Analysis and Enhancement of a Lattice-Based Data Outsourcing Scheme With Public Integrity Verification

Qingxuan Wang, Chi Cheng, *Member, IEEE*, Rui Xu, Jintai Ding, and Zhe Liu

Abstract—Recently, Zhang *et al.* proposed a lattice-based data outsourcing scheme with public integrity verification (DOPIV), which enables an original data owner to delegate a proxy to generate the signatures of data and outsource them to the cloud server. They employed a third party auditor (TPA) to check the integrity of the outsourced data and any TPA can verify the data integrity efficiently. DOPIV is claimed to achieve proxy-oriented secure data outsourcing as well as storage correctness guarantee. Unfortunately, we find that there exist vulnerabilities in DOPIV which allow the cloud server to simply delete the received data without being noticed by the TPA. Fortunately, we come up with a simple and efficient solution to thwart the proposed attack. Our improved scheme maintains all the features claimed in DOPIV.

Index Terms—Quantum-safe, identity-based data outsourcing, integrity verification, cloud storage

1 Introduction

Massive data are produced on a daily basis due to the rapid development of information and communication techniques. In order to access the data flexibly, data owners tend to outsource the data to the cloud server [1]. Although cloud users are relieved from complicated local storage management and maintenance, data integrity is still an important security concern [2]. In fact, the cloud server cannot always be fully trusted. Moreover, the cloud server may delete data that are not frequently used without notifying the cloud users. Therefore, it is necessary for cloud users to check the data integrity regularly. But the large amount of data make this unrealistic. Under this circumstance, the method of public integrity verification is proposed. The cloud user entrusts a third-party auditor (TPA) to perform the data integrity verification. At the same time, in the semi-trusted model, the TPA could be curious about the user's data.

There are two challenges in the public integrity verification. First of all, existing public verification schemes base their security on the hardness of number theoretic problems. But recent breakthrough results of quantum computers have brought threats to these data outsourcing system. It is well known that once there exit large-scale quantum computers which can run Shor's algorithm [3], existing public verification schemes would be broken [4]. Besides, most of the public verification schemes resort to complex and expensive public key infrastructure (PKI).

Therefore, great importance should be attached to the design of quantum-resistant data outsourcing scheme with public integrity verification. Lattice-based cryptosystem is regarded as one of the most promising candidates for quantum-resistant cryptography, according to a recent effort on standardization of post-quantum cryptography made by the National Institute of Standards and Technology [5].

Recently, taking the advantage of the lattice-based cryptosystem, Zhang *et al.* proposed a quantum-safe identity-based data outsourcing scheme with public integrity verification in cloud storage (DOPIV) [6]. They claimed that DOPIV achieved proxy-oriented secure data outsourcing and storage correctness guarantee. Unfortunately, we find that both of the above targets are vulnerable. We propose two attacks, an inside attack and a proof forgery attack, respectively, to show the vulnerabilities. Specially, we show that the cloud sever can simply delete the received data without being noticed by the the third-party auditor (TPA). Along the way, we also discuss the causes of such vulnerabilities. Fortunately, we find a simple and efficient improvement to the original scheme which thwarts the proposed attacks while maintaining all the security goals claimed in the original scheme.

- *Qingxuan Wang is with the College of Cyber Science, Nankai University, Tianjin 300350, China, and also with the State Key Laboratory of Cryptology, Beijing 100878, China. E-mail: cug_wqx@foxmail.com.*
- *Chi Cheng is with the Hubei Key Laboratory of Intelligent Geo-Information Processing, School of Computer Science, China University of Geosciences, Wuhan, Hubei 430074, China, with the State Key Laboratory of Cryptology, Beijing 100878, China, and also with the Guangxi Key Laboratory of Trusted Software, Guilin University of Electronic Technology, Guilin, Guangxi 541004, China. E-mail: chengchi@cug.edu.cn.*
- *Rui Xu is with the Hubei Key Laboratory of Intelligent Geo-Information Processing, School of Computer Science, China University of Geosciences, Wuhan, Hubei 430074, China, with the State Key Laboratory of Cryptology, Beijing 100878, China, and also with the Guangxi Key Laboratory of Trusted Software, Guilin University of Electronic Technology, Guilin, Guangxi 541004, China. E-mail: ruixu.cug@qq.com.*
- *Jintai Ding is with the Yau Mathematical Sciences Center, Tsinghua University, Beijing 100084, China, and also with the Ding Lab, Beijing Institute of Mathematical Sciences and Applications, Beijing 100084, China. E-mail: jintai.ding@gmail.com.*
- *Zhe Liu is with the Nanjing University of Aeronautics and Astronautics, Nanjing 210016, China. E-mail: zhe.liu@nuaa.edu.cn.*

Manuscript received 21 Feb. 2020; revised 12 Oct. 2020; accepted 24 Nov. 2020. Date of publication 30 Nov. 2020; date of current version 8 Aug. 2022. (Corresponding authors: Chi Cheng and Rui Xu.) Digital Object Identifier no. 10.1109/TSC.2020.3041324

2 PRELIMINARIES

2.1 Basic Definition

Let $B = \{b_1, \ldots, b_m\} \in \mathbb{R}^{m\times m}$ be a basis of the lattice Λ, which consists of m linearly independent vectors $b_1, \ldots, b_m$. The lattice Λ is an m-dimension full-rank lattice generated by B. Precisely, $\Lambda = \mathcal{L}(B) = \{y \in \mathbb{R}^m : \exists x \in \mathbb{Z}^m, y = Bx \sum_{i=1}^{i=m} x_i b_i\}$. We denote by $\|B\|$ the Gram-Schmidt norm of B.

