International Journal of Information Security (2019) 18:581–617
https://doi.org/10.1007/s10207-019-00428-z

REGULAR CONTRIBUTION

Signature schemes with a fuzzy private key

Kenta Takahashi[1] · Takahiro Matsuda[2] · Takao Murakami[2] · Goichiro Hanaoka[2] · Masakatsu Nishigaki[3]

Published online: 23 February 2019

Abstract

In this paper, we introduce a new concept of digital signature that we call *fuzzy signature*, which is a signature scheme that uses a noisy string such as biometric data as a private key, but *does not require user-specific auxiliary data* (which is also called a helper string in the context of fuzzy extractors), for generating a signature. Our technical contributions are threefold: (1) we first give the formal definition of fuzzy signature, together with a formal definition of a "setting" that specifies some necessary information for fuzzy data. (2) We give a generic construction of a fuzzy signature scheme based on a signature scheme that has certain homomorphic properties regarding keys and satisfies a kind of related key attack security with respect to addition, and a new tool that we call *linear sketch*. (3) We specify two concrete settings for fuzzy data, and for each of the settings give a concrete instantiation of these building blocks for our generic construction, leading to two concrete fuzzy signature schemes. We also discuss how fuzzy signature schemes can be used to realize a biometric-based PKI that uses biometric data itself as a cryptographic key, which we call the *public biometric infrastructure*.

Keywords Digital signature · Fuzzy signature · Public biometric infrastructure

1 Introduction

1.1 Background and motivation

As the information society grows rapidly, the public key infrastructure (PKI) plays a more significant role as an infrastructure for managing digital certificates. It is also expected to be widely used for personal use such as national IDs and e-government services. One of the biggest risks in the PKI, which needs to be considered in the personal use, lies in a user's private key [10]: since the user's identity is verified based only on his/her private key, the user needs to protect the private key in a highly secure manner. For example, the user is required to store his/her private key into a smart card (or USB token) and remember a password to activate the key. Such limitations reduce usability, and especially, carrying a dedicated device can be a burden to users. This becomes more serious for elderly people in an aging society.

One of the promising approaches to fundamentally solve this problem is to use *biometric data* (e.g., fingerprint, face, and iris) as a cryptographic private key. Since a user's biometrics is a part of human body, it can offer a more secure and usable way to link the individual with his/her private key (i.e., it is not forgotten unlike passwords and is much harder to steal than cards). Also, a sensor that captures multiple biometrics simultaneously (e.g., face and iris [5]; fingerprint and finger-vein [27]) has been widely developed to obtain a large amount of entropy at one time, and a recent study [22] has shown that very high accuracy [e.g., the false acceptance rate (FAR) is 2^{-133} (resp. 2^{-87}) when the false rejection rate (FRR) is 0.055 (resp. 0.0053)] can be achieved by combining four finger-vein features [28].

This is the merged full version of the earlier papers that appeared in the proceedings of ACNS 2015 [33] and the proceedings of ACNS 2016 [19].

✉ Takahiro Matsuda
t-matsuda@aist.go.jp

Kenta Takahashi
kenta.takahashi.bw@hitachi.com

Takao Murakami
takao-murakami@aist.go.jp

Goichiro Hanaoka
hanaoka-goichiro@aist.go.jp

Masakatsu Nishigaki
nisigaki@inf.shizuoka.ac.jp

[1] Hitachi Ltd., Yokohama, Japan

[2] National Institute of Advanced Industrial Science and Technology (AIST), Tokyo, Japan

[3] Shizuoka University, Hamamatsu, Japan

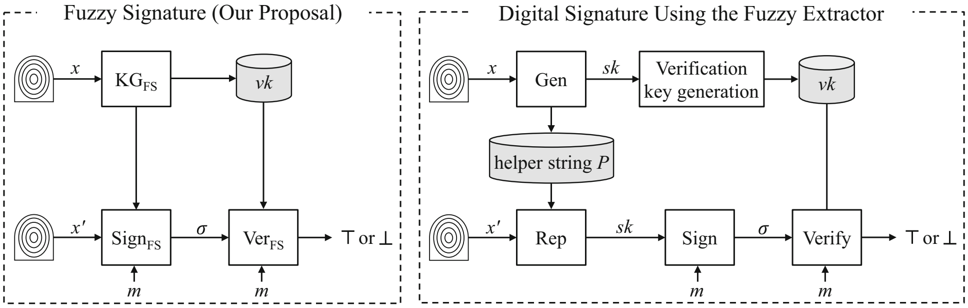
Fuzzy Signature (Our Proposal)
x
KG_FS
vk
x′
Sign_FS
σ
Ver_FS
⊤ or ⊥
m
m
Digital Signature Using the Fuzzy Extractor
x
Gen
sk
Verification key generation
vk
helper string P
x′
Rep
sk
Sign
σ
Verify
⊤ or ⊥
m
m

Fig. 1 Architecture of fuzzy signature (our proposal) (left), and that of digital signature using a fuzzy extractor (right) (x, x': noisy string, sk: signing key, vk: verification key, σ: signature, m: message, $\top$: valid, $\bot$: invalid)

However, since biometric data is noisy and fluctuates each time it is captured, it cannot be used directly as a cryptographic key. In this paper, we call such a noisy string *fuzzy data*. Intuitively, it seems that this issue can be immediately solved by using a *fuzzy extractor* [8], but this is not always the case. More specifically, for extracting a string by a fuzzy extractor, an auxiliary data called a helper string is necessary, and therefore, either the user is still enforced to carry a dedicated device that stores it, or it has to be stored in some server that has to be online at the time of the signing process. (We discuss the limitations of the approaches with helper data (i.e., the fuzzy-extractor-based approaches) in more detail in "Appendix A.")

Hence, it is considered that the above problem cannot be straightforwardly solved by using fuzzy extractors, and another cryptographic technique by which noisy data can be used as a cryptographic private key without relying on any auxiliary data, is necessary.

Fuzzy signature: digital signature with a fuzzy private key. In this paper, we introduce a new concept of digital signature that we call *fuzzy signature*. Consider an ordinary digital signature scheme. The signing algorithm Sign is defined as a (possibly probabilistic) function that takes a signing key sk and a message m as input, and outputs a signature $\sigma \leftarrow \mathsf{Sign}(sk, m)$.[1] Thus, it is natural to consider that its "fuzzy" version Sign should be defined as a function that takes a noisy string x and a message m as input, and outputs $\sigma \leftarrow \mathsf{Sign}(x, m)$. In this paper, we refer to such digital signature (i.e., digital signature that allows to use a noisy string itself as a signing key) as *fuzzy signature*. It should be noted that some studies proposed a fuzzy identity-based signature (FIBS) scheme [11,34,35,37,38], which uses a noisy string as a verification key. However, fuzzy signature is a totally different concept since it does not allow a fuzzy verification key, but allows a *fuzzy signing key* (i.e., *fuzzy private key*).

Figure 1 shows the architecture of fuzzy signature in the left, and that of digital signature using a fuzzy extractor in the right. In fuzzy signature, the key generation algorithm $\mathsf{KG}_{\mathsf{FS}}$ takes a noisy string (e.g., biometric feature) x as input, and outputs a verification key vk; The signing algorithm $\mathsf{Sign}_{\mathsf{FS}}$ takes another noisy string x' and a message m as input, and outputs a signature σ. The verification algorithm $\mathsf{Ver}_{\mathsf{FS}}$ takes vk, m, and σ as input, and verifies whether σ is valid or not. If x' is close to x, σ will be verified as valid. We emphasize that the signing algorithm $\mathsf{Sign}_{\mathsf{FS}}$ in a fuzzy signature scheme does not use the verification key in the signing process.[2] Hence, a fuzzy signature scheme cannot be constructed based on the straightforward combination of a fuzzy extractor and an ordinary signature scheme, since it requires a helper string P along with a noisy string x' to generate a signature σ on a message m. To date, to the best of our knowledge, the realization of fuzzy signature has been an open problem.

1.2 Our contributions

In this paper, we initiate the study of *fuzzy signature*, and give several results on it. Our main contributions are threefold: we give (1) the formal definitions for fuzzy signatures, (2) a generic construction of a fuzzy signature scheme from simpler primitives, and (3) two concrete constructions of a fuzzy signature scheme (each of which is obtained by instantiating the building blocks of our generic construction).

Below we detail each of the contributions as well as other results:

- *Formal definitions for fuzzy signatures*. Our first main contribution is the formalizations of fuzzy signature and concepts related to it, which we give in Sect. 4. More specifically, to formally define fuzzy signatures, we need to first somehow give a formalization of fuzzy data, e.g., a metric space to which fuzzy data belongs, a distribution from which each data is sampled, etc. Therefore, we first formalize it as a *fuzzy key setting* in Sect. 4.1. We then give a formal definition of a fuzzy signature

[1] Strictly speaking, in this paper we adopt the syntax in which Sign also takes a public parameter (generated by the setup algorithm) as input (see Sect. 2.5 for the formal definition). In the introduction, we omit it for simplicity.

[2] We note that like an ordinary signature scheme, the algorithms of a fuzzy signature scheme actually take as input a public parameter that is generated by the setup algorithm and does *not* contain any user-specific information. We omit it from the explanations in the introduction for simplicity. (See the formal definitions of a fuzzy signature in Sect. 4.)

scheme as a primitive that is associated with a fuzzy key setting in Sect. 4.2. We also introduce a new primitive that we call *linear sketch*, which incorporates a kind of encoding and error correction processes. This primitive is also associated with a fuzzy key setting and is one of the building blocks of our generic construction. We informally explain how it works and how it is used in our generic construction in Sect. 1.3, and give the formal definition in Sect. 4.3.

- *Generic construction*. Our second main contribution is a generic construction of a fuzzy signature scheme from simpler primitives, which we give in Sect. 5. Specifically, in order to ease understanding our ideas and the security proofs for our proposed schemes clearly and in a modular manner, we give a generic construction of a fuzzy signature scheme from the combination of a linear sketch scheme (that we introduce in Sect. 4.3) and an ordinary signature scheme. In this construction, we require that the underlying ordinary signature scheme has a certain natural homomorphic property regarding public/secret keys, and furthermore satisfy a kind of related key attack (RKA) security with respect to addition, denoted by Φ^{add}-$\mathtt{RKA}^*$ security. We give an overview of this generic construction in Sect. 1.3. Our concrete instantiations of a fuzzy signature scheme are derived from this generic construction by concretely instantiating the building blocks.
- *Concrete instantiations*. Our third main contribution is two concrete instantiations of a fuzzy signature scheme: the first construction is given in Sect. 6 and the second one is given in Sect. 7. For each of the constructions, we first specify a concrete fuzzy key setting,[3] then show how to concretely realize the underlying signature scheme and a linear sketch scheme that can be used in the generic construction for this fuzzy key setting.

In Sect. 1.3, we give an overview of how our proposed fuzzy signature scheme is constructed, and also an overview on what a linear sketch is like, how it works, as well as our strategies for designing it.

It is expected that our fuzzy signature schemes can be used to realize a biometric-based PKI that uses biometric data itself as a cryptographic key, which we call the *public biometric infrastructure* (PBI). We discuss it in Sect. 9 in more detail. We would like to emphasize that although so far we have mentioned biometric data as a main example of noisy data, our scheme is not restricted to it, and can also use other noisy data such as the output of a PUF (physically unclonable function) [23] as input, as long as it satisfies the requirements of fuzzy key settings.

On the requirements for the underlying signature scheme. As mentioned above, in our generic construction of a fuzzy signature scheme, we use an ordinary signature scheme that has some special structural/security properties (the homomorphic property regarding keys and Φ^{add}-$\mathtt{RKA}$ security). These special properties are formalized and studied in Sect. 3. That we require the underlying signature scheme to satisfy a version of RKA security, might sound a strong requirement. To better understand it and potentially make it easier to achieve, we show two technical results on them:

1. We show sufficient conditions for Φ^{add}-$\mathtt{RKA}^*$ security. More specifically, we show that if an ordinary signature scheme that satisfies standard $\mathtt{EUF\text{-}CMA}$ security and the above-mentioned homomorphic property regarding public/secret keys, additionally satisfies a similarly natural homomorphic property also regarding signatures, then it automatically satisfies Φ^{add}-$\mathtt{RKA}^*$.
2. We also show that the original Schnorr signature scheme [31] already satisfies Φ^{add}-$\mathtt{RKA}^*$ security in the random oracle model under the discrete logarithm (DL) assumption (i.e., the same assumption used for proving its standard $\mathtt{EUF\text{-}CMA}$ security in the random oracle model).

The first (resp. second) technical result listed above is used for our first (resp. second) concrete instantiation of a fuzzy signature scheme.

1.3 Technical overview

Linear sketch. As mentioned above, we introduce a new primitive that we call a *linear sketch* scheme, and use it as one of the building blocks in our generic construction. This primitive is somewhat similar to the one-time pad encryption scheme: recall that in the one-time pad encryption scheme (implemented over some finite additive group), a ciphertext c of a plaintext m under a key K is computed as $c = m + K$. Due to the linearity of the structure, the one-time pad encryption scheme satisfies the following properties: (1) given two ciphertexts $c = m + K$ and $c' = m' + K$ (under the same key K),[4] one can calculate the "difference" $\Delta m = m - m'$ between two plaintexts by calculating $c - c'$, and (2) given a ciphertext $c = m + K$ and "shift" values Δm and ΔK, one can calculate a ciphertext c' of the "shifted" message $m + \Delta m$ under a "shifted" key $K + \Delta K$ by calculating $c' = c + \Delta m + \Delta K$.

Linear sketch formalizes these functionalities of the one-time pad encryption scheme, except that we use fuzzy data

[3] The underlying metric space to which fuzzy data belongs, required of our instantiations of a fuzzy signature scheme, is assumed to be a real vector space $[0, 1)^n$, where we use the L_∞-distance as the distance function. For the details of the formal requirements, see Sects. 6.1 and 7.1.

[4] Of course, a key in the one-time pad encryption scheme should not be used more than once in a normal use!

as a key. The main algorithms of this primitive are Sketch and $\mathsf{DiffRec}$. (It additionally has the setup algorithm that produces a public parameter, but we omit it here for simplicity.) The first algorithm Sketch captures the encryption mechanism. It takes an element s (of some additive group) and a fuzzy data x as input, and outputs a “sketch” c (which is like an encryption of s using x as a key).[5] The second algorithm $\mathsf{DiffRec}$ (which stands for “Difference Reconstruction”) captures the above-mentioned property (1) of the one-time pad encryption scheme, but has an additional “error correction” property. Namely, given two sketches c and c' that, respectively, encrypt s and s' using fuzzy data x and x' as a key, *if x and x' are sufficiently “close” according to some metric*, then we can calculate the difference $\Delta s = s - s'$. We stress that x and x' need not be exactly the same value, and thus the algorithm $\mathsf{DiffRec}$ is required to somehow “absorb” the difference between two noisy data in addition to calculate the difference between s and s'.

In addition to these functional requirements, we also require two additional properties for a linear sketch scheme. The first property is what we call *linearity*, which is similar to the property (2) of the one-time pad encryption mentioned above. Namely, given a sketch c that encrypts s using a fuzzy data x as a key, and “shift” values Δs and Δx, one can generate a sketch c' that encrypts a shifted element $s + \Delta s$ under a shifted key $x + \Delta x$. The second property is a confidentiality notion (which we call *weak simulatability*), that roughly requires that c hides its content s if s and x come from appropriate distributions. These two properties are used in the security proof. For the details of the formalization, see Sect. 4.3.

For our concrete instantiations of a fuzzy signature scheme, we construct different linear sketch schemes. The linear sketch scheme for the first instantiation is given in Sect. 6.3, and that for the second instantiation is given in Sect. 7.2.

Generic construction. Our proposed fuzzy signature scheme Σ_{FS} is constructed based on an ordinary signature scheme (let us call it the “underlying scheme” Σ for the explanation here), and a linear sketch scheme. In Fig. 2, we illustrate an overview of our construction of a fuzzy signature scheme.

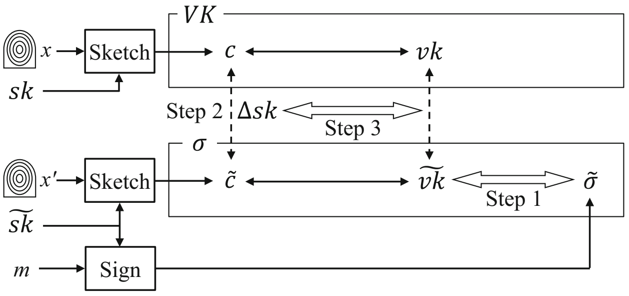

Fig. 2 An overview of our generic construction of a fuzzy signature scheme. The box “Sketch” indicates one of the algorithms of a primitive that we call “linear sketch,” which is formalized in Sect. 4.3

An overview of our generic construction is as follows: In the signing algorithm $\mathsf{Sign}_{\mathsf{FS}}(x', m)$ (where x' is a fuzzy data used as a signing key and m is a message to be signed), we do not extract a signing key sk (for the underlying scheme Σ) directly from x' (which is the idea of the fuzzy-extractor-based approach), but generate a random fresh “temporary” key pair $(\widetilde{vk}, \widetilde{sk})$ of the underlying signature scheme Σ, and generate a signature $\widetilde{\sigma}$ on m using $\widetilde{sk}$. This enables us to generate a fresh signature $\widetilde{\sigma}$ without being worried about the fuzziness of x'. Here, however, since $\widetilde{\sigma}$ is a valid signature only under $\widetilde{vk}$, we have to somehow link it with the noisy signing key x'. This is done by the linear sketch scheme.

More specifically, in the signing procedure, we additionally generate a “sketch” $\widetilde{c}$ (via the algorithm denoted by “Sketch” in Fig. 2) of the temporary signing key $\widetilde{sk}$ using the fuzzy data x'. (As explained above, this works like a one-time pad encryption of $\widetilde{sk}$ generated by using x' as a key.) Then, we let a signature σ of the fuzzy signature scheme consist of $(\widetilde{vk}, \widetilde{\sigma}, \widetilde{c})$.

Before seeing how we verify $\sigma = (\widetilde{vk}, \widetilde{\sigma}, \widetilde{c})$, we explain how a verification key in our fuzzy signature scheme is generated: In the key generation algorithm $\mathsf{KG}_{\mathsf{FS}}(x)$ (where x is also a fuzzy data measured at the key generation), we generate a fresh key pair (vk, sk) of the underlying signature scheme Σ, as well as a “sketch” c of the signing key sk using the noisy data x (in exactly the same way we generate $\widetilde{c}$ from x' and $\widetilde{sk}$), and put it as part of a verification key of our fuzzy signature scheme. Hence, a verification key VK in our fuzzy signature scheme Σ_{FS} consists of the verification key vk of the underlying scheme Σ, and the sketch c generated from sk and x. Then, in the verification algorithm $\mathsf{Ver}_{\mathsf{FS}}(VK, m, \sigma)$ where $VK = (vk, c)$ and $\sigma = (\widetilde{vk}, \widetilde{\sigma}, \widetilde{c})$, we first check the validity of $\widetilde{\sigma}$ under $\widetilde{vk}$ (Step 1), then recover the “difference” $\Delta sk = \widetilde{sk} - sk$ of the underlying secret keys from c and $\widetilde{c}$ via the $\mathsf{DiffRec}$ algorithm of the underling linear sketch scheme (Step 2), and finally check whether the difference between vk and $\widetilde{vk}$ indeed corresponds to Δsk (Step 3). The explanation so far is exactly what we do in our generic construction in Sect. 5.

Requirements on the underlying signature scheme. In order to realize Step 3 of the verification algorithm of our generic construction, we require the underlying signature scheme Σ to satisfy the property that given two verification keys $(vk, \widetilde{vk})$ and a (candidate) difference Δsk, one can verify that the dif-

[5] Unlike the one-time encryption scheme, decryption is not considered in this primitive, and hence it would be more appropriate to consider it as a (one-way) “encoding,” rather than “encryption.” This is also one of the reasons why we call c a “sketch” (something that contains the information of the input s), not a “ciphertext.”

ference between the secret keys sk and $\widetilde{sk}$ (corresponding to vk and $\widetilde{vk}$, respectively) is indeed Δsk. It turns out that such a property is satisfied if a signature scheme satisfies a certain natural homomorphic property regarding verification/secret keys, which we formalize in Sect. 3.1. This property is satisfied by many existing schemes, and in particular we will show that it is satisfied by our variant of the Waters signature scheme [36] (MWS scheme) and the Schnorr signature scheme [31].

The security[6] of our generic construction of a fuzzy signature scheme is, with the help of the properties of the underlying linear sketch scheme, reduced to our variant of the RKA security (with respect to addition), Φ^{add}-$\mathtt{RKA}^*$ security, of the underlying signature scheme Σ. Roughly speaking, this security notion requires that an adversary, who is initially given a verification vk (corresponding to a secret key sk) and can obtain signatures computed under "shifted" signing keys of the form $sk + \Delta sk$ (where the "shift" values Δsk can be chosen by the adversary) via the "RKA"-signing oracle, cannot generate a successfully forced message/signature pair, *even under a "shifted" verification key vk' corresponding to a shifted signing key of the form $sk + \Delta sk'$ (where again the "shift" $\Delta sk'$ can be chosen by the adversary)*. The formal definition is given in Sect. 3.2, where we also explain the difference between this security notion and the popular RKA security definition by Bellare et al. [2]. Roughly speaking, the reason why we require such "RKA" security for the underlying signature scheme Σ, is because in a sequence of games in the security proof, we change how the temporary key pair $(\widetilde{vk}, \widetilde{sk})$ is generated, in such a way that instead of picking a fresh key pair, (1) we first pick a random shift Δsk, (2) then compute $\widetilde{sk} = sk + \Delta sk$ (where sk is the secret key corresponding to vk in the verification key VK), and (3) finally compute $\widetilde{vk}$ from $\widetilde{sk}$. Then, the value $\widetilde{\sigma}$ appearing in a fuzzy signature $\sigma = (\widetilde{vk}, \widetilde{\sigma}, \widetilde{c})$ can be seen as a signature generated by using the "shifted" key $\widetilde{sk} = sk + \Delta sk$, which can be simulated without knowing sk if one has access to the "RKA"-signing oracle. For the details of the security proof, see Sect. 5.3.

First instantiation. Our first instantiation, denoted by Σ_{FS1} and given in Sect. 6, is constructed for a specific fuzzy key setting in which fuzzy data is a uniformly distributed vector over a metric space with the L_∞-distance.[7] For this fuzzy key setting, we propose a concrete linear sketch scheme based on the Chinese remainder theorem (CRT) and some form of linear coding and error correction methods. We also propose a variant of the Waters signature scheme [36], which we call *modified Waters signature* (MWS) scheme, that is compatible with the linear sketch scheme and furthermore satisfies all the requirements required of the underlying signature scheme in our generic construction. The resulting fuzzy signature scheme from these linear sketch and MWS schemes, is secure in the standard model under the computational Diffie–Hellman (CDH) assumption in bilinear groups.

Second instantiation. One drawback of our first instantiation is that it has to assume that fuzzy data is distributed uniformly. Our second construction based on the Schnorr signature scheme [31], denoted by Σ_{FS2} and given in Sect. 7, tries to overcome this drawback. Specifically, we consider another specific fuzzy key setting in which fuzzy data is assumed to come from a distribution that has high average min-entropy [8] given a part of the fuzzy data. (The exact specification of a fuzzy key setting is given in Sect. 7.1.) For this fuzzy key setting, we propose a concrete linear sketch scheme based on a universal hash family satisfying a natural linearity property. We use a version of the leftover hash lemma [8,14] to show that this scheme achieves the confidentiality notion required of a linear sketch scheme. Our second construction of a fuzzy signature scheme is obtained by combining this linear sketch scheme and the original Schnorr signature scheme [31] (which we will show to be Φ^{add}-$\mathtt{RKA}^*$). The resulting fuzzy signature scheme is secure in the random oracle model under the DL assumption. Although this construction relies on a random oracle, it assumes a weaker requirement for the distribution of fuzzy data, more efficient, easier to implement, and hence more practical, than our first construction.

1.4 Paper organization

The rest of the paper is organized as follows:

- In Sect. 1.5, we explain the relations between this paper and our earlier papers [19,33].
- In Sect. 2, we review basic notation and standard definitions.
- In Sect. 3, we formalize the homomorphic property and our variant of RKA security, as well as some facts on them that are useful for our instantiations of a fuzzy signature scheme.
- In Sect. 4, we provide the formal definition of fuzzy signature, together with the formalization of a "fuzzy key setting" over which a fuzzy signature is defined. We also give a formalization of linear sketch.
- In Sect. 5, we show a generic construction of a fuzzy signature scheme based on the combination of a linear sketch scheme and a signature scheme with (the weaker version of) the homomorphic property (defined in Sect. 3).

[6] The security of a fuzzy signature scheme is defined similarly to that of the standard $\mathtt{EUF}$-$\mathtt{CMA}$ security [13] of an ordinary signature scheme. See Sect. 4.2 for the formal definition.

[7] In practice, we have to consider the treatment of real numbers. We discuss how it is represented at the beginning of Sect. 6 and in Sect. 8.

- In Sect. 6, we give our first instantiation of a fuzzy signature scheme based on the Waters signature scheme [36].
- In Sect. 7, we give our second instantiation of a fuzzy signature scheme based on the Schnorr signature scheme [31].
- In Sect. 8, we discuss the treatment of real numbers for our fuzzy signature schemes in practical implementations.
- Finally, in Sect. 9, we discuss how a fuzzy signature scheme can be used to realize the public biometric infrastructure (PBI). There, we also give a discussion about the requirement on the fuzzy key settings for which our concrete instantiations are constructed, and several open problems.

1.5 Relation to earlier versions

This paper is the merged full version of our earlier papers [19,33]. Here, we first explain the overview of these papers and then clarify the correspondences of the contents between this paper and [19,33] and the additional contributions from them. (The reader who has not read our earlier papers [19,33] could skip this subsection.)

Overview of [33]. We introduced the formalizations of fuzzy signatures, including the formal definitions for a fuzzy key setting and a linear sketch scheme, and gave a generic construction of a fuzzy signature scheme from an ordinary signature scheme satisfying the single key generation process (Definition 7) and the homomorphic property (Definition 9). Then, we specified a concrete fuzzy key setting (in which the metric space for fuzzy data is $[0, 1)^n$ with L_∞-distance and fuzzy data is assumed to be distributed uniformly), and showed a concrete linear sketch scheme (denoted $\mathcal{S}_{\mathtt{CRT}}$) based on the Chinese remainder theorem and a concrete signature scheme [called *modified Waters signature* (MWS) scheme and denoted $\Sigma_{\mathtt{MWS}}$] based on the Waters signature scheme [36] that satisfy the requirements for the generic construction, and thus they led to the first instantiation of a fuzzy signature scheme, denoted Σ_{FS1}. We also introduced the notion of *Public Biometric Infrastructure* (PBI), which is a biometrics-analogue of public key infrastructure (PKI), and discussed how a fuzzy signature scheme can be used to realize it.

Overview of [19]. We gave some relaxations to the requirements for the underlying linear sketch scheme and the underlying signature scheme used in the generic construction in [33]. More specifically, for the underlying linear sketch scheme, we showed that weaker syntactical and confidentiality properties were sufficient. Regarding the underlying ordinary signature scheme, we showed that it only needs to have a weaker form of homomorphic property (called weak homomorphic property in Definition 9) if it satisfies a version of "related key attack" security (denoted "$\mathtt{RKA}^*$" in this paper) with respect to addition. (Security against related key attacks might seem a strong requirement, but we also showed that if a signature scheme satisfies the homomorphic property required in [33], then it automatically satisfies $\mathtt{RKA}^*$ security with respect to addition.) We then specified a concrete fuzzy key setting (in which the metric space is the same as in [33], but fuzzy data distribution is only required to have high average min-entropy given some leakage) and showed concrete instantiations of a linear sketch scheme (denoted $\mathcal{S}_{\mathtt{Hash}}$) based on a universal hash family (with linearity) and the Schnorr signature scheme [31] (denoted $\Sigma_{\mathtt{Sch}}$) satisfy the weakened requirements. From these ingredients, we obtained the second instantiation of a fuzzy signature scheme, denoted Σ_{FS2}.

Correspondences. Here, we explain the correspondences of the contents between the current paper and those in [19,33]. (See also the "*Additional Contributions*" paragraph below.)

In this paper, the formalizations for fuzzy signatures, fuzzy key setting, and linear sketch schemes in Sect. 4 are basically the ones used in [19]. However, we introduce a new relaxation to the confidentiality notion for a linear sketch scheme, which we call *weak simulatability*.

The generic construction and its proof given in Sect. 5 are based on [19,33], respectively, but the security proof in this paper has a new aspect in that we now use a weaker assumption on the linear sketch scheme than [19] (i.e., weak simulatability).

The results regarding the first instantiation Σ_{FS1} in Sect. 6 are based on [33], and those regarding the second instantiation Σ_{FS2} in Sect. 7 are based on [19]. The technical results regarding ordinary signature schemes in Sect. 3 are based on [19].

The discussion on the PBI in Sect. 9 is based on [33].

Additional contributions. Here, we list the additional contributions in this paper compared to our earlier papers [19,33].

- As mentioned above, we introduce a security definition called *weak simulatability* for a linear sketch scheme, which is weaker than the security definitions that we introduced in our earlier papers. This leads to weakening the assumption needed for the security proof of our generic construction of a fuzzy signature scheme to go through, and hence potentially makes it easier to construct a fuzzy signature scheme in the future.
- Corresponding to the above item, the security proof for our generic construction of a fuzzy signature scheme (in Sect. 5), and the security proofs for the concrete linear sketch schemes ($\mathcal{S}_{\mathtt{CRT}}$ in Sect. 6.3 and $\mathcal{S}_{\mathtt{Hash}}$ in Sect. 7.2), are changed from the ones we had for our earlier papers to accommodate the use of weak simulatability. In particular, the security proof for the linear sketch scheme $\mathcal{S}_{\mathtt{CRT}}$

is entirely renewed from the one we had in [33] (which is partly also due to the next item).

- As mentioned earlier, in our earlier papers [19,33], we left the treatment of real numbers in the constructions of our fuzzy signature schemes and linear sketch schemes somewhat ambiguous (and it was pointed out by Yasuda et al. [39] that our linear sketch schemes could be vulnerable to so-called "recovering attacks," if real numbers are improperly treated). In this paper, we clarify the treatment of real numbers in the "*On the Treatment of Real Numbers*" paragraph in the beginning of Sect. 6. (This also shows that Yasuda et al.'s attacks do not work for our linear sketch schemes, and we explain it in Sect. 6.3.)
- Section 8 is new to this paper, where we revisit and discuss the treatment of real numbers in our proposed fuzzy signature schemes by taking into account practical implementations. In particular, we consider variants of our fuzzy signature schemes in which the "decimal part" of real numbers are truncated, and then explain how the truncation affects the correctness and security of the modified schemes. We state the effect on the correctness as theorems and provide the formal proofs for them.
- We add discussions on the revocation functionality in the PBI in Sect. 9.
- The formal proofs of the most of the theorems and lemmas were omitted in [19,33] due to the space limitation, and they are all given in this paper.

2 Preliminaries

In this section, we review the basic notation, the definitions of standard primitives, and existing results that we use in this paper.

2.1 Basic notation

$\mathbb{N}$, $\mathbb{Z}$, $\mathbb{R}$, and $\mathbb{R}_{\geq 0}$ denote the sets of all natural numbers, all integers, all real numbers, and all nonnegative real numbers, respectively. If $n \in \mathbb{N}$, then we define $[n] := \{1, \ldots, n\}$. If $a, b \in \mathbb{N}$, then "$\mathsf{GCD}(a, b)$" denotes the greatest common divisor of a and b. If $a \in \mathbb{R}$, then "$\lfloor a \rfloor$" denotes the maximum integer which does not exceed a (i.e., the rounding-down operation), and "$\lfloor a \rceil$" denotes the integer that is the nearest to a (i.e., the rounding operation). Throughout the paper, we use the bold font to denote a vector (such as $\mathbf{x}$ and $\mathbf{a}$). We extend the definition of "$\lfloor \cdot \rceil$" to allow it to take a real vector $\mathbf{a} = (a_1, a_2, \ldots)$ as input, by $\lfloor \mathbf{a} \rceil := (\lfloor a_1 \rceil, \lfloor a_2 \rceil, \ldots)$.

"$x \leftarrow y$" denotes that y is (deterministically) assigned to x. If S is a finite set, then "$|S|$" denotes its size, and "$x \leftarrow_{\mathrm{R}} S$" denotes that x is chosen uniformly at random from S. If Φ is a distribution (over some set), then $x \leftarrow_{\mathrm{R}} \Phi$ denotes that x is chosen according to the distribution Φ. If x and y are bit-strings, then $|x|$ denotes the bit length of x, and "$(x||y)$" denotes the concatenation of x and y. "(P)PTA" denotes a (*probabilistic*) *polynomial time algorithm*.

If $\mathcal{A}$ is a probabilistic algorithm, then "$y \leftarrow_{\mathrm{R}} \mathcal{A}(x)$" denote that $\mathcal{A}$ computes y by taking x as input and using an internal randomness that is chosen uniformly at random, and if we need to specify the used randomness (say r), we denote by "$y \leftarrow \mathcal{A}(x; r)$" (in which case the computation of $\mathcal{A}$ is deterministic, taking x and r as input). If furthermore $\mathcal{O}$ is a (possibly probabilistic) algorithm or a function, then "$\mathcal{A}^{\mathcal{O}}$" denotes that $\mathcal{A}$ has oracle access to $\mathcal{O}$. Throughout the paper, "k" denotes a security parameter. A function $f(\cdot) : \mathbb{N} \to [0, 1]$ is said to be *negligible* if for all positive polynomials $p(\cdot)$ and all sufficiently large k, we have $f(k) < 1/p(k)$.

2.2 Basic definitions and lemmas related to probability and entropy

Definition 1 Let $\mathcal{X}$ be a distribution defined over a set X. The *min-entropy* of $\mathcal{X}$, denoted by $\mathbf{H}_\infty(\mathcal{X})$, is defined by

$$\mathbf{H}_\infty(\mathcal{X}) := -\log_2\Big(\max_{x' \in X} \Pr[\mathcal{X} = x']\Big).$$

Definition 2 [8] Let $(\mathcal{X}, \mathcal{Y})$ be a joint distribution defined over the direct product of sets $X \times Y$. The *average min-entropy of $\mathcal{X}$ given $\mathcal{Y}$*, denoted by $\widetilde{\mathbf{H}}_\infty(\mathcal{X}|\mathcal{Y})$, is defined by

$$\widetilde{\mathbf{H}}_\infty(\mathcal{X}|\mathcal{Y}) := -\log_2\Big(\mathop{\mathbf{E}}_{y \leftarrow_{\mathrm{R}} \mathcal{Y}}\Big[\max_{x' \in X} \Pr[\mathcal{X} = x' | \mathcal{Y} = y]\Big]\Big).$$

Definition 3 Let $\mathcal{X}$ and $\mathcal{X}'$ be distributions defined over the same set X. The *statistical distance between $\mathcal{X}$ and $\mathcal{X}'$*, denoted by $\mathbf{SD}(\mathcal{X}, \mathcal{X}')$, is defined by

$$\mathbf{SD}(\mathcal{X}, \mathcal{X}') := \frac{1}{2}\sum_{z \in X}\Big|\Pr[\mathcal{X} = z] - \Pr[\mathcal{X}' = z]\Big|.$$

We say that $\mathcal{X}$ and $\mathcal{X}'$ are statistically indistinguishable, if $\mathbf{SD}(\mathcal{X}, \mathcal{X}')$ is negligible.

