Cloud Storage Retrievability Based On Third Party Audit

Zhongyuan Qin[1,2*], Yunyan Song[1], Qunfang Zhang[3], Jie Huang[1]
[1]Information Science and Engineering School, Southeast University, Nanjing, Jiangsu, China,
[2]Key Lab of Information Network Security, Ministry of Public Security, Shanghai, China
[3]Computer Department, Nanjing Institute of Artillery Corps, Nanjing, Jiangsu, China

***Abstract*—Cloud storage can relieve users of the burden of local data storage and maintenance. How to ensure the integrity of data stored in Cloud is a key problem. Based on the state-of-the-art solutions, we present a scheme for data storage retrievability in Cloud Computing using third party audit (TPA), which can fulfill the demands of data integrity, data confidentiality, data extraction, credibility control of third-party audit etc. Finally performance evaluation is given.**

Keywords- cloud computing; retrievability; data integrity; third party audit

I. INTRODUCTION

IDC estimates that by 2020, nearly 40% of the information will be involved in cloud computing. The era of cloud has arrived. As a kind of emerging technology, which was put forward by Google in 2006[1], cloud computing is attracting more and more attention. Google, Amazon, Apple, Microsoft and other IT industry giants have been put the manpower and material resources into cloud computing.

In the help of cloud computing services, user put data into the cloud. Then he will not be able to control the data directly. One of the most concerns of the user, then, is the security of data. In fact, the security of cloud computing and service level do still exist some problems. Security accidents of cloud storage appear frequently. There is large-scale leak of user data of Google mail in February 2009 and March 2011. A large number of the servers in Amazon cloud data center crashed in April 2011. In addition, cloud service providers may act unfavorably to the security of the user data, such as deleting the data which has not been accessed for a long term to save costs, concealing the attacks, system failure or mismanagement and other data loss caused by events in order to maintain their own reputation, intentional tampering or leaking user data for some interest.

Many cloud data storage security solutions are proposed recently, including traditional cryptography schemes, Provable Data Possession (PDP), Proofs of Retrievability (POR), Third Party Audit(TPA), etc.

The traditional cryptography schemes include digital signature, message authentication code(MAC), etc. However, it needs to return a large amount of data during the process of data integrity detection in these solutions，and also brings a lot of computation cost. At present, many researchers have devoted to the improvement of traditional cryptography technology in cloud computing. E.g., Cloud Storage Data Integrity Verification (CS - DIV) Agreement [2] which was proposed Cao Xi is based on X.509 public key authentication framework, and it refers to the integrity protection scheme of third party authentication protocol.

Provable Data Possession(PDP) Model is used for users to verify whether the remote server do have the stored data. PDP, which is proposed by Ateniese et al.[3] for the first time in 2007, is a kind of validation interaction process. It allows users to store data on untrusted servers, while it does not need to retrieve all the data when verifying the server. MR-PDP [4], proposed by Curtmola et al, extends PDP system to multiple servers. It improves the availability and protects the integrity of data by producing data copies.

The concept of model POR was introduced formally by Juels and Kaliski in 2007[5] (brief for JK program). The POR model is that the storage services prove to the user that the data is kept intact on the server and to ensure that users can restore and use of these data, which means that users can verify the integrity of the data and extract the data when authentication is successful. In December 2008, Shacham and Waters improved the original POR model and put forward C-POR model[6](also called SW model). The program uses the same state certification (homomorphic authentication) to reduce the communication overhead, and inquired about unlimited number of ask. In November and December 2009, Bowers et al proposed a more practical significance POR model and HAIL(High-Availability and Integrity Layer)model that combines the JK and SW model[7, 8]. This model improves the POR system to a multi-server environment. C.Wang et al combines the C-POR model and Merkle hash tree(MHT)technology, and proposed a dynamic POR model cost O(log n)[9].

Third Party Audit (TPA) is a trusted third party to ensure the integrity of user data. Shah et al in HP proposed a new audit program which give the user's inspection task to a trusted third party to complete[10]. But this inevitably will increase the cost of storage service provider, and faced the risk of disclosure of information. C.Wang et al first proposed a TPA model in 2009, and then put forward several improved versions. In this model TPA audit the public data stored on the cloud server to verify the integrity of outsourced data. It uses the homomorphism certification and random masking technology to protect the integrity and confidentiality of data, support effective multi-audit task. What's more, it could achieve multi-user demand by introducing the bilinear polymerization signature, i.e. batch audit.

