Neural Computing and Applications (2019) 31:1513–1519
https://doi.org/10.1007/s00521-018-3546-6

S.I. : EMERGING INTELLIGENT ALGORITHMS FOR EDGE-OF-THINGS COMPUTING

Certificateless remote data integrity checking using lattices in cloud storage

C. Sasikala[1] · C. Shoba Bindu[2]

Received: 28 March 2018 / Accepted: 11 May 2018 / Published online: 6 June 2018

Abstract
Due to the existence of many security threats in the cloud, remote data integrity checking (RDIC) is crucial for implementing secure cloud storage. It allows the data users to check the integrity of their data without retrieving whole data. As a solution to this, so far many researchers designed RDIC protocols using public key infrastructure (PKI). In this scenario, a public verifier must approve the data users' certificates before the integrity checking task is to be performed. Thus, it suffers from the complex key management problem to approve the certificates, and it also incurs high computation cost over the public verifier. To solve the certificate management issue in PKI-based protocols and to provide security against quantum computer attacks, in this work, we design a Certificateless RDIC protocol using lattices. In this approach, the data integrity checking can be initiated using data owner's identity (his name or email address) along with some secret information, which can guarantee the right public key is used for RDIC. Therefore, we can avoid the certificate management issue in PKI-based protocols to verify the data integrity. Finally, our security analysis guarantees that our Certificateless RDIC protocol is secure and it provides the privacy against the verifier, and performance analysis guarantees that it makes the less computation overhead over the public verifier.

Keywords Remote data integrity checking · Certificateless signatures · Third-party auditor (TPA) · Public key infrastructure · ID-based signatures

1 Introduction

One of the important cloud computing services is cloud storage. It allows the data users to deploy their data on the remote servers and enjoy on-demand services from a cloud, without the obstruction of local storage management and maintenance. By doing so, the user cannot have his data locally. Even though, cloud resources are very powerful and reliable than that of users own resources, the data on the cloud are still vulnerable to many threats either from inside or outside due to loss of control over the data, lack of trust on cloud service provider (CSP) and multi-tenancy. For monetary reasons, untrusted CSPs might reclaim storage space by removing the data that is not used or rarely accessed, and even to maintain reputation, they may disguise data loss incidents. In addition to this, server failures, power outages and security attacks of popular cloud services come out from time to time [1, 2]. These threats may compromise confidentiality, integrity, and availability of data. Hence, designing a proper RDIC protocol is an essential need in a cloud environment. To address this problem, many RDIC protocols have been designed using PKI [3–9]. In this protocol, the third-party auditor (TPA) must approve the certificate of data users before performing the integrity checking to know the authenticity of a user's public key. But the main issue with PKI is, the cloud user is responsible for maintaining its public key certificate (generation, storage, update, and revocation) and it also brings a burden on TPA regarding computation and communication cost. Thus, this type of RDIC protocols has become a serious problem for the source constrained cloud users. In the further extension of this work, researchers designed identity-based RDIC protocols [10–14] using

✉ C. Sasikala
sasikalareddy27@gmail.com

C. Shoba Bindu
shobabindhu.cse@jntua.ac.in

[1] Department of CSE, JNTUA, Anantapur, AP, India

[2] Department of CSE, JNTUA College of Engineering, Anantapur, AP, India

identity-based signatures (IBS) to avoid certificate management problem in PKI-based RDIC protocols. Unfortunately, the main flaw of IBS is the key escrow problem due to its complete dependency on private key generator (PKG) to create the private keys.

By using Certificateless signatures (CLS) [15, 16] instead of IBS, TPA can verify the integrity of the deployed data without suffering from complicated certificate management problem in PKI and key escrow issue in IBS. In CLS, private keys are created by combining the partial private key produced by KGC along with users secrete information, and the public key itself is the data owner's identity such as her email address or name. Therefore, certificates are not required to handle the owner's proper public key, which is used to initiate data integrity checking task in the cloud.

Our contributions We designed public verifiable RDIC protocol based on Certificateless signatures using lattices. Which can eliminate the complex certificate management problem in PKI and resist quantum computer attacks. We proved that it is more secure against malicious cloud server attacks and also it provides privacy against TPA based on Small Integer Solution (SIS) hard problem assumption in lattices. The analysis of our protocol shows that Certificateless RDIC protocol is secure and efficient.

