Public Auditing for Ensuring Cloud Data Storage Security With Zero Knowledge Privacy

Wang Shao-hui[1,2*] , Chang Su-qin[1], Chen Dan-wei[1], Wang Zhi-wei[1]
1. College of Computer, Nanjing University of Posts and Telecommunications, Nanjing 210046, China;
2 Network and Data Security Key Laboratory of Sichuan Province
Email: wangshaohui@njupt.edu.cn

Abstract

In cloud storage service, clients upload their data together with authentication information to cloud storage server. To ensure the availability and integrity of clients' stored data, cloud server(CS) must prove to a verifier that he is actually storing all of the client's data unchanged. And, enabling public auditability for cloud storage is of critical importance to users with constrained computing resources, who can resort to a third party auditor (TPA) to check the integrity of outsourced data. However, most of the existing proofs of retrievability schemes or proof of data possession schemes do not consider data privacy problem. Zero knowledge privacy requires TPA or the adversary can not deduce any information of the file data from auditing system. In this paper, after giving a new construction of a recently proposed cryptographic primitive named aggregatable signature based broadcast (ASBB) encryption scheme, we present an efficient public auditing scheme with zero knowledge privacy. The new scheme is as efficient as the scheme presented by Shacham and Waters without considering privacy and is secure in the random oracle model.

Keywords*: Cloud Computing, Cloud Storage, Public Auditing, Zero-Knowledge Privacy, Integrity*

1. Introduction

Recently, cloud computing is receiving more and more attentions, from both industrial and academic community. Cloud computing separates usage of IT resources from their management and maintenance, so that users can focus on their core business and leave the expensive maintenance of IT services to cloud service provider. However users of outsourced storage are at the mercy of their storage providers for the continued availability of their data. Even Amazon's S3, the best-known storage service, has experienced significant downtime. Here we are considering scenarios where users may have concerns of the integrity and privacy of their data stored in the cloud storage.

As users no longer physically possess the storage of their data, traditional cryptographic primitives for the purpose of data security protection cannot be directly adopted. In particular, simply downloading all the data for its integrity verification is not a practical solution due to the expensiveness in I/O and transmission cost across the network.

In order to solve remote integrity checking problem in cloud storage, a lot of works[1-16] have been done focusing on various conditions of application and attempting to achieve different goals. Among these works, the methods can be divided as Proof of Data Possession(PDP) and Proofs of Retrievability (PoR). PDP scheme, first presented by Ateniese et al. [2,6] are related protocols that only detect a large amount of corruption in outsourced data. Their scheme utilizes the RSA-based homomorphic authenticators for auditing outsourced data and suggests randomly sampling a few blocks of the file. However, the schemes have to set a prior bound on the number of audits and doesn't support public audit ability. Public auditability allows an external party, in addition to the user himself, to verify the correctness of remotely stored data. While PoR scheme[3], first presented by Juels, is a challenge-response protocol that enables a cloud provider to demonstrate to a client that a file is retrievable, i.e., recoverable without any loss or corruption. Their scheme use spot-checking and error-correcting codes to ensure both "possession" and "retrievability" of remote data files. Erway et al.[8] was the first to propose dynamic PDP scheme. They developed a skip lists based method to enable provable data possession with dynamic support. however, the efficiency of their scheme remains in question. In [9], Wang et al. provided a dynamic architecture for public checking.

However, most of these schemes [2,4,13] do not consider the privacy protection of users' data against external auditors. Indeed, cloud service provider may potentially reveal users' data to auditors or adversaries during the auditing. From the perspective of protecting data privacy, this severe drawback greatly affects the security of these protocols in Cloud Computing[15]. Recently Wang et al. [16] presented two privacy-preserving public auditing schemes for cloud storage systems based on Shacham and Waters' scheme[4]. In their first scheme, the adversary can not deduce the data from the auditing system if the data have high entropy, but it is not secure if the data stored have low entropy, for the adversary can have a brute-force guess of the message offline. The second inefficient scheme they presented can provide zero knowledge privacy with much less efficiency than [4], which means the adversary has perfect zero knowledge from the auditing system.

In this paper, we tackle the problem of zero knowledge privacy-preserving public auditing problem for cloud storage system. To achieve the zero knowledge privacy, we first present a new construction of a recently proposed cryptographic primitive called aggregatable signature based broadcast (*ASBB* for short) encryption scheme[17]. Based on this new *ASBB* scheme, we propose a short, efficient homomorphic public verifiable scheme with zero knowledge privacy. The new scheme is almost as efficient as the original Shacham and Waters' scheme[4] and it is secure in the random oracle model.

