Identity-Based Remote Data Integrity Checking of Cloud Storage From Lattices

Zhangyun Liu
School of Information and Software Engineering
University of Electronic Science and Technology of China
Chengdu, China
Email: 604614445@qq.com

Yongjian Liao*
School of Information and Software Engineering
University of Electronic Science and Technology of China
Chengdu, China
**Corresponding Author*
Email: liaoyj@uestc.edu.cn

Xiaowei Yang
School of Information and Software Engineering
University of Electronic Science and Technology of China
Chengdu, China
Email: 985780275@qq.com

Yichuan He
School of Information and Software Engineering
University of Electronic Science and Technology of China
Chengdu, China
Email: 494108376@qq.com

Kun Zhao
School of Information and Software Engineering
University of Electronic Science and Technology of China
Chengdu, China
Email: 243199027@qq.com

***Abstract*—In cloud storage, remote data integrity checking is considered as a crucial technique about data owners who upload enormous data to cloud server provider. A majority of the existing remote data integrity checking protocols rely on the expensive public key infrastructure. In addition, the verification of certificates needs heavy computation and communication cost. Meanwhile, the existing some protocols are not secure under the quantum computer attacks. However, lattice-based constructed cryptography can resist quantum computer attacks and is fairly effective, involving matrix-matrix or matrix-vector multiplications. So, we propose an identity-based remote data integrity checking protocol from lattices, which can eliminate the certificate management process and resist quantum computer attacks. Our protocol is completeness and provably secure based on the hardness small integer solution assumption. The presented scheme is secure against cloud service provider attacks, and leaks no any blocks of the stored file to the third party auditor during verification stage, namely the data privacy against the curiosity third party auditor attacks. The cloud service provider attack includes lost attack and tamper attack. Furthermore, the performance analysis of some protocols demonstrate that our protocol of remote data integrity checking is useful and efficient.**

***Keywords*-data integrity checking; ID-Based cryptography; lattices; provably secure;**

I. Introduction

The cloud storage is a significant service of cloud computing, which allows data users to upload their abundant data from the local systems to the cloud server. It can relieve the burden of storage management and maintenance for data users. With these appealing features, there are many individuals and businesses start to store the data in the cloud. However, the cloud storage faces all kinds of security challenges, including hardware or software disasters, operational errors, external attackers and internal adversaries, which hinder development of cloud storage[1]. As a result, it is necessary for data owners to check the integrity of their stored data.

How to check integrity of remote data? The most s-traightforward way is downloading the whole file from the cloud storage server. Due to the cloud data is huge, this method is limited by bandwidth cost and communication efficiency. Moreover, traditional mechanisms for data integrity checking, including the hash function, digital signature and message authentication code, cannot apply here directly since the verifiers do not have a copy of the stored file. The provable data possession(PDP) was first proposed by Ateniese et al.[2] to provide accurate verification of data integrity with low computation and communication cost, which proposed two practical PDP schemes from the RSA assumption. Following Ateniese et al.'s work, the research of remote data integrity checking has considerable received attention from research communities in academia and company [3][4][5][6][7][9][22][23]. Curtmola et al.[3] is a multiple replica provable data possession scheme. Shacham et al [4] proposed the notion of compact proofs of retrievability by making use of the homomorphic authenticators from BLS signature. Ateniese et al.[5] proposed a dynamic PDP scheme based on hash functions and symmetric key encryptions, but this scheme does not support insert operation. In order to support the insert operation, Erway et al.[6] proposed a full-dynamic PDP scheme, which means the data owner can insertion, modification, deletion and appending. Wang et al.[7] improved the previous dynamic PDP models by employing the Merkle Hash Tree (MHT). Howerver, Liu et al.[8] showed this scheme may lead to replace attack. There are some schemes that open discussion about the security issues of some previously proposed PDP schemes[9][10]. In remote data integrity checking, data privacy against the third party auditor(TPA) is essential since the confidential or sensitive data of cloud users don't want to be known by the TPA. The privacy-preserving public auditing scheme [11] is the first proposed. Yu et al.

The previous researchs on remote data integrity checking take advantage of the RSA assumption and discrete logarithm problem. Recently, lattice-based cryptography has become a hot cryptography topic on account of its security on quantum computers. Many ID-Based cryptography schemes for cloud have been proposed[13][14][15][16]. Xu et al.[17] proposed a public verifiable proof of storage protocol from lattice assumption, which adopted the linearly homomorphic signature from lattices. Unfortunately, Xu's protocol doesn't support public verification. Subsequently, Liu et al.[18] proposed a proof of storage protocol with public auditing form lattice assumption, which claimed that the protocol satisfy security properties, such as secure against the lost attack and tamper attack from the cloud server provider(CSP), and secure against the curiosity attack from the TPA. Zhang et al.[19] demonstrate that Liu's scheme is not secure, namely any malicious CSP can cheat the TPA and the file blocks maybe recovered by any TPA through solving some linearly equations. Meanwhile, they give the improvements about that protocol. However, lattice-based protocols of remote data integrity checking is still rarely.

Currently, a majority of the existing remote data integrity checking constructions rely on public key infrastructure(PKI). These constructions incur complex key management procedures since certificate generation, certificate storage, certificate update and certificate revocation are time-consuming and expensive. To simplify certificate management, Shamir [20] presented ID-Based cryptographic concept. The first ID-Based PDP was proposed by the Ref.[21] . In 2015, Wang et al. [22] proposed ID-Based distributed provable data Possession scheme, which supports verification for single owner and multiple clouds simultaneously. Zhou et al. [23] proposed an ID-Based batch provable data possession (ID-BPDP) scheme, which is the first ID-Based provable data possession scheme supporting batch verification for multiple owners and multiple clouds. Qin et al.[24] proposed certificateless proxy re-encryption scheme for data sharing in cloud computing satisfying secure against adaptive chosen ciphertext attack under a stronger security model. Yu et al. [12] proposed a new construction of ID-Based remote data integrity checking protocol by making use of key-homomorphic cryptographic primitive, which provide detailed security proofs of the this protocol, including the soundness and zero-knowledge privacy of the stored data.

$Our\ Contributions$. We define an ID-Based remote data integrity checking protocol, which can eliminate numerous certificate management procedure and resist quantum computer attacks. We proved it is completeness and provably secure based on the hardness SIS assumption. This protocol is security against the CSP attacks and the data privacy against the curiosity TPA attacks. The analysis of protocols indicate that our protocol of the remote data integrity checking is useful and efficient.

