Lattice-Based Incremental Signature Scheme for the Authenticated Data Update in Fog Computing

JUNQUAN WANG¹, FENGHE WANG², SHAOQUAN SHI², AND WENFENG YANG³

¹School of Science, Shandong Jianzhu University, Jinan 250101, China
²School of Computer, Shandong Jianzhu University, Jinan 250101, China
³Tianbo Electronic Information Technology Company, Ltd., Qingdao 266003, China

Corresponding author: Fenghe Wang (fenghe2166@163.com)

This work was supported in part by the National Natural Science Foundation of China under Grant 61803228 and Grant 61303198, in part by the project of Shandong Province Higher Education Science and Technology Program under Grant J18KA361, and in part by the Doctor Foundation of Shandong Jianzhu University under Grant XNBS1624.

ABSTRACT To efficiently authenticate the updated data in fog computing, this paper proposed a lattice-based incremental digital signature scheme. The proposed scheme is provable secure in the standard model whose security is based on the hardness of the shortest integer solution (SIS) problem. There are several characters of the proposed scheme that are suitable for the application in fog computing. Such as, most of the time-consuming computing operations in the proposed scheme can be finished by the parallel computing. As a result, the computing speed of the proposed scheme can be improved efficiently. On the other hand, compared with a known lattice-based incremental digital signature, both the public key size and the signature length are shorter. Then the store resource and the bandwidth of the fog device can be saved efficiently when it is used in fog computing. Furthermore, a simulated experiment is given to check the functions of the proposed scheme by Java language on PC. The result shows that the computing cost to update a signature is much less than that to resign a message. At the same time, the parallel computing and pre-computing can really efficiently improve the computing speed of the proposed scheme. Since the SIS problem is hard even on quantum computers, the proposed scheme also gives a post-quantum secure solution to authenticate the updated data in fog computing.

INDEX TERMS Authentication, fog computing, incremental signature, lattice-based cryptography.

I. INTRODUCTION
Since the cloud servers are with super computing and professional analysis ability, the user can enjoy the professional service from the cloud, such as the service of storing, managing and processing data. More and more enterprises and personal clients store their data to cloud servers. Internet of thing (IoT) consists as millions of the physic sensors. Informally, IoT puts internet to the edge of world and connects everything with internet. And the cloud computing is a centralized architecture of the internet. Hence, we should combine the IoT with cloud computing to exploit their advantages respectively. While between IoT and cloud computing, a new mechanism is needed to collect, process and transfer data [1]. Fog computing is a such paradigm which can be seemed as a combination of IoT and cloud computing [2], [3].

Fog computing firstly proposed by CiSCO [4]. It is an extension of cloud computing paradigm from the core to edge of the networks. Figure 1 shows the work mechanism of the fog computing [5]. In fog computing, fog devices are used to collect, process and transfer data and fog nodes are used to connect these devices and cloud servers. Both devices and nodes have some storage and processing capabilities, while these capabilities are limited. As a result, we should try to save the resources when we wish to achieve some functions in fog computing [6]. On the other hand, since both the devices and the fog nodes are distributed and with complex working environment, there are vulnerable to be attacked by various types adversaries. Hence, the potential security risk of the fog computing should be resolved [7].

When the devices transfer data to fog node, or the fog nodes transfer document to severs, the digital signature should also be sent to against forgery attack. While the data collected by the devices or fog nodes possible is incomplete, the additional
supplement information sometimes need to update the original message. For example, in a weather forecasting system, the detection device has some new data \( u_{\text{add}} \) that need to modify some known data \( u \) which has been corrected to the fog node. We should note that the modification is partly, which means the most parts remain unchange between the messages \( u + u_{\text{add}} \) and \( u \). As a result, if we ask the device to sign the whole message \( u + u_{\text{add}} \), it is not efficient. Then, how to efficiently authenticate the updated data is an interesting topic for fog computing.

Incremental signature can be used to give an efficient update of the signature in fog computing. More precisely, if the document \( u \) needs to be modified by \( u_{\text{add}} \), then instead of re-signing the whole message \( u + u_{\text{add}} \), we just update the original signature of document \( u \) by signing the additional message \( u_{\text{add}} \) [8]. It is clearly that the incremental signature provides an efficient authentication method to save the bandwidth and the computing cost of the fog computing.

Incremental cryptography was introduced by Bellare, Goldreich and Goldwasse which gives an efficient solution to the secure data update [9]. Since then many incremental cryptographic schemes have been proposed [8], [10]–[14]. These schemes give very elegant constructions, while are not quantum security. [10] firstly considered the quantum security and proposed a lattice-based incremental signature scheme in the standard model. While the space size of the scheme in [10] is too large to be used in fog computing. In fact, both the public key size and the signature length of scheme in [10]should be improved to save the store resource and bandwidth, especially considering its application in fog computing.