In this paper, we will use the following simple and yet useful lemma shown by Dodis and Yu [9, Lemma 1].[8]

Lemma 1 (Adapted from [9, Lemma 1]) *Let X be a finite set, and let U_X be the uniform distribution over X. For any* (*deterministic*) *real-valued function $f : X \to \mathbb{R}_{\geq 0}$ and any distribution $\mathcal{X}$ over the set X, we have*

$$\mathbf{E}[f(\mathcal{X})] \leq |X| \cdot 2^{-\mathbf{H}_\infty(\mathcal{X})} \cdot \mathbf{E}[f(U_X)].$$

[8] Dodis and Yu [9] stated the lemma for the case in which the set X is of the form $\{0, 1\}^m$. However, it is straightforward to see that their proof carries over to the more general case stated here.

From the above lemma, we can derive the following lemma about the (in)distinguishability between the uniform distribution versus a distribution with high min-entropy:

Lemma 2 (Corollary of Lemma 1) *Let X be a finite set, and let U_X be the uniform distribution over X. For any computationally unbounded, probabilistic algorithm $\mathcal{A}: X \to \{0, 1\}$ and any distribution $\mathcal{X}$ over the set X, we have*

$$\Pr[\mathcal{A}(\mathcal{X}) = 1] \le |X| \cdot 2^{-\mathbf{H}_\infty(\mathcal{X})} \cdot \Pr[\mathcal{A}(U_X) = 1],$$

where both of the probabilities are also taken over $\mathcal{A}$'s internal randomness.

Proof of Lemma 2 Let $\mathcal{A}$ be any algorithm, and consider the function $f(x) := \Pr[\mathcal{A}(x) = 1]$ (where the probability is over $\mathcal{A}$'s internal randomness). Then, f is a deterministic function that maps $x \in X$ to the range $[0, 1]$. Furthermore, by definition, we have $\Pr[\mathcal{A}(\mathcal{X}) = 1] = \mathbf{E}[f(\mathcal{X})]$ and $\Pr[\mathcal{A}(U_X) = 1] = \mathbf{E}[f(U_X)]$. Hence, by Lemma 1, we obtain the lemma. □

2.3 Universal hash function family and the leftover hash lemma

Here, we first recall the definition of a universal hash function family, then its concrete construction, and finally the leftover hash lemma [8,14].

Definition 4 Let $\mathcal{H} = \{h_z : D \to R\}_{z \in Z}$ be a family of hash functions, where Z denotes the seed space of $\mathcal{H}$. We say that $\mathcal{H}$ is a *universal hash function family* if for all $x, x' \in D$ such that $x \neq x'$, we have $\Pr_{z \leftarrow_{\mathrm{R}} Z}[h_z(x) = h_z(x')] \le 1/|R|$.

Concrete universal hash family with linearity. In this paper, we will use the following concrete construction of a universal hash function family $\mathcal{H}_{\mathrm{lin}}$ whose domain is $\mathbb{F}_{p^n}$ and whose range is $\mathbb{F}_p$, where $\mathbb{F}_p$ is a finite field with prime order p and $n \in \mathbb{N}$. Note that $\mathbb{F}_{p^n}$, when viewed as a vector space, is isomorphic to the vector space $(\mathbb{F}_p)^n$. Let $\psi: (\mathbb{F}_p)^n \to \mathbb{F}_{p^n}$ be an isomorphism of the vector spaces, and ψ^{-1} be its inverse, which are both efficiently computable in terms of $\log_2(p^n)$.

Let the seed space be $Z = \mathbb{F}_{p^n}$, the domain be $D = (\mathbb{F}_p)^n$, and the range be $R = \mathbb{F}_p$. For each $z \in Z$, define the function $h_z: D \to R$ as follows: On input $\mathbf{x} \in (\mathbb{F}_p)^n$, $h_z(\mathbf{x})$ computes $y \leftarrow \psi(\mathbf{x}) \cdot z$, where the operation "$\cdot$" is the multiplication in the extension field $\mathbb{F}_{p^n}$. Let $(y_1, \dots, y_n) = \psi^{-1}(y)$. The output of $h_z(\mathbf{x})$ is $y_1 \in \mathbb{F}_p$. The family $\mathcal{H}_{\mathrm{lin}}$ consists of the hash functions $\{h_z\}_{z \in Z}$.

It is well known (see, e.g., [4]) that $\mathcal{H}_{\mathrm{lin}}$ is a universal hash function family. Furthermore, for every $z \in Z$, h_z satisfies linearity, in the following sense:

$\forall \mathbf{x}, \mathbf{x}' \in (\mathbb{F}_p)^n$ and $\alpha, \beta \in \mathbb{F}_p$:

$$\alpha \cdot h_z(\mathbf{x}) + \beta \cdot h_z(\mathbf{x}') = h_z(\alpha \cdot \mathbf{x} + \beta \cdot \mathbf{x}').$$

Leftover hash lemma. Roughly speaking, the leftover hash lemma [14] states that a universal hash function family is a good (strong) randomness extractor. Here, we recall a version of the leftover hash lemma shown by Dodis et al. [8] that allows leakage from the inputs to a universal hash function.

Lemma 3 [8] *Let $\mathcal{H} = \{h_z : D \to R\}_{z \in Z}$ be a universal hash function family. Let U_Z and U_R be the uniform distributions over Z and R, respectively. Furthermore, let $(\mathcal{X}, \mathcal{Y})$ be a joint distribution, where the support of $\mathcal{X}$ is contained in D. Then, when z is chosen uniformly as $z \leftarrow_{\mathrm{R}} Z$, it holds that*

$$\mathbf{SD}\Big((z, h_z(\mathcal{X}), \mathcal{Y}), (U_Z, U_R, \mathcal{Y})\Big) \le \frac{1}{2}\sqrt{2^{-\tilde{\mathbf{H}}_\infty(\mathcal{X}|\mathcal{Y})} \cdot |R|}.$$

2.4 (Bilinear) Groups and computational problems

Discrete logarithm assumption. Let GGen be a PPTA, which we call a "group generator," that takes 1^k as input and outputs a tuple $\mathcal{G} := (p, \mathbb{G}, g)$, where $\mathbb{G}$ is a (description of) group with prime order p such that $|p| = \Theta(k)$, and g is a generator of $\mathbb{G}$.

Definition 5 We say that the discrete logarithm (DL) assumption holds with respect to GGen if for all PPTAs $\mathcal{A}$, $\mathsf{Adv}^{\mathrm{DL}}_{\mathsf{GGen},\mathcal{A}}(k)$ defined below is negligible:

$$\mathsf{Adv}^{\mathrm{DL}}_{\mathsf{GGen},\mathcal{A}}(k)$$
$$:= \Pr\Big[\mathcal{G} = (p, \mathbb{G}, g) \leftarrow \mathsf{GGen}(1^k);\ x \leftarrow_{\mathrm{R}} \mathbb{Z}_p : \mathcal{A}(\mathcal{G}, g^x) = x\Big].$$

Bilinear groups and CDH assumption. We say that $\mathcal{BG} = (p, \mathbb{G}, \mathbb{G}_T, g, e)$ constitutes (symmetric) bilinear groups if p is a prime, $\mathbb{G}$ and $\mathbb{G}_T$ are cyclic groups with order p, g is a generator of $\mathbb{G}$, and $e: \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ is an efficiently (in $|p|$) computable mapping satisfying the following two properties:

(*Bilinearity*) For all $g' \in \mathbb{G}$ and $a, b \in \mathbb{Z}_p$, it holds that $e(g'^a, g'^b) = e(g', g')^{ab}$

(*Non-degeneracy*) For all generators g' of $\mathbb{G}$, $e(g', g') \in \mathbb{G}_T$ is not the identity element of $\mathbb{G}_T$.

For convenience, we denote by BGGen an algorithm (referred to as a "bilinear group generator") that, on input 1^k, outputs a description of bilinear groups $\mathcal{BG} = (p, \mathbb{G}, \mathbb{G}_T, g, e)$ such that $|p| = \Theta(k)$.

Definition 6 We say that the computational Diffie–Hellman (CDH) assumption holds with respect to BGGen if for all PPTAs $\mathcal{A}$, $\mathsf{Adv}^{\mathrm{CDH}}_{\mathsf{BGGen},\mathcal{A}}(k)$ defined below is negligible:

$$\mathsf{Adv}^{\mathrm{CDH}}_{\mathsf{BGGen},\mathcal{A}}(k)$$
$$:= \Pr\Big[\, \mathcal{BG} = (p, \mathbb{G}, \mathbb{G}_T, g, e) \leftarrow \mathsf{BGGen}(1^k);\ a, b \leftarrow_{\mathrm{R}} \mathbb{Z}_p:$$
$$\mathcal{A}(\mathcal{BG}, g^a, g^b) = g^{ab} \,\Big].$$

2.5 Signature schemes

Here, we review the standard definitions for (ordinary) signature schemes and some properties. We also review the descriptions of the Waters signature scheme [36] and the Schnorr signature scheme [31] on which the concrete constructions of our fuzzy signature schemes will be based.

Syntax and correctness. We model a signature scheme Σ as a quadruple of the PPTAs $(\mathsf{Setup}, \mathsf{KG}, \mathsf{Sign}, \mathsf{Ver})$ that are defined as follows:

- Setup is the setup algorithm that takes 1^k as input, and outputs a public parameter pp.
- KG is the key generation algorithm that takes pp as input, and outputs a verification/signing key pair (vk, sk).
- Sign is the signing algorithm that takes pp, sk, and a message m as input, and outputs a signature σ.
- Ver is the (deterministic) verification algorithm that takes pp, vk, m, and σ as input, and outputs either $\top$ or $\bot$. Here, "$\top$" (resp. "$\bot$") indicates that σ is a valid (resp. invalid) signature of the message m under the key vk.

We require for all $k \in \mathbb{N}$, all pp output by $\mathsf{Setup}(1^k)$, all (vk, sk) output by $\mathsf{KG}(pp)$, and all messages m, we have $\mathsf{Ver}(pp, vk, m, \mathsf{Sign}(pp, sk, m)) = \top$.

Simple key generation process. Here, we formalize the natural structural property of a signature scheme that we call the *simple key generation process* property, which says that the key generation algorithm KG first picks a secret key sk uniformly at random from the secret key space, and then computes the corresponding verification key vk deterministically from sk. Looking ahead, both of our concrete instantiations of fuzzy signature schemes are constructed from ordinary signature schemes with this property.

Definition 7 Let $\Sigma = (\mathsf{Setup}, \mathsf{KG}, \mathsf{Sign}, \mathsf{Ver})$ be a signature scheme. We say that Σ has a *simple key generation process* if each pp output by Setup specifies the secret key space $\mathcal{K}_{pp}$, and there exists a *deterministic* PTA KG' such that the key generation algorithm $\mathsf{KG}(pp)$ can be written as follows:

$$\mathsf{KG}(pp) \colon \Big[\, sk \leftarrow_{\mathrm{R}} \mathcal{K}_{pp};\ vk \leftarrow \mathsf{KG}'(pp, sk);\ \text{Return}\,(vk, sk). \Big]. \tag{1}$$

$\mathtt{EUF\text{-}CMA}$ *security.* Here, we recall the definition of *existential unforgeability against chosen message attacks* ($\mathtt{EUF\text{-}CMA}$ security) [13]. For a signature scheme $\Sigma = (\mathsf{Setup}, \mathsf{KG}, \mathsf{Sign}, \mathsf{Ver})$ and an adversary $\mathcal{A}$, consider the following $\mathtt{EUF\text{-}CMA}$ experiment $\mathsf{Expt}^{\mathtt{EUF\text{-}CMA}}_{\Sigma,\mathcal{A}}(k)$:

```
$\mathsf{Expt}^{\mathtt{EUF\text{-}CMA}}_{\Sigma,\mathcal{A}}(k)$ :
    $pp \leftarrow_{\mathrm{R}} \mathsf{Setup}(1^k)$
    $(vk, sk) \leftarrow_{\mathrm{R}} \mathsf{KG}(pp)$
    $\mathcal{Q} \leftarrow \emptyset$
    $(m', \sigma') \leftarrow_{\mathrm{R}} \mathcal{A}^{\mathcal{O}_{\mathsf{Sign}}(\cdot)}(pp, vk)$
    If $m' \notin \mathcal{Q} \wedge \mathsf{Ver}(pp, vk, m', \sigma') = \top$
                    then return 1 else return 0
```

where $\mathcal{O}_{\mathsf{Sign}}$ is the signing oracle that takes a message m as input, updates the "used message list" $\mathcal{Q}$ by $\mathcal{Q} \leftarrow \mathcal{Q} \cup \{m\}$, and returns a signature $\sigma \leftarrow_{\mathrm{R}} \mathsf{Sign}(pp, sk, m)$.

Definition 8 We say that a signature scheme Σ is $\mathtt{EUF\text{-}CMA}$ secure if for all PPTA adversaries $\mathcal{A}$,

$$\mathsf{Adv}^{\mathtt{EUF\text{-}CMA}}_{\Sigma,\mathcal{A}}(k) := \Pr[\mathsf{Expt}^{\mathtt{EUF\text{-}CMA}}_{\Sigma,\mathcal{A}}(k) = 1]$$

is negligible.

On "weak" distributions of signing keys. Let $\Sigma = (\mathsf{Setup}, \mathsf{KG}, \mathsf{Sign}, \mathsf{Ver})$ be a signature scheme with a simple key generation process (as per Definition 7) with secret key space $\mathcal{K}_{pp}$ for a public parameter pp, and thus there exists the algorithm KG' such that KG can be written as in Eq. (1). Let $u \colon \mathbb{N} \rightarrow \mathbb{N}$ be any function. For an $\mathtt{EUF\text{-}CMA}$ adversary $\mathcal{A}$ attacking Σ, let $\widetilde{\mathsf{Adv}}^{\mathtt{EUF\text{-}CMA}}_{\Sigma,\mathcal{A}}(k)$ be the advantage of $\mathcal{A}$ in the experiment that is the same as $\mathsf{Expt}^{\mathtt{EUF\text{-}CMA}}_{\Sigma,\mathcal{A}}(k)$, except that a secret key sk is chosen by $sk \leftarrow_{\mathrm{R}} \widetilde{\mathcal{K}}_{pp}$ (instead of $sk \leftarrow_{\mathrm{R}} \mathcal{K}_{pp}$) where $\widetilde{\mathcal{K}}_{pp}$ denotes an arbitrary (non-empty) subset of $\mathcal{K}_{pp}$ satisfying $|\mathcal{K}_{pp}|/|\widetilde{\mathcal{K}}_{pp}| \leq u(k)$.

We will use the following fact, which is obtained as a corollary of Lemma 1. For completeness, we provide its formal proof in "Appendix D."

Lemma 4 (Corollary of Lemma 1) *Under the above setting, for any PPTA adversary $\mathcal{A}$, it holds that $\widetilde{\mathsf{Adv}}^{\mathtt{EUF\text{-}CMA}}_{\Sigma,\mathcal{A}}(k) \leq u(k) \cdot \mathsf{Adv}^{\mathtt{EUF\text{-}CMA}}_{\Sigma,\mathcal{A}}(k)$.*

Waters signature scheme. Our first concrete instantiation of a fuzzy signature scheme given in Sect. 6 is based on the Waters signature scheme [36], and thus we review it here. We consider the version where the setup and the key generation for each user are separated so that the scheme fits our syntax.

Let $\ell = \ell(k)$ be a positive polynomial, and let BGGen be a bilinear group generator. Then, the Waters signature scheme $\Sigma_{\mathtt{Wat}}$ for ℓ-bit messages, is constructed as in Fig. 3 (left). It was shown by Waters [36] that $\Sigma_{\mathtt{Wat}}$ is $\mathtt{EUF\text{-}CMA}$ secure if the CDH assumption holds with respect to BGGen.

Schnorr signature scheme. Our second concrete instantiation of a fuzzy signature scheme given in Sect. 7 is based on the Schnorr signature scheme [31], and thus we review it here.

```
$\mathsf{Setup}_{\mathtt{Wat}}(1^k)$ :
  $\mathcal{BG} := (p, \mathbb{G}, \mathbb{G}_T, g, e) \leftarrow \mathsf{BGGen}(1^k)$
  $h, u', u_1, \ldots, u_\ell \leftarrow_{\mathtt{R}} \mathbb{G}$
  $pp \leftarrow (\mathcal{BG}, h, u', (u_i)_{i \in [\ell]})$
  Return $pp$.
$\mathsf{KG}_{\mathtt{Wat}}(pp)$ :
  $sk \leftarrow_{\mathtt{R}} \mathbb{Z}_p$
  $vk \leftarrow g^{sk}$
  Return $(vk, sk)$.
$\mathsf{Sign}_{\mathtt{Wat}}(pp, sk, m)$ :
  Parse $m$ as $(m_1 \| \ldots \| m_\ell) \in \{0,1\}^\ell$.
  $r \leftarrow_{\mathtt{R}} \mathbb{Z}_p$
  $\sigma_1 \leftarrow h^{sk} \cdot (u' \cdot \prod_{i \in [\ell]} u_i^{m_i})^r$
  $\sigma_2 \leftarrow g^r$
  Return $\sigma \leftarrow (\sigma_1, \sigma_2)$.
$\mathsf{Ver}_{\mathtt{Wat}}(pp, vk, m, \sigma)$ :
  $(\sigma_1, \sigma_2) \leftarrow \sigma$
  Parse $m$ as $(m_1 \| \ldots \| m_\ell) \in \{0,1\}^\ell$.
  If $e(\sigma_2, u' \cdot \prod_{i \in [\ell]} u_i^{m_i}) \cdot e(vk, h) = e(\sigma_1, g)$
        then return $\top$ else return $\bot$.
```

```
$\mathsf{Setup}_{\mathtt{Sch}}(1^k)$ :
  $\mathcal{G} := (p, \mathbb{G}, g) \leftarrow \mathsf{GGen}(1^k)$
  Let $H : \{0,1\}^* \rightarrow \mathbb{Z}_p$
        be a hash function.
  Return $pp \leftarrow (\mathcal{G}, H)$.
$\mathsf{KG}_{\mathtt{Sch}}(pp)$ :
  $sk \leftarrow_{\mathtt{R}} \mathbb{Z}_p$
  $vk \leftarrow g^{sk}$
  Return $(vk, sk)$.
$\mathsf{Sign}_{\mathtt{Sch}}(pp, sk, m)$ :
  $r \leftarrow_{\mathtt{R}} \mathbb{Z}_p$
  $R \leftarrow g^r$
  $h \leftarrow H(R \| m)$
  $s \leftarrow r + (sk) \cdot h \bmod p$
  Return $\sigma \leftarrow (h, s)$.
$\mathsf{Ver}_{\mathtt{Sch}}(pp, vk, m, \sigma)$ :
  $(h, s) \leftarrow \sigma$
  $R \leftarrow g^s \cdot (vk)^{-h}$
  If $H(R \| m) = h$ then
        return $\top$ else return $\bot$.
```

Fig. 3 The Waters signature scheme $\Sigma_{\mathtt{Wat}}$ [36] (left) and the Schnorr signature scheme $\Sigma_{\mathtt{Sch}}$ [31] (right)

Using a group generator GGen, the Schnorr signature scheme $\Sigma_{\mathtt{Sch}} = (\mathsf{Setup}_{\mathtt{Sch}}, \mathsf{KG}_{\mathtt{Sch}}, \mathsf{Sign}_{\mathtt{Sch}}, \mathsf{Ver}_{\mathtt{Sch}})$ is constructed as in Fig. 3 (right). It was formally shown by Pointcheval and Stern [25] that $\Sigma_{\mathtt{Sch}}$ is $\mathtt{EUF\text{-}CMA}$ secure in the random oracle model where the used hash function H is modeled as a random oracle, under the DL assumption with respect to GGen.

3 Special definitions for (ordinary) signatures

In this section, we formalize somewhat less standard and yet natural and useful properties for (ordinary) signature schemes with a simple key generation process, and also show some facts about them that will be utilized in the later sections.

This section is organized as follows: in Sect. 3.1, we formalize certain *homomorphic* properties regarding keys and signatures, and in Sect. 3.2, we introduce a variant of RKA security which we call Φ-$\mathtt{RKA}^*$ security. Finally, in Sect. 3.3, we show some useful facts about them.

3.1 Homomorphic properties

For building our fuzzy signature schemes, we will utilize a signature scheme that has certain homomorphic properties regarding keys and signatures, and thus we formalize the properties here. We define two versions, *normal* and *weak*. The weaker version only requires the first two requirements out of the three, which is sufficient for our security proof for the generic construction for fuzzy signatures given in Sect. 5 to go through. The benefit of considering the normal version will be made clear in Sect. 3.3.

Definition 9 Let $\Sigma = (\mathsf{Setup}, \mathsf{KG}, \mathsf{Sign}, \mathsf{Ver})$ be a signature scheme with a simple key generation process (i.e., there is a deterministic PTA KG' in Definition 7). We say that Σ is *homomorphic* if it satisfies the following three properties:

1. For all parameters pp output by Setup, the signing key space $\mathcal{K}_{pp}$ constitutes an abelian group $(\mathcal{K}_{pp}, +)$.
2. There exists a deterministic PTA M_{vk} that takes a public parameter pp (output by Setup), a verification key vk (output by $\mathsf{KG}(pp)$), and a "shift" $\Delta sk \in \mathcal{K}_{pp}$ as input, and outputs the "shifted" verification key vk'.
   We require for all pp output by Setup and all sk, $\Delta sk \in \mathcal{K}_{pp}$, it holds that
   $$\mathsf{KG}'(pp, sk + \Delta sk) = \mathsf{M}_{\mathsf{vk}}(pp, \mathsf{KG}'(pp, sk), \Delta sk). \quad (2)$$
3. There exists a deterministic PTA $\mathsf{M}_{\mathsf{sig}}$ that takes a public parameter pp (output by Setup), a verification key vk (output by $\mathsf{KG}(pp)$), a message m, a signature σ, and a "shift" $\Delta sk \in \mathcal{K}_{pp}$ as input, and outputs a "shifted" signature σ'.
   We require for all pp output by Setup, all messages m, and all sk, $\Delta sk \in \mathcal{K}_{pp}$, the following two distributions are identical:
   $$\left\{ \sigma' \leftarrow_{\mathrm{R}} \mathsf{Sign}(pp, sk + \Delta sk, m) : \sigma' \right\}, \quad \text{and}$$
   $$\left\{ \begin{array}{l} \sigma \leftarrow_{\mathrm{R}} \mathsf{Sign}(pp, sk, m); \\ \sigma' \leftarrow \mathsf{M}_{\mathsf{sig}}(pp, \mathsf{KG}'(pp, sk), m, \sigma, \Delta sk) \end{array} : \sigma' \right\}. \quad (3)$$

Furthermore, we require for all pp output by Setup, all $sk, \Delta sk \in \mathcal{K}_{pp}$, and all message/signature pairs (m, σ) satisfying $\mathsf{Ver}(pp, \mathsf{KG}'(pp, sk), m, \sigma) = \top$, it holds that

$$\mathsf{Ver}\Big(pp, \mathsf{KG}'(pp, sk + \Delta sk), m, \mathsf{M}_{\mathsf{sig}}(pp, \mathsf{KG}'(pp, sk), m, \sigma, \Delta sk) \Big) = \top. \tag{4}$$

If Σ satisfies only the first two properties, then we say that Σ is *weakly homomorphic*.

Looking ahead, in Sect. 6.4, we will show a variant of the Waters signature scheme [36] (that we call the *modified Waters signature* (MWS) scheme) that satisfies all of the above three properties of the homomorphic property. Furthermore, we note that the Schnorr signature scheme $\Sigma_{\mathtt{Sch}}$ [see Fig. 3 (right)] on which our second instantiation in Sect. 7 is based, satisfies the weak homomorphic property. We will state this in a formal manner in Lemma 6 in Sect. 3.3.

3.2 RKA* security

Here, we introduce an extension of the standard EUF-CMA security for signature schemes, which we call RKA* security, that considers security against an adversary who may mount a kind of related key attacks (RKA).[9] Like the popular definition of RKA security for signature schemes by Bellare et al. [2], RKA* is defined with respect to a class of functions that captures an adversary's ability to modify signing keys. However, our definition has subtle differences from the definition of [2]. The main difference is that in our definition, an adversary is allowed to modify the verification key under which its forgery is verified, while we do not allow an adversary to use a message to be used as its forgery if it has already been signed by the signing oracle. A more detailed explanation on the differences between our definition and the existing RKA security definitions is given in "Appendix B."

Formally, let $\Sigma = (\mathsf{Setup}, \mathsf{KG}, \mathsf{Sign}, \mathsf{Ver})$ be a signature scheme which has a simple key generation process, namely there exists a deterministic PTA KG' such that KG can be written as Eq. (1). Let Φ be a class of functions both of whose domain and range are the secret key space of Σ. For Σ, Φ, and an adversary $\mathcal{A}$, consider the following Φ-RKA* experiment $\mathsf{Expt}^{\Phi\text{-RKA}^*}_{\Sigma,\mathcal{A}}(k)$:

```
$\mathsf{Expt}^{\Phi\text{-RKA}^*}_{\Sigma,\mathcal{A}}(k)$:
    $pp \leftarrow_{\mathrm{R}} \mathsf{Setup}(1^k)$
    $(vk, sk) \leftarrow_{\mathrm{R}} \mathsf{KG}(pp)$
    $\mathcal{Q} \leftarrow \emptyset$
    $(\phi', m', \sigma') \leftarrow_{\mathrm{R}} \mathcal{A}^{\mathcal{O}_{\mathsf{Sign}}(\cdot,\cdot)}(pp, vk)$
    $vk' \leftarrow \mathsf{KG}'(pp, \phi'(sk))$
    If $\phi' \in \Phi \wedge m' \notin \mathcal{Q} \wedge \mathsf{Ver}(pp, vk', m', \sigma') = \top$
                              then return 1 else return 0
```

where $\mathcal{O}_{\mathsf{Sign}}$ is the RKA-signing oracle that takes (the description of) a function $\phi \in \Phi$ and a message m as input, updates the "used message list" $\mathcal{Q}$ by $\mathcal{Q} \leftarrow \mathcal{Q} \cup \{m\}$, and returns a signature $\sigma \leftarrow_{\mathrm{R}} \mathsf{Sign}(pp, \phi(sk), m)$. We stress that in the final step of the experiment, the adversary's forged message/signature pair (m', σ') is verified under the "modified" verification key $vk' = \mathsf{KG}'(pp, \phi'(sk))$.

Definition 10 We say that a signature scheme Σ (with a simple key generation process) is Φ-RKA* secure if for all PPTA adversaries $\mathcal{A}$,

$$\mathsf{Adv}^{\Phi\text{-RKA}^*}_{\Sigma,\mathcal{A}}(k) := \Pr\Big[\mathsf{Expt}^{\Phi\text{-RKA}^*}_{\Sigma,\mathcal{A}}(k) = 1\Big]$$

is negligible.

Note that if we consider Φ to be consisting only of the identity function in the above definition, then we recover the standard EUF-CMA security.

The class of functions. In this paper, we will treat RKA* security with respect to addition, which is captured by the following simple functions (where $\mathcal{K}$ denotes the signing key space of a signature scheme that we assume constitutes an abelian group):

Addition: $\Phi^{\mathrm{add}} := \{\phi^{\mathrm{add}}_a | a \in \mathcal{K}\}$, where $\phi^{\mathrm{add}}_a(x) := x + a$.

3.3 Useful facts

Here, we show some useful facts about the properties introduced in the previous subsections.

Sufficient conditions for Φ^{add}-RKA security.* It turns out that any EUF-CMA secure signature scheme that satisfies the three requirements of the homomorphic property (Definition 9) is automatically Φ^{add}-RKA* secure, and hence these are sufficient conditions for Φ^{add}-RKA* security.

Lemma 5 *Any EUF-CMA secure signature scheme satisfying the homomorphic property (Definition 9) is Φ^{add}-RKA* secure.*

This proof is almost straightforward from the definition of the homomorphic property, and we provide a proof sketch in "Appendix E." It is based on a simple observation that

[9] The asterisk (*) in the security notion indicates that the notion is different from the popular RKA security definition for signatures formalized by Bellare et al. [2].

the homomorphic property allows us to simulate the RKA-signing oracle in the Φ^{add}-$\mathtt{RKA}^*$ security experiment by only using the normal signing oracle (for the same signature scheme).

Weak homomorphic property and Φ^{add}-$\mathtt{RKA}^*$ security of the Schnorr signature scheme. It is straightforward to see that the Schnorr signature scheme $\Sigma_{\mathtt{Sch}}$ [Fig. 3 (right)] admits a simple key generation process and is weakly homomorphic. Specifically, given a public parameter $pp = (\mathcal{G} = (p, \mathbb{G}, g), H)$, we can specify its signing key space to be $\mathbb{Z}_p$, and then the deterministic PTA KG' can be defined by

$$\mathsf{KG}'(pp, sk) := g^{sk},$$

where $sk \in \mathbb{Z}_p$. Furthermore, its signing key space (given a public parameter pp) constitutes an abelian group $(\mathbb{Z}_p, +)$. Therefore, we can talk about its weak homomorphic property and Φ^{add}-$\mathtt{RKA}^*$ security. The following theorem formally states that the Schnorr signature scheme satisfies these functionality/security properties.

Lemma 6 *The Schnorr signature scheme $\Sigma_{\mathtt{Sch}}$ (Fig. 3 (right) in Sect. 2.5) satisfies the weak homomorphic property in the sense of Definition 9. Furthermore, if the DL assumption holds with respect to* GGen*, then $\Sigma_{\mathtt{Sch}}$ is Φ^{add}-$\mathtt{RKA}^*$ secure in the random oracle model where H is modeled as a random oracle.*

The weak homomorphic property should be fairly easy to see: For $vk = g^{sk}$ and $\Delta sk \in \mathbb{Z}_p$, we can just define

$$\begin{aligned}\mathsf{M}_{\mathsf{vk}}(pp, vk, \Delta sk) &:= (vk) \cdot g^{\Delta sk}\\ &= g^{sk+\Delta sk} = \mathsf{KG}'(pp, sk + \Delta sk).\end{aligned}$$

The proof for the Φ^{add}-$\mathtt{RKA}^*$ security can be shown very similarly to the proof of the $\mathtt{EUF}$-$\mathtt{CMA}$ security of the Schnorr scheme using the general forking lemma of Bellare and Neven [3], and its Φ^{add}-weak-RKA security shown by Morita et al. [20,21], and thus we provide its proof in "Appendix F."

4 Definitions for fuzzy signatures

In this section, we introduce the definitions for fuzzy signatures.

As mentioned in Sect. 1, to define fuzzy signatures, we need to first define some "setting" that models a space to which fuzzy data (used as a signing key) belongs, a distribution from which fuzzy data is sampled, etc. We therefore first formalize it as a *fuzzy key setting* in Sect. 4.1, and then define a fuzzy signature scheme that is associated with a fuzzy key setting in Sect. 4.2. Then, we also introduce a new tool that we call *linear sketch*, which is also associated with a fuzzy key setting and will be used as one of the main building blocks in our generic construction of a fuzzy signature scheme given in Sect. 5.

4.1 Fuzzy key setting

Consider a typical biometric authentication scheme: At the registration phase, a "fuzzy" biometric feature $x \in X$ (where X is some metric space) is measured and extracted from a user. Later at the authentication phase, a biometric feature $x' \in X$ is measured and extracted from a (possibly different) user, and this user is considered the user who generated the biometric data x and thus authentic if x and x' are sufficiently "close" according to the metric defined in the space X.

We abstract out and formalize this typical setting for "identifying fuzzy objects" as a *fuzzy key setting*. Roughly, a fuzzy key setting specifies (1) the metric space to which fuzzy data (such as biometric data) belongs (X in the above example), (2) the distribution of fuzzy data sampled at the "registration phase" (x in the above example), and (3) the error distribution that models "fuzziness" of the fuzzy data (the relationship between x and x' in the above example).

We adopt what we call the "universal error model," which assumes that for all objects U that produce fuzzy data that we are interested in, if U produces a data x at the first measurement (say, at the registration phase), and if the same object is measured next time, then the measured data x' follows the distribution $\{e \leftarrow_{\mathrm{R}} \Phi; x' \leftarrow x + e : x'\}$. That is, the error distribution Φ is independent of individual U. (We also assume that the metric space constitutes an abelian group so that addition is well defined.)

Formally, a fuzzy key setting $\mathcal{F}$ consists of $((\mathsf{d}, X), t, \mathcal{X}, \Phi, \epsilon)$, each of which is defined as follows:

(d, X) This is a metric space, where X is a space to which a possible fuzzy data x belongs, and $\mathsf{d} : X^2 \to \mathbb{R}$ is the corresponding distance function. We furthermore assume that X constitutes an abelian group.

t: $(\in \mathbb{R})$ This is the threshold value, determined by a security parameter k. Based on t, the false acceptance rate ($\mathtt{FAR}$) and the false rejection rate ($\mathtt{FRR}$) are determined. We require that $\mathtt{FAR} := \Pr[x, x' \leftarrow_{\mathrm{R}} \mathcal{X} : \mathsf{d}(x, x') < t]$ is negligible in k.

$\mathcal{X}$ This is a distribution of fuzzy data over X.

Φ This is an error distribution (see the above explanation).

ϵ $(\in [0, 1])$ This is an error parameter that represents $\mathtt{FRR}$. We require that for all $x \in X$, $\mathtt{FRR} := \Pr[e \leftarrow_{\mathrm{R}} \Phi : \mathsf{d}(x, x + e) \geq t] \leq \epsilon$.

