IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received May 11, 2020, accepted June 4, 2020, date of publication June 19, 2020, date of current version July 8, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3003685

Leveled Adaptively Strong-Unforgeable Identity-Based Fully Homomorphic Signatures

CAIFEN WANG[1,2], BIN WU[1], AND HAILONG YAO[1]
[1]College of Mathematics and Statistics, Northwest Normal University, Lanzhou 730070, China
[2]College of Big Data and Internet, Shenzhen Technology University, Shenzhen 518118, China

Corresponding author: Caifen Wang (soloren@yeah.net)

This work was supported by the National Natural Science Foundation of China under Grant 61562077.

ABSTRACT The identity-based homomorphic signature (IBHS) enables an untrusted server to run some computation over the outsourced data and derive a short signature, vouching for the correctness of the output of the computation, while greatly simplifying key management. To our knowledge, constructions of IBHS have been few and far between. However, the existing IBHS schemes, which either handle only linear functions or has a large public key parameter and satisfies only the artificial notion of selective security. In this work, we construct the first leveled adaptively secure identity-based fully homomorphic signature (IBFHS) schemes without additional public parameters, which can be used to sign many different datasets. Thereby positively answering the open question of constructing a leveled IBFHS scheme with short public parameters, proposed by Wang *et al.*, (ISC, 2015, Springer). We achieve the stronger security and better parameters by using the trapdoor vanishing and vector encoding technique. In our scheme, the size of every evaluated signature depends only logarithmically on the size of the input dataset, and the complexity of verifying a signature for a computation can be amortized when verifying the same computation on many different datasets. Furthermore, we prove that our construction is strongly-unforgeable against adaptively chosen identity and message attacks under the small integer solution (SIS) assumption in standard lattices.

INDEX TERMS Homomorphic signature, identity-based cryptography system, adaptive security, small integer solution.

I. INTRODUCTION

Motivated by the advances in cloud computing, an increasing number of users outsource digital data and computations to servers in the cloud. As an example, consider an application scenario where medical data are collected by some hospitals, stored and processed on remote cloud servers, and finally consumed by other users (e.g., other hospitals or medical researchers) on some devices. This computing paradigm is very attractive; however, one may be concerned about security: although users who collect and consume sensitive data may trust each other, trusting the cloud may be problematic for various reasons. More specifically, two main security issues that need to be addressed are the privacy and authenticity of data stored and calculated in an untrusted environment. While it is well known that fully homomorphic encryption (FHE) [1]–[3] enables us to compute over encrypted data, paving the way for achieving privacy in outsourcing, in this work, we focus on the dual problem of providing data authenticity during computation. Many flavors of verifiable outsourcing schemes have been proposed to deal with this problem (cf. [4]–[7]). A particularly natural method of verifiable outsourcing computing is through the concept of homomorphic signatures [8], [9].

In a homomorphic signature scheme, Alice has a number of datasets, each of which consists of l data entries $u_1, \cdots, u_l$. For each dataset, Alice uses her signing key to compute l "root" signatures $\sigma_1 \cdots \sigma_l$ and outsource both the dataset and the corresponding signatures to a remote server. Later, Alice asks the server to compute some circuit f on the specified dataset $\boldsymbol{u}$. In the meantime or after that, the server computes the function and produces a derived signature σ for the result $\mu = f(\boldsymbol{u})$ of applying f to the dataset $\boldsymbol{u}$. Given only the public key and signature σ on the circuit f and a message μ, anyone can verify that the signature μ is indeed the result of applying f to some set of signed messages $\boldsymbol{u}$. To bind the signature

The associate editor coordinating the review of this manuscript and approving it for publication was Luca Barletta.

to a specific dataset, we "tag" each dataset of messages and provide the tag to the verification algorithm. A key feature of this verification is that it can be done without knowing the original dataset $\boldsymbol{u}$. The signature σ "proves" that the server did not tamper with the results of the computation, in the sense that computing a signature σ for any pair (f, μ), where $\mu = f(\boldsymbol{u})$, is hard for any probabilistic polynomial time (PPT) adversary.

The identity-based signature (IBS), first introduced by Shamir [10], enables any pair of users to verify each other's signatures without exchanging public or private keys. In particular, the public key of IBS users can be any string, such as their email address, thus greatly simplifying the key management process of the certificate-based public key infrastructure (PKI). Naturally, constructing an IBS with a homomorphism is interesting. However, to the best of our knowledge, constructions of identity-based homomorphic signatures (IBHS) have been few and far between. In particular, a major drawback of the known IBHS scheme is that they either handle only linear functions [11]–[14] or they satisfy only the notion of selective security [15], where an adversary announces the identity it will attack at the beginning of the security experiment, or they have the size of large public key parameter that is linearly related to the size of the dataset [15]. In this paper, our goal is to overcome these limitations and construct an identity-based full homomorphic signature (IBFHS) for all circuits, which is strongly unforgeable against adaptively chosen identity and dataset attack, based on the small integer solution (SIS) problem on arbitrary lattices.

A. PREVIOUS WORK

In 2015, Gorbunov *et al.* [16] introduced a new powerful primitive called the homomorphic trapdoor functions (HTDF) which was constructed by using the technology developed in the work of fully homomorphic and attribute-based encryptions [17], [18], conceptually unifying homomorphic signatures and encryption. They then utilized HTDF as a basic building block to construct the first leveled FHS for evaluating arbitrary circuits based on the SIS problem. In [16], the size of the evaluated signature grows polynomially at the maximum depth d of the circuit, so does our leveled IBFHS scheme in this paper. Though they further convert the basic FHS scheme that only satisfies selective security into a scheme that satisfies adaptive security by using the generic complexity leveraging technique, this approach is highly inefficient: with complexity leveraging, the security reduction would suffer a loss that is exponential in both the number of messages allowable in each dataset and the length of each message.

Subsequently, Wang *et al.* [15] first extended the notion of HTDF, the underlying primitive of FHS in [16], to identity-based setting, and obtained the identity-based HTDF (IBHTDF) which has better parameters and stronger security. The maximum noise compared with Gorbunov *et al.*'s HTDF is roughly reduced from $O(m^d\beta)$ to $O(4^d m\beta)$, which will lead to a polynomial modulus $q = ploy(\lambda)$ when $d = O(\log\lambda)$, where d is the maximum depth of the circuit and λ is the security parameter. The stronger security requires that the IBHTDF is not only claw-free, but also collision-resistant. They then defined and constructed the first leveled strongly-unforgeable IBFHS schemes. Unfortunately, this IBFHS scheme has large public parameters, with a size that exceeds the maximum size of the dataset to be signed, and that can only be used to sign a single dataset. Assuming the hardness of the SIS problem, the leveled IBFHS scheme is only secure in the selective security model, where the adversary has to specify a target identity to be attacked and message data to be signed prior to seeing the master public key and public parameters. In addition, unlike the FHS scheme in [16], the IBFHS scheme cannot be converted from selective security to adaptive security by the generic complexity leveraging technique.

B. OUR CONTRIBUTIONS

Our starting point is Wang *et al.*'s selectively secure leveled IBFHS scheme [15]. As stated above, although the authors in [15] extend Gorbunov *et al.*'s leveled FHS scheme to the identity-based setting with better parameters and stronger security, there are some drawbacks in terms of the security level, size of common parameters, functionality, and efficiency in generating signatures. In this paper, we construct a leveled IBFHS scheme based on the SIS problem on arbitrary lattices, which is strongly-unforgeable against adaptively chosen identity and message attack. We describe our results in more detail below.

- We construct an efficient leveled strongly-unforgeable IBFHS scheme without additional public parameters, which can be used to sign many different datasets, thereby positively answering the open question of constructing a leveled IBFHS scheme with short public parameters in [15].
- We prove that our leveled IBFHS scheme is adaptively secure against chosen identity and message attacks under the small integer solution (SIS) assumption in standard lattices.
- The signatures in our leveled IBFHS scheme are succinct in the sense that the size of every evaluated signature depends only logarithmically on the size of the input dataset.

In addition, as done in [15], we can also use Barrington's theorem [19] to convert the Boolean circuit involved in our IBFHS scheme into a polynomial length, width-5 permutation branch program, which can greatly reduce the size of the evaluated signature from $m^{O(d_{max})}$ to $O(m \cdot 4^{d_{max}})$ for circuits with a maximum depth d_{max}. This process results in a smaller module q, so the efficiency and security of our IBFHS scheme can be further improved. Our method of using Barrington's theorem to optimize the scheme is the same as that used by Wang *et al.* [15], so it will not be presented herein. Regarding verification efficiency, our leveled IBFHS scheme is almost the same as Gorbunov and Luo [20], and

it also allows fast amortized verification of a circuit f on many different datasets. Specifically, the calculation of the verification key related to circuit f can be performed offline before receiving the signature, and can be amortized when verifying the same circuit on multiple datasets, so the online verification can be more efficient than computing f.

C. OVERVIEW OF OUR CONSTRUCTION

We construct the IBFHS scheme by relying on the techniques developed by Apon *et al.* [21] and Luo *et al.* [20] in the contexts of the identity-based encryption and fully homomorphic signature. We will first briefly review the key technologies involved in [21] that we will use.

In [21], the author constructed an IBE scheme based on the partition function $\boldsymbol{h}_{z,\boldsymbol{\alpha},\boldsymbol{\beta}} : \{0,1\}^n \to \mathbb{Z}_q^v$, with its associated evaluating algorithm $(\mathsf{PubEval}_{||}^{(\cdot,v,t)}, \mathsf{TrapEval}_{||}^{(\cdot,v,t)})$. Hash function $\boldsymbol{h}_{z,\boldsymbol{\alpha},\boldsymbol{\beta}}$ is required to realize the trapdoor vanishing technique discussed below. This hash function isolates the challenge identity from all the other identity queries. That is, given a set of identity queries Q and a challenge identity $\boldsymbol{a}^* \notin Q$, this hash function separates them with a noticeable probability, i.e., $\boldsymbol{h}_{z,\boldsymbol{\alpha},\boldsymbol{\beta}}(\boldsymbol{a}^*) = 0$, but $\boldsymbol{h}_{z,\boldsymbol{\alpha},\boldsymbol{\beta}}(\boldsymbol{a}) \neq 0$ for all $\boldsymbol{a} \in Q$. $\mathsf{PubEval}_{||}^{(\cdot,v,t)}(\mathbf{B}, \mathbf{B}', \boldsymbol{h}_{z,\boldsymbol{\alpha},\boldsymbol{\beta}})$ is a homomorphic evaluation algorithm for the hash function $\boldsymbol{h}_{z,\boldsymbol{\alpha},\boldsymbol{\beta}}$, which is responsible for encoding an identity $\boldsymbol{a}$ into an identity-specific lattice

$$[\mathbf{A}|\mathbf{Y}] = [\mathbf{A}|\mathsf{PubEval}_{||}^{(\boldsymbol{a},v,t)}(\mathbf{B}, \mathbf{B}', \boldsymbol{h}_{z,\boldsymbol{\alpha},\boldsymbol{\beta}})]$$

by a short public key $(\mathbf{A}, \mathbf{B}, \mathbf{B}')$. In particular, the ability to use a short public key matrix $(\mathbf{A}, \mathbf{B}, \mathbf{B}')$ to encode identity benefits from the vector coding method proposed by the author [21].

In the security proof, the simulator replaces $\mathbf{B}$ and $\mathbf{B}'$ with other matrices $\mathbf{AR} + \mathbf{E}$ and $\mathbf{AR}' + \mathbf{E}'$ for "short" matrices $\mathbf{R}, \mathbf{R}'$ respectively, the random "polluter" $\mathbf{E}, \mathbf{E}'$. This identity coding mechanism with the embedded hash function $\boldsymbol{h}_{z,\boldsymbol{\alpha},\boldsymbol{\beta}}$ makes the identity-specific matrix for $\boldsymbol{a}$, i.e., $[\mathbf{A}|\mathbf{Y}]$, become

$$[\mathbf{A}|\mathbf{Y}_{proof}] = [\mathbf{A}|\mathbf{AR}_{\boldsymbol{a}} + \mathbf{E}_{\boldsymbol{a}}\mathbf{G}_{vn}]$$

where matrix $\mathbf{R}_{\boldsymbol{a}}$ can be computed by running the algorithm $\mathsf{TrapEval}_{||}^{(\boldsymbol{a},v,t)}((z, \boldsymbol{\alpha}, \boldsymbol{\beta}), \mathbf{A}, \mathbf{R}, \mathbf{R}', \boldsymbol{h})$, $\mathbf{E}_{\boldsymbol{a}} = [h_{z_1,\alpha_1,\beta_1}(\boldsymbol{a})\mathbf{I}_n| \cdots |h_{z_u,\alpha_u,\beta_u}(\boldsymbol{a})\mathbf{I}_n|\mathbf{0}_n| \cdots |\mathbf{0}_n]$. The above system contains two types of lattices, denoted by $\mathbf{A}$ and $\mathbf{G}$, respectively, corresponding to the real and simulated system. Each of these two systems is associated with a different trapdoor. The trapdoor $\mathbf{T}_{\mathbf{A}}$ serves as the true master secret msk in the real system. This trapdoor is a "fully functional" trapdoor, which enables the generation of secret keys for every identity $\boldsymbol{a}$ allowed by the system. In contrast, the trapdoor $\mathbf{T}_{\mathbf{G}}$ is a "semifunctional" trapdoor that is only used for the security proof, which enables generating secret keys for every identity $\boldsymbol{a}$, except some adaptively chosen challenge identity $\boldsymbol{a}^*$. For the challenge identity $\boldsymbol{a}^*$ and adaptively queried identities $\{\boldsymbol{a}_1, \cdots, \boldsymbol{a}_{|Q|}\}$, the security reduction will proceed whenever $\boldsymbol{h}_{z,\boldsymbol{\alpha},\boldsymbol{\beta}}(\boldsymbol{a}^*) = 0$ and for $i \in [|Q|]$, $\boldsymbol{h}_{z,\boldsymbol{\alpha},\boldsymbol{\beta}}(\boldsymbol{a}_i) \neq 0$. This is due to the fact that the gadget matrix $\mathbf{G}$ survives in the identity-specific lattice for all of $\{\boldsymbol{a}_1, \cdots, \boldsymbol{a}_{|Q|}\}$, but the matrix $\mathbf{G}$ vanishes on the identity-specific lattice for the challenge identity $\boldsymbol{a}^*$, and therefore, the security reduction can $\mathsf{SampleD}$ with $\mathbf{T}_{\mathbf{G}}$ for every identity in the adversary's view, except the challenge identity $\boldsymbol{a}^*$.

Intuitively, to construct a secure IBFHS scheme adaptively under the chosen identity and message attack, we need to use the homomorphic evaluation algorithm $\mathsf{PubEval}$ to encode both identity id and dataset tag $\boldsymbol{\tau}$. Specifically, in our IBFHS, the method of answering the secret key of the identity chosen adaptively is the same as the method used by Apon *et al.* to construct the IBE. Briefly, we call algorithm $\mathsf{PubEval}_{||}^{(\mathsf{id},v,t)}(\mathbf{B}, \mathbf{B}', \boldsymbol{h}_{z,\boldsymbol{\alpha},\boldsymbol{\beta}})$ to generate identity matrix $\mathbf{B}_{\mathsf{id}}$, which satisfies $\mathbf{B}_{\mathsf{id}} = \mathbf{AR}_{\mathsf{id}} + \mathbf{E}_{\mathsf{id}}\mathbf{G}_{vn}$. In terms of adaptively chosen message attack, we introduce a standard identity-based signature scheme (IBS) S (non-homomorphic) to enable the simulator to answer the adversary's signing queries adaptively. The idea is that for every new identity id, we generate the random matrices $\mathbf{C}_{\mathsf{id}}, \mathbf{C}'_{\mathsf{id}}$ associated with the identity, which is needed to implement the algorithm $\mathsf{PubEval}$ and then use the standard identity-based signature scheme S to sign the identity id, together with the generated public key parameters. The signing queries must contain a tag $\boldsymbol{\tau}$ because the message is assigned to different datasets identified by tags. Then, for any identity id, we run the algorithm $\mathsf{PubEval}_{||}^{(\boldsymbol{\tau},v,t)}(\mathbf{C}_{\mathsf{id}}, \mathbf{C}'_{\mathsf{id}}, \boldsymbol{h}_{z_{\mathsf{id}},\boldsymbol{\alpha}_{\mathsf{id}},\boldsymbol{\beta}_{\mathsf{id}}})$ with the tag $\boldsymbol{\tau}$ as the input to generate the "identity-label" matrix $\mathbf{C}_{\mathsf{id},\boldsymbol{\tau}}$, which satisfies $\mathbf{C}_{\mathsf{id},\boldsymbol{\tau}} = \mathbf{A}_{\mathsf{id}}\mathbf{R}_{\mathsf{id},\boldsymbol{\tau}} + \mathbf{E}_{\mathsf{id},\boldsymbol{\tau}}\mathbf{G}_{vn}$, where $\mathbf{E}_{\mathsf{id},\boldsymbol{\tau}} = [h_{z_{\mathsf{id},1},\alpha_{\mathsf{id},1},\beta_{\mathsf{id},1}}(\boldsymbol{\tau})\mathbf{I}_n| \cdots |h_{z_{\mathsf{id},u},\alpha_{\mathsf{id},u},\beta_{\mathsf{id},u}}(\boldsymbol{\tau})\mathbf{I}_n|\mathbf{0}_n| \cdots |\mathbf{0}_n]$, and the matrix $\mathbf{R}_{\mathsf{id},\boldsymbol{\tau}}$ can be computed by running the algorithm $\mathsf{TrapEval}$. These matrices $\mathbf{B}_{\mathsf{id}}, \mathbf{C}_{\mathsf{id},\boldsymbol{\tau}}$ play an important role in helping the simulator answer the secret key and signing queries, and solve the SIS problem by using the forgery of the output in the security proof. For challenge identity id^*, tag $\boldsymbol{\tau}^*$ and adaptively queried identities $\{\mathsf{id}_1, \cdots, \mathsf{id}_{|Q_1|}\}$, tags $\{\boldsymbol{\tau}_1, \cdots, \boldsymbol{\tau}_{|Q_2|}\}$, the simulator can successfully answer the secret key and signing queries and solve SIS problems, as long as $\boldsymbol{h}_{z,\boldsymbol{\alpha},\boldsymbol{\beta}}(\mathsf{id}^*) = 0, \boldsymbol{h}_{z,\boldsymbol{\alpha},\boldsymbol{\beta}}(\mathsf{id}_i) \neq 0$ for $i \in [|Q_1|]$ and $\boldsymbol{h}_{z_{\mathsf{id}},\boldsymbol{\alpha}_{\mathsf{id}},\boldsymbol{\beta}_{\mathsf{id}}}(\boldsymbol{\tau}^*) = 0, \boldsymbol{h}_{z_{\mathsf{id}},\boldsymbol{\alpha}_{\mathsf{id}},\boldsymbol{\beta}_{\mathsf{id}}}(\boldsymbol{\tau}_i) \neq 0$ for $i \in [|Q_2|]$, is non-negligible, because this good probability ensures that the trapdoor vanishing technique can be successfully implemented under the adaptively chosen identity and message attack. Moreover, in the signature algorithm of our IBFHS scheme, we adopt the method proposed by Luo *et al.* [20] to improve the efficiency of signature generation, which only needs to call the signing algorithm twice to generate signatures for all messages in the dataset.

D. RELATED WORKS

The HS scheme was initially designed for authentication in networking coding to deal with pollution attacks [22]. Johnson *et al.* [23] first introduced the precise framework and formal definition of homomorphic signatures. Since then, homomorphic authenticators (HAs) (such as signatures and message authentication codes) have been very active research

fields. We briefly reviewed some of the salient results in this search.

1) RESTRICTED HOMOMORPHIC SIGNATURES

Many prior works consider the question of homomorphic signatures for restricted homomorphisms, and almost exclusively for linear functions [24]–[26]. Such signature schemes have interesting applications in network coding [27]–[29], and proofs of retrievability [30]. Boneh and Freeman [31] first consider the homomorphic signature beyond linear functions, which can compute constant degree polynomials on signed messages, and its security was based on the hardness of the ring SIS problem in the random oracle model. Catalano *et al.* [32] present an alternate scheme using multi-linear maps in the standard model at the cost of having large public parameters.

2) (LEVELED) HOMOMORPHIC SIGNATURES FROM LATTICES

Gorbunov *et al.* [16] proposed the first (leveled) FHS scheme that can evaluate arbitrary polynomial-depth circuits over signed data. As an extension of the leveled FHS in [16], Wang *et al.* [15] constructed the first leveled strongly-unforgeable IBFHS schemes, thereby extending the leveled FHS to identity-based settings with stronger security and better parameters. Two FHS schemes [33], [34] were proposed in 2014, both of which were are derived from the key-homomorphic functional encryption scheme for circuits of [18]. These two schemes are secure only in the selective security model, in which the adversary must announce in advance the message whose signature it intends to forge. Although the two schemes can be converted from selective security to adaptive security using the generic complexity leveraging technique [35], this occurs at the cost of inefficiency. In contrast, Boyen *et al.* [36] proposed a different approach to adaptively secure FHS, that is, making use of a new technique for improving the efficiency and applicability of partitioning type proofs [37]. The authors in [36] only improved the security of Gorbunov *et al.*'s FHS scheme, but not its efficiency. In [20], the author improved the efficiency of generating signatures in a FHS scheme while still achieving fully adaptive security. Specifically, an FHS scheme using vector coding technology, which generates signatures for all messages in the dataset and only needs to run the signature algorithm twice, was proposed.