A. OUR WORK

This paper proposed an efficient lattice-based incremental signature scheme in the standard model. We prove that the proposed scheme is unforgeable whose security is based on the shortest integer solution problem. Compared with the known the lattice-based incremental signature scheme in [10], we add a ring operation into the message coding to conceal the coding structure of message. This idea helps us to delete a matrix from public key and also avoids to run the random sample operations in both Sign and InSig algorithms. As a result, the proposed scheme shares some advantages about the signature length, the public key size and the running speed. Moreover, parallel computing and pre-computing can be widely used in the proposed scheme to improve the running speed of the proposed scheme. To check the computing speed of the proposed scheme, a simulated experiment by JAVA language on PC is given in this paper. The result shows that a updated data can be authenticated with more less cost than resign the whole data. Moreover, the run speeds of all algorithms in proposed scheme are higher than those of the scheme in [10], such as to generate a signature or update a signature.

B. RELATED WORK

As known as the post-quantum cryptography, lattice-based cryptography has gained more and more attentions in recently years. Considering the lattice-based signature scheme, there are two approaches for designing lattice-based digital signature scheme. One is proposed by Lyubashevsky which used the Fiat-Shamir technique [15]. The other is to use the hash-and-sign method in [16]. In [16], the trapdoor basis [17] of a lattice is used as a private key to sign a message. Since then trapdoor basis has been widely used in the design of lattice-based signature schemes [18], [19]. The other important and efficient lattice-based trapdoor is proposed by D. Micciancio and C. Peikert [20]. By the novel trapdoor of [20], the efficiency of the lattice-based digital signature has been improved. In fact, the lattice-based incremental signature scheme in [10] is designed by using the trapdoor in [20]. To conceal the linear structure of the message coding, an additional random matrix must be brought into the public keys of in [10]. For the same reason, the signer must choose a Gaussian vector in both Sign and InSig algorithms which occupies the computing resources of the signer in [10]. Just as we described, to improve the efficiency of the lattice-based incremental signature scheme is an important requirement to achieve the authentication of updated data in fog computing. This paper would use the trapdoor in [20] to design the incremental signature scheme.

II. PRIMITIVES

A. NOTATIONS

Matrices and vectors in column form are denoted by bold upper-case and the bold lower-case letters respectively. Let \( \| \cdot \| \) be the Euclidean norm. The matrix norm is defined to
be the norm of the longest column. An unspecified function \( f(n) = O(n^c) \) for a constant \( c \) is written as \( \text{poly}(n) \). Function \( g(n) \) is negligible if \( g(n) = 1/\text{poly}(n) \). We see a function \( g(n) = o(f(n)) \) if it grows faster than \( cf(n) \) for any constant \( c \). \( D_\alpha \) denotes the Gaussian distribution over \( \mathbb{R} \) with parameter \( \alpha \).

**B. LATTICE**

Given \( n \) linearly independent vectors \( B = \{b_1, b_2, \cdots, b_n\} \), a lattice is defined as

\[
\Lambda = \{Bc = \sum_{i \in [n]} c_ib_i, |c_i| \in \mathbb{Z}\}.
\]

We call \( B \) is the basis of the lattice. \( B \) is a trapdoor basis when all the basis vectors are with small Euclidean norms. Note that the trapdoor basis of the lattice can act as a trapdoor of the lattice-based cryptographic scheme [17].

Compared with the traditional trapdoor basis of lattice, D. Micciancio and C. Peikert proposed a more efficient trapdoor called G-trapdoor of cryptographic lattice [20]. Informally, it shares some advantages over the trapdoor basis, such as the smaller lattice dimension and the shorter trapdoor size. Moreover, G-trapdoor also can achieve the Gaussian sampler and the delegation functions which are important for the design of the lattice-based schemes.

Given parameters \( (n, m, q, k) \) and a primitive matrix \( G \in \mathbb{Z}_q^{n \times nk} \) [20], the G-trapdoor is defined as follows.

**Definition 1:** Given matrices \( A \in \mathbb{Z}_q^{n \times m}, G \in \mathbb{Z}_q^{n \times nk} \) and a tag \( H \in \mathbb{Z}_q^{n \times r} \), then a G-trapdoor of matrix \( A \) is defined as a matrix \( R \in \mathbb{Z}_q^{(m-nk) \times nk} \) satisfying

\[
A \begin{bmatrix} R \\ I \end{bmatrix} = HG.
\]

Given parameters \( (n, m, q) \) and a random vector \( y \in \mathbb{Z}_q^n \), the following special integer lattices are widely used in the design of the lattice-based cryptographic scheme,

\[
\Lambda^+_{q}(A) = \{x \in \mathbb{Z}_q^m, Ax = 0(\text{mod}q)\},
\]

\[
\Lambda^\perp_q(A) = \{x \in \mathbb{Z}_q^m, Ax = y(\text{mod}q)\}.
\]

Just like the known lattice-based signature schemes, we will also design the incremental signature scheme based on the SIS problem in this paper. More precisely, the SIS problem is described as follows.

**Definition 2:** Given a real number \( \beta \) and \( A \in \mathbb{Z}_q^{n \times m} \), the SIS problem is defined to find a nonzero vector \( e \in \mathbb{Z}_q^m \) which satisfies \( Ae = 0(\text{mod}q) \) and \( ||e||_2 \leq \beta \).