$Organizations$. The rest of the paper is organized as follows. In Sect. 2, we describe the system model and security model for ID-Based remote data integrity checking protocol. Section 3 proposes a concrete ID-Based remote data integrity checking protocol construction. Section 4 gives the analysis of completeness, security against the CSP, data privacy preserving, and performance. Section 5 concludes our work.

II. Preliminaries and Definitions

In this section, we firstly describe system architecture and give the definition of ID-Based remote data integrity checking. Then, we define security model for a remote data stored the cloud server.

A. Definition of a System Architecture

The remote data integrity checking protocols with public verifiability enable anyone to audit the integrity of the stored cloud data. To make the description of the protocol clearly, we assume there exits a third party auditor (TPA) who has expertise and capabilities to do the verification work. With this in mind, a representative ID-Based data integrity checking system model for the public auditing is illustrated in Fig.1. It includes four major entities,namely the data owners or cloud users, the cloud server provider(CSP), or a TPA, and PKG are involved in an ID-Based remote data integrity checking. The cloud users can be regarded as either individuals or companies. They have huge files to be stored and rely on the CSP for data maintenance and computation. However, CSP has a massive amount of storage space and computation resources and provides data storage service. The TPA's work is to check the integrity of remote data stored on the cloud. The PKG is responsible for generating secret private keys for users using their identity.

An ID-Based remote data integrity checking protocol consists of six algorithms, namely Setup, Extract, SignGen, Challeng, ProofGen and ProofCheck. The details are as follows.

Setup(n): This algorithm is a probabilistic algorithm. It takes a security parameter n as input and outputs the system parameters $param$, the master secret key msk and public key pk.

Extract$(param, msk, id)$: It is a probabilistic algorithm run by the PKG. It takes the system parameters $param$, the

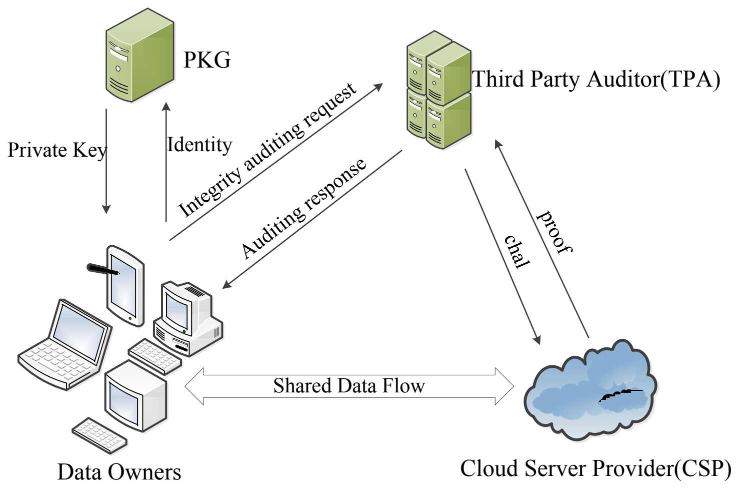


Figure 1. The system model of ID-Based remote data integrity checking.

master secret key msk and a user's identity $id \in \{0,1\}^*$ as input, outputs the secret key sk_{id} that corresponds to the identity information id.

SignGen$(param, \tau, sk_{id}, M, id)$: The file M can to be divided into l blocks, This algorithm is a probabilistic algorithm run by the data owner with identity id. It takes the system parameters $param$, the secret key of the user sk_{id} and a file $M \in \{0,1\}^*$ to upload as input, outputs the signatures $\mathbf{e}_1, \ldots, \mathbf{e}_l$ of each file block $\mathbf{u}_1, \cdots, \mathbf{u}_l$, which will be stored on the cloud together with the file M.

Challenge(τ, M, id): It is a randomized algorithm run by the TPA. It takes the system parameters $param$, the data owner's identity id, and $\tau \in \{0,1\}^*$ a unique identity of the file M, outputs a challenge $chal$ for the file on behalf of the user id.

ProofGen$(param, chal, M, \varPhi, id)$: It is a probabilistic algorithm run by the CSP. It takes the system parameters $param$, the challenge $chal$, the data owner's identity id, the combine signature $\varPhi$, the file M as input, outputs a data possession proof P of the challenged blocks.

ProofCheck$(param, P, chal, id, \tau)$: It is a deterministic algorithm run by the TPA. It takes the system parameters $param$, the challenge $chal$, the data owners identity id, and an alleged data possession proof P as input, outputs 1 or 0 to indicate whether the file M keeps integrity.

B. *Security Model*

In our cloud storage system, TPA is considered to be honest and curious. It performs honestly during the whole checking procedure, but it is curious in the sense that it is willing to learn some information about the received data. However, the CSP is considered to be dishonest. So, in our protocol, we consider two security properties of security against the CSP attacks, and privacy against the TPA attack (data privacy preserving). Wang et al[22] proposed an unforgeability security notion for their protocol. In their protocol, the challenged data blocks are not allowed for TagGen queries, which is not consistent with the cloud storage environment in reality. Shacham and waters [4] proposed the soundness security model of ID-Based PDP protocols. Yu et al. [12] formalized the security model of soundness for ID-Based remote data integrity checking protocols for the first time and proved soundness of the generic construction in their new security model.

Security Against the CSP: This security game indicates that an adversary cannot successfully generate a valid proof without possessing all the blocks of a user id corresponding to a given challenge. We now formalize the security model of ID-Based remote data integrity checking in random oracle model. An adversary who plays the role of the malicious server and a challenger who represents a data owner are involved. The game consists of the following.

Setup Phase: The challenger runs the Setup algorithm to obtain the system parameters $param$ and the master secret key msk, and forwards $param$to the adversary.

Queries Phase: The adversary makes a number of queries, including hash query, key extract query and sign query adaptively.

1) **Hash query**. According to query, the challenger sends the value of hash function to the adversary.

2) **Extract Query**. Given an identity id, the challenger runs Extract algorithm to gain private key corresponding to id, sends it to the adversary.

3) **SignGen Query**. The file M can to be divided into l blocks. The adversary queries each blocks. The challenger runs SignGen algorithm to generate signatures of the blocks of file M and returns the set of signatures to the adversary.

ProofGen: The SignGen query has been made about the blocks of file M, the adversary can undertake executions of the ProofGen algorithm by specifying an identity id of the cloud user. The challenger plays the role of the TPA and the adversary A behaves as the prover during the proof generation. Finally, the adversary can get the output of P from the challenger when a protocol execution completes.