4.2 Fuzzy signatures

A fuzzy signature scheme Σ_{FS} for a fuzzy key setting $\mathcal{F} = ((\mathsf{d}, X), t, \mathcal{X}, \Phi, \epsilon)$ consists of the four algorithms ($\mathsf{Setup}_{\mathsf{FS}}$, $\mathsf{KG}_{\mathsf{FS}}$, $\mathsf{Sign}_{\mathsf{FS}}$, $\mathsf{Ver}_{\mathsf{FS}}$):

$\mathsf{Setup}_{\mathsf{FS}}$ This is the setup algorithm that takes the description of the fuzzy key setting $\mathcal{F}$ and 1^k as input (where k determines the threshold value t of $\mathcal{F}$), and outputs a public parameter pp.

$\mathsf{KG}_{\mathsf{FS}}$ This is the key generation algorithm that takes pp and a fuzzy data $x \in X$ as input, and outputs a verification key vk.

$\mathsf{Sign}_{\mathsf{FS}}$ This is the signing algorithm that takes pp, a fuzzy data $x' \in X$, and a message m as input, and outputs a signature σ.

$\mathsf{Ver}_{\mathsf{FS}}$ This is the (deterministic) verification algorithm that takes pp, vk, m, and σ as input, and outputs either $\top$ (“accept”) or $\bot$ (“reject”).

δ-correctness. Let $\delta \in [0, 1]$. We say that a fuzzy signature scheme $\Sigma_{\mathsf{FS}} = (\mathsf{Setup}_{\mathsf{FS}}, \mathsf{KG}_{\mathsf{FS}}, \mathsf{Sign}_{\mathsf{FS}}, \mathsf{Ver}_{\mathsf{FS}})$ for a fuzzy key setting $\mathcal{F} = ((\mathsf{d}, X), t, \mathcal{X}, \Phi, \epsilon)$ satisfies δ-correctness if it holds that

$$\Pr\Big[\, pp \leftarrow_{\mathrm{R}} \mathsf{Setup}_{\mathsf{FS}}(1^k);\ x \leftarrow_{\mathrm{R}} \mathcal{X};\ vk \leftarrow_{\mathrm{R}} \mathsf{KG}_{\mathsf{FS}}(pp, x);\ e \leftarrow_{\mathrm{R}} \Phi;\ \sigma \leftarrow_{\mathrm{R}} \mathsf{Sign}_{\mathsf{FS}}(pp, x+e, m):\ \mathsf{Ver}_{\mathsf{FS}}(pp, vk, m, \sigma) = \top \,\Big] \geq 1 - \delta$$

for all $k \in \mathbb{N}$ and all messages m.[10]

$\mathtt{EUF\text{-}CMA}$ *security.* For a fuzzy signature scheme, we consider $\mathtt{EUF\text{-}CMA}$ security in a similar manner to that for an ordinary signature scheme, reflecting the universal error model of a fuzzy key setting.

Formally, for a fuzzy signature scheme $\Sigma_{\mathsf{FS}} = (\mathsf{Setup}_{\mathsf{FS}}, \mathsf{KG}_{\mathsf{FS}}, \mathsf{Sign}_{\mathsf{FS}}, \mathsf{Ver}_{\mathsf{FS}})$ for a fuzzy key setting $\mathcal{F} = ((\mathsf{d}, X), t, \mathcal{X}, \Phi, \epsilon)$ and an adversary $\mathcal{A}$, consider the following $\mathtt{EUF\text{-}CMA}$ experiment $\mathsf{Expt}^{\mathtt{EUF\text{-}CMA}}_{\Sigma_{\mathsf{FS}},\mathcal{F},\mathcal{A}}(k)$:

```
$\mathsf{Expt}^{\mathtt{EUF\text{-}CMA}}_{\Sigma_{\mathsf{FS}},\mathcal{F},\mathcal{A}}(k)$ :
    $pp \leftarrow_{\mathrm{R}} \mathsf{Setup}_{\mathsf{FS}}(\mathcal{F}, 1^k)$
    $x \leftarrow_{\mathrm{R}} \mathcal{X}$
    $vk \leftarrow_{\mathrm{R}} \mathsf{KG}_{\mathsf{FS}}(pp, x)$
    $\mathcal{Q} \leftarrow \emptyset$
    $(m', \sigma') \leftarrow_{\mathrm{R}} \mathcal{A}^{\mathcal{O}_{\mathsf{Sign}_{\mathsf{FS}}}(\cdot)}(pp, vk)$
    If $m' \notin \mathcal{Q} \wedge \mathsf{Ver}_{\mathsf{FS}}(pp, vk, m', \sigma') = \top$
            then return 1 else return 0
```

where $\mathcal{O}_{\mathsf{Sign}_{\mathsf{FS}}}$ is the signing oracle that takes a message m as input, and operates as follows: It updates the “used message list” $\mathcal{Q}$ by $\mathcal{Q} \leftarrow \mathcal{Q} \cup \{m\}$, samples $e \leftarrow_{\mathrm{R}} \Phi$, computes a signature $\sigma \leftarrow_{\mathrm{R}} \mathsf{Sign}_{\mathsf{FS}}(pp, x+e, m)$, and returns σ.

Definition 11 We say that a fuzzy signature scheme Σ_{FS} is $\mathtt{EUF\text{-}CMA}$ secure if for all PPTA adversaries $\mathcal{A}$,

$$\mathsf{Adv}^{\mathtt{EUF\text{-}CMA}}_{\Sigma_{\mathsf{FS}},\mathcal{F},\mathcal{A}}(k) := \Pr\left[\mathsf{Expt}^{\mathtt{EUF\text{-}CMA}}_{\Sigma_{\mathsf{FS}},\mathcal{F},\mathcal{A}}(k) = 1\right]$$

is negligible.

4.3 Linear sketch

Here, we give the definition of a linear sketch scheme. The syntactical definition here is the one we adopt in [19], and we introduce a new security requirement for a linear sketch scheme, which we call *weak simulatability*, which is weaker than the security requirements that we introduced in our earlier versions [19,33], but is nonetheless sufficient for proving the security of our generic construction of a fuzzy signature scheme in the next section. For completeness, we give the definitions in our earlier versions and discuss the differences between the definitions in “Appendix C.”

A linear sketch scheme is associated with a fuzzy key setting and an abelian group (in which addition is well defined), and is defined as follows:

Definition 12 Let $\mathcal{F} = ((\mathsf{d}, X), t, \mathcal{X}, \Phi, \epsilon)$ be a fuzzy key setting. We say that a tuple of PPTAs $\mathcal{S} = (\mathsf{Setup}, \mathsf{Sketch}, \mathsf{DiffRec})$ is a *linear sketch* scheme for $\mathcal{F}$, if it satisfies the following three properties:

Syntax and correctness. Each algorithm of $\mathcal{S}$ has the following interface:

- Setup is the “setup” algorithm that takes the description $\mathcal{F}$ of the fuzzy key setting and the description Λ of an abelian group $(\mathcal{K}, +)$ as input, and outputs a public parameter pp (which we assume contains the information of Λ).
- Sketch is the “sketching” algorithm that takes pp, an element $s \in \mathcal{K}$, and a fuzzy data $x \in X$ as input, and outputs a “sketch” c.
- $\mathsf{DiffRec}$ is the (deterministic) “difference reconstruction” algorithm that takes pp and two values c, c' (supposedly

[10] The definition of correctness here is slightly weakened from the one we used in the earlier versions [19,33], in which we used the following definition: For all $k \in \mathbb{N}$, all pp output by $\mathsf{Setup}_{\mathsf{FS}}(\mathcal{F}, 1^k)$, all $x, x' \in X$ such that $\mathsf{d}(x, x') < t$, and all messages m, it holds that $\mathsf{Ver}_{\mathsf{FS}}(pp, \mathsf{KG}_{\mathsf{FS}}(pp, x), m, \mathsf{Sign}_{\mathsf{FS}}(pp, x', m)) = \top$. Note that this definition implies ϵ-correctness, because $\Pr[x \leftarrow_{\mathrm{R}} \mathcal{X};\ e \leftarrow_{\mathrm{R}} \Phi : d(x, x+e) < t] \geq 1 - \epsilon$. Note also that as long as $\mathtt{FRR}$ is not zero, a fuzzy signature scheme cannot satisfy 0-correctness.

output by Sketch) as input, and outputs the "difference" $\Delta s \in \mathcal{K}$.

We require that for all $x, x' \in X$ such that $\mathsf{d}(x, x') < t$, all pp output by $\mathsf{Setup}(\mathcal{F}, \Lambda)$, and all $s, \Delta s \in \mathcal{K}$, it holds that

$$\mathsf{DiffRec}\Big(pp,\ \mathsf{Sketch}(pp, s, x),\ \mathsf{Sketch}(pp, s+\Delta s, x')\Big) = \Delta s. \tag{5}$$

Linearity. There exists a PPTA $\mathsf{M_c}$ satisfying the following: for all pp output by $\mathsf{Setup}(\mathcal{F}, \Lambda)$, all $x, e \in X$, and for all $s, \Delta s \in \mathcal{K}$, the following two distributions are statistically indistinguishable (in the security parameter k that is associated with t in $\mathcal{F}$):

$$\left\{ \begin{array}{l} c \leftarrow_{\mathrm{R}} \mathsf{Sketch}(pp, s, x); \\ c' \leftarrow_{\mathrm{R}} \mathsf{Sketch}(pp, s + \Delta s, x + e) \end{array} : (c, c') \right\}, \quad \text{and}$$
$$\left\{ \begin{array}{l} c \leftarrow_{\mathrm{R}} \mathsf{Sketch}(pp, s, x); \\ c' \leftarrow_{\mathrm{R}} \mathsf{M_c}(pp, c, \Delta s, e) \end{array} : (c, c') \right\}. \tag{6}$$

Weak Simulatability[11]. Let $\Lambda = (\mathcal{K}, +)$ be a (finite) abelian group. There exists a PPTA simulator Sim such that for all PPTA algorithms $\mathcal{A}$, there exist a positive polynomial[12] u and a negligible function ϵ such that the following inequality holds (where k is the security parameter k associated with t in $\mathcal{F}$):

$$\Pr[\mathcal{A}(\mathcal{D}_{\mathrm{real}}) = 1] \leq u(k) \cdot \Pr[\mathcal{A}(\mathcal{D}_{\mathrm{sim}}) = 1] + \epsilon(k), \tag{7}$$

where the distributions $\mathcal{D}_{\mathrm{real}}$ and $\mathcal{D}_{\mathrm{sim}}$ are defined as follows:

$$\mathcal{D}_{\mathrm{real}} := \left\{ \begin{array}{l} pp \leftarrow_{\mathrm{R}} \mathsf{Setup}(\mathcal{F}, \Lambda);\ x \leftarrow_{\mathrm{R}} \mathcal{X}; \\ s \leftarrow_{\mathrm{R}} \mathcal{K};\ c \leftarrow_{\mathrm{R}} \mathsf{Sketch}(pp, s, x) \end{array} : (pp, s, c) \right\},$$
$$\mathcal{D}_{\mathrm{sim}} := \left\{ \begin{array}{l} pp \leftarrow_{\mathrm{R}} \mathsf{Setup}(\mathcal{F}, \Lambda);\ s \leftarrow_{\mathrm{R}} \mathcal{K}; \\ c \leftarrow_{\mathrm{R}} \mathsf{Sim}(pp) \end{array} : (pp, s, c) \right\}.$$

We remark that the definition of weak simulatability is strictly weaker than the simulatability and the average-case indistinguishability that we used in our earlier versions [19,33]. In particular, we only require it to hold for a computationally bounded adversary, and unlike a typical simulation-based security notion we allow not only the additive simulation error (captured by $\epsilon(k)$) but also the *multiplicative* simulation error that is captured by $u(k)$ in Eq. (7). As mentioned above, these relaxations are still sufficient to prove the security of our generic construction in the next section.

5 Generic construction

In this section, we show a generic construction of a fuzzy signature scheme. Our construction uses an ordinary signature scheme (with the weak homomorphic property) and a linear sketch scheme as building blocks. The fuzzy key setting for which the fuzzy signature scheme is constructed is the one with which the underlying linear sketch scheme is associated.

We have already provided an overview of our generic construction in Sect. 1.3. Thus, we directly proceed to the construction in Sect. 5.1. We then provide the proof for correctness in Sect. 5.2, and finally the proof for security in Sect. 5.3.

5.1 Description of the construction

Let $\mathcal{F} = ((d, X), t, \mathcal{X}, \Phi, \epsilon)$ be a fuzzy key setting, and let $\mathcal{S} = (\mathsf{Setup}_l, \mathsf{Sketch}, \mathsf{DiffRec})$ be a linear sketch for $\mathcal{F}$. Let $\Sigma = (\mathsf{Setup}_s, \mathsf{KG}, \mathsf{Sign}, \mathsf{Ver})$ be a signature scheme with a simple key generation process (i.e., there exists a deterministic PTA KG'). We assume that Σ is weakly homomorphic (as per Definition 9), namely its secret key space (given pp) is an abelian group $(\mathcal{K}_{pp}, +)$ and has the additional algorithm $\mathsf{M_{vk}}$. Using $\mathcal{S}$ and Σ, the generic construction of a fuzzy signature scheme $\Sigma_{\mathsf{FS}} = (\mathsf{Setup_{FS}}, \mathsf{KG_{FS}}, \mathsf{Sign_{FS}}, \mathsf{Ver_{FS}})$ for the fuzzy key setting $\mathcal{F}$ is constructed as in Fig. 4.

5.2 Correctness

The correctness of the fuzzy signature scheme Σ_{FS} is guaranteed as follows.

Theorem 1 *If Σ and $\mathcal{S}$ satisfy correctness, then the fuzzy signature scheme Σ_{FS} in Fig. 4 is ϵ-correct.*

Proof of Theorem 1 Fix arbitrarily a message m. Let $x, x' \in X$ such that $\mathsf{d}(x, x') < t$. Also, let $pp = (pp_s, pp_l)$ be a public parameter output by $\mathsf{Setup_{FS}}(\mathcal{F}, 1^k)$, let $VK = (vk = \mathsf{KG}'(pp_s, sk), c)$ be a verification key output by $\mathsf{KG_{FS}}(pp, x)$, and let $\sigma = (\widetilde{vk} = \mathsf{KG}'(pp_s, \widetilde{sk}), \widetilde{\sigma}, \widetilde{c})$ be a signature output by $\mathsf{Sign_{FS}}(pp, x', m)$.

Recall that by the definition of the fuzzy key setting $\mathcal{F}$, we have $\Pr[e \leftarrow_{\mathrm{R}} \Phi : \mathsf{d}(x, x + e) < t] \geq 1 - \epsilon$. Hence, to prove the theorem, it is sufficient to show that if $\mathsf{d}(x, x') < t$, then it always holds that $\mathsf{Ver_{FS}}(pp, VK, m, \sigma) = \top$, which we do in the following.

Firstly, since $\widetilde{\sigma}$ is a signature of the message m generated using the signing key $\widetilde{sk}$, and $\widetilde{vk}$ is the verification key corresponding to $\widetilde{sk}$, we have $\mathsf{Ver}(pp_s, \widetilde{vk}, m, \widetilde{\sigma}) = \top$ due to the correctness of the underlying signature scheme Σ. Secondly, $\mathsf{d}(x, x') < t$ implies $\mathsf{DiffRec}(pp_l, c, \widetilde{c}) = \widetilde{sk} - sk$ due to the correctness of the underlying linear sketch scheme $\mathcal{S}$.

[11] The choice of the word "weak" in weak simulatability is because it is a weaker requirement than the simulatability we used in [33] in several aspects. See the explanation given after the definition.

[12] We call u a *multiplicative simulation error*.

$\mathsf{Setup}_{\mathsf{FS}}(\mathcal{F}, 1^k)$:	$\mathsf{KG}_{\mathsf{FS}}(pp, x)$:	$\mathsf{Sign}_{\mathsf{FS}}(pp, x', m)$:	$\mathsf{Ver}_{\mathsf{FS}}(pp, VK, m, \sigma)$:
$pp_s \leftarrow_{\mathrm{R}} \mathsf{Setup}_s(1^k)$ Let $\Lambda := (\mathcal{K}_{pp_s}, +)$. $pp_l \leftarrow_{\mathrm{R}} \mathsf{Setup}_l(\mathcal{F}, \Lambda)$ Return $pp \leftarrow (pp_s, pp_l)$.	$(pp_s, pp_l) \leftarrow pp$ $sk \leftarrow_{\mathrm{R}} \mathcal{K}_{pp_s}$ $vk \leftarrow \mathsf{KG}'(pp_s, sk)$ $c \leftarrow_{\mathrm{R}} \mathsf{Sketch}(pp_l, sk, x)$ Return $VK \leftarrow (vk, c)$.	$(pp_s, pp_l) \leftarrow pp$ $\widetilde{sk} \leftarrow_{\mathrm{R}} \mathcal{K}_{pp_s}$ $\widetilde{vk} \leftarrow \mathsf{KG}'(pp_s, \widetilde{sk})$ $\widetilde{\sigma} \leftarrow_{\mathrm{R}} \mathsf{Sign}(pp_s, \widetilde{sk}, m)$ $\widetilde{c} \leftarrow_{\mathrm{R}} \mathsf{Sketch}(pp_l, \widetilde{sk}, x')$ Return $\sigma \leftarrow (\widetilde{vk}, \widetilde{\sigma}, \widetilde{c})$.	$(pp_s, pp_l) \leftarrow pp$ $(vk, c) \leftarrow VK$ $(\widetilde{vk}, \widetilde{\sigma}, \widetilde{c}) \leftarrow \sigma$ If $\mathsf{Ver}(pp_s, \widetilde{vk}, m, \widetilde{\sigma}) = \bot$ then return $\bot$. $\Delta sk \leftarrow \mathsf{DiffRec}(pp_l, c, \widetilde{c})$ If $\mathsf{M_{vk}}(pp_s, vk, \Delta sk) = \widetilde{vk}$ then return $\top$ else return $\bot$.

Fig. 4 Our generic construction of a fuzzy signature scheme Σ_{FS} for a fuzzy key setting $\mathcal{F}$, based on a signature scheme Σ with the weak homomorphic property and a linear sketch scheme $\mathcal{S}$ for $\mathcal{F}$

Thirdly, due to the weak homomorphic property of Σ, letting $\Delta sk := \widetilde{sk} - sk$, we have

$$\begin{aligned}\mathsf{M_{vk}}(pp_s, vk, \Delta sk) &= \mathsf{M_{vk}}(pp_s, \mathsf{KG}'(pp_s, sk), \Delta sk)\\ &= \mathsf{KG}'(pp_s, sk + \Delta sk) = \mathsf{KG}'(pp_s, \widetilde{sk}) = \widetilde{vk}.\end{aligned}$$

The conditions seen so far are exactly those checked in the verification algorithm $\mathsf{Ver}_{\mathsf{FS}}(pp, VK, m, \sigma)$, and hence its output is guaranteed to be $\top$, as required. □

5.3 Security

The security of the fuzzy signature scheme Σ_{FS} is guaranteed as follows.

Theorem 2 *If Σ is Φ^{add}-*RKA* secure and $\mathcal{S}$ is a linear sketch scheme for $\mathcal{F}$ (in the sense of Definition 12), then the fuzzy signature scheme Σ_{FS} for $\mathcal{F}$ in Fig. 4 is* EUF-CMA *secure.*

Our proof is via the sequence of games argument. We gradually change the original EUF-CMA security experiment for an adversary $\mathcal{A}$ against our construction Σ_{FS} by using the weak homomorphic property of the underlying signature scheme Σ and the linearity property and weak simulatability of the underlying linear sketch scheme $\mathcal{S}$, so that $\mathcal{A}$'s success probability in the original EUF-CMA security experiment is not non-negligibly different from $\mathcal{A}$'s success probability in the final game (Game 5), and the latter is negligible due to the Φ^{add}-RKA* security of Σ.

Proof of Theorem 2 Let $\mathcal{A}$ be an arbitrary PPTA adversary that attacks the EUF-CMA security of Σ_{FS}. Below, we consider a sequence of five games, where the first game is $\mathsf{Expt}^{\mathtt{EUF\text{-}CMA}}_{\Sigma_{\mathsf{FS}},\mathcal{F},\mathcal{A}}(k)$ itself. For $i \in [5]$, let S_i be the event that in Game i, $\mathcal{A}$ succeeds in outputting a successful forgery (m', σ') satisfying $\mathsf{Ver}_{\mathsf{FS}}(pp, VK, m', \sigma') = \top$ and $m' \notin \mathcal{Q}$. Our goal is to show that $\mathsf{Adv}^{\mathtt{EUF\text{-}CMA}}_{\Sigma_{\mathsf{FS}},\mathcal{F},\mathcal{A}}(k) = \Pr[\mathsf{S}_1]$ is negligible.

Game 1 This is the EUF-CMA experiment $\mathsf{Expt}^{\mathtt{EUF\text{-}CMA}}_{\Sigma_{\mathsf{FS}},\mathcal{F},\mathcal{A}}(k)$. In this game, the public parameter pp and the verification key VK are generated as follows:

Generation of pp and VK in Game 1:

```
$pp_s \leftarrow_{\mathrm{R}} \mathsf{Setup}_s(1^k)$
$pp_l \leftarrow_{\mathrm{R}} \mathsf{Setup}_l(\mathcal{F}, \Lambda := (\mathcal{K}_{pp_s}, +))$
$pp \leftarrow (pp_s, pp_l)$
$x \leftarrow_{\mathrm{R}} \mathcal{X}$
$sk \leftarrow_{\mathrm{R}} \mathcal{K}_{pp_s}$
$vk \leftarrow \mathsf{KG}'(pp_s, sk)$
$c \leftarrow_{\mathrm{R}} \mathsf{Sketch}(pp_l, sk, x)$
$VK \leftarrow (vk, c)$
```

Furthermore, the signing oracle $\mathcal{O}_{\mathsf{Sign}_{\mathsf{FS}}}(m)$ generates a signature σ as follows:

Signing oracle $\mathcal{O}_{\mathsf{Sign}_{\mathsf{FS}}}(m)$ in Game 1:

```
$e \leftarrow_{\mathrm{R}} \Phi$
$\widetilde{sk} \leftarrow_{\mathrm{R}} \mathcal{K}_{pp_s}$
$\widetilde{vk} \leftarrow \mathsf{KG}'(pp_s, \widetilde{sk})$
$\widetilde{\sigma} \leftarrow_{\mathrm{R}} \mathsf{Sign}(pp_s, \widetilde{sk}, m)$
$\widetilde{c} \leftarrow_{\mathrm{R}} \mathsf{Sketch}(pp_l, \widetilde{sk}, x + e)$
$\sigma \leftarrow (\widetilde{vk}, \widetilde{\sigma}, \widetilde{c})$
```

Game 2 This game is the same as Game 1, except that in the signing oracle, $\widetilde{sk}$ is generated by firstly picking a random "difference" $\Delta sk \in \mathcal{K}_{pp_s}$, and then setting $\widetilde{sk} \leftarrow sk + \Delta sk$.

More specifically, the signing oracle $\mathcal{O}_{\mathsf{Sign}_{\mathsf{FS}}}(m)$ in this game generates a signature σ as follows: (The difference from Game 1 is underlined.)

Signing oracle $\mathcal{O}_{\mathsf{Sign}_{\mathsf{FS}}}(m)$ in Game 2:

```
$e \leftarrow_{\mathrm{R}} \Phi$
$\underline{\Delta sk \leftarrow_{\mathrm{R}} \mathcal{K}_{pp_s}}$
$\underline{\widetilde{sk} \leftarrow sk + \Delta sk}$
$\widetilde{vk} \leftarrow \mathsf{KG}'(pp_s, \widetilde{sk})$
$\widetilde{\sigma} \leftarrow_{\mathrm{R}} \mathsf{Sign}(pp_s, \widetilde{sk}, m)$
$\widetilde{c} \leftarrow_{\mathrm{R}} \mathsf{Sketch}(pp_l, \widetilde{sk}, x + e)$
$\sigma \leftarrow (\widetilde{vk}, \widetilde{\sigma}, \widetilde{c})$
```

Since the distribution of $\widetilde{sk}$ in Game 2 and that in Game 1 are identical, we have $\Pr[\mathsf{S}_2] = \Pr[\mathsf{S}_1]$.

Game 3 This game is the same as Game 2, except that in the signing oracle, $\widetilde{vk}$ is generated by using vk and Δsk via M_{vk}.

More specifically, the signing oracle $\mathcal{O}_{\mathsf{Sign}_{\mathsf{FS}}}(m)$ in this game generates a signature σ as follows: (The difference from Game 2 is underlined.)

Signing oracle $\mathcal{O}_{\mathsf{Sign}_{\mathsf{FS}}}(m)$ in Game 3:

```
$e \leftarrow_{\mathrm{R}} \Phi$
$\Delta sk \leftarrow_{\mathrm{R}} \mathcal{K}_{pp_s}$
$\widetilde{sk} \leftarrow sk + \Delta sk$
$\underline{\widetilde{vk} \leftarrow \mathsf{M}_{\mathsf{vk}}(pp_s, vk, \Delta sk)}$
$\widetilde{\sigma} \leftarrow_{\mathrm{R}} \mathsf{Sign}(pp_s, \widetilde{sk}, m)$
$\widetilde{c} \leftarrow_{\mathrm{R}} \mathsf{Sketch}(pp_l, \widetilde{sk}, x + e)$
$\sigma \leftarrow (\widetilde{vk}, \widetilde{\sigma}, \widetilde{c})$
```

By the property of M_{vk} [Eq. (2)], the distribution of $\widetilde{vk}$ in Game 3 and that in Game 2 are identical, and thus we have $\Pr[\mathsf{S}_3] = \Pr[\mathsf{S}_2]$.

Game 4 This game is the same as Game 3, except that in the signing oracle, $\widetilde{c}$ is generated by using c, e, and Δsk, via the auxiliary algorithm M_{c} of the linear sketch scheme $\mathcal{S}$.

More specifically, the signing oracle $\mathcal{O}_{\mathsf{Sign}_{\mathsf{FS}}}(m)$ in this game generates a signature σ as follows: (The difference from Game 3 is underlined.)

Signing oracle $\mathcal{O}_{\mathsf{Sign}_{\mathsf{FS}}}(m)$ in Game 4:

```
$e \leftarrow_{\mathrm{R}} \Phi$
$\Delta sk \leftarrow_{\mathrm{R}} \mathcal{K}_{pp_s}$
$\widetilde{sk} \leftarrow sk + \Delta sk$
$\widetilde{vk} \leftarrow \mathsf{M}_{\mathsf{vk}}(pp_s, vk, \Delta sk)$
$\widetilde{\sigma} \leftarrow_{\mathrm{R}} \mathsf{Sign}(pp_s, \widetilde{sk}, m)$
$\underline{\widetilde{c} \leftarrow_{\mathrm{R}} \mathsf{M}_{\mathsf{c}}(pp_l, c, \Delta sk, e)}$
$\sigma \leftarrow (\widetilde{vk}, \widetilde{\sigma}, \widetilde{c})$
```

By the linearity of the linear sketch scheme $\mathcal{S}$, the distribution of $\widetilde{c}$ generated in the signing oracle in Game 4 and that in Game 3 are statistically indistinguishable. We can apply this statistical indistinguishability query-by-query, to conclude that $\mathcal{A}$'s view in Game 4 and that in Game 3 are statistically indistinguishable.[13] This guarantees that $|\Pr[\mathsf{S}_4] - \Pr[\mathsf{S}_3]|$ is negligible.

Game 5 This game is the same as Game 4, except that the sketch c contained in VK is generated by the simulator Sim (without using $x \in \mathcal{X}$ or $sk \in \mathcal{K}_{pp_s}$), whose existence is guaranteed by the weak simulatability of the linear sketch scheme $\mathcal{S}$.

More specifically, in this game, the public parameter pp and the verification key VK are generated as follows: (The difference from Game 4 is underlined.)

Generation of pp and VK in Game 5:

```
$pp_s \leftarrow_{\mathrm{R}} \mathsf{Setup}_s(1^k)$
$pp_l \leftarrow_{\mathrm{R}} \mathsf{Setup}_l(\mathcal{F}, \Lambda := (\mathcal{K}_{pp_s}, +))$
$pp \leftarrow (pp_s, pp_l)$
$sk \leftarrow_{\mathrm{R}} \mathcal{K}_{pp_s}$
$vk \leftarrow \mathsf{KG}'(pp_s, sk)$
$\underline{c \leftarrow_{\mathrm{R}} \mathsf{Sim}(pp_l)}$
$VK \leftarrow (vk, c)$
```

(We no longer pick $x \in \mathcal{X}$, because it is not used in Game 5.)

Now, we show that due to the weak simulatability of the linear sketch scheme $\mathcal{S}$, there exists a polynomial $u = u(k)$ and a negligible function $\epsilon = \epsilon(k)$ such that $\Pr[\mathsf{S}_4] \leq u \cdot \Pr[\mathsf{S}_5] + \epsilon$ holds. To see this, let $pp_s \leftarrow_{\mathrm{R}} \mathsf{Setup}_s(1^k)$, and let $\Lambda = (\mathcal{K}_{pp_s}, +)$ be the abelian group that describes the secret key space of Σ. Then, consider the PPTA adversary $\mathcal{B}'$ that has pp_s hardwired, takes as input a tuple (pp_l, sk, c) that is generated by either

$$\mathcal{D}_{\mathrm{real}} = \left\{ \begin{array}{l} pp_l \leftarrow_{\mathrm{R}} \mathsf{Setup}_l(\mathcal{F}, \Lambda); \\ x \leftarrow_{\mathrm{R}} \mathcal{X};\ sk \leftarrow_{\mathrm{R}} \mathcal{K}_{pp_s}; \\ c \leftarrow_{\mathrm{R}} \mathsf{Sketch}(pp_l, sk, x) \end{array} : (pp_l, sk, c) \right\} \quad \text{or}$$

$$\mathcal{D}_{\mathrm{sim}} = \left\{ \begin{array}{l} pp_l \leftarrow_{\mathrm{R}} \mathsf{Setup}_l(\mathcal{F}, \Lambda); \\ sk \leftarrow_{\mathrm{R}} \mathcal{K}_{pp_s};\ c \leftarrow_{\mathrm{R}} \mathsf{Sim}(pp_l) \end{array} : (pp_l, sk, c) \right\},$$

simulates Game 4 for $\mathcal{A}$ by using these values,[14] and outputs 1 if and only if $\mathcal{A}$ succeeds in forging a signature. Then, it is straightforward to see that if the input (pp_l, sk, c) to $\mathcal{B}'$ comes from the distribution $\mathcal{D}_{\mathrm{real}}$ (resp. $\mathcal{D}_{\mathrm{sim}}$), then $\mathcal{B}'$ simulates Game 4 (resp. Game 5) in which pp_s is the one hardwired in $\mathcal{B}'$, perfectly for $\mathcal{A}$. Consequently, we have

$$\mathop{\mathbf{E}}_{pp_s \leftarrow_{\mathrm{R}} \mathsf{Setup}_s(1^k)} \left[\Pr[\mathcal{B}'(\mathcal{D}_{\mathrm{real}}) = 1] \right] = \Pr[\mathsf{S}_4], \quad \text{and}$$

$$\mathop{\mathbf{E}}_{pp_s \leftarrow_{\mathrm{R}} \mathsf{Setup}_s(1^k)} \left[\Pr[\mathcal{B}'(\mathcal{D}_{\mathrm{sim}}) = 1] \right] = \Pr[\mathsf{S}_5].$$

Also, by the weak simulatability of $\mathcal{S}$, it holds that $\Pr[\mathcal{B}'(\mathcal{D}_{\mathrm{real}}) = 1] \leq u \cdot \Pr[\mathcal{B}'(\mathcal{D}_{\mathrm{sim}}) = 1] + \epsilon$. Hence, by the linearity of expectation, we obtain

$$\Pr[\mathsf{S}_4] \leq u \cdot \Pr[\mathsf{S}_5] + \epsilon.$$

[13] If the statistical distance between the distributions considered in the linearity property [Eq. (6)] is a negligible value ϵ_{lin} and the adversary $\mathcal{A}$ makes $q = q(k)$ signing queries (where q is some polynomial), the difference $|\Pr[\mathsf{S}_4] - \Pr[\mathsf{S}_3]|$ is at most $q \cdot \epsilon_{\mathrm{lin}}$, which is still negligible.

[14] That is, $\mathcal{B}'$ runs $\mathcal{A}$ on input the public parameter $pp = (pp_s, pp_l)$ and the verification key $VK = (vk = \mathsf{KG}'(pp_s, sk), c)$, and answers $\mathcal{A}$'s signing queries as in the signing oracle in Game 4 does.

Putting everything together, we can estimate an upperbound of $\mathcal{A}$'s $\texttt{EUF-CMA}$ advantage as follows:

$$\begin{aligned}\mathsf{Adv}^{\texttt{EUF-CMA}}_{\Sigma_{\mathsf{FS}},\mathcal{F},\mathcal{A}}(k) &= \Pr[\mathsf{S}_1]\\ &\leq \sum_{i\in[3]}\Big|\Pr[\mathsf{S}_i]-\Pr[\mathsf{S}_{i+1}]\Big| + \Pr[\mathsf{S}_4]\\ &\leq \sum_{i\in[3]}\Big|\Pr[\mathsf{S}_i]-\Pr[\mathsf{S}_{i+1}]\Big| + u(k)\cdot\Pr[\mathsf{S}_5]+\epsilon(k),\\ &\leq u(k)\cdot\Pr[\mathsf{S}_5]+\epsilon'(k),\end{aligned}$$

where $u(k)$ is a polynomial and $\epsilon(k)$ is a negligible function that are both due to the weak simulatability of the linear sketch scheme $\mathcal{S}$ as seen above, and ϵ' is another negligible function such that $\epsilon' = \epsilon + |\Pr[\mathsf{S}_3] - \Pr[\mathsf{S}_4]|$. (Recall that $\Pr[\mathsf{S}_1] = \Pr[\mathsf{S}_2] = \Pr[\mathsf{S}_3]$.)