**C. DISCRETE GAUSSIAN DISTRIBUTION**

The discrete Gaussian distribution on lattice \( \Lambda^+_{q}(A) \) (\( A \in \mathbb{Z}_q^{n \times m} \)) pays an important role for the design of the lattice-based cryptography. More precisely, the discrete Gaussian on lattice is defined as a “conditional” distribution

\[
D_{\Lambda^+_{q}(A), \sigma, c}(x) = \frac{\rho_{\sigma, c}(x)}{\rho_{\sigma, c}(\Lambda^+_{q}(A))},
\]

where \( \rho_{\sigma, c}(x) \) is the smoothing parameter.\( \sigma > 0 \) is a constant real, smoothing parameter \( \eta_q(A) \) is defined to be the smallest positive \( \sigma \) satisfying \( p_{1/\sigma}(\Lambda^+_{q}\{0\}) \leq \epsilon \).

About smoothing parameter, the following holds.

1. When \( \sigma > \eta_q(A) \), every coset of \( A \) has roughly equal mass.

2. \( \eta_q(\Lambda^+_{q}(A)) > o(\sqrt{\log m}) \) hold with an overwhelming probability.

There are several lemmas which are important for the design in this paper.

**Lemma 1:** For parameters \( (n, k, m, q, w = n[\log q], s = o(\sqrt{\log m})) \), \( D_{\mathbb{Z}_q^{n \times w}, \sigma} \) is a Gaussian distribution. There is a PPT algorithm \( \text{GenTrap}(\bar{m}, w, H) \) in which \( H \in \mathbb{Z}_q^{n \times n} \) is a tag, outputs a matrix \( A \in \mathbb{Z}_q^{n \times (\bar{m}+w)} \) statistically close to the uniform distribution and with its G-trapdoor \( R \leftarrow D_{\mathbb{Z}_q^{n \times w}, \sigma} \).

**Lemma 2:** For the same parameters as lemma 1, inputting \( A \in \mathbb{Z}_q^{n \times m+w}, H \in \mathbb{Z}_q^{n \times m} \), G-trapdoor \( R \in \mathbb{Z}_q^{m \times w} \) and an aim vector \( u \in \mathbb{Z}_q^n \), there is a PPT algorithm \( \text{SampleD}(A, R, H, u, s) \) outputs vector \( e \) whose distribution is close to the Gaussian distribution.

As we have shown that the Gaussian distribution of the sampled vector can protect the information of the trapdoor information. It is very important to the security of the lattice-based signature scheme. For this reason, the \( \text{SampleD} \) algorithm [20] can be used to design the sign algorithm of the lattice-based scheme.

**Lemma 3:** \( t_i \in \mathbb{Z}_m \) and \( x_i \) are mutually independent random variables sampled from a Gaussian distribution \( D_{\mathbb{Z}_q^{n \times m}, \sigma} \) over \( t_i + A \) for \( i = 1, 2, \cdots, k \) in which \( A \) is a lattice and \( \sigma \in \mathbb{R} \) is a parameter. Let \( c = (c_1, \cdots, c_k) \in \mathbb{Z}_q^k \) and \( g = \gcd(c_1, \cdots, c_k) \), \( t = \sum_{i=1}^k c_i t_i \). If \( \sigma > ||c||_\sigma(A) \) for some negligible number \( \epsilon \), then \( z = \sum_{i=1}^k c_i x_i \) statistically closes to \( D_{\mathbb{Z}_q^{n \times m}, \sigma} \).

**Lemma 4:** [16] Given a random matrix \( A \in \mathbb{Z}_q^{n \times m} \) whose columns generates \( \mathbb{Z}_q^n, \epsilon \in (0, 1) \) and \( s > \eta_q(A^+_{q}(A)) \). If \( e \sim D_{\mathbb{Z}_q^{n \times m}, s} \), then the distribution of the syndrome \( u = Ae(\text{mod}q) \) is within statistical distance \( 2\epsilon \) of the uniform distribution.

The result of Lemma 4 still holds if we expanded vector \( e \) to be matrix

**D. INCREMENTAL SIGNATURE**

**Definition 3:** An incremental signature scheme consists as a 4-tuples of algorithms \( Kg, \text{Sign}, \text{InSig}, \text{Verify} \).

\( Kg \): Inputting a security parameter \( n \), it returns a key pair \((sk, pk)\) where \( sk \) is the private key and \( pk \) the public key.
**Sign**: It is a PPT algorithm, inputting private key \( sk \), a document \( D \) and its tag \( id \), that returns the signature \( s \) in which the document is divided into a sequence of blocks of fix size.

**InSig**: It is an incremental update algorithm, which inputs \( sk, (D, id) \), a modification file \( D’ \) and the primitive signature \( s \), that returns the modified signature \( s’ \) of the modification file \( (D’, id) \).

**Verify**: It is a deterministic polynomial time algorithm that inputs a public key \( pk \) and a signature \( s \) of document \( (D, id) \) not generated by Sign algorithm or the InSig algorithm and returns 1 if the signature is valid, 0 otherwise.