Output: The adversary chooses a user identity id^*, which must not have queried in key extraction queries and there exists a SignGen query with input id^*. The adversary outputs the description of a prover P^* which is ϵ-admissible defined paper [12]. In addition, there are two types of attacks, namely loss attacks and tamper attacks in the cloud storage system.

Data Privacy Preserving: By data privacy preserving indicates that the TPA retrieves no information of cloud user's data during verification process. In other words, no matter how TPA learns, namely the TPA could have learned by himself without any interaction with the cloud server provider (CSP), that cannot retrieve the blocks of file M.

III. Our Scheme

In this section, we put forward an ID-Based remote data integrity checking protocol for cloud storage from lattices. Our scheme works as follows. In the Setup phase, we make use of TrapGen function to generate master secret key msk. In the key extraction, we adopt basis delegation algorithm to gain the secret key sk corresponding to user's identity $id \in \{0,1\}^*$. In the SignGen phase, we use the Zhang et al's improvents protocol of linearly homomorphic signatures to generate signatures of the blocks [24]. In challenge, the TPA generates $chal$ and sends it to CSP. In the proof phase, the CSP receives the $chal$, computes the corresponding data, obtains the $proof$, and forwards $proof$ to TPA. In checkProof, the TPA verifies rationality. In addition, at the time of construction should pay attention to ensure data privacy-preserving, namely the TPA couldn't know any information about this file. The details of protocol are as follows.

Setup(n): Input a security parameter n, integers $m \geq 2n\log q$, large prime $q \geq \beta\omega(\log n)$, and a real number $\beta = poly(n)$. Four secure hash functions: $H : \{0,1\}^* \rightarrow \mathbb{Z}_q^{m\times m}$, $H_1 : \{0,1\}^* \rightarrow \mathbb{Z}_q^n$, $H_2 : \mathbb{Z}_q^n \rightarrow \mathbb{Z}_q$, $H_3 : \mathbb{Z}_q^{n\times m} \times \{0,1\}^* \rightarrow \mathbb{Z}_q^n$. The user runs $TrapGen(n,m,q)$ to generate a matrix $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$ with a short basis $\mathbf{T}_A \in \mathbb{Z}_q^{m\times m}$ of the lattice $\Lambda_q^{\perp}(\mathbf{A})$. The CSP runs $TrapGen(n,m,q)$ to generate a matrix $\mathbf{Q} \in \mathbb{Z}_q^{n\times m}$ with a short basis $\mathbf{T}_Q \in \mathbb{Z}_q^{m\times m}$ of the lattice $\Lambda_q^{\perp}(\mathbf{Q})$. The system parameter $param$ is $(\mathbf{A},\mathbf{Q},H,H_1,H_2,H_3)$. The user's master key is $msk = \mathbf{T}_A$.

Extract$(param, msk, id)$: Given an identity id, system parameter $param$, and the user's master key $\mathbf{T}_A$, do the following:

1) Set $\mathbf{R} = H(id) \in \mathbb{Z}_q^{m\times m}$, where $\mathbf{R}$ is an invertible matrix.

2) Run the $BasisDel(\mathbf{A},\mathbf{R},\mathbf{T}_A,s)$ algorithm to generate a $\mathbf{T}_{id}$, $\mathbf{T}_{id}$ is a basis $\Lambda_q^{\perp}(\mathbf{B})$, a matrix $\mathbf{B} = \mathbf{AR}^{-1}$, and s is a parameter for Gaussian sampling.

3) Outputs $sk_{id} = \mathbf{T}_{id}$ as the user's private key.

SignGen$(param, \tau, sk_{id}, M, id)$: Given an user's private key $\mathbf{T}_{id}$, an user's data file M. To store user's data file M in cloud server, M needs to be divided into l blocks $\mathbf{u}_1,\cdots,\mathbf{u}_l$, where $\mathbf{u}_i \in \mathbb{Z}_q^m$. Define the tag of the file M to be $\tau \in \{0,1\}^*$.

1) Compute $\mathbf{R} = H(id)$, and $\mathbf{B} = \mathbf{AR}^{-1}$, and compute $\alpha_j (j \leq n)$, $\alpha_j = H_3(\mathbf{B}||\tau||j) \in \mathbb{Z}_q^m$.

2) For each $\mathbf{u}_i$, $1 \leq i \leq l$, the user computes $\lambda_i \in \mathbb{Z}_q^n$: $\lambda_i = H_1(\tau||i) + \mathbf{Qu}_i$.

3) Compute the inner products $h_{ij} = \langle\lambda_i,\alpha_j\rangle$, $1 \leq i \leq l, 1 \leq j \leq n$. $\mathbf{h}_i = (h_{i1},h_{i2},\ldots,h_{in})$. Let $\mathbf{C} = (\alpha_1,\ldots,\alpha_n)$, $\mathbf{h}_i = \mathbf{C}\lambda_i$.

4) Cloud user runs $SamplePre$ algorithm to gain sign of block,

$$\mathbf{e}_i = SamplePre(\mathbf{B},\mathbf{T}_{id},\mathbf{h}_i,\sigma).$$

5) Let $\mathit{\Phi} = (\mathbf{e}_1,\ldots,\mathbf{e}_l)$. The cloud user sends the file $M = (\mathbf{u}_1,\cdots,\mathbf{u}_l$, where $\mathbf{u}_i \in \mathbb{Z}_q^m)$ and corresponding to signatures $\mathbf{e}_1,\ldots,\mathbf{e}_l$ to CSP, and deletes the data file M from local storage.

Challenge(τ, M): To check the integrity of $M = (\mathbf{u}_1,\cdots,\mathbf{u}_l)$, the user sends an auditing request to TPA. Subsequently, the TPA defines the subset of set $[1,l]$ to be $I = \{s_j\}(1 \leq j \leq \theta)$ and $s_1 \leq \ldots \leq s_\theta$. For each element $i \in I$, the TPA chooses a random $c_i \leftarrow \mathbb{Z}_q^*$, and generates $chal = \{\tau,i,c_i\}_{i\in I}$ as the challenge. Then, the TPA forward $chal$ to the CSP.

ProofGen$(param, chal, M, \mathit{\Phi}, \mathbf{T}_Q, \tau)$: Upon receiving $chal = \{\tau,i,c_i\}_{i\in I}$, the CSP firstly chooses the data blocks $\{\mathbf{u}_i\}_{i\in I}$ and the corresponding signatures $\{\mathbf{e}_i\}_{i\in I}$ in the cloud server.

