\text{IND}}(\lambda) + k \cdot \mathtt{Adv}_{\mathcal{S}}^{\mathrm{SIS}_{q,n,m,d}}(\lambda)\big). \qquad \square$$

6. Conclusion

In this work, we have proposed a lattice-based construction for remote user authentication from reusable fuzzy signature RFS. First, the RFS-based remote user authentication had biometrics reusability, such that the user's biometrics can be reused multiple times to secure user authentication. Second, the RFS-based remote user authentication had a privacy guarantee against the eavesdroppers. Third, the RFS-based remote user authentication had proven secure in the random oracle model under our defined security models (user authenticity and user privacy). As our future work, we leave the construction of RFS based on efficient ring-SIS or ring-LWE [15,24].

Acknowledgments

This work is partially supported by enPiT(Education Network for Practical Information Technologies) at MEXT, and Innovation Platform for Society 5.0 at MEXT. Yingjiu Li was supported in part by the Ripple University Blockchain Research Initiative. Robert Deng was supported in part by the AXA Research Fund.

Appendix A. Privacy concerns on [22,28,29]

We assume the Enrollment stage is also executed in a secure channel, which means the attackers cannot access user's enrolled public keys. According to the generic construction in [22] (also applicable to [28]), we assume an authorized user Alice wants to authenticate herself to an authentication server, and produces a transcript $\underline{(\mathsf{pk}'_A, \mathsf{SS}(w'_A, \mathsf{sk}'_A), \sigma'_A)}$ in one session, where the new public key is $\mathsf{pk}'_A = g^{\mathsf{sk}'_A}$. In another session, suppose a challenge identity ID^*_b generates a transcript $(\mathsf{pk}^*_{\mathrm{ID}_b}, \mathsf{SS}(w^*_{\mathrm{ID}_b}, \mathsf{sk}^*_{\mathrm{ID}_b}), \sigma^*_{\mathrm{ID}_b})$, where the new public key is $\mathsf{pk}^*_{\mathrm{ID}_b} = g^{\mathsf{sk}^*_{\mathrm{ID}_b}}$. Then attacker can link the challenge identity ID^*_b to user Alice. Specifically, an attacker will verify the signature $\sigma^*_{\mathrm{ID}_b}$ using the new public key $\mathsf{pk}^*_{\mathrm{ID}_b}$ first; then compute the "difference" $\Delta(\mathsf{sk}^*_{\mathrm{ID}_b}) \leftarrow \mathsf{SS}(w^*_{\mathrm{ID}_b}, \mathsf{sk}^*_{\mathrm{ID}_b}) - \mathsf{SS}(w'_A, \mathsf{sk}'_A)$; eventually, attachers can verify the relationship between new public keys $\mathsf{pk}^*_{\mathrm{ID}_b}$ and pk'_A via a "difference reconstruction" algorithm $M_{\mathsf{pk}'_A}(pp, \mathsf{pk}'_A, \Delta(\mathsf{sk}^*_{\mathrm{ID}_b})) \overset{?}{=} \mathsf{pk}^*_{\mathrm{ID}_b}$. Notice that if algorithm $M_{\mathsf{pk}'_A}$ outputs 1, then attacker can conclude that the challenge user ID_b is user Alice. The key point here is that the new public key of user Alice pk'_A in one session actually acts as an enrolled public key, and can be easily linked with the new public key $\mathsf{pk}'_{\mathrm{ID}_b}$ in another session of the same user, assuming user Alice (with biometrics w'_A and $\mathsf{dist}(w'_A, w^*_{\mathrm{ID}_b}) \leqslant t$) is successfully authenticated by an authentication server.

Our Treatment. We modify the verification process, in which the verification of signature requires the enrolled public key of user Alice pk_A. We follow the example above, user Alice first generates her new secret key sk'_A for signing a message, then computes a new sketch $\mathsf{SS}(w'_A, \mathsf{sk}'_A)$. The user Alice sends the transcript $\underline{(\mathsf{SS}(w'_A, \mathsf{sk}'_A), \sigma'_A)}$ to an authentication server. The authentication server first obtains the "difference" value $\Delta(\mathsf{sk}_A)$ from enrolled and new sketches, then computes her new public key $\mathsf{pk}'_A \leftarrow M_{\mathsf{pk}_A}(pp, \mathsf{pk}_A, \Delta(\mathsf{sk}_A))$ and verifies the message-signature pair (m, σ'_A) using the corresponding new public key pk'_A. Notice that the major difference between two transcripts (underline parts) is: our treatment does not transmit new public key in the public channel. In fact, the verification of message-signature pair is based on the designated verifier concept [17] such that the authentication server who holds the enrolled public key can verify it. We stress that the publicly verifiable of message-signature pair is the main privacy concern in [22,28,29].

