An Effective Scheme for Provable Data Possession

Shanyue Bu
Computer and Software Engineering School
Huaiyin Institute of Technology
Huaian 223003, China
bushanyue@126.com

Mengdie Huang
Electronic Information Engineering School
Huaiyin Institute of Technology
Huaian 223003, China

Kun Yu
Computer and Software Engineering School
Huaiyin Institute of Technology
Huaian 223003, China

***Abstract*—**At present, almost all the provable data possession schemes require a large number of exponent calculations. In this paper a new scheme based on the homomorphy of NTRU algorithm is proposed. The scheme includes modules of Key Generation, Tag Generation, Challenge, Proof Generation, and Verification. This scheme requires only plus, minus, multiplication and modular computation. The correctness and security of the scheme are analyzed and the result shows that this scheme is reliable. Compared with other existing schemes, this scheme is with higher security, less computation, storage and communication cost.

Keywords—Provable Data Possession; NTRU; cloud storage; data integrity;

I. INTRODUCTION

With growing popularity of cloud storage technology, an increasing amount of owners entrust their data to the Cloud Service Provider (CSP) for management, so as to reduce the data management cost. However, CSP is not necessarily trustable, because hacker may invade cloud server to steal, delete and tamper data of owner. And CSPs may conceal their defects in the damage and loss of data to shift the responsibility for their behaviors such as improper management. The Provable Data Possession (PDP) allows owner to verify whether or not incredible CSP has preserved complete data of the owner, so as to prevent the data from being deleted and tampered

The idea of PDP was firstly proposed by the research [1]. Then, the study [2] defined the PDP scheme for the first time. By dividing the data into several data blocks, the scheme can generate a homomorphic verifiable tag for each data block based on the homomorphic multiplication of the RSA algorithm and the hash function. Afterwards, according to demands of the owner, CSP can provide possessed data blocks and data possession evidence. And then owner verifies the correctness of data possession evidence provided by CSP. Based on PDP, an open and verifiable scheme of the PDP (PoS) was put forward [3]. The Proofs of Retrievability (POR) was proposed [4] for the first time. The scheme of dynamic PDP was put forward [5], and this scheme has realized insertion, modification and deletion of the data block based on the PDP scheme. The research [6] proposed the scheme of Multiple-replica PDP. The scheme of cooperative PDP in the mixed cloud environment was constructed [7].Meanwhile, according to the existing researches, PDP can be realized by utilizing the symmetric encryption algorithm, the homomorphic hash function and third party auditor[8-10].

Based on existing PDP schemes, one of the problems lies in the great computational overhead which is expected to be aggravated particularly in the case of great data volume. Based on the research [2], the paper established a new PDP scheme by using the addition homomorphism of NTRU algorithm and the hash function, which was called the N-PDP scheme. The N-PDP scheme no longer needs to conduct the power operation except for some simple addition, multiplication and modular operations. This can substantially improve the execution efficiency of the scheme.

II. RELEVANT KNOWLEDGE

In terms of the safety, compared with the RAS and ECC algorithm, the NTRU algorithm is capable of resisting attack from the quantum computation [11]. Considering demands on the algorithmic speed and the memory, the NTUT declared that the speed of the NTUR is 200 times higher than that of the RSA. As the key length of the NTRU increases by n, the operation times of the RSA per second decreases by n^3, while the NTRU in the same time only reduces by n^2.when the key length remains constant, the NTRU is safer than the RSA, and shows few demands on the memory resources [12]. With respect to the standardization, the NTRU has officially been

regarded as the IEEE P1363 standard, and agreed by the institutes such as CEES, IETF, IEEE 802.15.3 and so on. Detailed descriptions of the NTRU can be seen in research[12,13,14].

Suppose that there are two plaintexts (x, y). Here $E(\cdot)$ represents the encryption algorithm of NTRU, and $D(\cdot)$ as its decryption algorithm. (*h*, *f*) is a pair of keys in the NTRU algorithm, which can demonstrate addition homomorphism of the NTRU algorithm. If the user Bob encrypts plaintexts x and y with the NTRU respectively, the ciphertext of the E(x) and E(y) then can be obtained:

$$E_h(x+y) = E_h(x) + E_h(y) \quad (1)$$

$$D_f(E_h(x+y)) = x+y \quad (2)$$

That is to say, the user Alice can get value of the $E(x+y)$ by calculating the $E(x)+E(y)$, or obtain the value of $x+y$ by decrypting the $D(E(x)+E(y))$. However, Alice fails to know the value of x and y respectively. Therefore, it realizes that the operation on the ciphertext in the ciphertext space acts same role in the operation on the plaintext in the plaintext space.