1) The CSP computes $\mathbf{u}_c = \sum_{i=s_1}^{s_\theta} c_i\mathbf{u}_i$, where $\mathbf{u}_c \in \mathbb{Z}_q^m$.

2) The CSP chooses a random vector $\mathbf{w} \leftarrow \mathbb{Z}_q^n$, and runs the algorithm $SamplePre(\mathbf{Q},\mathbf{T}_Q,\mathbf{w},\sigma)$ to generate the signature ξ of $\mathbf{w}$.

3) Compute $\mathbf{u}_c^{'} = \mathbf{u}_c + \xi H_2(\mathbf{w})$, and $\mathbf{e}_c = \sum_{i=s_1}^{s_\theta} c_i\mathbf{e}_i$ $(\mathbf{u}_c^{'},\mathbf{e}_c \in \mathbb{Z}_q^m)$.

4) The CSP sends the response $proof$ information to TPA, namely $proof = (\tau,\mathbf{u}_c^{'},\mathbf{e}_c,\mathbf{w})$.

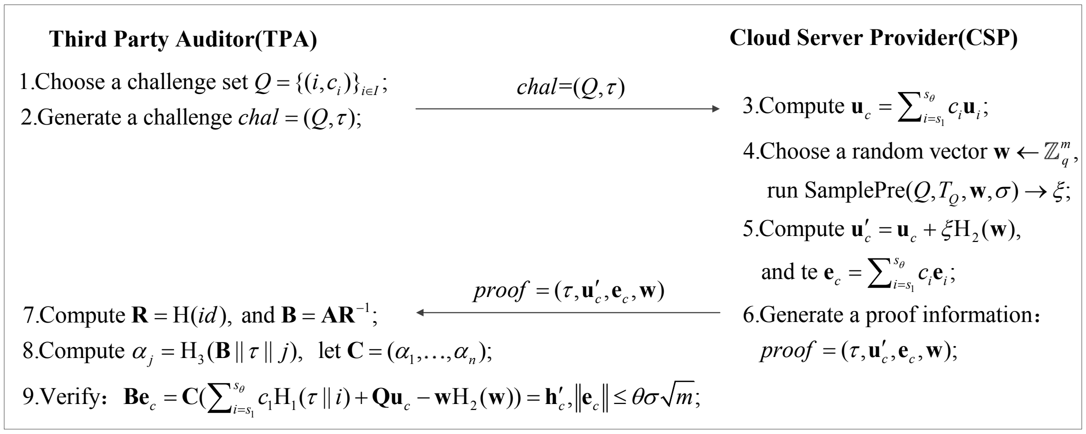
ProofCheck$(param, proof, chal, id, \tau)$: Upon receiving $proof = (\tau,\mathbf{u}_c^{'},\mathbf{e}_c,\mathbf{w})$ from CSP, the TPA does the following:

1) Compute $\mathbf{R} = H(id)$, and $\mathbf{B} = \mathbf{AR}^{-1}$.

2) Compute $\alpha_j (j \leq n)$, $\alpha_j = H_3(\mathbf{B}||\tau||j)$, let $\mathbf{C} = (\alpha_1,\ldots,\alpha_n)^{\top}$.

3) The TPA computes $\lambda_c^{'} = \sum_{i=s_1}^{s_\theta} c_iH_1(\tau||i) + \mathbf{Qu}_c^{'} - \mathbf{w}H_2(\mathbf{w})$, and then computes $\mathbf{h}_c^{'} = \mathbf{C}\lambda_c^{'}$.

4) Finally, the TPA checks $\mathbf{Be}_c^{'} = \mathbf{h}_c^{'}(\bmod q)$, and $||\mathbf{e}_c^{'}|| \leq \theta\sigma\sqrt{m}$. If the two holds, the TPA accepts the $proof$; Otherwise,the $proof$ is invalid. The process of Challenge, GenProof and CheckProof are summerised in Fig.2.


Third Party Auditor(TPA)

1.Choose a challenge set $Q=\{(i,c_i)\}_{i\in I}$;

2.Generate a challenge $chal=(Q,\tau)$;

$\xrightarrow{chal=(Q,\tau)}$

Cloud Server Provider(CSP)

3.Compute $\mathbf{u}_c=\sum_{i=s_1}^{s_\theta} c_i\mathbf{u}_i$;

4.Choose a random vector $\mathbf{w}\leftarrow\mathbb{Z}_q^m$, run SamplePre$(Q,T_Q,\mathbf{w},\sigma)\to\xi$;

5.Compute $\mathbf{u}'_c=\mathbf{u}_c+\xi\mathrm{H}_2(\mathbf{w})$, and te $\mathbf{e}_c=\sum_{i=s_1}^{s_\theta} c_i\mathbf{e}_i$;

6.Generate a proof information: $proof=(\tau,\mathbf{u}'_c,\mathbf{e}_c,\mathbf{w})$;

$\xleftarrow{proof=(\tau,\mathbf{u}'_c,\mathbf{e}_c,\mathbf{w})}$

7.Compute $\mathbf{R}=\mathrm{H}(id)$, and $\mathbf{B}=\mathbf{A}\mathbf{R}^{-1}$;

8.Compute $\alpha_j=\mathrm{H}_3(\mathbf{B}\,\|\,\tau\,\|\,j)$, let $\mathbf{C}=(\alpha_1,\ldots,\alpha_n)$;

9.Verify: $\mathbf{B}\mathbf{e}_c=\mathbf{C}(\sum_{i=s_1}^{s_\theta} c_1\mathrm{H}_1(\tau\,\|\,i)+\mathbf{Q}\mathbf{u}_c-\mathbf{w}\mathrm{H}_2(\mathbf{w}))=\mathbf{h}'_c, \|\mathbf{e}_c\|\le\theta\sigma\sqrt{m}$;

Figure 2. ID-Based remote data integrity checking protocol from lattices.

IV. Analysis

In this section, we show that the proposed scheme achieves the properties of completeness and security. The completeness guarantees the correctness of this protocol. Our protocol is security against the CSP attacks and the data privacy against the curiosity TPA attacks.