The rest of the paper is organized as follows. In Section 2, we present the preliminaries and the building blocks related to this paper; The definition and security requirements of public auditing scheme for cloud storage are revisited in section 3; In section 4, we present the detailed description of our constructions. Section 5 gives the security and performance analysis of the new scheme and conclusion is given in section 6.

2. Notations and Building Blocks

In this section, we give a brief descriptions of corresponding preliminaries and building blocks including bilinear maps, computation hard problems, ASBB schemes and knowledge proof system.

2.1. Bilinear Map and Hard problems

Definition 2.1 Bilinear Map. Let G and G_T be multiplicative cyclic groups of prime order p. Let g be a generator of group G. A bilinear map is a map $e: G \times G \to G_T$ satisfying:

1. For any $u, v \in G$ and $a, b \in Z_p$, $e(u^a, v^b) = e(u,v)^{ab}$. This bilinearity implies that for any $u_1, u_2 \in G$, $e(u_1 \cdot u_2, v) = e(u_1, v)e(u_2, v)$.

2. There exists an efficiently computable algorithm for computing e.

3. The map should be non-trivial, i.e., e is non-degenerate: $e(g,g) \neq 1$.

Some hard problem assumptions related to this paper are presented as follows:

Computational Diffie-Hellman (CDH) Assumption: Given g, g^a, g^b for unknown $a, b \in Z_p^*$, it is hard to compute g^{ab}.

Decisional Bilinear Diffie-Hellman (DBDH) Assumptions: Given g, g^a, g^b, g^c for unknown $a, b, c \in Z_p^*$, it is hard to distinguish the value $T = e(g,g)^{abc}$ with random number $Z \in G_T$.

Bilinear Pairing Assumption: Given G, its generator g, and the value of $e(X, g) \in G_T$, it is hard to compute $X \in G$.

Bilinear Diffie-Hellman Assumption: Given g, g^a, g^b for unknown $a, b \in Z_p^*$, for any random $h \neq g \in G$, it is hard to compute $e(h,g)^{ab}$.

2.2. Aggregatable Signature-Based Broadcast Encryption

As to aggregatable Signature-Based Broadcast(*ASBB*) Encryption scheme[17], the public key can be simultaneously used to verify signatures and encrypt messages, and any valid signature can be used to decrypt ciphertexts under this public key. In [17], ASBB scheme is presented to design one round asymmetric group key agreement.

Most significance of ASBB scheme is that it has key-homomorphic property, which means that, given two signatures on the same message under two secret keys, one can efficiently produce a signature of the same message under a new secret key derived from the original two keys. The security of an ASBB scheme incorporates the standard notion of security for a signature scheme, i.e., existential unforgeability under the chosen message attack (EUF-CMA)[18] and the security as an encryption scheme. In [17], Wu et al. presented an efficient ASBB scheme based on bilinear pairings, and the description of the scheme is depicted below:

– Public parameters: Let $(p, G, G_T, e) \leftarrow$ PairGen (1^λ), and g is the generator of G. Let $H:\{0,1\}^* \to G$ be a cryptographic hash function. The system parameters are (g, H, p, G, G_T, e).

– Public/secret keys: Select at random $r \in Z_p^*$, $X \in G \setminus \{1\}$. Compute $R = g^{-r}, A = e(X, g)$. The public key is $pk = (R, A)$ and the secret key is $sk = (r, X)$.

– Sign: The signature of any string $s \in \{0,1\}^*$ under the public key (R, A) is $\sigma = XH(s)^r$.

– Verify: Given a message-signature pair (s, σ), the verification equation is $e(\sigma, g)e(H(s), R) = A$. If the equation holds, output 1 to represent that σ is a valid signature. Otherwise output 0 and reject the signature.

– Encryption: For a plaintext $m \in G_T$, randomly select $t \in Z_p^*$ and compute $c_1 = g^t$, $c_2 = R^t$, $c_3 = mA^t$. The ciphertext is (c_1, c_2, c_3).

