sensors

MDPI

Article

Message Integration Authentication in the Internet-of-Things via Lattice-Based Batch Signatures

Xiuhua Lu [1,2], Wei Yin [1], Qiaoyan Wen [1], Kaitai Liang [3], Liqun Chen [3] and Jiageng Chen [4,*]

1 State Key Laboratory of Networking and Switching Technology, Beijing University of Posts and Telecommunications, Beijing 100876, China; luxiuhua2011@bupt.edu.cn (X.L.); yinwei24005@bupt.edu.cn (W.Y.); wqy@bupt.edu.cn (Q.W.)
2 Faculty of Mathematics and Information Science, Langfang Normal University, Langfang 065000, China
3 Department of Computer Science, University of Surrey, Guildford GU2 7XH, UK; k.liang@surrey.ac.uk (K.L.); liqun.chen@surrey.ac.uk (L.C.)
4 School of Computer Science, Central China Normal University, Wuhan 430079, China
* Correspondence: chinkako@gmail.com

Received: 23 October 2018; Accepted: 13 November 2018; Published: 20 November 2018

Abstract: The internet-of-things (also known as IoT) connects a large number of information-sensing devices to the Internet to collect all kinds of information needed in real time. The reliability of the source of a large number of accessed information tests the processing speed of signatures. Batch signature allows a signer to sign a group of messages at one time, and signatures' verification can be completed individually and independently. Therefore, batch signature is suitable for data integration authentication in IoT. An outstanding advantage of batch signature is that a signer is able to sign as many messages as possible at one time without worrying about the size of signed messages. To reduce complexity yielded by multiple message signing, a binary tree is usually leveraged in the construction of batch signature. However, this structure requires a batch residue, making the size of a batch signature (for a group of messages) even longer than the sum of single signatures. In this paper, we make use of the intersection method from lattice to propose a novel generic method for batch signature. We further combine our method with hash-and-sign paradigm and Fiat–Shamir transformation to propose new batch signature schemes. In our constructions, a batch signature does not need a batch residue, so that the size of the signature is relatively smaller. Our schemes are securely proved to be existential unforgeability against adaptive chosen message attacks under the small integer solution problem, which shows great potential resisting quantum computer attacks.

Keywords: IoT; message integration authentication; batch signature; tree structure; intersection method; lattice

1. Introduction

IoT connects all kinds of objects with the Internet, through various sensing technologies and various means of communication, to achieve remote monitoring and other purposes [1–4]. Because of large numbers of nodes, wide sources of information and fast updating of information, information authentication processing is very stressful, which brings forward a new research topic for digital signature.

Digital signature was firstly defined and designed in [5–7]. This security mechanism allows a message owner to put digital "stamp" on a message to declare the corresponding ownership. Since its introduction, digital signature has been widely employed in many real-world applications, e.g., authentication [8], message integrity check [9], electronic voting, electronic property ownership proof (cryptocurrencies—https://bitcoin.org/en/) and other cloud-based applications [10–12]. Due to

various of construction techniques, there are many variants of digital signature systems by far, e.g., El-Gamal [13], RSA-based [7], DSA and ECDSA. When it comes to the environment of IoT, batch signature, which is a variant of conventional digital signature, is a good choice.

1.1. Batch Signature

The notion of batch signature was firstly proposed by Fiat [14] in CRYPTO 1989. It allows a valid user to sign many messages with almost the cost of one signature operation. In other words, batch signature scheme could sign multiple messages simultaneously. Like carbon paper, someone only needs to sign the top file once by inserting each file in the middle of the carbon paper, all the documents are signed, and each message can be independently verified by recipients. This cryptographic primitive has greatly improved the efficiency of signing a large number of messages.

Following Fiat's seminal work, a lot of works on batch signature have been proposed. In 1996, M'Raïhi and Naccache [15] gave a batch exponentiation strategy, and applied it to the batch generation of fixed-g-based signatures. In 1999, Pavlovski and Boyd [16] presented a batch signature scheme based on binary tree structure. Binary tree structure is a general construction to transform a common signature algorithm into a batch signature algorithm. In addition, Cheng et al. [17] and Korkmaz [18] analysed the efficiency of existed batch signatures independently.

Besides theoretical construction, there are many more scenarios to apply batch signature technology. In 1999, Boyd et al. [19] proposed an efficient electronic cash using batch signatures. In 2008, Youn et al. [20] applied batch signature in imbalanced communication. We find batch signature is also indispensable in IoT and blockchain. In IoT, when messages from multiple sensor nodes are imported into the host, batch signature of messages is a good way to improve the efficiency of signature. In blockchain, multiple transactions could be handled simultaneously in one-block-generated time. We may save time and space cost by using batch signature scheme.

Faced with a large number of application requirements, the theoretical research of batch signature is not perfect. There are some defects about the existed batch signature, for example, the limited number of signed messages, dependence of signature verification on batch residue and the risk of anti-quantum algorithm attack.

1.2. Lattice-Based Signature

The above constructions are based on the traditional number theory assumptions. According to Shor's results [21], they can not resist the quantum computer's attack. In the aspect of anti-quantum, lattice-based cryptography is a hot spot for cryptologists, due to the following three advantages. Firstly, large integer factorization and discrete logarithm problems have been proven to be unable to resist quantum computer's attacks, meanwhile, there is no quantum algorithm that could solve hard problems in lattice. Secondly, cryptographic schemes based on the difficulty assumptions of the average case lattice problems can be reduced to the difficulty assumptions of the worst case lattice problems. It means that the security of cryptographic schemes built on average case lattice problems depends on the worst case lattice problems. The majority of public key cryptosystems are lack of this feature. Thirdly, most of the operations in the lattice are linear operations, so that lattice-based cryptographic schemes have potential computational efficiency.

Lattice-based cryptography has achieved many results. Ajtai [22] proposed the small integer solution problem, known as the SIS problem, in 1996. It is an average case problem hard to solve for appropriate parameter settings, and its difficulty is based on worst case lattice hard problems. The SIS problem, as well as its extension, the inhomogeneous small integer solution problem ISIS, forms the foundation of lattice-based signature schemes.

The most important theoretical breakthrough of lattice-based signature began with the signature scheme in [23]. The main structure of this signature scheme includes a trapdoor generation algorithm and preimage sampleable algorithm; these two algorithms are both with relatively large computational complexity, which hinders the practicability of signature schemes.

In order to solve the efficiency problem of signature schemes, cryptologists have considered the issue from many different perspectives. Alwen and Peikert [24] showed the techniques to get better trapdoor at a faster speed. Micciancio and Peikert [25] proposed a different structure, converted the general lattice trapdoor generation algorithm into a simple lattice trapdoor generation algorithm, and designed a more efficient trapdoor generation algorithm. As a by-product of this new algorithm, the efficiency of preimage sampleable algorithm has also been greatly improved. Therefore, the signature scheme in [25] has better efficiency and security.

Signature schemes in [23,25] have the same construction idea and both belong to the hash-and-sign paradigm. In 2012, Lyubashevsky [26] followed the Fiat–Shamir transformation, managed to avoid the use of trapdoor generation algorithm and preimage sampleable algorithm, and constructed more efficient signature schemes using matrix-vector multiplications and rejecting samplings. These signature schemes make the lattice-based signature schemes practical. Since then, lattice-based signature has continued with more and more contributions, but the core idea still follows the above mentioned signature schemes from [23,25,26].

1.3. Our Contributions

In this paper, we propose lattice-based batch signature schemes. Our batch signature schemes remove the batch residue in [19], which makes our batch signature has the same length as one ordinary signature.

1. We propose lattice-based batch signature schemes for the first time. Our schemes possess a general property, that is, our construction can be combined with any existing lattice-based signature scheme.
2. The technique we use is an extension of the intersection method from [27]. The intersection method is as follows: for $n-$dimensional integer lattices Λ_1 and Λ_2 such that $\Lambda_1 + \Lambda_2 = \mathbb{Z}^n$ and $\Lambda_1 \bigcap \Lambda_2 \neq \phi$, there exists a short vector $\mathbf{e}$, which belongs to $\mathbf{v}_1 + \Lambda_1 \cap \mathbf{v}_2 + \Lambda_2$ and can be viewed as a signature of $\mathbf{v}_1 \in \mathbb{Z}^n$ and $\mathbf{v}_2 \in \mathbb{Z}^n$.