A. Completeness

If both the cloud user and CSP are honest, for each valid signature $\mathbf{e}_i$ and a random challenge $chal$, the $proof$ of the CSP can always pass the verification. The completeness of the protocol can be detailed as follows.

$$\begin{aligned}\lambda_c^{'} &= \sum_{i=s_1}^{s_\theta} c_i\lambda_i = \sum_{i=s_1}^{s_\theta} c_i H_1(\tau||i) + \mathbf{Q}\sum_{i=s_1}^{s_\theta} c_i\mathbf{u}_i\\ &= \sum_{i=s_1}^{s_\theta} c_i H_1(\tau||i) + \mathbf{Q}\mathbf{u}_c\\ &= \sum_{i=s_1}^{s_\theta} c_i H_1(\tau||i) + \mathbf{Q}(\mathbf{u}_c^{'} - \xi H_2(\mathbf{w}))\\ &= \sum_{i=s_1}^{s_\theta} c_i H_1(\tau||i) + \mathbf{Q}\mathbf{u}_c^{'} - \mathbf{Q}\xi H_2(\mathbf{w})\\ &= \sum_{i=s_1}^{s_\theta} c_i H_1(\tau||i) + \mathbf{Q}\mathbf{u}_c^{'} - \mathbf{w}H_2(\mathbf{w})\end{aligned}$$

$$\begin{aligned}\mathbf{B}\mathbf{e}_c =& \mathbf{B}\sum_{i=s_1}^{s_\theta} c_i\mathbf{e}_i = \sum_{i=s_1}^{s_\theta} c_i\mathbf{B}\mathbf{e}_i = \sum_{i=s_1}^{s_\theta} c_i\mathbf{h}_i\\ =& \sum_{i=s_1}^{s_\theta} c_i\mathbf{C}\mathbf{e}_i = \sum_{i=s_1}^{s_\theta} c_i\mathbf{C}(H_1(\tau||i) + \mathbf{Q}\mathbf{u}_i)\\ =& \mathbf{C}(\sum_{i=s_1}^{s_\theta} c_i H_1(\tau||i) + \mathbf{Q}\mathbf{u}_c^{'} - \mathbf{w}H_2(\mathbf{w}))\\ =& \mathbf{C}\lambda_c^{'} = \mathbf{h}_c^{'}\end{aligned}$$

B. Security Against the CSP

We prove that our scheme described the above is secure under the security model described in Sect. 2. Regarding to the security, we have the following theorem.

In the random oracle model, if the digital signature scheme is existentially unforgeable and the SIS problem is intractable in lattices, except with negligible probability, no adversary could break the security of the proposed ID-Based remote data integrity checking protocol.

Proof: Given a security parameter n, with the adversary $\mathcal{A}$, we can construct the challenger $\mathcal{C}$ as follows:

Setup: The challenger $\mathcal{C}$ receives a SIS instance($\mathbf{A} \in \mathbb{Z}_q^{n\times m}, q, n, m, \sigma$), and hopes to find a vector $\mathbf{v}$ satisfying $\|\mathbf{v}\| \le 2L\sigma\sqrt{m}$ and $\mathbf{A}\mathbf{v} = 0(\mod q)$. Namely, the challenger $\mathcal{C}$ runs $TrapGen(q,n,m)$ to generate $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$ and a basis $\mathbf{T}_A \in \mathbb{Z}_q^{m\times m}$ of lattice $\Lambda_q^{\perp}(\mathbf{A})$. The matrix $\mathbf{T}_A$ is as the master secret key, and $\mathcal{C}$ sends $\mathbf{A}$ as the public parameter to $\mathcal{A}$. At the same time, the challenger $\mathcal{C}$ also runs $TrapGen(q,n,m)$ to generate $\mathbf{Q} \in \mathbb{Z}_q^{n\times m}$ and a basis $\mathbf{T}_Q \in \mathbb{Z}_q^{m\times m}$ of lattice $\Lambda_q^{\perp}(\mathbf{Q})$. The matrix $\mathbf{T}_Q$ and $\mathbf{Q}$ represent the cloud server's public key and private key.

Query: $\mathcal{A}$ makes a series of queries to $\mathcal{C}$, including hash query, extract query and SignGen query adaptively. In hash query, $\mathcal{C}$ simulates the random oracles. In query phase, $\mathcal{C}$

keeps five lists to store the output of query. L_1: the random oracle H_1 query, L_2: the random oracle H_2 query, L_3: the random oracle H_3 query, L_s: the SigGen query and L_e: the extract query. $\mathcal{A}$ issues the following queries.

H query: $\mathcal{A}$ adaptively queries $id_i(i=1,\ldots,q_H)$, where q_H is the maximum number of H query. The challenger checks the list L_e. If an entry in L_e is found, it returns the same answer to the adversary; Otherwise, $\mathcal{C}$ runs the algorithm $SampleRwithBasis(\mathbf{A})$ to gain a random matrix $\mathbf{R}_i$ satisfying the Gaussian distribution and a basis $\mathbf{T}_i$. The challenger saves tuple $(id_i, \mathbf{R}_i, \mathbf{B}_i, \mathbf{T}_i)$ to the list L_e, and returns $\mathbf{R}_i$ to the adversary.

Extract query: Let q_E denote the maximum number of Extract query. Input id_i, $\mathcal{C}$ checks the list L_e. If $\mathcal{C}$ can find the tuple $(id_i, \mathbf{R}_i, \mathbf{B}_i, \mathbf{T}_i)$ in the list, returns $\mathbf{T}_i$ to $\mathcal{A}$. if the challenger cannot find it, $\mathcal{C}$ runs the algorithm $SampleRwithBasis(\mathbf{A})$ to gain a random matrix $\mathbf{R}_i$ and a basis $\mathbf{T}_i$. $\mathcal{C}$ saves tuple $(id_i, \mathbf{R}_i, \mathbf{B}_i, \mathbf{T}_i)$ to the list L_e, and returns $\mathbf{T}_i$ to $\mathcal{A}$.

H_1 query: $\mathcal{A}$ inputs a tuple (τ_k, i) and queries the tag τ_k. $\mathcal{C}$ $\mathcal{C}$ checks the list in L_1 to gain corresponding to tuple $(\tau_k, \{\beta_{ki}\}_{i\in[l]})$. If $\mathcal{C}$ can't find $(\tau_k, \{\beta_{ki}\}_{i\in[l]})$, $\mathcal{C}$ randomly selects l vectors $\beta_{k1},\ldots,\beta_{kl}$ from $\mathbb{Z}_q^n$. $\mathcal{C}$ saves tuple $(\tau_k, \{\beta_{ki}\}_{i\in[l]})$ to the list L_1, and returns $\{\beta_{ki}\}_{i\in[l]}$ to $\mathcal{A}$.