The main security requirement of the incremental signature schemes is the existential unforgeability under the chosen message attack.

**Definition 4**: Definition

An incremental signature scheme is existential unforgeable under a chosen message attack if any PPT adversary’s advantage is negligible in the following game.

1. Setup. The challenger obtains a public and private key pair \((pk, sk)\) by running Kg algorithm. Public key \( pk \) is sent to the adversary.

2. Signing queries. The adversary adaptively queries the signing oracle \( q_1 \) times. The challenger should simulate the Sign algorithm to answer the signing queries.

3. Incremental update queries. The adversary adaptively queries the InSig oracle \( q_2 \) times. The challenger should simulate the InSig algorithm to answer these incremental update queries.

4. Forgery. When the challenger responds all the queries and the adversary satisfies all the responses, the adversary should output a forgeable message and signature pair \( (D^*, id^*, s^*) \). \( D^* \) is not a document signed by the signing oracle nor an updated document whose the signature has been obtained from the incremental update oracle. If \((D^*, id^*, s^*)\) can be accepted by the Verify algorithm, then the adversary wins the game.

### III. INCREMENTAL SIGNATURE SCHEME IN THE STANDARD MODEL

Given security parameters \( n \) and \( q \) > 2, others parameters \((k, m, \omega, \bar{m} = m - \omega, \sigma)\) can be generated by \( n \) and \( q \). The tag of document belongs to \( \mathbb{Z}_2^{n \times m} \) in which \( l \leq n \). Let the primitive matrix \( G \in \mathbb{Z}_2^{n \times m} \) and \( c \leq \sqrt{l} \) be the maximum times of the update of the incremental signature. The sender is Alice and the recipient verifier Bob.

**Kg**: Alice inputs the tag of the G-trapdoor \( I^{n \times n} \) and the random matrix \( \bar{A} \in \mathbb{Z}_2^{n \times m} \) into the algorithm in Lemma 1. Then,

Alice generates \( A \in \mathbb{Z}_2^{n \times m} \) and its G-trapdoor \( R \in \mathbb{Z}_2^{m \times (m - \bar{m})} \).

Alice randomly choose \( I \) matrices over \( \mathbb{Z}_2^{n \times m} \) denoted by \( C_1, \ldots, C_l \) respectively.

Then the public key of Alice is \((A, C_1, \ldots, C_l)\) and private key \( R \).

**Sign**: Message matrix \( D \) is divided into a sequence of blocks \( D = (d_1, d_2, \ldots, d_l) \) in which \( d_i = (d_{1i}, \ldots, d_{mi})^T \in \mathbb{Z}_2^{m} \). The tag of the message is \( id = (id_1, id_2, \ldots, id_l)^T \in \{0, 1\}^l \). Alice generates the signature of \( D \) as follows:

1. Computes \( u = \sum_{j=1}^{l} q^2(1)^{id_j} C_j d_j \pmod{2q} \).
2. Generates a preimage of \( u \) by SampleD algorithm:

\[
e \leftarrow \text{SampleD}(A, R, I, u, \sigma \sqrt{l}).\]

Hence, \((e, id)\) is a signature of \( D \).

**InSig** We suppose that only one block \( d_i \) of document needs to be updated and the others blocks remain unchanged for simply. Giving the updated message \( d_i' \) of \( d_i \), InSig algorithm runs as follows.

1. \( u' = q^2(1)^{id} C_i(d_i - d_i) \pmod{2q} \).
2. Generates a preimage of \( u' \):

\[
e' \leftarrow \text{SampleD}(A, R, I, u', \sigma \sqrt{l}).\]

The incremental signature of message \( D = (d_1, d_2, \ldots, d_{i-1}, d_i', \ldots, d_l) \) is \((e + e', id)\).

**Verify**. To verify the signature \((e, id)\) of the message \( D \), Bob does as follows:

1. Computes \( u = \sum_{j=1}^{l} q^2(1)^{id_j} C_j d_j \pmod{2q} \).
2. Accepts \((e, id)\) if and only if:

\[
(a). Ae = u \pmod{2q}; \ (b). \|e\| \leq \sigma \sqrt{m}.
\]

### IV. SCHEME ANALYSIS

#### A. CORRECTNESS

For security parameters \( n \) and \( q \) > 2, we set \( k = \left\lceil \log_2 q \right\rceil \), \( \bar{m} = O(kn) = m - \omega \), \( \omega = 2nk \) and \( \sigma = O(\sqrt{nk} \omega \sqrt{\log n}) \).

Then under the given parameters, the sign key of Alice in this paper is a G-trapdoor [20]. Hence, the SampleD algorithm in Lemma 2 works. Then, the correctness of the proposed scheme can be shown as follows.

Case 1: For a primitive signature \((e, id)\), it is an output of the SampleD algorithm. According to literature [20], we know that \( Ae = u \pmod{2q} \) and \( \|e\| \leq \sigma \sqrt{m} \) holds. Hence it can be accepted by the Verify algorithm.