   We demonstrate this technique with a concrete example in terms of $k \geq 2$. In detail, let $\Lambda_1 = p_1\mathbb{Z}^n$, $\Lambda_2 = p_2\mathbb{Z}^n, \cdots, \Lambda_k = p_k\mathbb{Z}^n$ with k primes $p_1, p_2, \cdots, p_k$. Because $p_1, p_2, \cdots, p_k$ are different primes, $p_1\mathbb{Z}^n + p_2\mathbb{Z}^n + \cdots + p_k\mathbb{Z}^n = \mathbb{Z}^n$ and $p_1\mathbb{Z}^n \cap p_2\mathbb{Z}^n \cap \cdots \cap p_k\mathbb{Z}^n = p_1p_2\cdots p_k\mathbb{Z}^n \neq \phi$. Therefore, for k messages $\mathbf{v}_1, \mathbf{v}_2, \cdots, \mathbf{v}_k \in \mathbb{Z}^n$, there exists a short vector $\mathbf{e} \in \mathbf{v}_1 + p_1\mathbb{Z}^n \cap \mathbf{v}_2 + p_2\mathbb{Z}^n \cap \cdots \cap \mathbf{v}_k + p_k\mathbb{Z}^n$, which binds $\mathbf{v}_1, \mathbf{v}_2, \cdots, \mathbf{v}_k$ and can be viewed as their batch signature.
3. With the intersection method as core technique, we give two batch signature schemes based on hash-and-sign paradigm and Fiat–Shamir transformation, as well as a lattice-based batch signature scheme based on binary tree.

1.4. Organization

Our paper is organized as follows. First, we give some basic definitions and facts about lattice-based cryptography in Section 2. Then, we describe batch signature scheme definition and security in Section 3. In Section 4, we give lattice-based batch signature scheme based on binary tree. In Section 5, we propose lattice-based batch signature scheme based on hash-and-sign paradigm and the intersection method. In Section 6, we demonstrate lattice-based batch signature scheme based on Fiat–Shamir transformation and the intersection method. In Section 7, we present the comparison of our schemes with other lattice-based batch signatures, then describe batch signature's application to IoT. Finally, we conclude the paper in Section 8.

2. Preliminaries

We make use of standard asymptotic notations in our paper. For any function $f(n)$ and $g(n)$ with positive real value set $\mathbb{R}$ as range, $f = O(g)$ means that there are constants a and b such that $f(n) \leq ag(n)$ for all $n \geq b$; $f = o(g)$ if and only if $\lim_{n\to\infty} f(n)/g(n) = 0$; $f = \omega(g)$ if and only

if $g = o(f)$; $f(n)$ is negligible if and only if $f = o(1/g)$ for any polynomial $g(n) = n^c$; $f(n)$ is overwhelming if and only if $1 - f(n)$ is negligible.

Definition 1. *$\mathcal{D}_{\mathbb{Z}^m,s,\mathbf{c}}$ is the discrete Gaussian distribution in $\mathbb{Z}^m$, its center is $\mathbf{c}$ and Gaussian parameter is s. If the center is vector $\mathbf{0}$, $\mathbf{0}$ may be omitted. If $\mathbf{e} \longleftarrow \mathcal{D}_{\mathbb{Z}^m,s}$, its Euclidean norm is $\|\mathbf{e}\| \leq s\sqrt{m}$ with overwhelming probability [23].*

Definition 2. *Trapdoor generation algorithm $TrapGen(n, q, m)$ inputs n, q and m, where n is an integer, $q \geq 3$ is an odd, and $m = \lceil 6n \log q \rceil$ is the minimum integer not less than $6n \log q$. The algorithm outputs a pair $(\mathbf{A} \in \mathbb{Z}_q^{n\times m}, \mathbf{T} \in \mathbb{Z}^{m\times m})$ such that $\mathbf{A}$ is statistically close to a uniform random matrix in $\mathbb{Z}_q^{n\times m}$, $\mathbf{T}$ is a basis for $\Lambda_q^{\perp}(\mathbf{A})$ satisfying $\|\widetilde{\mathbf{T}}\| \leq O(\sqrt{n \log q})$ and $\|\mathbf{T}\| \leq O(n \log q)$ with overwhelming probability. Here, $\widetilde{\mathbf{T}}$ is the Gram-Schmidt orthogonalization matrix of $\mathbf{T}$, $\|\mathbf{T}\|$ denotes the largest Euclidean norm of the column vectors in matrix $\mathbf{T}$ [24].*

Definition 3. *Let $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$, $\mathbf{T}$ is a basis for $\Lambda_q^{\perp}(\mathbf{A})$, and $s \geq \|\widetilde{\mathbf{T}}\| \cdot \omega(\sqrt{\log m})$. Then for $\mathbf{u} \in \mathbb{Z}_q^n$, preimage sampleable algorithm $SamplePre(\mathbf{A}, \mathbf{T}, \mathbf{u}, s)$ samples $\mathbf{x}$ satisfying $\|\mathbf{x}\| \leq s\sqrt{m}$ and $\mathbf{Ax} = \mathbf{u} \bmod q$ [23].*

Definition 4. *Small integer solution (SIS) [23]*

SIS problem is defined as: for integer q, real β and matrix $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$, search an integer vector $\mathbf{e} \in \mathbb{Z}^m$ satisfying $\mathbf{Ae} = \mathbf{0} \bmod q$, $\|\mathbf{e}\| \leq \beta$ and $\mathbf{e} \neq \mathbf{0}$.

Definition 5. *The intersection of lattice Λ_1 and lattice Λ_2 is not empty, and $\Lambda_1 + \Lambda_2 = \mathbb{Z}^n$ for element wise addition. $\mathbf{v}_1, \mathbf{v}_2 \in \mathbb{Z}^n$ are the coset representatives of Λ_1 and Λ_2, respectively. Then there exists a vector $\mathbf{e} \in \mathbb{Z}^n$ such that $\mathbf{e} = \mathbf{v}_1 \bmod \Lambda_1$ and $\mathbf{e} = \mathbf{v}_2 \bmod \Lambda_2$. This result can be generalized to multiple lattices [27].*

Definition 6. *Target Collision Resistant (TCR) Hash function [28]*

Let $h : \{0,1\}^n \to \{0,1\}^m$ is a collision-resistant hash function if it satisfies the following properties:

- *(length-compressing): $m < n$*
- *(hard to find collisions): For all PPT A, there exists a negligible function ϵ such that for all security parameters $n \in \mathbb{N}$,*

$$Pr[(x_0, x_1) \leftarrow A(1^n, h) : x_0 \neq x_1 \cap h(x_0) = h(x_1)] \leqslant \epsilon(n)$$

3. System Definition and Threat Model

In this section, we give batch signature definitions of generic algorithms and security, which divide into two parts.

3.1. Definition of Batch Signature System

Batch signature can use a signing action to complete the signing of a number of different messages, and the verification of individual message is independent. Besides, the system setup algorithm and key generation algorithm in batch signature scheme are as same as that of ordinary signature scheme.

- **Setup**(λ): Inputting security parameter λ, this algorithm determines necessary system public parameters PP.
- **KeyGen**(λ): With security parameter λ and system parameters PP as above, this algorithm provides public verification key vk and secret signing key sk.
- **Sign**($sk, \{\varpi_1, \cdots, \varpi_k\}$): Given signing key sk and messages set $\{\varpi_1, \cdots, \varpi_k\}$, this algorithm computes batch signature e.

- **Verify**$(vk, \varpi_j, e, j = 1, \cdots, k)$: Given message ϖ_j and its signature e associated with verification key vk, this algorithm tells whether the j-th message has gained valid authentication, and outputs 1 if the answer is yes, otherwise outputs 0.