Hence, in order to complete the proof, it is sufficient to show that $\Pr[\mathsf{S}_5]$ is negligible. We show this by relying on the Φ^{add}-$\texttt{RKA}^*$ security of the underlying signature scheme Σ. Specifically, using $\mathcal{A}$ as a building block, we construct the following PPTA adversary $\mathcal{B}$ that attacks the Φ^{add}-$\texttt{RKA}^*$ security of the underlying signature scheme Σ:

> $\mathcal{B}^{\mathcal{O}_{\mathsf{Sign}}(\cdot,\cdot)}(pp_s, vk)$: Let $\Lambda := (\mathcal{K}_{pp_s}, +)$. $\mathcal{B}$ first generates $pp_l \leftarrow_{\mathrm{R}} \mathsf{Setup}_l(\mathcal{F}, \Lambda)$ and sets $pp \leftarrow (pp_s, pp_l)$. Next, $\mathcal{B}$ computes $c \leftarrow_{\mathrm{R}} \mathsf{Sim}(pp_l)$, and then sets $VK \leftarrow (vk, c)$. Then, $\mathcal{B}$ runs $\mathcal{A}(pp, VK)$.
>
> For each signing query m from $\mathcal{A}$, $\mathcal{B}$ responds as follows:
>
> 1. Pick $e \leftarrow_{\mathrm{R}} \Phi$ and $\Delta sk \leftarrow_{\mathrm{R}} \mathcal{K}_{pp_s}$.
> 2. Submit $(\phi^{\mathrm{add}}_{\Delta sk}, m)$ to its own RKA-signing oracle $\mathcal{O}_{\mathsf{Sign}}$, and receive the result $\widetilde{\sigma}$. (Note that by definition, $\widetilde{\sigma}$ is computed by $\widetilde{\sigma} \leftarrow_{\mathrm{R}} \mathsf{Sign}(pp_s, sk + \Delta sk, m)$, where sk is the original signing key corresponding to vk that $\mathcal{B}$ received.)
> 3. Compute $\widetilde{vk} \leftarrow \mathsf{M}_{\mathsf{vk}}(pp_s, vk, \Delta sk)$ and $\widetilde{c} \leftarrow_{\mathrm{R}} \mathsf{M}_{\mathsf{c}}(pp_l, c, \Delta sk, e)$.
> 4. Return $\sigma = (\widetilde{vk}, \widetilde{\sigma}, \widetilde{c})$ to $\mathcal{A}$ as the result of the signing query.
>
> When $\mathcal{A}$ outputs $(m', \sigma' = (\widetilde{vk}', \widetilde{\sigma}', \widetilde{c}'))$ and terminates, $\mathcal{B}$ computes $\Delta sk' \leftarrow \mathsf{DiffRec}(pp_l, c, \widetilde{c}')$, and terminates with output $(\phi^{\mathrm{add}}_{\Delta sk'}, m', \widetilde{\sigma}')$.

The above completes the description of $\mathcal{B}$. It is not hard to see that $\mathcal{B}$ perfectly simulates Game 5 for $\mathcal{A}$. In particular, $\mathcal{B}$ generates pp and $VK = (vk, c)$ in exactly the same way as Game 5. Furthermore, since $\mathcal{B}$ can ask a RKA-signing query of the form $(\phi^{\mathrm{add}}_{\Delta sk}, m)$ in the Φ^{add}-$\texttt{RKA}^*$ experiment and is given a signature $\widetilde{\sigma}$ computed by using the "shifted" secret key $sk + \Delta sk$, we can view $sk + \Delta sk$ as $\widetilde{sk}$ generated for answering each signing query in Game 5. Note also that the "used messages list" $\mathcal{Q}$ by $\mathcal{A}$ and that of $\mathcal{B}$ are identical.

We finally show that whenever $\mathcal{A}$ succeeds in outputting a successful forgery pair $(m', \sigma' = (\widetilde{vk}', \widetilde{\sigma}', \widetilde{c}'))$ such that $\mathsf{Ver}_{\mathsf{FS}}(pp, VK, m', \sigma') = \top$, $\mathcal{B}$ also succeeds in outputting a successful forgery $(\phi^{\mathrm{add}}_{\Delta sk'}, m', \widetilde{\sigma}')$, such that

$$\begin{aligned}&\mathsf{Ver}(pp_s, \mathsf{KG}'(pp_s, sk + \Delta sk'), m', \widetilde{\sigma}') = \top\\ &\quad \text{where}\ \ \Delta sk' = \mathsf{DiffRec}(pp_l, c, \widetilde{c}'). \qquad (8)\end{aligned}$$

To see this, note that $\mathsf{Ver}_{\mathsf{FS}}(pp, VK, m', \sigma') = \top$ implies that $\mathsf{Ver}(pp_s, \widetilde{vk}', m', \widetilde{\sigma}') = \top$, $\mathsf{DiffRec}(pp_l, c, \widetilde{c}') = \Delta sk'$, and $\mathsf{M}_{\mathsf{vk}}(pp_s, vk, \Delta sk') = \widetilde{vk}'$ hold. The last condition implies $\widetilde{vk}' = \mathsf{KG}'(pp_s, sk + \Delta sk')$ due to the weak homomorphic property of Σ. Thus, if $\mathcal{A}$'s output (m', σ') satisfies the condition of violating the $\texttt{EUF-CMA}$ security of Σ_{FS}, $\mathcal{B}$'s output $(\phi^{\mathrm{add}}_{\Delta sk'}, m', \widetilde{\sigma}')$ satisfies the condition of violating the Φ^{add}-$\texttt{RKA}^*$ security of the underlying signature scheme Σ. Hence, we have $\mathsf{Adv}^{\Phi^{\mathrm{add}}\text{-}\texttt{RKA}^*}_{\Sigma,\mathcal{B}}(k) = \Pr[\mathsf{S}_5]$. Since Σ is assumed to be Φ^{add}-$\texttt{RKA}^*$ secure and $\mathcal{B}$ is a PPTA, we can conclude that $\Pr[\mathsf{S}_5]$ is negligible.

At this point, we have shown that $\mathsf{Adv}^{\texttt{EUF-CMA}}_{\Sigma_{\mathsf{FS}},\mathcal{F},\mathcal{A}}(k)$ is upperbounded to be negligible. This completes the proof of Theorem 2. □

6 First instantiation

This and next sections give the concrete instantiations of our generic construction of a fuzzy signature scheme given in Sect. 5. In this section, we give our first instantiation based on the Waters signature scheme [36] that uses bilinear groups and the security is proven in the standard model. One strong requirement of this instantiation is that it needs to assume that the fuzzy data is distributed uniformly. (This requirement is relaxed in our second instantiation given in the next section.)

The rest of this section is organized as follows. Since we treat real numbers in our instantiations (in this and next sections), below we first clarify how we treat real numbers. Then in Sect. 6.1, we first specify a concrete fuzzy key setting $\mathcal{F}_1$ for which our first instantiation is constructed. Next, in Sect. 6.2, we provide some mathematical preliminaries. Armed with them, in Sects. 6.3 and 6.4, we show the concrete linear sketch scheme $\mathcal{S}_{\texttt{CRT}}$ for $\mathcal{F}_1$ and the signature scheme $\Sigma_{\texttt{MWS}}$, respectively, which are used to instantiate the building blocks of our generic construction. The final description of the first instantiation of our fuzzy signature scheme, Σ_{FS1}, is given in Sect. 6.5.

On the treatment of real numbers. In this and next sections, we use real numbers to represent and process fuzzy data. We assume that a suitable representation with sufficient accuracy is chosen to encode the real numbers whenever they need to be treated by the considered algorithms.

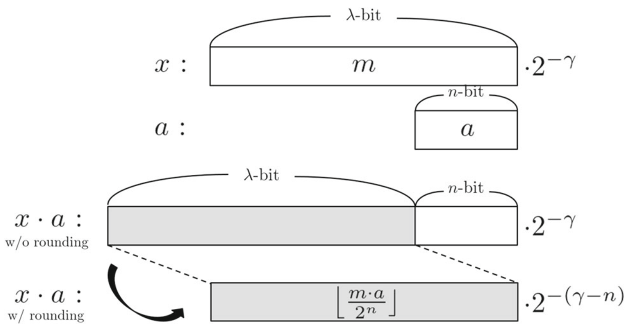

Fig. 5 An illustration of multiplication of a real number $x = \frac{m}{2^\gamma}$ and an n-bit integer a

Concretely, we assume that the significand of all real numbers is expressed in an a priori fixed length (in bits) λ, where λ is some natural number that is a polynomial of a security parameter k. That is, a real number is expressed in the form $\frac{m}{2^\gamma}$, where m is a λ-bit integer that represents the significand and $-\gamma \in \mathbb{Z}$ is the exponent. (For ease of treatment of decimal numbers, we use the convention that a positive γ implies a negative exponent.) Furthermore, if real numbers are involved in some arithmetic operations such as addition and multiplication, then the rounding-down operation is naturally applied to the significand of the resulting number, so that the result is always expressed in the above form (i.e., its significand is expressed with λ bits). We stress that this setting is natural, taking computer implementations into account.

For example, if we multiply a real number $x = \frac{m}{2^\gamma}$ (where m is a λ-bit integer and $0 \le \gamma \le \lambda$) with an n-bit integer a (where $n \le \gamma$), then the resulting number $x \cdot a$ of the multiplication of x and a is treated as

$$\left\lfloor \frac{m \cdot a}{2^n} \right\rfloor \cdot 2^{-(\gamma-n)}. \tag{9}$$

That is, its significand is a λ-bit integer $\lfloor \frac{m \cdot a}{2^n} \rfloor$ and its exponent is $-(\gamma - n)$. This might not look straightforward at first glance, but note that the significand $\lfloor \frac{m \cdot a}{2^n} \rfloor$ is the result of the multiplication $m \cdot a$ rounded down to have a λ-bit precision (the denominator 2^n is due to the fact that a is an n-bit integer). The exponent is correspondingly "shifted" to take into account that a is an n-bit integer. See Fig. 5 for an illustration for the calculation of $x \cdot a$. [Such multiplication of a real number in $[0, 1)$ with an integer appears in our concrete instantiations of linear sketch schemes in Sects. 6.3 and 7.2 (and thus in the final descriptions of our concrete fuzzy signature schemes that appear in Sects. 6.5 and 7.3).]

6.1 Specific fuzzy key setting

Here, we specify a concrete fuzzy key setting $\mathcal{F}_1 = ((\mathsf{d}, X), t, \mathcal{X}, \Phi, \epsilon)$ for which our first fuzzy signature scheme Σ_{FS1} is constructed.

Metric space (d, X). We define the space X by $X := [0, 1)^n \subset \mathbb{R}^n$, where n is a parameter specified by the context (e.g., an object from which we measure fuzzy data). We use the L_∞-distance as the distance function $\mathsf{d} : X \times X \to \mathbb{R}$. Namely, for $\mathbf{x} = (x_1, \ldots, x_n) \in X$ and $\mathbf{x}' = (x'_1, \ldots, x'_n) \in X$, we define $\mathsf{d}(\mathbf{x}, \mathbf{x}') := \|\mathbf{x} - \mathbf{x}'\|_\infty := \max_{i \in [n]} |x_i - x'_i|$. Note that X forms an abelian group with respect to coordinate-wise addition (modulo 1).

Threshold t. For a security parameter k, we define the threshold $t \in \mathbb{R}$ so that

$$k = \lfloor -n \log_2(2t) \rfloor. \tag{10}$$

Looking ahead, this guarantees that the algorithm "WGen" that we will introduce in the next subsection, is a PTA in k.

Furthermore, we require that $n = O(\log_2 k)$, so that 2^n can be considered to be upperbounded by some polynomial of k. Looking ahead, this property is used in showing the weak simulatability of the linear sketch scheme $\mathcal{S}_{\mathtt{CRT}}$. We do not directly show that $\mathtt{FAR}$ is negligible here, because it is indirectly implied by the $\mathtt{EUF\text{-}CMA}$ security of our proposed fuzzy signature scheme.

Distribution $\mathcal{X}$. The uniform distribution over a "discretized" version of $X = [0, 1)^n$. Specifically, let $\lambda \in \mathbb{N}$ be the natural number that denotes the representation length of a real number as introduced at the beginning of this section. We require that each coordinate x_i of a data $\mathbf{x} = (x_1, \ldots, x_n) \in X$ is distributed as $\{j \leftarrow_{\mathrm{R}} \mathbb{Z}_{2^\lambda} : \frac{j}{2^\lambda}\}$. Furthermore, we require λ to be sufficiently large (at least k/n).

Error distribution Φ *and Error parameter* ϵ. Φ can be any efficiently samplable (according to k) distribution over X such that $\mathtt{FRR} \le \epsilon$ for all $x \in X$.

6.2 Mathematical preliminaries

Group isomorphism based on Chinese remainder theorem. Let $n \in \mathbb{N}$. Let $w_1, \ldots, w_n \in \mathbb{N}$ be positive integers with the same bit length (i.e., $\lceil \log_2 w_1 \rceil = \cdots = \lceil \log_2 w_n \rceil$), such that

$$\forall i \in [n] \colon w_i \le \frac{1}{2t}, \quad \text{and} \quad \forall i \ne j \in [n] \colon \mathtt{GCD}(w_i, w_j) = 1, \tag{11}$$

and $W = \prod_{i \in [n]} w_i = \Theta(2^k)$, where k is defined as in Eq. (10). Note that Eqs. (10) and (11) imply that we have $w_i \le 2^{k/n}$ for all $i \in [n]$.

We assume that there exists a deterministic algorithm WGen that on input (t, n) outputs $\mathbf{w} = (w_1, \ldots, w_n)$ satisfying the above.

Fig. 6 The linear sketch scheme $\mathcal{S}_{\mathrm{CRT}} =$ (Setup, Sketch, DiffRec) for the fuzzy key setting $\mathcal{F}_1$ (left), and the auxiliary algorithms M_c for showing linearity and the simulator Sim for showing weak simulatability (right). In the figure, all addition are done in $\mathbb{R}^n_{\mathbf{w}}$, and $\ell' = \lambda - \lceil k/n \rceil$

```
$\mathsf{Setup}(\mathcal{F}_1, \Lambda = (\mathbb{Z}_W, +))$ :
    Return $pp \leftarrow \Lambda$.
$\mathsf{Sketch}(pp, s \in \mathbb{Z}_W, \mathbf{x} \in [0,1)^n)$ :
    $\mathbf{c} \leftarrow (\mathsf{CRT}^{-1}_{\mathbf{w}}(s) + \mathsf{E}_{\mathbf{w}}(\mathbf{x})) \bmod \mathbf{w}$
    Return $\mathbf{c}$.
$\mathsf{DiffRec}(pp, \mathbf{c}, \mathbf{c}')$ :
    $\Delta \mathbf{s} \leftarrow \mathsf{C}_{\mathbf{w}}(\mathbf{c}' - \mathbf{c})$
    $\Delta s \leftarrow \mathsf{CRT}_{\mathbf{w}}(\Delta \mathbf{s})$
    Return $\Delta s$.
```

```
$\mathsf{M}_\mathsf{c}(pp, \mathbf{c}, \Delta s, \mathbf{e})$ :
    $\mathbf{c}' \leftarrow (\mathbf{c} + \mathsf{CRT}^{-1}_{\mathbf{w}}(\Delta s) + \mathsf{E}_{\mathbf{w}}(\mathbf{e})) \bmod \mathbf{w}$
    Return $\mathbf{c}'$.
$\mathsf{Sim}(pp)$ :
    $\mathbf{c}_{in} \leftarrow_{\mathrm{R}} \mathbb{Z}^n_{\mathbf{w}}$
    $\mathbf{j} \leftarrow_{\mathrm{R}} (\mathbb{Z}_{2^{\ell'}})^n$
    $\mathbf{c}_{de} \leftarrow 2^{-\ell'} \cdot \mathbf{j}$
    $\mathbf{c} \leftarrow \mathbf{c}_{in} + \mathbf{c}_{de}$
    Return $\mathbf{c}$.
```

For vectors $\mathbf{v} = (v_1, \ldots, v_n) \in \mathbb{N}^n$ and $\mathbf{w} = (w_1, \ldots w_n) \in \mathbb{N}^n$, we define

$$\mathbf{v} \bmod \mathbf{w} := (v_1 \bmod w_1, \ldots, v_n \bmod w_n). \tag{12}$$

For vectors $\mathbf{v}_1, \mathbf{v}_2 \in \mathbb{N}^n$, we define the equivalence relation "$\sim$" by

$$\mathbf{v}_1 \sim \mathbf{v}_2 \quad \overset{\text{def}}{\iff} \quad \mathbf{v}_1 \bmod \mathbf{w} = \mathbf{v}_2 \bmod \mathbf{w},$$

and let $\mathbb{Z}^n_{\mathbf{w}} := \mathbb{Z}^n / \sim$ be the quotient set of $\mathbb{Z}^n$ by $\sim$. Note that $(\mathbb{Z}^n_{\mathbf{w}}, +)$ constitutes an abelian group, where the addition is modulo $\mathbf{w}$ as defined in Eq. (12).

Consider the following system of equations: given $\mathbf{v}, \mathbf{w} \in \mathbb{N}^n$, find V such that $V \bmod w_i = v_i$ $(i \in [n])$. According to the Chinese remainder theorem (CRT), the solution V is determined uniquely modulo W. Thus, for a fixed $\mathbf{w} \in \mathbb{N}^n$, we can define a mapping $\mathsf{CRT}_{\mathbf{w}} \colon \mathbb{Z}^n_{\mathbf{w}} \to \mathbb{Z}_W$ such that $\mathsf{CRT}_{\mathbf{w}}(\mathbf{v}) = V \in \mathbb{Z}_W$. Note that this mapping is a bijection, and we denote by $\mathsf{CRT}^{-1}_{\mathbf{w}}$ the "inverse" procedure of $\mathsf{CRT}_{\mathbf{w}}$.

Note that $\mathsf{CRT}_{\mathbf{w}}$ satisfies the following homomorphism: For all $\mathbf{v}_1, \mathbf{v}_2 \in \mathbb{Z}^n_{\mathbf{w}}$, it holds that

$$\mathsf{CRT}_{\mathbf{w}}(\mathbf{v}_1 + \mathbf{v}_2) = \mathsf{CRT}_{\mathbf{w}}(\mathbf{v}_1) + \mathsf{CRT}_{\mathbf{w}}(\mathbf{v}_2) \bmod W.$$

Since $\mathsf{CRT}_{\mathbf{w}}$ is bijective between $\mathbb{Z}^n_{\mathbf{w}}$ and $\mathbb{Z}_W$, $\mathsf{CRT}_{\mathbf{w}}$ is an isomorphism.

Coding and error correction. Let $\mathbf{w} = (w_1, \ldots, w_n) \in \mathbb{N}^n$ be the n-dimensional vector satisfying the requirements in Eq. (11). Similarly to $\mathbb{Z}^n_{\mathbf{w}}$, we define $\mathbb{R}^n_{\mathbf{w}} := \mathbb{R}^n / \sim$ be the quotient set of real vector space $\mathbb{R}^n$ by the equivalence relation $\sim$, where for a real number $y \in \mathbb{R}$, we define $r = y \bmod w_i$ by the number such that $\exists n \in \mathbb{Z} \colon y = nw_i + r$ and $0 \le r < w_i$.

Let $\mathsf{E}_{\mathbf{w}} \colon \mathbb{R}^n \to \mathbb{R}^n_{\mathbf{w}}$ be the following function:

$$\mathsf{E}_{\mathbf{w}}(\mathbf{x}) := (w_1 x_1, \ldots, w_n x_n) \in \mathbb{R}^n_{\mathbf{w}},$$

where $\mathbf{x} = (x_1, \ldots, x_n) \in \mathbb{R}^n$. Note that it holds that

$$\mathsf{E}_{\mathbf{w}}(\mathbf{x} + \mathbf{e}) = \mathsf{E}_{\mathbf{w}}(\mathbf{x}) + \mathsf{E}_{\mathbf{w}}(\mathbf{e}) \pmod{\mathbf{w}}. \tag{13}$$

Therefore, $\mathsf{E}_{\mathbf{w}}$ can be viewed as a kind of linear coding.

Let $\mathsf{C}_{\mathbf{w}} \colon \mathbb{R}^n_{\mathbf{w}} \to \mathbb{Z}^n_{\mathbf{w}}$ be the following function:

$$\mathsf{C}_{\mathbf{w}}\Big((y_1, \ldots, y_n) \Big) := \Big(\lfloor y_1 + 0.5 \rfloor, \ldots, \lfloor y_n + 0, 5 \rfloor \Big). \tag{14}$$

We note that the round-down operation $\lfloor y_i + 0.5 \rfloor$ in $\mathsf{C}_{\mathbf{w}}$ can be regarded as a kind of error correction. Specifically, by the conditions in Eq. (11), the following properties are satisfied: For any $\mathbf{x}, \mathbf{x}' \in X$, if $\|\mathbf{x} - \mathbf{x}'\|_\infty < t$, then we have

$$\Big\| \mathsf{E}_{\mathbf{w}}(\mathbf{x}) - \mathsf{E}_{\mathbf{w}}(\mathbf{x}') \Big\|_\infty < t \cdot \max_{i \in [n]} \{w_i\} \le 0.5.$$

Therefore, for such $\mathbf{x}, \mathbf{x}'$, it always holds that

$$\mathsf{C}_{\mathbf{w}}\Big(\mathsf{E}_{\mathbf{w}}(\mathbf{x}) - \mathsf{E}_{\mathbf{w}}(\mathbf{x}') \Big) = \mathbf{0}. \tag{15}$$

Additionally, for any $\mathbf{x} \in \mathbb{R}^n$ and $\mathbf{s} \in \mathbb{Z}^n_{\mathbf{w}}$, the following holds:

$$\mathsf{C}_{\mathbf{w}}(\mathbf{x} + \mathbf{s}) = \mathsf{C}_{\mathbf{w}}(\mathbf{x}) + \mathbf{s} \pmod{\mathbf{w}}. \tag{16}$$

6.3 Concrete linear sketch

Let $\mathcal{F}_1 = ((\mathsf{d}, X), t, \mathcal{X}, \Phi, \epsilon)$ be the fuzzy key setting defined in Sect. 6.1, and let $\mathbf{w} = (w_1, \ldots, w_n) = \mathsf{WGen}(t, n)$, where n is the dimension of X, and let $W = \prod_{i \in [n]} w_i$. Let $\mathsf{CRT}_{\mathbf{w}}$, $\mathsf{CRT}^{-1}_{\mathbf{w}}$, $\mathsf{E}_{\mathbf{w}}$, and $\mathsf{C}_{\mathbf{w}}$ be the functions defined in Sect. 6.2. Using these objects, we consider the linear sketch scheme $\mathcal{S}_{\mathrm{CRT}} = (\mathsf{Setup}, \mathsf{Sketch}, \mathsf{DiffRec})$ for $\mathcal{F}_1$ and the additive group $(\mathbb{Z}_W, +)$ $(=: \Lambda)$, as described in Fig. 6 (left). In the right of the figure, we also describe the auxiliary algorithm M_c that is used to show the linearity of $\mathcal{S}_{\mathrm{CRT}}$, and the simulator Sim that is used to show its weak simulatability.

The setup algorithm Setup in this linear sketch scheme actually does nothing, and the main algorithms Sketch and DiffRec as well as the auxiliary algorithm M_c are all deterministic. Furthermore, recall that we assume that the decimal part of each coordinate $w_i x_i$ in the computation of $\mathsf{E}_{\mathbf{w}}(\cdot)$ is rounded down so that its precision is the same as x_i. Concretely, since the significand of each x_i is expressed in λ bits and w_i is a ($\lceil k/n \rceil$)-bit natural number, the decimal part of each $w_i x_i$ is truncated to $\ell' := \lambda - \lceil k/n \rceil$ bits. Correspondingly, the simulator also picks an element in $\mathbb{R}^n_{\mathbf{w}}$, such that the integer part of each of its coordinates is sampled uniformly from $\mathbb{Z}_{w_i}$, and its decimal part is distributed uniformly in $\{\frac{j}{2^{\ell'}} | j \in \mathbb{Z}_{2^{\ell'}}\}$.

Remark on hypothetical recovering attacks and why they do not work. Let $s \in \mathbb{Z}_W$ and $\mathbf{s} = (s_1, \ldots, s_n) := \mathsf{CRT}^{-1}_{\mathbf{w}}(s) \in \mathbb{Z}_{\mathbf{w}}$. Let $c_i = s_i + w_i \cdot x_i \bmod w_i$ be the ith coordinate of a sketch $\mathbf{c}$ output from $\mathsf{Sketch}(pp, s, \mathbf{x})$, where $\mathbf{x} = (x_1, \ldots, x_n) \leftarrow_{\mathtt{R}} \mathcal{X}$, and thus each w_i is of the form $x_i = \frac{j}{2^\lambda}$ for some λ-bit integer j. Notice that in our linear sketch scheme $\mathcal{S}_{\mathtt{CRT}}$, if it were not for the rounding-down operation after multiplication of w_i and x_i, it holds that $2^\lambda \cdot c_i = 2^\lambda \cdot s_i + w_i \cdot j \bmod w_i = 2^\lambda \cdot s_i \bmod w_i$. Hence, if furthermore $\mathtt{GCD}(2^\lambda, w_i) = 1$, we can recover s_i from c_i by computing $s_i = (2^\lambda \cdot c_i) \cdot (2^\lambda)^{-1} \bmod w_i$, from which we can also recover x_i. (Yasuda et al. [39] pointed out recovering attacks of this kind.)

Similarly, notice that the "decimal" part $c^{(i)}_{\mathrm{de}}$ of c_i is dependent only on w_i and x_i. Hence, if it were not for the rounding-down operation after multiplication of w_i and x_i, $c^{(i)}_{\mathrm{de}}$ would be $w_i \cdot x_i \bmod 1 = \frac{w_i \cdot j}{2^\lambda} \bmod 1$. This would in turn imply $2^\lambda \cdot c^{(i)}_{\mathrm{de}} = w_i \cdot j \bmod 2^\lambda$. If furthermore $\mathtt{GCD}(2^\lambda, w_i) = 1$, then we can calculate $(2^\lambda \cdot c^{(i)}_{\mathrm{de}}) \cdot (w_i)^{-1} = j \bmod 2^\lambda$. Hence, j (and hence x_i) could be recovered from $c^{(i)}_{\mathrm{de}}$ as well.

However, such recovering attacks mentioned above do not apply to our proposed linear sketch scheme $\mathcal{S}_{\mathtt{CRT}}$ due to the rounding-down operation. As explained in the "*On the Treatment of Real Numbers*" paragraph, since each w_i is a k/n-bit integer, each $x'_i = w_i \cdot x_i$ results in $\lfloor \frac{w_i \cdot j}{2^{\lceil k/n \rceil}} \rfloor \cdot 2^{-(\lambda - \lceil k/n \rceil)}$. Thus, the ith coordinate c_i of $\mathbf{c}$, and its decimal part $c^{(i)}_{\mathrm{de}}$, are actually of the following forms:

$$c_i = s_i + \left\lfloor \frac{w_i \cdot j}{2^{\lceil k/n \rceil}} \right\rfloor \cdot 2^{-(\lambda - \lceil k/n \rceil)} \bmod w_i, \quad \text{and}$$

$$c^{(i)}_{\mathrm{de}} = \left\lfloor \frac{w_i \cdot j}{2^{\lceil k/n \rceil}} \right\rfloor \cdot 2^{-(\lambda - \lceil k/n \rceil)} \bmod 1,$$

for which the above-mentioned methods for calculating $x_i = \frac{j}{2^\lambda}$ from c_i (in case $\mathtt{GCD}(2^\lambda, w_i) = 1$) are not applicable. In fact, the weak simulatability of $\mathcal{S}_{\mathtt{CRT}}$ that we show in Lemma 7 below implies that if $\mathbf{x}$ is distributed as required in the fuzzy key setting $\mathcal{F}_1$ (specified in Sect. 6.1) and s is chosen uniformly, then recovering fuzzy data $\mathbf{x}$ or the input s from $\mathbf{c}$ is not possible (except for a negligible probability).

The following lemma guarantees that our construction $\mathcal{S}_{\mathtt{CRT}}$ satisfies all the requirements.

Lemma 7 *The linear sketch scheme $\mathcal{S}_{\mathtt{CRT}}$ in Fig. 6 (left) satisfies Definition 12.*

Proof of Lemma 7 We firstly show correctness, then linearity, and finally weak simulatability.

Correctness. The correctness of $\mathcal{S}_{\mathtt{CRT}}$ follows from the properties of the functions $\mathsf{CRT}_{\mathbf{w}}$, $\mathsf{E}_{\mathbf{w}}$, and $\mathsf{C}_{\mathbf{w}}$. Specifically, let $\mathbf{x}, \mathbf{x}' \in X$ be such that $\mathsf{d}(\mathbf{x}, \mathbf{x}') = \|\mathbf{x} - \mathbf{x}'\|_\infty < t$. Let pp be a public parameter output by Setup, let $s, \Delta s \in \mathbb{Z}_W$, and let $\mathbf{s} = \mathsf{CRT}^{-1}_{\mathbf{w}}(s)$ and $\Delta\mathbf{s} = \mathsf{CRT}^{-1}_{\mathbf{w}}(\Delta s)$. Furthermore, let $\mathbf{c} = \mathsf{Sketch}(pp, s, \mathbf{x}) = (\mathbf{s} + \mathsf{E}_{\mathbf{w}}(\mathbf{x})) \bmod \mathbf{w}$ and $\mathbf{c}' = \mathsf{Sketch}(pp, s + \Delta s, \mathbf{x}') = (\mathbf{s} + \Delta\mathbf{s} + \mathsf{E}_{\mathbf{w}}(\mathbf{x}')) \bmod \mathbf{w}$. Then, we have

$$\begin{aligned}
\mathsf{C}_{\mathbf{w}}(\mathbf{c}' - \mathbf{c}) &= \mathsf{C}_{\mathbf{w}}\Big(\mathbf{s} + \Delta\mathbf{s} + \mathsf{E}_{\mathbf{w}}(\mathbf{x}') - (\mathbf{s} + \mathsf{E}_{\mathbf{w}}(\mathbf{x}))\Big) \\
&\overset{(*)}{=} \Delta\mathbf{s} + \mathsf{C}_{\mathbf{w}}\Big(\mathsf{E}_{\mathbf{w}}(\mathbf{x}') - \mathsf{E}_{\mathbf{w}}(\mathbf{x})\Big) \\
&\overset{(\dagger)}{=} \Delta\mathbf{s},
\end{aligned}$$

where (*) is due to Eq. (16) (we omit to write " mod $\mathbf{w}$"), and (†) is due to Eq. (15) and $\|\mathbf{x} - \mathbf{x}'\|_\infty < t$. Thus,

$$\begin{aligned}
&\mathsf{DiffRec}(pp, \mathbf{c}, \mathbf{c}') \\
&\quad = \mathsf{DiffRec}\Big(pp, \mathsf{Sketch}(pp, s, \mathbf{x}), \mathsf{Sketch}(pp, s + \Delta s, \mathbf{x}')\Big) \\
&\quad = \mathsf{CRT}_{\mathbf{w}}\Big(\mathsf{C}_{\mathbf{w}}(\mathbf{c}' - \mathbf{c})\Big) \\
&\quad = \mathsf{CRT}_{\mathbf{w}}(\Delta\mathbf{s}) \\
&\quad = \Delta s,
\end{aligned}$$

which shows that the correctness condition [Eq. (5)] is satisfied.

Linearity. We consider the auxiliary algorithm M_{c} as described in Fig. 6 (right-top). To see that M_{c} satisfies the required property, let $\mathbf{x}, \mathbf{e} \in \mathbb{R}^n_{\mathbf{w}}$ and $s, \Delta s \in \mathbb{Z}_W$, and let $\mathbf{s} = \mathsf{CRT}^{-1}_{\mathbf{w}}(s)$ and $\Delta\mathbf{s} = \mathsf{CRT}^{-1}_{\mathbf{w}}(\Delta s)$. Then, note that $\mathsf{Sketch}(pp, s, \mathbf{x}) = (\mathbf{s} + \mathsf{E}_{\mathbf{w}}(\mathbf{x})) \bmod \mathbf{w}$ and $\mathsf{CRT}^{-1}_{\mathbf{w}}(s + \Delta s) = (\mathbf{s} + \Delta\mathbf{s}) \bmod \mathbf{w}$. Thus, it holds that

$$\begin{aligned}
&\mathsf{M}_{\mathsf{c}}\Big(pp, \mathsf{Sketch}(pp, s, \mathbf{x}), \Delta s, \mathbf{e}\Big) \\
&\quad = \Big(\mathbf{s} + \mathsf{E}_{\mathbf{w}}(\mathbf{x}) + \Delta\mathbf{s} + \mathsf{E}_{\mathbf{w}}(\mathbf{e})\Big) \bmod \mathbf{w} \\
&\quad \overset{(*)}{=} \Big(\mathbf{s} + \Delta\mathbf{s} + \mathsf{E}_{\mathbf{w}}(\mathbf{x} + \mathbf{e})\Big) \bmod \mathbf{w} \\
&\quad = \mathsf{Sketch}(pp, s + \Delta s, \mathbf{x} + \mathbf{e}),
\end{aligned}$$

where (*) is due to the linearity of $\mathsf{E}_{\mathbf{w}}$ [Eq. (13)]. This equation implies that the two distributions in Eq. (6) are identical, and hence the linearity is satisfied.

Weak simulatability. We consider the simulator Sim as described in Fig. 6 (right-bottom). Let $\mathcal{D}_{\mathrm{real}}$ and $\mathcal{D}_{\mathrm{sim}}$ be the distributions for the weak simulatability of $\mathcal{S}_{\mathtt{CRT}}$, which are defined as follows:

$$\mathcal{D}_{\text{real}} := \left\{ \begin{array}{l} \mathbf{x} \leftarrow_{\mathrm{R}} \mathcal{X};\ s \leftarrow_{\mathrm{R}} \mathbb{Z}_W; \\ \mathbf{c} \leftarrow \mathsf{CRT}_{\mathbf{w}}^{-1}(s) + \mathsf{E}_{\mathbf{w}}(\mathbf{x}) \end{array} : (s, \mathbf{c}) \right\}$$
$$= \left\{ \begin{array}{l} \mathbf{j} \leftarrow_{\mathrm{R}} (\mathbb{Z}_{2^\lambda})^n;\ \mathbf{x} \leftarrow 2^{-\lambda} \cdot \mathbf{j}; \\ s \leftarrow_{\mathrm{R}} \mathbb{Z}_W;\ \mathbf{c} \leftarrow \mathsf{CRT}_{\mathbf{w}}^{-1}(s) + \mathsf{E}_{\mathbf{w}}(\mathbf{x}) \end{array} : (s, \mathbf{c}) \right\},$$
$$\mathcal{D}_{\text{sim}} := \left\{ s \leftarrow_{\mathrm{R}} \mathbb{Z}_W;\ \mathbf{c} \leftarrow_{\mathrm{R}} \mathsf{Sim}(pp) : (s, \mathbf{c}) \right\}$$
$$= \left\{ \begin{array}{l} s \leftarrow_{\mathrm{R}} \mathbb{Z}_W;\ \mathbf{c}_{\text{in}} \leftarrow_{\mathrm{R}} \mathbb{Z}_{\mathbf{w}}^n;\ \mathbf{j} \leftarrow_{\mathrm{R}} (\mathbb{Z}_{2^{\ell'}})^n; \\ \mathbf{c}_{\text{de}} \leftarrow 2^{-\ell'} \cdot \mathbf{j};\ \mathbf{c} \leftarrow \mathbf{c}_{\text{in}} + \mathbf{c}_{\text{de}} \end{array} : (s, \mathbf{c}) \right\},$$

where $pp = \Lambda = (\mathbb{Z}_W, +)$ and $\ell' = \lambda - \lceil k/n \rceil$. We will show that for any (even computationally unbounded) algorithm $\mathcal{A}$, the following inequality holds:

$$\Pr[\mathcal{A}(\mathcal{D}_{\text{real}}) = 1] \le 2^n \cdot \Pr[\mathcal{A}(\mathcal{D}_{\text{sim}}) = 1]. \tag{17}$$

Recall that we are requiring that $n = O(\log_2 k)$, equivalently 2^n is smaller than some polynomial of k, and hence Eq. (17) implies weak simulatability.