Given a matrix $A \in \mathbb{Z}_q^{n\times m}$ where q is a prime and a vector $y \in \mathbb{Z}_q^n$, we define three q-modular integer lattices as follows:

1) $\Lambda_q(A) = \{y \in \mathbb{Z}_q^m : \exists x \in \mathbb{Z}_q^n, y = A^\top x \bmod q\}$.
2) $\Lambda_q^\perp(A) = \{z \in \mathbb{Z}^m : Az = 0 \bmod q\}$.
3) $\Lambda_q^y(A) = \{z \in \mathbb{Z}^m : Az = y \bmod q\}$.

2.2 SIS Problem and ISIS Problem

- *SIS (Short Integer Solution) problem* [7]: Given a prime q, a matrix $A \in \mathbb{Z}_q^{n\times m}$, the goal of SIS problem is to find a nonzero integer vector $e \in \mathbb{Z}^m$ such that $Ae = 0 \bmod q$ and $\|e\| \le \zeta$.
- *ISIS (Inhomogeneous SIS) problem*: Given a prime q, a matrix $A \in \mathbb{Z}_q^{n\times m}$, a vector $y \in \mathbb{Z}_q^n$ and a positive real number ζ, the goal of ISIS problem is to find a nonzero integer vector $e \in \mathbb{Z}^m$ such that $Ae = y \bmod q$ and $\|e\| \le \zeta$.

For any sufficiently large $n \in \mathbb{N}$, with $m, q \in \mathbb{N}$ and $\zeta \in \mathbb{R}$ satisfying $q > \zeta \mathbf{poly}(n)$ (for any polynomial), the SIS problem and ISIS are regraded as hard.

3 REVIEW OF DOPIV

In this section, we briefly introduce the DOPIV scheme proposed by Zhang *et al*. The system model is consisted of five different entities: the original data owner, the proxy, the cloud server, TPA, and the key generation center (KGC).

- *The original data owner* is a system user who owns a lot of data that are needed to be stored in the cloud server.
- *The cloud server* is a service provider who has huge storage space and trades it for profit. But the cloud server is not trusted in the sense that it might not faithfully store a client's data for various reasons (eg., reducing cost, server malfunction).
- *The proxy* is a cooperator of the original data owner. The proxy will outsource the data to the cloud server on behalf of the original data owner, upon getting authorization from the later.
- *TPA* is a third party. In response to the data owner's request, he or she regularly asks the cloud server for data integrity proofs.
- *The KGC* is a trusted institution that generates keys for all participants involved in the system.

DOPIV is consisting of five phases: *Setup*, *KeyExtract*, *Proxy-oriented Signing KeyGen*, *Proxy-oriented TagGen* and *Auditing Outsourced Data*. Besides, there are three primitives employed in DOPIV as follows:

- **TrapGen**$(\mathbf{q}, \mathbf{n})$ [8]: Given (q, n) as input, this function generates $(A \in \mathbb{Z}_q^{n\times m}, T_A \in \mathbb{Z}_q^{m\times m})$, A is a matrix and T_A is a short lattice basis of $\Lambda_q^\perp(A)$ that $AT_A = 0 \,(\mathrm{mod}\ q)$.
- **SamplePre**$(\mathbf{A}, \mathbf{T_A}, \mathbf{y}, \sigma)$ [9]: Given a matrix $A \in \mathbb{Z}_q^{n\times m}$ and its corresponding short lattice basis T_A, a parameter $\sigma \geqslant \|\widetilde{T_A}\| \cdot \omega(\sqrt{\log m})$ and a vector $y \in \mathbb{Z}_q^n$. The output is a nonzero integer vector e such that $Ae = y \,(\mathrm{mod}\ q)$.
- **NewBasisDel**$(\mathbf{A}, \mathbf{R}, \mathbf{T_A}, \delta)$ [10]: Given a matrix $A \in \mathbb{Z}_q^{n\times m}$ and its corresponding short lattice basis T_A, a parameter $\delta \geqslant \|\widetilde{T_A}\| \cdot \delta_R(\sqrt{\log^{\frac{3}{2}} m})$, $\delta_R = \sqrt{n \log q} \cdot \omega(\sqrt{\log m})$ and a $\mathbb{Z}_q$-invertible matrix R. The output is a short lattice basis T_B of $\Lambda_q^\perp(Q)$, where $Q = AR^{-1}$.