3.2. Threat Model

Batch signature scheme should also satisfy existential unforgeability against adaptive chosen message attacks (EUF-CMA). We introduce a challenger $\mathcal{C}$ and an adversary $\mathcal{A}$ interacting with each other in the next game, to describe batch signature scheme's security.

- **Initialization**: In this period, challenger $\mathcal{C}$ executes algorithms **Setup** and **KeyGen**, provides system public parameters PP and public verification key vk for adversary $\mathcal{A}$.
- **Signing queries**: In this stage, adversary $\mathcal{A}$ selects a set of messages $(\varpi_1, \varpi_2, \cdots, \varpi_k)$, sends the messages' set to challenger $\mathcal{C}$ for the associated signature. Challenger $\mathcal{C}$ invokes **Sign** algorithm, returns the result to adversary $\mathcal{A}$. Adversary $\mathcal{A}$ may repeat the query polynomial times in his favorite manner.
- **Forgery**: Once adversary $\mathcal{A}$ terminates signing queries, he offers a new message-signature pair $(\varpi_1^*, \varpi_2^*, \cdots, \varpi_k^*, e)$.

If message-signature pair $(\varpi_1^*, \varpi_2^*, \cdots, \varpi_k^*, e)$ is valid and has not been queried, adversary $\mathcal{A}$ wins the game.

Theorem 1. *Batch signature scheme is existential unforgeability against adaptive chosen message attacks(EUF-CMA), if for all adversary $\mathcal{A}$ with polynomial bounded computational power, the probability of he wins above game is negligible.*

4. Lattice-Based Batch Signature with Binary Tree

4.1. Proposed Construction

In this part, we combine the signature scheme in [23] and the structure of binary tree in [19], propose the first lattice-based batch signature scheme. The scheme includes the following steps, and the Figure 1 shows the binary tree for message processing.

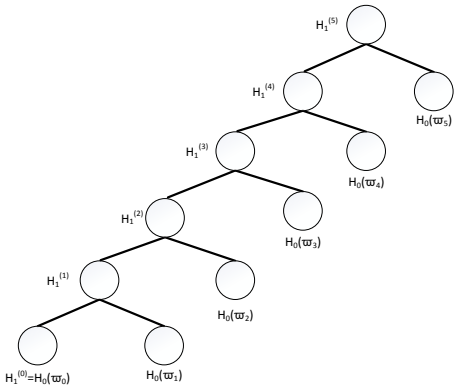

Figure 1. The Schematic of the Binary Tree.

- **Setup**(λ): In this stage, system parameters are provided with knowledge of security parameter λ.
  1. n is a polynomial of λ, $q \geq 3$ is a polynomial of n, $m = \lceil 6n \log q \rceil$, $t = O(\sqrt{n \log q})$.
  2. k is the number of messages to batch sign, and $s \geq t \cdot \omega(\sqrt{\log m})$ is the Gaussian parameter.
  3. $H_0 : \{0,1\}^* \longrightarrow \mathbb{Z}_q^n$ and $H_1 : \mathbb{Z}_q^{2n} \longrightarrow \mathbb{Z}_q^n$ are two collision resistant hash functions.
- **KeyGen**(λ): With system parameters as above, public verification key vk and secret signing key sk are obtained as follows. Invoke trapdoor generation algorithm TrapGen(n, q, m) to get a uniform and random matrix $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$, and the short basis $\mathbf{T} \in \mathbb{Z}^{m \times m}$ for lattice $\Lambda_q^{\perp}(\mathbf{A})$ with $\|\widetilde{\mathbf{T}}\| \leq t$.

Finally output $vk = \mathbf{A}$, $sk = \mathbf{T}$.

- **Sign**$(sk, \{\varpi_0, \cdots, \varpi_{k-1}\} \in \{\{0,1\}^*\}^k)$: Given $sk = \mathbf{T}$ and the set of messages $\{\varpi_0, \cdots, \varpi_{k-1}\} \in \{\{0,1\}^*\}^k$, the following steps lead to batch signature on such messages.

  1. Compute $H_0(\varpi_0)$, $H_0(\varpi_1)$,$\cdots$, $H_0(\varpi_{k-1})$, let $H_1^{(0)} = H_0(\varpi_0)$, and execute for-loop as follows.
     for $i = 1$ to $k-1$:
     $H_1^{(i)} = H_1(H_1^{(i-1)} \| H_0(\varpi_i))$
     $i = i + 1$
  2. Compute $\mathbf{e}$ =SamplePre$(\mathbf{A}, \mathbf{T}, H_1^{(k-1)}, s)$.
  3. For $\varpi_i, i = 0, \cdots, k-1$, compute its brother B_i. Firstly, $B_0 = (H_0(\varpi_1), \mathfrak{R})$, the left are shown in the next for-loop.
     for $i = 1$ to $k-1$:
     $B_i = (H_1^{(i-1)}, \mathfrak{L})$
     $i = i + 1$
     Here, for ϖ_i's brother B_i, its first entry denotes ϖ_i's brother note, its second entry denotes the brother locates on ϖ_i's left($\mathfrak{L}$) or right($\mathfrak{R}$).
  4. For $\varpi_i, i = 0, \cdots, k-1$, compute its residue R_i. At first,
     $R_0 = \{(H_0(\varpi_1), \mathfrak{R}), (H_0(\varpi_2), \mathfrak{R}), \cdots, (H_0(\varpi_{k-1}), \mathfrak{R})\}$
     $= \{B_0, (H_0(\varpi_2), \mathfrak{R}), \cdots, (H_0(\varpi_{k-1}), \mathfrak{R})\}$,
     the left are shown in the next for-loop.
     for $i = 1$ to $k-2$:
     $R_i = \{(H_1^{(i-1)}, \mathfrak{L}), (H_0(\varpi_{i+1}), \mathfrak{R}), \cdots, (H_0(\varpi_{k-1}), \mathfrak{R})\}$
     $= \{B_i, (H_0(\varpi_{i+1}, \mathfrak{R}), \cdots, (H_0(\varpi_{k-1}), \mathfrak{R})\}$
     $i = i + 1$
     When $i = k-1$, $R_i = B_{k-1} = \{(H_1^{(k-2)}, \mathfrak{L})\}$.
     Here, ϖ_i's residue R_i includes ϖ_i's brother B_i and all of its ancestor nodes's brothers.
  5. For $\varpi_i, i = 0, \cdots, k-1$, its signature is $(\mathbf{e}, R_i)$.

- **Verify**$(vk, \varpi_j, (\mathbf{e}, R_j), j = 0, \cdots, k-1)$: Given message ϖ_j and its signature $(\mathbf{e}, R_j)$ associated with verification key $vk = \mathbf{A}$,

  1. $H_1^{(k-1)}$ should be recovered firstly.
     (1) If $j = 0$,
     $R_0 = \{(H_0(\varpi_1), \mathfrak{R}), (H_0(\varpi_2), \mathfrak{R}), \cdots, (H_0(\varpi_{k-1}), \mathfrak{R})\}$
     $= \{B_0, (H_0(\varpi_2), \mathfrak{R}), \cdots, (H_0(\varpi_{k-1}), \mathfrak{R})\}$,
     for $i = 1$ to $k-1$:
     $H_1^{(i)} = H_1(H_1^{(i-1)} \| H_0(\varpi_i))$
     $i = i + 1$
     When for-loop terminates, $H_1^{(k-1)}$ is obtained.
     (2) If $j \neq 0$,
     $R_j = \{(H_1^{(j-1)}, \mathfrak{L}), (H_0(\varpi_{j+1}), \mathfrak{R}), \cdots, (H_0(\varpi_{k-1}), \mathfrak{R})\}$
     $= \{B_j, (H_0(\varpi_{j+1}), \mathfrak{R}), \cdots, (H_0(\varpi_{k-1}), \mathfrak{R})\}$,
     for $i = j$ to $k-1$:
     $H_1^{(i)} = H_1(H_1^{(i-1)} \| H_0(\varpi_i))$
     $i = i + 1$
     When for-loop terminates, $H_1^{(k-1)}$ is obtained.
  2. Check whether $\|\mathbf{e}\| \leq s\sqrt{m}$ and $\mathbf{Ae} = H_1^{(k-1)} \bmod q$. If both relations are true, return 1 and accept the message-signature pair $(\varpi_j, (\mathbf{e}, R_j))$; otherwise, return 0 and reject the message-signature pair.