Instead of directly showing Eq. (17) for any algorithm $\mathcal{A}$, we first slightly simplify the setting. Specifically, consider the following two distributions $\mathcal{D}'_{\text{real}}$ and $\mathcal{D}'_{\text{sim}}$:

$$\mathcal{D}'_{\text{real}} := \left\{ \mathbf{j} \leftarrow_{\mathrm{R}} (\mathbb{Z}_{2^\lambda})^n;\ \mathbf{x} \leftarrow_{\mathrm{R}} 2^{-\lambda} \cdot \mathbf{j};\ \mathbf{x}' \leftarrow \mathsf{E}_{\mathbf{w}}(\mathbf{x}) : \mathbf{x}' \right\}$$
$$\mathcal{D}'_{\text{sim}} := \left\{ \begin{array}{l} \mathbf{x}'_{\text{in}} \leftarrow_{\mathrm{R}} \mathbb{Z}_{\mathbf{w}}^n;\ \mathbf{j} \leftarrow_{\mathrm{R}} (\mathbb{Z}_{2^{\ell'}})^n; \\ \mathbf{x}'_{\text{de}} \leftarrow 2^{-\ell'} \cdot \mathbf{j};\ \mathbf{x}' \leftarrow \mathbf{x}'_{\text{in}} + \mathbf{x}'_{\text{de}} \end{array} : \mathbf{x}' \right\}.$$

We now show that for any algorithm $\mathcal{A}$ considered for weak simulatability, there exists a corresponding algorithm $\mathcal{B}$ (with almost the same running time as $\mathcal{A}$) such that $\Pr[\mathcal{A}(\mathcal{D}_{\text{real}}) = 1] = \Pr[\mathcal{B}(\mathcal{D}'_{\text{real}}) = 1]$ and $\Pr[\mathcal{A}(\mathcal{D}_{\text{sim}}) = 1] = \Pr[\mathcal{B}(\mathcal{D}'_{\text{sim}}) = 1]$. Specifically, $\mathcal{B}$ takes $\mathbf{x}' \in \mathbb{R}_{\mathbf{w}}^n$ as input, picks $s \in \mathbb{Z}_W$ uniformly at random, sets $\mathbf{c} \leftarrow \mathsf{CRT}_{\mathbf{w}}^{-1}(s) + \mathbf{x}'$, and outputs $\mathcal{A}(s, \mathbf{c})$. If $\mathbf{x}'$ that is input to $\mathcal{B}$ is sampled from $\mathcal{D}'_{\text{real}}$, then the pair $(s, \mathbf{c})$ that $\mathcal{B}$ inputs to $\mathcal{A}$ is distributed identically to $\mathcal{D}_{\text{real}}$, while if $\mathbf{x}'$ is sampled from $\mathcal{D}'_{\text{sim}}$, then $(s, \mathbf{c})$ is distributed identically to $\mathcal{D}_{\text{sim}}$. (In particular, the "integer part" of $\mathbf{c}$ is uniformly distributed over $\mathbb{Z}_{\mathbf{w}}^n$, even if $\mathsf{CRT}_{\mathbf{w}}^{-1}(s)$ is added.) Clearly, this $\mathcal{B}$ satisfies $\Pr[\mathcal{B}(\mathcal{D}'_{\text{real}}) = 1] = \Pr[\mathcal{A}(\mathcal{D}_{\text{real}}) = 1]$ and $\Pr[\mathcal{B}(\mathcal{D}'_{\text{sim}}) = 1] = \Pr[\mathcal{A}(\mathcal{D}_{\text{sim}}) = 1]$.

Hence, in order to show Eq. (17) for any algorithm $\mathcal{A}$, it is sufficient to show the following inequality for any algorithm $\mathcal{B}$:

$$\Pr[\mathcal{B}(\mathcal{D}'_{\text{real}}) = 1] \le 2^n \cdot \Pr[\mathcal{B}(\mathcal{D}'_{\text{sim}}) = 1]. \tag{18}$$

Furthermore, notice that $\mathcal{D}'_{\text{sim}}$ is nothing but the uniform distribution over the set $\mathbb{Z}_{\mathbf{w}}^n \times \{\frac{j}{2^{\ell'}} | j \in \mathbb{Z}_{2^{\ell'}}\}^n$, whose size is $\prod_{i \in [n]}(w_i \cdot 2^{\ell'})$. Hence, by applying Lemma 2, we obtain

$$\Pr[\mathcal{B}(\mathcal{D}'_{\text{real}}) = 1] \le \prod_{i \in [n]} (w_i \cdot 2^{\ell'}) \cdot 2^{-\mathbf{H}_\infty(\mathcal{D}'_{\text{real}})} \cdot \Pr[\mathcal{B}(\mathcal{D}'_{\text{sim}}) = 1]. \tag{19}$$

To complete the proof, we will show

$$2^{-\mathbf{H}_\infty(\mathcal{D}'_{\text{real}})} \le \prod_{i \in [n]} \Big(\frac{1}{w_i \cdot 2^{\ell'}} + \frac{1}{2^\lambda} \Big). \tag{20}$$

Before showing the above, note that Eq. (20) implies that $\prod_{i \in [n]}(w_i \cdot 2^{\ell'}) \cdot 2^{-\mathbf{H}_\infty(\mathcal{D}'_{\text{real}})}$ [appearing in the right hand side of Eq. (19)] is upperbounded as follows:

$$\prod_{i \in [n]} (w_i \cdot 2^{\ell'}) \cdot \prod_{i \in [n]} (\frac{1}{w_i \cdot 2^{\ell'}} + \frac{1}{2^\lambda}) \le \prod_{i \in [n]} (1 + 2^{\lceil k/n \rceil + \ell' - \lambda}) = 2^n,$$

where the inequality uses $w_i \le 2^{\lceil k/n \rceil}$, and the equality uses $\ell' = \lambda - \lceil k/n \rceil$. Thus, if indeed we can show Eq. (20), then by combining it with Eq. (19), we can obtain Eq. (18).

Hence, it remains to show Eq. (20). For each $i \in [n]$, let $\mathcal{D}'^{(i)}_{\text{real}}$ be the distribution of the ith coordinate in $\mathcal{D}'_{\text{real}}$. Recall that each w_i is a k/n-bit integer, each $x_i \in [0, 1)$ is of the form $\frac{j}{2^\lambda}$ where $j \leftarrow_{\mathrm{R}} \mathbb{Z}_{2^\lambda}$, and x'_i is a multiplication of w_i and x_i. Recall also that $\ell' = \lambda - k/n$. Hence, $\mathcal{D}'^{(i)}_{\text{real}}$ is distributed as follows [see also Eq. (9)]:

$$\mathcal{D}'^{(i)}_{\text{real}} = \left\{ j \leftarrow_{\mathrm{R}} \mathbb{Z}_{2^\lambda};\ x_i \leftarrow 2^{-\lambda} \cdot j \colon \lfloor w_i \cdot x_i \cdot 2^{\ell'} \rfloor \cdot 2^{-\ell'} \right\}$$
$$= \left\{ j \leftarrow_{\mathrm{R}} \mathbb{Z}_{2^\lambda} \colon \lfloor w_i \cdot j \cdot 2^{\ell' - \lambda} \rfloor \cdot 2^{-\ell'} \right\}.$$

We can thus calculate $2^{-\mathbf{H}_\infty(\mathcal{D}'_{\text{real}})}$ as follows:

$$\begin{aligned} &2^{-\mathbf{H}_\infty(\mathcal{D}'_{\text{real}})} \\ &\quad = \prod_{i \in [n]} 2^{-\mathbf{H}_\infty(\mathcal{D}'^{(i)}_{\text{real}})} = \prod_{i \in [n]} \Big(\max_{z \in \mathbb{R}_{w_i}} \Pr_{x'_i \leftarrow_{\mathrm{R}} \mathcal{D}'^{(i)}_{\text{real}}} [\, x'_i = z \,] \Big) \\ &\quad = \prod_{i \in [n]} \Big(\max_{z \in \mathbb{R}_{w_i}} \Pr_{j \leftarrow_{\mathrm{R}} \mathbb{Z}_{2^\lambda}} \Big[\lfloor w_i \cdot j \cdot 2^{\ell' - \lambda} \rfloor \cdot 2^{-\ell'} = z \Big] \Big) \\ &\quad = \prod_{i \in [n]} \Big(\max_{z \in \mathbb{R}_{w_i}} \Pr_{j \leftarrow_{\mathrm{R}} \mathbb{Z}_{2^\lambda}} \Big[z \cdot 2^{\ell'} \le w_i \cdot j \cdot 2^{\ell' - \lambda} < z \cdot 2^{\ell'} + 1 \Big] \Big) \\ &\quad = \prod_{i \in [n]} \Big(\max_{z \in \mathbb{R}_{w_i}} \Pr_{j \leftarrow_{\mathrm{R}} \mathbb{Z}_{2^\lambda}} \Big[\frac{z \cdot 2^\lambda}{w_i} \le j < \frac{z \cdot 2^\lambda}{w_i} + \frac{2^\lambda}{w_i \cdot 2^{\ell'}} \Big] \Big). \end{aligned} \tag{21}$$

Now, for each $z \in \mathbb{R}_{w_i}$, let a_z be the number of integers that belong to the interval $[\frac{z \cdot 2^\lambda}{w_i}, \frac{(z \cdot 2^{\ell'} + 1) \cdot 2^\lambda}{w_i \cdot 2^{\ell'}})$. By definition, the probability appearing in Eq. (21) is $\frac{a_z}{2^\lambda}$. Furthermore, the number of integers that belong to an interval $[l, r)$ is at most $r - l + 1$, and thus we have $a_z \le \frac{2^\lambda}{w_i \cdot 2^{\ell'}} + 1$. (Note that the right hand side is independent of z.) Using this, we can upperbound $2^{-\mathbf{H}_\infty(\mathcal{D}'_{\text{real}})}$ as follows:

$$2^{-\mathbf{H}_\infty(\mathcal{D}'_{\text{real}})} = \prod_{i\in[n]} \left(\max_{z\in\mathbb{R}_{w_i}} \frac{a_z}{2^\lambda} \right) \le \prod_{i\in[n]} \left(\frac{1}{w_i \cdot 2^{\ell'}} + \frac{1}{2^\lambda} \right),$$

which is exactly Eq. (20), as required. This completes the proof that $\mathcal{S}_{\mathtt{CRT}}$ satisfies weak simulatability, and the entire proof of Lemma 7. □

6.4 Modified Waters signature scheme

Here, we show a variant of the Waters signature scheme [36], which we call the *modified Waters signature* (MWS) scheme $\Sigma_{\mathtt{MWS}}$. We then show that $\Sigma_{\mathtt{MWS}}$ satisfies $\mathtt{EUF}\text{-}\mathtt{CMA}$ security and the homomorphic property (Definition 9), which in turn implies that it is $\Phi^{\text{add}}\text{-}\mathtt{RKA}^*$ secure (due to Lemma 5).

Specific bilinear group generator $\mathsf{BGGen}_{\mathtt{MWS}}$. In the MWS scheme, we use a (slightly) non-standard way for specifying bilinear groups, namely the order p of (symmetric) bilinear groups is generated based on an integer $W = \prod_{i\in[n]} w_i$, where $\mathbf{w} = (w_1, \ldots, w_n) \in \mathbb{N}^n$ satisfies the conditions in Eq. (11), so that p is the smallest prime satisfying $W|p-1$. More concretely, we consider the following algorithm PGen for choosing the order p based on W:

> $\mathsf{PGen}(W)$: on input $W \in \mathbb{N}$, for $i = 1, 2, \ldots$ check if $p = iW + 1$ is a prime and return p if this is the case. Otherwise, increment $i \leftarrow i + 1$ and go to the next iteration.

According to the prime number theorem, the density of primes among the natural numbers that are less than N is roughly $1/\ln N$, and thus for i's that are exponentially smaller than W, the probability that $iW+1$ is a prime can be roughly estimated as $1/\ln W$. Therefore, by using the above algorithm PGen, one can find a prime p satisfying $W|p-1$ by performing the primality testing for $O(\ln W) = O(k)$ times on average (recall that $W = \Theta(2^k)$). Furthermore, if $\mathsf{PGen}(W)$ outputs p, then it is guaranteed that $p/W = O(k)$. (This fact is used for security.)

Let $\mathsf{BGGen}_{\mathtt{MWS}}$ denote an algorithm that, given 1^k, runs $\mathbf{w} \leftarrow \mathsf{WGen}(t, n)$ where t and n are the parameters from the fuzzy data setting $\mathcal{F}$ corresponding the security parameter k, computes $W \leftarrow \prod_{i\in[n]} w_i$, $p \leftarrow \mathsf{PGen}(W)$, and outputs a description of bilinear groups $\mathcal{BG} = (p, \mathbb{G}, \mathbb{G}_T, g, e)$, where $\mathbb{G}$ and $\mathbb{G}_T$ are cyclic groups with order p and $e\colon \mathbb{G}\times\mathbb{G} \to \mathbb{G}_T$ is a bilinear map.

Construction. Using $\mathsf{BGGen}_{\mathtt{MWS}}$ and the algorithms in the original Waters signature scheme $\Sigma_{\mathtt{Wat}} = (\mathsf{Setup}_{\mathtt{Wat}}, \mathsf{KG}_{\mathtt{Wat}}, \mathsf{Sign}_{\mathtt{Wat}}, \mathsf{Ver}_{\mathtt{Wat}})$ in Fig. 3 (left), the MWS scheme $\Sigma_{\mathtt{MWS}} = (\mathsf{Setup}_{\mathtt{MWS}}, \mathsf{KG}_{\mathtt{MWS}}, \mathsf{Sign}_{\mathtt{MWS}}, \mathsf{Ver}_{\mathtt{MWS}})$ is constructed as in Fig. 7 (left). Note that the component $pp_{\mathtt{Wat}}$ in a public parameter pp (generated by $\mathsf{Setup}_{\mathtt{MWS}}$) is distributed identically to that generated in the original Waters scheme $\Sigma_{\mathtt{Wat}}$ in which the bilinear group generator $\mathsf{BGGen}_{\mathtt{MWS}}$ is used. Therefore, $\Sigma_{\mathtt{MWS}}$ can be viewed as the original Waters scheme $\Sigma_{\mathtt{Wat}}$, except that

1. we specify how to generate the parameter of bilinear groups by $\mathsf{BGGen}_{\mathtt{MWS}}$, and
2. we use a secret key sk' (for the Waters scheme) of the form $sk' = z^{sk} \bmod p$, thereby we change the signing key space from $\mathbb{Z}_p$ to $\mathbb{Z}_W$.

Because of these changes, it is immediate to see that the MWS scheme inherits the perfect correctness of the Waters signature scheme.

In the following, we show that $\Sigma_{\mathtt{MWS}}$ satisfies $\mathtt{EUF}\text{-}\mathtt{CMA}$ security (based on the CDH assumption with respect to $\mathsf{BGGen}_{\mathtt{MWS}}$) and the homomorphic property (Definition 9). These properties, combined with Lemma 5, imply that $\Sigma_{\mathtt{MWS}}$ satisfies $\Phi^{\text{add}}\text{-}\mathtt{RKA}^*$ security, and thus satisfies the assumption required in Theorem 2. (One might suspect the plausibility of the CDH assumption with respect to $\mathsf{BGGen}_{\mathtt{MWS}}$ due to our specific choice of the order p. We discuss it in "Appendix G.")

Lemma 8 *If the CDH assumption holds with respect to* $\mathsf{BGGen}_{\mathtt{MWS}}$*, then the MWS scheme* $\Sigma_{\mathtt{MWS}}$ *is* $\mathtt{EUF}\text{-}\mathtt{CMA}$ *secure.*

Let $pp = (pp_{\mathtt{Wat}}, z)$ be a public parameter output by $\mathsf{Setup}_{\mathtt{MWS}}$, let $D^{(1)}_{pp} = \{sk \leftarrow_{\mathrm{R}} \mathbb{Z}_W; sk' \leftarrow z^{sk} \bmod p\colon sk'\}$ and $D^{(2)}_{pp} = \{sk' \leftarrow_{\mathrm{R}} \mathbb{Z}_p\colon sk'\}$. Note that the support of $D^{(1)}_{pp}$ is a strict subset of that of $D^{(2)}_{pp}$.

Now, let $\mathcal{A}$ be any PPTA adversary attacking the $\mathtt{EUF}\text{-}\mathtt{CMA}$ security of the MWS scheme $\Sigma_{\mathtt{MWS}}$. Let Expt_1 be the original $\mathtt{EUF}\text{-}\mathtt{CMA}$ experiment, i.e., $\mathsf{Expt}^{\mathtt{EUF}\text{-}\mathtt{CMA}}_{\Sigma_{\mathtt{MWS}},\mathcal{A}}(k)$, and let Expt_2 be the experiment that is defined in the same manner as Expt_1, except that sk' is sampled according to the distribution $D^{(2)}_{pp}$. For both $i \in \{1, 2\}$, let Adv_i be the advantage of $\mathcal{A}$ (i.e., the probability of $\mathcal{A}$ outputting a successful forgery) in Expt_i. Then, by Lemma 4, we have $\mathsf{Adv}_1 \le (p/W) \cdot \mathsf{Adv}_2 = O(k) \cdot \mathsf{Adv}_2$. Furthermore, it is straightforward to see that succeeding in forging in Expt_2 is as difficult as succeeding in breaking the $\mathtt{EUF}\text{-}\mathtt{CMA}$ security of the original Waters scheme $\Sigma_{\mathtt{Wat}}$ (in which the bilinear group generator $\mathsf{BGGen}_{\mathtt{MWS}}$ is used), and thus Adv_2 is negligible if $\Sigma_{\mathtt{Wat}}$ is $\mathtt{EUF}\text{-}\mathtt{CMA}$ secure.

Finally, due to Waters [36], if the CDH assumption holds with respect to $\mathsf{BGGen}_{\mathtt{MWS}}$, then the Waters scheme $\Sigma_{\mathtt{Wat}}$ (in which $\mathsf{BGGen}_{\mathtt{MWS}}$ is used,) is $\mathtt{EUF}\text{-}\mathtt{CMA}$ secure. Hence, Adv_2 is negligible. Combining all the explanations above proves the lemma. □

Lemma 9 *The MWS scheme* $\Sigma_{\mathtt{MWS}}$ *is homomorphic* (*as per Definition* 9).

```
$\mathsf{Setup}_{\mathtt{MWS}}(1^k)$:
    $\mathcal{BG} = (p, \mathbb{G}, \mathbb{G}_T, g, e) \leftarrow_{\mathrm{R}} \mathsf{BGGen}_{\mathtt{MWS}}(1^k)$
    $h, u', u_1, \dots, u_\ell \leftarrow_{\mathrm{R}} \mathbb{G}$
    $pp_{\mathtt{Wat}} \leftarrow (\mathcal{BG}, h, u', (u_i)_{i \in [\ell]})$
    Let $z \in \mathbb{Z}_p^*$ be an element of order $W$.
    Return $pp \leftarrow (pp_{\mathtt{Wat}}, z)$.
$\mathsf{KG}_{\mathtt{MWS}}(pp)$:
    $sk \leftarrow_{\mathrm{R}} \mathbb{Z}_W$
    $vk \leftarrow g^{z^{sk}}$
    Return $(vk, sk)$.
$\mathsf{Sign}_{\mathtt{MWS}}(pp, sk, m)$:
    $sk' \leftarrow z^{sk} \bmod p$
    Return $\mathsf{Sign}_{\mathtt{Wat}}(pp_{\mathtt{Wat}}, sk', m)$.
$\mathsf{Ver}_{\mathtt{MWS}}(pp, vk, m, \sigma)$:
    Return $\mathsf{Ver}_{\mathtt{Wat}}(pp_{\mathtt{Wat}}, vk, m, \sigma)$.
```

```
$\mathsf{KG}'(pp, sk)$:
    $vk \leftarrow g^{z^{sk}}$
    Return $vk$.
$\mathsf{M}_{\mathsf{vk}}(pp, vk, \Delta sk)$:
    $vk' \leftarrow (vk)^{z^{\Delta sk}}$
    Return $vk'$.
$\mathsf{M}_{\mathsf{sig}}(pp, vk, m, \sigma, \Delta sk)$:
    $\sigma_1' \leftarrow \sigma_1^{z^{\Delta sk}}$
    $\sigma_2' \leftarrow \sigma_2^{z^{\Delta sk}}$
    Return $\sigma' \leftarrow (\sigma_1', \sigma_2')$.
```

Fig. 7 The modified Waters signature (MWS) scheme $\Sigma_{\mathtt{MWS}}$ (left), and the auxiliary algorithms (KG', M_{vk}, $\mathsf{M}_{\mathsf{sig}}$) for showing the homomorphic property (right). Note that the signing algorithm $\mathsf{Sign}_{\mathtt{MWS}}$ (resp. the verification algorithm $\mathsf{Ver}_{\mathtt{MWS}}$) of the MWS scheme $\Sigma_{\mathtt{MWS}}$ uses the signing algorithm $\mathsf{Sign}_{\mathtt{Wat}}$ (resp. the verification algorithm $\mathsf{Ver}_{\mathtt{Wat}}$) of the original Waters scheme $\Sigma_{\mathtt{Wat}}$ [described in Fig. 3 (left)] as a subroutine

Proof of Lemma 9 Consider the algorithms (KG', M_{vk}, $\mathsf{M}_{\mathsf{sig}}$) that are described in Fig. 7 (right). KG' is the algorithm for showing that this scheme has a simple key generation process. That is, using this algorithm, $\mathsf{KG}_{\mathtt{MWS}}$ can be rewritten with the process in Eq. (1). The secret key space is $\mathbb{Z}_W$, and $(\mathbb{Z}_W, +)$ constitutes an abelian group, as required.

Next, it should be easy to see that M_{vk} satisfies the requirement in Eq. (2). Indeed, let $pp = (pp_{\mathtt{Wat}}, z)$ be a public parameter, and let $sk, \Delta sk \in \mathbb{Z}_W$. Then, it holds that

$$\mathsf{M}_{\mathsf{vk}}(pp, \mathsf{KG}'(pp, sk), \Delta sk) = (g^{z^{sk}})^{z^{\Delta sk}} = g^{z^{sk + \Delta sk}}$$
$$= \mathsf{KG}'(pp, sk + \Delta sk),$$

which is exactly Eq. (2).

Finally, we observe that $\mathsf{M}_{\mathsf{sig}}$ satisfies the requirements in Eq. (3). Let $pp = (pp_{\mathtt{Wat}}, z)$ and $sk, \Delta sk \in \mathbb{Z}_W$ as above, and $m = (m_1 \| \dots \| m_\ell) \in \{0,1\}^\ell$ be a message to be signed. Let (σ_1, σ_2) be a signature on the message m that is generated by $\mathsf{Sign}_{\mathtt{MWS}}(pp, sk, m; r)$, where $r \in \mathbb{Z}_p$ is a randomness. By definition, σ_1 and σ_2 are of the form $\sigma_1 = h^{z^{sk}} \cdot (u' \cdot \prod_{i \in [\ell]} u_i^{m_i})^r$ and $\sigma_2 = g^r$, respectively. Thus, if $\sigma' = (\sigma_1', \sigma_2')$ is output by $\mathsf{M}_{\mathsf{sig}}(pp, vk, m, \sigma, \Delta sk)$, then it holds that

$$\sigma_1' = \sigma_1^{z^{\Delta sk}} = h^{z^{sk + \Delta sk}} \cdot \left(u' \cdot \prod_{i \in [\ell]} u_i^{m_i} \right)^{r \cdot z^{\Delta sk}},$$
$$\sigma_2' = \sigma_2^{z^{\Delta sk}} = g^{r \cdot z^{\Delta sk}}.$$

This implies $\sigma' = (\sigma_1', \sigma_2') = \mathsf{Sign}_{\mathtt{MWS}}(pp, sk + \Delta sk, m; r \cdot z^{\Delta sk})$. Note that for any $\Delta sk \in \mathbb{Z}_W$, if $r \leftarrow_{\mathrm{R}} \mathbb{Z}_p$, then $((r \cdot z^{\Delta sk}) \bmod p)$ is uniformly distributed in $\mathbb{Z}_p$. This implies that the distributions considered in Eq. (3) are identical. Furthermore, by the property of the MWS scheme (which is inherited from the original Waters scheme [36]), any signature $\sigma' = (\sigma_1', \sigma_2')$ satisfying $\mathsf{Ver}_{\mathtt{MWS}}(pp, vk, m, \sigma') = \top$ must satisfy the property that there exists $r' \in \mathbb{Z}_p$ such that $\mathsf{Sign}_{\mathtt{MWS}}(pp, sk, m; r') = \sigma'$. Putting everything together implies that for any $sk, \Delta sk \in \mathbb{Z}_W$, any message $m \in \{0,1\}^\ell$, and any signature σ such that $\mathsf{Ver}_{\mathtt{MWS}}(pp, vk, m, \sigma) = \top$, if $vk = \mathsf{KG}'(pp, sk)$, $vk' = \mathsf{M}_{\mathsf{vk}}(pp, vk, \Delta sk)$ and $\sigma' = \mathsf{M}_{\mathsf{sig}}(pp, vk, m, \sigma, \Delta sk)$, then it holds that $\mathsf{Ver}_{\mathtt{MWS}}(pp, vk', m, \sigma') = \top$. Therefore, the requirement regarding Eq. (4) is satisfied as well. This completes the proof of Lemma 9. □

The combination of Lemmas 5, 8, and 9 shows that $\Sigma_{\mathtt{MWS}}$ satisfies Φ^{add}-$\mathtt{RKA}^*$ security.

Corollary 1 *If the CDH assumption holds with respect to* $\mathsf{BGGen}_{\mathtt{MWS}}$*, then the MWS scheme* $\Sigma_{\mathtt{MWS}}$ *is* Φ^{add}-$\mathtt{RKA}^*$ *secure.*

6.5 Full description

Here, we give the full description of our first instantiation of a fuzzy signature scheme, by instantiating the underlying linear sketch and signature schemes in the generic construction, with the concrete linear sketch scheme $\mathcal{S}_{\mathtt{CRT}}$ (given in Sect. 6.3) and the MWS scheme $\Sigma_{\mathtt{MWS}}$ (given in Sect. 6.4), respectively.

Let $\ell = \ell(k)$ be a positive polynomial that denotes the length of messages. Let $\mathcal{F}_1 = ((\mathsf{d}, X), t, \mathcal{X}, \Phi, \epsilon)$ be the fuzzy key setting defined in Sect. 6.1, where t (and n) are determined according to the security parameter k. let $\mathbf{w} = (w_1, \dots, w_n) = \mathsf{WGen}(t, n)$, where n is the dimension of X, and let $W = \prod_{i \in [n]} w_i$. Let $\mathsf{CRT}_{\mathbf{w}}$, $\mathsf{CRT}_{\mathbf{w}}^{-1}$, $\mathsf{E}_{\mathbf{w}}$, and $\mathsf{C}_{\mathbf{w}}$ be the functions defined in Sect. 6.2. Let $\mathsf{BGGen}_{\mathtt{MWS}}$ be the bilinear group generator defined in Sect. 6.4. Then, using these ingredients, our first proposed fuzzy signature scheme

```
$\mathsf{Setup}_{\mathsf{FS1}}(\mathcal{F}_1, 1^k)$ :
  Let $\mathcal{BG} := (p, \mathbb{G}, \mathbb{G}_T, g, e) \leftarrow \mathsf{BGGen}_{\mathtt{MWS}}(1^k)$
  $h, u', u_1, \dots, u_\ell \leftarrow_{\mathtt{R}} \mathbb{G}$
  Let $z$ be an element of $\mathbb{Z}_p^*$ of order $W$.
  Let $\Lambda := (\mathbb{Z}_W, +)$.
  Return $pp \leftarrow (\mathcal{BG}, h, u', (u_i)_{i \in [\ell]}, z, \Lambda)$.

$\mathsf{KG}_{\mathsf{FS1}}(pp, \mathbf{x})$ :
  $sk \leftarrow_{\mathtt{R}} \mathbb{Z}_W$
  $vk \leftarrow g^{z^{sk}}$
  $\mathbf{c} \leftarrow (\mathsf{CRT}_{\mathbf{w}}^{-1}(sk) + \mathsf{E}_{\mathbf{w}}(\mathbf{x})) \bmod \mathbf{w}$
  $\boxed{\mathbf{c} \leftarrow \mathtt{Round}_\ell(\mathbf{c})}^{(\dagger)}$
  Return $VK \leftarrow (vk, \mathbf{c})$.

$\mathsf{Sign}_{\mathsf{FS1}}(pp, \mathbf{x}', m)$ :
  Parse $m$ as $(m_1 \| \dots \| m_\ell) \in \{0,1\}^\ell$.
  $\widetilde{sk} \leftarrow_{\mathtt{R}} \mathbb{Z}_W$
  $\widetilde{vk} \leftarrow g^{z^{\widetilde{sk}}}$
  $r \leftarrow_{\mathtt{R}} \mathbb{Z}_p$
  $\widetilde{\sigma}_1 \leftarrow h^{z^{\widetilde{sk}}} \cdot (u' \cdot \prod_{i \in [\ell]} u_i^{m_i})^r$
  $\widetilde{\sigma}_2 \leftarrow g^r$
  $\widetilde{\mathbf{c}} \leftarrow (\mathsf{CRT}_{\mathbf{w}}^{-1}(\widetilde{sk}) + \mathsf{E}_{\mathbf{w}}(\mathbf{x}')) \bmod \mathbf{w}$
  $\boxed{\widetilde{\mathbf{c}} \leftarrow \mathtt{Round}_\ell(\widetilde{\mathbf{c}})}^{(\dagger)}$
  Return $\sigma \leftarrow (\widetilde{vk}, \widetilde{\sigma}_1, \widetilde{\sigma}_2, \widetilde{\mathbf{c}})$.

$\mathsf{Ver}_{\mathsf{FS1}}(pp, VK, m, \sigma)$ :
  $(vk, \mathbf{c}) \leftarrow VK$
  $(\widetilde{vk}, \widetilde{\sigma}_1, \widetilde{\sigma}_2, \widetilde{\mathbf{c}}) \leftarrow \sigma$
  Parse $m$ as $(m_1 \| \dots \| m_\ell) \in \{0,1\}^\ell$.
  If $e(\widetilde{\sigma}_2, u' \cdot \prod_{i \in [\ell]} u_i^{m_i}) \cdot e(\widetilde{vk}, h)$
          $\neq e(\widetilde{\sigma}_1, g)$ then return $\bot$.
  $\Delta\mathbf{s} \leftarrow \mathsf{C}_{\mathbf{w}}(\widetilde{\mathbf{c}} - \mathbf{c})$
  $\Delta sk \leftarrow \mathsf{CRT}_{\mathbf{w}}(\Delta\mathbf{s})$
  If $(vk)^{z^{\Delta sk}} = \widetilde{vk}$ then return $\top$
          else return $\bot$.
```

Fig. 8 Our first instantiation of a fuzzy signature scheme Σ_{FS1}. $^{(\dagger)}$ The steps involving "$\mathtt{Round}_\ell$" enclosed by a box in $\mathsf{KG}_{\mathsf{FS1}}$ and $\mathsf{Sign}_{\mathsf{FS1}}$ are those at which we perform the "rounding" operation of the decimal part, which we will explain in Sect. 8. (The reader who has not read there is expected to ignore them)

$\Sigma_{\mathsf{FS1}} = (\mathsf{Setup}_{\mathsf{FS1}}, \mathsf{KG}_{\mathsf{FS1}}, \mathsf{Sign}_{\mathsf{FS1}}, \mathsf{Ver}_{\mathsf{FS1}})$ for the fuzzy key setting $\mathcal{F}_1$ is constructed as in Fig. 8.[15]

The following theorem guarantees the correctness and security of our scheme Σ_{FS1}, which is obtained as a corollary of the combination of Theorems 1 and 2, Lemma 7, and Corollary 1.

Theorem 3 *The fuzzy signature scheme Σ_{FS1} for the fuzzy key setting $\mathcal{F}_1$ in Fig. 8 is ϵ-correct. Furthermore, if the CDH assumption holds with respect to $\mathsf{BGGen}_{\mathtt{MWS}}$, then Σ_{FS1} is* EUF-CMA *secure.*

7 Second instantiation

In this section, we propose our second instantiation of a fuzzy signature scheme, based on the Schnorr signature scheme. The strong requirement for our first instantiation proposed in Sect. 6 is that the fuzzy data is assumed to be distributed uniformly. This strong requirement is relaxed in our second instantiation.

The rest of this section is organized as follows. In Sect. 7.1, we specify a concrete fuzzy key setting $\mathcal{F}_2$ for which our second instantiation is constructed. Next, in Sect. 7.2, we show the concrete linear sketch scheme $\mathcal{S}_{\mathtt{Hash}}$ for $\mathcal{F}_2$. Combining this linear sketch scheme $\mathcal{S}_{\mathtt{Hash}}$ and the Schnorr signature scheme $\Sigma_{\mathtt{Sch}}$ (Fig. 3 (right)), we obtain our second instantiation of a fuzzy signature scheme Σ_{FS2}. The description of this fuzzy signature scheme Σ_{FS2} is given in Sect. 7.3.

In this section, we treat real numbers in the same way as in Sect. 6.

7.1 Specific fuzzy key setting

Here, we specify a concrete fuzzy key setting $\mathcal{F}_2 = ((\mathsf{d}, X), t, \mathcal{X}, \Phi, \epsilon)$ for which our linear sketch scheme $\mathcal{S}_{\mathtt{Hash}}$ and our Schnorr-based fuzzy signature scheme Σ_{FS2} are constructed.

Metric space (d, X). The space X is defined by $X := [0, 1)^n \subset \mathbb{R}^n$, where $n \in \mathbb{N}$ is a parameter specified by the context (e.g., an object from which we measure fuzzy data) and a security parameter k. The distance function $\mathsf{d}: X \times X \rightarrow \mathbb{R}$ is the L_∞-distance. Namely, for $\mathbf{x} = (x_1, \dots, x_n) \in X$ and $\mathbf{x}' = (x'_1, \dots, x'_n) \in X$, we define $\mathsf{d}(\mathbf{x}, \mathbf{x}') := \|\mathbf{x} - \mathbf{x}'\|_\infty := \max_{i \in [n]} |x_i - x'_i|$. Note that X forms an abelian group with respect to coordinate-wise addition (modulo 1).