The construction of DOPIV is as follows:

Setup. Taking security parameters κ, q as inputs, the system is initialized as follows:

1) The discrete Gaussian distribution χ is determined by the security Gaussian parameter δ, σ.
2) KGC chooses six hash functions: $H_1 : \{0,1\}^{\kappa_1} \to \mathbb{Z}_q^{m\times m}$, $H_2 : \{0,1\}^{\kappa_1} \times \{0,1\}^{\kappa_1} \times \{0,1\}^{\kappa_2} \times \mathbb{Z}_q^n \to \mathbb{Z}_q^n$, $H_3 : \{0,1\}^{\kappa_1} \times \{0,1\}^{\kappa_1} \times \{0,1\}^{\kappa_2} \times \mathbb{Z}_q^m \to \mathbb{Z}_q^{m\times m}$, $H_4 : \{0,1\}^* \times \{0,1\}^* \to \mathbb{Z}_q^n$, $H_5 : \{0,1\}^{\kappa_1} \times \{0,1\}^* \to \mathbb{Z}_q^n$, and $H_6 : \{0,1\}^{\kappa_2} \times \mathbb{Z}_q^n \times \mathbb{Z}_q^m \times \mathbb{Z}_q^n \to \times\mathbb{Z}_q$.
3) KGC runs $TrapGen(q, n)$ to generate his public-private key pair (A, T_A) such that $A \cdot T_A = 0$.

KeyExtract. KGC uses his public-private key pair to generate key pairs for the original data owner ($ID_o \in \{0,1\}^{\kappa_1}$) as follows:

1) KGC computes $R_{ID_o} = H_1(ID_o)$ and $Q_{ID_o} = A(R_{ID_o})^{-1} \in \mathbb{Z}_q^{n\times m}$.
2) Run $NewBasisDel(A, R_{ID_o}, T_A, \delta)$ to generate $T_{ID_o} \in \mathbb{Z}_q^{m\times m}$ which is a random short lattice basis of $\Lambda_q^\perp(Q_{ID_o})$ and kept by the ID_o as the private key.

   The public-private key pair of ID_o is (Q_{ID_o}, T_{ID_o}), similarly, KGC generates (Q_{ID_p}, T_{ID_p}) for the proxy, and (Q_{ID_c}, T_{ID_c}) for the cloud server, respectively.

Proxy-Oriented Signing KeyGen. In this phase, ID_p will generate proxy-oriented signing private key for himself as follows:

1) ID_o creates the warrant $w \in \{0,1\}^{\kappa_2}$ which contains the delegation policy, valid period of delegation, and the identities of the original data owner and the proxy.
2) ID_o randomly chooses a vector $v_w \leftarrow \mathbb{Z}_q^n$, computes $u_w = H_2(ID_o||ID_p||w||v_w)$, and $\theta_w = SamplePre\,(Q_{ID_o}, T_{ID_o}, u_w, \sigma)$. Then, ID_o sends (w, v_w, θ_w) directly to the proxy ID_p.
3) After receiving (w, v_w, θ_w), ID_p computes $u_w = H_2(ID_o||ID_p||w||v_w)$ and validates the signed warrant w by the equation $Q_{ID_o}\theta_w = u_w$. If the equation holds, ID_p computes $R_w = H_3(ID_o||ID_p||w||\theta_w)$ and generates the proxy-oriented private key $T_{pro} = NewBasisDel(Q_{ID_p}, R_w, T_{ID_p}, \delta)$ and the corresponding proxy-oriented signing public key $Q_{pro} = Q_{ID_p}(R_w)^{-1}$.

Proxy-Oriented TagGen. ID_p divides file F into l subfiles $F = \{F_1, F_2, \ldots, F_l\}$, where $F_i \in \mathbb{Z}_q^m$ and each F_i has its corresponding name $N_i \in \{0,1\}^*$. The proxy generates proxy-oriented signing private key and generates the signature of each subfile F_i:

1) ID_p computes $Q_{ID_c} = AR_{ID_c}^{-1}$ and $\eta_i = H_4(N_i||i) + Q_{ID_c}F_i$.
2) ID_p computes $\lambda_j = H_5(ID_p||j)$, where $1 \leqslant j \leqslant n$ and inner products $\rho_{i,j} = \langle \eta_i, \lambda_j \rangle$, then sets $\rho_i = (\rho_{i,1}, \ldots, \rho_{i,n})^\top \in \mathbb{Z}_q^n$.
3) ID_p computes $Q_{pro} = Q_{ID_p}R_w^{-1}$ to generate ρ_i's signature $e_i = SamplePre(Q_{pro}, T_{pro}, \rho_i, \sigma)$.

After that, ID_p outsources $(F_i, N_i, e_i), (1 \leqslant i \leqslant l)$ and (w, v_w, θ_w) to the cloud server. The cloud server will first compute $u_w = H_2(ID_o||ID_p||w||v_w)$ and check the ID_p's validity by the equation $Q_{ID_o}\theta_w = u_w$. If the equation holds, the cloud server stores $(w, v_w, \theta_w, F_i, N_i, e_i)$.