Appendix B. Fuzzy extractor

A fuzzy extractor converts a non-uniform data to uniformly random strings, which can be used in cryptographic applications. A typical application of fuzzy extractor is to extract reproducible string from biometric information. Then, the extracted string is considered as a secret for user authentication.

Secure sketch and fuzzy extractor. Secure sketch is a building block of fuzzy extractors. A secure sketch scheme takes noisy information w as input, outputs a sketch SS which is an auxiliary string. Secure sketch schemes normally use error correcting techniques to recover w given SS and w', if the given input w' is statistically close to w. The sketch SS can be published since it does not reveal any information about w. Let $\mathcal{M}$ be a metric space on N points with distance function $\mathsf{dist} : \mathcal{M} \times \mathcal{M} \to \mathbb{R}^+ = [0, \infty)$, where $N = |\mathcal{M}|$. A secure sketch is defined below.

Definition B.1. A secure sketch consists of two randomized procedures $(\mathsf{SS}, \mathsf{Rec})$.

- The procedure SS takes $w \in \mathcal{M}$ as input, outputs a sketch $\mathsf{SS} \in \{0, 1\}^*$.
- The procedure Rec takes $w' \in \mathcal{M}$ and a sketch SS as input, outputs w if $\mathsf{dist}(w, w') \leqslant t$.

Fuzzy extractor extracts a non-uniform string from a noisy input $w \in \mathcal{M}$. The difference is that fuzzy extractor returns a uniform string, but secure sketch returns a non-uniform string.

Definition B.2. A fuzzy extractor consists of two randomized procedures $(\mathsf{Gen}, \mathsf{Rep})$.

- The generation procedure Gen takes $w \in \mathcal{M}$ as input, outputs a secret string $R \in \{0, 1\}^{\ell}$, and public helper data $P \in \{0, 1\}^*$ such that $(R, P) \leftarrow \mathsf{Gen}(w)$.
- The reproduction procedure Rep takes $w' \in \mathcal{M}$ and helper data P as input, outputs R such that $R \leftarrow \mathsf{Rep}(w', P)$ iff $\mathsf{dist}(w, w') \leqslant t$.

Since sketch reconstructs the original input from noisy data, it can be used to construct fuzzy extractor. So, the helper data P contains sketch SS.

Appendix C. Digital signature

Definition C.1. A digital signature $\Sigma = (\mathsf{KG}, \mathsf{Sign}, \mathsf{Verify})$ is defined as follows.

- KG: The algorithm takes a security parameter λ as input, outputs a key pair $(\mathtt{s}k, \mathtt{p}k)$.
- Sign: The algorithm takes a secret key $\mathtt{s}k$ and a message *msg* as input, outputs a signature $\sigma \leftarrow \Sigma.\mathsf{Sign}(\mathtt{s}k, msg)$.
- Verify: The algorithm takes a public key $\mathtt{p}k$ and a message *msg* as input, outputs "1" if $1 \leftarrow \Sigma.\mathsf{Verify}(\mathtt{p}k, msg, \sigma)$.

References

[1] A. Akavia, S. Goldwasser and V. Vaikuntanathan, Simultaneous hardcore bits and cryptography against memory attacks, in: *TCC*, 2009, pp. 474–495.

[2] D. Apon, C. Cho, K. Eldefrawy and J. Katz, Efficient, reusable fuzzy extractors from LWE, in: *International Conference on Cyber Security Cryptography and Machine Learning*, 2017, pp. 1–18.

[3] M. Barni, T. Bianchi, D. Catalano, M. Di Raimondo, R. Donida Labati, P. Failla, D. Fiore, R. Lazzeretti, V. Piuri, F. Scotti et al., Privacy-preserving fingercode authentication, in: *Proceedings of the 12th ACM Workshop on Multimedia and Security*, 2010, pp. 231–240. doi:10.1145/1854229.1854270.

[4] M. Bellare, D. Cash and R. Miller, Cryptography secure against related-key attacks and tampering, in: *ASIACRYPT*, 2011, pp. 486–503.

[5] M. Bellare, D. Pointcheval and P. Rogaway, Authenticated key exchange secure against dictionary attacks, in: *EUROCRYPT*, 2000, pp. 139–155.

[6] X. Boyen, Reusable cryptographic fuzzy extractors, in: *ACM CCS*, 2004, pp. 82–91.

[7] X. Boyen, Y. Dodis, J. Katz, R. Ostrovsky and A.D. Smith, Secure remote authentication using biometric data, in: *EUROCRYPT*, Vol. 3494, 2005, pp. 147–163.

[8] R. Canetti, B. Fuller, O. Paneth, L. Reyzin and A.D. Smith, Reusable fuzzy extractors for low-entropy distributions, in: *EUROCRYPT*, 2016, pp. 117–146.