3) SYMMETRIC-KEY HOMOMORPHIC MACs

Some important advances have been achieved in constructing homomorphic message authentication (MAC) for various homomorphic classes [38]–[40]. Unlike signatures, MAC only allows for the private verification based on the symmetric key. Initially, Agrawal and Boneh [41] carried out this work focusing on network coding applications. Gennaro and Wichs [42] proposed a fully homomorphic MAC based on FHE; however, in the presence of verification queries, their scheme is not secure. Later, a more effective scheme [43] was proposed, which is secure in the presence of verification queries at the cost of only supporting restrictively homomorphic functions.

4) VERIFIABLE COMPUTATION

The achievement of outsourcing calculation correctness is also the purpose of verifiable delegation of computation (VC) [44], [45]. In this scenario, VC schemes delegate the computation of a function f on some input x to an untrusted cloud server. In addition, the server can generate a succinct proof such that the client can verify the correctness of the output computation result without having to perform the computation itself. The main difference between HAs and VC is that the verifier in VC must know the input of the computation; in fact, x may be an enormous dataset, while in HAs, verification can be performed without knowing and storing x.

5) OTHER TYPES OF HOMOMORPHIC AUTHENTICATION

The abovediscussed notion of HAs only supports computations performed on data authenticated by a single user. In many realistic scenarios (ubiquitous computing, a distributed network of sensors, etc.), large datasets contain data provided by multiple users, which has led researchers to study multi-key HAs [46]–[49]. In short, multi-key HAs are similar to HAs with the additional functions that allow for the computation of data authenticated under different secret keys.

Quantum-based protocols have been applied to HS schemes to deal with quantum network environments. Shang *et al.* [50] regarded entanglement swapping as a homomorphic operation and creatively proposed the first quantum HS scheme, which only allows a verifier to verify the signature once. To overcome this limitation, Shang *et al.* [51] proposed a repeatable quantum HS scheme by using a parallel validation model and serial verification model. Li *et al.* [52] proposed two quantum MAC schemes based on the quantum circuit, which can protect against the pollution attack launched by untrusted internal nodes over a general quantum network.

E. ORGANIZATION

The remainder of this paper is organized as follows: In section II, we present our preliminary findings, including some basic related concepts and complex assumptions. In section III, we introduce the notion of the IBFHS scheme and its security model. We construct a concrete IBFHS scheme and prove its security under the standard model in section IV and section V, respectively. Finally, we conclude the paper in section VI.

II. PRELIMINARIES

Let PPT denote the probabilistic polynomial time. For an integer $q \geq 2$, we represent $\mathbb{Z}_q$ as integers in $\left(-\frac{q}{2}, \frac{q}{2}\right]$. We use bold uppercase letters (e.g., $\mathbf{A}$) to denote matrices and bold lowercase letters (e.g., $\mathbf{x}$) to denote vectors. The $\mathbf{A}^\top$ denotes the transposition of the matrix $\mathbf{A}$ for a positive

integer n, $[n]$ denotes the set $\{1, \cdots, n\}$, and $|\boldsymbol{\alpha}|$ denotes the number of elements of vector or the number of bits in a string $\boldsymbol{\alpha}$. In addition, We use $[\mathbf{A_1}|\mathbf{A_2}]$ to denote the concatenation of matrices $\mathbf{A_1}$ and $\mathbf{A_2}$, $\tilde{\mathbf{A}}$ to denote the Gram-Schmidt orthogonalization of matrix $\mathbf{A}$ and $\|\mathbf{A}\|$ to denote the norm of matrix $\mathbf{A} \in \mathbb{R}^{m\times k}$. The $\mathbf{A} \xleftarrow{\$} \mathbb{Z}_q^{n\times m_0}$ denotes the process of uniformly randomly choosing matrix $\mathbf{A}$ from $\mathbb{Z}_q^{n\times m}$, and $\mathbf{R} \sim \mathcal{D}_{\mathbb{Z}^{m\times m},s}$ denotes $\mathbf{R}$ being sampled from $\mathcal{D}_{\mathbb{Z}^{m\times m},s}$.

A. BACKGROUND ON LATTICES AND HARD PROBLEMS

1) LATTICES

Let $\mathbf{X} = \{\mathbf{x}_1, \cdots, \mathbf{x}_n\} \subset \mathbb{Z}^n$ consist of n linearly independent vectors, the full-rank n-dimensional integer lattice $\Lambda \subset \mathbb{Z}^n$, generated by the basis $\mathbf{X}$ is

$$\Lambda = \mathcal{L}(\mathbf{X}) = \{\mathbf{Xc} = \sum\nolimits_{i\in[n]} c_i \cdot \mathbf{x}_i : \mathbf{c} \in \mathbb{Z}^n\}.$$

which is a discrete additive subgroup.

For a matrix $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$, the "q-ary" integer lattices are described as:

$$\Lambda_q^{\perp}(\mathbf{A}) = \{\boldsymbol{e} \in \mathbb{Z}^m | \mathbf{A}\boldsymbol{e} = 0 \mod q\},$$
$$\Lambda_q^{\mathbf{u}}(\mathbf{A}) = \{\boldsymbol{e} \in \mathbb{Z}^m | \mathbf{A}\boldsymbol{e} = \mathbf{u} \mod q\}.$$

It is obvious that $\Lambda_q^{\mathbf{u}}(\mathbf{A})$ is a coset of $\Lambda_q^{\perp}(\mathbf{A})$.

For any positive parameter $s \in \mathbb{R}$, define the Gaussian function on $\mathbb{R}^n$ with center $\mathbf{c}$ and parameter s: $\rho_{s,\mathbf{c}}(\mathbf{x}) = \exp(-\pi\|\mathbf{x}-\mathbf{c}\|^2/s^2)$. The subscripts $\mathbf{c}$ and s are taken to be $\mathbf{0}$ and 1(respectively) when omitted.

For any $\mathbf{c} \in \mathbb{R}^n$, positive parameter s, and n-dimensional lattice Λ, we set $\rho_{s,\mathbf{c}}(\Lambda) = \sum_{\mathbf{x}\in\Lambda} \rho_{s,\mathbf{c}}(\mathbf{x})$ and define the discrete Gaussian distribution over Λ as: $\forall \mathbf{x} \in \Lambda$, $\mathcal{D}_{\Lambda,s,\mathbf{c}}(\mathbf{x}) = \dfrac{\rho_{s,\mathbf{c}}(\mathbf{x})}{\rho_{s,\mathbf{c}}(\Lambda)}$.

2) NORM OF A RANDOM MATRIX

Let S^k denote the set of vectors in $\mathbb{R}^k$ of length 1. The norm of a matrix $\mathbf{A} \in \mathbb{R}^{m\times k}$ is defined to be $\mathsf{sup}_{\mathbf{x}\in S^k}\|\mathbf{Ax}\|$.

The following lemma bounds the norm for some specified distributions.

Lemma 1:([53])**.** Regarding the norm and the Gaussian distribution defined above, we have the following bounds:

- Let $\mathbf{R} \in \{-1, 1\}^{m\times m}$ be chosen at random, the $\Pr[\|\mathbf{R}\| > 12\sqrt{2m}] < e^{-2m}$.
- Let $\mathbf{R} \sim \mathcal{D}_{\mathbb{Z}^{m\times m},s}$, then we have $\Pr[\|\mathbf{R}\| \geq s\sqrt{m}] < e^{-2m}$.

Lemma 2: ([54], [55])**.**

- Let n, q be positive integers with q prime, and $m \geq 2n\log q$. Then for any $s \geq \omega\sqrt{\log m}$ and for all but a $2q^{-n}$ fraction of all $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$ we have: for $\mathbf{R} \sim \mathcal{D}_{\mathbb{Z}^{m\times m},s}$, the distribution $\mathbf{AR} \mod q$ is statistically close to uniform over $\mathbb{Z}_q^{n\times m}$.
- Let $\mathbf{R} \sim \mathcal{D}_{\mathbb{Z}^{m\times m},s}$, then for any matrix $\mathbf{U}$ chosen uniformly at random from $\{-1, 1\}^{m\times m}$, we have the distribution $\mathbf{UR}$ is statistically close to $\mathcal{D}_{\mathbb{Z}^{m\times m},s\sqrt{m}}$.

Lemma 3:([53])**.** Suppose that $m > (n+1)\log q + \omega(\log n)$. Let $\mathbf{U} \in \{-1, 1\}^{m\times k}$ be chosen uniformly at random for some polynomial $k = k(n)$. Let $\mathbf{A}, \mathbf{B}$ be matrix chosen uniformly from $\mathbb{Z}_q^{n\times m}$, $\mathbb{Z}_q^{n\times k}$ respectively. Then, for all vectors $\boldsymbol{\omega} \in \mathbb{Z}^m$, the two distributions are statistically close: $(\mathbf{A}, \mathbf{AU}, \mathbf{U}^{\top}\boldsymbol{\omega}) \approx (\mathbf{A}, \mathbf{B}, \mathbf{U}^{\top}\boldsymbol{\omega})$.

3) SMALL INTEGER SOLUTION

The average-case $\mathsf{SIS}_{q,n,m,\beta}$ problem is as hard as approximating certain problems (e.g., GapSVP and SIVP) in the worst case to within ploy(n) factors [56].

Definition 4: (SIS)**.** For any $n \in \mathbb{Z}$, and any functions $q(n)$, $m(n)$ and $\beta(n)$, the average-case Small Integer Solution problem ($\mathsf{SIS}_{q,m,n,\beta}$) is as follows: Given an integer q, a real $\beta \in \mathbb{R}$, a matrix $\mathbf{A} \in \mathbb{Z}^{n\times m}$ chosen uniformly at random, find a integer vector $\boldsymbol{e} \in \mathbb{Z}^m \setminus \{\mathbf{0}\}$, such that $\mathbf{A}\boldsymbol{e} = 0 \mod q$ and $\|\boldsymbol{e}\| \leq \beta$.

B. TRAPDOOR FOR LATTICES AND SAMPLING ALGORITHMS

Lemma 5:([54]) **.** Let n, m, q be positive integers with $q \geq 2$ and $m \geq 6n\log q$. There exists a PPT algorithm $\mathsf{TrapGen}(n, m, q)$ that outputs a pair ($\mathbf{A} \in \mathbb{Z}_q^{n\times m}, \mathbf{T_A} \in \mathbb{Z}_q^{m\times m}$) such that $\mathbf{A}$ is statistically close to uniform over $\mathbb{Z}_q^{n\times m}$ and $\mathbf{T_A}$ is a basis for $\Lambda_q^{\perp}(\mathbf{A})$ satisfying $\|\mathbf{T_A}\| \leq O(n\log q)$.

Definition 6: (Gadget matrix [55])**.** Let $m = n \cdot \lceil\log q\rceil$, the gadget matrix is defined as: $\mathbf{G}_n = \mathbf{g} \otimes \mathbf{I}_n \in \mathbb{Z}_q^{n\times m}$, where vector $\mathbf{g} = (1, 2, 4, \cdots, 2^{\lceil\log q\rceil}) \in \mathbb{Z}_q^{\lceil\log q\rceil}$. The inverse function is difined as $\mathbf{G}_n^{-1} : \mathbb{Z}_q^{n\times m} \to \{0, 1\}^{m\times m}$ which expands each entry $x \in \mathbb{Z}_q$ of the input matrix into a column consisting of the bits of binary representations. And for any $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$, it holds that $\mathbf{G}_n \cdot \mathbf{G}_n^{-1}(\mathbf{A}) = \mathbf{A}$.

Moreover, by padding zero, $\mathbf{G}_n \in \mathbb{Z}_q^{n\times m}$ can be padded into matrix $\bar{\mathbf{G}}_n \in \mathbb{Z}_q^{n\times m'}$ for $m' > m$.

Definition 7: (Generalized **G**-trapdoor [AFL17])**.** Let $\mathbf{A} \in \mathbb{Z}_q^{n\times 2m}$ and $\mathbf{G}_{\nu n} \in \mathbb{Z}^{\nu n\times m}$ be matrices with integer $\nu \geq 1$ and $m \geq n$. A $\mathbf{G}_{\nu n}$-trapdoor for $\mathbf{A}$ is a matrix $\mathbf{R} \in \mathbb{Z}^{m\times m}$ such that $\mathbf{A}\begin{pmatrix}\mathbf{R}\\ \mathbf{I}_m\end{pmatrix} = \mathbf{HG}_{\nu n}$, for some full column rank matrix $\mathbf{H} \in \mathbb{Z}_q^{n\times \nu n}$. We refer to $\mathbf{H}$ as the tag of the trapdoor.

Lemma 8: ([55], Algorithm)**.** Let $n, q > 2, m_0 \geq 1, m \geq 2n\log q$ be integers, for $\mathbf{A}_0 \xleftarrow{\$} \mathbb{Z}_q^{n\times m_0}$ and $\mathbf{H} \in \mathbb{Z}_q^{n\times n}$, there exists an randomized algorithm $\mathsf{GenTrap}(\mathbf{A}_0, \mathbf{H})$ to generate a parity-check matrix $\mathbf{A} = [\mathbf{A}_0| - \mathbf{A}_0\mathbf{R} + \mathbf{HG}_n] \in \mathbb{Z}_q^{n\times m}$ with trapdoor $\mathbf{R}$ such that $\mathbf{R} \sim \mathcal{D}_{\mathbb{Z}^{m_0\times\omega}}$, and the distribution of $\mathbf{A}$ statistically close to uniform. Moreover, one can use the trapdoor $\mathbf{R}$ and any basis $\mathbf{T}_{\mathbf{G}_n}$ for $\Lambda_q^{\perp}(\mathbf{G}_n)$ to generate a basis $\mathbf{T_A}$ for lattices $\Lambda_q^{\perp}(\mathbf{A})$, and the basis $\mathbf{T_A}$ satisfies $\|\tilde{\mathbf{T}_A}\| \leq 3.8\sqrt{n\log q}$.

Since $\mathbf{G}_n$ is gadget matrix and its trapdoor is publicly know, as shown in [18], it is easy to compute a basis $\mathbf{T}_{\mathbf{G}_n}$ for $\Lambda_q^{\perp}(\mathbf{G}_n)$ that satisfies $\|\mathbf{T}_{\mathbf{G}_n}\| \leq \sqrt{5}$.

Lemma 9:([21], [53])**.** Let $q > 2, m > n$, there are two algorithms as follows:

- $\mathbf{R} \leftarrow \mathsf{SampleLeft}(\mathbf{A}, \mathbf{B}, \mathbf{T_A}, \mathbf{D}, s)$: Takes as input a full rank matrix $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$, a short basis $\mathbf{T_A}$ of lattice

$\Lambda_q^{\perp}(\mathbf{A})$, a matrix $\mathbf{B} \in \mathbb{Z}_q^{n\times m_1}$, a matrix $\mathbf{D} \in \mathbb{Z}_q^{n\times m}$ and a Gaussian parameter $s > \|\tilde{\mathbf{T}_A}\| \cdot \omega\sqrt{\log(m+m_1)}$, then outputs a matrix $\mathbf{R} \in \mathbb{Z}_q^{(m+m_1)\times m}$ distributed statistically close to $\mathcal{D}_{\Lambda_q^{\mathbf{D}}(\mathbf{F}),s}$, where $\mathbf{F} := [\mathbf{A}|\mathbf{B}]$.

- $\mathbf{R}^* \leftarrow \mathsf{SampleD}^{\mathcal{O}}(\mathbf{A}_0, \mathbf{R}, \mathbf{H}, \mathbf{D}, s)$: Takes as input a $\mathbf{G}_{vn}$-trapdoor $\mathbf{R}$ for a matrix $\mathbf{A}_0$ with full rank tag $\mathbf{H}$, a matrix $\mathbf{D} \in \mathbb{Z}_q^{n\times m}$ and an oracle $\mathcal{O}$ for Gaussian sampling over a desired coset $\Lambda_q^{\mathbf{D}}(\mathbf{G}_{vn})$. This efficient algorithm will output a matrix $\mathbf{R}^*$ drawn from a distribution within negligible statistical distance of $\mathcal{D}_{\Lambda^{\mathbf{D}}(\mathbf{A}),s}$, where $\mathbf{A} := [\mathbf{A}_0| - \mathbf{A}_0\mathbf{R} + \mathbf{H}\mathbf{G}_{vn}] \in \mathbb{Z}_q^{n\times 2m}$ and $s \geq n^{4+\varepsilon}m^{3.5}$.

C. ENCODING FOR VECTORS AND APPLICATION IN GSW-FHS SETTING

1) ENCODING FOR VECTORS

For vector $\boldsymbol{v} = (v_1, \cdots, v_d) \in \mathbb{Z}_q^d$, the encoding algorithm maps a d-dimensional vector to an $n \times m$ matrix as follows:

$$\mathsf{encode}(\boldsymbol{v}) = \mathbf{E}_{\boldsymbol{v}} = [v_1\mathbf{I}_n| \cdots |v_d\mathbf{I}_n] \cdot \mathbf{G}_{dn}.$$

packing and unpacking. For vector $\boldsymbol{v} = (v_1, \cdots, v_d) \in \mathbb{Z}_q^d$, let $\mathbf{E}_{\boldsymbol{v}}$ be the encoding of vector $\boldsymbol{v}$, and $\mathbf{E}_{v_i}$ be the encoding of component v_i, for $i \in [d]$, matrix $\mathbf{T}_i$ is defined as $dn \times n$ extended unit matrix: $\mathbf{T}_i = [\mathbf{0}_n| \cdots |\mathbf{0}_n|\mathbf{I}_n| \cdots |\mathbf{0}_n]$, then, there are two algorithms as follows:

- $\mathsf{Pack}(\{\mathbf{E}_{v_i}\}_{i=1}^d)$: On input d encodings $\mathbf{E}_{v_i}$, it outputs $\mathbf{E}_{\boldsymbol{v}} = \sum_{i\in[d]} \mathbf{E}_{v_i} \cdot \mathbf{G}_n^{-1}(\mathbf{T}_i \cdot \mathbf{G}_{dn}) = [v_1\mathbf{I}_n| \cdots |v_d\mathbf{I}_n] \cdot \mathbf{G}_{dn}$.
- $\mathsf{UnPack}(\mathbf{E}_v, i)$: On input an encodings $\mathbf{E}_v$ for a vector $\boldsymbol{v}$ and an index i, it outputs $\mathbf{E}_{v_i} = \mathbf{E}_{\boldsymbol{v}} \cdot \mathbf{G}_{dn}^{-1}(\mathbf{T}_i^{\top}\mathbf{G}_n) = v_i\mathbf{G}_n$.

2) RECURSIVE PACKING AND UNPACKING

The following two algorithms are the key step leading towards [20], [21] and also our leveled FHS scheme, which has a constant number of public key matrices. We only outline these two algorithms, omitting the calculation detail of the algorithms, interested readers can refer to the [21].

For vector $\boldsymbol{v} = (v_1, \cdots, v_d)$ in $[\xi]^d$ for some small $\xi = 2^v$ and $v = \omega(1)$, i.e. each element v_i can be decomposed into v bits, $v_{ij} \in \{0, 1\}, i \in [d], j \in [v]$. Let $\mathbf{E}_{\boldsymbol{v}}$ be the encoding of vector $\boldsymbol{v}$, and $\mathbf{E}_{v_{ij}} = v_{ij}\mathbf{G}_n$ be the encoding of v_{ij}. Then we have:

- $\mathsf{RecPack}(\{\mathbf{E}_{v_{ij}}\}_{i\in[d],j\in[v]})$: On input $d \cdot v$ encodings $\mathbf{E}_{v_{ij}}$, it outputs $\mathbf{E}_{\boldsymbol{v}} = [v_1\mathbf{I}_n| \cdots |v_d\mathbf{I}_n] \cdot \mathbf{G}_{dn}$.
- $\mathsf{RecUnPack}(\mathbf{E}_v, (i, j))$: On input an encodings $\mathbf{E}_{\boldsymbol{v}}$ and index (i, j), it outputs $\mathbf{E}_{v_{ij}}$.

3) APPLICATION IN GSW-FHS SETTING

Gentry *et al.* [57] proposed a novel homomorphic encryption scheme (GSW). Alperin-Sheriff and Peikert [55] simplified the original GSW scheme using the gadget matrix, in which the ciphertext expression is $\mathbf{C} = \mathbf{A}\mathbf{R} + \mu\mathbf{G}_n$, where μ is a message, $\mathsf{PK} = \mathbf{A} \in \mathbb{Z}_q^{n\times m}$ is the public key, $\mathbf{R} \xleftarrow{\$} \{0, 1\}^{m\times m}$.

Applying the new encode method to the GSW scheme such that a single matrix can encrypt a vector of d integers, e.g., the GSW ciphertext of message vector $\boldsymbol{v}$ is $\mathbf{C} = \mathbf{A}\mathbf{R} + \mathbf{E}_{\boldsymbol{v}}$, where $\mathbf{E}_{\boldsymbol{v}} = [v_1\mathbf{I}_n| \cdots |v_d\mathbf{I}_n] \cdot \mathbf{G}_{dn}$.