– Decryption: After receiving a ciphertext (c_1, c_2, c_3), anyone with a valid message-signature pair (s, σ) can extract the plaintext as follow: $m = c_3 / (e(\sigma, c_1)e(H(s), c_2))$

2.3. Knowledge Proof for Equality of Discrete Logarithm

In cryptography, knowledge proof is an interactive proof in which the prover succeeds in convincing a verifier that he knows something. The question related to the paper is how to prove knowledge that two public data
have the same discrete logarithm without revealing any other information about this value.

Let G and G_T be multiplicative cyclic groups of prime order p. Let g be generators of G and random value $A \in G_T$. Given two public data $h \in G$ and $R \in G_T$, the prover should prove that $\log_g h = \log_A R = x$, but he can not leak any information of x. We adopt the knowledge proof scheme presented in [19], which is given as follows:

1. The prover chooses randomly $s \in Z_p^*$ and computes:

$$(a, b) = (g^s, A^s),\ c = H(a, b),\ r = s + cx$$

where $H(\cdot)$ is a secure hash function. Prover sends a, b, r to the verifier.

2. The verifier first computes $c = H(a, b)$, and accepts the proof if $g^r = ah^c$, $A^r = bR^c$.

3. System Architecture and Security Model

In this section, we give the definitions of cloud storage public auditing scheme and the corresponding security requirements including completeness, soundness and privacy.

We used the basic cloud system architecture which is given in [16]. The cloud data storage service involves three different entities, as illustrated in Fig. 1: the cloud user, who has the potential data files to be stored in the cloud; the cloud server (CS), which is managed by the cloud service provider (CSP) to provide data storage service; the third party auditor (TPA), who is trusted to assess the cloud storage service reliability on behalf of the cloud user upon request.

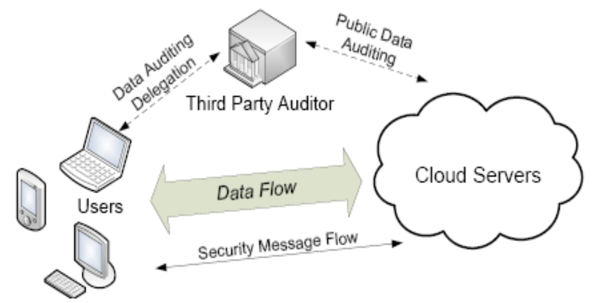

Fig. 1: The architecture of cloud data storage service

Definition 3.1 Public Auditing for Cloud Storage. A public auditing scheme for cloud storage is defined through three algorithms: $Keygen$, $Gentag$ and $Audit$, which behave as below:

− $Keygen(1^{\lambda})$. Given security parameter λ as input, this randomized algorithm generates scheme's public parameters and cloud users' public/private key pair (pk, sk).

− $Gentag(sk, F)$. The randomized algorithm takes user's secret key sk and data file $F \in \{0,1\}^{*}$ as inputs, and produces the authentication tags t, which contains information on the file being stored and additional secret information encrypted under the secret key sk. The file F and tag t will be stored in the CS.

− $Audit$. The randomized auditing algorithms can be defined as an interactive protocol $(CS(pk, F, t) \leftrightarrow TPA(pk))$ for proving file integrity. During protocol execution, TPA(verifier), taking the public key pk and some processed file description that is output by $Gentag$ as input, issues an auditing challenge to the cloud server. And the CS (prover) will derive a response message using file F stored and the file tag t as inputs. At the end of the protocol run, TPA outputs 1, which means that the file is stored unchanged on the cloud server; Otherwise outputs 0.

We assume TPA, who is in the business of auditing, is reliable and independent. While CS is presumed to be potentially untrustworthy. It may corrupt the file-system in a fully Byzantine manner. The cloud may alter or drop file-system operations transmitted by the portal; it may corrupt or erase files. We also assume that CS has no incentives to reveal their hosted data to external parties because of some regulations requirements. Therefore, the auditor or adversary will extract the outsourced data through the communication between CS and TPA.

The security requirements of the public auditing scheme for cloud storage including completeness, soundness and privacy. The detail definitions are given below:

Definition 3.2 Completeness. Completeness requires that, for all key pairs (pk, sk) output by $Keygen$, for all files $M \in \{0,1\}^{*}$ and tag t output by $Gentag(sk, M)$, the TPA will always output 1 when interacting with the valid CS via auditing algorithm:

$$prob(Audit(CS(pk, M, t) \leftrightarrow TPA(sk)) = 1) = 1$$

Definition 3.3 Soundness. A public auditing scheme is assumed to be sound if any cheating CS without storing the unchanged file M can not convince the TPA. We utilize the definition of soundness presented in [4] which is formulized by the following game between an adversary A and a challenger C:

Step1. The challenger generates key pair (pk, sk) by running algorithm $Keygen(1^{\lambda})$, and provides public key pk to A.