Auditing Outsourced Data. In this phase, TPA will verify the primitive data's integrity. There are three steps:

- *Challenge:* The TPA chooses a c-element subset $\Omega = \{l_1, \ldots, l_c\}$ randomly and a random binary string $\beta = (\beta_{l_1}, \ldots, \beta_{l_c})$. Then TPA sends the challenge message $chal = \{i, \beta_i\}_{i\in\Omega}$ to the cloud server.
- *ProofGen:* The cloud server computes $f' = \sum_{i=l_1}^{i=l_c} \beta_i F_i \in \mathbb{Z}_q^m$, $e = \sum_{i=l_1}^{i=l_c} \beta_i e_i \in \mathbb{Z}_q^m$, then, ID_c chooses a random vector $\xi \in \mathbb{Z}_q^n$ and comutes $h = SamplePre(Q_{ID_c}, T_{ID_C}, \xi, \sigma)$, $f = f' + hH_6(w||v_w||\theta_w||\xi)$. After that, ID_c sends $Proof = (f, e, \xi)$ and (w, v_w, θ_w) to the TPA.
- *ProofVerify:* TPA validates (w, v_w, θ_w) to check ID_p's validity. Next, TPA computes $\lambda_j = H_5(ID_p||j)$, $(1 \leqslant j \leqslant n)$, set $B = (\lambda_1, \ldots, \lambda_n)^\top$ and computes $\mu = B(\sum_{i=l_1}^{i=l_c} \beta_i H_4(N_i||i) + Q_{ID_c}f - \xi H_6(w||v_w||\theta_w||\xi))$, if the equation $Q_{pro}e = \mu \pmod q, (0 \leqslant \|e\| = \|\sum_{i=l_1}^{i=l_c} e_i\| \leqslant c\sigma\sqrt{m})$ holds, the primative data has not been modified.

4 Security Analysis of DOPIV

In this section we propose two attacks, an inside attack and a proof forgery attack, to show the vulnerabilities of Zhang *et al.*'s scheme.

4.1 Inside Attack

In the *Proxy-oriented Signing KeyGen* stage, there may be a malicious user with identity ID_a whose public-private key pair is (Q_{ID_a}, T_{ID_a}). In the beginning of this stage, ID_a could claim that he is the proxy ID_p and interact with the ID_o. ID_o creates the warrant $w \in \{0,1\}^{\kappa_2}$, which contains the identity ID_p. Then, ID_o computes $u_w = H_2(ID_o||ID_p||w||v_w)$ and $\theta_w = SamplePre(Q_{ID_o}, T_{ID_o}, u_w, \sigma)$ after that, ID_o sends (w, v_w, θ_w) to the ID_a.

Since ID_a is a fake proxy, when ID_a receives the (w, v_w, θ_w), he just skips the step of checking the validity of w and goes on to compute $R_w = H_3(ID_o||ID_p||w||\theta_w)$. Next, ID_a computes the private key $T'_{pro} = NewBasisDel(Q_{ID_a}, R_w, T_{ID_a}, \delta)$ and the corresponding public key $Q'_{pro} = Q_{ID_a}(R_w)^{-1}$.

The proxy-oriented signing keyGen process seemed to run smoothly, despite there is a malicious user ID_a take part in. The reason is that this scheme is short of identity authentication. Besides, there is no difference among the parties involved. That is the parities ID_o, ID_p and ID_c only have the similar public-private key pairs (Q, T). Obviously, the proxy needs more to prove his validity.

Next, we show that there exist another attack which can bypass the verification of TPA.

4.2 Proof Forgery Attack

In the *Auditing Outsourced Data* stage, TPA sends challenge message $chal = \{i, \beta_i\}_{i\in\Omega}$ to the cloud server. After receiving the message, the cloud server computes proof information of storage correctness $Proof = (f, e, \xi)$, where $f = f' + hH_6(w||v_w||\theta_w||\xi)$. Then, the cloud server will send the $Proof = (f, e, \xi)$ to the TPA. TPA will check whether the equation $Q_{pro}e = \mu$ holds. If yes, TPA can believe that the cloud correctly stores the original files.

According to this scheme, the calculation of f is related to whether the original file is stored correctly. The reason is that $f' = \sum_{i=l_1}^{i=l_c} \beta_i F_i$. Only if the cloud server has the correct F_i can f' be calculated. But things are not always going as we wish, a malicious cloud server may calculate f as follows:

1) ID_c computes $\mu = Q_{pro}e$.
2) ID_c constructs the equation

$$Q_{pro}e = B\left(\sum_{i=l_1}^{i=l_c} \beta_i H_4(N_i||i) + Q_{ID_c}f - \xi H_6(w||v_w||\theta_w||\xi)\right). \tag{1}$$

In the above equation, all the parameters are public except f. Thus, ID_c could transform the above equation into

$$Q_{ID_c}f = B^{-1}Q_{pro}e + \xi H_6(w||v_w||\theta_w||\xi) - \sum_{i=l_1}^{i=l_c} \beta_i H_4(N_i||i). \tag{2}$$

Here $Q_{ID_c} \in \mathbb{Z}_q^{n\times m}$ is the cloud server's public key and the proof $f \in \mathbb{Z}_q^m$. Besides, the right side of the equation is also a known vector which belongs to $\mathbb{Z}_q^n$.

Recall that the security of DOPIV is based on the hardness of the inhomogeneous small integer solution (ISIS) problem. Generally speaking, the ISIS problem is to find a solution $\mathbf{x} \in Z^m$ with $|\mathbf{x}| < \delta$ for $\mathbf{Ax} = \mathbf{y}$, where $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$, $\mathbf{y} \in \mathbb{Z}_q^n$, and $\delta \in R$ is a positive number. However, in Equation (2) we cannot assume that f is bounded by a prefixed number. The reason is that f is generated by $f = f' + hH_6(w||v_w||\theta_w||\xi)$, where $f' = \sum_{i=l_1}^{i=l_c} \beta_i F_i$. Since F_i is the file generated by the user, the resulted f cannot be bounded in advance. Now we have the Equation (2), but we do not need to find a bounded solution f. Therefore, it is easy to solve this equation and then launch the attack.