In particular, the (recursive) packing / unpacking operations can easily applied to the GSW ciphertexts, we show two of them, the other two are similar.

- $\mathsf{RecPack}(\{\mathbf{C}_{ij}\}_{i\in[d],j\in[v]})$: Recursively pack $d \cdot v$ GSW bit-encryptions to one packed ciphertexts $\mathbf{C}^*$.
- $\mathsf{RecUnPack}(\mathbf{C}^*, (i, j))$: Recurisively unpack a packed ciphertext $\mathbf{C}^*$ into $d \cdot v$ GSW bit-encryptions, output $\mathbf{C}_{ij}$ for $i \in [d], j \in [v]$.

The following concept bound the noise growth after homomorphic evaluating of GSW ciphertext for a function f.

Definition 10: ([21] δ-expanding evaluation) Deterministic algorithms ($\mathsf{PubEval}$, $\mathsf{TrapEval}$) are δ-expanding with a circuit $f : \mathcal{X}^u \to \mathcal{Y}$, if they satisfy the follow properties:

- $\mathsf{PubEval}(\{\mathbf{B_i} \in \mathbb{Z}_q^{n\times m}\}_{i\in[u]}, f)$: Takes as input matrices $\{\mathbf{B_i}\}_{i\in[u]}$ that are GSW-encryption of $\boldsymbol{v} = \{v_i\}_{i\in[u]}$ and a function $f \in \mathcal{F}$, the public evaluating algorithm outputs the result $\mathbf{B}_{f(x)} \in \mathbb{Z}_q^{n\times m}$.
- $\mathsf{TrapEval}(\boldsymbol{v} \in \mathcal{X}^u, \mathbf{A} \in \mathbb{Z}_q^{n\times m}, \{\mathbf{R_i}\}_{i\in[u]}, f)$: The trapdoor evaluation algorithm outputs $\mathbf{R}_f$, such that $\mathsf{PubEval}(\{\mathbf{A}\mathbf{R}_i + v_i\mathbf{G}\}_{i\in[u]}, f) = \mathbf{A}\mathbf{R}_f + f(x)\mathbf{G}$. Furthermore, we have $\|\mathbf{R}_f\| \leq \delta \cdot \max_{i\in[u]} \|\mathbf{R}_i\|$.

Remark 11: ([21]). For a partitioning function $h_{z,\boldsymbol{\alpha},\boldsymbol{\beta}}(\boldsymbol{a})$ defined by $\mathsf{Eq}(1)$ below, let $\mathbf{B}$ as the matrix that recursively encodes $\{z_{ij}\}_{i\in[v],j\in[t]}$ and $\mathbf{B}'$ as the matrix that encodes $(\alpha_1, \beta_1, \cdots, \alpha_v, \beta_v)$, and let $\boldsymbol{h}'_{\boldsymbol{a}}$ be the function $h_{z,\boldsymbol{\alpha},\boldsymbol{\beta}}(\boldsymbol{a})$. Then the algorithms

$$(\mathsf{PubEval}_{||}^{(\boldsymbol{a},v,t)}(\mathbf{B}, \mathbf{B}', \boldsymbol{h}'_{\boldsymbol{a}}), \mathsf{TrapEval}_{||}^{(\boldsymbol{a},v,t)}((z, \boldsymbol{\alpha}, \boldsymbol{\beta}), \mathbf{A}, \mathbf{R}, \mathbf{R}', \boldsymbol{h}'_{\boldsymbol{a}}))$$

are $vn^2m^3z^{O(z)}\log^3 q$-expanding.

D. PARTITIONING STRATEGY

The partitioning strategy required by the trapdoor vanishing technique need to design a hash function to isolate the challenge identity from all other identity queries. That is, given a challenge identity $\boldsymbol{a}^* \notin Q$, and a set of identity queries Q, the hash function should separate them with non-negligible probability, i.e. $H(\boldsymbol{a}^*) = 0$ and $H(\boldsymbol{a}) \neq 0$ for all $\boldsymbol{a} \in Q$.

Apon *et al.* [21] show that in fact, almost pairwise independent hash functions satisfying the following properties have the isolation property.

Lemma 12: ([21], Lemma 6.1). Let $Q \subseteq \{0, 1\}^n$, A, B be integers such that $B \leq A$, $|Q| \leq \delta B$ for some $\delta \in (0, 1)$, and let $\mathcal{H} : \{0, 1\}^n \to \mathcal{Y}$ be an almost pairwise independent hash function family which has the following parameters:

- $\forall \boldsymbol{a} \in \{0, 1\}^n$, $\Pr_{H\leftarrow\mathcal{H}}[H(\boldsymbol{a}) = 0] = 1/A$;
- $\forall \boldsymbol{a} \neq \boldsymbol{b} \in \{0, 1\}^n$, $\Pr_{H\leftarrow\mathcal{H}}[H(\boldsymbol{a}) = 0|H(\boldsymbol{b}) = 0] \leq 1/B$.

Then for any element $\boldsymbol{a} \notin Q$, we have

$$\Pr_{H\leftarrow\mathcal{H}}[H(\boldsymbol{a}) = 0 \wedge H(\boldsymbol{a}') \neq 0, \forall \boldsymbol{a}' \in Q] \in (\frac{1-\delta}{A}, \frac{1}{A}).$$

New Partitioning Function. Apon *et al.* [21] defined a basic partition family $\mathcal{H} : \{0, 1\}^n \to \mathbb{Z}_q$, each hash function

$h_{z,\alpha,\beta} \in \mathcal{H}$ is indexed by an element $z \in [2n^2]$, $\alpha \in \mathbb{Z}_q$, $\beta \in \mathbb{Z}_q$, and maps a boolean string $\boldsymbol{a} = (a_0, \cdots, a_{n-1}) \in \{0,1\}^n$ to an integer in $\mathbb{Z}_q$ as

$$h_{z,\alpha,\beta}(\boldsymbol{a}) = \alpha f_{\boldsymbol{a}}(z) + \beta,$$

where $f_{\boldsymbol{a}}(x) = \sum_{i=0}^{n-1} a_i x^i$ is a polynomial.

In order to approximate the conditional probability of $1/|Q|$ in the second property for the set $|Q|$ of all different polynomial sizes, as required by Lemma 12, [21] also proposed a parallel repetition version of the partition function as follows:

Let $\nu \geq u$ be parameters, the family $\mathcal{H}^{(u,\nu)} : \{0,1\}^n \to \mathbb{Z}_q^\nu$ is considered as the u-parallel repetition of the basic hash family $\mathcal{H}$. Each hash function $\boldsymbol{h}_{\boldsymbol{z},\boldsymbol{\alpha},\boldsymbol{\beta}} \in \mathcal{H}^{(u,\nu)}$ is indexed by vectors $\boldsymbol{z} \in [2n^2]^u$, $\boldsymbol{\alpha}, \boldsymbol{\beta} \in \mathbb{Z}_q^u$. The function $\boldsymbol{h}_{\boldsymbol{z},\boldsymbol{\alpha},\boldsymbol{\beta}}$ is defined as

$$\boldsymbol{h}_{\boldsymbol{z},\boldsymbol{\alpha},\boldsymbol{\beta}}(\boldsymbol{a}) = (h_{z_1,\alpha_1,\beta_1}(\boldsymbol{a}), \cdots, h_{z_u,\alpha_u,\beta_u}(\boldsymbol{a}), 0, \cdots, 0) \quad (1)$$

Lemma 13: ([21])**.** For a random hash function $\boldsymbol{h}_{\boldsymbol{z},\boldsymbol{\alpha},\boldsymbol{\beta}} \in \mathcal{H}^{(u,\nu)}$, where $\boldsymbol{z} \in [2n^2]^u$, $\boldsymbol{\alpha}, \boldsymbol{\beta} \in \mathbb{Z}_q^u$, we have

- $\forall \boldsymbol{a} \in \{0,1\}^n$, $\Pr[\boldsymbol{h}_{\boldsymbol{z},\boldsymbol{\alpha},\boldsymbol{\beta}}(\boldsymbol{a}) = 0] = (1/q)^u$;
- $\forall \boldsymbol{a} \neq \boldsymbol{b} \in \{0,1\}^n$, $\Pr[\boldsymbol{h}_{\boldsymbol{z},\boldsymbol{\alpha},\boldsymbol{\beta}}(\boldsymbol{b}) = 0 | \boldsymbol{h}_{\boldsymbol{z},\boldsymbol{\alpha},\boldsymbol{\beta}}(\boldsymbol{a}) = 0] \leq (1/n)^u$.

As shows in [21], in the security proof, in order to make the conditional probability approximate $1/|Q|$, we have freedom to set u appropriately. Furthermore, the above Lemmas 12 and 13 lead to the following Lemma on which the security of the schemes in [20], [21], and our leveled IBFHS scheme depend.

Lemma 14: ([21])**.** For a random hash function $h_{\boldsymbol{z},\boldsymbol{\alpha},\boldsymbol{\beta}}(\boldsymbol{a}) : \{0,1\}^n \to \mathbb{Z}_q^\nu$, where $u \leq \nu, \boldsymbol{z} \in [2n^2]^u, \boldsymbol{\alpha}, \boldsymbol{\beta} \in \mathbb{Z}_q^u$. Let $Q \subseteq \{0,1\}^n$, $\varepsilon \in [0,1]$, $A = q^u$, $B = n^u$ be integers such that $B \leq A$, $|Q| \leq \delta B$ for some $\delta \in (0,1)$. Let $u = \lceil \log_n(2|Q|/\varepsilon) \rceil$, it follow that $|Q| \leq 0.5n^u$. By setting $\delta = 0.5\varepsilon$, then for any element $\boldsymbol{a} \notin Q$, we have

$$\Pr_{\boldsymbol{h} \leftarrow \mathcal{H}}[\boldsymbol{h}_{\boldsymbol{z},\boldsymbol{\alpha},\boldsymbol{\beta}}(\boldsymbol{a}) = 0 \wedge \boldsymbol{h}_{\boldsymbol{z},\boldsymbol{\alpha},\boldsymbol{\beta}}(\boldsymbol{a}') \neq 0, \forall \boldsymbol{a}' \in Q] \in ((1 - 0.5\varepsilon)/q^u, 1/q^u).$$

The quantity $(1 - 0.5\varepsilon)/q^u \geq 0.5\varepsilon/q^u$ is non-negligible as long as ε is non-negligible for parameter $|Q|$ that is polynomially bounded. In this paper, we remark that ε denotes the probability that a adversary wins the security experiment.

III. DEFINITIONS AND SECURITY MODEL

A. LEVELED IDENTITY-BASED FULLY HOMOMORPHIC SIGNATURES

We use λ to denote the security parameter. The security parameter λ defines the identity space $\mathcal{ID}$ and message space $\mathcal{M}$. Let $\mathcal{C}$ be a collection of circuits $f : \mathcal{M}^l \to \mathcal{M}$ that takes l messages and outputs a result message in $\mathcal{M}$ (l denotes the maximum size of the dataset).

Definition 15: A multi-dataset IBHS scheme for the class of circuit $\mathcal{C}$ is a tuple of polynomial-time algorithms (**Setup, Extract, Sign, Eval, Verify**) specified as follows:

- **Setup** $(1^\lambda, l)$**:** When a security parameter λ and the maximum data-size l are input, this algorithm outputs a master key pair $(\mathsf{mpk}, \mathsf{msk})$.
- **Extract** $(\mathsf{mpk}, \mathsf{msk}, \mathsf{id})$**:** On input a master key pair $(\mathsf{mpk}, \mathsf{msk})$ and an identity id, it outputs identity-key $\mathsf{sk}_{\mathsf{id}}$ for the identity id.
- **Sign** $(\mathsf{id}, \mathsf{sk}_{\mathsf{id}}, \boldsymbol{\tau}, u_i, i)$**:** On input an identity and its secret key $\mathsf{sk}_{\mathsf{id}}$, a dataset tag $\boldsymbol{\tau}$, a message $u_i \in \mathcal{M}$ and its index $i \in [l]$, it outputs a signatures σ.
- **Eval** $(\mathsf{id}, \boldsymbol{\tau}, \boldsymbol{u}, \boldsymbol{\sigma}, f)$**:** On input an identity id, a tag $\boldsymbol{\tau}$, a vector of messages $\boldsymbol{u} = (u_1, \cdots, u_l)$ and corresponding signatures $\boldsymbol{\sigma} = (\sigma_1, \cdots, \sigma_l)$, and a circuit $f \in \mathcal{C}$. It outputs a derived signature σ_f, along with the identity id, the tag $\boldsymbol{\tau}$, an evaluated message $\mu = f(\boldsymbol{u})$ and the circuit f.
- **Verify** $(\mathsf{id}, \boldsymbol{\tau}, \sigma_f, \mu, f)$**:** On input a identity id and a tag $\boldsymbol{\tau}$, an evaluated message/signature pair (μ, σ_f) and a circuit $f \in \mathcal{C}$. It outputs 0 (reject) or 1 (accept).

The tags are used to differentiate the datasets with the intent that only signatures with the same tags be combinable homomorphically.

Correctness. We say that the $\mathcal{C}$-IBHS scheme is correct, if for any identity $\mathsf{id} \in \{0,1\}^{|\mathsf{id}| \geq \lambda}$, any tag $\boldsymbol{\tau} \in \{0,1\}^{|\boldsymbol{\tau}| \geq \lambda}$, any circuit $f \in \mathcal{C}$, any set of messages $\boldsymbol{u} = (u_1, \cdots, u_l) \in \mathcal{M}^l$, and corresponding set of signatures $\boldsymbol{\sigma} = (\sigma_1, \cdots, \sigma_l)$, we have

$$\Pr[\mathbf{Verify}(\mathsf{id}, \boldsymbol{\tau}, \sigma_f, \mu, f) = 1] = 1$$

where $\mathsf{sk}_{\mathsf{id}} \leftarrow \mathbf{Extract}(\mathsf{mpk}, \mathsf{msk}, \mathsf{id})$, $\sigma_i \leftarrow \mathbf{Sign}(\mathsf{id}, \mathsf{sk}_{\mathsf{id}}, \boldsymbol{\tau}, u_i, i)$, for $i \in [l]$, $\sigma_f \leftarrow \mathbf{Eval}(\mathsf{id}, \boldsymbol{\tau}, \boldsymbol{u}, \boldsymbol{\sigma}, f)$ and $\mu = f(\boldsymbol{u})$.

We remark that the circuit f can also be a projection circuit π_i, i.e., $\pi_i(u_1, \cdots, u_l) = u_i$, which also implies that the correctness must hold for single-message signatures.

B. SECURITY MODELS

The experiment $\mathsf{Exp}^{\mathrm{SU-ID-CMA}}_{\mathcal{A},\mathsf{IBFHS}}(1^\lambda)$ defined in below describes the strongly-unforgeable adaptively chosen identity and message attack security game, where the adversary can perform arbitrary private key query and then select an arbitrary target identity without declaring which identity will be attacked before obaining the master public key and public parameters, and can also make adaptive queries on individual messages.

- **Setup:** The challenger C obtains the master key pair $(\mathsf{mpk}, \mathsf{msk})$ by running the **Setup** $(1^\lambda, l)$ algorithm, and then sends the master public key mpk to adversary $\mathcal{A}$ while keeping msk a secret.
- **Queries:** The adversary's attack capabilities are modeled by allowing its access to two kinds of oracles, so the adversary can ask a polynomial number of queries, as shown below:
  – *Extract Queries*: Proceeding adaptively, the adversary queries a sequence of identities. On the k-th query, the challenger runs **Extract** $(\mathsf{mpk}, \mathsf{msk}, \mathsf{id})$ algorithm, and sends the result $\mathsf{sk}_{\mathsf{id}_k}$ to $\mathcal{A}$.

– *Signing Queries*: Proceeding adaptively, the $\mathcal{A}$ specifies a sequence of signature queries. Each query consists of an identity $\mathsf{id} \in \{0,1\}^{|\mathsf{id}|\geq\lambda}$, the dataset tag $\boldsymbol{\tau} \in \{0,1\}^{|\boldsymbol{\tau}|\geq\lambda}$, the message $u_i \in \mathcal{M}$ and its index $i \in [l]$. The challenger C runs the algorithm **Sign** $(\mathsf{id}, \mathsf{sk}_{\mathsf{id}}, \boldsymbol{\tau}, u_i, i)$, and sends the result signature σ_i to $\mathcal{A}$.

- **Forgery:** The $\mathcal{A}$ outputs a forgery tuple $(\mathsf{id}^*, \boldsymbol{\tau}^*, \mu^*, \sigma^*_{f^*}, f^*)$, where $\sigma^*_{f^*} \leftarrow$**Eval** $(\mathsf{id}^*, \boldsymbol{\tau}^*, \boldsymbol{u}^*, \boldsymbol{\sigma}^*, f^*)$.
  The adversary wins if **Verify** $(\mathsf{id}^*, \boldsymbol{\tau}^*, \mu^*, \sigma^*_{f^*}, f^*) = 1$, the identity id^* does not appear in Extract queries, and one of the following conditions is met:
  (1) **Type I forgery.** The identity $\mathsf{id}^* \neq \mathsf{id}_k$ for all id_k that appears in the signing queries.
  (2) **Type II forgery.** The identity $\mathsf{id}^* = \mathsf{id}_k$ for some id_k that appears in the signing queries, but the dataset identifier $\boldsymbol{\tau}^* \neq \boldsymbol{\tau}_k$ for all $\boldsymbol{\tau}_k$ that appear in the adversary's signing queries.
  (3) **Type III forgery.** The identity $\mathsf{id}^* = \mathsf{id}_k$ and the dataset identifier $\boldsymbol{\tau}^* = \boldsymbol{\tau}_k$ for some id_k and $\boldsymbol{\tau}_k$ that appear in the adversary's signing queries, and one of the following properties is satisfied:
  –**Type (a) forgery.** With the exception of $\mu^* \neq f^*(\boldsymbol{u}^*)$, the other components of the forgery are the same as those produced honestly.
  –**Type (b) forgery.** With the exception of $\sigma^*_{f^*} \neq \sigma_{f^*}$, the other components of the forgery are the same as those produced honestly, where σ_{f^*} is the evaluation signature generated honestly.

Definition 16 (Unforgeability): An identity-based homomorphic signature scheme is unforgeable against adaptively chosen identity and message attacks with respect to a circuit family $\mathcal{C}$ taking l inputs, if for all security parameters λ, no PPT adversary $\mathcal{A}$ can win the experiment $\mathsf{Exp}^{\mathrm{SU-ID-CMA}}_{\mathcal{A},\mathsf{IBFHS}}(1^\lambda)$ with non-negligible probability.