4.2. Security Analysis

Correctness of the scheme comes from the preimage sampleable algorithm. According to Definition 3, for messages set $\{\omega_1, \omega_2, \cdots, \omega_k\}$, assume the root of the binary tree is $H_1^{(k-1)}$, due to $\mathbf{e} = \text{SamplePre}(\mathbf{A}, \mathbf{T}, H_1^{(k-1)}, s)$, $\|\mathbf{e}\| \leq s\sqrt{m}$ and $\mathbf{Ae} = H_1^{(k-1)} \bmod q$ hold. Moreover, without secret signing key $\mathbf{T}$, no one can call preimage sampleable algorithm to get a vector $\mathbf{e}$ that meets the verification criteria. Therefore, there is no problem with the correctness of the scheme.

Security of the scheme comes from the following Theorem 2.

Theorem 2. *If SIS problem is hard to solve, the lattice-based batch signature scheme based on binary tree has existential unforgeability against adaptive chosen message attacks (EUF-CMA).*

Proof. We assume that adversary $\mathcal{A}$ has breached the signature scheme, taking advantage of this attack power, challenger $\mathcal{C}$ can solve SIS problem for matrix $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$. Because SIS problem is a hard problem, we ca not find the answer to SIS instance $\mathbf{A}$, which conflicts with our result. In this way, we get that no such adversary exists, and our scheme is secure.

- **Initialization**: In this period, challenger $\mathcal{C}$ executes setup algorithm to set system parameters, he sets public verification key $vk = \mathbf{A}$, sends all of them to adversary $\mathcal{A}$.
- **Hash queries**: Challenger $\mathcal{C}$ creates a list to save the binary tree for k messages, and sets $\mathcal{H} = \{((\omega_0, \cdots, \omega_{k-1}), (H_0(\omega_0), \cdots, H_0(\omega_{k-1})), (H_1^{(1)}, \cdots, H_1^{(k-1)}), \mathbf{e})\}$.

  When adversary $\mathcal{A}$ sends a set of messages $(\omega_0, \cdots, \omega_{k-1})$ to challenger $\mathcal{C}$ for hash values. $\mathcal{C}$ searches list $\mathcal{H}$,

  If $(\omega_0, \cdots, \omega_{k-1})$ do not exist in list $\mathcal{H}$, $\mathcal{C}$ chooses $\mathbf{e} \longleftarrow \mathcal{D}_{\mathbb{Z}^m, s}$, sets $H_1^{(k-1)} = \mathbf{Ae} \bmod q$. Then $\mathcal{C}$ randomly picks $H_0(\omega_i), i = 0, \cdots, k-1$, sets $H_1^{(i)} = H_1(H_1^{(i-1)} \| H_0(\omega_i))$ for $i = 1$ to $k-1$, here $H_1^{(0)} = H_0(\omega_0)$. $\mathcal{C}$ saves $((\omega_0, \cdots, \omega_{k-1}), (H_0(\omega_0), \cdots, H_0(\omega_{k-1})), (H_1^{(1)}, \cdots, H_1^{(k-1)}), \mathbf{e})$ in the list $\mathcal{H}$.

  If $(\omega_0, \cdots, \omega_{k-1})$ exist in list $\mathcal{H}$, $\mathcal{C}$ does nothing.

  At last, $\mathcal{C}$ returns $((H_0(\omega_0), \cdots, H_0(\omega_{k-1})), (H_1^{(1)}, \cdots, H_1^{(k-1)}))$ to adversary $\mathcal{A}$.
- **Signature queries**: In this stage, adversary $\mathcal{A}$ selects a set of messages $(\omega_0, \cdots, \omega_{k-1})$, sends the messages' set to challenger $\mathcal{C}$ for the associated signature. Challenger $\mathcal{C}$ firstly searches list $\mathcal{H}$ for the messages' set. If it exists, $\mathcal{C}$ returns $((H_0(\omega_0), \cdots, H_0(\omega_{k-1})), (H_1^{(1)}, \cdots, H_1^{(k-1)}), \mathbf{e})$ to adversary $\mathcal{A}$. If the messages' set does not exist, $\mathcal{C}$ executes hash query firstly. Adversary $\mathcal{A}$ may repeat the query polynomial times in his favorite manner.
- **Forgery**: Once adversary $\mathcal{A}$ terminates signing queries, he forges a valid message-signature pair $((\omega_0^*, \cdots, \omega_{k-1}^*), (H_0(\omega_0^*), \cdots, H_0(\omega_{k-1}^*)), (H_{1*}^{(1)}, \cdots, H_{1*}^{(k-1)}), \mathbf{e}^*)$.

$\mathcal{C}$ searches $((\omega_0^*, \cdots, \omega_{k-1}^*), (H_0(\omega_0^*), \cdots, H_0(\omega_{k-1}^*)), (H_{1*}^{(1)}, \cdots, H_{1*}^{(k-1)}), \mathbf{e}')$ in list $\mathcal{H}$, then computes $\mathbf{e}' - \mathbf{e}^*$ as the solution to the SIS instance $\mathbf{A}$, and the analysis is as following.

Due to the validity of the message-signature pair, adversary $\mathcal{A}$ has not made signing query on $(\omega_0^*, \cdots, \omega_{k-1}^*)$, and hash query on $(\omega_0^*, \cdots, \omega_{k-1}^*)$ has been done. Given $H_{1*}^{(k-1)}$, according to preimage min-entropy property of hash function [23], the min-entropy of $\mathbf{e}^*$ is $\omega(\log n)$, so that $\mathbf{e}' - \mathbf{e}^* \neq \mathbf{0}$ with overwhelming probability. Because $\mathbf{e}' \longleftarrow \mathcal{D}_{\mathbb{Z}^m, s}$, $\|\mathbf{e}'\| \leq s\sqrt{m}$. $\|\mathbf{e}^*\| \leq s\sqrt{m}$ depends on the validity of forged signature. Therefore, $\|\mathbf{e}' - \mathbf{e}^*\| \leq 2s\sqrt{m}$. □

5. Lattice-Based Batch Signature Based on Hash-and-Sign Paradigm

Lattice-based batch signature scheme based on binary tree is successfully constructed and proved, but the signature should associate with all other messages in the batch to complete signature verification, and batch signature length is, thus, longer. Inspired by [27,29], we make use of an intersection method to accomplish the second and third lattice-based batch signature schemes.

These two schemes' signature verification does not require involvement of other messages, so that the length of the signature is shorter, and their schematic of algorithms is shown in the Figure 2.