Algorithm 1. The Process of Solving f

Input: $q = 11, n = 512, m = 3543$
1 Sage: $Z_q = Zmod(q)$
2 Sage: $MS = MatrixSpace(Z_q, n, m)$
3 Sage: $VS = MatrixSpace(Z_q, n)$
4 Sage: $Q_{ID_c} = MS.random\_element()$
5 Sage: $Mu = VS.random\_element()$
6 Sage: $f = Q_{ID_c}.solve\_right(Mu)$
Output: $f = (10, 0, 6, 1, 0, 8, 10, 3, 3, 10, \overbrace{\cdots}^{3530}, 0, 0, 0)$

In DOPIV q is set as a prime with $q \geqslant 3, m \geqslant \lceil 2n\log q \rceil$, to give an example, we let $q = 11, n = 512$, and $m = 3543$. We

randomly choose a matrix $Q_{ID_c} \in \mathbb{Z}_q^{n\times m}$ and a vector $Mu = B^{-1}Q_{pro}e + \xi H_6(w||v_w||\theta_w||\xi) - \sum_{i=l_1}^{i=l_c}\beta_i H_4(N_i||i) \in \mathbb{Z}_q^n$. After that, we use SageMath to solve the equation $Q_{ID_c}f = Mu$. As expected, it is easy to solve the vector f. Although f may not be a unique solution, we only need one solution which makes the equation hold. The details are given in Example 4.2.

By doing this, ID_c can easily pass the TPA's verification without storing the original file. According to the verification process μ is the most important parameter in the *Auditing Outsourced Data* stage. The cause of this attack lies in the fact that the cloud can not only calculate the vector μ by the equation $\mu = B(\sum_{i=l_1}^{i=l_c}\beta_i H_4(N_i||i) + Q_{ID_c}f - \xi H_6(w||v_w||\theta_w||\xi))$, but also through $\mu = Q_{ID_c}e$. Coincidentally, in the Section 4 part C of [6] the authors showed the correctness of the verification equation, and this also provides the correctness of f's calculation.

In the Section 5 of [6], the authors gave a security proof on the storage correctness guarantee. They showed the feasibility of an attack by tampering the original subfile F_i and forging the corresponding signature e_i. Indeed, if this attack works, the adversary may solve the ISIS problem. However, in our proposed proof forgery attack, the adversary does not need to modify the signature e_i. Therefore, the security proof in [6] cannot cover this attack.

5 The Improved Scheme

In this section we propose a simple and efficient method to improve Zhang *et al.*'s scheme. Our improved scheme can resist the above two attacks and achieve all the security goals of the original DOPIV scheme.

Similarly, our improved scheme also composed of five phases: *Setup*, *KeyExtract*, *Proxy-oriented Signing KeyGen*, *Proxy-oriented TagGen* and *Auditing Outsourced Data*. The main enhancement is in the *Proxy-oriented Signing KeyGen* part.

5.1 Our Improved Scheme

Setup: we assume that TPA is trusted but curious and the ID_p will not collude with the cloud server ID_c. And taking security parameters κ, q as inputs, the system is initialized as follows:

1) The discrete Gaussian distribution χ is determined by the security Gaussian parameter δ, σ.
2) KGC chooses six hash functions: $H_1 : \{0,1\}^{\kappa_1} \to \mathbb{Z}_q^{m\times m}$, $H_2 : \{0,1\}^{\kappa_1} \times \{0,1\}^{\kappa_1} \times \{0,1\}^{\kappa_2} \times \mathbb{Z}_q^n \to \mathbb{Z}_q^n$, $H_3 : \{0,1\}^{\kappa_1} \times \{0,1\}^{\kappa_1} \times \{0,1\}^{\kappa_2} \times \mathbb{Z}_q^m \to \mathbb{Z}_q^{m\times m}$, $H_4 : \{0,1\}^* \times \{0,1\}^* \to \mathbb{Z}_q^n$, $H_5 : \{0,1\}^{\kappa_1} \times \{0,1\}^* \to \mathbb{Z}_q^n$, and $H_6 : \{0,1\}^{\kappa_2} \times \mathbb{Z}_q^n \times \mathbb{Z}_q^m \times \mathbb{Z}_q^n \to \times\mathbb{Z}_q$.
3) KGC runs $TrapGen(q,n)$ to generate his public-private key pair (A, T_A) such that $A \cdot T_A = 0$.

KeyExtract: KGC use his key pair (A, T_A) to generate key pairs for the $ID_o \in \{0,1\}^{\kappa_1}$ as follows:

1) KGC computes $R_{ID_o} = H_1(ID_o)$ and $Q_{ID_o} = A(R_{ID_o})^{-1} \in \mathbb{Z}_q^{n\times m}$.
2) Run $NewBasisDel(A, R_{ID_o}, T_A, \delta)$ to generate $T_{ID_o} \in \mathbb{Z}_q^{m\times m}$ which is a random short lattice basis of $\Lambda_q^{\perp}(Q_{ID_o})$ and kept by the ID_o as the private key.