IV. OUR CONSTRUCTION

Our leveled IBFHS scheme for the class of circuits $\mathcal{C}$ is described as follows:

- **Setup** $(1^\lambda, 1^{|\mathsf{id}|}, 1^{|\boldsymbol{\tau}|}, 1^l, 1^{d_{max}})$: The setup algorithm takes in the security parameter λ, the number of bits $|\mathsf{id}|$ for the identity id, the number of bits $|\boldsymbol{\tau}|$ for the tag $\boldsymbol{\tau}$, the maximum number l of inputs for the circuit family $\mathcal{C}$ and the maximum depth d_{max} for the circuit family $\mathcal{C}$, and proceeds as follows.
  (1) Set the lattices parameters $n = n(\lambda, d_{max}), q = q(n, d_{max}), m = m(n, d_{max})$. Select parameters $v_1 = \log\log|\mathsf{id}|, t_1 = 2\log 2|\mathsf{id}|$ and $v_2 = \log\log|\boldsymbol{\tau}|, t_2 = 2\log 2|\boldsymbol{\tau}|$. Let $s_1 = s_1(n)$ and $s_2 = s_2(n)$ denote Gaussian parameters, and let $B = B(n, d_{max})$ denote an upper bound on the size of signatures.
  (2) Let S =(**Setup**′, **Extract**′, **Sign**′, **Eval**′, **Verify**′) is a standard identity-based signature scheme. Compute $(\mathsf{mpk}', \mathsf{msk}') \leftarrow$ **Setup**′(1^λ) as the master public/secret key for S.
  (3) Sample four random matrices: $\mathbf{A}, \mathbf{A_1}, \mathbf{B}, \mathbf{B}'$ in $\mathbb{Z}_q^{n\times m}$.
  (4) Sample one matrix with associated trapdoor
  $$(\mathbf{A}^*, \mathbf{T}_{\mathbf{A}^*}) \leftarrow \mathsf{TrapGen}(1^n, 1^m, q)$$
  (5) Output the master public key $\mathsf{mpk} = (\mathbf{A}^*, \mathbf{A}, \mathbf{A_1}, \mathbf{B}, \mathbf{B}', \mathsf{mpk}')$ and the master secret key $\mathsf{msk} = (\mathbf{T}_{\mathbf{A}^*}, \mathsf{msk}')$.
- **Extract** $(\mathsf{mpk}, \mathsf{msk}, \mathsf{id})$: On input $\mathsf{mpk}, \mathsf{msk}$ and an identity $\mathsf{id} \in \{0,1\}^{|\mathsf{id}|}$, the algorithm proceeds as follows.
  (1) Compute $\mathsf{sk}^{(1)}_{\mathsf{id}} \leftarrow$ **Extract**′$(\mathsf{msk}', \mathsf{id})$.
  (2) Compute $\mathbf{B}_{\mathsf{id}} \leftarrow \mathsf{PubEval}^{(\mathsf{id}, v_1, t_1)}_{||}(\mathbf{B}, \mathbf{B}', \boldsymbol{h}'_{\mathsf{id}})$.
  (3) Let $\mathbf{A}'_{\mathsf{id}} := [\mathbf{A}^*|\mathbf{A} + \mathbf{B}_{\mathsf{id}}]$ denote the "identity matrix".
  (4) Let
  $$\boldsymbol{r}_{\mathsf{id}} \leftarrow \mathsf{SampleLeft}(\mathbf{A}^*, \mathbf{A} + \mathbf{B}_{\mathsf{id}}, \mathbf{T}_{\mathbf{A}^*}, \mathbf{G}_n - \mathbf{A_1}, s_1)$$
  such that
  $\mathbf{A}'_{\mathsf{id}} \cdot \boldsymbol{r}_{\mathsf{id}} = \mathbf{G}_n - \mathbf{A_1}$ $(i.e.,\ \mathbf{A_1} = -\mathbf{A}'_{\mathsf{id}} \cdot \boldsymbol{r}_{\mathsf{id}} + \mathbf{G}_n)$.
  (5) Set identity-specific public key $\mathbf{A}_{\mathsf{id}} = [\mathbf{A}'_{\mathsf{id}}|\mathbf{A}_1]$. Note that $\boldsymbol{r}_{\mathsf{id}}$ is a $\mathbf{G}_n$-trapdoor of $\mathbf{A}_{\mathsf{id}}$ with tag $\mathbf{I}_n$.
  (6) Based on Lemma 8, use the $\mathbf{G}$-trapdoor $\boldsymbol{r}_{\mathsf{id}}$ and a random basis of $\Lambda^\perp_q(\mathbf{G}_n)$ to generate a short basis $\mathbf{T}_{\mathbf{A}_{\mathsf{id}}}$ for lattice $\Lambda^\perp_q(\mathbf{A}_{\mathsf{id}})$.
  Finally, let $\mathsf{sk}^{(2)}_{\mathsf{id}} = \mathbf{T}_{\mathbf{A}_{\mathsf{id}}}$ and output a secret key $\mathsf{sk}_{\mathsf{id}} = (\mathsf{sk}^{(1)}_{\mathsf{id}}, \mathsf{sk}^{(2)}_{\mathsf{id}})$.
- **sign** $(\mathsf{id}, \mathsf{sk}_{\mathsf{id}}, \boldsymbol{\tau}, u_i, i)$: Suppose that the algorithm has stored a list L of all previously returned identities id with the related information $\sigma^{(1)}$ and $\mathsf{V}_{\mathsf{id}} = (\mathbf{C}_{\mathsf{id}}, \mathbf{C}'_{\mathsf{id}}, \mathbf{A}''_{\mathsf{id}}, \{\mathbf{D}_{\mathsf{id},k}\}_{k\in\{0,1\}})$ defined below.
  Take an identity id, the secret key $\mathsf{sk}_{\mathsf{id}}$, a dataset tag $\boldsymbol{\tau} \in \{0,1\}^{|\boldsymbol{\tau}|}$, a message $u_i \in \{0,1\}$ and its index $i \in [l]$ as input, the signing algorithm responses according to the type of id input:
  – If id appears in L, then retrieve the associated $\sigma^{(1)}$ and V_{id} from L.
  – Otherwise, choose uniformly at random three matrices $\mathbf{C}_{\mathsf{id}}, \mathbf{C}'_{\mathsf{id}}$ and $\mathbf{A}''_{\mathsf{id}}$ in $\mathbb{Z}_q^{n\times m}$, and two random matrices $\{\mathbf{D}_{\mathsf{id},k}\}_{k\in\{0,1\}} \in \mathbb{Z}_q^{n\times m}$. Let $\mathsf{V}_{\mathsf{id}} = (\mathbf{C}_{\mathsf{id}}, \mathbf{C}'_{\mathsf{id}}, \mathbf{A}''_{\mathsf{id}}, \{\mathbf{D}_{\mathsf{id},k}\}_{k\in\{0,1\}})$, compute $\sigma^{(1)} \leftarrow$**Sign**′$(\mathsf{id}, \mathsf{sk}^{(1)}_{\mathsf{id}}, \mathsf{V}_{\mathsf{id}})$, and then store $(\mathsf{id}, \mathsf{V}_{\mathsf{id}}, \sigma^{(1)})$ in L.
  Next, the algorithm proceeds as follows:
  (1) Compute $\mathbf{C}_{\mathsf{id},\boldsymbol{\tau}} \leftarrow \mathsf{PubEval}^{(\boldsymbol{\tau}, v_2, t_2)}_{||}(\mathbf{C}_{\mathsf{id}}, \mathbf{C}'_{\mathsf{id}}, h'_{\boldsymbol{\tau}})$.
  (2) Let $\mathbf{A}_{\mathsf{id},\boldsymbol{\tau}} := [\mathbf{A}_{\mathsf{id}}|\mathbf{A}''_{\mathsf{id}} + \mathbf{C}_{\mathsf{id},\boldsymbol{\tau}}]$ denote the "identity-dataset matrix".
  (3) For $k = 0, 1$, take as input $\{\mathbf{D}_{\mathsf{id},k}\}_{k\in\{0,1\}}$ and sample two matrices $\sigma'^{(2)}_k \in \mathbb{Z}_q^{2m\times m}$ using,
  $$\sigma'^{(2)}_k \leftarrow \mathsf{SampleLeft}(\mathbf{A}_{\mathsf{id}}, \mathbf{A}''_{\mathsf{id}} + \mathbf{C}_{\mathsf{id},\boldsymbol{\tau}}, \mathbf{T}_{\mathbf{A}_{\mathsf{id}}}, \mathbf{D}_{\mathsf{id},k} + [k\mathbf{I}_n|\cdots|k\mathbf{I}_n]\mathbf{G}_{ln}, s_1).$$

such that

$$\mathbf{A}_{\mathsf{id},\boldsymbol{\tau}} \cdot \sigma_k'^{(2)} = \mathbf{D}_{\mathsf{id},k} + [k\mathbf{I}_n| \cdots |k\mathbf{I}_n]\mathbf{G}_{ln}.$$

We follow the naming in [20], calling the two small norm matrices $\{\sigma_k'^{(2)}\}_{k\in\{0,1\}}$ "presignatures".

(4) For $i \in [l], u_i \in \{0,1\}$, compute $\sigma_{u_i}^{(2)} = \sigma_{u_i}'^{(2)}\mathbf{G}_{ln}^{-1}(\mathbf{T}_i\mathbf{G}_n)$, such that

$$\mathbf{A}_{\mathsf{id},\boldsymbol{\tau}} \cdot \sigma_{u_i}^{(2)} = \mathbf{D}_{\mathsf{id},u_i}\mathbf{G}_{ln}^{-1}(\mathbf{T}_i\mathbf{G}_n) + u_i\mathbf{G}_n.$$

Finally, output the signature $\sigma_i = (\mathsf{V}_{\mathsf{id}}, \sigma^{(1)}, \sigma_{u_i}^{(2)})$.

- **Eval** $(\mathsf{id}, \boldsymbol{\tau}, \boldsymbol{u}, \boldsymbol{\sigma}, f)$: Given an identity id, a dataset tag $\boldsymbol{\tau}$, a circuit $f \in \mathcal{C}$, and a pair of message/signature vectors $(\boldsymbol{u}, \boldsymbol{\sigma})$, where $\boldsymbol{u} = (u_1, \cdots, u_l)$, $\boldsymbol{\sigma} = ((\mathsf{V}_{\mathsf{id},1}, \sigma_1^{(1)}, \sigma_{u_1}^{(2)}), \cdots, (\mathsf{V}_{\mathsf{id},l}, \sigma_l^{(1)}, \sigma_{u_l}^{(2)}))$. This algorithm checks if $\mathsf{V}_{\mathsf{id},i}$ are not all equal, then output $\perp$. Else, it computes a homomorphic signature recursively gate by gate as follows:
  (1) Let $g = (u, v, w)$ be a NAND gate. For each wire in the gate, let $\mathbf{D}_{\mathsf{id},i}$ be the public matrix associated with that wire. (Such matrices are fixed by the **Sign** algorithm). Also construct the "identity-dataset matrix" $\mathbf{A}_{\mathsf{id},\boldsymbol{\tau}} := [\mathbf{A}_{\mathsf{id}}|\mathbf{A}''_{\mathsf{id}} + \mathbf{C}_{\mathsf{id},\boldsymbol{\tau}}]$.
  (2) By induction on the public matrix associated with each wire, we have $(\sigma_u^{(2)}, \sigma_v^{(2)})$ such that:

$$\mathbf{A}_{\mathsf{id},\boldsymbol{\tau}} \cdot \sigma_u^{(2)} = \mathbf{D}_{\mathsf{id},u}\mathbf{G}_{ln}^{-1}(\mathbf{T}_i\mathbf{G}_n) + x\mathbf{G}_n,$$
$$\mathbf{A}_{\mathsf{id},\boldsymbol{\tau}} \cdot \sigma_v^{(2)} = \mathbf{D}_{\mathsf{id},v}\mathbf{G}_{ln}^{-1}(\mathbf{T}_j\mathbf{G}_n) + y\mathbf{G}_n.$$

  where (x, y) are the messages carried by wires (u, v), and i, j are indices of the messages x, y, respectively.
  (3) Define the public matrix associated with the output wire w as

$$\mathbf{D}_{\mathsf{id},w} = \mathbf{D}_{\mathsf{id},v}\mathbf{G}_{ln}^{-1}(\mathbf{T}_j\mathbf{G}_n)\tilde{\mathbf{D}}_{id,u} - \mathbf{G}_n,$$

  where $\tilde{\mathbf{D}}_{\mathsf{id},u} = \mathbf{G}_n^{-1}(\mathbf{D}_{\mathsf{id},u}\mathbf{G}_{ln}^{-1}(\mathbf{T}_i\mathbf{G}_n)) \in \mathbb{Z}^{m\times m}$, so that $\mathbf{G}_n\tilde{\mathbf{D}}_{\mathsf{id},u} = \mathbf{D}_{\mathsf{id},u}\mathbf{G}_{ln}^{-1}(\mathbf{T}_i\mathbf{G}_n)$.
  (4) Output the homomorphic signature: $\sigma_w = (\mathsf{V}_{\mathsf{id}}, \sigma^{(1)}, \sigma_w^{(2)})$, where $\mathsf{V}_{\mathsf{id}} = \mathsf{V}_{\mathsf{id},i}$ and $\sigma^{(1)} = \sigma_i^{(1)}$ for $i \in [l]$, and $\sigma_w^{(2)} := \sigma_v^{(2)}\tilde{\mathbf{D}}_{\mathsf{id},u} - y\sigma_u^{(2)}$.
- **Verify**$(\mathsf{id}, \boldsymbol{\tau}, \mu, \sigma, f)$: The verification algorithm takes as input an identity id, a tag $\boldsymbol{\tau}$, a message / signature pair $(\mu, \sigma = (\mathsf{V}_{\mathsf{id}}, \sigma^{(1)}, \sigma_f^{(2)}))$, and a circuit f. It accept only if the following requirements are satisfied:
  (1) **Verify**$'(\mathsf{id}, \sigma^{(1)}, \mathsf{V}_{\mathsf{id}}) = 1$.
  (2) Verify that $\|\sigma_f^{(2)}\|_\infty \le B$.
  (3) Let $\mathbf{A}_{\mathsf{id},\boldsymbol{\tau}} := [\mathbf{A}_{\mathsf{id}}|\mathbf{A}''_{\mathsf{id}} + \mathbf{C}_{\mathsf{id},\boldsymbol{\tau}}] \in \mathbb{Z}_q^{n\times 4m}$, which is computed as in the signing algorithm. Additionally, $\mathbf{D}_f$ denotes the public matrix associated with the circuit f as computed by the abovedescribed evaluation algorithm. Next, check that

$$\mathbf{A}_{\mathsf{id},\boldsymbol{\tau}} \cdot \sigma_f^{(2)} = \mathbf{D}_f + \mu\mathbf{G}_n.$$

  Here, μ is the message resulting from the circuit evaluation $\mu = f(\boldsymbol{u})$, and $\sigma_f^{(2)}$ is the corresponding signature homomorphically evaluated from the signature vector $\{\sigma_{u_i}^{(2)}\}_{i\in[l]}$.

Correctness. We show that for the choice of lattice parameters specified later, the verification algorithm accepts a honestly computed evaluated signature. We adapt the induction method given in [36] to state our claim.

Lemma 17: Let $\mathbf{A}_{\mathsf{id},\boldsymbol{\tau}}$ be the identity-dataset matrix, $\mathbf{D}_f$ be the public matrix derived with respect to circuit f, and $\sigma = (\mathsf{V}_{\mathsf{id}}, \sigma^{(1)}, \sigma_f^{(2)})$ be the homomorphically computed signature. It then hold that **Verify**$'(\mathsf{id}, \sigma^{(1)}, \mathsf{V}_{\mathsf{id}}) = 1$ and $\mathbf{A}_{\mathsf{id},\boldsymbol{\tau}} \cdot \sigma_f^{(2)} = \mathbf{D}_f + f(\boldsymbol{u})\mathbf{G}_n$. Moreover, we have $\|\sigma_f^{(2)}\|_\infty \le B = m^{O(d_{max})}$.

Proof: Without loss of generality, we consider the correctness of the NAND gate $g = (u, v, w)$ with input wires (u, v) carrying values (x, y) respectively, and with output wire w. **Base case.** Let $\sigma_i = (\mathsf{V}_{\mathsf{id}}, \sigma^{(1)}, \sigma_{u_i}^{(2)})$ be the signature for message u_i under identity id and tag $\boldsymbol{\tau}$, by construction of the **Sign** algorithm, we have **Verify** $'(\mathsf{id}, \sigma^{(1)}, \mathsf{V}_{\mathsf{id}}) = 1$, and

$$\mathbf{A}_{\mathsf{id},\boldsymbol{\tau}} \cdot \sigma_{u_i}^{(2)} = \mathbf{D}_{\mathsf{id},u_i}\mathbf{G}_{ln}^{-1}(\mathbf{T}_i\mathbf{G}_n) + u_i\mathbf{G}_n.$$

Inductive step. Let matrices $\sigma_u = (\mathsf{V}_{\mathsf{id}}, \sigma^{(1)}, \sigma_u^{(2)})$, $\sigma_v = (\mathsf{V}_{\mathsf{id}}, \sigma^{(1)}, \sigma_v^{(2)})$ be signatures of messages x, y respectively, under public keys $\mathbf{D}_{\mathsf{id},u}, \mathbf{D}_{\mathsf{id},v}$, such that

$$\mathbf{A}_{\mathsf{id},\boldsymbol{\tau}} \cdot \sigma_u^{(2)} = \mathbf{D}_{\mathsf{id},u}\mathbf{G}_{ln}^{-1}(\mathbf{T}_i\mathbf{G}_n) + x\mathbf{G}_n,$$
$$\mathbf{A}_{\mathsf{id},\boldsymbol{\tau}} \cdot \sigma_v^{(2)} = \mathbf{D}_{\mathsf{id},v}\mathbf{G}_{ln}^{-1}(\mathbf{T}_j\mathbf{G}_n) + y\mathbf{G}_n.$$

Let

$$\sigma_w^{(2)} = \sigma_v^{(2)}\tilde{\mathbf{D}}_{id,u} - y\sigma_u^{(2)},$$
$$\mathbf{D}_{\mathsf{id},w} = \mathbf{D}_{\mathsf{id},v}\mathbf{G}_{ln}^{-1}(\mathbf{T}_j\mathbf{G}_n)\tilde{\mathbf{D}}_{\mathsf{id},u} - \mathbf{G}_n.$$

Then we must show that

$$\mathbf{A}_{\mathsf{id},\boldsymbol{\tau}} \cdot \sigma_w^{(2)} = \mathbf{D}_{id,w} + (x \text{ NAND } y)\mathbf{G}_n.$$

Indeed, we have

$$\begin{aligned}
&\mathbf{A}_{\mathsf{id},\boldsymbol{\tau}} \cdot \sigma_w^{(2)} \\
&= \mathbf{A}_{\mathsf{id},\boldsymbol{\tau}} \cdot (\sigma_v^{(2)} \cdot \tilde{\mathbf{D}}_{\mathsf{id},u} - y\sigma_u^{(2)}) \\
&= \mathbf{A}_{\mathsf{id},\boldsymbol{\tau}}\sigma_v^{(2)} \cdot \tilde{\mathbf{D}}_{\mathsf{id},u} - y\mathbf{A}_{\mathsf{id},\boldsymbol{\tau}} \cdot \sigma_u^{(2)} \\
&= (\mathbf{D}_{\mathsf{id},v}\mathbf{G}_{ln}^{-1}(\mathbf{T}_j\mathbf{G}_n) + y\mathbf{G}_n)\tilde{\mathbf{D}}_{\mathsf{id},u} \\
&\quad -y(\mathbf{D}_{\mathsf{id},u}\mathbf{G}_{ln}^{-1}(\mathbf{T}_i\mathbf{G}_n) + x\mathbf{G}_n) \\
&= \mathbf{D}_{\mathsf{id},v}\mathbf{G}_{ln}^{-1}(\mathbf{T}_j\mathbf{G}_n)\tilde{\mathbf{D}}_{\mathsf{id},u} + y\mathbf{D}_{\mathsf{id},u}\mathbf{G}_{ln}^{-1}(\mathbf{T}_i\mathbf{G}_n) \\
&\quad -y\mathbf{D}_{\mathsf{id},u}\mathbf{G}_{ln}^{-1}(\mathbf{T}_i\mathbf{G}_n) - xy\mathbf{G}_n \\
&= \mathbf{D}_{\mathsf{id},w} + \mathbf{G}_n - xy\mathbf{G}_n \\
&= \mathbf{D}_{\mathsf{id},w} + (1 - xy)\mathbf{G}_n \\
&= \mathbf{D}_{\mathsf{id},w} + (x \text{ NAND } y)\mathbf{G}_n.
\end{aligned}$$

Furthermore, $\|\sigma_w^{(2)}\| \le \|\sigma_v^{(2)}\| \cdot \mathsf{poly}(m) + \|\sigma_u^{(2)}\|$, the bound on the size of $\|\sigma_w^{(2)}\|$ follows from the observation that $\|\tilde{\mathbf{D}}_{\mathsf{id},u}\| \le \mathsf{poly}(m)$.

By induction, we have

$$\mathbf{A}_{\mathsf{id},\boldsymbol{\tau}} \cdot \sigma_f^{(2)} = \mathbf{D}_f + f(\boldsymbol{u})\mathbf{G}_n.$$

In addition, the size of the signature is $\|\sigma_f^{(2)}\|_\infty \leq B = m^{O(d_{max})}$.

V. SECURITY ANALYSIS

In this section, we prove the unforgeability of the proposed scheme against the adaptively chosen-message and identity attack. Assume that there exists a PPT adversary $\mathcal{A}$ that wins the adaptive-unforgeability security experiment defined in section 3, we construct an adversary $\mathcal{B}$ that can leverage the adversary $\mathcal{A}$ to break either the security of the identity-based signature scheme S or the $\mathsf{SIS}_{q,n,m,\beta}$ problem for lattice defined by the basis $\mathbf{A}^* \in \mathbb{Z}_q^{n\times m}$.

Theorem 18: (Unforgeability)**.** Assuming the hardness of the $\mathsf{SIS}_{q,n,m,\beta}$, and that the identity-based signature scheme S =(**Setup**′, **Extract**′, **Sign**′, **Eval**′, **Verify**′) is unforgeable under adaptive chosen identity and message attack, our IBFHS construction is strongly unforgeable against adaptive chosen identity and message attack.

The proof proceeds by contradiction. Assuming the existence of an adversary $\mathcal{A}$ that breaks the adaptive security of IBFHS, we show how to build a simulator $\mathcal{B}$ that breaks either the security of the identity-based signature scheme S or the $\mathsf{SIS}_{q,n,m,\beta}$ assumption. To see this, let $(\mathsf{id}^*, \boldsymbol{\tau}^*, \sigma^*, \mu^*, f^*)$ be a valid forgery returned by the adversary $\mathcal{A}$. We observe that this forgery can be of either one of the following types:

Type I. The list L_{id^*} has never been initialized during the game (i.e., the identity id^* has never been asked for the signature).

Type II. The list L_{id^*} has been initialized, but the dataset identifier $\boldsymbol{\tau}^* \neq \boldsymbol{\tau}_k$ for all $\boldsymbol{\tau}_k$ that appear in the adversary's signing queries.

Type III. The list L_{id^*} has been initialized and the dataset identifier $\boldsymbol{\tau}^* = \boldsymbol{\tau}_k$ for some $\boldsymbol{\tau}_k$ that appear in the adversary's signing queries, and one of the following properties is satisfied:

–**Type (a) forgery.** With the exception of $\mu^* \neq f^*(\boldsymbol{u}^*)$, the other components of the forgery are the same as those produced honestly.

–**Type (b) forgery.** With the exception of $\sigma_{f^*}^{*(2)} \neq \sigma_{f^*}^{(2)}$, the other components of the forgery are the same as those produced honestly, where $\sigma_{f^*}^{(2)}$ is generated honestly.