The paper organization is as follows: We explain the system and security models of Certificateless RDIC protocol in Sect. 2, Sect. 3, describes preliminaries about lattices. In Sect. 4, we discuss the Certificateless RDIC protocol. In Sect. 5, we explain the security analysis of our protocol. In Sect. 6, we discuss the performance analysis of the Certificateless RDIC protocol.

2 System and security models

In this section, we have defined our Certificateless RDIC protocol system model and security model.

2.1 System model

The system model of Certificateless RDIC Protocol in the cloud is shown in Fig. 1. It includes the four network entities: The user, the cloud server, third-party auditor (TPA) and private key generator (PKG). (i) User: who deploy their data in the cloud, it may be an enterprise or an individual. (ii) Cloud Server: It provides significant storage space for the client to store their data in the cloud. (iii) TPA: Who has the expertise and the capabilities that users do not have, and it acts on behalf of the client on request. (iv) PKG: who generates the partial private key of a user based on her identity information?

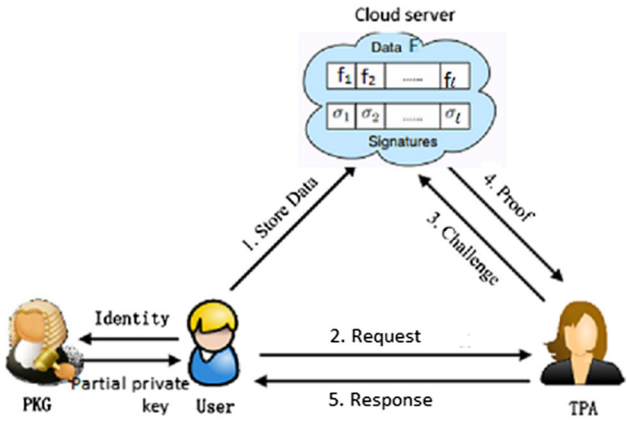

Fig. 1 System model for Certificateless data integrity checking protocol

2.2 Security model

In our cloud storage system, we assume that TPA is honest but curious. Even though it works honestly during the overall integrity checking process, but it is having the interest to disclose some information about the received data. And, we assume that the CSP is untrusted. So, in this work, we considered two security requirements that could affect the cloud data integrity checking, such as privacy against TPA and security against untrusted CSP. We define the security against untrusted CSP based on Proof of Retrievability model [6]. This definition includes a challenger; it represents either cloud user or TPA and an adversary represents the untrusted cloud server.

2.2.1 Security against untrusted CSP

The proposed RDIC protocol is secure against untrusted CSP, if and only if the probability of adversary winning the game is negligible. The communication between the challenger and adversary is defined as follows:

Step 1 The challenger executes the setup algorithm to generate public parameters and the master secret key msk, and then, he sends the public parameters to the adversary by keeping the msk as secrete.
Step 2 The adversary repeatedly makes some queries, such as hash query, sign query and key extract query to get the authenticated tags from the challenger. The challenge calculates the authenticated tag for every data block and forwards it to the adversary.
Step 3 Challenger enumerates the challenge message for the data blocks in the file and forwards it to the adversary.
Step 4 With the challenge, adversary calculates the proof as a response to the requested data blocks in the challenge.
Step 5 Then, the challenger verifies the proof, if it is valid means the adversary won the game.

2.2.2 Privacy against TPA

It indicates that there is no information about user's data is leaked to the TPA during the protocol verification process.

3 Preliminaries

An m-dimensional lattice L [17] in Euclidean space R^m is the set of integral combinations of n linearly independent vectors $b_1, \ldots, b_n$. The lattice L is represented by

$$L(b_1, \ldots, b_n) = \left\{ \sum_{i=1}^{n} x_i b_i, x_i \in Z \right\} \quad (1)$$

where $B = (b_1, \ldots, b_n)$ are the basis for the lattice, integer n is some vectors in a basis and m is the dimensions of the lattice. Let $A \in Z_q^{n \times m}$ and n and q are two positive integers, then q-array lattices is defined as follows:

$$L^{\perp}(A) = \{v \in Z^m : Av = 0 \mod q\} \quad (2)$$

$$L(A^T) = \{v \in Z^m : \exists\, s \in Z_q^n, \text{ such that } v = A^T s \mod q\} \quad (3)$$

For any vector $u \in Z_q^n$, an integral solution to $AX = u \bmod q$ defines the shifted lattice or coset of the lattice.