The public-private key pair of ID_o is (Q_{ID_o}, T_{ID_o}), similarly, KGC generates (Q_{ID_p}, T_{ID_p}) for the proxy, (Q_{ID_c}, T_{ID_c}) for the cloud server. Besides, KGC chooses a identity tag $v_p \in \{0,1\}^{\kappa_2}$ for ID_p, ID_p keeps this tag as a secret and ID_p informs TPA about his identity tag v_p though a security channel (ID_p may go to the TPA for registration in person).

Proxy-Oriented Signing KeyGen. ID_o first authenticates the identity of ID_p, after that, ID_p will generate proxy-oriented signing private key for himself. The whole process is as follows:

1) ID_o creates the warrant $w \in \{0,1\}^{\kappa_2}$ which contains the delegation policy, valid period of delegation, and the identities of the original data owner and the proxy. After that, ID_o choose a vector $v_1 \in \mathbb{Z}_q^m$ randomly, then, ID_o sends (ID_o, ID_p, w, v_1) to TPA.
2) On receiving the (ID_o, ID_p, w, v_1), TPA computes $R_p = H_3(ID_o||ID_p||v_p||v_1)$. Then, TPA sends R_p to ID_o.
3) ID_o sends (ID_o, v_1) to ID_p.
4) ID_p computes $R_p^* = H_3(ID_o||ID_p||v_p||v_1)$ and sends it back to ID_o.
5) ID_o verifies the ID_p's identity by checking the equation $R_p^* = R_p$, if it holds, ID_o randomly chooses a vector $v_w \in \mathbb{Z}_q^n$, computes $u_w = H_2(ID_o||ID_p||w||v_w)$ and $\theta_w = SamplePre(Q_{ID_o}, T_{ID_o}, u_w, \sigma)$. Then, ID_o sends (w, v_w, θ_w) directly to the proxy ID_p.
6) After receiving (w, v_w, θ_w), ID_p computes $u_w = H_2(ID_o||ID_p||w||v_w)$ and validates the signed warrant w by the equation $Q_{ID_o}\theta_w = u_w$. If the equation holds, ID_p computes $R_w = H_3(ID_o||ID_p||v_p||\theta_w)$ and generates the proxy-oriented private key $T_{pro} = NewBasisDel(Q_{ID_p}, R_w, T_{ID_p}, \delta)$ and the corresponding proxy-oriented signing public key $Q_{pro} = Q_{ID_p}(R_w)^{-1}$.

Proxy-Oriented TagGen. ID_p divides file F into l subfiles $F = \{F_1, F_2, \ldots, F_l\}$, where $F_i \in \mathbb{Z}_q^m$ and each F_i has its corresponding name $N_i \in \{0,1\}^*$. The proxy generates proxy-oriented signing private key and generates the signature of each subfile F_i:

1) ID_p computes $Q_{ID_c} = AR_{ID_c}^{-1}$ and $\eta_i = H_4(N_i||i) + Q_{ID_c}F_i$.
2) ID_p computes $\lambda_j = H_5(ID_p||j)$, where $1 \leqslant j \leqslant n$ and inner products $\rho_{i,j} = \langle \eta_i, \lambda_j \rangle$, then sets $\rho_i = (\rho_{i,1}, \ldots, \rho_{i,n})^{\top} \in \mathbb{Z}_q^n$.
3) ID_p computes $Q_{pro} = Q_{ID_p}R_w^{-1}$ to generate ρ_i's signature $e_i = SamplePre(Q_{pro}, T_{pro}, \rho_i, \sigma)$.

After that, ID_p outsources $(F_i, N_i, e_i), (1 \leqslant i \leqslant l)$ and (w, v_w, θ_w) to the cloud server. The cloud server will first compute $u_w = H_2(ID_o||ID_p||w||v_w)$ and check the ID_p's validity by the equation $Q_{ID_o}\theta_w = u_w$. If the equation holds, the cloud server stores $(w, v_w, \theta_w, F_i, N_i, e_i)$.

Auditing Outsourced Data. In this phase, TPA will verify the primitive data's integrity. There are three steps:

- *Challenge:* The TPA chooses a c-element subset $\Omega = \{l_1, \ldots, l_c\}$ randomly and a random binary string $\beta = (\beta_{l_1}, \ldots, \beta_{l_c})$. Then TPA sends the challenge message $chal = \{i, \beta_i\}_{i\in\Omega}$ to the cloud server.
- *ProofGen:* The cloud server computes $f' = \sum_{i=l_1}^{i=l_c}\beta_i F_i \in \mathbb{Z}_q^m$, $e = \sum_{i=l_1}^{i=l_c}\beta_i e_i \in \mathbb{Z}_q^m$, then, ID_c chooses a random vector $\xi \in \mathbb{Z}_q^n$ and comutes $h = SamplePre(Q_{ID_c}, T_{ID_C}, \xi, \sigma)$, $f = f' + hH_6(w||v_w||\theta_w||\xi)$. After that, ID_c sends $Proof = (f, e, \xi)$ and (w, v_w, θ_w) to the TPA.