In this paper, we propose an authentication scheme of cloud storage security based on third party audit. This model could meet the user's demand of data integrity, data confidentiality, data recovery and extraction and TPA

credibility control requirements. The trustiness of TPA is also guaranteed by a mutual authentication protocol. Finally performance evaluation is given.

The rest of the paper is organized as follows. Section 2 provides the framework and process of cloud storage retrievability scheme. Section 3 presents the credibility control of TPA. Performance evaluations are given in Section 4. Finally section 5 gives the conclusion.

II. CLOUD STORAGE RETRIEVABILITY SCHEME

We proposed an authentication scheme of data storage security of cloud computing environment base on POR model and TPA.

A. The Framework

The framework is shown in figure1. There are three entities in this framework: CS(Cloud Server), client and TPA(Third Party Auditor).

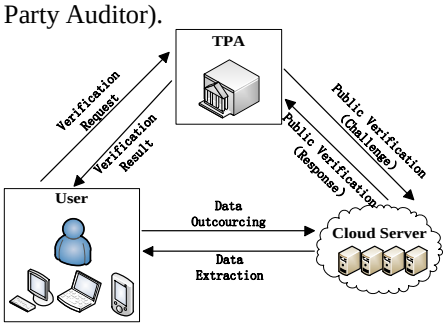

Figure 1. The authentication architecture base on TPA.

Client may be individuals or organizations. CS is usually managed by CSP, and it has a huge storage devices and computing resources. It can also provide availability and shared services such as share data on CS to an authorized visitors, beyond the data storage services.

TPA is used to perform data authentication and auditing tasks on behalf of the client. When doing this, the client needn't to do the auditing tasks itself, which is very important to reduce the costs of cloud computing. What's more, the TPA has the expertise and capability which a Client usually has not. So client can trust the evaluation of cloud storage service and warning against security threats provided by TPA.

Of course, client wants to enjoy the service provided by TPA, but don't want their privacy data leaked to TPA. By using the combination of public certification, homomorphic certification and random conceal program, TPA can audit and verify the data stored in the cloud without a local backup. This public audit system allowed the client to initialize their privacy parameters and transmit the certification and metadata to TPA in the initial set-up phase. And then the TPA verifies the validity of proof value generated by the server. Above all, this approach can protect the integrity and confidentiality of data in the cloud server.

In addition, this scheme supports multiple servers. The original POR model and PDP model are based only on single server. But the cloud computing is composed of massive servers, so a multiple server environment is necessary. It's found that a multiple server support can improve the efficiency, strengthen the safety certification of security. We can use redundancy technology to support multiple servers in cloud storage, including copies of redundancy and the redundancy coding. The redundancy coding means that to encode with different data in every server. In this paper dispersal code program is adopted. It can also reduce the overhead of server, improve efficiency of certification and increased the security of data.

B. Process of the Scheme

This section will show us how to design a basic integrity certification program, including three phase: setup phase, verification phase and extract phase, which is show in figure 2. Our process mainly from Wang Cong's work[[9, 11]]. The interested readers can refer to them.

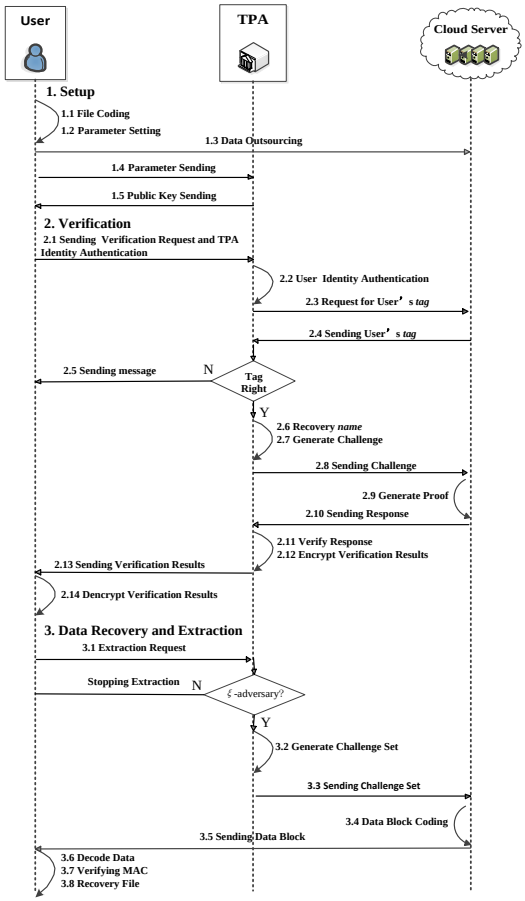

Figure 2. The process of our scheme

1) Setup.