Step2. The adversary can now interact with the challenger for some oracle queries. It can make queries to $Gentag$ oracle, and for each query, the challenger chooses some file M and computes $t \leftarrow Gentag(sk, M)$. Both M and t are returned to the adversary. In addition, the adversary can undertake auditing executions with the challenger. In these protocol executions, the challenger plays the part of the verifier and the adversary plays the part of the prover.

Step3. The adversary generates and sends the challenger some data file M_1 and the challenger answers the corresponding tag t. Then the adversary changes the file to $M^{*} \neq M_1$ and undertakes auditing executions with the challenger: $Audit(C(pk) \leftrightarrow A(pk, M^*, t))$.

We say a cloud storage public auditing scheme is sound if the following probability is negligible:

$$prob(Audit(C(pk) \leftrightarrow A(pk, M^*, t)) = 1)$$

Definition 3.4 Zero-knowledge Privacy. Zero-knowledge privacy means the adversary can obtain zero knowledge information of the files data stored from the auditing scheme. We formalize the definition by the following game between the adversary A and the challenger C:

Step1 and Step2 are almost the same as the ones in the definition of soundness. The difference is in the protocol executions, the challenger plays the part of the prover, i.e. CS, and the adversary plays the part of the verifier, i.e. TPA.

3. Finally, as to any new file data M_1 which is unknown to the adversary, the challenger first produces the tag t; Then the adversary will undertake auditing executions with the challenger: $Audit(A(pk) \leftrightarrow C(pk, M_1, t))$.

We say a public auditing scheme is zero-knowledge privacy if for any function f on the file M_1, the following probability is negligible:

$$prob(A(pk): f(M_1)) - prob(A(pk) \leftrightarrow C(pk, M_1, t): f(M_1))$$

Here $prob(A(pk): f(M_1))$ means the probability that the adversary guesses the value of $f(M_1)$ successfully without any auditing procession; and the latter probability $prob(A(pk) \leftrightarrow C(pk, M_1, t): f(M_1))$ means adversary guesses $f(M_1)$ through the auditing scheme. So the zero knowledge privacy definition indicates the adversary can not get any more useful information from the auditing scheme to guess $f(M_1)$ successfully.

4. Our Constructions

In this section, we first give a new efficient construction of *ASBB* scheme that has aggregatable property; Then we present a public auditing scheme satisfying zero knowledge privacy based on the new *ASBB* scheme.

4.1. A New Construction of ASBB Scheme

The new proposed *ASBB* scheme is almost as efficient as the one presented in [17], and the detail description of the scheme is as follow:

− Public parameters: Let $(p, G, G_T, e) \leftarrow$ PairGen(1^λ), and g is the generator of G. Let $H: \{0,1\}^* \rightarrow G$ be a cryptographic hash function. The system parameters are (g, H, p, G, G_T, e).

− Public/Secret keys: Select a random number $r \in Z_p^*$, $X \in G \setminus \{1\}$. Compute $R = g^{-r}, A = e(X, g)$. The public key $pk = (R, A)$ and the secret key $sk = (r, X)$.

− Sign: To give the signature of any string $m \in Z_p^*$ under the public key pk, first choose randomly $s \in \{0,1\}^*$, compute $\sigma = X^m H(s)^r$, and the signature is (s, σ).

− Verify: Given a message-signature pair (m, s, σ), the verification equation is $e(\sigma, g)e(H(s), R) = A^m$. If the equation holds, output 1 to represent that signature is valid; Otherwise output 0 and reject the signature.

− Encryption: For any plaintext $\omega \in G_T$, select a random number $t \in Z_p^*$ and compute $c_1 = g^t$, $c_2 = R^t$, $c_3 = \omega A^t$. The ciphertext is (c_1, c_2, c_3).

− Decryption: Given the ciphertext (c_1, c_2, c_3), anyone with a valid message-signature pair (m, s, σ) can extract the plaintext as: $\omega = c_3 / (e(\sigma, c_1)e(H(s), c_2))^{m^{-1}}$.