We show that for an adversary $\mathcal{A}$ producing a forgery of Type I, it is possible to break the security of S, whereas for every adversary $\mathcal{A}$ producing forgeries of Type II and Type III (a) (b), there exists an adversary that can solve the $\mathsf{SIS}_{q,n,m,\beta}$ problem. Theorem 18 is proven by combining the results of lemma 19 and lemma 20 given below.

Lemma 19: If $\mathcal{A}$ is an adversary against the security of IBFHS by producing a Type I forgery, then it is possible to build an adversary $\mathcal{B}$ that breaks the security of the identity-based signature scheme S.

Proof: Assuming C is a challenger for the underlying identity-based signature scheme **S**, first, $\mathcal{B}$ receives mpk' from C, chooses $(\mathbf{A}^*, \mathbf{T}_{\mathbf{A}^*}, \mathbf{A}, \mathbf{A_1}, \mathbf{B}, \mathbf{B}')$ and initializes an empty list L, as in the description of the real scheme. Then, $\mathcal{B}$ sends the master public key $(\mathbf{A}^*, \mathbf{A}, \mathbf{A_1}, \mathbf{B}, \mathbf{B}', \mathsf{mpk}')$ to $\mathcal{A}$.

- **Extract Queries.** When $\mathcal{A}$ requests the secret key for identity id, $\mathcal{B}$ uses its Extracting oracle for S to get $\mathsf{sk}_{\mathsf{id}}^{(1)}$ and can easily compute $\mathsf{sk}_{\mathsf{id}}^{(2)}$ as in the real case. Then, $\mathcal{B}$ sets $\mathsf{sk}_{\mathsf{id}} \leftarrow (\mathsf{sk}_{\mathsf{id}}^{(1)}, \mathsf{sk}_{\mathsf{id}}^{(2)})$ to $\mathcal{A}$.
- **Signing Queries.** When $\mathcal{A}$ asks for a signature on an identity id, a dataset $\boldsymbol{\tau}$ and a message u_i, $\mathcal{B}$ checks if id does appear in L, it retrieves the associated $(\mathsf{V}_{\mathsf{id}}, \sigma^{(1)})$ from L. Otherwise, $\mathcal{B}$ randomly chooses parameter $\mathbf{V}_{\mathsf{id}}$ as described in the above scheme, uses its Signing oracle for S to compute $\sigma^{(1)}$ for V_{id} and stores $(\mathsf{id}, \mathsf{V}_{\mathsf{id}}, \sigma^{(1)})$ in L. Then, $\mathcal{B}$ can easily compute $\sigma_{u_i}^{(2)}$. So the signature is $\sigma_i = (\mathsf{V}_{\mathsf{id}}, \sigma^{(1)}, \sigma_{u_i}^{(2)})$, it is easy to check that σ_i is valid.
- **Output.** Finally, the adversary $\mathcal{A}$ is supposed to output a forgery tuple $(\mathsf{id}^*, \boldsymbol{\tau}^*, \sigma^*, \mu^*, f^*)$ such that $\sigma^* = (\mathsf{V}_{\mathsf{id}^*}, \sigma^{*(1)}, \sigma_{f^*}^{*(2)})$, and **Verify** $((\mathsf{id}^*, \boldsymbol{\tau}^*, \sigma^*, \mu^*, f^*) = 1$. According to the definition of Type I forgery, the identity $\mathsf{id}^* \neq \mathsf{id}_k$ for all id_k that appears in signing queries. So $\mathcal{B}$ can output $(\mathsf{id}^*, \mathsf{V}_{\mathsf{id}^*}, \sigma^{*(1)})$ as a forgery for S.

It is straightforward to ascertain whether $\mathcal{A}$ has an advantage ε in forging the signature scheme, then $\mathcal{B}$ has a probability of at least $\varepsilon/3$ in breaking the security of S.

Lemma 20: If $\mathcal{A}$ is an adversary against the security of IBFHS by producing a Type II (or Type III) forgery, then it is possible to build an adversary $\mathcal{B}$ to solve the $\mathsf{SIS}_{q,n,m,\beta}$ problem with non-negligible probability.

Proof:

- **Invocation.** Adversary $\mathcal{B}$ is invoked on a random instance $\mathbf{A}^* \in \mathbb{Z}_q^{n\times m}$ of the $\mathsf{SIS}_{q,n,m,\beta}$ assumption, and is asked to output an solution $\boldsymbol{e} \in \mathbb{Z}^m \setminus \{\mathbf{0}\}$ such that $\mathbf{A}^*\boldsymbol{e} = 0 \mod q$ and $\|\boldsymbol{e}\| \leq \beta$.
- **Setup.** $\mathcal{B}$ runs the following algorithm **Setup** using the matrix $\mathbf{A}^*$ from the SIS challenge.
  (1) $\mathcal{B}$ runs $(\mathsf{mpk}', \mathsf{msk}') \leftarrow$**Sign**′$(1^\lambda)$ as the master public / secret key for S.
  (2) Set the parameters n, q, m, β according to the $\mathsf{SIS}_{q,n,m,\beta}$ problem. Let B denote the upper bound on the size of signatures. Let $s_1 = s_1(n), s_2 = s_2(n)$ denote the Gaussian parameters and $\nu_1 = \log\log|\mathsf{id}|, \nu_2 = \log\log|\boldsymbol{\tau}|, t_1 = 2\log 2|\mathsf{id}|, t_1 = 2\log 2|\boldsymbol{\tau}|$ be public parameters.
  (3) $\mathcal{B}$ randomly chooses integers $z_i \in [2|\mathsf{id}|^2], \alpha_i, \beta_i \in \mathbb{Z}_q$. For $i \in [\nu_1], j \in [t_1]$, set matrix $\mathbf{E}_{ij}$ as

$$\mathbf{E}_{ij} = \begin{cases} z_{ij}\mathbf{G}_n, & \text{if} \quad i \leq u \\ \mathbf{0}_{n\times m}, & \text{otherwise} \end{cases}$$

and then, compute

$$\mathbf{E} = \mathsf{RePack}(\{\mathbf{E}_{ij}\}_{i\in[\nu_1], j\in[t_1]})$$

Next, set matrix $\mathbf{E}'$ for encoding $\{\alpha_i\}$, $\{\beta_i\}$ as

$$\mathbf{E}' = [\alpha_1\mathbf{I}_n|\beta_1\mathbf{I}_n|\cdots|\alpha_u\mathbf{I}_n|\beta_u\mathbf{I}_n|\mathbf{0}_n|\cdots|\mathbf{0}_n]\cdot\mathbf{G}_{2\nu_1 n}$$

Then, $\mathcal{B}$ sets $\mathbf{B} = \mathbf{A}^*\mathbf{R} + \mathbf{E}$, $\mathbf{B}' = \mathbf{A}^*\mathbf{R}' + \mathbf{E}'$, where $\mathbf{R}, \mathbf{R}' \in \{-1, 1\}^{m\times m}$ are randomly chosen.

(4) Randomly choose two matrices $\mathbf{U}, \mathbf{U}_1 \in \{-1, 1\}^{m\times m}$ and set $\mathbf{A} = \mathbf{A}^*\mathbf{U}$, $\mathbf{A}_1 = \mathbf{A}^*\mathbf{U}_1$.

Finally, $\mathcal{B}$ sets the master public key $\mathsf{mpk} = (\mathbf{A}^*, \mathbf{A}, \mathbf{A}_1, \mathbf{B}, \mathbf{B}', \mathsf{mpk}')$.

- **Extract queries.** On input a master public key mpk and an identity id, perform the following:

(1) Abort the simulation if

$$\boldsymbol{h}_{z,\boldsymbol{\alpha},\boldsymbol{\beta}}(\mathsf{id}) = (h_{z_1,\alpha_1,\beta_1}(\mathsf{id}), \cdots, h_{z_u,\alpha_u,\beta_u}(\mathsf{id}), 0, \cdots, 0) = \mathbf{0}.$$

(2) Otherwise, compute

$$\mathbf{B}_{\mathsf{id}} \leftarrow \mathsf{PubEval}_{||}^{(\mathsf{id},\nu_1,t_1)}(\mathbf{B}, \mathbf{B}', \boldsymbol{h}'_{\mathsf{id}})$$

such that $\mathbf{B}_{\mathsf{id}} = \mathbf{A}^*\mathbf{R}_{\mathsf{id}} + \mathbf{E}_{\mathsf{id}}\cdot\mathbf{G}_{\nu_1 n}$, where

$$\mathbf{R}_{\mathsf{id}} \leftarrow \mathsf{TrapEval}_{||}^{(\mathsf{id},\nu_1,t_1)}((z, \boldsymbol{\alpha}, \boldsymbol{\beta}), \mathbf{A}^*, \mathbf{R}, \mathbf{R}', \boldsymbol{h}'_{\mathsf{id}})$$
$$\mathbf{E}_{\mathsf{id}} = [h_{z_1,\alpha_1,\beta_1}(\mathsf{id})\mathbf{I}_n|\cdots|h_{z_u,\alpha_u,\beta_u}(\mathsf{id})\mathbf{I}_n| \mathbf{0}_n|\cdots|\mathbf{0}_n].$$

Note that we let $\boldsymbol{h}'$ be the function $h_{z,\boldsymbol{\alpha},\boldsymbol{\beta}}$.

(3) Set the identity matrix $\mathbf{A}'_{id} = [\mathbf{A}^*|\mathbf{A}+\mathbf{B}_{id}]$, and use the algorithm $\mathsf{SampleD}^{\mathcal{O}}$ (c.f. Lemma 9) with the trapdoor $\mathbf{T}_{\mathbf{G}_{\nu_1 n}}$ to compute the following:

$$\boldsymbol{r}_{\mathsf{id}} \leftarrow \mathsf{SampleD}^{\mathcal{O}}(\mathbf{A}^*, -\mathbf{U}-\mathbf{R}_{\mathsf{id}}, \mathbf{E}_{\mathsf{id}}, \mathbf{G}_n-\mathbf{A}_1, s_2)$$

such that

$$\mathbf{A}'_{\mathsf{id}}\boldsymbol{r}_{\mathsf{id}} = \mathbf{G}_n - \mathbf{A}_1 \ \ (i.e.\ \mathbf{A}_1 = -\mathbf{A}'_{\mathsf{id}}\boldsymbol{r}_{\mathsf{id}} + \mathbf{G}_n).$$

(4) Set $\mathbf{A}_{\mathsf{id}} = [\mathbf{A}'_{\mathsf{id}}|\mathbf{A}_1]$, and use the $\mathbf{G}$-trapdoor $\boldsymbol{r}_{\mathsf{id}}$ for $\mathbf{A}_{\mathsf{id}}$ and a random basis of $\Lambda_q^{\perp}(\mathbf{G}_{\nu_1 n})$ to generate a short basis $\mathbf{T}_{\mathbf{A}_{\mathsf{id}}}$ for lattice $\Lambda_q^{\perp}(\mathbf{A}_{\mathsf{id}})$. Let $\mathsf{sk}_{\mathsf{id}}^{(2)} = \mathbf{T}_{\mathbf{A}_{\mathsf{id}}}$.

(5) $\mathcal{B}$ generates $\mathsf{sk}_{\mathsf{id}}^{(1)} \leftarrow \mathbf{Extract}'(\mathsf{id}, \mathsf{msk})$.

Finally, output a secret key $\mathsf{sk}_{\mathsf{id}} = (\mathsf{sk}_{\mathsf{id}}^{(1)}, \mathsf{sk}_{\mathsf{id}}^{(2)})$.

- **Sign queries.** Guess the type of forgery returned by adversary $\mathcal{A}$. Based on the guess, the adversary $\mathcal{B}$ with one of the following two subroutine:

– Type II forgery:

Phase1. When $\mathcal{A}$ adaptively asks for a signature on an identity id, a dataset tag $\boldsymbol{\tau}$ and a message u_i, $\mathcal{B}$ checks if id does appear in L, it retrieves the associated $\sigma^{(1)}$ and $\mathsf{V}_{\mathsf{id}} = (\mathbf{C}_{\mathsf{id}}, \mathbf{C}'_{\mathsf{id}}, \mathbf{A}''_{\mathsf{id}}, \{\mathbf{D}_{\mathsf{id},k}\}_{k\in\{0,1\}})$ from L.

Otherwise, $\mathcal{B}$ randomly chooses the integers $z_{\mathsf{id},i} \in [2|\boldsymbol{\tau}|^2]$, $\alpha_{\mathsf{id},i}, \beta_{\mathsf{id},i} \in \mathbb{Z}_q$. For $i \in [\nu_2], j \in [t_2]$, set matrix $\mathbf{E}_{\mathsf{id},ij}$ as follows:

$$\mathbf{E}_{\mathsf{id},ij} = \begin{cases} z_{\mathsf{id},ij}\mathbf{G}_n, & \text{if} \quad i \le u \\ \mathbf{0}_{n\times m}, & \text{otherwise} \end{cases}$$

Then compute

$$\mathbf{E}_{\mathsf{id}} = \mathsf{RePack}(\{\mathbf{E}_{\mathsf{id},ij}\}_{i\in[\nu_2],j\in[t_2]})$$

Next, set matrix $\mathbf{E}'_{\mathsf{id}}$ for encoding $\{\alpha_{\mathsf{id},i}\}$, $\{\beta_{\mathsf{id},i}\}$ as

$$\mathbf{E}'_{\mathsf{id}} = [\alpha_{\mathsf{id},1}\mathbf{I}_n|\beta_{\mathsf{id},1}\mathbf{I}_n|\cdots|\alpha_{\mathsf{id},u}\mathbf{I}_n|\beta_{\mathsf{id},u}\mathbf{I}_n| |\mathbf{0}_n|\cdots|\mathbf{0}_n]\cdot\mathbf{G}_{2\nu_2 n}$$

Then, $\mathcal{B}$ sets

$$\mathbf{C}_{\mathsf{id}} = \mathbf{A}_{\mathsf{id}}\hat{\mathbf{R}}_{\mathsf{id}} + \mathbf{E}_{\mathsf{id}},$$
$$\mathbf{C}'_{\mathsf{id}} = \mathbf{A}\hat{\mathbf{R}}'_{\mathsf{id}} + \mathbf{E}'_{\mathsf{id}}$$

where $\hat{\mathbf{R}}_{\mathsf{id}}, \hat{\mathbf{R}}'_{\mathsf{id}} \in \{-1, 1\}^{3m\times m}$ are randomly chosen. $\mathcal{B}$ randomly chooses a matrices $\mathbf{U}''_{\mathsf{id}} \in \{-1, 1\}^{3m\times m}$ and two matrices $\{\mathbf{U}_{\mathsf{id},k}\}_{k\in\{0,1\}} \in \{-1, 1\}^{3m\times m}$, and sets $\mathsf{V}_{\mathsf{id}} = (\mathbf{C}_{\mathsf{id}}, \mathbf{C}'_{\mathsf{id}}, \mathbf{A}''_{\mathsf{id}} = \mathbf{A}_{\mathsf{id}}\mathbf{U}''_{\mathsf{id}}, \{\mathbf{D}_{\mathsf{id},k}\} = \{\mathbf{A}_{\mathsf{id}}\mathbf{U}_{\mathsf{id},k}\}_{k\in\{0,1\}})$. Then, $\mathcal{B}$ computes $\sigma^{(1)} \leftarrow \mathbf{Sign}'(\mathsf{id}, \mathsf{sk}_{\mathsf{id}}^{(1)}, \mathsf{V}_{\mathsf{id}})$ and stores $(\mathsf{id}, \mathsf{V}_{\mathsf{id}}, \sigma^{(1)})$ in L.

Phase2. $\mathcal{B}$ continues the simulation as follows:

(1) Abort the simulation if

$$\boldsymbol{h}_{z_{\mathsf{id}},\boldsymbol{\alpha}_{\mathsf{id}},\boldsymbol{\beta}_{\mathsf{id}}}(\boldsymbol{\tau}) = (h_{z_{\mathsf{id},1},\alpha_{\mathsf{id},1},\beta_{\mathsf{id},1}}(\boldsymbol{\tau}), \cdots, h_{z_{\mathsf{id},u},\alpha_{\mathsf{id},u},\beta_{\mathsf{id},u}}(\boldsymbol{\tau}), 0, \cdots, 0) = \mathbf{0}.$$

(2) Otherwise, compute

$$\mathbf{C}_{\mathsf{id},\boldsymbol{\tau}} \leftarrow \mathsf{PubEval}_{||}^{(\boldsymbol{\tau},\nu_2,t_2)}(\mathbf{C}_{\mathsf{id}}, \mathbf{C}'_{\mathsf{id}}, \boldsymbol{h}'_{\boldsymbol{\tau}})$$

such that $\mathbf{C}_{\mathsf{id},\boldsymbol{\tau}} = \mathbf{A}_{\mathsf{id}}\cdot\mathbf{R}_{\mathsf{id},\boldsymbol{\tau}} + \mathbf{E}_{\mathsf{id},\boldsymbol{\tau}}\cdot\mathbf{G}_{\nu_2 n}$, where

$$\mathbf{R}_{\mathsf{id},\boldsymbol{\tau}} \leftarrow \mathsf{TrapEval}_{||}^{(\boldsymbol{\tau},\nu_2,t_2)}((z_{\mathsf{id}}, \boldsymbol{\alpha}_{\mathsf{id}}, \boldsymbol{\beta}_{\mathsf{id}}), \mathbf{A}_{\mathsf{id}}, \hat{\mathbf{R}}_{\mathsf{id}}, \hat{\mathbf{R}}'_{\mathsf{id}}, \boldsymbol{h}'_{\boldsymbol{\tau}})$$
$$\mathbf{E}_{\mathsf{id},\boldsymbol{\tau}} = [h_{z_{\mathsf{id},1},\alpha_{\mathsf{id},1},\beta_{\mathsf{id},1}}(\boldsymbol{\tau})\mathbf{I}_n|\cdots| h_{z_{\mathsf{id},u},\alpha_{\mathsf{id},u},\beta_{\mathsf{id},u}}(\boldsymbol{\tau})\mathbf{I}_n|\mathbf{0}_n|\cdots|\mathbf{0}_n].$$

(3) Set $\mathbf{A}_{\mathsf{id},\boldsymbol{\tau}} = [\mathbf{A}_{\mathsf{id}}|\mathbf{A}''_{\mathsf{id}} + \mathbf{C}_{\mathsf{id},\boldsymbol{\tau}}]$, for $k \in \{0, 1\}$, use the algorithm $\mathsf{SampleD}^{\mathcal{O}}$ with the trapdoor $\mathbf{T}_{\mathbf{G}_{\nu_2 n}}$ to compute the following:

$$\sigma_k'^{(2)} \leftarrow \mathsf{SampleD}^{\mathcal{O}}(\mathbf{A}_{\mathsf{id}}, -\mathbf{U}''_{\mathsf{id}} - \mathbf{R}_{\mathsf{id},\boldsymbol{\tau}}, \mathbf{E}_{id,\boldsymbol{\tau}}, \mathbf{D}_{\mathsf{id},k} + [k\mathbf{I}_n|\cdots|k\mathbf{I}_n]\mathbf{G}_{ln}, s_2)$$

such that

$$\mathbf{A}_{\mathsf{id},\boldsymbol{\tau}}\cdot\sigma_k'^{(2)} = \mathbf{D}_{\mathsf{id},k} + [k\mathbf{I}_n|\cdots|k\mathbf{I}_n]\mathbf{G}_{ln}.$$

(4) For $u_i \in \{0, 1\}, i \in [l]$, compute $\sigma_{u_i}^{(2)} = \sigma_{u_i}'^{(2)}\mathbf{G}_{ln}^{-1}(\mathbf{T}_i\mathbf{G}_n)$, such that

$$\mathbf{A}_{\mathsf{id},\boldsymbol{\tau}}\cdot\sigma_{u_i}^{(2)} = \mathbf{D}_{\mathsf{id},u_i}\mathbf{G}_{ln}^{-1}(\mathbf{T}_i\mathbf{G}_n) + u_i\mathbf{G}_n.$$

Finally, $\mathcal{B}$ returns the signature $\sigma_i = (\mathsf{V}_{\mathsf{id}}, \sigma^{(1)}, \sigma_{u_i}^{(2)})$.

– Type III forgery:

Phase1. Except for the $\mathbf{A}''_{\mathsf{id}}$ and $\{\mathbf{D}_{\mathsf{id},k}\}_{i\in\{0,1\}}$, $\mathbf{C}_{\mathsf{id}}$, $\mathbf{C}'_{\mathsf{id}}$ are generated in the same way in forgery III as in forgery II, so we omit the same part here.