Case 2: If \((e + e', id)\) is an output of the Insig algorithm and only one block is updated, it also can be accepted by the Verify algorithm. Let \( D = (d_1, d_2, \ldots, d_{i-1}, d_i', \ldots, d_l) \). Then

\[
A(e + e') = Ae + Ae' = \sum_{j=1}^{l} q^2(1)^{id_j} C_j d_j \pmod{2q} + q^2(1)^{id} C_i(d_i' - d_i) \pmod{2q}.
\]

On the other hand, \( \|e + e'\| \leq 2\sigma \sqrt{m} \leq \sigma \sqrt{m} \) holds.
So that \((e + e', id)\) is a signature of the message vector \(D = (d_1, d_2, \ldots, d_l, d'_1, \ldots, d_l)\).

Case 3. If the message \((\sum_{i=1}^{c'} e_i, id)\) is an output of the Insig algorithm and \(c' \leq e\) blocks of document are updated, then this incremental signature also can be accepted by the Verify algorithm. Let the updated message be \(D = (d_1, d_2, \ldots, d_l, d'_1, \ldots, d_l)\) for simply. Then,

\[
||\sum_{i=1}^{c'} e_i|| \leq c' \frac{\sigma}{\sqrt{l}} \sqrt{m} \leq \sigma \sqrt{m}.
\]

On the other hand,

\[
A(\sum_{i=1}^{c'} e_i) = \sum_{j=1}^{l} q^2(\sum_{i=1}^{c'} (-1)^i d_i) C_j d_j.
\]

The correctness of the proposed scheme is proven.

### B. SECURITY

**Theorem 1:** If there is an adversary can win the unforgeability game of the incremental signature scheme with a probability \(\varepsilon\), there is a challenger can solve the SIS problem with a probability approaching to \(\varepsilon/(q - 1)\).

Suppose there exists a PPT adversary \(A\) gaining an advantage \(\varepsilon\) for winning the unforgeability game, a PPT challenger \(C\) can be constructed to solve the SIS problem with advantage \(\varepsilon/(q - 1)\).

Suppose that the challenger \(C\) wants to solve an SIS instance \((x_0 \in \mathbb{Z}_q^m, q, n, \sigma, L)\). That is, \(C\) hopes to find a vector \(e\) satisfying \(||e|| \leq 2\sigma \sqrt{m}\) and \(A_0e = 0\) (mod 2q). \(C\) does as follows:

1. Sets \(A = q^2A_0\) (mod 2q).
2. Randomly chooses \(E_i\) for \(i = 1, 2, \ldots, l\) according to the distribution \(D_q^{n \times m}\) where \(s = \frac{a}{\sigma}\) and ensures row vectors of these matrices are linearly independent. According to [16], we can ensure that \(E_i\) are row linearly independent by at most choosing \(n^2\) vectors.
3. Sets \(C_i = A_0E_i\) (mod 2q).
4. The Gaussian parameter is defined to be \(L\sigma\).

\(C\) sends \(\{A, \{C_i\}_{i=1}^{l}\}\) to \(A\) as the public key. To finish the security game, \(C\) keeps a list to store the answers to the signature oracle. The Gaussian parameter is defined to be the sufficiently large number \(L\sigma\).

Since \(A_0\) is random and uniform, then \(A = qA_0\) (mod 2q) would also be random and uniform. On the other hand, since the Gaussian parameter \(s\) is larger than the smoothing parameters, then the public matrices \(C_i\) are closed to the uniform distribution. As result, \(C\) can simulate perfectly the attack environment.

**Sign Query.** When \(C\) needs to answer a sequence of sign queries, it firstly checks the freshness of this query by the list. If the message \(D\) had been queried then the same answers are returned. For a fresh message, if \(D_i\) is represented by \(l\) blocks \(d_{ij} \in \mathbb{Z}_q^m\) for \(1 \leq j \leq l\), \(C\) chooses a fresh identifier \(id^{(i)} \in\{0, 1\}^l\) and computes \(e_i = \sum_{j=1}^{l} (-1)^{id_{ij}} E_j d_{ij}\) as the signature of the query message. \(C\) outputs \((id^{(i)}, e_i)\) as the answer. \(C\) stores \((id^{(i)}, D_i, *, e_i)\) to the list \(L\).

The adversary verifies the signature by the Verify algorithm. In fact,

\[
||e_i|| = ||\sum_{t=1}^{l} (-1)^{id_{ij}} E_j d_{ij}|| \leq \frac{l}{\sqrt{q}} \sqrt{m} \sqrt{t} = \frac{\sigma}{\sqrt{L}} \sqrt{m},
\]

and

\[
Ae_i = q^2A_0e_i = q^2A_0(\sum_{t=1}^{l} (-1)^{id^{(i)}} E_j d_{ij})
= \sum_{t=1}^{l} q^2(-1)^{id^{(i)}} A_0E_j d_{ij} \text{ (mod 2q)}
= \sum_{t=1}^{l} q^2(-1)^{id^{(i)}} C_j d_{ij} \text{ (mod 2q)}.
\]

**Incremental Update Queries.** \(A\) can query the incremental update oracle adaptively. For a message \(D_l\) and the updated message \(D'_l\) to \(j\)th block of \(D_l\), the challenger firstly checks the list \(L\) to sure that the message \(D_l\) has been queried and \(D'_l = (d_{1j}, \ldots, d_{lj}, \ldots, d_{lj})\) has never been queried. Otherwise, if \(D'_l\) has been queried then the same answer is responded. For a fresh incremental update query, the challenger firstly reads \((id^{(i)}, D_l, e_i)\) from the list \(L\). Then the challenger computes \((d_{ij} - dij)\) (mod 2q) and \(e'_i = (-1)^{id^{(i)}} E_i (d_{ij}' - d_{ij})\).