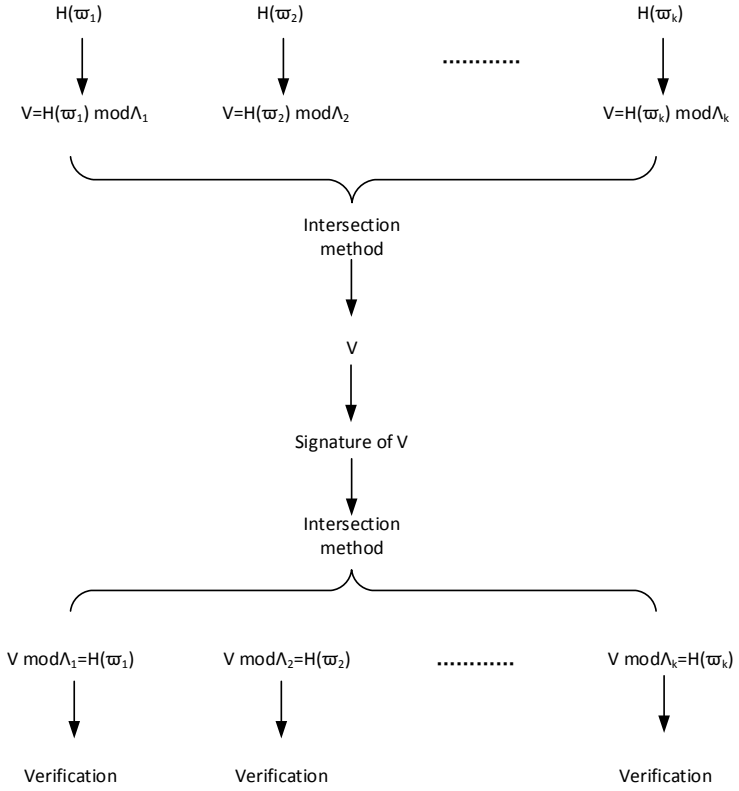

Figure 2. The Schematic of the Intersection Method.

5.1. Design

Here is our second lattice-based batch signature scheme, which follows the hash-and-sign paradigm and the core technique is the intersection method.

- **Setup**(λ): In this stage, system parameters are provided with knowledge of security parameter λ.
  1. n is a polynomial of λ, $q \geq 3$ is a polynomial of n, $m = \lceil 6n \log q \rceil$, $l = O(\sqrt{n \log q})$.
  2. k is the number of messages to batch sign, and $s \geq l \cdot \omega(\sqrt{\log m})$ is the Gaussian parameter.
  3. $H : \{0,1\}^* \longrightarrow \mathbb{Z}^n$ is a collision resistant hash function.
- **KeyGen**(λ): With system parameters as above, public verification key vk and secret signing key sk are obtained in the following manners.
  1. Invoke trapdoor generation algorithm TrapGen(n, q, m) to get a uniform and random matrix $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$, and the short basis $\mathbf{T} \in \mathbb{Z}^{m \times m}$ for lattice $\Lambda_q^{\perp}(\mathbf{A})$ with $\|\widetilde{\mathbf{T}}\| \leq l$.
  2. Compute k different lattices $\Lambda_i, i = 1, \cdots, k$, such that $\Lambda_1 + \Lambda_2 + \cdots + \Lambda_k = \mathbb{Z}^n$ and $\Lambda_1 \cap \Lambda_2 \cap \cdots \cap \Lambda_k = \Lambda$, which takes q as modulus.

     Then $vk = (\mathbf{A}, \Lambda, \Lambda_i, i = 1, \cdots, k)$, $sk = \mathbf{T}$.
- **Sign**($sk, \{\omega_1, \cdots, \omega_k\} \in \{\{0,1\}^*\}^k$): Given $sk = \mathbf{T}$ and the set of messages $\{\omega_1, \cdots, \omega_k\} \in \{\{0,1\}^*\}^k$, the following steps lead to batch signature on such messages.
  1. Construct equations:

$$\begin{cases} \mathbf{v} = H(\omega_1) \mathrm{mod} \Lambda_1 \\ \mathbf{v} = H(\omega_2) \mathrm{mod} \Lambda_2 \\ \cdots \\ \mathbf{v} = H(\omega_k) \mathrm{mod} \Lambda_k \end{cases}$$

compute its solution $\mathbf{v}$.
2. Invoke preimage sampleable algorithm SamplePre$(\mathbf{A}, \mathbf{T}, \mathbf{v}, s)$ to get the signature $\mathbf{e}$.

- **Verify**$(vk, \varpi_j, \mathbf{e})$: For the j-th message ϖ_j and the signature $\mathbf{e}$, validation involves the following two relations:

1. $\|\mathbf{e}\| \leq s\sqrt{m}$.
2. $\mathbf{Ae} = H(\varpi_j) \text{mod} \Lambda_j$.

If they are both true, accept message ϖ_j; otherwise, reject it.

5.2. Security Analysis

Correctness of the second scheme is similar to that of the first scheme. By Definition 3, $\|\mathbf{e}\| \leq s\sqrt{m}$ and $\mathbf{Ae} = \mathbf{v} \bmod q$. By Definition 2, $\mathbf{v} = H(\varpi_j) \bmod \Lambda_j$. Combining $\mathbf{Ae} = \mathbf{v} \bmod q$ and $\mathbf{v} = H(\varpi_j) \bmod \Lambda_j$, $\mathbf{Ae} = H(\varpi_j) \bmod \Lambda_j$. Moreover, without signing key $\mathbf{T}$, nobody can invoke preimage sampleable algorithm to get a vector $\mathbf{e}$ satisfying the verification relations.

Security of the scheme comes from the following Theorem 3.

Theorem 3. *If SIS problem is a hard problem, the lattice-based batch signature scheme based on hash-and-sign paradigm has existential unforgeability against adaptive chosen message attacks (EUF-CMA).*

Proof. If there exists an adversary $\mathcal{A}$ who has the ability to forge batch signature for some messages, there exists a challenger $\mathcal{C}$ has the ability to give the solution to SIS instance $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$, here the challenger $\mathcal{C}$ will seek the help of the adversary $\mathcal{A}$. Since SIS problem is a hard problem, the solution to SIS instance $\mathbf{A}$ is hard to obtained, this is in contradiction with our result. Therefore, the adversary $\mathcal{A}$ who has the ability to forge batch signature does not exist, and our lattice-based batch signature scheme based on hash-and-sign paradigm has EUF-CMA security.

- **Initialization**: In this period, challenger $\mathcal{C}$ sets appropriate system parameters, lets public verification key $vk = \mathbf{A}$, sends all of them to adversary $\mathbf{A}$.
- **Hash queries**: Challenger $\mathcal{C}$ creates a list to save the hash values for k messages, and sets $\mathcal{H} = \{((\varpi_1, \cdots, \varpi_k), (H(\varpi_1), \cdots, H(\varpi_k)), \mathbf{e})\}$.

  When adversary $\mathcal{A}$ sends a set of messages $(\varpi_1, \cdots, \varpi_k)$ to challenger $\mathcal{C}$ for hash values. $\mathcal{C}$ searches list $\mathcal{H}$.

  If $(\varpi_1, \cdots, \varpi_k)$ exist in list $\mathcal{H}$, $\mathcal{C}$ returns $(H(\varpi_1), \cdots, H(\varpi_k))$ directly.

  If $(\varpi_1, \cdots, \varpi_k)$ do not exist in list $\mathcal{H}$, $\mathcal{C}$ samples $\mathbf{e} \longleftarrow \mathcal{D}_{\mathbb{Z}^m, s}$, sets $\mathbf{v} = \mathbf{Ae} \bmod q$, and lets $\mathbf{v} \bmod \Lambda_i = H(\omega_i)$, $i = 1, \cdots, k$. Then $\mathcal{C}$ saves $((\varpi_1, \cdots, \varpi_k), (H(\varpi_1), \cdots, H(\varpi_k)), \mathbf{e})$ in list $\mathcal{H}$, and returns $(H(\varpi_1), \cdots, H(\varpi_k))$ to adversary $\mathcal{A}$.
- **Signing queries**: In this stage, adversary $\mathcal{A}$ selects a set of messages $(\varpi_1, \varpi_2, \cdots, \varpi_k)$, sends the messages' set to challenger $\mathcal{C}$ for the associated signature. Challenger $\mathcal{C}$ searches list $\mathcal{H}$ for messages $(\varpi_1, \varpi_2, \cdots, \varpi_k)$.

  If the messages exist in list $\mathcal{H}$, challenger $\mathcal{C}$ returns $\mathbf{e}$ directly.

  If the messages do not exist in list $\mathcal{H}$, Challenger $\mathcal{C}$ firstly executes Hash query, then returns $\mathbf{e}$ to adversary $\mathcal{A}$.
- **Forgery**: Once adversary $\mathcal{A}$ terminates signing queries, he offers a new message-signature pair $(\varpi_1^*, \varpi_2^*, \cdots, \varpi_k^*, \mathbf{e}^*)$.