Threshold t. For a security parameter k, we require the threshold $t \in \mathbb{R}$ to satisfy

$$k \le \lfloor -n \log_2(2t) \rfloor. \tag{22}$$

For notational convenience, let $T := 1/(2t)$.

Distribution $\mathcal{X}$. An efficiently samplable distribution over a "discretized" version of $X = [0, 1)^n$. That is, letting $\lambda \in \mathbb{N}$ denote the length of the significand of a real number, if $\mathbf{x} = (x_1, \dots, x_n)$ is sampled from $\mathcal{X}$, then each x_i is of the form $\frac{m}{2^\lambda}$, where m is a λ-bit integer. (See the "*On the Treatment of Real Numbers*" paragraph at the beginning of Sect. 6.) We require $T \le 2^\lambda$.

Furthermore, we require that $\mathcal{X}$ satisfy the assumption on the average min-entropy that we state later.

Error distribution Φ and *Error parameter* ϵ. Φ can be any efficiently samplable (according to k) distribution over X such that $\mathtt{FRR} \le \epsilon$ for all $x \in X$.

Here, before going into the actual requirement on the distribution $\mathcal{X}$, we quickly highlight the difference between the fuzzy key setting $\mathcal{F}_2$ and $\mathcal{F}_1$ (where the latter is the one

[15] In Fig. 8, the operations involving "$\mathtt{Round}_\ell$" enclosed by a box in $\mathsf{KG}_{\mathsf{FS1}}$ and $\mathsf{Sign}_{\mathsf{FS1}}$ are those for concerning practical treatment of real numbers explained in Sect. 8. The reader who has not read there is expected to ignore them.

for which we constructed our first concrete fuzzy signature scheme in Sect. 6): the only difference between $\mathcal{F}_2$ and $\mathcal{F}_1$, other than $\mathcal{X}$, is in the threshold t. Here, we need a more strict threshold for t, so that we can use the leftover hash lemma, as we will see in the proof of Lemma 10.

The requirement on the distribution of fuzzy data $\mathcal{X}$. Let $\mathcal{X}'$ be the "scaled-up" version of $\mathcal{X}$, namely $\mathcal{X}'$ is the distribution obtained by multiplying the value $T = 1/(2t)$ to the outcome of the distribution $\mathcal{X}$, where the rounding-down operation is performed for each coordinate of $\mathcal{X}'$ as explained at the "*On the Treatment of Real Numbers*" paragraph in the beginning of Sect. 6. Since $\mathcal{X}$ is a distribution over $[0, 1)^n$, $\mathcal{X}'$ is a distribution over $[0, T)^n$. Now, let us divide $\mathcal{X}'$ into the "integer" part $\mathcal{X}'_{\mathrm{in}}$ and the "decimal" part $\mathcal{X}'_{\mathrm{de}}$. Namely, let $\mathbf{x}' = (x'_1, \ldots, x'_n)$ be a vector produced from $\mathcal{X}'$. Then, $\mathcal{X}'_{\mathrm{in}}$ is the distribution of the n-dimensional vector whose ith element is the integer part of x'_i. Similarly, $\mathcal{X}'_{\mathrm{de}}$ is the distribution of the n-dimensional vector whose ith element is the decimal part of x'_i. Note that each coordinate of the integer part $\mathcal{X}'_{\mathrm{in}}$ is represented by $\lceil \log_2 T \rceil$ bits, and thus each coordinate of the decimal part $\mathcal{X}'_{\mathrm{de}}$ will have $(\lambda - \lceil \log_2 T \rceil)$-bit precision, so that the significand of the entire x'_i is expressed in λ bits. Note also that the joint distribution $(\mathcal{X}'_{\mathrm{in}}, \mathcal{X}'_{\mathrm{de}})$ contains the same information as $\mathcal{X}'$ (and hence as $\mathcal{X}$).

The requirement we impose on the distribution $\mathcal{X}$ is that we have

$$\widetilde{\mathbf{H}}_\infty(\mathcal{X}'_{\mathrm{in}} | \mathcal{X}'_{\mathrm{de}}) \geq \log_2 p + \omega(\log_2 k),$$

where p is the order of the field over which we consider the universal hash family $\mathcal{H}_{\mathrm{lin}}$. We note that $\widetilde{\mathbf{H}}_\infty(\mathcal{X}'_{\mathrm{in}} | \mathcal{X}'_{\mathrm{de}}) = \widetilde{\mathbf{H}}_\infty(\mathcal{X}' | \mathcal{X}'_{\mathrm{de}})$. Looking ahead, p will also be the order of the group over which the Schnorr scheme is constructed, and thus we typically set $|p| = \lceil \log_2 p \rceil = \Theta(k)$.

We would like to emphasize that our requirement on the distribution $\mathcal{X}$ in $\mathcal{F}_2$ is arguably much more natural and relaxed than requiring that $\mathcal{X}$ is the uniform distribution over (the discretized version of) X (as is required of $\mathcal{F}_1$). Specifically, in order for the above requirement for $\mathcal{X}$ to be satisfied, it is necessary that $\mathcal{X}'_{\mathrm{de}}$ does not leak much about $\mathcal{X}'_{\mathrm{in}}$. Intuitively, when fuzzy data $\mathbf{x}$ is sampled from an object according to some distribution, the upper part of (in the representation of the significand of) $\mathbf{x}$ should be dominant for identifying the object. On the other hand, the lower part of $\mathbf{x}$ should be dominated by noise caused at the measurement of $\mathbf{x}$. Since we are adopting the universal error model in which the measurement error captured by the error distribution Φ is independent of individual objects producing fuzzy data, the lower part of $\mathbf{x}$ contains information that is less dependent on the original object. In our requirement for the fuzzy data distribution $\mathcal{X}$, the distribution of the upper (resp. lower) part of fuzzy data corresponds to $\mathcal{X}'_{\mathrm{in}}$ (resp. $\mathcal{X}'_{\mathrm{de}}$), and thus requiring that $\mathcal{X}'_{\mathrm{de}}$ does not leak much information about $\mathcal{X}'_{\mathrm{in}}$, is arguably a natural requirement.

7.2 Concrete linear sketch

Let $\mathcal{F}_2 = ((\mathsf{d}, X), t, \mathcal{X}, \Phi, \epsilon)$ be the fuzzy key setting as defined above. Let $\mathbb{F}_p$ be a finite field with prime order p satisfying $p \geq T = 1/(2t)$. Here, we identify $\mathbb{F}_p$ with $\mathbb{Z}_p$, and thus we freely interpret an element in the former set as an element in the latter set, and vice versa. Let $\mathcal{H}_{\mathrm{lin}} = \{h_z : (\mathbb{F}_p)^n \to \mathbb{F}_p\}_{z \in \mathbb{F}_{p^n}}$ be the universal hash function family with linearity, which is described in Sect. 2.3. For each $z \in \mathbb{F}_{p^n}$ and $s \in \mathbb{F}_p$, we define "$h_z^{-1}(s)$" as the set of preimages of s under h_z. That is, $h_z^{-1}(s) := \{\mathbf{a} \in (\mathbb{F}_p)^n | h_z(\mathbf{a}) = s\}$. Hence, the notation "$\mathbf{a} \leftarrow_{\mathrm{R}} h_z^{-1}(s)$" means that we choose a vector $\mathbf{a}$ uniformly from the set $h_z^{-1}(s)$ (which can be performed efficiently in terms of $\log_2(p^n)$). Furthermore, recall that $T = 1/(2t)$.

Then, using these ingredients, our linear sketch scheme $\mathcal{S}_{\mathtt{Hash}} = (\mathsf{Setup}, \mathsf{Sketch}, \mathsf{DiffRec})$ for $\mathcal{F}_2$ and the additive group $(\mathbb{Z}_p, +)$ $(=: \Lambda)$ is constructed as described in Fig. 9 (left), where for convenience, we also give the description of the auxiliary algorithm $\mathsf{M_c}$ used for showing its linearity and that of the simulator Sim for showing its weak simulatability (right).

We remind the reader that we are treating real numbers as explained in the "*On the Treatment of Real Numbers*" paragraph at the beginning of Sect. 6. We remark that as in our first linear sketch scheme $\mathcal{S}_{\mathtt{CRT}}$ proposed in Sect. 6.3, if the rounding-down operation were not performed after multiplication $T \cdot \mathbf{x}$ in the computation of $\mathsf{Sketch}(pp, s, \mathbf{x})$, then a hypothetical recovering attack (that recovers $\mathbf{x}$ and s from a sketch $\mathbf{c}$) could work [39]. However, due to our treatment of real numbers, if $\mathbf{x}$ is distributed as required in the fuzzy key setting $\mathcal{F}_2$ (specified in Sect. 7.1), then recovering $\mathbf{x}$ or s is not possible.

The following lemma guarantees that our construction $\mathcal{S}_{\mathtt{Hash}}$ satisfies all the requirements.

Lemma 10 *The linear sketch scheme* $\mathcal{S}_{\mathtt{Hash}}$ *in Fig.* 9 (*left*) *satisfies Definition* 12.

Proof of Lemma 10 Roughly speaking, the correctness follows from the linearity of the universal hash family $\mathcal{H}_{\mathrm{lin}}$ and a simple algebra; the linearity property of $\mathcal{S}$ follows from the linearity of $\mathcal{H}_{\mathrm{lin}}$; The weak simulatability follows from the leftover hash lemma together with the requirement on the average min-entropy satisfied by the distribution $\mathcal{X}$ of fuzzy data in the fuzzy key setting $\mathcal{F}_2$ specified in Sect. 7.1.

Below, we first show correctness, then linearity, and finally weak simulatability.[16]

[16] In fact, the construction shown here satisfies average-case indistinguishability that we defined in [19]. See "Appendix C" for its definition.

Fig. 9 The linear sketch scheme $\mathcal{S}_{\mathtt{Hash}} =$ (Setup, Sketch, DiffRec) for the fuzzy key setting $\mathcal{F}_2$ (left), and the auxiliary algorithm M_c for showing linearity and the simulator Sim for showing weak simulatability (right). $^{(\dagger)}$ The operation "+" (resp. "−") in $(\mathbb{R}_p)^n$ are the coordinate-wise addition (resp. subtraction) in $\mathbb{R}_p$

```
Setup($\mathcal{F}_2, \Lambda = (\mathbb{Z}_p, +)$) :
    $z \leftarrow_{\mathrm{R}} \mathbb{F}_{p^n}$
    $pp \leftarrow (\Lambda, z)$
    Return $pp$.
Sketch($pp, s, \mathbf{x}$) : (where $s \in \mathbb{Z}_p$ and $\mathbf{x} \in [0,1)^n$)
    $\mathbf{a} \leftarrow_{\mathrm{R}} h_z^{-1}(s)$
    $\mathbf{c} \leftarrow \mathbf{a} + T \cdot \mathbf{x}$ $^{(\dagger)}$
    Return $\mathbf{c}$.
DiffRec($pp, \mathbf{c}, \mathbf{c}'$) :
    $\Delta\mathbf{c} \leftarrow \mathbf{c}' - \mathbf{c}$ $^{(\dagger)}$
    $\Delta s \leftarrow h_z(\lfloor \Delta\mathbf{c} \rceil)$
    Return $\Delta s$.
```

```
M_c($pp, \mathbf{c}, \Delta s, \mathbf{e}$) :
    $\Delta\mathbf{a} \leftarrow_{\mathrm{R}} h_z^{-1}(\Delta s)$
    $\mathbf{c}' \leftarrow (\mathbf{c} + \Delta\mathbf{a} + T \cdot \mathbf{e})$ $^{(\dagger)}$
    Return $\mathbf{c}'$.
Sim($pp$) :
    $\mathbf{x} \leftarrow_{\mathrm{R}} \mathcal{X}$
    $s' \leftarrow_{\mathrm{R}} \mathbb{Z}_p$
    $\mathbf{c} \leftarrow$ Sketch($pp, s', \mathbf{x}$)
    Return $\mathbf{c}$.
```

Correctness. Fix $pp = (\Lambda = (\mathbb{Z}_p, +), z)$, $\mathbf{x}, \mathbf{x}' \in X$ such that $\mathsf{d}(\mathbf{x}, \mathbf{x}) = \|\mathbf{x} - \mathbf{x}'\|_\infty < t$, and $s, \Delta s \in \mathbb{F}_p$. Recall that $T = 1/(2t)$. Note that $\|\mathbf{x} - \mathbf{x}'\|_\infty < t$ implies $\|T \cdot (\mathbf{x} - \mathbf{x}')\|_\infty < 1/2$, and hence $\lfloor T \cdot (\mathbf{x} - \mathbf{x}')\rceil = \mathbf{0}$. Now, suppose $\mathbf{c}$ and $\mathbf{c}'$ are output by $\mathsf{Sketch}(pp, s, x)$ and $\mathsf{Sketch}(pp, s + \Delta s, x')$, respectively. Then, by the definition of Sketch, it holds that $\mathbf{c} = \mathbf{a} + T \cdot \mathbf{x}$ for some $\mathbf{a} \in h_z^{-1}(s)$ and $\mathbf{c}' = \mathbf{a}' + T \cdot \mathbf{x}'$ for some $\mathbf{a}' \in h_z^{-1}(s + \Delta s)$. Therefore,

$$\begin{aligned}
\mathsf{DiffRec}(pp, \mathbf{c}, \mathbf{c}') &= h_z(\lfloor \mathbf{c}' - \mathbf{c} \rceil) \\
&= h_z(\lfloor (\mathbf{a}' + T \cdot \mathbf{x}') - (\mathbf{a} + T \cdot \mathbf{x}) \rceil) \\
&= h_z(\mathbf{a}' - \mathbf{a} + \lfloor T \cdot (\mathbf{x}' - \mathbf{x}) \rceil) \\
&\overset{(*)}{=} h_z(\mathbf{a}' - \mathbf{a}) \\
&\overset{(**)}{=} h_z(\mathbf{a}') - h_z(\mathbf{a}) \\
&= (s + \Delta s) - s = \Delta s,
\end{aligned}$$

where the equality (*) is due to $\lfloor T \cdot (\mathbf{x} - \mathbf{x}')\rceil = \mathbf{0}$, and the equality (**) is due to the linearity of $\mathcal{H}_{\mathrm{lin}}$. This shows that Eq. (5) is satisfied, and thus $\mathcal{S}_{\mathtt{Hash}}$ satisfies correctness.

Linearity. We use the auxiliary algorithm M_c in Fig. 9 (right-top). Fix $pp = (\Lambda = (\mathbb{Z}_p, +), z)$, $\mathbf{x}, \mathbf{e} \in X$, and $s, \Delta s \in \mathbb{F}_p$. For showing linearity, it is sufficient to show that the following distributions $\mathcal{D}_1$ and $\mathcal{D}_2$ are equivalent:

$$\begin{aligned}
\mathcal{D}_1 &:= \left\{ \begin{array}{l} \mathbf{c} \leftarrow_{\mathrm{R}} \mathsf{Sketch}(pp, s, \mathbf{x}); \\ \mathbf{c}' \leftarrow_{\mathrm{R}} \mathsf{Sketch}(pp, s + \Delta s, \mathbf{x} + \mathbf{e}) \end{array} : (\mathbf{c}, \mathbf{c}') \right\} \\
&= \left\{ \begin{array}{l} \mathbf{a} \leftarrow_{\mathrm{R}} h_z^{-1}(s);\ \mathbf{c} \leftarrow \mathbf{a} + T \cdot \mathbf{x}; \\ \mathbf{a}' \leftarrow_{\mathrm{R}} h_z^{-1}(s + \Delta s); \\ \mathbf{c}' \leftarrow \mathbf{a}' + T \cdot (\mathbf{x} + \mathbf{e}) \end{array} : (\mathbf{c}, \mathbf{c}') \right\}, \\
\mathcal{D}_2 &:= \left\{ \begin{array}{l} \mathbf{c} \leftarrow_{\mathrm{R}} \mathsf{Sketch}(pp, s, \mathbf{x}); \\ \mathbf{c}' \leftarrow_{\mathrm{R}} \mathsf{M}_\mathsf{c}(pp, c, \Delta s, \mathbf{e}) \end{array} : (\mathbf{c}, \mathbf{c}') \right\} \\
&= \left\{ \begin{array}{l} \mathbf{a} \leftarrow_{\mathrm{R}} h_z^{-1}(s);\ \mathbf{c} \leftarrow \mathbf{a} + T \cdot \mathbf{x}; \\ \Delta\mathbf{a} \leftarrow_{\mathrm{R}} h_z^{-1}(\Delta s);\ \mathbf{c}' \leftarrow \mathbf{c} + \Delta\mathbf{a} + T \cdot \mathbf{e} \end{array} : (\mathbf{c}, \mathbf{c}') \right\} \\
&= \left\{ \begin{array}{l} \mathbf{a} \leftarrow_{\mathrm{R}} h_z^{-1}(s);\ \mathbf{c} \leftarrow \mathbf{a} + T \cdot \mathbf{x}; \\ \Delta\mathbf{a} \leftarrow_{\mathrm{R}} h_z^{-1}(\Delta s); \\ \mathbf{c}' \leftarrow \mathbf{a} + \Delta\mathbf{a} + T \cdot (\mathbf{x} + \mathbf{e}) \end{array} : (\mathbf{c}, \mathbf{c}') \right\}.
\end{aligned}$$

To this end, focusing on the difference between the above $\mathcal{D}_1$ and $\mathcal{D}_2$, and also on how $\mathbf{c}'$ is generated, it is sufficient to show that the following two distributions $\mathcal{D}_1'$ and $\mathcal{D}_2'$ are equivalent:

$$\begin{aligned}
\mathcal{D}_1' &:= \left\{ \mathbf{a} \leftarrow_{\mathrm{R}} h_z^{-1}(s);\ \mathbf{a}' \leftarrow_{\mathrm{R}} h_z^{-1}(s + \Delta s) : (\mathbf{a}, \mathbf{a}') \right\}, \\
\mathcal{D}_2' &:= \left\{ \begin{array}{l} \mathbf{a} \leftarrow_{\mathrm{R}} h_z^{-1}(s);\ \Delta\mathbf{a} \leftarrow_{\mathrm{R}} h_z^{-1}(\Delta s); \\ \mathbf{a}' \leftarrow \mathbf{a} + \Delta\mathbf{a} \end{array} : (\mathbf{a}, \mathbf{a}') \right\}.
\end{aligned}$$

Here, $\mathcal{D}_1'$ is the uniform distribution over the direct product $(h_z^{-1}(s)) \times (h_z^{-1}(s + \Delta s))$. We show that $\mathcal{D}_2'$ is also the uniform distribution over the same set. Indeed, by the linearity of $\mathcal{H}_{\mathrm{lin}}$, for any $s', s'' \in \mathbb{F}_p$, the set $h_z^{-1}(s')$ and the set $h_z^{-1}(s'')$ have the same size, and the second element $\mathbf{a}'$ produced from D_2' belongs to the set $h_z^{-1}(s + \Delta s)$. This means that for each fixed element $\widetilde{\mathbf{a}} \in h_z^{-1}(s)$, the distribution $\mathcal{D}' = \{\Delta\mathbf{a} \leftarrow_{\mathrm{R}} h_z^{-1}(\Delta s) : \widetilde{\mathbf{a}} + \Delta\mathbf{a}\}$ yields the uniform distribution over $h_z^{-1}(s + \Delta s)$. This in turn means that $\mathcal{D}_2'$ is the uniform distribution over the direct product $(h_z^{-1}(s)) \times (h_z^{-1}(s + \Delta s))$. Hence, we can conclude that the original distributions $\mathcal{D}_1$ and $\mathcal{D}_2$ are equivalent, and thus $\mathcal{S}_{\mathtt{Hash}}$ satisfies linearity.

Weak simulatability. We use the simulator Sim in Fig. 9 (right-bottom). We will show that the statistical distance between the following two distributions $\mathcal{D}_{\mathrm{real}}$ and $\mathcal{D}_{\mathrm{sim}}$ is negligibly small:

$$\begin{aligned}
\mathcal{D}_{\mathrm{real}} &:= \left\{ \begin{array}{l} pp \leftarrow_{\mathrm{R}} \mathsf{Setup}(\mathcal{F}_2, \Lambda);\ \mathbf{x} \leftarrow_{\mathrm{R}} \mathcal{X}; \\ s \leftarrow_{\mathrm{R}} \mathbb{F}_p;\ \mathbf{c} \leftarrow_{\mathrm{R}} \mathsf{Sketch}(pp, s, \mathbf{x}) \end{array} : (pp, s, \mathbf{c}) \right\} \\
&= \left\{ \begin{array}{l} z \leftarrow_{\mathrm{R}} Z;\ \mathbf{x} \leftarrow_{\mathrm{R}} \mathcal{X};\ s \leftarrow_{\mathrm{R}} \mathbb{F}_p; \\ \mathbf{a} \leftarrow_{\mathrm{R}} h_z^{-1}(s);\ \mathbf{c} \leftarrow \mathbf{a} + T \cdot \mathbf{x} \end{array} : (z, s, \mathbf{c}) \right\}, \\
\mathcal{D}_{\mathrm{sim}} &:= \left\{ \begin{array}{l} pp \leftarrow_{\mathrm{R}} \mathsf{Setup}(\mathcal{F}_2, \Lambda);\ s \leftarrow_{\mathrm{R}} \mathbb{F}_p; \\ \mathbf{c} \leftarrow_{\mathrm{R}} \mathsf{Sim}(pp) \end{array} : (pp, s, \mathbf{c}) \right\} \\
&= \left\{ \begin{array}{l} z \leftarrow_{\mathrm{R}} Z;\ \mathbf{x} \leftarrow_{\mathrm{R}} \mathcal{X};\ s, s' \leftarrow_{\mathrm{R}} \mathbb{F}_p; \\ \mathbf{a} \leftarrow_{\mathrm{R}} h_z^{-1}(s');\ \mathbf{c} \leftarrow \mathbf{a} + T \cdot \mathbf{x} \end{array} : (z, s, \mathbf{c}) \right\},
\end{aligned}$$

where $Z = \mathbb{F}_{p^n}$ is the seed space of $\mathcal{H}_{\mathrm{lin}}$. Note that this implies weak simulatability, because for all (even computationally unbounded) algorithms $\mathcal{A}$, it holds that $\Pr[\mathcal{A}(\mathcal{D}_{\mathrm{real}}) = 1] \le \Pr[\mathcal{A}(\mathcal{D}_{\mathrm{sim}}) = 1] + \mathbf{SD}(\mathcal{D}_{\mathrm{real}}, \mathcal{D}_{\mathrm{sim}})$,[17] and thus shows that $\mathcal{S}_{\mathtt{Hash}}$ satisfies weak simulatability.

Firstly, note that for every $z \in Z$, the distribution $\{s \leftarrow_{\mathrm{R}} \mathbb{F}_p;\ \mathbf{a} \leftarrow_{\mathrm{R}} h_z^{-1}(s) \colon (s, \mathbf{a})\}$ and the distribution $\{\mathbf{a} \leftarrow_{\mathrm{R}} (\mathbb{F}_p)^n;\ s \leftarrow h_z(\mathbf{a}) \colon (s, \mathbf{a})\}$ are equivalent. Hence, the above distributions $\mathcal{D}_{\mathrm{real}}$ and $\mathcal{D}_{\mathrm{sim}}$ are, respectively, equivalent to the following distributions $\mathcal{D}'_{\mathrm{real}}$ and $\mathcal{D}'_{\mathrm{sim}}$:

$$\mathcal{D}'_{\mathrm{real}} := \left\{ \begin{array}{l} z \leftarrow_{\mathrm{R}} Z;\ \mathbf{x} \leftarrow_{\mathrm{R}} \mathcal{X};\ \mathbf{a} \leftarrow_{\mathrm{R}} (\mathbb{F}_p)^n; \\ \mathbf{c} \leftarrow \mathbf{a} + T \cdot \mathbf{x} \end{array} : (z, h_z(\mathbf{a}), \mathbf{c}) \right\},$$

$$\mathcal{D}'_{\mathrm{sim}} := \left\{ \begin{array}{l} z \leftarrow_{\mathrm{R}} Z;\ \mathbf{x} \leftarrow_{\mathrm{R}} \mathcal{X};\ \mathbf{a} \leftarrow_{\mathrm{R}} (\mathbb{F}_p)^n; \\ \mathbf{c} \leftarrow \mathbf{a} + T \cdot \mathbf{x};\ s \leftarrow_{\mathrm{R}} \mathbb{F}_p \end{array} : (z, s, \mathbf{c}) \right\}.$$

Clearly we have $\mathbf{SD}(\mathcal{D}_{\mathrm{real}}, \mathcal{D}_{\mathrm{sim}}) = \mathbf{SD}(\mathcal{D}'_{\mathrm{real}}, \mathcal{D}'_{\mathrm{sim}})$.

Now, we define the joint distribution (A, C) as follows:

$$(A, C) := \left\{ \mathbf{x} \leftarrow_{\mathrm{R}} \mathcal{X};\ \mathbf{a} \leftarrow_{\mathrm{R}} (\mathbb{F}_p)^n;\ \mathbf{c} \leftarrow \mathbf{a} + T \cdot \mathbf{x} \colon (\mathbf{a}, \mathbf{c}) \right\}.$$

We can think of this joint distribution as the one specifying the "input" $\mathbf{a}$ for a hash function h_z and "leakage" $\mathbf{c}$ (about the input $\mathbf{a}$). Hence, if we can show that $\widetilde{\mathbf{H}}_\infty(A|C)$ is "sufficiently large," then we can apply the leftover hash lemma (Lemma 3) to upperbound $\mathbf{SD}(\mathcal{D}'_{\mathrm{real}}, \mathcal{D}'_{\mathrm{sim}}) = \mathbf{SD}(\mathcal{D}_{\mathrm{real}}, \mathcal{D}_{\mathrm{sim}})$ to be "small," leading to the desired conclusion that $\mathcal{S}_{\mathtt{Hash}}$ satisfies weak simulatability. To this end, in the following we show that $\widetilde{\mathbf{H}}_\infty(A|C) = \widetilde{\mathbf{H}}_\infty(\mathcal{X}'_{\mathrm{in}}|\mathcal{X}'_{\mathrm{de}})$ holds, where $\mathcal{X}'_{\mathrm{in}}$ and $\mathcal{X}'_{\mathrm{de}}$ are, respectively, the "integer" part and the "decimal" part of the "scaled-up" version $\mathcal{X}'$ of the original distribution $\mathcal{X}$ of fuzzy data that we introduced in Sect. 7.1.

Note that the distribution $\mathcal{X}'_{\mathrm{in}}$ (resp. $\mathcal{X}'_{\mathrm{de}}$) is over $(\mathbb{F}_p)^n$ (resp. $[0, 1)^n$). Furthermore, by definition, all the information regarding $\mathcal{X}'$ can be expressed as the joint distribution $(\mathcal{X}'_{\mathrm{in}}, \mathcal{X}'_{\mathrm{de}})$. Using the distributions $\mathcal{X}'_{\mathrm{in}}$ and $\mathcal{X}'_{\mathrm{de}}$, and dividing the "integer" part and "decimal" part of C into C_{in} and C_{de} in the same manner as $\mathcal{X}'_{\mathrm{in}}$ and $\mathcal{X}'_{\mathrm{de}}$, we can equivalently rewrite the joint distribution (A, C) as the joint distribution $(A, C_{\mathrm{in}}, C_{\mathrm{de}})$ in the following way:

$$(A, C_{\mathrm{in}}, C_{\mathrm{de}}) := \left\{ \begin{array}{l} (\mathbf{x}'_{\mathrm{in}}, \mathbf{x}'_{\mathrm{de}}) \leftarrow_{\mathrm{R}} (\mathcal{X}'_{\mathrm{in}}, \mathcal{X}'_{\mathrm{de}}); \\ \mathbf{a} \leftarrow_{\mathrm{R}} (\mathbb{F}_p)^n; \\ \mathbf{c}_{\mathrm{in}} \leftarrow \mathbf{a} + \mathbf{x}'_{\mathrm{in}};\ \mathbf{c}_{\mathrm{de}} \leftarrow \mathbf{x}'_{\mathrm{de}} \end{array} : (\mathbf{a}, \mathbf{c}_{\mathrm{in}}, \mathbf{c}_{\mathrm{de}}) \right\}.$$

By focusing on the relation among $\mathbf{x}'_{\mathrm{in}}$, $\mathbf{c}_{\mathrm{in}}$, and $\mathbf{a}$, we can further equivalently rewrite the joint distribution $(A, C_{\mathrm{in}}, C_{\mathrm{de}})$ as follows:

$$(A, C_{\mathrm{in}}, C_{\mathrm{de}}) = \left\{ \begin{array}{l} \mathbf{x}'_{\mathrm{de}} \leftarrow_{\mathrm{R}} \mathcal{X}'_{\mathrm{de}}; \\ \mathbf{x}'_{\mathrm{in}} \leftarrow_{\mathrm{R}} (\mathcal{X}'_{\mathrm{in}}|\mathcal{X}'_{\mathrm{de}} = \mathbf{x}'_{\mathrm{de}}); \\ \mathbf{c}_{\mathrm{in}} \leftarrow_{\mathrm{R}} (\mathbb{F}_p)^n; \\ \mathbf{a} \leftarrow \mathbf{c}_{\mathrm{in}} - \mathbf{x}'_{\mathrm{in}};\ \mathbf{c}_{\mathrm{de}} \leftarrow \mathbf{x}'_{\mathrm{de}} \end{array} : (\mathbf{a}, \mathbf{c}_{\mathrm{in}}, \mathbf{c}_{\mathrm{de}}) \right\},$$

where $(\mathcal{X}'_{\mathrm{in}}|\mathcal{X}'_{\mathrm{de}} = \mathbf{x}'_{\mathrm{de}})$ denotes the distribution $\mathcal{X}'_{\mathrm{in}}$ conditioned on $\mathcal{X}'_{\mathrm{de}} = \mathbf{x}'_{\mathrm{de}}$. Note that guessing $\mathbf{a} = \mathbf{c}_{\mathrm{in}} - \mathbf{x}'_{\mathrm{in}}$ given $(\mathbf{c}_{\mathrm{in}}, \mathbf{c} = \mathbf{x}'_{\mathrm{de}})$, is equivalent to guessing $\mathbf{x}'_{\mathrm{in}}$ given $\mathbf{x}'_{\mathrm{de}}$. Hence, we have $\widetilde{\mathbf{H}}_\infty(A|C_{\mathrm{in}}, C_{\mathrm{out}}) = \widetilde{\mathbf{H}}_\infty(\mathcal{X}'_{\mathrm{in}}|\mathcal{X}'_{\mathrm{de}})$. Furthermore, since $\widetilde{\mathbf{H}}_\infty(A|C) = \widetilde{\mathbf{H}}_\infty(A|C_{\mathrm{in}}, C_{\mathrm{de}})$ holds by definition, we can conclude that $\widetilde{\mathbf{H}}_\infty(A|C) = \widetilde{\mathbf{H}}_\infty(\mathcal{X}'_{\mathrm{in}}|\mathcal{X}'_{\mathrm{de}})$.

Recall that we are requiring

$$\widetilde{\mathbf{H}}_\infty(\mathcal{X}'_{\mathrm{in}}|\mathcal{X}'_{\mathrm{de}}) \ge \log_2 p + \omega(\log_2 k).$$

Thus, by the leftover hash lemma (Lemma 3), we have

$$\begin{aligned} \mathbf{SD}(\mathcal{D}_{\mathrm{real}}, \mathcal{D}_{\mathrm{sim}}) &= \mathbf{SD}(\mathcal{D}'_{\mathrm{real}}, \mathcal{D}'_{\mathrm{sim}}) \\ &\le \frac{1}{2}\sqrt{2^{-\widetilde{\mathbf{H}}_\infty(A|C)} \cdot |\mathbb{Z}_p|} \\ &= \frac{1}{2}\sqrt{2^{-\widetilde{\mathbf{H}}_\infty(\mathcal{X}'_{\mathrm{in}}|\mathcal{X}'_{\mathrm{de}})} \cdot p} \\ &\le \frac{1}{2}\sqrt{2^{-\log_2 p - \omega(\log_2 k)} \cdot p} \\ &= k^{-\omega(1)}, \end{aligned}$$

which is negligible, as required. This completes the proof that $\mathcal{S}_{\mathtt{Hash}}$ satisfies weak simulatability, and the entire proof of Lemma 10. □

7.3 Full description

Here, we give the full description of our second instantiation of a fuzzy signature scheme, by instantiating the underlying linear sketch and signature schemes in the generic construction, with the concrete linear sketch scheme $\mathcal{S}_{\mathtt{Hash}}$ (given in Sect. 7.2) and the Schnorr signature scheme $\Sigma_{\mathtt{Sch}}$ (described in Fig. 3 (right)), respectively.

Let $\mathcal{F}_2 = ((d, X), t, \mathcal{X}, \Phi, \epsilon)$ be the fuzzy key setting that we specified in Sect. 7.1, and suppose the dimension of the fuzzy data space is n. Let GGen be a group generator (which we assume to produce a description of a group whose order is p). Let $\mathcal{H}_{\mathrm{lin}} = \{h_z \colon (\mathbb{F}_p)^n \to \mathbb{F}_p\}_{z \in \mathbb{F}_{p^n}}$ be the universal hash family with linearity introduced in Sect. 2.3. (As in previous sections, we identify $\mathbb{F}_p$ with $\mathbb{Z}_p$.) Let $H \colon \{0, 1\}^* \to \mathbb{Z}_p$ be a cryptographic hash function which will be modeled as a random oracle. Using these building blocks, our second fuzzy signature scheme $\Sigma_{\mathsf{FS2}} = (\mathsf{Setup}_{\mathsf{FS2}}, \mathsf{KG}_{\mathsf{FS2}}, \mathsf{Sign}_{\mathsf{FS2}}, \mathsf{Ver}_{\mathsf{FS2}})$ for the fuzzy key setting $\mathcal{F}_2$ is constructed as in Fig. 10.[18]

[17] Hence, it in fact achieves weak simulatability with the optimal multiplicative simulation error $u(k) = 1$, while in order for the security proof of our generic construction to go through, it is sufficient for u to be some polynomial of k.