III. THE N-PDP SCHEME

The N-PDP scheme consists of data storage and data possession inspection. In the data storage part, owner utilizes the key to generate the key required by the production scheme of module KeyGen and divides data into *n* data blocks. And then it generates corresponding verification tag for each data block by employing the TagGen model. Afterwards, the owner sends data blocks and data block tags to CSP. In the case of inspecting data possession, the owner produces challenge information based on the Challenge module and then sends them to CSP. Afterwards, according to the challenge information, CSP generates data possession evidence by the ProofGen module, and finally dispatches them to the owner. Finally, the owner is required to verify the validity of the data possession evidence offered by CSP based on the Verification module. If it passes the verification, the data stored in CSP is integral, otherwise, the data is damaged. The symbols used in the scheme are described as follows:

M refers to the outsourced data of the owner

n is the number of data blocks. M is divided into n blocks with same size

m_i indicates the data blocks contained in M, $m_i \in M$, $(1 \le i \le n)$

$H_{k2}(\cdot)$ is the hash function with key

t_i is the data block tag

$\phi(\cdot)$ represents pseudo-random function

$\varphi(\cdot)$ is pseudo-random permutation

$E_h(\cdot)$ is the encryption algorithm of NTRU

A.KeyGen $(1^k) \rightarrow (k1, k2, k3)$

Firstly, the KeyGen generates keys (*k*1 and *k*2) using the NTRU algorithm, and then produces the key *k*3 randomly, which can be used to generate the data block tag and verify the data possession evidence. Specific processes are given as following:

Step 1: input the safety parameter *k*

Step 2: select relevant parameters including *N*, *p* and *q* of the NTRU algorithm according to using specific method in reference[14].

Step 3: compute key pair(*k*1 and *k*2)of the NTRU algorithm

Step 4: select the key *k*3 randomly

Step 5: output keys (*k*1, *k*2 and *k*3)

B. TagGen $(k1, k2, k3, M) \rightarrow (m_i, t_i, k2, \varphi_{k4}(\cdot), \phi_{k5}(\cdot))$

As to main responsibility of the TagGen, the owner divides the data M into n data blocks m_i of same size and then computes verifiable tag t_i for each data block. Afterwards, m_i and t_i are dispatched to CSP for storage. The owner is expected to delete the local data M, m_i and t_i, only few data such as the key are retained. The detailed processes are shown as below:

Step 1: input (*k*1, *k*2,*k*3 *M*)

Step 2: divide *M* into n data blocks of same size and marks each block with $m_i, 1 \le i \le n$.

Step 3: compute $h_i = H_{k2}(k3 \| i)$, $h_i^m = H_{k2}(m_i)$. Where h_i can prevent the attacker from forging the tag, and h_i^m can reduce the tag length

Step 4: compute $t_i = E_{k1}(h_i^m + h_i), 1 \le i \le n$.

Step 5: produce the pseudo-random permutation $\varphi_{k4}(\cdot)$, which can prevent CSP from generating data block and corresponding verifiable tag in advance.

Step 6: produce the pseudo-random function $\phi_{k5}(\cdot)$, so as to prevent CSP from producing the sum of the challenge data blocks in advance.

Step 7: dispatch $(m_i, t_i, k2, \varphi_{k4}(\cdot), \phi_{k5}(\cdot))$ to CSP.

Step 8: delete the local stored data *M*, m_i and t_i, and keep *k*1,*k*2,*k*3, $\varphi_{k4}(\cdot)$, $\phi_{k5}(\cdot)$.

C. Challenge()→(c, k4,k5)

In data possession inspection, owner produces a challenge by calling the challenge module. The challenge is composed of the number of the challenge data block, random permutation key, and the key for random data production. The specific process is as follows:

Step 1: the owner offers the number of the challenge data blocks c, and then generates the random permutation key *k*4 and the key for random data production *k*5.

Step 2: dispatch (*c*, *k*4, *k*5) to CSP.