H_2 query: $\mathcal{A}$ inputs a vector $\mathbf{w}_i$ and queries it. $\mathcal{C}$ checks the list in L_2 to gain corresponding to the value $r_i \in \mathbb{Z}_q$. If $\mathcal{C}$ can't find r_i, he chooses a random r_i from $\mathbb{Z}_q$, saves list L_2 and forward it to adversary.

H_3 query: $\mathcal{A}$ inputs a tuple (τ_k, j, id). First, $\mathcal{C}$ checks the list in L_3 to gain corresponding to tuple $(\tau_k, \{\alpha_{kj}\}_{j\in[n]})$. Then, if it exists L_3, $\mathcal{C}$ returns the results $\alpha_1, \alpha_2, \ldots, \alpha_n$ to $\mathcal{A}$; Otherwise, $\mathcal{C}$ randomly selects n vectors $\mathbf{h}_{kj}$ from the Gaussian distribution satisfying $\|\mathbf{h}_{kj}\| \le \sigma\sqrt{m}$ for $j = 1, 2, \ldots, n$, checks list L_e to find $\mathbf{B}$ and computes $\mathbf{B}\mathbf{h}_{kj} = \alpha_{kj}$ and returns $\alpha_{k1}, \alpha_{k2}, \ldots, \alpha_{kn}$ as the query answer. Finally, the challenger $\mathcal{C}$ saves tuple $(\tau_k, \{\mathbf{h}_{kj}, \alpha_{kj}\}_{j\in[n]}$ to the list L_3.

SigGen query: The file M_k can to be divided into l blocks, namely $M_k = (\mathbf{u}_{k1}, \cdots, \mathbf{u}_{kl})$. $\mathcal{C}$ needs to generate a signature about vectors $\{\mathbf{u}_{ki}\}_{i\in[l]}$. $\mathcal{A}$ asks for the signing for $(id, \tau_k, \mathbf{u}_{ki})$. If the vectors $\mathbf{u}_{ki}$ has been queried, $(id, \tau_k, \mathbf{u}_{ki}, \lambda_{ki}, \mathbf{h}_{ki}, \mathbf{e}_{ki})$ can be obtained from the list L_s, and returns the signature $\mathbf{e}_{ki}$ to $\mathcal{A}$. Otherwise, by checking the list L_1, $\mathcal{C}$ gets (τ_k, β_{ki}). If $\mathcal{C}$ doesn't get this value, then $\mathcal{C}$ will generate it according to the above, and saves list. Then, $\mathcal{C}$ computes vector $\lambda_{ki} = \beta_{ki} + \mathbf{Q}\mathbf{u}_{ki}$. $\mathcal{C}$ check list L_3, and gains tuple $(\tau_k, \{\mathbf{h}_{kj}, \alpha_{kj}\}_{j\in[n]}$. Let the vector $\mathbf{h}_{kj}$ is the *j-th* column of $\mathbf{H} \in \mathbb{Z}_q^{m\times n}$, and

$$\begin{aligned}\mathbf{B}\mathbf{e}_{ki} =& \mathbf{B}\mathbf{H}\lambda_{ki} = \mathbf{B}(\mathbf{h}_{k1}, \ldots, \mathbf{h}_{kn})\lambda_{ki} \\ =& (\alpha_{k1}, \ldots, \alpha_{kn})\lambda_{ki} = \mathbf{h}_{ki}.\end{aligned}$$

Then, $\mathcal{C}$ computes $\mathbf{e}_{ki} = \mathbf{H}\lambda_{ki}(\mod q)$, saves $(id, \tau_k, \mathbf{u}_{ki}, \lambda_{ki}, \mathbf{h}_{ki}, \mathbf{e}_{ki})$ and sends $(id, \tau_k, \mathbf{e}_{ki})$ as signatures to $\mathcal{A}$.

ProofGen: Now, TPA plays the role of a adversary and CSP plays the role of challenger in the protocol. The TPA defines the subset of set $[1, l]$ to be $I = \{s_j\}(1 \le j \le \theta)$ and $s_1 \le \ldots \le s_\theta$. For each element $i \in I$, the TPA chooses a random $c_i \leftarrow \mathbb{Z}_q^*$, and generates $chal = \{\tau, i, c_i\}_{i\in I}$ as the challenge. Then, the TPA forward $chal$ to the CSP. The CSP gains this and computes $\mathbf{u}_c = \sum_{i=s_1}^{s_\theta} c_i\mathbf{u}_i$. The adversary selects random vector w, runs $SamplePre(\mathbf{Q}, \mathbf{T}_Q, \mathbf{w}, \sigma) \to \xi$, checks list L_2 to gain corresponding to the value r, and computes $\mathbf{u}_c^{'} = \mathbf{u}_c + r\xi$ and $\mathbf{e}_c = \sum_{i=s_1}^{s_\theta} c_i\mathbf{e}_i$. Finally, the CSP forwards $proof = (\tau, \mathbf{u}_c^{'}, \mathbf{e}_c, \mathbf{w})$ to TPA.

Output: In this phase, the CSP forges $proof$, namely $proof^*$.

If CSP forges $proof$ that can pass the verification, we can construct an algorithm that can solve the $SIS_{q,m,n,\beta}$ problem. It is assumed that there are two types of attacks, namely loss attacks and tamper attacks in the cloud system.

1) Suppose there are loss attacks in the cloud storage system. The CSP losses the user's data $\mathbf{u}_k$, and tries to keep the truth from auditing from TPA. So, the CSP forges $proof^* = (\tau, \mathbf{u}_c^* = \sum_{i=s_1, i\ne k}^{s_\theta} c_i\mathbf{u}_i, \mathbf{u}_c^{'*}, \mathbf{e}_c, \mathbf{w})$. If the TPA receives the $proof^*$, the following equation holds.

$$\begin{aligned}\mathbf{B}\sum_{i=s_1}^{s_\theta} c_i\mathbf{e}_i =& \mathbf{h}_c^{'} \ mod\ q = \mathbf{C}\lambda_c^{'} \\ =& \mathbf{C}(\sum_{i=s_{1,i\ne k}}^{s_\theta} c_i H_1(\tau||i) + \mathbf{Q}\sum_{i=s_{1,i\ne k}}^{s_\theta} c_i\mathbf{u}_i)\end{aligned}$$

Through the verification equation, the following equation holds.

$$\begin{aligned}\mathbf{B}\sum_{i=s_1}^{s_\theta} c_i\mathbf{e}_i =& \mathbf{h}_c^{'} \ mod\ q = \mathbf{C}\lambda_c^{'} \\ =& \mathbf{C}(\sum_{i=s_1}^{s_\theta} c_i H_1(\tau||i) + \mathbf{Q}\sum_{i=s_1}^{s_\theta} c_i\mathbf{u}_i)\end{aligned}$$

Due to the above two equations are contradictory, the loss attacks can be detected by the $CheckProof$ phase.