Hence, the incremental signature is \((id^{(i)}, e_i + e'_i)\) (mod 2q)) and \((id^{(i)}, D_l, D'_l, e_i + e'_i)\) is stored into the List.

\[
A(e + e') = Ae + Ae'
= \sum_{j=1}^{l} q^2(-1)^{id^{(i)}} A_0E_j d_{ij} \text{ (mod 2q)}
+ q^2(-1)^{id^{(i)}} A_0E_j (d_{ij}' - d_{ij}) \text{ (mod 2q)}.
= \sum_{j=1}^{l} (q^2(-1)^{id^{(i)}} C_j d_{ij}
+ q^2(-1)^{id^{(i)}} C_j d_{ij}' \text{ (mod 2q)}.
\]

On the other hand, \(||e + e'|| \leq L\sigma \sqrt{m}\) holds. As a result, in view of the adversary, \((id^{(i)}, e_i + e'_i)\) (mod 2q)) is a signature of \(D'_l\).

If more than one block of document are asked to be update by \(A\), \(C\) can answer the query by the similar operations as shown above mentioned in which only one block need to be update.

After both the Sign and the Incremental update queries are finished, \(A\) forges a signature \((e^*, id^*)\) of the message \(D^*\) with the probability \(\varepsilon\) in which \(D^*\) is never queried in the Sign or Incremental Update queries.

The challenger computes \(e = \sum_{j=1}^{l} (-1)^{id^*_j} E_j d^*_j\).

Then

\[
Ae = \sum_{j=1}^{l} q^2(-1)^{id^*_j} C_j d^*_j \text{ (mod 2q)}
\]

Hence \(A(e^* - e) = 0\) (mod 2q).

We know that \(qA_0(e^* - e) = 0\) (mod 2q).
TABLE 1. Efficiency comparison.

| Schemes | Public key size (bits) | Signature Length (bits) |
|---------|------------------------|-------------------------|
| Our scheme | \(mn(l + 1)\log(2q)\) | \(m\log(2q) + l\) |
| [10] | \((m\log pq + (l + 1)\times n\log p + d\times n\log p + 2m\log pq\) |

It means \(A_0(e^* - e) = 2lc(\mod 2q)\) where \(0 \leq l < q\) and \(c \in \mathbb{Z}_{2q}^n\).

Since \(A_0\) is uniform and \(e^*, e\) are Gaussian vectors, according to [16] and Lemma 2, \(A_0(e^* - e)\) is a uniform vector with an overwhelm probability. Hence the coefficient \(l = 0, 1, \cdots, (q - 1)\) with equal probability. On the other hand, \(|e^* - e| \leq 2Ls\sqrt{m}\) and \(e^* \neq e\) with probability \(1 - 2^{\omega(\log n)}\) [16]. It is clear that \(C\) solves the SIS problem if \(k = 0\) holds with a probability \(e(1 - 2^{\omega(\log n)})/(q - 1)\).

As a result, \(C\) would solve the SIS problem with a probability \(\frac{1}{q-1} - \frac{2^{\omega(\log n)}}{q-1}\).

C. EFFICIENCY

We give a comparison about the public key size and the signature length between the proposed scheme and the scheme in [10]. Let \((n, m)\) be the row and column of the lattice matrix and \(l\) be the length of the tag respectively. Then Table 1 gives the details of the space efficiency comparison. Furthermore, to find the concrete comparison effect, we give three group concrete parameters as shown as Table 2, under which the message length \(|ID|\) equal to 48K, 90.75Kb and 180.5Kb respectively. Under these parameters, more concrete comparison between this paper and [10] are given in Figure 2 and Figure 3. It can be found that the proposed scheme shares some advantages with respect about the public key size and signature length. Especially the signature length of the proposed scheme has been reduced efficiently. As a result, the space efficiency of the proposed scheme can help to save the storing resource and the bandwidth when we consider the application of the proposed scheme in fog computing.