[18] In Fig. 10, the operations involving "$\mathtt{Round}_\ell$" enclosed by a box in $\mathsf{KG}_{\mathsf{FS2}}$ and $\mathsf{Sign}_{\mathsf{FS2}}$ are those for concerning practical treatment of

```
$\mathsf{Setup}_{\mathsf{FS2}}(\mathcal{F}_2, 1^k)$ :
  $\mathcal{G} := (p, \mathbb{G}, g) \leftarrow \mathsf{GGen}(1^k)$
  Let $H : \{0,1\}^* \rightarrow \mathbb{Z}_p$
            be a hash function.
  $z \leftarrow_{\mathtt{R}} \mathbb{F}_{p^n}$
  Return $pp \leftarrow (\mathcal{G}, z, H)$.
```

```
$\mathsf{KG}_{\mathsf{FS2}}(pp, \mathbf{x})$ :
  $(\mathcal{G}, z, H) \leftarrow pp$
  $sk \leftarrow_{\mathtt{R}} \mathbb{Z}_p$
  $vk \leftarrow g^{sk}$
  $\mathbf{a} \leftarrow_{\mathtt{R}} h_z^{-1}(sk)$
  $\mathbf{c} \leftarrow \mathbf{a} + T \cdot \mathbf{x}$ (†)
  $\boxed{\mathbf{c} \leftarrow \mathtt{Round}_\ell(\mathbf{c})}$ (‡)
  Return $VK \leftarrow (vk, \mathbf{c})$.
```

```
$\mathsf{Sign}_{\mathsf{FS2}}(pp, \mathbf{x}', m)$ :
  $(\mathcal{G}, z, H) \leftarrow pp$
  $\widetilde{sk} \leftarrow_{\mathtt{R}} \mathbb{Z}_p$
  $\widetilde{vk} \leftarrow g^{\widetilde{sk}}$
  $r \leftarrow_{\mathtt{R}} \mathbb{Z}_p$
  $R \leftarrow g^r$
  $\widetilde{h} \leftarrow H(R \| m)$
  $\widetilde{s} \leftarrow r + (\widetilde{sk}) \cdot \widetilde{h} \bmod p$
  $\mathbf{a}' \leftarrow_{\mathtt{R}} h_z^{-1}(\widetilde{sk})$
  $\widetilde{\mathbf{c}} \leftarrow \mathbf{a}' + T \cdot \mathbf{x}'$ (†)
  $\boxed{\widetilde{\mathbf{c}} \leftarrow \mathtt{Round}_\ell(\widetilde{\mathbf{c}})}$ (‡)
  $\sigma \leftarrow (\widetilde{vk}, \widetilde{h}, \widetilde{s}, \widetilde{\mathbf{c}})$.
  Return $\sigma$.
```

```
$\mathsf{Ver}_{\mathsf{FS2}}(pp, VK, m, \sigma)$ :
  $(\mathcal{G}, z, H) \leftarrow pp$
  $(vk, \mathbf{c}) \leftarrow VK$
  $(\widetilde{vk}, \widetilde{h}, \widetilde{s}, \widetilde{\mathbf{c}}) \leftarrow \sigma$
  $R \leftarrow g^{\widetilde{s}} \cdot (\widetilde{vk})^{-\widetilde{h}}$
  If $H(R \| m) \neq \widetilde{h}$ then return $\perp$.
  $\Delta\mathbf{c} \leftarrow \widetilde{\mathbf{c}} - \mathbf{c}$ (†)
  $\Delta sk \leftarrow h_s(\lfloor \Delta\mathbf{c} \rceil)$
  If $vk \cdot g^{\Delta sk} = \widetilde{vk}$ then
        return $\top$ else return $\perp$.
```

Fig. 10 Our second instantiation of a fuzzy signature scheme Σ_{FS2}. $^{(\dagger)}$ The operation "+" (resp. "−") in $(\mathbb{R}_p)^n$ are the coordinate-wise addition (resp. subtraction) in $\mathbb{R}_p$. $^{(\ddagger)}$ The operations involving "$\mathtt{Round}_\ell$" enclosed by a box in $\mathsf{KG}_{\mathsf{FS2}}$ and $\mathsf{Sign}_{\mathsf{FS2}}$ are those for concerning practical treatment of decimal numbers explained in Sect. 8. (The reader who has not read there is expected to ignore them)

The following theorem guarantees the correctness and security of our second scheme Σ_{FS2}, which is obtained as a corollary of the combination of Theorems 1 and 2, and Lemmas 6 and 10.

Theorem 4 *The fuzzy signature scheme Σ_{FS2} for the fuzzy key setting $\mathcal{F}_2$ in Fig. 10 is ϵ-correct. Furthermore, if the DL assumption holds with respect to* GGen*, then Σ_{FS2} is* $\mathtt{EUF\text{-}CMA}$ *secure in the random oracle model where H is modeled as a random oracle.*

Although our second instantiation Σ_{FS2} can be shown to be secure only in the random oracle model due to the reliance on the Schnorr scheme, it has several practical advantages compared to our first instantiation Σ_{FS2} given in Sect. 6. Specifically, Σ_{FS2} does not require bilinear maps, and the public parameter size can be much shorter than that in Σ_{FS1}. More importantly, Σ_{FS2} works for the fuzzy key setting in which fuzzy data cannot be assumed to be distributed uniformly over the data space (which was required in Σ_{FS1}), but that only its average min-entropy (given some parts of the fuzzy data) is sufficiently high.

8 On the treatment of real numbers in implementations

In this section, we revisit and discuss the treatment of real numbers in our proposed fuzzy signature schemes.

Let us quickly remind the reader: As mentioned at the "*On the Treatment of Real Numbers*" paragraph at the beginning of the Sect. 6, in Sects. 6 and 7, we adopt the natural setting in which all real numbers are expressed so that it has a significand of an a priori fixed length λ. Treatments of real numbers are especially relevant to our concrete linear sketch schemes $\mathcal{S}_{\mathtt{CRT}}$ proposed in Sect. 6.3 and $\mathcal{S}_{\mathtt{Hash}}$ proposed in Sect. 7.2, where we showed in Lemmas 7 and 10 that our schemes $\mathcal{S}_{\mathtt{CRT}}$ and $\mathcal{S}_{\mathtt{Hash}}$ satisfy the requirements of a linear sketch scheme in Definition 12, respectively. These results in turn enable us to derive Theorems 3 and 4 that guarantee the security of our concrete fuzzy signature schemes Σ_{FS1} in Sect. 6.5 (Fig. 8) and Σ_{FS2} in Sect. 7.3 (Fig. 10).

However, naively using data with a priori fixed-size format for real numbers, is not always desirable from the viewpoint of efficiency, because it directly affects the space (or communication) complexity. During the computation, we should use as precise values as possible for them, while from the viewpoint of the space (communication) complexity, the representation size of them should be minimized.

Hence, motivated by this practical consideration, here we consider the "truncated" versions of our concrete fuzzy signature schemes in which the decimal part of the real numbers in the vectors $\mathbf{c}$ and $\widetilde{\mathbf{c}}$ appearing in our concrete fuzzy signature schemes Σ_{FS1} and Σ_{FS2} are explicitly truncated (i.e., rounded down) to some length, and discuss its effects on the correctness and security of each scheme. Fortunately, in our fuzzy signature schemes, truncating the decimal part of $\mathbf{c}$ and $\widetilde{\mathbf{c}}$ affects the correctness of the schemes, but not the security of them, as we will see in the following.

$\widehat{\Sigma_{\mathsf{FS1}}}$: *Truncated version of our first instantiation.* For a natural number $\ell \leq \ell' = \lambda - \lceil k/n \rceil$, let $\mathtt{Round}_\ell$ be the operation that takes an n-dimensional vector of real numbers as input, and outputs an n-dimensional vector such that the decimal part of each element of the vector is rounded down to an ℓ-bit value. Then, consider the fuzzy signature schemes Σ_{FS1} in Fig. 8 in which the operation $\mathtt{Round}_\ell$ enclosed in the boxes is

real numbers explained in Sect. 8. The reader who has not read there is expected to ignore them.

executed in $\mathsf{KG}_{\mathsf{FS1}}$ and $\mathsf{Sign}_{\mathsf{FS1}}$. To differentiate this truncated version from the original one Σ_{FS1}, we simply call the former the *truncated* scheme and denote it by $\widehat{\Sigma_{\mathsf{FS1}}}$. We remark that in general, to make the calculation error as small as possible, the variables appearing during calculations should be treated as accurate as possible, and thus the "rounding" operations should be applied only to the very last of the values that are stored/transmitted. The operation "$\mathtt{Round}_\ell$" in $\mathsf{KG}_{\mathsf{FS1}}$ and $\mathsf{Sign}_{\mathsf{FS1}}$ is used with this principle.

We first note that the truncated scheme $\widehat{\Sigma_{\mathsf{FS1}}}$ is as secure as the original scheme Σ_{FS1} (regardless of the value ℓ). Specially, if there exists an adversary $\mathcal{A}$ against the truncated scheme $\widehat{\Sigma_{\mathsf{FS1}}}$, we can straightforwardly convert it into another adversary $\mathcal{B}$ that attacks the security of the original scheme. The adversary $\mathcal{B}$ running in the security experiment for the original scheme Σ_{FS1} can easily simulate the security experiment for $\widehat{\Sigma_{\mathsf{FS1}}}$, and a forgery for the truncated scheme is a forgery for the original scheme.

Hence, all we need to see is what effect the truncation causes on correctness. The following theorem formally shows that if the error distribution Φ has some natural property, then the effect of the truncation on correctness is moderate.

Theorem 5 *Let $\mathcal{F}_1$ be the fuzzy key setting considered for our first instantiation Σ_{FS1}. Assume that the error distribution Φ in $\mathcal{F}_1$ satisfies the additional property that there exists a constant c such that* $\Pr[\mathbf{e} \leftarrow_R \Phi \colon \|\mathsf{E}_{\mathbf{w}}(\mathbf{e})\|_\infty < 0.5 - \delta] \geq 1 - \epsilon - c \cdot \delta$ *holds for all $\delta \in [0, 0.5)$. Then, the truncated scheme $\widehat{\Sigma_{\mathsf{FS1}}}$ is $(2c \cdot 2^{-\ell} + \epsilon)$-correct.*

Recall that the fuzzy key setting $\mathcal{F}_1$ for our first instantiation Σ_{FS1} originally requires that $\Pr[\mathbf{e} \leftarrow_{\mathrm{R}} \Phi \colon \|\mathbf{e}\|_\infty < t] \geq 1 - \epsilon$, which implies $\Pr[\mathbf{e} \leftarrow_{\mathrm{R}} \Phi \colon \|\mathsf{E}_{\mathbf{w}}(\mathbf{e})\|_\infty < 0.5] \geq 1 - \epsilon$. Note that this corresponds to the case that $\delta = 0$ in the assumption on the error distribution Φ. We can interpret the additional assumption on Φ as the requirement that the probability distribution of Φ has monotonically non-increasing tails. Such a condition is satisfied by most natural error distributions, such as the Gaussian distribution and the uniform distribution.

Proof of Theorem 5 Suppose $\mathbf{x}$ is a fuzzy data that is used to generate a verification key $VK = (vk = g^{z^{sk}}, \mathbf{c} = \mathsf{CRT}_{\mathbf{w}}^{-1}(sk) + \mathsf{E}_{\mathbf{w}}(\mathbf{x}))$, and $\mathbf{x}' = \mathbf{x} + \mathbf{e}$ is a fuzzy data used for generating a signature $\sigma = (\widetilde{vk} = g^{z^{\widetilde{sk}}}, \widetilde{\sigma}_1, \widetilde{\sigma}_2, \widetilde{\mathbf{c}} = \mathsf{CRT}_{\mathbf{w}}^{-1}(\widetilde{sk}) + \mathsf{E}_{\mathbf{w}}(\mathbf{x} + \mathbf{e}))$ of some message m, where $\mathbf{e} \leftarrow_{\mathrm{R}} \Phi$. Let

$$\mathbf{c}' = \mathtt{Round}_\ell(\mathbf{c}) = \mathsf{CRT}_{\mathbf{w}}^{-1}(sk) + \mathtt{Round}_\ell(\mathsf{E}_{\mathbf{w}}(\mathbf{x})) \quad \text{and}$$
$$\widetilde{\mathbf{c}}' = \mathtt{Round}_\ell(\widetilde{\mathbf{c}}') = \mathsf{CRT}_{\mathbf{w}}^{-1}(\widetilde{sk}) + \mathtt{Round}_\ell(\mathsf{E}_{\mathbf{w}}(\mathbf{x} + \mathbf{e})).$$

Let $VK' = (vk, \mathbf{c}')$ and $\sigma' = (\widetilde{vk}, \widetilde{\sigma}_1, \widetilde{\sigma}_2, \widetilde{\mathbf{c}}')$. (Note that VK' and σ' are the "truncated" versions of VK and σ, respectively.)

Now, consider the verification of (m, σ') under the verification key VK'. Due to our design of Σ_{FS1}, $\mathsf{Ver}_{\mathsf{FS1}}(pp, VK', m, \sigma') = \top$ occurs as long as $\mathsf{C}_{\mathbf{w}}(\widetilde{\mathbf{c}}' - \mathbf{c}') = \mathsf{CRT}^{-1}(\widetilde{sk} - sk)$ holds, and the latter condition is in turn implied by the condition $\|\mathtt{Round}_\ell(\mathsf{E}_{\mathbf{w}}(\mathbf{x} + \mathbf{e})) - \mathtt{Round}_\ell(\mathsf{E}_{\mathbf{w}}(\mathbf{x}))\|_\infty < 0.5$. We can upperbound the left hand side of this condition as follows:

$$\begin{aligned}
&\left\|\mathtt{Round}_\ell(\mathsf{E}_{\mathbf{w}}(\mathbf{x} + \mathbf{e})) - \mathtt{Round}_\ell(\mathsf{E}_{\mathbf{w}}(\mathbf{x}))\right\|_\infty \\
&\quad \leq \left\|\mathtt{Round}_\ell(\mathsf{E}_{\mathbf{w}}(\mathbf{x} + \mathbf{e})) - \mathsf{E}_{\mathbf{w}}(\mathbf{x} + \mathbf{e})\right\|_\infty \\
&\qquad + \left\|\mathsf{E}_{\mathbf{w}}(\mathbf{x} + \mathbf{e}) - \mathsf{E}_{\mathbf{w}}(\mathbf{x})\right\|_\infty \\
&\qquad + \left\|\mathsf{E}_{\mathbf{w}}(\mathbf{x}) - \mathtt{Round}_\ell(\mathsf{E}_{\mathbf{w}}(\mathbf{x}))\right\|_\infty \\
&\quad \leq 2 \cdot 2^{-\ell} + \left\|\mathsf{E}_{\mathbf{w}}(\mathbf{e})\right\|_\infty,
\end{aligned}$$

where the first inequality is due to the triangle inequality, and in the second inequality we used $\|\mathtt{Round}_\ell(\mathbf{y}) - \mathbf{y}\|_\infty \leq 2^{-\ell}$ holds for any $\mathbf{y} \in \mathbb{R}^n_{\mathbf{w}}$ (because $\mathtt{Round}_\ell(\mathbf{y})$ just truncates all but ℓ bits of the decimal part of $\mathbf{y}$), and $\mathsf{E}_{\mathbf{w}}(\mathbf{x} + \mathbf{e}) = \mathsf{E}_{\mathbf{w}}(\mathbf{x}) + \mathsf{E}_{\mathbf{w}}(\mathbf{e})$ which is due to the linearity of $\mathsf{E}_{\mathbf{w}}$ [Eq. (13)]. Hence, if $\|\mathsf{E}(\mathbf{e})\|_\infty < 0.5 - 2 \cdot 2^{-\ell}$ holds, we have $\mathsf{Ver}_{\mathsf{FS1}}(pp, VK', m, \sigma') = \top$. Due to the given condition on Φ, it occurs with probability at least $1 - \epsilon - c \cdot (2 \cdot 2^{-\ell})$ when $e \leftarrow_{\mathrm{R}} \Phi$. Hence, we can conclude that the truncated scheme $\widehat{\Sigma_{\mathsf{FS1}}}$ is $(2c \cdot 2^{-\ell} + \epsilon)$-correct. □

$\widehat{\Sigma_{\mathsf{FS2}}}$: *Truncated version of our second instantiation.* Let $\ell \leq \lambda - \lceil \log_2 T \rceil$ be a natural number. Similarly to the above, consider the fuzzy signature scheme Σ_{FS2} in Fig. 10 in which the operation $\mathtt{Round}_\ell$ enclosed in the boxes is executed in $\mathsf{KG}_{\mathsf{FS2}}$ and $\mathsf{Sign}_{\mathsf{FS2}}$. We call it the truncated scheme and denote it by $\widehat{\Sigma_{\mathsf{FS2}}}$.

Then, as is the case with $\widehat{\Sigma_{\mathsf{FS1}}}$, the truncated scheme $\widehat{\Sigma_{\mathsf{FS2}}}$ is as secure as our original second instantiation Σ_{FS2}.

Furthermore, with essentially the same way as in $\widehat{\Sigma_{\mathsf{FS1}}}$, we can prove the following theorem for $\widehat{\Sigma_{\mathsf{FS2}}}$. (Since the proof is essentially the same as that of Theorem 5, we omit it.)

Theorem 6 *Let $\mathcal{F}_2$ be the fuzzy key setting considered for our second instantiation Σ_{FS2}. Assume that the error distribution Φ in $\mathcal{F}_2$ satisfies the additional property that there exists a constant c such that* $\Pr[\mathbf{e} \leftarrow_R \Phi \colon \|T \cdot \mathbf{e}\|_\infty < 0.5 - \delta] \geq 1 - \epsilon - c \cdot \delta$ *holds for all* $\delta \in [0, 0.5)$. *Then, the truncated scheme $\widehat{\Sigma_{\mathsf{FS2}}}$ is $(2c \cdot 2^{-\ell} + \epsilon)$-correct.*

Relaxing the requirement on fuzzy data by truncation. Finally, we remark that the truncation for the second scheme also enables us to weaken the requirement on the distribution $\mathcal{X}$ of fuzzy data. Specifically, let $\mathcal{X}'$ be the scaled-up version of $\mathcal{X}$ (by T), and let $\mathcal{X}'_{\mathrm{in}}$ and $\mathcal{X}'_{\mathrm{de}}$ be the integer and decimal part of $\mathcal{X}'$, respectively. Then, in order to carry out the security proof for the truncated version $\widehat{\Sigma_{\mathsf{FS2}}}$, we only need to require $\widetilde{\mathbf{H}}_\infty(\mathcal{X}'_{\mathrm{in}}|\mathtt{Round}_\ell(\mathcal{X}'_{\mathrm{de}})) \geq \log_2 p + \omega(\log_2 k)$. Note that this is a strict relaxation compared to requiring $\widetilde{\mathbf{H}}_\infty(\mathcal{X}'_{\mathrm{in}}|\mathcal{X}'_{\mathrm{de}}) \geq \log_2 p + \omega(\log_2 k)$. This is because $\widetilde{\mathbf{H}}_\infty(\mathcal{X}'_{\mathrm{in}}|\mathtt{Round}_\ell(\mathcal{X}'_{\mathrm{de}})) \geq \widetilde{\mathbf{H}}_\infty(\mathcal{X}'_{\mathrm{in}}|\mathcal{X}'_{\mathrm{de}})$ holds, which is in turn because $\mathtt{Round}_\ell(\mathcal{X}'_{\mathrm{de}})$ is a (strict) part of $\mathcal{X}'_{\mathrm{de}}$, and thus $\widetilde{\mathbf{H}}_\infty(\mathcal{X}'_{\mathrm{in}}|\mathtt{Round}_\ell(\mathcal{X}'_{\mathrm{de}})) \geq \widetilde{\mathbf{H}}_\infty(\mathcal{X}'_{\mathrm{in}}|\mathcal{X}'_{\mathrm{de}})$ holds.[19]

9 Toward public biometric infrastructure

As one of the promising applications of our fuzzy signature schemes, we discuss how it can be used to realize a biometric-based PKI that we call the *public biometric infrastructure* (PBI).

The PBI is a biometric-based PKI that allows to use biometric data itself as a private key. Since it does not require a helper string to extract a private key, it does not require users to carry a dedicated device that stores it. Like the PKI, it provides the following functionalities: (1) registration, (2) digital signature, (3) authentication, and (4) cryptographic communication. At the time of registration, a user presents his/her biometric data x, from which the public key pk is generated. A certificate authority (CA) issues a public key certificate to ensure the link between pk and the user's identify (in the same way as the PKI). It must be sufficiently hard to restore x or estimate any "acceptable" biometric feature (i.e., biometric feature $\tilde{x}$ that is sufficiently close to x) from pk. This requirement is often referred to as *irreversibility* [15,32]. Note that the irreversibility is clearly included in the unforgeability, since the adversary who obtains x or $\tilde{x}$ can forge a signature σ for any message m. Since our fuzzy signature schemes are proved to be secure, it also satisfies the irreversibility.

It is well known that a digital signature scheme can be used to realize authentication and cryptographic communication, as standardized in [16]. Firstly, a challenge-response authentication protocol can be constructed based on a digital signature scheme (refer to [30] for details). Secondly, an authenticated key exchange (AKE) protocol can also be constructed based on a digital signature scheme and the Diffie–Hellman key exchange protocol. In the same way, we can construct an authentication protocol and a cryptographic communication protocol in the PBI using our fuzzy signature schemes.

On the revocation functionality in the PBI. One of the fundamental functionalities in a standard PKI is the revocation functionality. When considering the revocation functionality in the PBI, we think the following two basic functionalities should be considered: (1) revocation of a certificate (and thereby revoking the corresponding secret key). (2) Re-issuance of a certificate for a user whose public key had a certificate but was revoked previously.

In the PBI, revocation of a certificate can be realized just as in a standard PKI: We can just add the information of a certificate to be revoked into the certificate revocation list (CRL) maintained by a CA. Then, we can just treat transactions involving fuzzy signatures under a public key with a revoked certificate, as invalid.

Whether re-issuance of a certificate can be realized exactly as in a standard PKI, depends on the cause of the revocation of a user's previous certificate. If the cause of the previous revocation is on the CA's side (say, due to the leak of the CA's secret key) and the confidentiality of the user's secret key has not been affected, then re-issuance of a certificate on the user's public key is possible just as in a standard PKI: the user can ask for a new certificate on his/her public key (from another CA or from the same CA with its new secret key). However, if the cause of the previous revocation is on the user's side (say, due to the leak of the user's secret key from which his/her public key is generated), then things are not so easy: In the PBI, a secret key is generated from a biometric feature and thus, unlike in a standard PKI with standard signature schemes, a new (fresh) secret key cannot be generated as many times as one wants from one person. This is an inherent limitation of the PBI, compared to a standard PKI. (However, let us remark that the problem that the number of times we can extract fresh biometric information is limited, is not unique to the PBI or fuzzy signatures, but rather it is a problem that exists virtually in any biometrics-based authentication technologies.) How many times fresh secret keys can be generated from one person, will depend on what biometric features are adopted in an actual implementation of a fuzzy signature scheme.

Although how to extract biometric information from actual biometric features in the form of fuzzy data formalized in this paper is beyond the scope of our paper, we note that if multiple biometric features (individually or in combination) are supported, the number of times one person can generate a fresh secret key could be increased. Furthermore, in the literature of biometrics, there are several researches that could be useful to overcome the above limitation. For example, recently Fujita et al. [12] proposed the "micro biometrics authentication mechanism," which is a biometric authentication method by using minute patterns of human body parts, such as a very small area of human skin texture measured via a microscope, as a biometric feature. Such biometric features allow us to increase the number of times one can extract biometric information from one person. If fuzzy signature schemes for this type of biometric feature are realized, the

[19] Note that the definition of average min-entropy implies that $\widetilde{\mathbf{H}}_\infty(A|B, C) \leq \widetilde{\mathbf{H}}_\infty(A|B)$ holds for any joint distribution (A, B, C).

number of times one person can generate a fresh secret key could be increased.

On the plausibility of our requirement on the distribution of fuzzy data. For the security proofs to go through, our first concrete fuzzy signature scheme (given in Sect. 6) requires that the fuzzy data is uniformly distributed, and our second scheme (given in Sect. 7) requires that the average min-entropy in the presence of leakage (where the leakage is the "decimal" part of the "scaled-up version" of fuzzy data, $\widetilde{\mathbf{H}}_\infty(\mathcal{X}'_{\mathrm{in}}|\mathcal{X}'_{\mathrm{de}})$ in our notation).

A natural question would be whether practical fuzzy key settings can satisfy our requirements. The requirement that fuzzy data is uniformly distributed, is somewhat a strong assumption, and may not be suitable for biometrics-based applications, and hence we focus on the latter requirement.

In the biometric setting, which is one of the main motivations for considering fuzzy signature schemes (and thus is one of the most important settings that should be captured by the formalization of a fuzzy data setting), a well-known approach to measure the biometric entropy is *discrimination entropy* proposed by Daugman [6]. He considered a distribution of a Hamming distance m between two iriscodes (well-known iris features [7]) that are extracted from two different irises, and showed that it can be quite well approximated using the binomial distribution $B(n, p)$, where $n = 249$ and $p = 0.5$. He referred to the parameter n $(= 249)$ as a discrimination entropy. The probability that two different iriscodes exactly match can be approximated to be 2^{-249}. This is a positive news for us, and for the future of related research.

However, of course, that the probability of two different iriscodes matching is approximated as 2^{-249}, does not necessarily mean that using iriscode x as fuzzy data gives us 249-bit security. Especially, in our case, we need to take into account the leakage (information leaked from the "decimal" part $\mathcal{X}'_{\mathrm{de}}$), when the data is cast into our setting. We have to choose the threshold t by taking into account various other things, such as $\mathtt{FAR}$ and $\mathtt{FRR}$. (Note that an adversary does not have to estimate the original iriscode x, but only has to estimate an iriscode $\tilde{x}$ that is sufficiently close to x.) Therefore, it seems not so easy to use the results from [6,7] just as it is.

If a single biometric feature does not have enough entropy, then one of the promising solutions to the problem would be to combine multiple biometric features. For example, Murakami et al. [22] recently showed that by combining four finger-vein features, $\mathtt{FAR} = 2^{-133}$ (resp. $\mathtt{FAR} = 2^{-87}$) can be achieved in the case when $\mathtt{FRR} = 0.055$ (resp. $\mathtt{FRR} = 0.0053$). Also, a multibiometric sensor that simultaneously acquires multiple biometrics (e.g., iris and face [5]; fingerprint and finger-vein [27]) has also been widely developed. Thus, we believe that using multiple biometrics is a promising direction for increasing entropy without affecting usability (which is also an important factor in practice).

It is also important to note that (an approximation of) $\widetilde{\mathbf{H}}_\infty(\mathcal{X}'_{\mathrm{in}}|\mathcal{X}'_{\mathrm{de}})$ could be experimentally estimated by using real fuzzy data (in a similar manner done in [22]). This is an important feature in order for fuzzy signature schemes (and security systems based on them) to be used in practice.

Open problems. It would be important to tackle the problem of whether we can realize the fuzzy key setting required in our work by some practical biometric settings/systems. It is also worth tackling whether further relaxing the requirement than our specific fuzzy key setting is possible. In particular, for our second scheme, we used the leftover hash lemma to guarantee the weak simulatability of the linear sketch scheme, but it achieves the optimal simulation error $u = 1$ and is stronger than what is required for our proof to go through. Can we use other tools (e.g., the more recent version of the leftover hash lemma by Barak et al. [1]) to further weaken the requirement on the average min-entropy?

It is also an interesting open problem to consider constructing fuzzy signature schemes over fuzzy key settings that are different from ours. For example, can we construct a fuzzy signature scheme with other types of metric spaces (e.g., Euclid distance, Hamming distance, edit distance, etc.)? It would also be worth clarifying whether we can construct more fuzzy signature schemes based on other existing signature schemes.

Acknowledgements The authors would like to thank the anonymous reviewers of ACNS 2015, ACNS 2016, and International Journal of Information Security for their invaluable comments and suggestions.

Compliance with ethical standards

Conflict of interest The authors declare that they have no conflict of interest.

Ethical approval This article does not contain any studies with human participants or animals performed by any of the authors.

A More on the limitations of fuzzy-extractor-based approaches

The right of Fig. 1 shows an example of a digital signature system using a fuzzy extractor. Assume that the client generates a signature on a message, and the server verifies it. At the time of registration, a signing key sk and a helper string

P are generated from a noisy string (e.g., biometric feature) x, and a verification key vk corresponding to sk is generated and stored in a server-side DB. At the time of signing, the client generates a signature σ on a message m using P and another noisy string x', and sends σ to the server. The server verifies whether σ is a valid signature on m under vk. If x' is close to x, it outputs "$\top$" (valid). Otherwise, it outputs "$\perp$" (invalid). The important point here is that the helper string P has to be stored in some place so that the client can retrieve it at the time of signing.

There are three possible models for storing the helper string: *Store-on-Token* (SOT), *Store-on-Client* (SOC), and *Store-on-Server* (SOS). In the SOT, the helper string is stored in a hardware token (e.g., smart card, USB token). Since this model requires each user to possess a token, it reduces usability. In the SOC, the helper string is stored in a client device. Although this model can be applied to the applications where each user has his/her own client device, it cannot be employed if the client device is shared by general public (e.g., bank ATM, POS, and kiosk terminal). In the SOS, the helper string is stored in a server-side DB, and the client queries for the helper string to the server at the time of signing. However, it cannot be used in an offline environment (i.e., a user generates a signature, which is sent to the server later, offline).

To sum up, the SOT reduces usability, and the SOC/SOS limits the client environment. Although a digital signature scheme using biometrics is proposed in [17,18] and an extended version of the PKI based on biometrics is discussed in [29], all of them require additional data like the helper string and suffer from this kind of problem.

B Differences among $\mathtt{RKA}^*$ security and existing RKA security definitions

As mentioned earlier, our definition of $\mathtt{RKA}^*$ security has subtle differences with the popular definition of RKA security for signature schemes by Bellare et al. [2]. Specifically, an adversary in the RKA security experiment of [2] has to come up with a forgery pair (m', σ') that is under the original verification key vk, while an adversary in our definition is allowed to additionally output a function ϕ', and is considered successful if (m', σ') is a valid forgery under the "related" verification key $vk' = \mathsf{KG}'(pp, \phi'(sk))$. In this aspect, our definition is less restrictive than that of [2]. On the other hand, in the RKA security experiment of [2], a message m used as a signing query (ϕ, m) is included into the "used message list" $\mathcal{Q}$ only if $\phi(sk) = sk$, while in our definition, any message used as a signing query is included in $\mathcal{Q}$. Since the message m' used as a forgery needs to satisfy $m' \notin \mathcal{Q}$, in this aspect our adversary is more restrictive than that of [2]. Because of the differences, there seem to be no obvious implications from one notion to another in both directions.

Recently, Morita et al. [20,21] defined the so-called Φ-weak-RKA security, which is defined in the same manner as the RKA security definition of [2], except that an adversary has to forge a new message that has not been signed by the signing oracle (like in our definition). However, their definition does not allow an adversary to modify the verification key. Therefore, our definition of $\Phi\text{-}\mathtt{RKA}^*$ security is strictly stronger than the Φ-weak-RKA security of [20,21] (for the same function class Φ).

C Our previous definitions of linear sketch

In this section, we review the definition of a linear sketch scheme that we introduced in ACNS'15 [33] and in ACNS'16 [19] for self-containment, and discuss the difference with the one we give in Sect. 4.3.

C.1 ACNS'15 version

Definition 13 Let $\mathcal{F} = ((\mathsf{d}, X), t, \mathcal{X}, \Phi, \epsilon)$ be a fuzzy key setting. In [33], a linear sketch scheme $\mathcal{S}$ for $\mathcal{F}$ was defined as a pair of *deterministic* PTAs (Sketch, $\mathsf{DiffRec}$) that satisfies the following three properties:

Syntax and correctness. Sketch is the "sketching" algorithm that takes the description Λ of an abelian group $(\mathcal{K}, +)$, an element $s \in \mathcal{K}$, and a fuzzy data $x \in X$ as input, and outputs a "sketch" c; $\mathsf{DiffRec}$ is the "difference reconstruction" algorithm that takes Λ and two values c, c' (supposedly output by Sketch) as input, and outputs the "difference" $\Delta sk \in \mathcal{K}$.
It is required that for all $x, x' \in X$ such that $\mathsf{d}(x, x') < t$, and for all $s, \Delta s \in \mathcal{K}$, it holds that

$$\mathsf{DiffRec}\Big(\Lambda, \mathsf{Sketch}(\Lambda, s, x), \mathsf{Sketch}(\Lambda, s+\Delta s, x')\Big) = \Delta s.$$

Linearity. There exists a deterministic PTA $\mathsf{M_c}$ satisfying the following: For all $x, e \in X$,[20] and for all $s, \Delta s \in \mathcal{K}$, it holds that

$$\mathsf{Sketch}(\Lambda, s+\Delta s, x+e) = \mathsf{M_c}(\Lambda, \mathsf{Sketch}(\Lambda, s, x), \Delta s, e).$$

Simulatability. There exists a PPTA Sim such that for all $s \in \mathcal{K}$, the following two distributions are statistically

[20] In the original version [33], x and e were quantified as "for all $x, e \in X$ such that $\mathsf{d}(x, x+e) < t$." However, the "such that $\mathsf{d}(x, x+e) < t$" condition should not be there, and the definition here reflects this correction.

indistinguishable (in the security parameter k that is associated with $t \in \mathcal{F}$):

$$\{x \leftarrow_{\mathrm{R}} \mathcal{X};\ c \leftarrow \mathsf{Sketch}(\Lambda, s, x) \colon c\} \quad \text{and}$$
$$\{c \leftarrow_{\mathrm{R}} \mathsf{Sim}(\Lambda) \colon c\}.$$

Difference with Definition 12. The differences between the definition recalled above (ACNS'15 version) and Definition 12 in Sect. 4.3 are as follows:

1. Definition 12 introduces a setup algorithm that produces a public parameter used by all algorithms.
2. Definition 12 allows the sketching algorithm Sketch, and the auxiliary algorithm $\mathsf{M_c}$, to be probabilistic (as opposed to being deterministic required in the ACNS'15 version).
3. Definition 12 relaxes the linearity property to a weaker "distributional" variant, while in the ACNS'15 version it is defined like a correctness property that needs to be satisfied without any failure.
4. Definition 12 relaxes the simulatability property (which captures confidentiality of sketches produced by Sketch) of the ACNS'15 version, so that
   - (1) the simulatability is required only for the case the element $s \in \mathcal{K}$ is chosen uniformly at random,
   - (2) the indistinguishability of the output of Sketch and that of Sim is required to hold only against computationally bounded distinguishers, and
   - (3) most importantly, the "multiplicative" simulation error is allowed in Definition 12, which is captured by p. (In contrast, only the case of optimal simulation error $u = 1$ is allowed in the ACNS'15 version.)