$$L_u^{\perp}(A) = \{v \in Z^m = Av = u \mod q\} = L^{\perp}(A) + X \quad (4)$$

3.1 Discrete Gaussians on lattices

For any vector $c \in R^m$, and positive $s > 0$ then the Gaussian function on R^m centered at c with parameter s is defined as:

$$R^m \forall x \in R^m,\ \rho_{s,c}(X) = \exp(-\pi \|x - c\| / s^2) \quad (5)$$

The discrete Gaussian distribution over m-dimensional lattice L is defined as:

$$\forall x \in L, D_{L,s,c}(X) = \frac{\rho_{s,c}(X)}{\rho_{s,c}(L)}, \quad \text{where } \rho_{s,c}(L) = \sum_{x \in A} \rho_{s,c}(X) \quad (6)$$

3.2 Trapdoor and sample basis functions

Some one-way trapdoors and Sample basis functions which are used in our signature generation algorithm is described as follows.

TrapGen (n, m, q) (Lemma 3.1 in [18]) For integers n, m, q with $q \geq 2$ and $m \geq 5n\lg q$, it generates a matrix $A \in Z_q^{n \times m}$ and T_A as the short basic of the lattice $L_q^{\perp}(A)$ and $\left\|T_A^{\sim}\right\| \leq m \cdot \omega \cdot \sqrt{\log m}$.

SampleBasis (Theorem 3.3 in [18]) Given the integers n, m, q with $q \geq 2$ and $m \geq 5n\lg q$ and on input of $A = (A_1, A_2, \ldots A_k) \in Z_q^{n \times km}$, on the set $S \subseteq [k]$, a trapdoor basis T_s of $L_q^{\perp}(A_S)$ and an integer $I \geq \left\|T_S^{\sim}\right\| \cdot \sqrt{km} \cdot \omega \cdot \sqrt{\log km}$, by taking these parameters as input SampleBasis (A, T_s, S, I) generates a matrix B as a basis for the lattice $L_q^{\perp}(A)$ with $\|B^{\sim}\| \leq I$.

SamplePre (Lemma 3.2 in [18]) For integers n, m, q with $q \geq 2$ and $m \geq 5n\lg q$, on the input of a matrix $A \in Z_q^{n \times m}$ and trapdoor basis T_A of the lattice $L_q^{\perp}(A)$, a vector $v \in Z_q^n$ and an integer $i \geq \left\|T_A^{\sim}\right\| \cdot \omega \cdot \sqrt{\log m}$, the PPT SamplePre (A, T_A, v, i) generates a vector y such that distribution of y is within the negligible statistical distance of $D\, L_q^v(A), i$.

3.3 SIS hard problem in lattices

Small integer Solution (SIS) problem [17] is described as, for an integer q, real β and a matrix $A \in Z_q^{n \times m}$, to get a nonzero vector $v \in Z^m$ such that $Av = 0 \pmod q$ where $\|v\| \leq \beta$ is hard.

4 Certificateless remote data integrity checking protocol

The proposed Certificateless remote data integrity checking protocol for cloud storage based on lattices works as follows: It includes the following algorithms:{Setup, Extract-Partial-Privatekey, Set-Secretvalue, Set-privatekey, Set-publickey, SignGen, Challenge, ProofGen, ProofCheck}. In the Setup phase, we use TrapGen function to get the master secret key msk. In the Extract-partial-PrivateKey algorithm, we adopt SampleBasis algorithm to get the partial private key corresponding the user $ID \in \{0, 1\}^*$. In set-secret value phase, the user randomly selects one matrix as his secrete value. By using Setsecretkey algorithm, the user gets his full private key by combining partial private key with secrete value. In the Signgen phase, we generate the signatures of the blocks by using SamplePre algorithm. In the challenge phase, the TPA generates the challenge and forward it to the CSP. In the ProofGen, after receiving the challenge, the CSP enumerates the proof and sends it to the TPA. In proof check phase, the TPA checks the proof means whether it is valid proof or not. Flow diagram of our protocol is shown in Fig. 2, and the description of the protocol is as follows:

Let n be a security parameter, $m > cn \log q$ for a fixed constant $c > 0$ and $q \geq \delta\omega(\log n)$ be a large prime for $\delta = \text{poly}(n)$. Let $s = \Omega\sqrt{n \log q}$ be a Gaussian parameter.