Let $|Q_1|$ be the upper bound on the number of signing queries. $\mathcal{B}$ randomly selects a tag $\boldsymbol{\tau}^* \in \{\boldsymbol{\tau}_i\}_{i\in[|Q_1|]}$, guesses that the adversary $\mathcal{A}$ will make a Type III forgery on it, and computes $\mathbf{C}_{\mathsf{id},\boldsymbol{\tau}^*}$ by invoking

$$\mathbf{C}_{\mathsf{id},\boldsymbol{\tau}^*} \leftarrow \mathsf{PubEval}_{||}^{(\boldsymbol{\tau}^*,\nu_2,t_2)}(\mathbf{C}_{\mathsf{id}}, \mathbf{C}'_{\mathsf{id}}, \boldsymbol{h}_{\boldsymbol{\tau}^*}).$$

Randomly choose a matrix $\mathbf{U}''_{\mathsf{id}} \in \{-1,1\}^{3m\times m}$, and then, set $\mathbf{A}''_{\mathsf{id}} = \mathbf{A}_{\mathsf{id}}\mathbf{U}''_{\mathsf{id}} - \mathbf{C}_{\mathsf{id},\boldsymbol{\tau}^*}$. For $k \in \{0,1\}$, sample two matrices $\sigma_k'^{(2)} \leftarrow (\mathcal{D}_{\mathbb{Z}^m,s_1})^{4m}$ as "pre-signatures", and let

$$\mathbf{D}_{\mathsf{id},k} = [\mathbf{A}_{\mathsf{id}}|\mathbf{A}_{\mathsf{id}}\mathbf{U}''_{\mathsf{id}}]\sigma_k'^{(2)} - [k\mathbf{I}_n|\cdots|k\mathbf{I}_n]\mathbf{G}_{ln}.$$

Next, set $\mathsf{V}_{\mathsf{id}} = (\mathbf{C}_{\mathsf{id}}, \mathbf{C}'_{\mathsf{id}}, \mathbf{A}''_{\mathsf{id}}, \{\mathbf{D}_{\mathsf{id},k}\}_{k\in\{0,1\}})$, compute $\sigma^{(1)} \leftarrow \mathbf{Sign}'(\mathsf{id}, sk_{\mathsf{id}}^{(1)}, \mathsf{V}_{\mathsf{id}})$, and then store $(\mathsf{id}, \mathsf{V}_{\mathsf{id}}, \sigma^{(1)})$ in L.

Phase2. $\mathcal{B}$ continues the simulation as follows: If the tag $\boldsymbol{\tau} \neq \boldsymbol{\tau}^*$, check if $\boldsymbol{h}_{z_{\mathsf{id}},\boldsymbol{\alpha}_{\mathsf{id}},\boldsymbol{\beta}_{\mathsf{id}}}(\boldsymbol{\tau}) - \boldsymbol{h}_{z_{\mathsf{id}},\boldsymbol{\alpha}_{\mathsf{id}},\boldsymbol{\beta}_{\mathsf{id}}}(\boldsymbol{\tau}^*)$ is equal to $\mathbf{0}$. If so, abort. Otherwise, answer the queries using the trapdoor $\mathbf{T}_{\mathbf{G}_{\nu_2 n}}$ for $\mathbf{G}_{\nu_2 n}$, in a similar fashion as in answering signing queries in Type II queries.

- **Forgery.** From Lemma 4.3, the output of the above games is statistically indistinguishable from the real experiment. Adversary $\mathcal{B}$ receives from $\mathcal{A}$ a forgery tuple $(\mathsf{id}^*, \boldsymbol{\tau}^*, \sigma^*, \mu^*, f^*)$, where $\sigma^* = (\mathsf{V}_{id^*}, \sigma^{*(1)}, \sigma_{f^*}^{*(2)})$.

If the type of forgery outputted by $\mathcal{A}$ is different than the type that $\mathcal{B}$ guessed, then abort the simulation. Otherwise, to generate a solution for the $\mathsf{SIS}_{q,n,m,\beta}$ challenge $\mathbf{A}^* \in \mathbb{Z}_q^{n\times m}$ we distinguish between the two cases:

–**Type II forgery:** In this case, the adversary $\mathcal{A}$ never asked to sign any messages associated with the challenge tag $\boldsymbol{\tau}^*$. Taking as input the parameters $z_{\mathsf{id}^*}, \boldsymbol{\alpha}_{\mathsf{id}^*}, \boldsymbol{\beta}_{\mathsf{id}^*}$ and the tag $\boldsymbol{\tau}^*$, $\mathcal{B}$ compute and output the value

$$\boldsymbol{h}_{z_{\mathsf{id}^*},\boldsymbol{\alpha}_{\mathsf{id}^*},\boldsymbol{\beta}_{\mathsf{id}^*}}(\boldsymbol{\tau}^*) = (h_{z_{\mathsf{id}^*,1},\alpha_{\mathsf{id}^*,1},\beta_{\mathsf{id}^*,1}}(\boldsymbol{\tau}^*), \cdots, h_{z_{\mathsf{id}^*,u},\alpha_{\mathsf{id}^*,u},\beta_{\mathsf{id}^*,u}}(\boldsymbol{\tau}^*), 0, \cdots, 0).$$

If $\boldsymbol{h}_{z_{\mathsf{id}^*},\boldsymbol{\alpha}_{\mathsf{id}^*},\boldsymbol{\beta}_{\mathsf{id}^*}}(\boldsymbol{\tau}^*) \neq 0$, abort this simulation. Otherwise, we have

$$\mathbf{A}_{\mathsf{id}^*,\boldsymbol{\tau}^*} \cdot \sigma_{f^*}^{*(2)} = \mathbf{D}_{f^*}^* + \mu^*\mathbf{G}_n. \tag{2}$$

where $\mathbf{D}_{f^*}^*$ is the public matrix derived with respect to the circuit f^*, and

$$\begin{aligned}\mathbf{A}_{\mathsf{id}^*,\boldsymbol{\tau}^*} &= [\mathbf{A}_{\mathsf{id}^*}|\mathbf{A}''_{\mathsf{id}^*} + \mathbf{C}_{\mathsf{id}^*,\boldsymbol{\tau}^*}] \\ &= \mathbf{A}_{\mathsf{id}^*}[\mathbf{I}_{3m}|\mathbf{U}''_{\mathsf{id}^*} + \mathbf{R}_{\mathsf{id}^*,\boldsymbol{\tau}^*}] \qquad (3)\end{aligned}$$

$$\mathbf{C}_{\mathsf{id}^*,\boldsymbol{\tau}^*} = \mathbf{A}_{\mathsf{id}^*} \cdot \mathbf{R}_{\mathsf{id}^*,\boldsymbol{\tau}^*} + \mathbf{E}_{\mathsf{id}^*,\boldsymbol{\tau}^*} \cdot \mathbf{G}_{\nu_2 n} \tag{4}$$

$$\mathbf{R}_{\mathsf{id}^*,\boldsymbol{\tau}^*} \leftarrow \mathsf{TrapEval}_{||}^{(\boldsymbol{\tau}^*,\nu_2,t_2)}((z_{\mathsf{id}^*}, \boldsymbol{\alpha}_{\mathsf{id}^*}, \boldsymbol{\beta}_{\mathsf{id}^*}), \mathbf{A}_{\mathsf{id}^*}, \hat{\mathbf{R}}_{\mathsf{id}^*}, \hat{\mathbf{R}}'_{\mathsf{id}^*}, \boldsymbol{h}'_{\boldsymbol{\tau}^*}) \tag{5}$$

$$\mathbf{E}_{\mathsf{id}^*,\boldsymbol{\tau}^*} = [h_{z_{\mathsf{id}^*,1},\alpha_{\mathsf{id}^*,1},\beta_{\mathsf{id}^*,1}}(\boldsymbol{\tau}^*)\mathbf{I}_n|\cdots| h_{z_{\mathsf{id}^*,u},\alpha_{\mathsf{id}^*,u},\beta_{\mathsf{id}^*,u}}(\boldsymbol{\tau}^*)\mathbf{I}_n|\mathbf{0}_n|\cdots|\mathbf{0}_n] \tag{6}$$

$$\begin{aligned}\mathbf{A}_{\mathsf{id}^*} &= [\mathbf{A}'_{\mathsf{id}^*}|\mathbf{A}_1] \\ &= [\mathbf{A}^*|\mathbf{A}^*\mathbf{U} + \mathbf{A}^*\mathbf{R}_{\mathsf{id}^*}|\mathbf{A}^*\mathbf{U}_1] \\ &= \mathbf{A}^*[\mathbf{I}_m|\mathbf{U} + \mathbf{R}_{\mathsf{id}^*}|\mathbf{U}_1] \qquad (7)\end{aligned}$$

According to Lemma 4.4, we have

$$\mathbf{D}_{f^*}^* = \mathbf{A}_{\mathsf{id}^*} \cdot \mathbf{U}_{f^*}^* + k\mathbf{G}_n \tag{8}$$

for some small norm matrix $\mathbf{U}_{f^*}^*$ and integer k.

We let $\mathbf{S}_{\mathsf{id}^*} = [\mathbf{I}_m|\mathbf{U} + \mathbf{R}_{\mathsf{id}^*}|\mathbf{U}_1]$ and let $\sigma_{f^*}^{*(2)} = [\mathbf{R}_{f^*,1}^{*\top}|\mathbf{R}_{f^*,2}^{*\top}]^\top$. Then, combining equations (2) − (8), we have

$$\mathbf{A}^*\mathbf{S}_{\mathsf{id}^*}([\mathbf{I}_{3m}|\mathbf{U}''_{\mathsf{id}^*} + \mathbf{R}_{\mathsf{id}^*,\boldsymbol{\tau}^*}]\begin{bmatrix}\mathbf{R}_{f^*,1}^* \\ \mathbf{R}_{f^*,2}^*\end{bmatrix} - \mathbf{U}_{f^*}^*) = (k+\mu^*)\mathbf{G}_n$$

Hence, we have

$$\mathbf{A}^*\mathbf{S}_{id^*}(\mathbf{R}_{f^*,1}^* + (\mathbf{U}''_{id^*} + \mathbf{R}_{id^*,\tau^*})\mathbf{R}_{f^*,2}^* - \mathbf{U}_{f^*}^*) = (k+\mu^*)\mathbf{G}_n$$

Since the matrices $\mathbf{R}, \mathbf{R}' \in \{0,1\}^{m\times m}$, $\hat{\mathbf{R}}_{\mathsf{id}}, \hat{\mathbf{R}}'_{\mathsf{id}} \in \{0,1\}^{3m\times m}$ are chosen uniformly at random, the $\mathbf{B}, \mathbf{B}'$ and $\mathbf{C}_{\mathsf{id}}, \mathbf{C}'_{\mathsf{id}}$ are distributed statistically close to uniform by Lemma 3, the adversary $\mathcal{A}$ obtains no information about the resulting matrix $\mathbf{R}_{\mathsf{id}^*}$ and $\mathbf{R}_{\mathsf{id}^*,\boldsymbol{\tau}^*}$ generated using the algorithm $\mathsf{TrapEval}_{||}$. Therefore, the probability that the term, i.e., $\mathbf{S}_{\mathsf{id}^*}(\mathbf{R}_{f^*,1}^* + (\mathbf{U}''_{\mathsf{id}^*} + \mathbf{R}_{\mathsf{id}^*,\boldsymbol{\tau}^*})\mathbf{R}_{f^*,2}^* - \mathbf{U}_{f^*}^*) = 0$ is negligible. If $k + \mu^* = 0$, then output $\mathbf{S}_{\mathsf{id}^*}(\mathbf{R}_{f^*,1}^* + (\mathbf{U}''_{\mathsf{id}^*} + \mathbf{R}_{\mathsf{id}^*,\boldsymbol{\tau}^*})\mathbf{R}_{f^*,2}^* - \mathbf{U}_{f^*}^*)$ as the solution for SIS-instance, and otherwise, output $\mathbf{S}_{\mathsf{id}^*}(\mathbf{R}_{f^*,1}^* + (\mathbf{U}''_{\mathsf{id}^*} + \mathbf{R}_{\mathsf{id}^*,\boldsymbol{\tau}^*})\mathbf{R}_{f^*,2}^* - \mathbf{U}_{f^*}^*)\mathbf{T}_{\mathbf{G}_n}$ as the solution.

–**Type III (a) forgery:** In this case, the tag $\boldsymbol{\tau}^*$ has been used for generating signature of message μ^* before, but $f^*(\boldsymbol{u}^*) \neq \mu^*$. If the forgery tag is not $\boldsymbol{\tau}^*$ which is selected at the sign phase in the Type III forgery, then abort this simulation. Otherwise, $\mathcal{B}$ uses the "pre-signatures" presampled in the sign phase, which has been queried under tag $\boldsymbol{\tau}^*$ to honestly compute $\sigma_{f^*}^{(2)}$ by

$$\sigma_{f^*}^{(2)} = [\mathbf{R}_{f^*,1}^\top|\mathbf{R}_{f^*,2}^\top]^\top \leftarrow \mathsf{Eval}(\mathsf{id}^*, \boldsymbol{\tau}^*, \boldsymbol{u}^*, \boldsymbol{\sigma}^*, f^*),$$

Per correctness, we have

$$\mathbf{A}_{\mathsf{id}^*,\boldsymbol{\tau}^*}\sigma_{f^*}^{(2)} = \mathbf{D}_{f^*} + f^*(\boldsymbol{u}^*)\mathbf{G}_n \tag{9}$$

By the validity of forgery, we have

$$\mathbf{A}_{\mathsf{id}^*,\boldsymbol{\tau}^*}\sigma_{f^*}^{*(2)} = \mathbf{D}_{f^*}^* + \mu^*\mathbf{G}_n \tag{10}$$

We let $\sigma_{f^*}^{*(2)} = [\mathbf{R}_{f^*,1}^{*\top}|\mathbf{R}_{f^*,2}^{*\top}]^\top$, since $\sigma_{f^*}^{*(2)}$ is a forged signature on message $f^*(\boldsymbol{u}^*) \neq \mu^*$, and it follows that $\sigma_{f^*}^{(2)} - \sigma_{f^*}^{*(2)} \neq 0$. Subtracting Eq.(9) by Eq.(10), and note that $\mathbf{D}_{f^*} = \mathbf{D}_{f^*}^*$. Thus, we have

$$\mathbf{A}_{\mathsf{id}^*,\boldsymbol{\tau}^*}\begin{bmatrix}\mathbf{R}_{f^*,1} - \mathbf{R}_{f^*,1}^* \\ \mathbf{R}_{f^*,2} - \mathbf{R}_{f^*,2}^*\end{bmatrix} = (f^*(\boldsymbol{u}^*) - \mu^*)\mathbf{G}_n$$

Expanding $\mathbf{A}_{id^*,\boldsymbol{\tau}^*}$ from its definition and rearranging, we can obtain

$$\mathbf{A}^*\mathbf{S}_{\mathsf{id}^*}((\mathbf{R}_{f^*,1}-\mathbf{R}^*_{f^*,1})+\mathbf{U}''_{\mathsf{id}^*}(\mathbf{R}_{f^*,2}-\mathbf{R}^*_{f^*,2}) = (f^*(\boldsymbol{u}^*)-\mu^*))\mathbf{G}_n$$

Output $\mathbf{S}_{\mathsf{id}^*}((\mathbf{R}_{f^*,1}-\mathbf{R}^*_{f^*,1})+\mathbf{U}''_{\mathsf{id}^*}(\mathbf{R}_{f^*,2}-\mathbf{R}^*_{f^*,2}))\mathbf{T}_{\mathbf{G}_n}$ as the SIS solution for matrix $\mathbf{A}^*$.

–**Type III (b) forgery:** In this case, the tag $\boldsymbol{\tau}^*$ has been used for generating signature of message μ^* before, but $\sigma^{(2)}_{f^*} \neq \sigma^{*(2)}_{f^*}$, where $\sigma^{(2)}_{f^*}$ is generated honestly using the "pre-signatures" presampled in the sign phase of the Type III forgery and implementing **Eval** algorithm. If the forgery tag is not $\boldsymbol{\tau}^*$, which is selected at the sign phase in the Type III forgery, then abort this simulation. Otherwise, similar to Type III (a) forgery, the adversary $\mathcal{B}$ gets the above (9) and (10), but in this case, it holds that $\mathbf{D}_{f^*}=\mathbf{D}^*_{f^*}, f^*(\boldsymbol{u}^*)=\mu^*$. Subtracting Eq. (9) by (10) and rearranging, we can get

$$\mathbf{A}^*\mathbf{S}_{\mathsf{id}^*}((\mathbf{R}_{f^*,1}-\mathbf{R}^*_{f^*,1})+\mathbf{U}''_{\mathsf{id}^*}(\mathbf{R}_{f^*,2}-\mathbf{R}^*_{f^*,2}))=0.$$

Since the matrices $\mathbf{R},\mathbf{R}' \in \{0,1\}^{m\times m}$ are chosen uniformly at random, the $\mathbf{B},\mathbf{B}'$ are distributed statistically close to uniform by Lemma 3, and the adversary $\mathbf{A}$ obtains no information about the resulting matrix $\mathbf{R}_{\mathsf{id}^*}$ generated using the algorithm $\mathsf{TrapEval}_{||}$. Furthermore, the matrix $\sigma^{(2)}_{f^*} = [\mathbf{R}^\top_{f^*,1}|\mathbf{R}^\top_{f^*,2}]^\top$ is derived from $\{\sigma'^{(2)}_k\}_{k\in\{0,1\}}$, which are presampled randomly from $(\mathcal{D}_{\mathbb{Z}^m,s_1})^{4m}$ in sign phase of the Type III forgery, the public matrices $\{\mathbf{D}_{\mathsf{id}^*,k}\}_{k\in\{0,1\}}$ that are distributed statistically close to uniform, and the deterministic matrices $\{\mathsf{G}^{-1}_{ln}(\mathbf{T}_i\mathsf{G}_n)\}_{i\in[l]}$ and f^*, Hence, the adversary $\mathcal{A}$ gains no information about $\sigma^{(2)}_{f^*} = [\mathbf{R}^\top_{f^*,1}|\mathbf{R}^\top_{f^*,2}]^\top$. Finally, the matrices $\mathbf{U},\mathbf{U}_1 \in \{-1,1\}^{m\times m}$ and $\mathbf{U}''_{\mathsf{id}^*} \in \{-1,1\}^{3m\times m}$ are chosen uniformly at random in the setup and sign phase, respectively. Therefore, the probability that the

$$\mathbf{S}_{\mathsf{id}^*}((\mathbf{R}_{f^*,1}-\mathbf{R}^*_{f^*,1})+\mathbf{U}''_{\mathsf{id}^*}(\mathbf{R}_{f^*,2}-\mathbf{R}^*_{f^*,2}))=0$$

is negligible, then output $\mathbf{S}_{\mathsf{id}^*}((\mathbf{R}_{f^*,1}-\mathbf{R}^*_{f^*,1})+\mathbf{U}''_{\mathsf{id}^*}(\mathbf{R}_{f^*,2}-\mathbf{R}^*_{f^*,2}))$ as the solution.

Lemma 21: Let $\{\mathsf{mpk},\mathsf{sk}_{id},\boldsymbol{\sigma}\}$ be the output in real execution and $\{\mathsf{mpk}^*,\mathsf{sk}^*_{id},\boldsymbol{\sigma}^*\}$ be the output in simulated execution by the algorithms Setup, Extract and Sign respectively. We show that the two distributions are statistically indistinguishable.

Proof: In type I forgery, it is obvious that the output in real execution is indistinguishable from that in simulated execution, so we only prove the case in type II and type III.

Type II forgery. We first argue the claim for the type II forgery simulation. The differences in the execution of the algorithm can be summarized as follows:

- **Setup.**
  – In real Setup, matrix $\mathbf{A}^* \in \mathbb{Z}_q^{n\times m}$ is sampled with the trapdoor $\mathbf{T}_{\mathbf{A}^*}$ by running algorithm TrapGen. In the simulated Setup algorithm, $\mathbf{A}^*$ is chosen uniformly at random by the SIS generator without its trapdoor.
  – In real Setup, matrices $(\mathbf{A},\mathbf{A}_1,\mathbf{B},\mathbf{B}')$ are chosen uniformly at random from $\mathbb{Z}_q^{n\times m}$. In the simulated Setup algorithm, the matrices $\mathbf{A}=\mathbf{A}^*\mathbf{U}, \mathbf{A}_1=\mathbf{A}^*\mathbf{U}_1$ for two uniformly random matrices $\mathbf{U},\mathbf{U}_1 \in \{-1,1\}^{m\times m}$, and $\mathbf{B}=\mathbf{A}^*\mathbf{R}+\mathbf{E}, \mathbf{B}'=\mathbf{A}^*\mathbf{R}'+\mathbf{E}'$ for two uniformly random matrices $\mathbf{R},\mathbf{R}' \in \{-1,1\}^{m\times m}$.
- **Extract.**
  In the real Extract algorithm, every $\mathbf{G}$-trapdoor $\boldsymbol{r}_{\mathsf{id}}$ for identity matrix $\mathbf{A}_{\mathsf{id}}$ is obtained by running the SampleLeft algorithm and a trapdoor for matrix $\mathbf{A}^*$. Whereas, in the simulated Extract algorithm, the $\mathbf{G}$-trapdoor $\boldsymbol{r}_{\mathsf{id}}$ is obtained by using the $\mathsf{SampleD}^{\mathcal{O}}$ algorithm and a trapdoor for $\mathbf{G}_{v_1 n}$.
- **Sign.**
  – In the real Sign algorithm, the related information about all previously returned identities contained in list L, i.e. matrices $(\mathbf{C}_{\mathsf{id}},\mathbf{C}'_{\mathsf{id}},\mathbf{A}''_{\mathsf{id}},\{\mathbf{D}_{\mathsf{id},k}\}_{k\in\{0,1\}})$ are chosen uniformly at random. Whereas, in the simulated Sign algorithm, the matrix $\mathbf{A}''_{\mathsf{id}}=\mathbf{A}_{\mathsf{id}}\mathbf{U}''_{\mathsf{id}}$ for a random matrix $\mathbf{U}''_{\mathsf{id}} \in \{-1,1\}^{3m\times m}$, $\{\mathbf{D}_{\mathsf{id},k}=\mathbf{A}_{\mathsf{id}}\mathbf{U}_{\mathsf{id},k}\}_{\in\{0,1\}}$ for two random matrices $\{\mathbf{U}_{\mathsf{id},k}\}_{\in\{0,1\}} \in \{-1,1\}^{3m\times m}$, and $\mathbf{C}_{\mathsf{id}}=\mathbf{A}_{\mathsf{id}}\hat{\mathbf{R}}_{\mathsf{id}}+\mathbf{E}_{\mathsf{id}}, \mathbf{C}'_{\mathsf{id}}=\mathbf{A}_{\mathsf{id}}\hat{\mathbf{R}}'_{\mathsf{id}}+\mathbf{E}'_{\mathsf{id}}$ for random matrices $\hat{\mathbf{R}}_{\mathsf{id}},\hat{\mathbf{R}}'_{\mathsf{id}} \in \{-1,1\}^{3m\times m}$.
  – In the real Sign algorithm, generates every signatures σ_i by using algorithm SampleLeft associated with the trapdoor $\mathbf{T}_{\mathbf{A}_{\mathsf{id}}}$. Whereas, the simulated Sign algorithm generates the signatures using algorithm $\mathsf{SampleD}^{\mathcal{O}}$ with the trapdoor $\mathbf{T}_{\mathbf{G}_{v_2 n}}$.