To check the computing speed of the proposed scheme, we firstly analyze the whole computing costs of the schemes in [10] and this paper by Table 3. In Table 3, let \(T_s, T_m, T_a, T_{sampled}\) denote the running times of random Gaussian sample, multiplication of matrices and 0-1 vectors, addition of vectors and SampleD algorithm respectively. We only consider to update one block in Table 3. And then, we simulated the proposed scheme by using Java language on PC with Intel(R) Core (TM) i7-8700K (3.7GHz), 32G RAM. We change the parameter \(l\) to check the running times of Sign and InSign algorithms and the others parameters are set to be the firstly group parameters in Table 2. More precisely, \(l = 8, 16, 24, 32, 40, 48\) are considered in the simulation. All the simulations are run 10 times and the average running time are used in the following analysis. We also test the scheme in [10] on the same PC. Then Figure 4 shows the comparison details of the Sign algorithm. If we only consider that only one block is updated in the InSig algorithm, the running times comparison of our scheme with [10] is 0.181s vs 0.446s. That is, the running time of InSig in our proposed scheme is about 40.5% of that in [10]. In order to check the efficiency of the InSig algorithm, more blocks updates are considered in the following simulation. Then Figure 5 shows the running times of the InSig algorithm in the simulation.

We explain the simulation results in Figure 4 and Figure 5 as follows.

1. When we run the Sign algorithm, we use parallel computing to compute the multiplication of matrices and vectors.

FIGURE 2. Signature length comparison.

FIGURE 3. Public key size comparison.

TABLE 2. Parameters choice.

| \(|D|\)  | \([10] (n, d, \omega, p, q)\) | Our scheme \((n, k, q, m, l)\) |
|--------|-----------------------------|-----------------------------|
| 48Kb   | \((128, 12288, 32, 2^{44}, 2^{42})\) | \((128, 32, 2^{32} - 1, 12288, 32)\) |
| 90.75Kb| \((128, 16896, 44, 2^{44}, 2^{42})\) | \((128, 44, 2^{44} - 1, 16896, 44)\) |
| 180.5Kb| \((256, 33792, 44, 2^{44}, 2^{42})\) | \((256, 44, 2^{44} - 1, 33792, 44)\) |
FIGURE 4. Run time of Sign.

FIGURE 5. Run time of InSig.

FIGURE 6. CPU status in parallel computing.

in our simulation. And we only consider the online time of the SampleD algorithm in our simulation. In fact, according to the perfect work in [20], there are many off-line operations in SampleD algorithm can be finished by pre-computing. As a result, these parallel computing and pre-computing operations have improved the run speed of our proposed scheme. At last, Figure 6 shows the CPU status when we used the parallel computing to improve the run speed of Sign algorithm in our simulation.

2. Under the given parameters as shown as the first entry of Table 2, the size of one block of the message is about 1.5K. Hence, we use the InSig algorithm to update 1-6 blocks of the message to check the running speed. In this process, we do not use the parallel computing. Figure 5 shows the running time of the InSig algorithm. In fact, if we need update more than one blocks, the parallel computing also can be used to compute the multiplication of matrices and vectors. Clearly, the InSig algorithm can achieve the signature update with more little computation. Hence the proposed scheme helps us to update the data with less cost than to re-sign the data. Figure 4 and Figure 5 show that the advantage of the incremental signature scheme would be more clearer when it is used to update the bigger document.

V. CONCLUSIONS

Data update is a common demand of the fog computing. In order the achieve the authentication of the updated data efficiently, we present an efficient lattice-based incremental signature scheme in the standard model. The security of the proposed incremental signature scheme is based on the standard SIS problem. By using the proposed scheme, the updated data can be authenticated with more less cost than re-sign the whole updated data. The analysis shows that the computing speed of the proposed scheme can be easily improved by parallel computing and pre-computing operations. Compared with a known incremental signature scheme over lattice, the proposed scheme shares some advantages about small public key size and signature length. Hence, the proposed scheme gives an efficient solution to the authentication of the updated data in fog computing. There are still many topics that need to be studied, such as how to reduce the space size of the lattice-based incremental signature scheme, especially the public key size. On the other hand, how to achieve more incremental update operations by lattice tools, such as the deletion and insertion of the message, needs to be studied in the future.