In this phase, there are three steps, file coding, parameter setting and data outsourcing.

a) File Coding.

In File Coding phase, the user firstly needs to generate some keys for coding using *keygen* function, and then call encode function to block and code the file.

Key Generation: generating three keys by the keygen function. Firstly, (n-l) keys for Dispersal-Code are generated,

which are $\{\kappa_j, \kappa_j'\}_{j=[l+1,n]}$. κ_j and κ_j' are keys respectively for UHF(general hash function) and PRF(pseudo-random function). Keys for Server–Code and MAC are then generated.

File Blocking: chunking the file F into l segments, and then sent the file segments $F^{(j)}$ to the primary server j, and $j=[1,l]$. Then it will get a matrix $\{F_{ij}\}_{i=[1,m_F],j=[1,l]}$, and $m_F = |F|/l$ is the number of blocks in each file segment.

Server-Code: encoding the file segments $F^{(j)}$ in the server j using server code, then it will get a segment, the length of which is m. And the data blocks $m_F+1,\cdots,m$ are parity blocks.

Dispersal-Code: encoding the rows of the matrix got from the last step using dispersal code ECCd. $F^{(l+1)},...,F^{(n)}$ will be the results.

And after dispersal code encoding, the matrix of the entire file can be defined as: $F^d = \{F_{ij}^d\}_{i=[1,m],j=[1,n]}$, which is the same as the original file when $i=[1,m_F]$, $j=[1,l]$, $F_{ij}^d = F_{ij} \in I$. $\{F_{ij}^d\}_{i=[m_F+1,m],j=[1,l]}$ is parity blocks generated by the server code.

b) Parameter Setting.

Key Generation: generating a pair of signature keys(spk, ssk) by calling the keygen function, and generating $x \leftarrow Z_p$, $u \leftarrow G_1$, then calculating $v \leftarrow g^x$. As a result, secret keys are $sk=(x,ssk)$, public keys are $pk=(spk,v,g,u,e(u,v))$, where G1 is a multiplication cyclic group, g is the generator of G1, e is a bilinear mapping.

Signature Generation: generating the signatures of the coded data $F^d = \{F_{ij}^d\}_{i=[1,m],j=[1,n]}$ file by calling the siggen function,

$$\sigma_{ij} \leftarrow (H(W_{ij}) \cdot u^{F_{ij}^d})^x \in G_1 \quad (1)$$

Where $W_{ij} = name \| i \| j$. Name is selected from a random value by the user, which is used as the file identifier. So the signature set is $\Phi = \{\sigma_{ij}\}_{i\in[1,m],j\in[1,n]}$.

File Tag Generation: In order to ensure that the integrity of the file identifier name, users need to calculate $tag = name \| Sig_{ssk}(name)$ as the file tag for F, where $Sig_{ssk}(name)$ is to sign name using the secret key ssk.

c) Data Outsourcing

In this step, the users need to send the data to cloud servers and TPA, respectively. In particular:

Sending $\{tag, F^d, \mathrm{MAC}_{k_{MAC}^{file}}(F^d), \Phi\}_{i=[1,m],j=[1,n]}$ to the cloud server, and then delete the local copy of the file.

Sending the public key $pk=(spk,v,g,u,e(u,v))$ to TPA, which will be used in the later phases.

In the end, users need to get *tpk*, that is the public key of TPA

2) Verification

Verification phase is mainly for TPA to process the user's request, and send challenge to the cloud server, receive the corresponding response, finally carry on the verification of the data integrity. This phase can be divided into four steps: user request for verification, TPA generate challenge, server generate proof and TPA verify response.

a) User Request for Verification

When sending TPA verification request, users need to sign the message by its own secret key ssk, then encode it with by TPA's public key *tpk*. The role of this step is to ensure that only trustworthy TPA by users can perform data authentication. It is because only the target TPA can get the user's verification request message. Furthermore, TPA can determine the source of the request message.

b) TPA Generate Challenge

TPA generates the value of challenge by calling the *challenge* function.