TABLE 1
Computation Costs

Schemes	Delegation verification	Integrity verification
DOPIV [6]	$Hash + nm \cdot mul$	$(n + c + 1)Hash + (n^2 + 2nm + n)mul$
Our scheme	$3Hash + nm \cdot mul$	$(n + c + 2)Hash + (n^2 + 3nm + n)mul$

- *ProofVerify:* TPA validates (w, v_w, θ_w) to check ID_p's validity. Next, TPA computes $R_w = H_3(ID_o||ID_p||v_p||\theta_w)$ and $Q_{pro} = Q_{ID_p}(R_w)^{-1}$. After that, TPA computes $\lambda_j = H_5(ID_p||j), (1 \leqslant j \leqslant n)$, sets $B = (\lambda_1, \ldots, \lambda_n)^\top$ and computes $\mu = B(\sum_{i=l_1}^{i=l_c} \beta_i H_4(N_i||i) + Q_{ID_c} f - \xi H_6(w||v_w||\theta_w||\xi))$, if the equation $Q_{pro} e = \mu \pmod q$, $(0 \leqslant \|e\| = \|\sum_{i=l_1}^{i=l_c} e_i\| \leqslant c\sigma\sqrt{m})$ holds, the primitive data has not been modified.

5.2 Security Analysis

In this section, we analyze the security of our improved scheme.

1) Resistance to Inside Attack: The success of the inside attack lies in that ID_p doesn't own a unique identity tag and ID_o will not authenticate the identity of the ID_p before the *Proxy-oriented Signing KeyGen* phase. In our improved scheme, KGC gives ID_p a identity tag v_p which is only known by the TPA and himself. Since TPA is always trusted, when ID_o ask for ID_p's identity, TPA will calculate the $R_p = H_3(ID_o||ID_p||v_p||v_1)$ honestly. On receiving the R_p, ID_o will also ask the current ID_p to calculate the same R_p. Clearly, if the current ID_p doesn't own his identity tag v_p, he cannot finish this calculation. Besides, the vector v_1 is randomly chosen by the ID_o each time before interaction with the TPA, so it is impossible for replaying the R_p to cheat the ID_o.
2) Resistance to Proof Forgery Attack: The cause of the proof forgery attack lies in the leakage of μ, this parameter supposed to be a secret but there is a public method for its calculation. In the original scheme, the authors intended to calculate the μ only through the equation $\mu = B(\sum_{i=l_1}^{i=l_c} \beta_i H_4(N_i||i) + Q_{ID_c} f - \xi H_6(w||v_w||\theta_w||\xi))$, where f is related to the stored data and only in this way can ID_c compute the μ. Unfortunately, ID_c can calculate the μ just like what the TPA does, that is the equation $Q_{pro} e = \mu \pmod q$. In our improved scheme, when generating the proxy-oriented signing public key $Q_{pro} = Q_{ID_p}(R_w)^{-1}$, ID_p computes $R_w = H_3(ID_o||ID_p||v_p||\theta_w)$. Since only TPA knows the ID_p's identity tag v_p, ID_c can no longer calculate the parameter μ.

5.3 Performance Comparisons

In this section, we compare our improved scheme with the original mechanism in communication cost and computation cost. In the original scheme, the key factors that affect performance of a practical DOPIV are actually communication costs in integrity verification between the TPA and the cloud server and the computation costs on the side of TPA. Thus, in order to show that our improved scheme can solve the vulnerabilities of the original scheme while maintaining the efficiency of it, we show the comparison result in communication costs and computation costs in the integrity verification phase and computation costs in delegation verification phase.

TABLE 2
Communication Costs in Integrity Verification

Schemes	Communication overhead
DOPIV [6]	$(2m + n)\|q\| + (\|l\| + 1)c$
Our scheme	$(2m + n)\|q\| + (\|l\| + 1)c$

The comparison results are shown in Tables 1 and 2. Specifically, $|q|$ denotes the bit length of an element in $\mathbb{Z}_q$ and $|l|$ denotes the number of the total data blocks. The number of the challenge data blocks is denoted by c. Besides, we denote the running time of a general hash function by $Hash$ and the running time of a general multiplication by mul.

6 Conclusion

In this paper, we have proposed an inside attack and a proof forgery attack against DOPIV. The cause of the inside attack is that the proxy lacks identity tag and the original data owner doesn't authenticate the identity of the proxy. The cause of the proof forgery attack is that the cloud server can calculate the data storage proof without using the stored data. Fortunately, we have come up with a simple and efficient solution to thwart the proposed attacks and our improved scheme maintains all the features of the original protocol. The design of a quantum-secure data outsourcing scheme with public integrity verification is very important and we hope the proposed attacks and the improvement can help inspire new designs.

Acknowledgments

This work was supported in part by the National Natural Science Foundation of China under Grants 61672029 and 61802354, and the Guangxi Key Laboratory of Trusted Software. J. Ding would like to thank Taft Foundation, NSF and US Air Force for partial support.