REFERENCES

[1] M. Aazam and E.-N. Huh, “Fog computing: The cloud-IoT-IoE middleware paradigm,” IEEE Potentials, vol. 35, no. 3, pp. 40–44, May 2016.
[2] A. Yousefpour, C. Fung, T. Nguyen, K. Kadiyala, F. Jalali, A. Niakanlahiji, J. Kong, and J. P. Jue, “All one needs to know about fog computing and related edge computing paradigms: A complete survey,” J. Syst. Archit., vol. 98, pp. 289–330, Sep. 2019.
[3] M. Mukherjee, L. Shu, and D. Wang, “Survey of fog computing: Fundamental, network applications, and research challenges,” IEEE Commun. Surveys Tuts., vol. 20, no. 3, pp. 1826–1857, 3rd Quart., 2018.
[4] F. Bonomi, R. Milito, J. Zhu, and S. Addepalli, “Fog computing and its role in the Internet of Things,” in Proc. 1st Ed. MCC Workshop Mobile Cloud Comput., 2012, pp. 13–16.
[5] B. P. L. Lau, S. H. Marakkalage, Y. Zhou, N. U. Hassan, C. Yuen, M. Zhang, and U.-X. Tan, “A survey of data fusion in smart city applications,” Inf. Fusion, vol. 52, pp. 357–374, Dec. 2019.
[6] J. Ni, K. Zhang, X. Lin, and X. Shen, “Securing fog computing for Internet of Things applications: Challenges and solutions,” IEEE Commun. Surveys Tuts., vol. 20, no. 1, pp. 601–628, 1st Quart., 2018.
[7] M. Mukherjee, R. Matam, L. Shu, L. Maglaras, M. A. Ferrag, N. Choudhury, and V. Kumar, “Security and privacy in fog computing: Challenges,” IEEE Access, vol. 5, pp. 19293–19304, 2017.
[8] L. Khati and D. Vergnaud, “Analysis and improvement of an authentication scheme in incremental cryptography,” in Selected Areas in Cryptography—SAC (Lecture Notes in Computer Science), vol. 11349. Berlin, Germany: Springer-Verlag, 2018, pp. 50–70.
[9] M. Bellare, O. Goldreich, and S. Goldwasser, “Incremental cryptography: The case of hashing and signing,” in Proc. Crypto., in Lecture Notes in Computer Science, vol. 839. Berlin, Germany: Springer-Verlag, 1994, pp. 216–233.
[10] J. Chen, M. Tian, C. Gao, and Z. Chen, “A lattice-based incremental signature scheme,” IEEE Access, vol. 7, pp. 21201–21210, 2019.
[11] I. Mironov, O. Pandey, O. Reingold, and G. Segev, “Incremental deterministic public-key encryption,” *J. Cryptol.*, vol. 31, no. 1, pp. 134–161, Jan. 2018.

[12] D. Micciancio, “Oblivious data structures: Applications to cryptography,” in *Proc. STOC*, 1997, pp. 456–464.

[13] M. Bellare, O. Goldreich, and S. Goldwasser, “Incremental cryptography and application to virus protection,” in *Proc. STOC*, 1995, pp. 45–56.

[14] M. Bellare and D. Micciancio, “A new paradigm for collision-free hashing: Incrementality at reduced cost,” in *Proc. Eurocrypt*, in Lecture Notes in Computer Science, vol. 1233. 1997, pp. 163–192.

[15] V. Lyubashevsky and D. Micciancio, “Asymptotically efficient lattice-based digital signatures,” in *Proc. TCC*, in Lecture Notes in Computer Science. Berlin, Germany: Springer-Verlag, 2008, pp. 37–54.

[16] C. Gentry, C. Peikert, and V. Vaikuntanathan, “Trapdoors for hard lattices and new cryptographic constructions,” in *Proc. 40th Ann. ACM Symp. Theory Comput. (STOC)*, BC, Canada, 2008, pp. 197–206.

[17] J. Alwen and C. Peikert, “Generating shorter bases for hard random lattices,” in *Proc. 26th Int. Symp. Theor. Aspects Comput. Sci.*, vol. 09001, Freiburg, Germany, pp. 75–86.

[18] D. Cash, D. Hofheinz, and Z. Kiltz, “Bonsai trees, or how to delegate a lattice basis,” in *Proc. Eurocrypt*, in Lecture Notes in Computer Science, vol. 6110. Berlin, Germany: Springer-Verlag, 2010, pp. 523–552.

[19] D. Boneh and D. M. Freeman, “Linearly homomorphic signatures over binary fields and new tools for lattice-based signatures,” in *Proc. PKC*, in Lecture Notes in Computer Science, vol. 6571. Berlin, Germany: Springer-Verlag, 2011, pp. 1–16.

[20] D. Micciancio and C. Peikert, “Trapdoors for lattices: Simpler, tighter, faster, smaller,” in *Advances in Cryptology—EUROCRYPT* (Lecture Notes in Computer Science), vol. 7237. Berlin, Germany: Springer-Verlag, 2012, pp. 700–718.

[21] D. Micciancio and O. Regev, “Worst-case to average-case reductions based on Gaussian measures,” in *Proc. 45th Ann. IEEE Symp. Found. Comput. Sci.*, Rome, Italy, Oct. 2004, pp. 372–381.

JUNQUAN WANG received the Ph.D. degree in computer science from Shandong University, Jinan, China. He is currently a Teacher with Shandong Jianzhu University. His research interests include security of the IoT and cloud computing.

WENFENG YANG received the Ph.D. degree in network security from Xidian University, Xi’an, China, in 2011. He is currently a Security Specialist with Tianbo Electronic Information Technology Company, Ltd. His research interest includes cloud security.

FENGHE WANG received the Ph.D. degree in cryptography from Xidian University, Xi’an, China, in 2012. He is currently an Associate Professor with Shandong Jianzhu University. His research interests include network security and post-quantum cryptography.

SHAOQUAN SHI is currently pursuing the master’s degree with Shandong Jianzhu University. His research interests include lattice-based cryptography and digital signature.

**IEEE Access**

**J. Wang et al.: Lattice-Based Incremental Signature Scheme for the Authenticated Data Update in Fog Computing**

VOLUME 8, 2020