TPA firstly need to decode verification request message from the user by its own secret key tsk, then decode it by the user's public key *spk*, Then TPA can authenticate the user's identity.

TPA gets file tag from the server and authenticate it using the user's public key *spk*. If the authentication is successful, recover the name, otherwise output FALSE. Note that before TPA outputs the message, it needs to encode the message using its own secret key tsk, obtaining $\mathrm{Sig}_{tsk}(\text{"FALSE"})$ in order to ensure the source of this message.

To generate challenge C, the user needs to randomly select a subset $I=\{s_1,\cdots,s_c\}$, from the set $[1,n]$. Then randomly selects an element from Z_p for each of the selected elements, that is $v_i \leftarrow Z_p$. Thus a challenge $C=\{(i,v_i)\}_{i\in I}$ has generated. TPA sends this challenge C to each server.

c) Server Generate Proof

In this step, the server generates the proof value by calling the *respond* function. Every server *k* will randomly select an element $r_k \leftarrow Z_p$ by PRF, calculating the masked value $R_k^{mask} = e(u,v)^{r_k} \in G_T$. The role of the masked value is to protect the privacy information of the homomorphism authentication, namely, to protect the confidentiality of the data. Definition $\mu'_k = \sum_{i\in I} v_i F_{ik}^d$, using r_k to mask μ'_k, obtaining $\mu_k = r_k + \gamma\mu'_k$, where $\gamma = h(R_k^{mask}) \in Z_p$. Then calculate the value of the proof for each server *k*.

$$\mu_k = r_k + h(R_k^{mask}) \sum_{i=s_1}^{s_c} v_i F_{ik}^d \in Z_p \quad (2)$$

$$\sigma_k = \prod_{i=s_1}^{s_c} \sigma_{ik}^{v_i} \in G_1 \quad (3)$$

Each server sends the proof $P_k = \{\mu_k, \sigma_k, R_k^{mask}\}$ to TPA, respectively.

d) TPA verify response

In this step, by calling the *verify* function, TPA determines whether the data block is damaged for each server *k* respectively.

The following verification equation is generated by revising formula (1) in paper[11] :

$$R_k^{mask} \cdot e(\sigma_k{}^{\gamma}, g) \stackrel{?}{=} e((\prod_{i=s_1}^{s_c} H(W_{ik})^{v_i})^{\gamma} \cdot u^{\mu_k}, v) \qquad (4)$$

III. CREDIBILITY CONTROL OF TPA

TPA is a scheme that use a third party to audit and guarantee the integrity of Client's data. It will save the time and resource of the Client when introducing the TPA into cloud computing. It's also valuable to the scale of cloud computing economy. TPA is usually considered a credible third party. But indeed, it may deceive users for personal interests, or the authentication requests sent from user to TPA may be intercepted by a malicious third party. In addition, TPA must authenticate the user's identity to ensure the authenticity of the user's information when received a request from the user. These issues must be effectively solved before the TPA technology is widely used in cloud computing.

In our scheme, we control the credibility of TPA from three aspects.

Firstly, the user's sensitive data can not be leaked to TPA, because TPA may leaks user data to an unauthorized or even malicious third party for personal interest. This scheme uses random masking technology based on the same state certification. A masking code(r_k) is used to protect the sensitive information of user(μ'_k): $\mu_k = r_k + \gamma\mu'_k$ ($\mu_k = r_k + \gamma\mu'_k$ is a linear combination of the user's sensitive data). Because the presence of random values, the linear equations cannot be solved no matter how many linear combinations are collected. This method is equivalent to use random masking to protect the same state certification, the user data is also been protected.

Secondly, a mutual authentication between user and TPA is been used in this scheme. The mutual authentication is achieved by using public key cryptography technology. User obtains the public key of TPA(tpk) when performs the audit outsourcing. When it needs to send information to TPA, it will first sign the message with its own private key(ssk), and then encrypts the message with tpk. When TPA gets the message, first it must decrypt the message with his own public key (tsk), and then decrypt the message with user's public key(spk). After that, TPA gets the message sent by the user and can ensure the sources of this request information. On the other hand, such a certification process can ensure that only the credible target TPA can decrypt the message, because only the TPA has a corresponding private key(tsk). When TPA sends a message to user, it'll sequentially encrypt the message whit tsk and spk. The user will sequentially decrypt the message with ssk and tpk when receive it. When doing this, user can ensure the resource of information received, and information from TPA won't leak a malicious third party. This method adds a litter overhead but gains a higher security, it's a desirable scheme. The identity authentication between user and TPA is shown in figure 3.