We now argue that the distribution $(\mathbf{A}^*,\mathbf{A},\mathbf{A}_1,\mathbf{B},\mathbf{B}')$ is statistically indistinguishable in real and simulated execution. Observe that by Lemma 10, for sufficiently large $m=\Omega(n\log q)$, $\mathbf{A}^* \in \mathbb{Z}_q^{n\times m}$ is distributed statistically close to uniform. Then, by Lemma 3, it holds that

$$(\mathbf{A}^*,\mathbf{A},\mathbf{A}_1,\mathbf{B},\mathbf{B}') \approx (\mathbf{A}^*,\mathbf{A}^*\mathbf{U},\mathbf{A}^*\mathbf{U}_1,\mathbf{A}^*\mathbf{R}+\mathbf{E},\mathbf{A}^*\mathbf{R}'+\mathbf{E}')$$

for random matrices $\mathbf{A}^*,\mathbf{A},\mathbf{A}_1,\mathbf{B},\mathbf{B}'$ from $\mathbb{Z}_q^{n\times m}$.

Therefore, we conclude that mpk in real is indistinguishable from mpk^* in simulated experiment.

Now, consider a secret key extract oracle query on an identity id, by Lemma 9, for sufficiently large Gaussian parameter s, the output of algorithm SampleLeft and $\mathsf{SampleD}^{\mathcal{O}}$ are both distributed statistically close to $\mathcal{D}_{\Lambda_q^{\mathbf{G}-\mathbf{A}_1}(\mathbf{A}'_{\mathsf{id}}),s}$. Therefore, every secret key $\mathsf{sk}_{\mathsf{id}}$ in the real Extract algorithm is statistically indistinguishable from the secret key $\mathsf{sk}^*_{\mathsf{id}}$ in the simulated Extract algorithm.

For the signing algorithm in both executions, according to the expression of $\mathbf{A}_{\mathsf{id}}$, it is easy to know the fact that $\mathbf{A}_{\mathsf{id}} \in \mathbb{Z}^{n\times 3m}$ is distributed statistically close to uniform, so by Lemma 3, it holds that

$$(\mathbf{C}_{\mathsf{id}},\mathbf{C}'_{\mathsf{id}},\mathbf{A}''_{\mathsf{id}},\{\mathbf{D}_{\mathsf{id},k}\}_{k\in\{0,1\}}) \approx (\mathbf{A}_{\mathsf{id}}\hat{\mathbf{R}}_{\mathsf{id}}+\mathbf{E}_{\mathsf{id}},\mathbf{A}_{\mathsf{id}}\hat{\mathbf{R}}'_{\mathsf{id}}+\mathbf{E}'_{\mathsf{id}},\mathbf{A}_{\mathsf{id}}\mathbf{U}''_{\mathsf{id}},\{\mathbf{A}_{\mathsf{id}}\mathbf{U}_{\mathsf{id},k}\}_{\in\{0,1\}}).$$

In addition, by Lemma 9, for sufficiently large Gaussian parameter s, the output of SampleLeft and $\mathsf{SampleD}^{\mathcal{O}}$ are both distributed statistically close to $\mathcal{D}_{\Lambda_q(\mathbf{A}_{\mathsf{id},\boldsymbol{\tau}}),s}$. Therefore, we prove the claim for Type II forgery.

Type III forgery. We start the analysis for Type III by summarizing the differences between real and simulated executions:

- **Setup.**
  – In the real Setup, matrix $\mathbf{A}^* \in \mathbb{Z}_q^{n\times m}$ is sampled with the trapdoor $\mathbf{T}_{\mathbf{A}^*}$ by running algorithm TrapGen. In the simulated Setup algorithm, $\mathbf{A}^*$ is chosen uniformly at random by the SIS generator.
  – In real Setup, matrices $(\mathbf{A}, \mathbf{A}_1, \mathbf{B}, \mathbf{B}')$ are chosen uniformly at random from $\mathbb{Z}_q^{n\times m}$. However, in the simulated Setup algorithm, the matrices $\mathbf{A} = \mathbf{A}^*\mathbf{U}$, $\mathbf{A}_1 = \mathbf{A}^*\mathbf{U}_1$ for two uniformly random matrices $\mathbf{U}, \mathbf{U}_1 \in \{-1, 1\}^{m\times m}$, and $\mathbf{B} = \mathbf{A}^*\mathbf{R} + \mathbf{E}$, $\mathbf{B}' = \mathbf{A}^*\mathbf{R}' + \mathbf{E}'$ for two uniformly random matrices $\mathbf{R}, \mathbf{R}' \in \{-1, 1\}^{m\times m}$.
- **Extract.**
  In the real Extract algorithm, every $\boldsymbol{r}_{\mathsf{id}}$ is obtained by running the SampleLeft algorithm and a trapdoor for matrix $\mathbf{A}^*$. Whereas, in the simulated Extract algorithm, the $\boldsymbol{r}_{\mathsf{id}}$ is obtained by using the $\mathsf{SampleD}^{\mathcal{O}}$ algorithm and a trapdoor for $\mathbf{G}_{v_1 n}$.
- **Sign.**
  – In the real Sign algorithm, the matrices $(\mathbf{C}_{\mathsf{id}}, \mathbf{C}'_{\mathsf{id}}, \mathbf{A}''_{\mathsf{id}}, \{\mathbf{D}_{\mathsf{id},k}\}_{k\in\{0,1\}})$ are chosen uniformly at random. Whereas, in the simulated Sign algorithm, the matrix $\mathbf{A}''_{\mathsf{id}} = \mathbf{A}_{\mathsf{id}}\mathbf{U}''_{\mathsf{id}} - \mathbf{C}_{\mathsf{id},\boldsymbol{\tau}^*}$ for a random matrix $\mathbf{U}''_{\mathsf{id}} \in \{-1, 1\}^{3m\times m}$, $\mathbf{C}_{\mathsf{id}} = \mathbf{A}_{\mathsf{id}}\hat{\mathbf{R}}_{\mathsf{id}} + \mathbf{E}_{\mathsf{id}}$, $\mathbf{C}'_{\mathsf{id}} = \mathbf{A}_{\mathsf{id}}\hat{\mathbf{R}}'_{\mathsf{id}} + \mathbf{E}'_{\mathsf{id}}$ for random matrices $\hat{\mathbf{R}}_{\mathsf{id}}, \hat{\mathbf{R}}'_{\mathsf{id}} \in \{-1, 1\}^{3m\times m}$, and $\{\mathbf{D}_{\mathsf{id},k} = [\mathbf{A}_{\mathsf{id}}|\mathbf{A}_{\mathsf{id}}\mathbf{U}''_{\mathsf{id}}]\sigma_k'^{(2)} - [k\mathbf{I}_n|\cdots|k\mathbf{I}_n]\mathbf{G}_{ln}\}$ for two matrices $\{\sigma_k'^{(2)} = [\mathbf{R}'^{\top}_{k,1}|\mathbf{R}'^{\top}_{k,1}]^{\top}\}_{k\in\{0,1\}}$ sampled randomly from $(\mathcal{D}_{\mathbb{Z}^m,s})^{4m}$.
  – In the real Sign algorithm, every signatures $\sigma_i^{(2)}$ is obtained by using algorithm SampleLeft. In the simulated Sign algorithm, for $\boldsymbol{\tau} \neq \boldsymbol{\tau}^*$, the signature $\sigma_i^{(2)}$ is generated in forgery III in the same way as in forgery II; for $\boldsymbol{\tau} = \boldsymbol{\tau}^*$, the signature $\sigma_i^{(2)}$ is pre-generated in the phase of setting identity related information in sign algorithm. So we only need to show that the distributions in both Sign algorithm are statistically indistinguishable for $\boldsymbol{\tau} = \boldsymbol{\tau}^*$.

We now argue that the distribution $(\mathbf{A}^*, \mathbf{A}, \mathbf{A}_1, \mathbf{B}, \mathbf{B}')$ is statistically indistinguishable in real and simulated executions. Observe that by Lemma 10, for sufficiently large $m = \Omega(n \log q)$, $\mathbf{A}^* \in \mathbb{Z}_q^{n\times m}$ is distributed statistically close to uniform. Then, by Lemma 3, it holds that

$$(\mathbf{A}^*, \mathbf{A}, \mathbf{A}_1, \mathbf{B}, \mathbf{B}') \approx (\mathbf{A}^*, \mathbf{A}^*\mathbf{U}, \mathbf{A}^*\mathbf{U}_1, \mathbf{A}^*\mathbf{R} + \mathbf{E}, \mathbf{A}^*\mathbf{R}' + \mathbf{E}')$$

for random matrices $\mathbf{A}^*, \mathbf{A}, \mathbf{A}_1, \mathbf{B}, \mathbf{B}'$ from $\mathbb{Z}_q^{n\times m}$.

Therefore, we conclude that mpk in real is indistinguishable from mpk^* in simulated experiment.

Now, consider a Extract oracle query on an identity id, by Lemma 9, for sufficiently large Gaussian parameter s, the output of algorithm SampleLeft and $\mathsf{SampleD}^{\mathcal{O}}$ are both distributed statistically close to $\mathcal{D}_{\Lambda_q^{\mathbf{G}-\mathbf{A}_1}(\mathbf{A}'_{\mathsf{id}}),s}$.

TABLE 1. IBFHS parameters and example setting.

Parameter	Setting	Definition
s_1	$3.8\sqrt{n\log q}\cdot\omega\sqrt{\log 4m}$	SampleLeft width.
s_2	$(n^{4+\epsilon})m^{3.5}$	$\mathsf{SampleD}^{\mathcal{D}}$ width.
s	$n^{4+\epsilon}m^{4.5}$	$\mathsf{SampleD}^{\mathcal{D}}$ and $\mathsf{SampleD}^{\mathcal{D}}$ width.
ν_1	$\log\log\lvert\mathsf{id}\rvert$	Upper bound of repetitions.
t_1	$2\log 2\lvert\mathsf{id}\rvert$	The z range in partitioning function.
ν_2	$\log\log\lvert\boldsymbol{\tau}\rvert$	Upper bound of repetitions.
t_2	$2\log 2\lvert\boldsymbol{\tau}\rvert$	The z range in partitioning function.

Therefore, every secret key $\mathsf{sk}_{\mathsf{id}}$ in the real Extract algorithm is statistically indistinguishable from the secret key $\mathsf{sk}^*_{\mathsf{id}}$ in the simulated Extract algorithm.

Finally, we consider a signing oracle, by Lemma 3, it holds that

$$\begin{aligned}&(\mathbf{A}''_{\mathsf{id}}, \mathbf{C}_{\mathsf{id}}, \mathbf{C}'_{\mathsf{id}}, \{\mathbf{D}_{\mathsf{id},k}\}_{k\in\{0,1\}})\\ &\quad\approx (\mathbf{A}_{\mathsf{id}}\mathbf{U}''_{\mathsf{id}} - \mathbf{C}_{\mathsf{id},\boldsymbol{\tau}^*}, \mathbf{A}_{\mathsf{id}}\hat{\mathbf{R}}_{\mathsf{id}} + \mathbf{E}_{\mathsf{id}}, \mathbf{A}_{\mathsf{id}}\hat{\mathbf{R}}'_{\mathsf{id}} + \mathbf{E}'_{\mathsf{id}},\\ &\qquad\{[\mathbf{A}_{\mathsf{id}}|\mathbf{A}_{\mathsf{id}}\mathbf{U}''_{\mathsf{id}}]\sigma_k'^{(2)} - [k\mathbf{I}_n|\cdots|k\mathbf{I}_n]\mathbf{G}_{ln}\}_{k\in\{0,1\}}.\end{aligned}$$

for random matrices $\mathbf{A}''_{\mathsf{id}}, \mathbf{C}_{\mathsf{id}}, \mathbf{C}'_{\mathsf{id}} \in \mathbb{Z}_q^{n\times m}$ and $\{\mathbf{D}_{\mathsf{id},k}\}_{k\in\{0,1\}} \in \mathbb{Z}_q^{n\times m}$.

As we analyzed above, for the signature $\sigma_i^{(2)}$ in signing algorithm in both executions, we need to consider the case $\boldsymbol{\tau} = \boldsymbol{\tau}^*$, which is the challenge tag. By Lemma 9, for the Gaussian parameter s set appropriately, the output of the algorithm SampleLeft is distributed statistically close to $\mathcal{D}_{\Lambda_q^{\mathbf{D}_{\mathsf{id},k}+[k\mathbf{I}_n|\cdots|k\mathbf{I}_n]\mathbf{G}_{ln}}(\mathbf{A}_{\mathsf{id},\boldsymbol{\tau}}),s}$. Put together, the master public key, secret key for identity and signature are indistinguishable from real, we prove the claim for Type III forgery.

Lemma 22: Let f be an arbitrary circuit taking as input at most l bits and outputting one bit. Let id be a arbitrary identity and $\mathbf{A}_{\mathsf{id}}$ be an "identity matrix". Fix the public matrices for all input wires as $\mathbf{D}_{\mathsf{id},i} = \mathbf{A}_{\mathsf{id}}\mathbf{U}_{\mathsf{id},i} + k\mathbf{G}_n$ for $\mathbf{U}_{\mathsf{id},i} \in \{-1, 1\}^{3m\times m}$, where $k \in \{0, 1\}$. For each gate $g = (u, v, w)$, assume input public matrices $\mathbf{D}_{\mathsf{id},u}, \mathbf{D}_{\mathsf{id},v}$ are fixed and let

$$\mathbf{D}_{\mathsf{id},w} = \mathbf{D}_{\mathsf{id},v}\cdot\tilde{\mathbf{D}}_{\mathsf{id},u} + \mathbf{G}$$

where $\tilde{\mathbf{D}}_{\mathsf{id},u} = \mathbf{G}_n^{-1}(-\mathbf{D}_{\mathsf{id},u})$. Then, the public matrix associated with the output wire of the circuit f is of the form $\mathbf{A}_{\mathsf{id}}\mathbf{U}_f + k\mathbf{G}$, where $\mathbf{U}_f$ is of low norm and $k \in \{0, 1\}$.

Proof: We proceed by the induction on the circuit depth:

Base case. For each input wires, the associated public key is $\mathbf{D}_{\mathsf{id},k} = \mathbf{A}_{\mathsf{id}}\mathbf{U}_{\mathsf{id},k}$, where $\mathbf{U}_{\mathsf{id},i} \in \{-1, 1\}^{3m\times m}$. Thus, it certainly satisfied the form $\mathbf{A}_{\mathsf{id}}\mathbf{U}_f + k\mathbf{G}$ with $k = 0$.

Induction step. Now, we consider a gate $g = (u, v, w)$ where $\mathbf{D}_{\mathsf{id},u} = \mathbf{A}_{\mathsf{id}}\mathbf{U}_{\mathsf{id},u} + i_u\mathbf{G}$ and $\mathbf{D}_{\mathsf{id},v} = \mathbf{A}_{\mathsf{id}}\mathbf{U}_{\mathsf{id},v} + i_v\mathbf{G}$ for integers i_u, i_u. Then, we have

$$\begin{aligned}&\mathbf{D}_{\mathsf{id},w}\\ &= \mathbf{D}_{\mathsf{id},v}\cdot\tilde{\mathbf{D}}_{\mathsf{id},u} + \mathbf{G}_n\\ &= (\mathbf{A}_{\mathsf{id}}\mathbf{U}_{\mathsf{id},v} + i_v\mathbf{G}_n)\mathbf{G}_n^{-1}(-\mathbf{D}_{\mathsf{id},u}) + \mathbf{G}_n\\ &= \mathbf{A}_{\mathsf{id}}\mathbf{U}_{\mathsf{id},v}\mathbf{G}_n^{-1}(-\mathbf{D}_{\mathsf{id},u}) + i_v\mathbf{G}_n\mathbf{G}_n^{-1}(-\mathbf{D}_{\mathsf{id},u}) + \mathbf{G}_n\\ &= \mathbf{A}_{\mathsf{id}}\mathbf{U}_{\mathsf{id},v}\mathbf{G}_n^{-1}(-\mathbf{D}_{\mathsf{id},u}) - i_v\mathbf{A}_{\mathsf{id}}\mathbf{U}_{\mathsf{id},u} - i_u i_v\mathbf{G}_n + \mathbf{G}_n\\ &= \mathbf{A}_{\mathsf{id}}(\mathbf{U}_{\mathsf{id},v}\mathbf{G}_n^{-1}(-\mathbf{D}_{\mathsf{id},u}) - i_v\mathbf{U}_{\mathsf{id},u}) + i_w\mathbf{G}_n\end{aligned}$$

TABLE 2. Comparisons between related schemes.

Items	[16]	[15]	Ours
ID-PKC	No	Yes	Yes
adaptively strong-unforgeable	No	No	Yes
Multi-Data	Yes	No	Yes
public key sizes	$1 \times \mathbb{Z}_q^{n\times m}$	Random strings in $\{0,1\}^\lambda$	Random strings in $\{0,1\}^\lambda$
secret key sizes	$1 \times \mathbb{Z}_q^{m\times m}$	$1 \times \mathbb{Z}_q^{(m_0+m_1)\times m}$	$1 \times \mathbb{Z}_q^{2m\times m}$
public parameters sizes	$N \times \mathbb{Z}_q^{m\times m}$	$N \times \mathbb{Z}_q^{m\times m}$	0
maximum noise-level	$O(m^d\beta)$	$O(4^d m\beta)$	$O(4^d m\beta)$
signature sizes	$1 \times \mathbb{Z}_q^{m\times m}$	$1 \times \mathbb{Z}_q^{m\times m}$	$5 \times \mathbb{Z}_q^{n\times m} + 1 \times \mathbb{Z}_q^{2m\times m}$

Hence,by induction we obtain that $\mathbf{D}_f = \mathbf{A}_{\mathsf{id}}\mathbf{U}_f + k\mathbf{G}_n$ for some small norm matrix $\mathbf{U}_f$.

Next, we calculate the size of the matrix $\mathbf{U}_f$. For a gate $g = (u, v, w)$, we have $\mathbf{U}_{\mathsf{id},w} = \mathbf{U}_{\mathsf{id},v}\mathbf{G}_n^{-1}(-\mathbf{D}_{\mathsf{id},u}) - i_v\mathbf{U}_{\mathsf{id},u}$. Hence, $\|\mathbf{U}_{\mathsf{id},w}\|_\infty = O(s_1s_2m^3)$. By induction, we obtain the size of the output matrix $\|\mathbf{U}_f\|_\infty = O(s_1s_2m^{O(d_{max})})$.

A. PARAMETER SELECTION

There are two instantiations for the same construction: one is for $\mathsf{polylog}(\lambda)$-depth circuit, another is for $\mathsf{poly}(\lambda)$-depth circuit. We set the corresponding lattice parameters as follows:

The lattice parameters for the $\mathsf{polylog}(\lambda)$-depth circuit with standard SIS assumption are set as

$$n = \mathsf{poly}(\lambda), l = \mathsf{poly}(\lambda), m = O(nld_{max}),$$
$$q = n^{O(d_{max})}, \beta = 2^{\mathsf{poly}(\lambda)}, q > \beta.$$

The lattice parameters for the $\mathsf{poly}(\lambda)$-depth circuit with sub-exponential SIS assumption are set as

$$n = 2^{\lambda^\epsilon}(0 < \epsilon < 1), l = 2^{\mathsf{poly}(\lambda)}, m = \tilde{\Theta}(nld_{max}),$$
$$q = n^{\tilde{\Theta}(d_{max})}, \beta = 2^{\mathsf{poly}(\lambda)}, q > \beta.$$

We set other public parameters as follows.