Setup(n) Given security parameter n, first the PKG runs TrapGen(n, m, q) algorithm to get a matrix $A \in Z_q^{n \times m}$ along

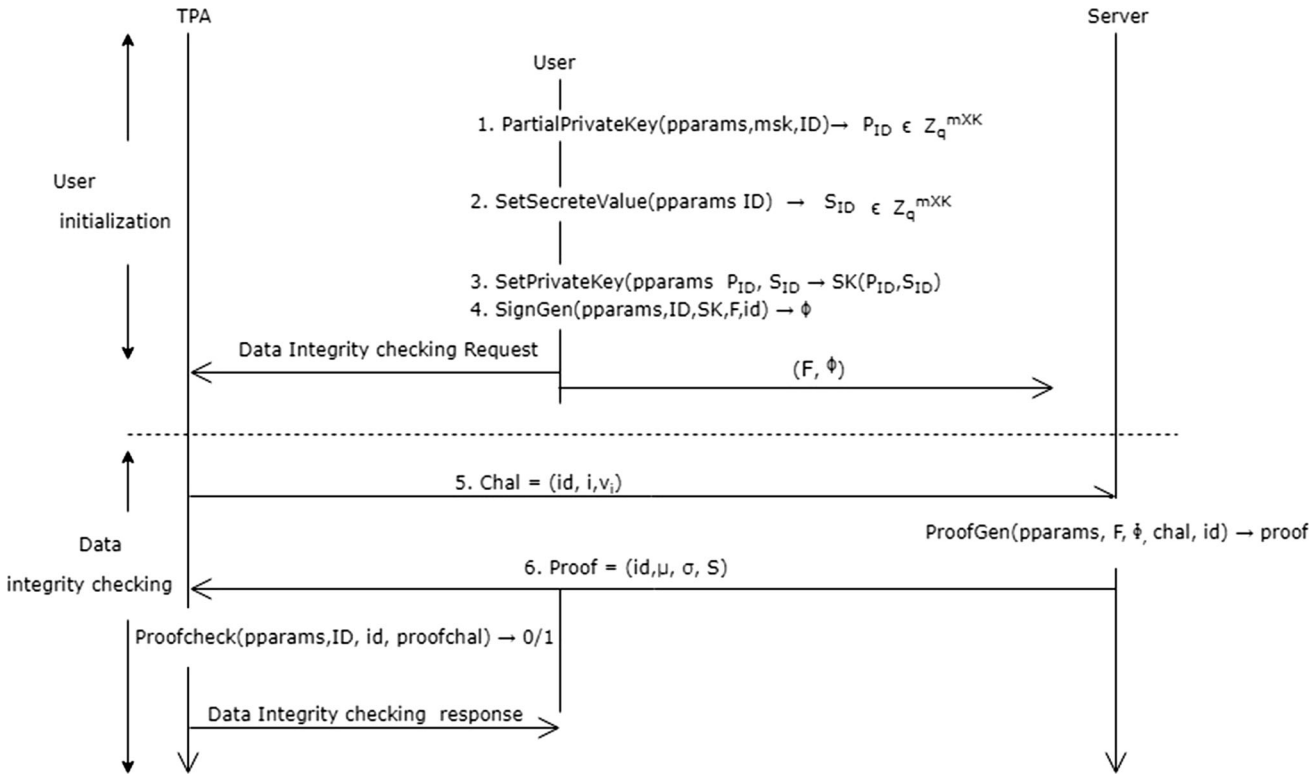

Fig. 2 Flow diagram of Certificateless RDIC protocol

with a short trapdoor basis $\mathbf{T}_A$ of the lattice $L_q^{\perp}(A)$. The cloud server also runs the TrapGen(n, m, q) to generate a matrix $B \in Z_q^{n\times m}$ with a short trapdoor basis T_B. Then, PKG chooses three secure hash functions defined as: $H_1 : \{0,1\}^* \rightarrow Z_q^m$, $H_2 : \{0,1\}^* \times Z_q^{n\times m} \rightarrow Z_q^m$ and keeps $\mathbf{T}_A$ as the master secrete key, i.e., msk = $\mathbf{T}_A$.

Extract-Partial-Privatekey (PParams, msk, ID) On input the public parameters PParams, the master secrete key and the user ID, PKG executes the SampleBasis (A, $\mathbf{T}_A$, s, H_1 (ID)) algorithm to get a matrix $P_{ID} \in Z_q^{m\times k}$ and send it to the user and the user sets his partial private key as $p_{ID} = P_{ID}$.