C.2 ACNS'16 version

Definition 14 Let $\mathcal{F} = ((\mathsf{d}, X), t, \mathcal{X}, \Phi, \epsilon)$ be a fuzzy key setting. In [19], a linear sketch scheme $\mathcal{S}$ for $\mathcal{F}$ was defined as a tuple of PPTAs $\mathcal{S} = (\mathsf{Setup}, \mathsf{Sketch}, \mathsf{DiffRec})$ that satisfies the following three properties:

Syntax and correctness. Same as in Definition 12.
Linearity. Same as in Definition 12.
Average-case indistinguishability[21]. For all (finite) abelian groups $\Lambda = (\mathcal{K}, +)$, the following two distributions are statistically indistinguishable (in the security parameter k that is associated with t in $\mathcal{F}$):

$$\left\{ \begin{array}{l} pp \leftarrow_{\mathrm{R}} \mathsf{Setup}(\mathcal{F}, \Lambda);\ x \leftarrow_{\mathrm{R}} \mathcal{X};\ s \leftarrow_{\mathrm{R}} \mathcal{K}; \\ c \leftarrow_{\mathrm{R}} \mathsf{Sketch}(pp, s, x) \end{array} : (pp, s, c) \right\}, \quad \text{and}$$
$$\left\{ \begin{array}{l} pp \leftarrow_{\mathrm{R}} \mathsf{Setup}(\mathcal{F}, \Lambda);\ x \leftarrow_{\mathrm{R}} \mathcal{X};\ s, s' \leftarrow_{\mathrm{R}} \mathcal{K}; \\ c \leftarrow_{\mathrm{R}} \mathsf{Sketch}(pp, s, x) \end{array} : (pp, s', c) \right\} \tag{23}$$

Difference with Definition 12. We note that average-case indistinguishability implies weak simulatability. Specifically, we can define the following canonical simulator $\mathsf{Sim}(pp)$:

> $\mathsf{Sim}(pp)$: Let $\Lambda = (\mathcal{K}, +)$ be an abelian group specified in pp. Sim picks $x \leftarrow_{\mathrm{R}} \mathcal{X}$ and $s' \leftarrow_{\mathrm{R}} \mathcal{K}$. Then, Sim computes $c \leftarrow_{\mathrm{R}} \mathsf{Sketch}(pp, x, s')$, and outputs c.

It is straightforward to see that if a linear sketch scheme satisfies average-case indistinguishability, then the linear sketch with the simulator Sim defined above satisfies weak simulatability, because the "simulated" distribution $\mathcal{D}_{\mathrm{sim}} = \{ pp \leftarrow_{\mathrm{R}} \mathsf{Setup}(\mathcal{F}, \Lambda); s \leftarrow_{\mathrm{R}} \mathcal{K}; c \leftarrow_{\mathrm{R}} \mathsf{Sim}(pp) \colon (pp, s, c)\}$ is equivalent to the second distribution in Eq. (23) (where the roles of s and s' are swapped). Also, the real distribution $\mathcal{D}_{\mathrm{real}}$ considered in weak simulatability is equivalent to the first distribution in Eq. (23). Hence, by the average-case indistinguishability, $\mathbf{SD}(\mathcal{D}_{\mathrm{real}}, \mathcal{D}_{\mathrm{sim}})$ is negligible, which means that there exists a negligible function $\epsilon = \epsilon(k)$ such that for all (even computationally unbounded) algorithms $\mathcal{A}$, it holds that $\Pr[\mathcal{A}(\mathcal{D}_{\mathrm{real}}) = 1] \le \Pr[\mathcal{A}(\mathcal{D}_{\mathrm{sim}}) = 1] + \epsilon$. In fact, this is stronger than what is required for showing weak simulatability, because it shows the case in which the optimal multiplicative simulation error $u = 1$ is achieved, while it is sufficient that u is any polynomial for showing weak simulatability. The construction of the simulator shown here is used in our second concrete linear sketch scheme in Sect. 7.2.

D Proof of Lemma 4

Fix the security parameter $k \in \mathbb{N}$ and a PPTA adversary $\mathcal{A}$. For each pp [output by $\mathsf{Setup}(1^k)$], let Adv_{pp} be $\mathsf{Adv}^{\mathtt{EUF-CMA}}_{\Sigma,\mathcal{A}}(k)$ in which the public parameter is fixed as pp. We define $\widetilde{\mathsf{Adv}}_{pp}$ similarly. Note that by definition, the following equations hold:

$$\mathop{\mathbf{E}}_{pp \leftarrow_{\mathrm{R}} \mathsf{Setup}(1^k)}[\mathsf{Adv}_{pp}] = \mathsf{Adv}^{\mathtt{EUF-CMA}}_{\Sigma,\mathcal{A}}(k) \quad \text{and}$$
$$\mathop{\mathbf{E}}_{pp \leftarrow_{\mathrm{R}} \mathsf{Setup}(1^k)}[\widetilde{\mathsf{Adv}}_{pp}] = \widetilde{\mathsf{Adv}}^{\mathtt{EUF-CMA}}_{\Sigma,\mathcal{A}}(k).$$

[21] The word "average-case" in the name of average-case indistinguishability is due to the property that its definition guarantees that the element s in a sketch c is hidden only when it is chosen randomly from $\mathcal{K}$.

Next, for each pp, we define the function f_{pp} that takes a secret key $sk \in \mathcal{K}_{pp}$ as input, and outputs $\mathcal{A}$'s success probability in forging a signature in $\mathsf{Expt}^{\mathtt{EUF\text{-}CMA}}_{\Sigma,\mathcal{A}}(k)$ in which the public parameter and the secret key are fixed as pp and sk, respectively. Then, by definition, we have $\mathbf{E}[f_{pp}(U_{\mathcal{K}_{pp}})] = \mathsf{Adv}_{pp}$ and $\mathbf{E}[f_{pp}(U_{\widetilde{\mathcal{K}}_{pp}})] = \widetilde{\mathsf{Adv}}_{pp}$, where $U_{\mathcal{K}_{pp}}$ (resp. $U_{\widetilde{\mathcal{K}}_{pp}}$) is the uniform distribution over $\mathcal{K}_{pp}$ (resp. $\widetilde{\mathcal{K}}_{pp}$).

Now, by using Lemma 1, we obtain

$$\begin{aligned}\mathbf{E}\Big[f_{pp}(U_{\widetilde{\mathcal{K}}_{pp}})\Big] &\le |\mathcal{K}_{pp}| \cdot 2^{-\mathbf{H}_\infty(U_{\widetilde{\mathcal{K}}_{pp}})} \cdot \mathbf{E}\Big[f_{pp}(U_{\mathcal{K}_{pp}})\Big] \\ &= \frac{|\mathcal{K}_{pp}|}{|\widetilde{\mathcal{K}}_{pp}|} \cdot \mathbf{E}\Big[f_{pp}(U_{\mathcal{K}_{pp}})\Big] \\ &\le u(k) \cdot \mathbf{E}\Big[f_{pp}(U_{\mathcal{K}_{pp}})\Big].\end{aligned}$$

Hence, we obtain $\widetilde{\mathsf{Adv}}_{pp} \le u(k) \cdot \mathsf{Adv}_{pp}$, from which we obtain $\widetilde{\mathsf{Adv}}^{\mathtt{EUF\text{-}CMA}}_{\Sigma,\mathcal{A}}(k) \le u(k) \cdot \mathsf{Adv}^{\mathtt{EUF\text{-}CMA}}_{\Sigma,\mathcal{A}}(k)$. □

E Proof sketch of Lemma 5

For any PPTA adversary $\mathcal{A}$ that attacks the $\Phi^{\mathrm{add}}\text{-}\mathtt{RKA}^*$ security of a signature scheme satisfying the homomorphic property (as per Definition 9), one can immediately construct another adversary $\mathcal{B}$ that attacks the $\mathtt{EUF\text{-}CMA}$ security of the same signature scheme, in a fairly straightforward manner, using the algorithms $\mathsf{M}_{\mathsf{sig}}$ and M_{vk} that are guaranteed to exist due to the homomorphic property.

Specifically, when the $\mathtt{EUF\text{-}CMA}$ security experiment begins, $\mathcal{B}$ receives (pp, vk) as input from the experiment, then inputs them to $\mathcal{A}$, and starts simulating the $\Phi^{\mathrm{add}}\text{-}\mathtt{RKA}^*$ experiment for $\mathcal{A}$. For a RKA-signing query $(\phi^{\mathrm{add}}_{\Delta sk}, m)$ from $\mathcal{A}$, $\mathcal{B}$ firstly submits m to its own signing oracle and obtains a signature $\widehat{\sigma}$, and then computes $\sigma \leftarrow \mathsf{M}_{\mathsf{sig}}(pp, vk, m, \widehat{\sigma}, \Delta sk)$, which is distributed identically to a signature generated by using a secret key $sk + \Delta sk$ due to the property of $\mathsf{M}_{\mathsf{sig}}$. Furthermore, when $\mathcal{A}$ finally outputs a forgery $(\phi^{\mathrm{add}}_{\Delta sk'}, m', \sigma')$, $\mathcal{B}$ can compute $\widehat{\sigma}' \leftarrow \mathsf{M}_{\mathsf{sig}}(pp, vk', m', \sigma', -\Delta sk')$ where $vk' = \mathsf{M}_{\mathsf{vk}}(pp, vk, \Delta sk') = \mathsf{KG}'(pp, sk + \Delta sk')$. Due to the property of $\mathsf{M}_{\mathsf{sig}}$ and M_{vk}, $\widehat{\sigma}'$ is a valid signature on the message m' under the verification key vk, whenever (m', σ') is a valid forgery pair under the verification key vk'. Therefore, $\mathcal{B}$'s $\mathtt{EUF\text{-}CMA}$ advantage is exactly the same as the $\Phi^{\mathrm{add}}\text{-}\mathtt{RKA}^*$ advantage of $\mathcal{A}$.

Hence, if a signature scheme with the homomorphic property is $\mathtt{EUF\text{-}CMA}$ secure, it is $\Phi^{\mathrm{add}}\text{-}\mathtt{RKA}^*$ secure as well. □

F Proof of Lemma 6

We first recall the general forking lemma shown by Bellare and Neven [3], which will be used in the proof of Lemma 6.

Lemma 11 (General forking Lemma [3]) *Let S be a finite set with $|S| \ge 2$, $Q > 0$ be an integer, and IG be a probabilistic algorithm, called an* instance generator*, that outputs a string X (called an instance). Let $\mathcal{F}$ be a probabilistic algorithm that takes an instance (output by IG) and Q values $h_1, \dots, h_Q \in S$ as input, and outputs a pair (J, V), where J is an integer between 0 and Q, and V is any string.*

For such an algorithm $\mathcal{F}$, we consider the corresponding "forking" algorithm $\mathsf{Fork}_{\mathcal{F}}$ that takes an instance X (output by IG) as input, and runs as follows:

```
Fork_F(X):
 1. Pick a randomness $r_\mathcal{F}$ for $\mathcal{F}$ uniformly at random.
 2. $h_1, \dots, h_Q \leftarrow_R S$.
 3. $(J, V) \leftarrow \mathcal{F}(X, h_1, \dots, h_Q; r_\mathcal{F})$.
 4. If $J = 0$ then return $(0, \perp, \perp)$.
 5. $h'_J, \dots, h'_Q \leftarrow_R S$.
 6. $(J', V') \leftarrow \mathcal{F}(X, h_1, \dots, h_{J-1}, h'_J, \dots, h'_Q; r_\mathcal{F})$.
 7. If $J = J'$ and $h_J \ne h'_J$ then return $(1, V, V')$ else
    return $(0, \perp, \perp)$.
```

Let $\mathsf{acc}_{\mathcal{F}}$ and $\mathsf{frk}_{\mathcal{F}}$ be the probabilities defined as follows:

$$\mathsf{acc}_{\mathcal{F}} := \Pr\left[\begin{array}{l} X \leftarrow_R \mathsf{IG};\ h_1, \dots, h_Q \leftarrow_R S; \\ (J, V) \leftarrow_R \mathcal{F}(X, h_1, \dots, h_Q) \end{array} : J \ge 1\right],$$
$$\mathsf{frk}_{\mathcal{F}} := \Pr\Big[X \leftarrow_R \mathsf{IG};\ (b, V, V') \leftarrow_R \mathsf{Fork}_{\mathcal{F}}(X) : b = 1\Big].$$

Then, it holds that

$$\mathsf{acc}_{\mathcal{F}} \le \frac{Q}{|S|} + \sqrt{Q \cdot \mathsf{frk}_{\mathcal{F}}}. \tag{24}$$

Now, we are ready to proceed to the proof of Lemma 6.

Proof of Lemma 6 First of all, note that for a public parameter $pp = (\mathcal{G} = (\mathbb{G}, p, g), H)$, a key pair $(vk, sk) = (y = g^x, x)$, and a "shift" $\Delta sk = \Delta x$, it holds that $\mathsf{KG}'(pp, sk + \Delta sk) = g^{x+\Delta sk} = y \cdot g^{\Delta x}$. Hence, we can define $\mathsf{M}_{\mathsf{vk}}(pp, vk, \Delta sk) := (vk) \cdot g^{\Delta sk}$, which clearly shows that $\Sigma_{\mathtt{Sch}}$ satisfies the weak homomorphic property.

Then, we go on to the proof of $\Phi^{\mathrm{add}}\text{-}\mathtt{RKA}^*$ security. Let $\mathcal{A}$ be any PPTA adversary that attacks the $\Phi^{\mathrm{add}}\text{-}\mathtt{RKA}^*$ security of the Schnorr signature scheme $\Sigma_{\mathtt{Sch}}$ in the random oracle model, and makes in total $q = q(k) > 0$ queries (where q is the total number of RKA-signing and hash queries). Without loss of generality, and for simplicity, we assume that when

$\mathcal{A}$ finally outputs $(\phi^{\mathrm{add}}_{a^*}, m^*, \sigma^* = (h^*, s^*))$ at the end of the $\Phi^{\mathrm{add}}\text{-}\mathtt{RKA}^*$ experiment,[22]

(1) m^* is different from any of messages that $\mathcal{A}$ has used as its RKA-signing queries, and
(2) at some point $\mathcal{A}$ makes a hash query of the form $(R^* \| m^*)$, where $R^* = g^{s^* - a^* \cdot h^*} \cdot y^{-h^*} (= g^{s^*} \cdot (g^{x+a^*})^{-h^*})$ and $y = vk (= g^x)$ is a verification key that $\mathcal{A}$ receives at the beginning of the $\Phi^{\mathrm{add}}\text{-}\mathtt{RKA}^*$ experiment.[23]

For such $\mathcal{A}$, we will show that there exists a PPTA $\mathcal{B}$ for solving the DL problem with respect to GGen, whose running time is almost twice that of $\mathcal{A}$, such that

$$\mathsf{Adv}^{\Phi^{\mathrm{add}}\text{-}\mathtt{RKA}^*}_{\Sigma_{\mathtt{Sch}}, \mathcal{A}}(k) \le \frac{q(q+1)}{p} + \sqrt{q \cdot \mathsf{Adv}^{\mathtt{DL}}_{\mathsf{GGen}, \mathcal{B}}(k)}, \qquad (25)$$

which is sufficient for proving Lemma 6, because due to our assumption that the DL assumption holds with respect to GGen and $p = \Theta(2^k)$, the right hand side is negligible in k, and thus so is $\mathcal{A}$'s $\Phi^{\mathrm{add}}\text{-}\mathtt{RKA}^*$ advantage.

We will use the general forking lemma (Lemma 11) for showing the above inequality, and thus we specify the set S, the number Q, the instance generator IG, and the algorithm $\mathcal{F}$, as follows: Let IG be the "instance generator" that runs $\mathcal{G} := (\mathbb{G}, p, g) \leftarrow \mathsf{GGen}(1^k)$, picks $x \leftarrow_{\mathrm{R}} \mathbb{Z}_p$, computes $y \leftarrow g^x$, and outputs $X = (\mathcal{G}, y)$. We specify the set S to be $\mathbb{Z}_p$, and the number Q to be q. Let $\mathcal{F}$ be an algorithm whose randomness $r_{\mathcal{F}}$ consists of a randomness $r_{\mathcal{A}}$ for $\mathcal{A}$ and q values $s_1, \ldots, s_q \in \mathbb{Z}_p$, which takes $X = (\mathcal{G}, y = g^x)$ and $h_1, \ldots, h_q \in \mathbb{Z}_p$ as input, and internally runs $\mathcal{A}$ as follows:

> $\mathcal{F}(X = (\mathcal{G}, y), h_1, \ldots, h_q; r_{\mathcal{F}} = (r_{\mathcal{A}}, s_1, \ldots, s_q))$: $\mathcal{F}$ sets $pp \leftarrow \mathcal{G}$ (H is modeled as a random oracle for $\mathcal{A}$ and thus is not included in pp here), and sets $vk \leftarrow y$. $\mathcal{F}$ also prepares a list L_H which is initially empty. Then, $\mathcal{F}$ runs $\mathcal{A}(pp, vk; r_{\mathcal{A}})$.
> For $\mathcal{A}$'s ith query (where $i \in [q]$), $\mathcal{F}$ responds as follows:
>
> – If the ith query is a hash query of the form $(R_i \| m_i)$, then $\mathcal{F}$ checks if there is an entry of the form $(m_i, R_i, h, *)$ for some $h \in \mathbb{Z}_p$ in the list L_H (where $*$ is any value). If this is the case, then $\mathcal{F}$ returns h to $\mathcal{A}$. Otherwise, $\mathcal{F}$ adds an entry of the form $(m_i, R_i, h_i, \bot)$ into L_H (where h_i is the value that appears in $\mathcal{F}$'s input), and returns h_i to $\mathcal{A}$.
>
> – If the ith query is a RKA-signing query of the form $(\phi^{\mathrm{add}}_{a_i}, m_i)$, then $\mathcal{F}$ first computes $R_i \leftarrow g^{s_i - a_i \cdot h_i} \cdot y^{-h_i} (= g^{s_i} \cdot (g^{x+a_i})^{-h_i})$. If there is already an entry of the form $(m_i, R_i, *, *)$ in the list L_H, then $\mathcal{F}$ gives up, and terminates with output $(0, \bot)$. (In this case, we say that $\mathcal{F}$ *fails* to answer $\mathcal{A}$'s RKA-signing query.) Otherwise, $\mathcal{F}$ adds an entry of the form (m_i, R_i, h_i, s_i) into L_H (where h_i is the value that appears in $\mathcal{F}$'s input, and s_i is the value that appears in $\mathcal{F}$'s randomness $r_{\mathcal{F}}$), and then returns a signature $\sigma_i = (h_i, s_i)$ to $\mathcal{A}$.
>
> When $\mathcal{A}$ terminates with output $(\phi^{\mathrm{add}}_{a^*}, m^*, \sigma^* = (h^*, s^*))$, $\mathcal{F}$ proceeds as follows. Let $R^* = g^{s^* - a^* \cdot h^*} \cdot y^{-h^*} = g^{s^*} \cdot (g^{x+a^*})^{-h^*}$. $\mathcal{F}$ finds an entry of the form $(m^*, R^*, h, *)$ for some $h \in \mathbb{Z}_p$ in the list L_H, where it is guaranteed that h is equal to one of $h_1, \ldots, h_q$, because by our assumption $\mathcal{A}$ must have made a hash query of the form $(R^* \| m^*)$, which must have been answered with one of $h_1, \ldots, h_q$. Let $J \in [q]$ be the index such that $h = h_J$ found in this process. (Therefore, $(m^*, R^*, h) = (m_J, R_J, h_J)$.) If $h^* = h_J$, then $\mathcal{F}$ sets $V \leftarrow (a^*, h^*, s^*)$ and terminates with output (J, V). (Note that this case corresponds to the case that $H(R^* \| m^*) = h^*$ occurs, and hence $\sigma^* = (h^*, s^*)$ is a valid signature for m^* under the "shifted verification key" $vk^* = g^{x+a^*}$ in the experiment simulated by $\mathcal{F}$.) Otherwise (i.e., $h^* \ne h_J$), $\mathcal{F}$ terminates with output $(0, \bot)$.

The above completes the description of $\mathcal{F}$. Note that the interface of $\mathcal{F}$ matches that of the algorithm $\mathcal{F}$ considered in the general forking lemma (Lemma 11).

We argue that when $\mathcal{F}$ receives an instance X (output from IG) and random elements $h_1, \ldots, h_q \in \mathbb{Z}_p$ as input, and uniformly chosen values $r_{\mathcal{F}} = (r_{\mathcal{A}}, s_1, \ldots, s_q)$ as its randomness, the probability that $\mathcal{F}$ fails to answer $\mathcal{A}$'s RKA-signing queries is upperbounded by q^2/p. This is because the value $R_i = g^{s_i - a_i \cdot h_i} \cdot y^{-h_i}$ computed by $\mathcal{F}$ when answering $\mathcal{A}$'s RKA-signing query is information-theoretically hidden from $\mathcal{A}$'s view at the point $\mathcal{A}$ makes the query (because s_i and h_i are hidden from $\mathcal{A}$'s view at the point the query is made), and thus for one particular RKA-signing query, the probability that an entry of the form $(m_i, R_i, *, *)$ has already been defined in the list L_H (and thus $\mathcal{F}$ fails to answer it) is at most q/p. Since $\mathcal{A}$ makes at most q queries, the union bound tells us that the probability that $\mathcal{F}$ fails to answer $\mathcal{A}$'s RKA-signing queries is at most q^2/p. Furthermore, note that unless $\mathcal{F}$ fails to answer $\mathcal{A}$'s RKA-signing queries, $\mathcal{F}$ perfectly simulates the $\Phi^{\mathrm{add}}\text{-}\mathtt{RKA}$ experiment (in the random oracle model) for $\mathcal{A}$. Therefore, the probability that $\mathcal{A}$ outputs a successful forgery $(\phi^{\mathrm{add}}_{a^*}, m^*, \sigma^* = (h^*, s^*))$, and correspondingly $\mathcal{F}$ outputs $(J, V = (a^*, h^*, s^*))$ such that $J \ge 1$ and

[22] In this proof, we use the asterisk (*) for representing the values regarding $\mathcal{A}$'s final output (i.e., the forgery).

[23] Note that these conditions are indeed without loss of generality, because for any PPTA adversary $\mathcal{A}$ that does not respect these conditions, we can always consider a "wrapper" algorithm $\mathcal{A}'$ that satisfies them and has exactly the same $\Phi^{\mathrm{add}}\text{-}\mathtt{RKA}^*$ advantage as $\mathcal{A}$.

$h^* = h_J$, namely $\mathsf{acc}_{\mathcal{F}}$, is at least $\mathsf{Adv}^{\Phi^{\mathrm{add}}\text{-}\mathtt{RKA}^*}_{\Sigma_{\mathtt{Sch}},\mathcal{A}}(k) - q^2/p$. Recall that by the general forking lemma [Eq. (24)], we have $\mathsf{acc}_{\mathcal{F}} \leq \frac{q}{p} + \sqrt{q \cdot \mathsf{frk}_{\mathcal{F}}}$. Consequently, we have the following inequality:

$$\mathsf{Adv}^{\Phi^{\mathrm{add}}\text{-}\mathtt{RKA}^*}_{\Sigma_{\mathtt{Sch}},\mathcal{A}}(k) \leq \frac{q(q+1)}{p} + \sqrt{q \cdot \mathsf{frk}_{\mathcal{F}}}. \qquad (26)$$

Next, we relate $\mathsf{frk}_{\mathcal{F}}$ with the advantage of another algorithm $\mathcal{B}$ for solving the DL problem. $\mathcal{B}$ receives an instance $(\mathcal{G} = (\mathbb{G}, p, g), y = g^x)$ of the problem, and tries to compute $x = \log_g y$ as follows:

> $\mathcal{B}(\mathcal{G}, y)$: $\mathcal{B}$ sets $X \leftarrow (\mathcal{G}, y)$ and runs the "forking algorithm" $\mathsf{Fork}_{\mathcal{F}}(X)$ corresponding to $\mathcal{F}$ that we described above. Let (b, V, V') be the output of $\mathsf{Fork}_{\mathcal{F}}$. If $b = 0$, then $\mathcal{B}$ gives up and aborts. Otherwise (i.e., $b = 1$), let $V = (a^*, h^*, s^*)$ and $V' = (a'^*, h'^*, s'^*)$. We have $h^* \neq h'^*$ by the definition of $\mathsf{Fork}_{\mathcal{F}}$, and we also have $R^* = g^{s^* - a^* \cdot h^*} \cdot y^{-h^*} = g^{s'^* - a'^* \cdot h'^*} \cdot y^{-h'^*}$ due to our design of $\mathcal{F}$.[24] $\mathcal{B}$ now computes
>
> $$x \leftarrow \frac{(s^* - a^* \cdot h^*) - (s'^* - a'^* \cdot h'^*)}{h'^* - h^*} \bmod p,$$
>
> and terminates with output x.

The above completes the description of $\mathcal{B}$. Note that the running time of $\mathcal{B}$ is essentially the same as that of $\mathsf{Fork}_{\mathcal{F}}$. Since $\mathsf{Fork}_{\mathcal{F}}$ runs $\mathcal{F}$ twice, and $\mathcal{F}$ in turn runs $\mathcal{A}$ once, the running time of $\mathcal{B}$ is almost twice that of $\mathcal{A}$. Furthermore, whenever $\mathsf{Fork}_{\mathcal{F}}$ outputs (b, V, V') such that $b = 1$, $\mathcal{B}$ succeeds in computing the discrete logarithm x such that $y = g^x$. Therefore, we have $\mathsf{Adv}^{\mathrm{DL}}_{\mathsf{GGen},\mathcal{B}}(k) = \mathsf{frk}_{\mathcal{F}}$. Combining this equality with Eq. (26), we obtain Eq. (25), as required. This completes the proof of Lemma 6. □

G On the plausibility of the CDH assumption with respect to $\mathsf{BGGen}_{\mathtt{MWS}}$

For the security of the MWS scheme $\Sigma_{\mathtt{MWS}}$ constructed in Sect. 6.4, we need to assume that the CDH assumption holds with respect to $\mathsf{BGGen}_{\mathtt{MWS}}$. One might suspect the plausibility of this assumption because of our specific choice of the order p. However, to the best of our knowledge, there is no effective attack on the discrete logarithm assumption in the groups $\mathbb{G}$ and $\mathbb{G}_T$, let alone the CDH assumption.

Actually, the discrete logarithm problem for the multiplicative group $(\mathbb{Z}_p^*, \cdot)$ could be easy because $W | p-1$ and $W = \prod_{i \in [n]} w_i$, and thus we can apply the Pohlig–Hellman algorithm [24] to reduce an instance of the discrete logarithm problem in $\mathbb{Z}_p^*$ to instances of the discrete logarithm problems in $\mathbb{Z}_{w_i}$. *However, it does not mean that the Pohlig–Hellman algorithm is applicable to the discrete logarithm problem in* $\mathbb{G}$ *or* $\mathbb{G}_T$*, whose order is a prime.*

Note that a verification/signing key pair (vk, sk) of the MWS scheme $\Sigma_{\mathtt{MWS}}$ is of the following form $(vk, sk) = (g^{z^{sk}}, sk)$, where $sk \leftarrow_{\mathtt{R}} \mathbb{Z}_W$, and z and W are in a public parameter pp. In fact, due to the existence of the bilinear map $e: \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$, a variant of Pollard' ρ-algorithm [26] is applicable, and one can recover sk from vk (and pp) with $O(\sqrt{W})$ steps. However, this is exponential time in a security parameter k. (Recall that $W = \Theta(2^k)$.) This also does not contradict the $\mathtt{EUF}\text{-}\mathtt{CMA}$ security of the MWS scheme shown in Lemma 8.

References

1. Barak, B., Dodis, Y., Krawczyk, H., Pereira, O., Pietrzak, K., Standaert, F.-X., Yu, Y.: Leftover hash lemma, revisited. In: CRYPTO 2011, LNCS 6841, pp. 1–20 (2011)
2. Bellare, M., Cash, D., Miller, R.: Cryptography secure against related-key attacks and tampering. In: ASIACRYPT 2011, LNCS 7073, pp. 486–503 (2011)
3. Bellare, M., Neven, G.: Multi-signatures in the plain public-key model and a general forking lemma. In: CCS 2006, pp. 390–399 (2006)
4. Cheraghchi, M.: Capacity Achieving Codes from Randomness Condensers, 2011. arxiv:0901.1866v2.pdf. Preliminary version appeared in ISIT (2009)
5. Connaughton, R., Bowyer, K.W., Flynn, P.J.: Fusion of face and iris biometrics, chapter 12. In: Burge, M.J., Bowyer, K.W. (eds.) Handbook of Iris Recognition, pp. 219–237. Springer, Berlin (2013)
6. Daugman, J.: The importance of being random: statistical principles of iris recognition. Pattern Recogn. **36**(2), 279–291 (2003)
7. Daugman, J.: How iris recognition works. IEEE Trans. Circuits Syst. Video Technol. **14**(1), 21–30 (2004)
8. Dodis, Y., Ostrovsky, R., Reyzin, L., Smith, A.: Fuzzy extractors: how to generate strong keys from biometrics and other noisy data. SIAM J. Comput. **38**(1), 97–139 (2008)
9. Dodis, Y., Yu, Y.: Overcoming weak expectations. In: TCC 2013, LNCS 7785, pp. 1–22 (2013)
10. Ellison, C., Schneier, B.: Ten risks of PKI: what you're not being told about public key infrastructure. Comput. Secur. J. **16**(1), 1–7 (2000)
11. Fan, L., Zheng, J., Yang, J.: A biometric identity based signature in the standard model. IC-NIDC **2009**, 552–556 (2009)
12. Fujita, M., Mano, Y., Kaneko, T., Takahashi, K., Nishigaki, M.: A micro biometric authentication mechanism considering minute patterns of the human body: a proposal and the first attempt. In: NBiS 2016, pp. 159–164 (2016)
13. Goldwasser, S., Micali, S., Rivest, R.L.: A digital signature scheme secure against adaptive chosen-message attacks. SIAM J. Comput. **17**(2), 281–308 (1988)
14. Håstad, J., Impagliazzo, R., Levin, L., Luby, M.: Construction of a pseudorandom generator from any one-way function. SIAM J. Comput. **28**(4), 1364–1396 (1999)

[24] Note that if $b = 1$ holds, then there is an index $J' \in [q]$ such that the first execution and the second execution of $\mathcal{F}$ in $\mathsf{Fork}_{\mathcal{F}}$ have run identically up until the point $\mathcal{A}$ makes the J'th query. Furthermore, $\mathcal{A}$'s J'th query in both of the executions is a hash query of the form $(R^* \| m^*)$, and hence $R^* = R_{J'} (= R'^* = g^{s'^* - a'^* \cdot h'^*} \cdot y^{-h'^*})$ holds.

15. ISO/IEC JTC 1/SC 27 24745. Biometric information protection (2011)
16. ISO/IEC JTC 1/SC 27 9798-3. Mechanisms using digital signature techniques (1998)
17. Jo, J.-G., Seo, J.-W., Lee, H.-W.: Biometric digital signature key generation and cryptography communication based on fingerprint. In: FAW 2007, LNCS 4613, pp. 38–49 (2007)
18. Kwon, T., Lee, H.H., Lee, J.: A practical method for generating digital signatures using biometrics. IEICE Trans. **E90–B**(6), 1381–1389 (2007)
19. Matsuda, T., Takahashi, K., Murakami, T., Hanaoka, G.: Fuzzy signatures: relaxing the requirements and a new construction. In: ACNS 2016, LNCS 9696, pp. 97–116 (2016)
20. Morita, H., Schuldt, J.C.N., Matsuda, T., Hanaoka, G., Iwata, T.: On the security of the Schnorr signature scheme and DSA against related-key attacks. In: ICISC 2015, LNCS 9558, pp. 20–35 (2016)
21. Morita, H., Schuldt, J.C.N., Matsuda, T., Hanaoka, G., Iwata, T.: On the security of the Schnorr signatures, DSA, and ElGamal signatures against related-key attacks. IEICE Trans. **E100–A**(1), 73–90 (2017)
22. Murakami, T., Ohki, T., Takahashi, K.: Optimal sequential fusion for multibiometric cryptosystems. Inf. Fusion **32**, 93–108 (2016)
23. Pappu, R., Recht, B., Taylor, J., Gershenfeld, N.: Physical one-way functions. Science **297**(5589), 2026–2030 (2002)
24. Pohlig, S.C., Hellman, M.E.: An improved algorithm for computing logarithms over $gf(p)$ and its cryptographic significance (corresp.). IEEE Trans. Inf. Theory **24**(1), 106–110 (1978)
25. Pointcheval, D., Stern, J.: Security proofs for signature schemes. In: EUROCRYPT 1996, LNCS 1992, pp. 387–398 (1996)
26. Pollard, J.M.: Monte Carlo methods for index computation (mod p). Math. Comput. **32**(143), 918–924 (1978)
27. Raghavendra, R., Raja, K.B., Surbiryala, J., Busch, C.: A low-cost multimodal biometric sensor to capture finger vein and fingerprint. In: IJCB 2014, pp. 1–7 (2014)
28. Ross, A., Nandakumar, K., Jain, A.K.: Handbook of Multibiometrics. Springer, Berlin (2006)
29. Scheirer, W.J., Bishop, B., Boult, T.E.: Beyond PKI: the biocryptographic key infrastructure. In: WIFS 2010, pp. 1–6 (2010)
30. Schneier, B.: Applied Cryptography. Wiley, New York (1995)
31. Schnorr, C.P.: Efficient signature generation for smart cards. In: CRYPTO 1989, LNCS 435, pp. 239–252 (1990)
32. Simoens, K., Yang, B., Zhou, X., Beato, F., Busch, C., Newton, E., Preneel, B.: Criteria towards metrics for benchmarking template protection algorithms. In: ICB 2012, pp. 498–505 (2012)
33. Takahashi, K., Matsuda, T., Murakami, T., Hanaoka, G., Nishigaki, M.: A signature scheme with a fuzzy private key. In: ACNS 2015. LNCS 9092, pp. 105–126 (2015)
34. Wang, C., Chen, W., Liu, Y.: A fuzzy identity based signature scheme. In: EBISS 2009, pp. 1–5 (2009)
35. Wang, C., Kim, J.-H.: Two constructions of fuzzy identity based signature. In: BMEI 2009, pp. 1–5 (2009)
36. Waters, B.: Efficient identity-based encryption without random oracles. In: EUROCRYPT 2005, LNCS 3494, pp. 114–127 (2005)
37. Wu, Q.: Fuzzy biometric identity-based signature in the standard model. J. Comput. Inf. Syst. **8**(20), 8405–8412 (2012)
38. Yang, P., Cao, Z., Dong, X.: Fuzzy identity based signature with applications to biometric authentication. Comput. Electr. Eng. **37**(4), 532–540 (2011)
39. Yasuda, M., Shimoyama, T., Takenaka, M., Abe, N., Yamada, S., Yamaguchi, J.: Recovering attacks against linear sketch in fuzzy signature schemes of ACNS 2015 and 2016. In: ISPEC 2017, LNCS 10701, pp. 409–421 (2017)