  Challenger $\mathcal{C}$ searches $((\varpi_1^*, \varpi_2^*, \cdots, \varpi_k^*), (H(\varpi_1^*), \cdots, H(\varpi_k^*)), \mathbf{e}')$ in list $\mathcal{H}$, then computes $\mathbf{e}^* - \mathbf{e}'$ as the solution to the SIS instance $\mathbf{A}$.

  Because message-signature pair $(\varpi_1^*, \varpi_2^*, \cdots, \varpi_k^*, \mathbf{e}^*)$ is valid, adversary $\mathcal{A}$ has not made signing query on $(\varpi_1^*, \cdots, \varpi_k^*)$, and hash query on $(\varpi_1^*, \cdots, \varpi_k^*)$ has been done. Given $\mathbf{Ae}' = \mathbf{v} \bmod q$, according to preimage min-entropy property of hash function [23], the min-entropy of $\mathbf{e}^*$ is

$\omega(\log n)$, so that $\mathbf{e}' - \mathbf{e}^* \neq \mathbf{0}$ with overwhelming probability. Because $\mathbf{e}' \longleftarrow \mathcal{D}_{\mathbb{Z}^m,s}$, $\|\mathbf{e}'\| \leq s\sqrt{m}$. $\|\mathbf{e}^*\| \leq s\sqrt{m}$ depends on the validity of forged signature. Therefore, $\|\mathbf{e}' - \mathbf{e}^*\| \leq 2s\sqrt{m}$.

□

6. Lattice-Based Batch Signature Based on FS Transformation

In lattice-based cryptography, trapdoor generation algorithm (Definition 2) and preimage sampleable algorithm (Definition 3) are fundamental algorithms of signature scheme, but both algorithms have high computational complexity. To improve signature scheme's efficiency, we take lattice signature based on Fiat–Shamir transformation in [26], apply to lattice-based batch signature with intersection method, obtain a new and more efficient batch signature scheme.

6.1. Design

- **Setup**(n): In this stage, system parameters are provided with knowledge of security parameter n.

  1. q may be 2^{25}, d may be 1, r may be 512.
  2. $m > 64 + n \cdot \log q / \log(2d+1)$, κ satisfies $2^\kappa \cdot \begin{pmatrix} n \\ \kappa \end{pmatrix} \geq 2^{100}$.
  3. s may be $12d\kappa\sqrt{m}$, and M may be $e^{(12d\kappa\sqrt{m}/s + (d\kappa\sqrt{m}/(2s))^2)}$.
  4. $H_1 : \{0,1\}^* \longrightarrow \mathbb{Z}^n$ and $H_2 : \{0,1\}^* \longrightarrow \{\mathbf{g} : \mathbf{g} \in \{-1,0,1\}^r, \|\mathbf{g}\|_1 \leq \kappa\}$ are collision resistant hash functions, where $\|\mathbf{g}\|_1$ is the 1-norm of vector $\mathbf{g}$, namely, it is the sum of the absolute values of each element of vector $\mathbf{g}$.

- **KeyGen**(n): With system parameters as above, public verification key vk and secret signing key sk are obtained in the following manners.

  1. Select $\mathbf{S} \leftarrow \{-d, \cdots, 0, \cdots, d\}^{m \times r}$ randomly as secret signing key.
  2. Select $\mathbf{A} \leftarrow \mathbb{Z}_q^{n \times m}$, compute $\mathbf{T} = \mathbf{AS}$ as public verification key.
  3. Compute k different lattices $\Lambda_i, i = 1, \cdots, k$, such that $\Lambda_1 + \Lambda_2 + \cdots + \Lambda_k = \mathbb{Z}^n$ and $\Lambda_1 \cap \Lambda_2 \cap \cdots \cap \Lambda_k = \Lambda$, which takes q as modulus.

     Then $vk = (\mathbf{A}, \mathbf{T}, \Lambda, \Lambda_i, i = 1, \cdots, k)$, $sk = \mathbf{S}$.

- **Sign**($sk, \{\varpi_1, \cdots, \varpi_k\} \in \{\{0,1\}^*\}^k$): Given $sk = \mathbf{S}$ and the set of messages $\{\varpi_1, \cdots, \varpi_k\} \in \{\{0,1\}^*\}^k$, the following steps lead to batch signature on such messages.

  1. Construct equations:
  $$\begin{cases} \mathbf{v} = H_1(\varpi_1) \bmod \Lambda_1 \\ \mathbf{v} = H_1(\varpi_2) \bmod \Lambda_2 \\ \cdots \\ \mathbf{v} = H_1(\varpi_k) \bmod \Lambda_k \end{cases}$$
  compute its solution $\mathbf{v}$.
  2. Sample $\mathbf{y} \leftarrow \mathcal{D}_{\mathbb{Z}^m,s}$ randomly, compute $\mathbf{c} = H_2(\mathbf{Ay}, \mathbf{v})$.
  3. Let $\mathbf{z} = \mathbf{Sc} + \mathbf{y}$, output $(\mathbf{v}, \mathbf{z}, \mathbf{c})$ as signature with probability $\min(\frac{\mathcal{D}_{\mathbb{Z}^m,s}(\mathbf{z})}{M\mathcal{D}_{\mathbb{Z}^m,s,\mathbf{Sc}}(\mathbf{z})}, 1)$.

- **Verify**($vk, \varpi_j, (\mathbf{v}, \mathbf{z}, \mathbf{c})$): For the j-th message ϖ_j and the signature $(\mathbf{v}, \mathbf{z}, \mathbf{c})$, validation involves the following three relations:

  1. $\|\mathbf{z}\| \leq 2s\sqrt{m}$.
  2. $\mathbf{c} = H_2(\mathbf{Az} - \mathbf{Tc}, \mathbf{v})$.
  3. $\mathbf{v} \bmod \Lambda_j = H_1(\varpi_j)$.

     If they are true, accept message ϖ_j; otherwise, reject it.

6.2. Security Analysis

The batch signature scheme in Section 5 and the batch signature scheme in Section 6 are different in terms of basic signature schemes: the first basic signature scheme comes from the literature [23], the second basic signature scheme comes from the literature [26]. According to literature [30], for the same security, the second bath signature scheme has better efficiency.

The same as batch signature scheme in Section 5, the batch signature scheme in Section 6's correctness and security can be reduced to its basic signature scheme, and the details are omitted here.

7. Efficiency Comparison and the Application to IoT

Refs. [14–16] are among the several most important batch signature schemes and improved techniques so far. Ref. [14] first introduced the idea and provide the construction based on RSA scheme. Later in [15], the authors focused on speeding up the modular exponentiation operation which is used in many DLP based signatures. In other words, the work [15] focuses on signature schemes which were built in the traditional multiplicative group, and then the authors in [16] managed to achieve the constant complexity for signature generation and verification, which does not rely on the number of messages. Our second and third schemes enjoy the same advantage as [16] regarding constant generation and verification complexity. Comparing the concrete efficiency for those schemes boils down to the issue of comparing the underlined algebraic primitives. According to [31], the key parameters, namely n, q in our lattice based scheme should be chosen at around 500 and 200,000 to achieve approximately the 128-bit level security. Generally speaking, our signature size will be larger than the ones constructed in the group where DLP or ECDLP problem is hard. According to [32], the computation of the lattice is very fast, and at the same security level the current lattice scheme will outperform the RSA and DLP or ECDLP based schemes. Most importantly, up to now, none of the previous listed batch signature schemes are able to resist against quantum attacks, which makes our scheme a very attractive choice in a long run. Lattice primitives are also being optimized by taking advantage of the modern CPU instruction such as AVX, AVX2 and so on [33], thus, the computation speed can be expected to be further improved.