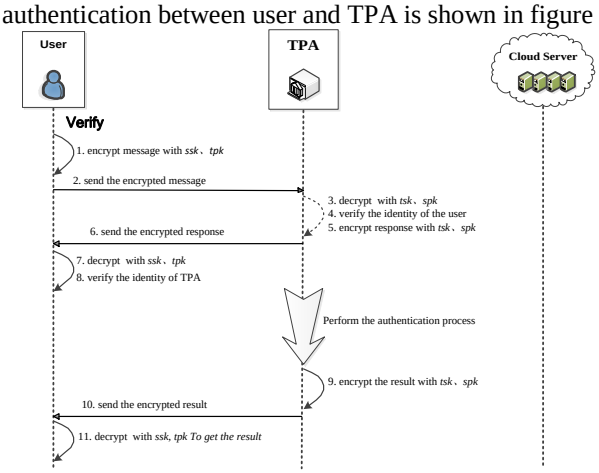

Figure 3. Identity authentication of user and TPA.

Thirdly, user needs to confirm the certified behavior of TPA. User delegates the authentication work to TPA, but TPA may not doing so exactly to reduce cost. For example, TPA will reduce the number of times of certified. To solve this problem, a parameter on the client is suggested in this paper to record the challenging time the user needs TPA to perform, denoted as T_{cli} . As the same, parameter both on TPA and CS are set to record the challenging time the TPA send to CS, which is denoted T_{tpa} and T_{cs} . So whenever the user wanted, it can send request to TPA and CS to obtain the challenging time and compare it. By doing this it will know whether TPA has done exactly as the user wants. It also means that TPA cheats user if $T_{tpa} \neq T_{cli}$ (suppose that CS has record the truly information of authentication), as is shown in figure 4.

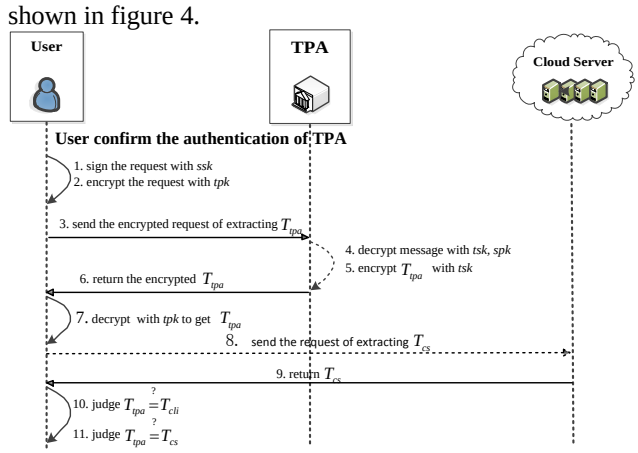

Figure 4. User confirm the authentication of TPA .

IV. PERFORMANCE EVALUATION

In this section, we will carry on performance evaluation in communication cost and storage cost.

A. Communication Cost

The elliptic curve used in this paper is MNT curve, while sampling parameter is A-bilinear mapping. Here we set the security level as 160 bits, the order p of multiplication cyclic group G_1, G_2 and G_t as 160 bits, the size of base domain as 159, namely $|q|$=159, and the embedded level as 2, then we will get $|v_i| = 80$, $|p|$=160.

What system algorithm this solution mainly focuses on is challenging challenge algorithm, algorithm response respond algorithm and authentication verify algorithm, which are the key parts of the solution and are used so frequently to mainly affect the performance of the solution.

The main influence factor is that TPA sends a challenge value to the server and the return value $P = \{\mu, \sigma, R^{mask}\}$ of the server back to the TPA in communication cost.

Therefore, the main communication cost in this solution is $c \cdot (\log_2(n) + |v_i|) + |p| + 2|q|$.

Finally, communication complexity can be considered as O(log n). Here, the length of the proof value sent by the server is $|p| + 2|q|$, and it is a constant, independent of the size of the data block.