D. ProofGen $(c,k4,k5)\rightarrow(T,M^{'})$

According to the (*c*, *k*4, *k*5) given by the owner, the physical location of *c* challenge data blocks can be obtained by the Pseudo-random permutation $\varphi_{k4}(\cdot)$. And *c* pseudo ransom numbers can be obtained based on the pseudo-random function $\phi_{k5}(\cdot)$. And then, CPS computes *T* and $M^{'}$, and sends them to the owner as the data possession evidence. Specific processes are as follows:

Step 1: input (*c*, *k*4, *k*5)

Step 2: compute the subscript of the data block, $i_j=\varphi_{k4}(j),1\le j\le c$.

Step 3: calculate the challenge random number, $a_i=\phi_{k5}(j),1\le j\le c$.

Step 4: calculate $T=\sum_{j=1}^{c}a_j\cdot t_{i_j} \bmod N$,

$$M^{'}=\sum_{j=1}^{c}a_j\cdot H_{k2}(m_{i_j})\bmod N .$$

Step 5: dispatch (*T*, *M*') to the owner.

E.Verification $(c,k1,k2,k3,k4,k5,T,M^{'})\rightarrow$ ***(Success,Failure)***

Once receiving data possession evidence, the Owner computes the physical location of the *c* challenge data blocks by $\varphi_{k4}(\cdot)$, and calculates *c* pseudo-random numbers based on $\phi_{k5}(\cdot)$. Thus, the hash value of the corresponding challenge data blocks is obtained. Afterwards, one τ value is calculated. Then, by comparison, if τ and *T* are equal, CPS has a complete data possession, and then output the Success. Otherwise the data are damaged, and the output is Failure. The specific process is as follows:

Step 1: input $(c,k1,k2,k3,k4,k5,T,M^{'})$

Step 2: compute $i_j=\varphi_{k4}(j)$, $a_j=\phi_{k5}(j),1\le j\le c$.

Step 3: calculate $h=\sum_{j=1}^{c}a_j\cdot H_{k2}(k3\|i_j)\bmod N$.

Step 4: compute $\tau=E_{k1}(M^{'}+h)$, if $\tau=T$, and then output the *Success*, which confirms the successful verification on data possession evidence. Otherwise, output the *Failure*, which indicating failure of the verification on data possession evidence.

IV. SECURITY ANALYSIS

Theorem 1: Suppose that both the owner and CSP are credible, and data stored by CSP are complete, the data possession evidence provided by CSP can pass the N-PDP.

Proof: If all the data blocks and data block tags are stored in CSP perfectly, according to the $T=\sum_{j=1}^{c}a_jt_{i_j} \bmod N$, $h_{i_j}^{m}=H_{k2}(m_{i_j})$, $M^{'}=\sum_{j=1}^{c}a_jh_{i_j}^{m} \bmod N$ provided by CSP, we obtain:

$$h=\sum_{j=1}^{c}a_jH_{k2}(k1\|i_j)\bmod N=\sum_{j=1}^{c}a_jh_{i_j} \bmod N \quad (3)$$

$$T=\sum_{j=1}^{c}a_jt_{i_j} \bmod N$$

$$=\sum_{j=1}^{c}a_j\cdot E_{k1}((h_{i_j}^{m}+h_{i_j})\bmod N)$$

$$=E_{k1}((\sum_{j=1}^{c}a_jh_{i_j}^{m}+\sum_{j=1}^{c}a_jh_{i_j})\bmod N)=E_{k1}(M^{'}+h) \quad (4)$$

The proof is completed.

Theorem 2: If the data stored in CSP by the Owner are damaged or lost, the probability of the data possession evidence provided by CSP for its successful pass of the N-PDP can nearly be ignored.

Proof: Suppose that the attacker acquires *n* data block sequences *M*= $\{m_1,m_2,\ldots,m_n\}$ and their tag sequences *T*=$\{t_1,t_2,\ldots,t_n\}$. As the challenger produces *c* data blocks, the attacker requires providing the data possession evidence. When the attacker possesses all the data blocks but loses part of the tags, if the data possession evidence is able to pass the data possession inspection of the challenger, it indicates N-PDP is unsafe.

The challenger sends a challenge value to the attacker chal= (*c*, *k*4, *k*5).