Set-Secrete-value (PParams, ID) Given PParams and user ID, the user randomly chooses a matrix $S_{ID} \in Z_q^{m\times k}$ (which must satisfy $\|S_{ID}\| \leq b$, where b is a positive integer) and set it as his secrete value $s_{ID} = S_{ID}$.

Set-Privatekey (PParams, p_{ID}, s_{ID}) It takes the public parameters and users partial privatekey and secrete value as the input and the user computes his full private key $sk = (P_{ID}, S_{ID})$.

Set-Public key(PParams, s_{ID}) Given the public parameters and user secrete value as input the user computes his public key $Pk = AS_{ID}$.

SignGen(PParams, ID, sk, F, id) To deploy the data file F in the cloud server, user divides F into l blocks $f_1, f_2, \ldots, f_k$, $f_2, \ldots, f_l$, where $f_i \in Z_q^m$ and id $\in \{0,1\}^*$ is the identity of the file F. Then, the user runs the SignGen algorithm to get the signature for each block of the file.

Algorithm 1: SignGen

Input: PParams, ID,sk,F, id

Output: σ_i

Step 1: Caluclates $\alpha_j \leftarrow H_1(\mathrm{ID}\|\mathrm{id}\|\ \mathrm{j}) \in Z_q^m$, where j ≤ n.

Step 2: For each f_i , 1≤ i ≤ l , the useComputes$\beta_i \leftarrow H_1$ (id ‖ i) ,where $\beta_i \in Z_q^{1\times m}$ and β_i is the row vector for i=1,2,….l .Let C =$(\alpha_1, \alpha_2, \ldots\ldots, \alpha_n)^{\mathrm{T}} \in Z_q^{m\times n}$

Step 3: Computes the inner products h_{ij}=<β_i, α_j>, for 1≤ i ≤ l and 1≤ j ≤ n. where h_i =(h_{i1}, h_{12},….,h_{in}). So h_i =β_i.C.

Step 4: To get the signature of each block user runs samplepreimage function.i.e. σ_i = Samplepre(A,sk,h_i,s).

Step 5: Let us assume that $\phi = \{\sigma_1, \sigma_2, \ldots\ldots, \sigma_l\}$, now the cloud user forwards the File F along with the signatures set ϕ to the cloud server and deletes the file from the local storage.

Challenge(id, F) The user forwards the auditing request to TPA, to verify the integrity of the file F. Then, TPA chooses a subset I of the set [1, l] to be $I = \{c_i\}$ where $1 \leq i \leq r$. For each $i \in I$, TPA randomly selects a value $v_i \in Z_q$ and computes the challenge message as chal = {id, i, v_i} where $i \in I$. Then, TPA sends the challenge message to the cloud server.

ProofGen (PParams, F, ϕ, chal, id) Upon receiving the challenge, the cloud server chooses the data blocks and corresponding signatures in it. Then, the cloud server runs the ProofGen algorithm to generate the proof for the challenged message.

Algorithm 2: ProofGen

Input: PParams,F, ϕ,chal,id

Output: proof=(id, μ, σ,S)

Step 1: First, the cloud server computes the aggregation of both data blocks and signatures as follows.

$$\mu^1 = \sum_{i=c_1}^{c_r} v_i f_i \bmod q \text{ ,where } \mu^1 \in Z_q^m$$

$$\sigma = \sum_{i=c_1}^{c_r} v_i \sigma_i \bmod q \text{ ,where } \sigma \in Z_q^m$$

Step 2: Then, cloud server chooses a random vector w$\in Z_q^m$ and verifies whether$\| w \| \leq \beta$ or not, if it does not holds again the cloud server chooses it until $\| w \| \leq \beta$.

Step 3: The cloud server computes S= Bw mod q and γ=H_2 (S).

Step 4: Then, the linear combination of μ^1 and w can be written as $\mu = \gamma \mu^1 + w \bmod q$.

Step 5: Finally, cloud server forwards the proof message as proof=(id, μ, σ,S) to the TPA.

Proofcheck (PParams, ID, id, Proof, Chal) Upon receiving the proof, TPA calculates $\gamma = H_2(S)$ and verifies whether the equation γ. A $\sigma + S = B$. $\mu \bmod q$ holds or not. If it holds, TPA accepts the proof; otherwise, proof is invalid.