In our constructions, we make use of two different approaches to integrate a group of messages to fulfill batch signatures, namely, binary tree and intersection methods. The schemes are existentially unforgeable against adaptive chosen message attacks. Table 1 shows the efficiency of our scheme regarding the different signature stages and parameter sizes in the asymptotic manner. From Table 1, it can be seen that all of the schemes require $O(n^2 log^3 n)$ complexity on the secret key generation, while the 2nd and 3rd constructions only take $O(n)$ for the size of signature, which means that a batch signature is independent of the parameter k. For computational comparison, it can be seen that the 3rd construction requires least computational complexity, $O(n^3)$, while others have to take S and other operations. In the verification stage, the 2nd and 3rd scheme are constant cost instead of being linear with k, unlike the 1st construction. In all aspects, 3rd construction is the most efficient, and we describe its application in wireless body sensor network, which is a typical application of IoT.

Table 1. Efficiency comparison of our schemes.

Scheme (Size/Computation)	Our 1st Scheme (with Binary Tree)	Our 2nd Scheme (With Intersection Method)	Our 3rd Scheme (With FS Transformation)
sk	$\mathcal{O}(n^2 log^3 n)$	$\mathcal{O}(n^2 log^3 n)$	$\mathcal{O}(n^2 log^3 n)$
vk	$\mathcal{O}(n^2 log^2 n)$	$\mathcal{O}(kn^2)$	$\mathcal{O}(kn^2)$
signature	$\mathcal{O}(kn)$	$\mathcal{O}(n)$	$\mathcal{O}(n)$
$Sign$	$(2k-1)\mathcal{H} + \mathcal{S} + \mathcal{O}(n^2)$	$\mathcal{O}(n^3) + \mathcal{S}$	$\mathcal{O}(n^3)$
$Verify$	$2\mathcal{V} + k\mathcal{H}$	$2\mathcal{V} + \mathcal{H}$	$3\mathcal{V}$+$\mathcal{H}$
Anti-quantum	√	√	√

Notation: $\mathcal{H}$ denotes Hash function, $\mathcal{S}$ denotes SamplePre, $\mathcal{V}$ denotes verification.

A wireless body sensor network [34] is composed of three sides: a receiver station(RS), many central control units (CCU), and many sensors (SS). A receiver station manages multiple central control units and a central control unit manages many sensors. Specifically, A patient's body is implanted with many sensors and a central control unit, which collects human medical data and sends it to the receiver station. The receiver station is responsible for verifying data and warehousing for all central control units, that is, the receiver station checks medical data and signs all of them. When a large number of medical data, which is from different patients, come in at the same time, the receiver station will become the bottleneck of data processing. Batch signature can solve the requirement of batch signing and individual verification for patients' medical data; the process is as follows.

Firstly, according to the 3rd batch signature scheme, wireless body sensor network sets up system parameters and public/private keys for the receiver station. A central control unit collects medical data in real time and sends it to the receiver station. The receiver station divides k central control unit data into one group, such as $\{\varpi_1, \cdots, \varpi_k\}$, executes algorithm **Sign**$(sk, \{\varpi_1, \cdots, \varpi_k\})$, obtain $(\mathbf{v}, \mathbf{z}, \mathbf{c})$, store it with the message ϖ_i, $i = 1, \cdots, k$. When the j-th central control unit's data ϖ_j is called, $(\varpi_j, (\mathbf{v}, \mathbf{z}, \mathbf{c}))$ is provided, and the algorithm **Verify**$(\varpi_j, (\mathbf{v}, \mathbf{z}, \mathbf{c}))$ can be invoked to verify the validity of data ϖ_j. If the answer is yes, the data is authoritative and credible. Otherwise, the data is unusable. The data flow diagram is shown in the Figure 3.

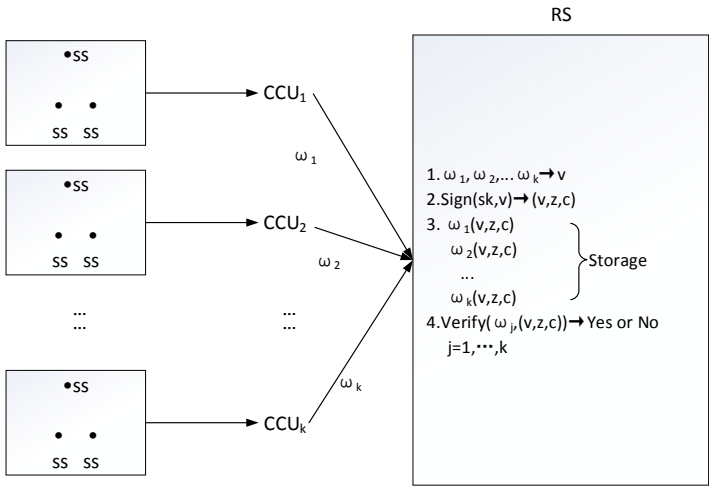

Figure 3. Data Flow Diagram in Wireless Body Sensor Network.

8. Conclusions

In this paper, we presented three new lattice-based batch signature schemes by using binary tree and intersection methods, with hash-and-sign paradigm and Fiat–Shamir transformation, respectively. Our schemes were existential unforgeability against adaptive chosen message attacks based on the difficulty of a small integer solution problem, which provided quantum security. A detailed efficiency analysis was also given, which showed that our schemes optimized the size of the public key, private key and signature. Moreover, we applied our batch signature schemes to a wireless body sensor network, which was a typical application of IoT, improved signature efficiency and security. In addition, batch signatures can also be applied to blockchain systems for speeding up the process of block signing operations, which we aim to be our next work.

Author Contributions: The first author X.L., proposed the main idea "intersection method" for batch signature and gave three schemes. The second author W.Y., gave all the figures and tables, as well as Sections 1 and 2. The third author Q.W., is the doctoral supervisor of the first two authors, guided the whole writing. The fourth author K.L., proposed the idea to describe signature from two frames: hash-and-sign paradigm and Fiat–Shamir transformation, which make the paper more comprehensive and systematic. The fifth author L.C., was responsible for the English writing of the whole paper. The sixth author J.C., the corresponding author, was responsible for efficiency analysis and application scenario description as well as the management of the research project.

Funding: This work was funded by the National Natural Science Foundation of China [no. 61502044, 61402015, 61702212]; the Fundamental Research Funds for the Central Universities [no. 2015RC23]; the Natural Science Foundation of Hebei Province [no. F2018408040]; the Natural Science Foundation of Shandong Province [no. ZR201702180067]; and the Hebei Education Funds for Youth Project [no. QN2018047].

Conflicts of Interest: We declare that no conflict of interest among six authors.