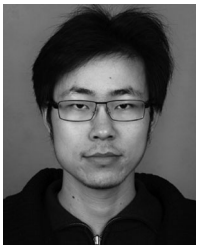
References

[1] D. Song, E. Shi, I. Fischer, and U. Shankar, "Cloud data protection for the masses," *Computer*, vol. 45, no. 1, pp. 39–45, 2012.
[2] C. K. Chu *et al.*, "Security concerns in popular cloud storage services," *IEEE Pervasive Comput.*, vol. 12, no. 4, pp. 50–57, Oct.–Dec. 2013.
[3] P. W. Shor, "Algorithms for quantum computation: Discrete logarithms and factoring," in *Proc. 35th IEEE Annu. Symp. Foundations Comput. Sci.*, 1994, pp. 124–134.
[4] C. Cheng, R. Lu, A. Petzoldt, and T. Takagi, "Securing the Internet of Things in a quantum world," *IEEE Commun. Mag.*, vol. 55, no. 2, pp. 116–120, Feb. 2017.
[5] G. Alagic *et al.*, "Status report on the first round of the NIST post-quantum cryptography standardization process," 2019. [Online]. Available: https://nvlpubs.nist.gov/nistpubs/ir/2019/NIST.IR.8240.pdf
[6] X. Zhang, J. Zhao, C. Xu, H. Wang, and Y. Zhang, "DOPIV: Post-quantum secure identity-based data outsourcing with public integrity verification in cloud storage," *IEEE Trans. Services Comput.*, to be published, doi: 10.1109/TSC.2019.2942297.

[7] M. Ajtai, "Generating hard instances of lattice problems," in *Proc. 28th Annu. ACM Symp. Theory Comput.*, 1996, pp. 99–108.
[8] J. Alwen and C. Peikert, "Generating shorter bases for hard random lattices," *Theory Comput. Syst.*, vol. 48, no. 3, pp. 535–553, 2011.
[9] C. Gentry, C. Peikert and V. Vaikuntanathan, "Trapdoors for hard lattices and new cryptographic constructions," in *Proc. 40th Annu. ACM Symp. Theory Comput.*, 2008, pp. 197–206.
[10] S. Agrawal, D. Boneh and X. Boyen, "Lattice basis delegation in fixed dimension and shorter-ciphertext hierarchical IBE," in *Proc. Annu. Cryptology Conf.*, 2010, pp. 98–115.

Qingxuan Wang received the MS degree in information security from the China University of Geosciences, Wuhan, P. R. China, in Jun. 2020. He is currently working toward the PhD degree from the College of Cyber Science, Nankai University, Tianjin, P. R. China. His research interests include applied cryptography and password-based authentication.

Chi Cheng (Member, IEEE) received the PhD degree in information and communication engineering from the Huazhong University of Science and Technology, Wuhan, P. R. China, in Dec. 2013. He is currently an associate professor with the School of Computer Science, China University of Geosciences, Wuhan, P. R. China. From 2014 to 2016, he was an International Research Fellow of the Japan Society for the Promotion of Science (JSPS), Institute of Mathematics for Industry, Kyushu University, Japan. His research interests include cryptography and information security, especially lattice-based cryptography and its applications.



Rui Xu received the BS and MS degrees in engineering from the University of Science and Technology of China, and the PhD degree in mathematics from Kyushu University, in 2015. He worked as an associate researcher with the information security group of KDDI Research Inc., Japan from 2015 to 2018. He is currently with the China University of Geosciences (Wuhan). His research interests include secret sharing schemes and cloud security. He is also interested in multiparty computation and privacy preserving techniques.

Jintai Ding received the BA degree from Xian Jiao tong University, in 1988, the MA degree from the University of Science and technology of China, in 1990, and the PhD degree from Yale, in 1995. He is currently a professor with the Yau Mathematical Sciences Center, Tsinghua University and the director of Ding Lab in Privacy Protection and Blockchain Security, Beijing Institute of Mathematical Sciences and Applications. Before that he was a Charles Phelps Taft professor with the Department of Mathematical Sciences, University of Cincinnati. He was a lecturer with the Research Institute of Mathematical Sciences of Kyoto University from 1995 to 1998. In 2006-2007, he was a visiting professor and Alexander Von Humboldt fellow with TU Darmstadt. He received the Zhong Jia Qing Prize from the Chinese Mathematical Society in 1990 for his master's thesis. His research was originally in quantum affine algebras and its representation theory, where he was credited for the invention of the Ding-Iohara-Miki algebra. His current research interests include post-quantum cryptography, in particular, multivariate cryptography, latticed-based cryptography, and quantum-proof blockchain. He was a co-chair of the 2nd,10th, and 11th international conference on post-quantum cryptography. He and his colleagues developed the Rainbow signature, the Simple Matrix encryption and the LWE-based key exchange schemes. Rainbow is a third round candidate for the NIST post-quantum standardization process. He and his colleagues completely broke a NIST second round post-quantum signature candidate LUOV and a third round candidate GeMSS (HFEv-).

Zhe Liu received the BS and MS degrees from Shandong University, China, in 2008 and 2011, respectively, and the PhD degree from the Laboratory of Algorithmics, Cryptology and Security, University of Luxembourg, Luxembourg, in 2015. He is currently a professor with the College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, China. His research interests include security, privacy, and cryptography solutions for the Internet of Things. He has coauthored more than 80 research peer-reviewed journal and conference papers. He was a recipient of the prestigious FNR Awards-Outstanding PhD Thesis Award in 2016, ACM CHINA SIGSAC Rising Star Award in 2017, the outstanding young researcher from China Association for Science and Technology in 2018, as well as DAMO Academy Young Fellow in 2019. He has served as program committee member in more than 50 international conferences.