B. Storage Cost

In data security authentication solutions, the extra storage space for authentication behavior is known as additional storage, namely the part that the required storage space is bigger than the original file after pretreatment.

The extra storage cost needed on TPA terminal comes mainly from the storage of the user's public key $pk = (spk, v, g, u, e(u, v))$ in the basic authentication solution proposed in this paper. Because the size of pk is independent of the user's files, the fixed storage cost will not become the restriction factor of the whole scheme with large amounts of data.

The server terminal (CS) needs additional storage costs mainly including the signature collection Φ of the data blocks, the MAC value of documents and the storage space of file's tag, where the signature collection is $\Phi = \{\sigma_{ij}\}_{i \in [1,m], j \in [1,n]}$, file MAC is $\mathrm{MAC}_{k^{file}_{MAC}}(F^d)$, the file's tag $tag = name \| Sig_{ssk}(name)$, whose size is independent of the size of the original file, and related to the size of the data block. Therefore, it can be considered that the additional storage needed by the server for safety certification has nothing to do with the size of the original file. For the same file, the greater the size of the divided data block, the smaller the extra storage cost. But the above analysis shows that the size of the data block will affect the computational cost.

V. Conclusion

The research of Cloud storage security is still in the start-up phase. This paper put forward a scheme for data storage security verification based on third party in Cloud Computing to meet demand of user in data integrity, data confidentiality, data recovery and retrieval, credibility control of TPA and so on.

Finally, it's still too early to use the scheme in this paper for actual Cloud Computing products. So the future research must be about that how theoretical research can be applied to the Cloud computing.

Acknowledgment

This work is supported by the Key Lab of Information Network Security, the Ministry of Public Security, Information Security Special fund of National Development and Reform Commission (Project name: Development of Security Test Service Capabilities in Wireless Intelligent Terminals) and the National High Technology Research and Development Program of China (863 program) under grant 2013AA014001.

References

[1] D. Zissis,D. Lekkas. Addressing cloud computing security issues[J]. Future Generation Computer Systems-the International Journal of Grid Computing and Escience, 2012, 28(3): 583-592.

[2] Cao Xi, Xu Li,Chen Lanxiang. Data integrity verification protocol in cloud storage system[J]. Computer Application, 2012(01): 8-12.

[3] Ateniese Giuseppe, Burns Randal, Curtmola Reza, et al. Provable data possession at untrusted stores. in CCS '072007, ACM: Alexandria, Virginia, USA

[4] Ateniese Giuseppe, Di Pietro Roberto, Luigi V Mancini, et al. Scalable and efficient provable data possession. in SecureComm '08 2008, ACM: Istanbul, Turkey

[5] Curtmola Reza, Khan Osama, Burns Randal, et al. MR-PDP : Multiple-Replica Provable Data Possession. in ICDCS '082008, IEEE Computer Society. 411 - 420.

[6] A. Juels,B. S. Kaliski. PORs: Proofs of Retrievability for Large Files[J]. Ccs'07: Proceedings of the 14th Acm Conference on Computer and Communications Security, 2007584-597.

[7] Shacham Hovav,Waters Brent. Compact Proofs of Retrievability. in the Theory and Application of Cryptology and Information Security2008, Springer-Verlag: Melbourne, Australia

[8] Kevin D Bowers, Juels Ari,Oprea Alina. Proofs of retrievability : theory and implementation. in CCSW '09 2009, ACM: Chicago, Illinois, USA

[9] K. D. Bowers, A. Juels,A. Oprea. HAIL: A High-Availability and Integrity Layer for Cloud Storage[J]. Ccs'09: Proceedings of the 16th Acm Conference on Computer and Communications Security, 2009187-198.

[10] Q Wang, C Wang, K Ren, et al. Enabling Public Auditability and Data Dynamics for Storage Security in Cloud Computing[J]. Parallel and Distributed Systems, IEEE Transactions on, 2010, PP(99): 1.

[11] Mehul A Shah, Baker Mary, Jeffrey C Mogul, et al. Auditing to keep online storage services honest. in HOTOS'072007, USENIX Association: San Diego, CA

[12] Cong Wang, Sherman S. M. Chow, Qian Wang, et al. Privacy-preserving public auditing for secure cloud storage[J]. IEEE Transactions on Computers, 2013, 62(2): 362-375.