These values in Table 1 are chosen in order to satisfy the following constraints:

- For $\mathsf{SampleLeft}$, we know $\tilde{\mathbf{T}}_{\mathbf{A}} \leq 3.8\sqrt{n\cdot\log q}$ by Lemma 8 and Lemma 9, so require that the sampling width s_1 satisfies $3.8\sqrt{n\log q}\cdot\omega\sqrt{\log 4m}$.
- For $\mathsf{SampleD}^{\mathcal{O}}$, we know $\tilde{\mathbf{T}}_{\mathbf{G}_{vn}} \leq \sqrt{l^2+1}$ and $s_2 \geq O((n^{4+\epsilon})m^{3.5}$ by Lemma 9.
  Hence, we need the joint sampling width s to, satisfy the indistinguishability of the output of $\mathsf{SampleLeft}$ and $\mathsf{SampleD}^{\mathcal{O}}$ algorithms
  $$s \geq n^{4+\epsilon}m^{4.5}.$$
- To ensure correctness, we need to establish the maximum size of the evaluated signatures.
  – For Type II forgery, the size of the output matrix as the solution for $\mathsf{SIS}_{q,n,m,\beta}$ assumption is bounded as
  $$\|[\mathbf{I}_m|\mathbf{U} + \mathbf{R}_{\mathsf{id}^*}|\mathbf{U}_1](\mathbf{R}^*_{f^*,1} + (\mathbf{U}''_{\mathsf{id}^*} + \mathbf{R}_{\mathsf{id}^*,\tau^*})\mathbf{R}^*_{f^*,2} - \mathbf{U}^*_{f^*})\mathbf{T}_{\mathbf{G}_n}\| \leq O(m^{O(d_{max})})$$
  due to $\|\mathbf{U}\|, \|\mathbf{U}_1\| \leq 12\sqrt{2m}$, $\|\mathbf{U}''_{\mathsf{id}^*}\| \leq 12\sqrt{6m}$ by Lemma 1, $\|\mathbf{R}_{\mathsf{id}^*}\|, \|\mathbf{R}_{\mathsf{id}^*,\tau^*}\| \leq \nu n^2m^3z^{O(z)}\log^3 q$ by Remark 11, $\|\mathbf{R}^*_{f^*,1}\|, \|\mathbf{R}^*_{f^*,2}\|, \|\mathbf{U}^*_{f^*}\| \leq s_1s_2m^{\frac{3}{2}}(m+1)^{d_{max}}$ by lemma 17 and Lemma 22, and $\|\mathbf{T}_{\mathbf{G}_n}\| \leq \sqrt{5}$ by Lemma 8.
  – For Type III forgery, the size of the output matrix as the solution for $\mathsf{SIS}_{q,n,m,\beta}$ assumption is bounded as
  $$\|[\mathbf{I}_m|\mathbf{U} + \mathbf{R}_{\mathsf{id}^*}|\mathbf{U}_1]((\mathbf{R}_{f^*,1} - \mathbf{R}^*_{f^*,1}) + \mathbf{U}''_{\mathsf{id}^*}(\mathbf{R}_{f^*,2} - \mathbf{R}^*_{f^*,2}))\mathbf{T}_{\mathbf{G}_n}\| \leq O(m^{O(d_{max})})$$
  due to $\|\mathbf{U}\|, \|\mathbf{U}_1\| \leq 12\sqrt{2m}$, $\|\mathbf{U}''_{\mathsf{id}^*}\| \leq 12\sqrt{6m}$ by Lemma 1, $\|\mathbf{T}_{\mathbf{G}_n}\| \leq \sqrt{5}$, $\|\mathbf{R}_{f^*,1}\|, \|\mathbf{R}_{f^*,2}\|\|\mathbf{R}^*_{f^*,1}\|, \|\mathbf{R}^*_{f^*,2}\| \leq s_1s_2m^{\frac{3}{2}}(m+1)^{d_{max}}$ by lemma 17 and Lemma 22.

B. PERFORMANCE ANALYSIS

In this subsection, we compare our scheme with two previously developed FHS schemes: [15] and [16]. The compared items include the following: ID-PKC(whether a scheme is based on a identity-based cryptosystem), adaptively strong-unforgeable, Multi-Data(whether a signer can sign many different datasets of arbitrary size), public key sizes, secret key sizes, public parameters sizes, maximum noise-level and signature sizes. The details are presented in Table 2. The notation $x \times \mathbb{Z}_q^{n\times m}$ represents the number of matrices in $\mathbb{Z}_q^{n\times m}$ is x, N represents the the maximum data-size. The basic parameter m, n, q are all determined by algorithm $\mathsf{TrapGen}$ in these three schemes, which have the same parameter setting.

According to Table 2, we can see that comparing to scheme in [16], the maximum noise-level of our scheme and the scheme in [15] roughly reduces from $O(m^d\beta)$ to $O(4^dm\beta)$. Although the signature size of our scheme is larger than that of the other two schemes, our scheme can achieve adaptive strong-unforgeability without public parameters, especially the other two schemes scheme have large public parameters, with a size that is linearly related to the size of the dataset.

VI. CONCLUSION

In this work, we construct the first leveled adaptively secure IBFHS schemes without additional public parameters,, which can be used to sign many different datasets, thereby positively answering the open question regarding the construction of a leveled IBFHS scheme with short public parameters in [15]. The signatures in our IBFHS scheme are succinct, in the sense that the size of every evaluated signature depends only logarithmically on the size of the input dataset and our

construction allows fast amortized verification of a circuit on many different datasets. Furthermore, we prove that our IBFHS scheme is strongly-unforgeable against chosen identity and message attacks under the small integer solution (SIS) assumption in standard lattices. However, our IBFHS scheme only supports leveled homomorphic computations. It remains an open problem to design an IBFHS scheme to get rid of the leveled aspect, that is, there is no prior constraint on the depth of the circuits that can be evaluated.

ACKNOWLEDGMENT

The authors would like to thank the anonymous reviewers for their valuable comments.

REFERENCES

[1] C. Gentry, "Fully homomorphic encryption using ideal lattices," in *Proc. STOC*, 2009, pp. 169–178, doi: 10.1145/1536414.1536440.

[2] I. Chillotti, N. Gama, M. Georgieva, and M. Izabachène, "TFHE: Fast fully homomorphic encryption over the torus," *J. Cryptol.*, vol. 33, no. 1, pp. 34–91, Jan. 2020, doi: 10.1007/s00145-019-09319-x.

[3] S. Halevi, Y. Polyakov, and V. Shoup, "An improved RNS variant of the BFV homomorphic encryption scheme," in *Proc. CT-RSA*, in Lecture Notes in Computer Science, vol. 11405, 2019, pp. 83–105, doi: 10.1007/978-3-030-12612-4_5.

[4] Y. T. Kalai, R. Raz, and R. D. Rothblum, "How to delegate computations: The power of no-signaling proofs," in *Proc. STOC*, 2014, pp. 485–494, doi: 10.1145/2591796.2591809.

[5] Y. T. Kalai, R. Raz, and R. D. Rothblum, "Delegation for bounded space," in *Proc. 45th Annu. ACM Symp. Theory Comput. (STOC)*, 2013, pp. 565–574, doi: 10.1145/2488608.2488679.

[6] M. Liu, Y. Wu, R. Xue, and R. Zhang, "Verifiable outsourcing computation for modular exponentiation from shareable functions," *Cluster Comput.*, vol. 23, no. 1, pp. 43–55, Mar. 2020, doi: 10.1007/s10586-019-02930-4.

[7] J. Duan, J. Zhou, and Y. Li, "Secure and verifiable outsourcing of large-scale nonnegative matrix factorization (NMF)," *IEEE Trans. Services Comput.*, early access, Apr. 15, 2019, doi: 10.1109/TSC.2019.2911282.

[8] D. Boneh and D. M. Freeman, "Linearly homomorphic signatures over binary fields and new tools for lattice-based signatures," in *Proc. PKC*, 2011, pp. 1–16, doi: 10.1007/978-3-642-19379-8_19.

[9] F. Wang, S. Shi, and C. Wang, "Leveled lattice-based linearly homomorphic signature scheme in the standard model for network coding," in *Proc. FCS*, 2019, pp. 84–94, doi: 10.1007/978-981-15-0818-9_6.

[10] A. Shamir, "Identity-based cryptosystems and signature schemes," in *Proc. CRYPTO*, in Lecture Notes in Computer Science, vol. 196, 1984, pp. 47–53, doi: 10.1007/3-540-39568-7_5.

[11] Y. Zhang, Y. Jiang, B. Li, and M. Zhang, "An efficient identity-based homomorphic signature scheme for network coding," in *Proc. EIDWT*, 2017, pp. 524–531, doi: 10.1007/978-3-319-59463-7_52.

[12] S. Sadrhaghighi and S. Khorsandi, "An identity-based digital signature scheme to detect pollution attacks in intra-session network coding," in *Proc. ISCISC*, 2016, pp. 7–12, doi: 10.1109/ISCISC.2016.7736444.

[13] Q. Lin, H. Yan, Z. Huang, W. Chen, J. Shen, and Y. Tang, "An ID-based linearly homomorphic signature scheme and its application in blockchain," *IEEE Access*, vol. 6, pp. 20632–20640, 2018, doi: 10.1109/ACCESS.2018.2809426.

[14] J. Chang, H. Ma, A. Zhang, M. Xu, and R. Xue, "RKA security of identity-based homomorphic signature scheme," *IEEE Access*, vol. 7, pp. 50858–50868, 2019, doi: 10.1109/ACCESS.2019.2908244.

[15] F. Wang, K. Wang, B. Li, and Y. Gao, "Leveled strongly-unforgeable identity-based fully homomorphic signatures," in *Proc. ISC*, 2015, pp. 42–60, doi: 10.1007/978-3-319-23318-5_3.

[16] S. Gorbunov, V. Vaikuntanathan, and D. Wichs, "Leveled fully homomorphic signatures from standard lattices," in *Proc. STOC*, 2015, pp. 469–477, doi: 10.1145/2746539.2746576.

[17] J. Alperin-Sheriff and C. Peikert, "Faster bootstrapping with polynomial error," in *Proc. CRYPTO*, 2014, pp. 297–314, doi: 10.1007/978-3-662-44371-2_17.

[18] D. Boneh, C. Gentry, S. Gorbunov, S. Halevi, V. Nikolaenko, G. Segev, V. Vaikuntanathan, and D. Vinayagamurthy, "Fully key-homomorphic encryption, arithmetic circuit ABE and compact garbled circuits," in *Proc. EUROCRYPT*, 2014, pp. 533–556, doi: 10.1007/978-3-642-55220-5_30.

[19] D. A. Barrington, "Bounded-width polynomial-size branching programs recognize exactly those languages in NC^1," *J. Comput. Syst. Sci*, vol. 38, no. 1, pp. 150–164, 1989, doi: 10.1016/0022-0000(89)90037-8.

[20] F. Luo, F. Wang, K. Wang, and K. Chen, "A more efficient leveled strongly-unforgeable fully homomorphic signature scheme," *Inf. Sci.*, vol. 480, pp. 70–89, Apr. 2019, doi: 10.1016/j.ins.2018.12.025.

[21] D. Apon, X. Fan, and F. Liu, "Vector encoding over lattices and its applications," IACR Cryptol. ePrint Arch., Las Vegas, NV, USA, Tech. Rep. 2017/455, 2017, p. 455.

[22] D. Boneh, D. Freeman, J. Katz, and B. Waters, "Signing a linear subspace: Signature schemes for network coding," in *Proc. PKC*, 2009, pp. 68–87, doi: 10.1007/978-3-642-00468-1_5.

[23] R. Johnson, D. Molnar, D. X. Song, and D. A. Wagner, "Homomorphic signature schemes," in *Proc. CT-RSA*, 2002, pp. 244–262, doi: 10.1007/3-540-45760-7_17.

[24] W. Chen, H. Lei, and K. Qi, "Lattice-based linearly homomorphic signatures in the standard model," *Theor. Comput. Sci.*, vol. 634, pp. 47–54, Jun. 2016, doi: 10.1016/j.tcs.2016.04.009.

[25] Q. Lin, J. Li, Z. Huang, W. Chen, and J. Shen, "A short linearly homomorphic proxy signature scheme," *IEEE Access*, vol. 6, pp. 12966–12972, 2018, doi: 10.1109/ACCESS.2018.2809684.

[26] L. Schabhüser, J. Buchmann, and P. Struck, "A linearly homomorphic signature scheme from weaker assumptions," in *Proc. IMACC*, 2017, pp. 261–279, doi: 10.1007/978-3-319-71045-7_14.

[27] S.-Y.-R. Li, R. W. Yeung, and N. Cai, "Linear network coding," *IEEE Trans. Inf. Theory*, vol. 49, no. 2, pp. 371–381, Feb. 2003, doi: 10.1109/TIT.2002.807285.

[28] X. Liu, J. Huang, Y. Wu, and G. Zong, "A privacy-preserving signature scheme for network coding," *IEEE Access*, vol. 7, pp. 109739–109750, 2019, doi: 10.1109/ACCESS.2019.2933870.

[29] X. Hu, S. Zheng, J. Gong, G. Cheng, G. Zhang, and R. Li, "Enabling linearly homomorphic signatures in network coding-based named data networking," in *Proc. 14th Int. Conf. Future Internet Technol.*, Aug. 2019, pp. 1–4, doi: 10.1145/3341188.3341191.

[30] F. Armknecht, J.-M. Bohli, G. Karame, and W. Li, "Outsourcing proofs of retrievability," *IEEE Trans. Cloud Comput.*, early access, Aug. 15, 2018, doi: 10.1109/TCC.2018.2865554.

[31] D. Boneh and D. Freeman, "Homomorphic signatures for polynomial functions," in *Proc. AEUROCRYPT*, 2011, pp. 149–168, doi: 10.1007/978-3-642-20465-4_10.

[32] D. Catalano, D. Fiore, and B. Warinschi, "Homomorphic signatures with efficient verification for polynomial functions," in *Proc. CRYPTO*, 2014, pp. 371–389, doi: 10.1007/978-3-662-44371-2_21.

[33] D. Wichs, "Leveled fully homomorphic signatures from standard lattices," IACR Cryptol. ePrint Arch., Las Vegas, NV, USA, Tech. Rep. 2014/451, 2014, p. 451. [Online]. Available: http://eprint.iacr.org/2014/451

[34] S. Gorbunov and V. Vaikuntanathan, "Leveled fully homomorphic signatures from lattices," IACR Cryptol. ePrint Arch., Las Vegas, NV, USA, Tech. Rep. 2014/463, 2014, p. 463. [Online]. Available: http://eprint.iacr.org/2014/463

[35] D. Boneh and X. Boyen, "Efficient selective identity-based encryption without random oracles," *J. Cryptol.*, vol. 24, no. 4, pp. 659–693, Oct. 2011, doi: 10.1007/s00145-010-9078-6.

[36] X. Boyen, X. Fan, and E. Shi, "Adaptively secure fully homomorphic signatures based on lattices," Cryptol. ePrint Arch., Las Vegas, NV, USA, Tech. Rep. 2014/916Q, 2014, p. 916.

[37] X. Boyen, "Lattice mixing and vanishing trapdoors: A framework for fully secure short signatures and more," in *Proc. PKC*, 2010, pp. 499–517, doi: 10.1007/978-3-642-13013-7_29.

[38] J. Chang, Y. Ji, M. Xu, and R. Xue, "General transformations from single-generation to multi-generation for homomorphic message authentication schemes in network coding," *Future Gener. Comput. Syst.*, vol. 91, pp. 425–426, Feb. 2019, doi: 10.1016/j.future.2018.08.039.

[39] J. Chang, B. Shao, A. Zhang, G. Bian, Y. Ji, and M. Xu, "Security analysis of an efficient null space-based homomorphic MAC scheme against tag pollution attacks in RLNC," *IEEE Access*, vol. 7, pp. 88393–88398, 2019, doi: 10.1109/ACCESS.2019.2926401.

[40] P. Bai, W. Zhang, X. A. Wang, Y. Liu, H. Yang, and C. Shan "Homomorphic authentication based on rank-based Merkle hash tree," in *Proc. EIDWT*, in Lecture Notes in Computer Science, vol. 17, 2018, pp. 841–848, doi: 10.1007/978-3-319-75928-9_76.

[41] S. Agrawal and D. Boneh, "Homomorphic MACs: MAC-based integrity for network coding," in *Proc. ACNS*, in Lecture Notes in Computer Science, vol. 5536, 2009, pp. 292–305, doi: 10.1007/978-3-642-01957-9_18.

[42] R. Gennaro and D. Wichs, "Fully homomorphic message authenticators," in *Proc. ASIACRYPT*, in Lecture Notes in Computer Science, vol. 8270, 2013, pp. 301–320, doi: 10.1007/978-3-642-42045-0_16.

[43] D. Catalano, D. Fiore, R. Gennaro, and L. Nizzardo, "Generalizing homomorphic MACs for arithmetic circuits," in *Proc. PKC*, in Lecture Notes in Computer Science, vol. 8383, 2014, pp. 538–555, doi: 10.1007/978-3-642-54631-0_31.

[44] B. Parno, J. Howell, C. Gentry, and M. Raykova, "Pinocchio: Nearly practical verifiable computation," in *Proc. IEEE Symp. Secur. Privacy*, May 2013, pp. 238–252, doi: 10.1109/SP.2013.47.

[45] S. Zhang, H. Li, Y. Dai, J. Li, M. He, and R. Lu, "Verifiable outsourcing computation for matrix multiplication with improved efficiency and applicability," *IEEE Internet Things J.*, vol. 5, no. 6, pp. 5076–5088, Dec. 2018, doi: 10.1109/JIOT.2018.2867113.

[46] D. F. Aranha and E. Pagnin, "The simplest multi-key linearly homomorphic signature scheme," in *Proc. LATINCRYPT*, in Lecture Notes in Computer Science, vol. 11774, 2019, pp. 280–300, doi: 10.1007/978-3-030-30530-7_14.

[47] D. Fiore and E. Pagnin, "Matrioska: A compiler for multi-key homomorphic signatures," in *Proc. SCN*, 2018, pp. 43–62, doi: 10.1007/978-3-319-98113-0_3.

[48] R. W. F. Lai, R. K. H. Tai, H. W. H. Wong, and S. S. M. Chow, "Multi-key homomorphic signatures unforgeable under insider corruption," in *Proc. ASIACRYPT*, 2018, pp. 465–492, doi: 10.1007/978-3-030-03329-3_16.

[49] L. Schabhüser, D. Butin, and J. Buchmann, "Context hiding multi-key linearly homomorphic authenticators," in *Proc. CT-RSA*, 2019, pp. 493–513, doi: 10.1007/978-3-030-12612-4_25.

[50] T. Shang, X.-J. Zhao, C. Wang, and J.-W. Liu, "Quantum homomorphic signature," *Quantum Inf. Process.*, vol. 14, pp. 393–410, Jan. 2015, doi: 10.1007/s11128-014-0853-4.

[51] T. Shang, Z. Pei, R. Chen, and J. Liu, "Quantum homomorphic signature with repeatable verification," *Comput. Mater. Continua*, vol. 159, no. 1, pp. 149–165, 2019, doi: 10.32604/cmc.2019.05360.

[52] Z.-Z. Li, G. Xu, L.-B. Chen, and Y. Yang, "Secure quantum network coding based on quantum homomorphic message authentication," *Quantum Inf. Process.*, vol. 18, no. 1, pp. 1–21, 2019, doi: 10.1007/s11128-018-2127-z.

[53] S. Agrawal, D. Boneh, and X. Boyen, "Efficient lattice (H)IBE in the standard model," in *Proc. EUROCRYPT*, 2010, pp. 553–572, doi: 10.1007/978-3-642-13190-5_28.

[54] C. Gentry, C. Peikert, and V. Vaikuntanathan, "Trapdoors for hard lattices and new cryptographic constructions," in *Proc. STOC*, 2008, pp. 197–206, doi: 10.1145/1374376.1374407.

[55] D. Micciancio and C. Peikert, "Trapdoors for lattices: Simpler, tighter, faster, smaller," in *Proc. EUROCRYPT*, 2012, pp. 700–718, doi: 10.1007/978-3-642-29011-4_41.

[56] D. Micciancio and O. Regev, "Worst-case to average-case reductions based on Gaussian measures," *SIAM J. Comput.*, vol. 37, no. 1, pp. 267–302, 2007, doi: 10.1137/S0097539705447360.

[57] C. Gentry, A. Sahai, and B. Waters, "Homomorphic encryption from learning with errors: Conceptually-simpler, asymptotically-faster, attribute-based," in *Proc. CRYPTO*, in Lecture Notes in Computer Science, vol. 8042, 2013, pp. 75–92, doi: 10.1007/978-3-642-40041-4_5.

CAIFEN WANG received the Ph.D. degree in cryptography from the School of Communication Engineering, Xidian University, in 2003. She is currently a Professor with Shenzhen Technology University. Her main research interests include cryptography and information security, in particular, applied cryptography and security in cloud computing. She has been selected as the Director of the China Cryptography Society, and a member of the Special Committee of Cryptography Algorithms.

BIN WU received the B.E. degree in mathematics and applied mathematics and the M.S. degree in homology algebra from Northwest Normal University, China, in 2008 and 2018, respectively, where she is currently pursuing the Ph.D. degree. The focus of her research has been on cryptography and information security.

HAILONG YAO received the Ph.D. degree in computer cryptography from Northwest Normal University, Lanzhou, China, in 2020. His main research interests include public key cryptography and security in distributed systems.