5 Security analysis

In this section, we analyze the security of proposed protocol regarding the correctness, and we show that our protocol is secure against untrusted CSP and it preserves Privacy against TPA.

5.1 Correctness

If both cloud server and TPA runs the protocol honestly, then the proof of the cloud server must pass the verification successfully. The correctness of the protocol can be described as follows.

$$\begin{aligned}
\gamma \cdot A\sigma + S &= \gamma \cdot A \sum_{i=c_1}^{c_r} v_i f_i \\
&= \gamma \cdot \left(\sum_{i=c_1}^{c_r} v_i A \sigma_i \right) + S \\
&= \gamma \sum_{i=c_1}^{c_r} v_i B f_i + Bw \,, \bmod q \\
&= \gamma (B\mu^1) + Bw \,, \bmod q \\
&= B(\gamma \mu^1 + w) \\
&= B\mu \bmod q
\end{aligned}$$

5.2 Security against untrusted Cloud server

We prove that our protocol is secure against untrusted cloud server by assuming SIS hard problem in lattices.

Proof To compute the proof, we consider cloud server as adversary $\mathcal{A}$ and cloud user/TPA as challenger $\mathcal{C}$. Here, challenger $\mathcal{C}$ simulates the random oracles and give the appropriate answers to the hash queries issued by $\mathcal{A}$. If there exist a PPT $\mathcal{A}$ that wins the game described in Sect. 2.2, then we can illustrate the construction of $\mathcal{C}$ as follows.

Step 1 we consider the SIS instance of the challenger as $(A \in Z_q^{n \times m}, q, n, m, s)$ and the aim of the $\mathcal{C}$ is to find a non zero vector $v \in Z_q^m$ that must satisfy $Av = 0 (\bmod q)$ and $\|v\| \leq 2\sigma\sqrt{m}$.
Step 2 we assume that the signature σ_i of data f_i (i = 1,2,…l) are pubic to both adversary and challenger.
Step 3 after receiving the $H_1(\mathrm{id} \| i)$ query, $\mathcal{C}$ computes the ith row of matrix A. Then, $\mathcal{C}$ forwards the challenge message to $\mathcal{A}$.
Step 4 after receiving the $H_2(S)$ query, $\mathcal{C}$ computes γ and send it to $\mathcal{A}$. Then, c outputs (F, σ, S) such that

$$F = \gamma \cdot \sum_{i=c_1}^{c_r} v_i f_i + w \tag{7}$$

Step 5 we assume that $\mathcal{C}$ can interact with the $\mathcal{A}$ just before the query $H_2(S)$. At that time, $\mathcal{C}$ computes $H_2(S)$ as γ^*, where $\gamma \neq \gamma^*$. Then, $\mathcal{A}$ outputs (F^*, σ, S), such that

$$F^* = \gamma^* \cdot \sum_{i=c_1}^{c_r} v_i f_i + w \tag{8}$$

$\therefore$ From Eqs. (7) and (8) $\mathcal{C}$ computes the following:

$$\sum_{i=c_1}^{c_r} v_i f_i = (F - F^*)/(\gamma - \gamma^*) \tag{9}$$

Based on Eq. [9], $\mathcal{C}$ computes w as follows:

$$w = F - \gamma \cdot (F - F^*)/(\gamma - \gamma^*) \tag{10}$$

If the value of w satisfies $B\,w = S \bmod q$ and $\|w\| \leq \beta$, then w is the solution to the given SIS instance. So we conclude that, if there exist as an adversary that wins the security game, then we can construct the challenger to find the solution for the given SIS instance but our protocol is designed based on SIS assumption in lattices, so it is hard to find the value of w. Therefore, our protocol is secure against untrusted cloud server attacks.

5.3 Privacy against TPA

It refers to no data blocks could be accessed by TPA during verification phase. That is, the TPA tries to retrieve any block in F = ($F = (f_1, f_2, \ldots, f_l)$) by selecting a random value $r \in Z_q^n$ and computes $\mu = \mu = \gamma\mu^1 + r \bmod q$, but it fails because TPA knows aggregation of data blocks only. So, in our protocol, it is not possible to TPA to get any data block in F with the help of proof message.

6 Performance analysis

6.1 Computation cost

To evaluate the performance of our work, we used .NET Programming language & NTRU-CRYPTO library and tested on Green Cloud Open Source Simulator, using the system with 2.33 GHz Intel Core 2 Duo processor with 3 GB RAM and Ubuntu 14.04 OS. We assume that the file F of size 2 GB is divided into l = 1,00,000 blocks, $|n|$ = 20 bits and $|q|$ = 60 bits.