[9] Y. Dodis, L. Reyzin and A. Smith, Fuzzy extractors: How to generate strong keys from biometrics and other noisy data, in: *EUROCRYPT*, 2004, pp. 523–540.

[10] N. Döttling and J. Müller-Quade, Lossy codes and a new variant of the learning-with-errors problem, in: *EUROCRYPT*, 2013, pp. 18–34.

[11] L. Ducas, A. Durmus, T. Lepoint and V. Lyubashevsky, Lattice signatures and bimodal Gaussians, in: *CRYPTO*, 2013, pp. 40–56.

[12] A. Fiat and A. Shamir, How to prove yourself: Practical solutions to identification and signature problems, in: *CRYPTO*, 1986, pp. 186–194.

[13] B. Fuller, X. Meng and L. Reyzin, Computational fuzzy extractors, in: *ASIACRYPT*, 2013, pp. 174–193.

[14] C. Gentry, C. Peikert and V. Vaikuntanathan, Trapdoors for hard lattices and new cryptographic constructions, in: *STOC*, 2008, pp. 197–206.

[15] T. Güneysu, V. Lyubashevsky and T. Pöppelmann, Practical lattice-based cryptography: A signature scheme for embedded systems, in: *CHES*, 2012, pp. 530–547.

[16] A.K. Jain, S. Prabhakar, L. Hong and S. Pankanti, FingerCode: A filterbank for fingerprint representation and matching, in: *Computer Vision and Pattern Recognition*, IEEE Computer Society Conference on., Vol. 2, 1999, pp. 187–193.

[17] M. Jakobsson, K. Sako and R. Impagliazzo, Designated verifier proofs and their applications, in: *EUROCRYPT*, 1996, pp. 143–154.
[18] A. Juels and M. Wattenberg, A fuzzy commitment scheme, in: *ACM CCS*, 1999, pp. 28–36.
[19] J. Kamp and D. Zuckerman, Deterministic extractors for bit-fixing sources and exposure-resilient cryptography, *SIAM* **36**(5) (2006), 1231–1247. doi:10.1137/S0097539705446846.
[20] N. Li, F. Guo, Y. Mu, W. Susilo and S. Nepal, Fuzzy extractors for biometric identification, in: *ICDCS*, 2017, pp. 667–677.
[21] V. Lyubashevsky, Lattice signatures without trapdoors, in: *EUROCRYPT*, 2012, pp. 738–755.
[22] T. Matsuda, K. Takahashi, T. Murakami and G. Hanaoka, Fuzzy signatures: Relaxing requirements and a new construction, in: *ACNS*, 2016, pp. 97–116.
[23] D. Micciancio, Lattice-based cryptography, in: *Encyclopedia of Cryptography and Security*, 2011, pp. 713–715. doi:10.1007/978-1-4419-5906-5_417.
[24] C. Peikert, A decade of lattice cryptography, *Foundations and Trends® in Theoretical Computer Science* **10**(4) (2016), 283–424. doi:10.1561/0400000074.
[25] O. Regev, On lattices, learning with errors, random linear codes, and cryptography, *JACM* **56**(6) (2009), 34. doi:10.1145/1568318.1568324.
[26] C.-P. Schnorr, Efficient identification and signatures for smart cards, in: *CRYPTO*, 1989, pp. 239–252.
[27] P.W. Shor, Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer, *SIAM Review* **41**(2) (1999), 303–332. doi:10.1137/S0036144598347011.
[28] K. Takahashi, T. Matsuda, T. Murakami, G. Hanaoka and M. Nishigaki, A signature scheme with a fuzzy private key, in: *ACNS*, 2015, pp. 105–126.
[29] K. Takahashi, T. Matsuda, T. Murakami, G. Hanaoka and M. Nishigaki, Signature schemes with a fuzzy private key, *IACR Cryptology ePrint Archive* **2017** (2017), 1188.
[30] Y. Tian, Y. Li, R.H. Deng, N. Li, P. Wu and A. Liu, A new framework for privacy-preserving biometric-based remote user authentication, *Journal of Computer Security* (2020), 1–30.
[31] B. Waters, Efficient identity-based encryption without random oracles, in: *EUROCRYPT*, 2005, pp. 114–127.
[32] Y. Wen and S. Liu, Robustly reusable fuzzy extractor from standard assumptions, in: *ASIACRYPT*, 2018, pp. 459–489.
[33] Y. Wen, S. Liu and S. Han, Reusable fuzzy extractor from the decisional Diffie–Hellman assumption, *Designs, Codes and Cryptography* **86**(11) (2018), 2495–2512. doi:10.1007/s10623-018-0459-4.
[34] M. Yasuda, T. Shimoyama, M. Takenaka, N. Abe, S. Yamada and J. Yamaguchi, Recovering attacks against linear sketch in fuzzy signature schemes of ACNS 2015 and 2016, in: *ISPEC*, 2017, pp. 409–421.