2) Suppose there are tamper attacks in the cloud storage system. The CSP tampers the user's data $\mathbf{u}_k$ to $\mathbf{u}_k^*$, and tries to keep the truth from auditing from TPA. So, the CSP forges $proof^* = (\tau, \mathbf{u}_c^* = \sum_{i=s_1, i\ne k}^{s_\theta} c_i\mathbf{u}_i + c_k\mathbf{u}_k^*, \mathbf{u}_c^{'*}, \{\mathbf{e}_i\}_{i=s_{1,i\ne k}}^{s_\theta}, \mathbf{e}_k^*, \mathbf{w})$. Then, the following equation holds. Let $\Theta = \sum_{i=s_1}^{s_\theta} c_i H_1(\tau||i)$, we have

$$\begin{aligned}\mathbf{B}(\sum_{i=s_{1,i\ne k}}^{s_\theta} c_i\mathbf{e}_i + \mathbf{B}c_k\mathbf{e}_k^* =& \mathbf{h}_c^{'*} = \mathbf{C}\lambda_c^{'*} \\ =& \mathbf{C}\Theta + \mathbf{Q}\sum_{i=s_{1,i\ne k}}^{s_\theta} c_i\mathbf{u}_i + \mathbf{Q}c_k\mathbf{u}_k^*\end{aligned}$$

Through the verification equation, the following equation holds.

$$\mathbf{B}\sum_{i=s_{1,i\neq k}}^{s_\theta} c_i\mathbf{e}_i = \mathbf{C}\sum_{i=s_{1,i\neq k}}^{s_\theta}(c_iH_1(\tau||i)+\mathbf{Q}c_i\mathbf{u}_i)$$

If the above two equation are true, there is a way to compute the forged signature $\mathbf{e}_k^*$ of the message $\mathbf{u}_k^*$, which satisfied $\mathbf{Be}_k^* = H_1(\tau||k) + \mathbf{CQu}_k^*$. Without the trapdoor basis $\mathbf{T}_{id}$ and $\mathbf{T}_Q$, it is inconsistent with the GPV signature and the $SIS_{q,m,n,\beta}$ assumption.

So, if there are two attacks, the attacks be able to be recognised by the TPA in the $CheckProof$ phase.

C. Data Privacy Preserving

The data privacy preserving is that no blocks could be retrieved by the TPA during verification phase. The TPA receives the $proof = (\tau, \mathbf{u}_c^{'}, \mathbf{e}_c, \mathbf{w})$ from the CSP, it is impossible to recover any block in $M = (\mathbf{u}_1, \cdots, \mathbf{u}_l)$. Suppose the TPA are curiosity attacks in the cloud s-torage server. The TPA tries to recover any block in $M = (\mathbf{u}_1, \cdots, \mathbf{u}_l)$. There are combined message $\mathbf{u}_c = \sum_{i=s_1}^{s_\theta} c_i\mathbf{u}_i$, chooses a random vector $\mathbf{w} \leftarrow \mathbb{Z}_q^n$, runs the algorithm $SamplePre(\mathbf{Q}, \mathbf{T}_Q, \mathbf{w}, \sigma)$ to generate the signature ξ, and compute $\mathbf{u}_c^{'} = \mathbf{u}_c + \xi H_2(\mathbf{w})$, and $\mathbf{e}_c = \sum_{i=s_1}^{s_\theta} c_i\mathbf{e}_i$. By the GPV signature scheme and $SIS_{q,m,n,\beta}$ assumption, namely the TPA don't know the private key corresponding to the cloud server provider, the TPA couldn't recover the file's blocks.

So, if there are curiosity attacks from TPA, it is impossible to recover any block in $M = (\mathbf{u}_1, \cdots, \mathbf{u}_l)$ only with $proof = (\tau, \mathbf{u}_c^{'}, \mathbf{e}_c, \mathbf{w})$.

D. Probabilistic Checking

The proposed ID-Based remote data integrity checking protocol uses spot checking technique [2]. The CSP has n block-sign pairs which are stored in the cloud user. The total of t blocks are damaged, such that modifications and losses. The verifier(TPA) checks the integrity of the whole file containing n blocks by randomly sampling(challenged) c different blocks. Let X be a discrete random variable that is defined to be the number of challenged blocks that match the blocks damaged, P_X denotes the probability that at least one block chosen by the TPA matches one of the blocks damaged by the cloud server. Clearly we have:

$$\begin{aligned} P_X =& Pr[X \geqslant 1]1 - Pr[X = 0] \\ =& 1 - \frac{n-t}{n} \times \frac{n-1-t}{n-1} \times \ldots \times \frac{n-c+1-t}{n-c+1}. \end{aligned}$$

Thus, we can get the following inequality.

$$1 - (\frac{n}{n-c})^c \leqslant P_X \leqslant (\frac{n-c+1-t}{n-c+1})^c.$$

E. Performance Evaluation

Now we compare the assumption, the IBS scheme, the security model, the $chal$ length, the $proof$ length, the computation cost and the communication cost with some other data integrity checking schemes. The proposed scheme is ID-Based remote data integrity checking for cloud storage from lattices. It can resist a quantum computer attacks and eliminate the certificate management. The computation is fairly effective, including matrix-matrix or matrix-vector multiplications. The communication between TPA and CSP is also effective, namely the $chal$ length and the $proof$ length are acceptable. The comparison summarize the result in Table 1.

$Notes$: The psf denotes the cost of running $SamplePre$ algorithm, l denotes the number of block of file, c denotes the total number of challenged blocks, C_{exp} denotes the time cost of single exponentiation on the group, C_h denotes the time cost of single hash operation, C_e denotes the time cost of single bilinear pairing, and C_{mm} denotes the time cost of product of two matrices operation or product of two vectors operation.

V. Conclusion

In this paper, we propose a concrete ID-Based remote data integrity checking protocol, which can resist quantum computer attacks. Without the implementation of PKI, ID-Based remote data integrity checking eliminates the resource consuming certificate management. Meanwhile, This protocol is completeness, provably security against the CSP attacks and the data privacy against the curiosity TPA attacks. Therefore the proposed protocol is efficient and especially suitable for data owners with limited storage space and computation resources.