Our RDIC protocol involves only matrix–vector or matrix–matrix multiplications only, whereas existing protocols are designed using many pairing and exponential operations. In our protocol, computation cost of SignGen algorithm is 1100.00 ms, and when the challenge message is formed with 800 random blocks, the computation cost for the ProofGen algorithm is 2696.02 ms and the time is taken by the ProofVerify algorithm is 1157.06 ms. We compare the computation complexity of our protocol with Wang C et al. and Yong Yu et al. protocols that is shown in Table 1.

We graphically compare the computation overhead of our protocol with PKI-based Privacy-Preserving data integrity checking protocol designed by Wang et al. using pairing-based cryptography and ID-based data integrity checking protocol proposed by Yu et al. using pairing-based cryptography, shown in Fig. 3.

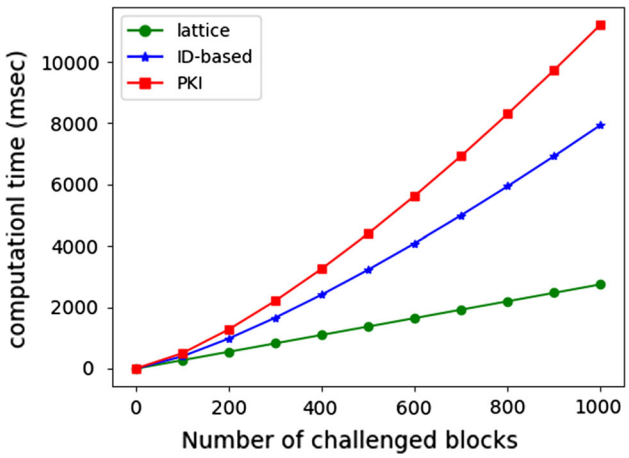

Fig. 3 Computation cost comparison among PKI versus ID-based versus lattice-based protocols

6.2 Communication cost

The data integrity checking task is initiated by TPA by sending challenge = $\{\mathrm{id}, i, v_i\}$ where $i \in I$ to the cloud server, then the server returns the response to the challenge as proof = $(\mathrm{id}, \mu, \sigma, S))$ to the TPA. Here, the communication cost for the challenge is $r(|q|)$ bits and in the proof message $\mu \in Z_q^m$, $\sigma \in Z_q^m$ and $S \in Z_q^m$ so the communication cost for proof message is 3 $|q|$ bits, where $|q|$ is the length of an element of Z_q^m. The communication cost of the proposed protocol is same as that of the previous protocols described in Table 1.

6.3 The probability of detecting corruption

In Fig. 4, if the number of corrupted blocks c = 70, then the challenge message is formed with 1000 random blocks, then the probability of detecting the corrupted blocks is greater than 95%. If a challenged message is formed with less number of random blocks, then the TPA completes the integrity checking task in a short amount of time, but the probability of detecting corrupted blocks decreases. For example, in Fig. 4, if c = 50, the probability of detecting the corrupted blocks is greater than 90% when challenge message is formed with 800 random blocks.

Table 1 Comparison of computation cost (in milliseconds)

Property	Wang et al. [4]	Yong et al. [14]	Our protocol
Signature type	PKI-based signatures	Identity-based signatures	Certificateless signatures
Assumption	CDH	CDH	SIS
SignGen	3256.72	2413.67	1100.00
ProofGen	2696.02	1892.56	798.23
ProofVerify	5713.67	3420.00	1157.06