Assuming that the attacker loses some data block tags t_{i_j}, …, t_{i_k}, $(j,\cdots,k)\subseteq(1,\cdots,c)$, he can forge some data block tags $t_{i_j}^{'},\cdots,t_{i_k}^{'}$ and calculate:

$$T=(a_1t_{i_1}+\ldots+a_jt_{i_j}^{'}+\ldots+a_kt_{i_k}^{'}+\ldots+a_ct_{i_j})\bmod N \quad (5)$$

$$M^{'}=\sum_{j=1}^{c}a_j\cdot H_{k2}(m_{i_j})\bmod N \quad (6)$$

The attacker firstly sends data possession evidence $(T,M^{'})$ back to the challenger. And then,the challenger computes $E_{k1}(M^{'}+h)$ by calling verifying process. If $E_{k1}(M+h)=T$,there is $a_jt_{i_j}+\ldots+a_kt_{i_k}=a_jt_{i_j}^{'}+\ldots+a_kt_{i_k}^{'}$.

It is obvious that if the attacker wants to win the game, he has to successfully forge the $a_j t_{i_j} + \ldots + a_k t_{i_k} = a_j t'_{i_j} + \ldots + a_k t'_{i_k}$. Owing to m_{i_j} is possessed by the attacker, in order to get t_{i_j} from m_{i_j} , the attacker has to solve the mathematical problem regard calculating the shortest vector in the lattice without knowing *k*1 and *k*3. Meanwhile, he has to find a collision in the anti-collision hash function, which is expected to destroy security of the NTRU algorithm and unidirectionality of the hash function. As a matter of fact, it is of great difficulty to calculate the shortest vector in the lattice within the polynomial time, and find a collision in the anti-collision hash function. Therefore, in the case some data block tags are lost or modified, it is nearly impossible that the attacker can pass the data possession inspection of the challenger.

Meanwhile, it can demonstrate with same method: when the attacker loses some data blocks but possesses all the tags, or when it loses partial data blocks and their tags, it is little possibility that the attacker can pass the data possession inspection of the challenger.

Because the challenger only challenges part of the data blocks in the verification process, some damaged or lost data may not be detected. Therefore, as the number of the data blocks being challenged in each time increases, the correctness rate of the data possessed by CSP is expected to be higher than before. Supposing that the damaging or lost rate of the data block is 1%, according to the study [2], when the number of the data blocks verified in each time is c=460, the probability of discovering data damage or loss is more than 99%. Therefore, when the data blocks to be challenged are large enough in size or the challenge times is increased, the possibility that the data possession evidence provided by CSP passes the inspection of the challenger is nearly zero under the condition of damaged or lost data blocks. This ends the proof.

V. COMPUTATIONAL OVERHEAD

Most of the relevant research results on the PDP schemes are based on the RSA algorithm and derive from research [2]. Therefore, comparative analysis is conducted by utilizing the PDP scheme [2] and the N-PDP scheme designed in the paper.

The time of one encryption operation of the RSA algorithm as T_R , the time of one encryption operation of the NTRU algorithm as T_N , here T_R is far greater than T_N [12]; the time of the one hash operation is T_h ;the time of one multiplication or division operation denotes T_m ; the time of one additive operation is T_a .Because the times of the N-PDP scheme and PDP scheme using the pseudo-random permutation and pseudo-random function are same, the KeyGen process takes little computational overhead in the scheme, they are therefore neglected. The computational overhead of the PDP scheme and N-PDP scheme are illustrated in Table 1. As shown in Table 1, the computational overhead of the N-PDP is much lower than that of the PDP.

TABLE Ⅰ
COMPARISON OF THE COMPUTATIONAL OVERHEAD

	PDP scheme	N-PDP scheme
TagGen	$n*(T_h+T_m+2T_R)$	$n*(2T_h+T_a+T_N)$
ProofGen	$c*(2T_m+T_a+T_R)+T_h+T_R$	$c*(2T_m+2T_a+T_h)$
Verify	$c*(T_h+T_R)+2T_R+T_m+T_h$	$c*(T_m+T_a+T_h)+T_N+T_a$

VI. STORAGE OVERHEAD

As to the storage overhead of the N-PDP, the PDP needs to store the key, data block tag and random function except for the storage of the data block, which has to occupy extra storage space. As the size of the key and random function can be ignored, the storage overhead by the data block tag is mainly analyzed.

N-PDP uses SAH-1 as the hash function with the size of each data block tag of 2*128 bit=256 bit. However, in order to ensure security of the PDP, the size of each data block requires being at least 1024 bit. When data are divided into *n* data blocks with same size, the extra storage overhead of the N-PDP and PDP are *n**256 bit and *n**1024 bit respectively. Obviously, as to the same data, the greater the data blocks to be divided, the higher storage overhead it takes. For instance, as to the data of 4 GB, if each data block is 4 KB, there are2^{20}data blocks in total. The extra storage required by PDP is 2^{20}*1024=128 MB, which occupies 3.125% of the original data. However, the extra storage required by N-PDP is2^{20}*256=32 MB, which only takes up 0.789% of the original data.