References

1. Gubbi, J.; Buyya, R.; Marusic, S.; Palaniswami, M. Internet of Things (IoT): A vision, architectural elements, and future directions. *Future Gener. Comput. Syst.* **2013**, *29*, 1645–1660. [CrossRef]
2. Bennett, T.R.; Savaglio, C.; Lu, D.; Massey, H.; Wang, X.; Wu, J.; Jafari, R. MotionSynthesis Toolset (MoST): A Toolset for Human Motion Data Synthesis and Validation. In Proceedings of the 4th ACM MobiHoc Workshop on Pervasive Wireless Healthcare (MobileHealth '14), Philadelphia, PA, USA, 11 August 2014; pp. 25–30. [CrossRef]
3. Li, S.; Xu, L.D.; Zhao, S. The internet of things: A survey. *Inf. Syst. Front.* **2015**, *17*, 243–259. [CrossRef]
4. Fortino, G.; Russo, W.; Savaglio, C.; Viroli, M.; Zhou, M. Modeling Opportunistic IoT Services in Open IoT Ecosystems. In Proceedings of the 18th Workshop "From Objects to Agents" (WOA 2017), Scilla (RC), Italy, 15–16 June 2017; pp. 90–95.
5. Diffie, W.; Hellman, M.E. New directions in cryptography. *IEEE Trans. Inf. Theory* **1976**, *22*, 644–654. [CrossRef]
6. Goldwasser, S.; Micali, S.; Yao, A.C. On Signatures and Authentication. In *Advances in Cryptology: Proceedings of CRYPTO '82, Santa Barbara, CA, USA, 23–25 August 1982*; Chaum, D., Rivest, R.L., Sherman, A.T., Eds.; Plenum Press: New York, NY, USA, 1982; pp. 211–215.
7. Rivest, R.L.; Shamir, A.; Adleman, L.M. A Method for Obtaining Digital Signatures and Public-Key Cryptosystems (Reprint). *Commun. ACM* **1983**, *26*, 96–99. [CrossRef]
8. Huang, X.; Liu, J.K.; Tang, S.; Xiang, Y.; Liang, K.; Xu, L.; Zhou, J. Cost-Effective Authentic and Anonymous Data Sharing with Forward Security. *IEEE Trans. Comput.* **2015**, *64*, 971–983. [CrossRef]
9. Liang, K.; Chu, C.; Tan, X.; Wong, D.S.; Tang, C.; Zhou, J. Chosen-ciphertext secure multi-hop identity-based conditional proxy re-encryption with constant-size ciphertexts. *Theor. Comput. Sci.* **2014**, *539*, 87–105. [CrossRef]
10. Ning, J.; Cao, Z.; Dong, X.; Liang, K.; Ma, H.; Wei, L. Auditable σ-Time Outsourced Attribute-Based Encryption for Access Control in Cloud Computing. *IEEE Trans. Inf. Forensics Secur.* **2018**, *13*, 94–105. [CrossRef]
11. Zhou, J.; Duan, H.; Liang, K.; Yan, Q.; Chen, F.; Yu, F.R.; Wu, J.; Chen, J. Securing Outsourced Data in the Multi-Authority Cloud with Fine-Grained Access Control and Efficient Attribute Revocation. *Comput. J.* **2017**, *60*, 1210–1222. [CrossRef]
12. Shao, J.; Lu, R.; Lin, X.; Liang, K. Secure bidirectional proxy re-encryption for cryptographic cloud storage. *Pervasive Mob. Comput.* **2016**, *28*, 113–121. [CrossRef]
13. Gamal, T.E. A public key cryptosystem and a signature scheme based on discrete logarithms. *IEEE Trans. Inf. Theory* **1985**, *31*, 469–472. [CrossRef]
14. Fiat, A. Batch RSA. In Proceedings of the Advances in Cryptology—CRYPTO '89, 9th Annual International Cryptology Conference, Santa Barbara, CA, USA, 20–24 August 1989; pp. 175–185. [CrossRef]
15. M'Raïhi, D.; Naccache, D. Batch Exponentiation: A Fast DLP-Based Signature Generation Strategy. In Proceedings of the 3rd ACM Conference on Computer and Communications Security (CCS '96), New Delhi, India, 14–16 March 1996; pp. 58–61. [CrossRef]
16. Pavlovski, C.; Boyd, C. Efficient batch signature generation using tree structures. In Proceedings of the International Workshop on Cryptographic Techniques and E-Commerce (CrypTEC), Hong Kong, China, 5–8 July 1999; Volume 99, pp. 70–77.
17. Cheng, W.C.; Chou, C.; Golubchik, L. Performance of Batch-Based Digital Signatures. In Proceedings of the 10th International Workshop on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems (MASCOTS 2002), Fort Worth, TX, USA, 11–16 October 2002; p. 291. [CrossRef]
18. Korkmaz, T.; Tek, S. Analyzing Response Time of Batch Signing. *J. Internet Serv. Inf. Secur.* **2011**, *1*, 70–85.

19. Boyd, C.; Foo, E.; Pavlovski, C. Efficient Electronic Cash Using Batch Signatures. In Proceedings of the Information Security and Privacy, 4th Australasian Conference, ACISP'99, Wollongong, NSW, Australia, 7–9 April 1999; pp. 244–257. [CrossRef]
20. Youn, T.; Park, Y.; Kwon, T.; Kwon, S.; Lim, J. Efficient Flexible Batch Signing Techniques for Imbalanced Communication Applications. *IEICE Trans.* **2008**, *91-D*, 1481–1484. [CrossRef]
21. Shor, P.W. Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer. *SIAM J. Comput.* **1997**, *26*, 1484–1509. [CrossRef]
22. Ajtai, M. Generating Hard Instances of Lattice Problems (Extended Abstract). In Proceedings of the Twenty-Eighth Annual ACM Symposium on the Theory of Computing, Philadelphia, PA, USA, 22–24 May 1996; pp. 99–108. [CrossRef]
23. Gentry, C.; Peikert, C.; Vaikuntanathan, V. Trapdoors for hard lattices and new cryptographic constructions. In Proceedings of the 40th Annual ACM Symposium on Theory of Computing, Victoria, BC, Canada, 17–20 May 2008; pp. 197–206. [CrossRef]
24. Alwen, J.; Peikert, C. Generating Shorter Bases for Hard Random Lattices. *Theory Comput. Syst.* **2011**, *48*, 535–553. [CrossRef]
25. Micciancio, D.; Peikert, C. Trapdoors for Lattices: Simpler, Tighter, Faster, Smaller. In *Advances in Cryptology—EUROCRYPT 2012—31st Annual International Conference on the Theory and Applications of Cryptographic Techniques*; Lecture Notes in Computer Science; Pointcheval, D., Johansson, T., Eds.; Springer: Cambridge, UK, 2012; Volume 7237, pp. 700–718. [CrossRef]
26. Lyubashevsky, V. Lattice Signatures without Trapdoors. In *Advances in Cryptology—EUROCRYPT 2012—31st Annual International Conference on the Theory and Applications of Cryptographic Techniques*; Lecture Notes in Computer Science; Pointcheval, D., Johansson, T., Eds.; Springer: Cambridge, UK, 2012; Volume 7237, pp. 738–755. [CrossRef]
27. Boneh, D.; Freeman, D.M. Homomorphic Signatures for Polynomial Functions. In *Advances in Cryptology—EUROCRYPT 2011—30th Annual International Conference on the Theory and Applications of Cryptographic Techniques*; Lecture Notes in Computer Science; Paterson, K.G., Ed.; Springer: Tallinn, Estonia, 2011; Volume 6632, pp. 149–168. [CrossRef]
28. Bellare, M.; Rogaway, P. Collision-Resistant Hashing: Towards Making UOWHFs Practical. In Proceedings of the Advances in Cryptology—CRYPTO '97, 17th Annual International Cryptology Conference, Santa Barbara, CA, USA, 17–21 August 1997; pp. 470–484. [CrossRef]
29. Brakerski, Z.; Gentry, C.; Vaikuntanathan, V. (Leveled) Fully Homomorphic Encryption without Bootstrapping. *TOCT* **2014**, *6*, 13. [CrossRef]
30. Howe, J.; Pöppelmann, T.; O'Neill, M.; O'Sullivan, E.; Güneysu, T. Practical Lattice-Based Digital Signature Schemes. *ACM Trans. Embed. Comput. Syst.* **2015**, *14*, 41. [CrossRef]
31. Alkadri, N.A.; Buchmann, J.; Bansarkhani, R.; Krämer, J. *A Framework to Select Parameters for Lattice-Based Cryptography*; Cryptology ePrint Archive, Report 2017/615. Available online: https://eprint.iacr.org/2017/615 (accessed on 26 June 2017).
32. Yuan, Y.; Cheng, C.M.; Kiyomoto, S.; Miyake, Y.; Takagi, T. Portable implementation of lattice-based cryptography using JavaScript. *Int. J. Netw. Comput.* **2016**, *6*, 309–327. [CrossRef]
33. Longa, P.; Naehrig, M. Speeding up the number theoretic transform for faster ideal lattice-based cryptography. In Proceedings of the International Conference on Cryptology and Network Security, Milan, Italy, 14–16 November 2016; pp. 124–139.
34. Yuce, M.R.; Ng, S.W.P.; Myo, N.L.; Khan, J.Y.; Liu, W. Wireless Body Sensor Network Using Medical Implant Band. *J. Med. Syst.* **2007**, *31*, 467–474. [CrossRef] [PubMed]