Acknowledgment

The deepest gratitude goes to the anonymous reviewers for their careful work and thoughtful suggestions that have helped improve this paper substantially. At the same time, our also would like to thank the reviewers' diligent and careful comments of our paper which make our work more complete.

References

[1] Armbrust, M., Fox, A., Griffith, R: A view of cloud computing. Commun. ACM 53(4), 50C58 (2010).

[2] Ateniese, G., Burns, R., Curtmola, R.: Provable data possession at untrusted stores. In: Proceedings of CCS, pp. 598C609 (2007).

[3] Curtmola, R., Khan, O., Burns, R., Ateniese, G.: MR-PDP: multiple-replica provable data possession. In: Proceedings of ICDCS, pp. 411C420 (2008).

[4] H. Shacham and B. Waters, "Compact proofs of retrievability", in Proc. Adv. Cryptol. ASIACRYPT, 2008, pp. 90C107.

Table I
COMPARISON OF SEVERAL SCHEMES

Property	Our scheme	Liu et al[18]	Wang et al[22]	Yu et al[12]
IBS	Yes	No	Yes	Yes
Assumption	SIS	SIS	CDH	CDH
Security Model	strong	no	weak	strong
Chal Length	$(c+m)\log q$	$(m+c)\log q$	$c(\log n+\log q)$	$(3+c)\log q$
Proof Length	$(3+c)\log q$	$(2+c)m\log q$	$2c\log q$	$l+2\log q$
SignGen/TagGen	$nC_h+nC_{mm}+lpsf$	$nC_h+lnC_{mm}+lpsf$	$n(l+1)C_{exp}+nlC_h$	$3lC_{exp}+lC_h$
Prove/ProofGen	$psf+C_h$	C_h+C_{mm}	$cC_{exp}+clC_h$	$2C_h+2C_e+cC_{exp})$
Verify/ProofCheck	$(n+c)C_h+3C_{mm}$	nC_h+nlC_{mm}	$2nC_e+(c+nl)C_{exp}$	$3cC_2+cC_h$

[5] Ateniese, G., Pietro, R.D., Mancini, L.V., Tsudik, G.: Scalable and efficient provable data possession. In: Proceedings of SecureComm, pp. 1C10 (2008).

[6] Erway, C.C., K up? c u, A., Papamanthou, C., Tamassia, R.: Dynamic provable data possession. In: Proceedings of CCS, pp. 213C222 (2009).

[7] Q. Wang, C. Wang, K. Ren, W. Lou, and J. Li, "Enabling public auditability and data dynamics for storage security in cloud computing", IEEE Trans. Parallel Distrib. Syst., vol. 22, no. 5, pp. 847C859, May 2011.

[8] C. Liu, R. Ranjan, C. Yang, X. Zhang, L. Wang, and J. Chen, MuR-DPA: Top-down levelled multi-replica merkle hash tree based secure public auditing for dynamic big data storage on cloud, IEEE Trans. Comput., vol. 64, no. 9, pp.

[9] Yu, Y., Ni, J., Au, M.H.: Comments on a public auditing mechanism for shared cloud data service. IEEE Trans. Serv. Comput. 8(6), 998C999(2015).

[10] Yu, Y., Li, Y., Ni, J., Yang, G, Mu.: Comments on "public integrity auditing for dynamic data sharing with multiuser modification". IEEE Trans. Inf. Forensics Secur. 11(3), 658C659 (2016).

[11] C. Wang, Q. Wang, K. Ren, and W. Lou, "Privacy-preserving public auditing for data storage security in cloud computing", in Proc. IEEE INFOCOM, Mar. 2010, pp. 1-9.

[12] Yong Yu, Man Ho Au, Giuseppe Ateniese, Identity-Based Remote Data Integrity Checking With Perfect Data Privacy Preserving for Cloud Storage.IEEE transactions on information forensics and security, vol.12, No.4,appil 2017.

[13] C. Gentry, C. Peikert, V. Vaikuntanathan, "Trapdoors for hard lattices and new cryptographic constructions", Proceedings of the 40th annual ACM Symposium on Theory of Computing (S-TOC 2008), Victoria, British Columbia, Canada, pp.197C206, 2008.

[14] D. Boneh, D.M. Freeman, Linearly homomorphic signatures over binary fields and new tools for lattice-based signatures, Proceedings of PKC 2011, LNCS 6571, Berlin: Springer-Verlag, pp.1C16, 2011.

[15] D. Boneh, D.M. Freeman, Homomorphic signatures for polynomial functions, Proceedings of EUROCRYPT 2011, LNCS 6632, Berlin: Springer-Verlag, pp.149C168, 2011.

[16] F. Wang, Y. Hu, B. Wang, Lattice-based linearly homomorphic signature scheme over binary field, Science China Information Sciences, Vol.55, pp.1C9, 2012.

[17] W. Xu, D. Feng, J. Liu, Public verifiable proof of storage protocol from lattice assumption, 2012 IEEE International Conference on Intelligent Control, Automatic Detection and High-End Equipment, Beijing, China, pp.133-137, 2012.

[18] H. Liu and W. Cao, Public proof of cloud storage from lattice assumption, Chinese Journal of Electronics, Vol.23, No.1, pp.186C190, 2014.

[19] H.Zang Xiaojun, Xu Chunxiang, Zhagn Yuan, Insecurity of a Public proof of cloud storage from lattice assumption, Chinese Journal of Electronics, Vol.26, No.1, pp.88C92, 2017.

[20] Shamir, A.: Identity-based cryptosystems and signature schemes. In: Blakely, G.R.,Chaum, D. (eds.) CRYPTO 1984. LNCS, vol. 196, pp. 47-53. Springer, Heidelberg (1985).

[21] J. Zhao, C. Xu, F. Li, and W. Zhang, Identity-based public verification with privacy-preserving for data storage security in cloud computing, IEICE Trans. Fundam. Electron., Commun. Comput. Sci., vol. E96-A, pp. 2709C2716, Dec. 2013.

[22] Wang, H.: Identity-based distributed provable data possession in multicloud storage. IEEE Trans. Serv. Comput. 8(2), 328C340 (2015).

[23] Fucai Zhou1, Su Peng, Jian Xu, and Zifeng Xu.: Identity-Based Batch Provable Data Possession. ProvSec 2016, LNCS 10005, pp. 112C129, 2016.

[24] Zhiguang Qin, Shikun Wu, and Hu Xiong.: Strongly Secure and Cost-Effective Certificateless Proxy Re-encryption Scheme for Data Sharing in Cloud Computing. BigCom 2015, LNCS 9196, pp. 205C216, 2015.