VII. COMMUNICATION OVERHEAD

In the case of storing data, the owner sends the data block, data block tag, key and random function to CSP. Its communication traffic is directly related to data block size, the number and size of the data block tag. The communication traffic of the key and random function can be ignored. In spite of the data block, extra communication overhead of the N-PDP is *n**256 bit and communication overhead of the PDP is *n**1024 bit. It is evident that communication overhead of the N-PDP is far lower than that of the PDP. With regard to the data possession inspection and CSP dispatches the data possession evidence to the owner, which is basically equal to the communication traffic of the PDP scheme.

VIII. CONCLUSION

A scheme supporting the N-PDP in the cloud storage is put forward in the paper, to check the integrity of the data stored in the cloud server. The scheme adopts the NTRU algorithm under the precondition of ensured security. Compared with the RSA algorithm, the scheme proposed is able to avoid large

amounts of exponential operation, reduce the overhead of computation, communication and storage, and increase efficiency and security in the execution of provable data possession scheme. It is proved to be suitable to various network environments. The scheme allows the modification and append operations on the data block, but not supports the deletion and insertion. The appropriate modification on the scheme can make it support the multiple-replica PDP and the verifiable third party PDP. The NTRU is expected to be used to design dynamic PDP scheme supporting deletion and insertion operations in the study in near future.

REFERENCES

[1] Y. Deswarte, J.J.Quisquater and A.Saidane,Remote integrity checking//*Proc of Integrity and Internal Control in Information Systems*, Berlin, Heidelberg:Springer-Verlag, pp.1-11,2003.

[2] G.Ateniese, R.Burns, R.Curtmola,et al,Provable data possession at untrusted stores[C] //*Proc of 14th ACM Conference on Computer and Communications Security*, New York, USA, pp.598-609,2007.

[3] G.Ateniese, S.Kamara and J.Katz.Proofs of storage from homomorphic identification protocols// *Proc of ASIACRYPT 09*, Tokyo, Japan, pp.319-333,2009.

[4] A.Juels and B.S.Kaliski,Pors:Proofs of retrievability for large files//*Proc of 14th ACM Conference on Computer and Communications Security*, New York, USA, pp.584-597,2007

[5] C.Gritti, W.Susilo and T.Plantard,Efficient Dynamic Provable Data Possession with Public Verifiability and Data Privacy,*Lecture Notes in Computer Science* ,Vol. 9144, pp.395-412,2015.

[6] Y.Y Fu,M.Zhang, K.Q.Chen and D.G.Feng,Proofs of Data Possession of Multiple Copies,*Journal of Computer Research and Development* ,Vol.51 No.7, pp.1410-1416,2014.

[7] H.Q.Wang and Y.Q.Zhang,On the Knowledge Soundness of a Cooperative Provable Data Possession Scheme in Multicloud Storage,I*EEE transactions on parallel and distributed systems*, vol.25, no.1, pp.264-267,2014.

[8] G.Ateniese, R.D.Pietro, L.V.Mancini,et al.Scalable and efficient provable data possession//*Proc of Secure Comm 08*.New York, pp.1-10,2008.

[9] L.X.Chen,A Homomorphic Hashing Based Provable Data Possession,*Journal of Electronics & Information Technology*, Vol.33,No.9,pp.2219-2204,2011.

[10] H.Zhuo, Z.Sheng and N.H.Yu,A privacy-preserving remote data integrity checking protocol with data dynamics and public verifiability,*IEEE Transactions on Knowledge and Data Engineering*, Vol.23,No.9, pp.1432-1437, 2011.

[11] C.Ludwig,A Faster Lattice Reduction Method Using Quantum Search,*Lecture Notes in Computer Science*,Vol.2906, pp.199-208,2003.

[12] NTRU Crypto – up to 200x faster than RSA,https://www.securityinnovation.com/products/encryption-libraries/NTRU-crypto/

[13] Download the IEEE P1363.1 Draft,http://grouper.ieee.org/groups/1363/lattPK/draft.html.

[14] J.Hoffstein and J.Silverman,Optimizations for NTRU//*Proc of In Proceedings of Public-Key Cryp tography and Computational Number Theory*, Warsaw, Poland, pp.11-15,2000.