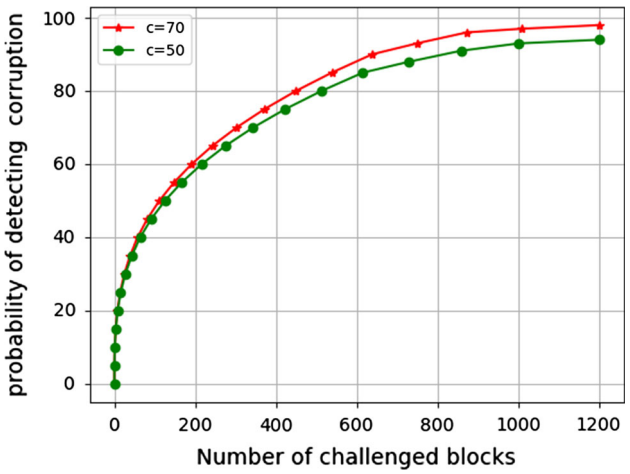

Fig. 4 The probability of detecting corrupted blocks

7 Conclusion

In this work, we have proposed a Certificateless RDIC protocol using lattices. This approach allows the TPA to verify the integrity of the data in the cloud by eliminating the certificate management problem associated with PKI in the existing solutions. By using SIS assumption in lattices, we also proved the security of our protocol and the experimental results guarantee that our protocol is efficient especially for source constrained devices. Even though our protocol works effectively on single cloud and single user environment, in future we would extend our work to support batch verification in multi-cloud and multi-user environment to increase the efficiency of the proposed protocol. The proposed protocol supports only static data verification; in future, it is possible to extend our work to support dynamic data operations also.

Compliance with ethical standards

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

References

1. Armbrust M, Fox A, Griffith R, Joseph AD, Katz RH, Konwinski A, Lee G, Patterson DA, Rabkin A, Stoica I, Zaharia M (2010) A view of cloud computing. Commun ACM 53(4):50–58
2. Krebs B (2009) Payment processor breach may be largest ever. http://voices.washingtonpost.com/securityfix/2009/01/payment processorbreachmayb.html
3. Ateniese G, Burns R, Curtmola R, Herring J, Kissner L, Peterson Z, Song D (2007) Provable data possession at untrusted stores. In: The Proceedings of ACM CCS 2007, pp 598–610
4. Wang C, Wang Q, Ren K, Lou WJ (2010) Privacy—preserving public auditing for data storage security in Cloud Computing, In: Proceedings IEEE INFOCOM, San Diego, pp 1–9
5. Ateniese G, Pietro RD, Mancini LV, Sudik T (2008) Scalable and efficient Provable data possession. In: Proceedings of the 4th international conference on security and privacy in communication networks. Istanbul, Turkey. ACM, pp 1–10
6. Shacham H, Waters B (2008) Compact proofs of retrievability. In: International conference on Advances in Cryptography-ASIACRYPT 2008. Springer, Berlin, pp. 90–107
7. Juels A, Kaliski BS (2007) Proofs of retrievability for large files. In: Proc. 14th ACM Conf. Computer and Communication Security (CCS'07), pp 584–597
8. Zhu Y, Wang H, Hu Z, Ahn GJ, Hu H, Yau SS (2011) Dynamic audit services for integrity verification of outsourced storage in clouds. In: the Proceedings of ACM SA, pp 1550–1557
9. Wang B, Li H, Li M (2013) Privacy-preserving public auditing for shared cloud data supporting group dynamics. In: the Proceedings of IEEE ICC 2013, pp 62–74
10. Zhao JN, Xu CX, Li FG, Zhang W (2013) Identity-based public verification with privacy-preserving for data storage security in cloud computing. IEICE Trans 96-A(12):2709–2716
11. Wang H (2015) Identity-based distributed provable data possession in multi cloud storage. IEEE Trans Serv Comput 8(2):328–340
12. Wang H, Wu Q, Qin B, Domingo-Ferrer J (2014) Identity-based remote data possession checking in public clouds. IET Inf Secur 8(2):114–121
13. Zhang J, Dong Q (2016) Efficient id-based public auditing for the outsourced data in cloud storage. Inf Sci 343:1–14
14. Yu Y, Au MH, Ateniese G (2017) Identity-based remote data integrity checking with perfect data privacy preserving for cloud storage. IEEE Tran Inf Forensics Secur 12(4):767–778
15. Al-Riyami S, Paterson KG (2003) Certificateless public key cryptography. In: the Proceedings of ASIACRYPT 2003. Springer, pp 452–473
16. Wang B, Li B, Li H, Fenghua (2013) Certificateless public auditing for data integrity in the cloud. In: 2013 IEEE Conference on Communications and Network Security, CNS 2013, pp 136–144. https://doi.org/10.1109/cns.2013.6682701
17. Gentry C, Peikert C, Vaikuntanathan V (2008) Trapdoors for hard lattices and new cryptographic constructions. In: Proceedings of the 40th annual ACM symposium on Theory of computing. ACM, pp 197–206
18. David C, Hofheinz D, Kiltz E (2009) How to delegate a lattice basis. J IACR Cryptol ePrint Arch 25:351–362