We can reduce the security of the new *ASBB* scheme to the security of the scheme in [17]. Taken the signature scheme as an example, It is easy to see if the adversary can forge a valid signature (m, s, σ) in the new scheme with the public key (R, A), he can forge a valid signature $(s, \sigma^{m^{-1}})$ for the scheme

[17] with the public key $(R^{m^{-1}}, A)$. Using the same proof method in [17], we can conclude the following theorem:

Theorem 4.1. Let G be a bilinear group of prime order p, the following claims hold:

(1) The proposed *ASBB* scheme is aggregatable against non-adaptive chosen message attacks in the random oracle model assuming the decision BDHE assumption holds in G;

(2) The proposed *ASBB* scheme is existentially unforgeable under adaptive chosen-message attack and indistinguishable under chosen plaintext attack in the random oracle model under the CDH and DBDH assumptions.

4.2. Public Auditing Scheme with Zero-knowledge Privacy

In the following, we present the auditing schemes with zero-knowledge privacy utilizing the new proposed *ASBB* scheme. The method is easy to understand. Using the new *ASBB* scheme, TPA encrypts arbitrary message and sends the ciphertext as the challenge to CS, and CS can decrypt the ciphertext as the response only if the file stored is in good condition.

The detail description of the scheme is as follow:

1. $Keygen(1^{\lambda})$. Let G and G_T be multiplicative cyclic groups of prime order p, and $e: G \times G \to G_T$ be a bilinear map. Let g be a generator of G. $H(\cdot)$ is a secure map-to-point hash function: $\{0,1\}^* \to G$, which maps strings uniformly to G. The system parameters are (g, H, p, G, G_T, e). Cloud user select a random number $r \in Z_p^*$, $X \in G \setminus \{1\}$. Compute $R = g^{-r}$ and $A = e(X, g)$. The public key $pk = (R, A)$ and the secret key $sk = (r, X)$.

2. $Gentag(sk, F)$. Given the data file $F = \{m_i\}_{i=1,\ldots,n}$ and each $m_i \in Z_p^*$, the user computes the authenticator tag as $\sigma_i = X^{m_i} H(id \| i)^r \in G$ for each i, where id is chosen by the user uniformly at random from Z_p^* as the identifier of file F.

3. $Audit$. The interactive proof process between TPA and CS is proceeded as follows:

Step1. To generate the challenge message for the auditing, the TPA first picks randomly $t \in Z_p^*$, c-element subset $I = \{s_1, s_2, \ldots, s_c\} \in [1, n]$, and for each $i \in I$, TPA chooses a random number v_i. Then the TPA chooses a random number $m \in G_T$, and computes

$$c_1 = g^t,\ c_2 = R^t,\ c_3 = A^t,\ c = m \cdot e(\prod\nolimits_{i \in I} H(id \| i)^{v_i}, c_2).$$

In addition, using the method in [19], TPA must give a proof of knowledge that c_1 and c_3 have the equal discrete logarithms corresponding to g and A: $POK\{(g, A, c_1, c_3) : \log_g c_1 = \log_A c_3\}$

The final auditing challenge is $\{(i, v_i)_{i \in I}, c_1, c_3, c, POK\}$.

Step2. Upon receiving the challenge, CS first verifies whether the proof POK is valid. If it is not valid, the auditing fails; Otherwise CS computes:

$$\sigma = \prod\nolimits_{i \in I} \sigma_i^{v_i},\ \mu = \sum\nolimits_{i \in I} v_i m_i,\ B = c_3^{\mu} = e(X, g^t)^{\sum_{i \in I} v_i m_i}$$

Then CS decrypts the message c and sends the plaintext to TPA as the response: $m^* = (c \cdot e(\sigma, c_1)) / B$

Step3. After receiving the message m^*, TPA checks whether $m = m^*$. If they are equal, TPA accepts the proof; Otherwise CS does not pass the auditing proof.

The correctness of the above verification equation is elaborated as follows, we replace $H(id \| i)$ with H_i for convenience :

$$e(\sigma, c_1) \cdot c = e(\prod\nolimits_{i \in I} \sigma_i^{v_i}, g^t) \cdot m \cdot e(\prod\nolimits_{i \in I} H_i^{v_i}, g^{-rt}) = m \cdot e(\prod\nolimits_{i \in I} (X^{m_i} (H_i)^r)^{v_i}, g^t) \cdot e(\prod\nolimits_{i \in I} H_i^{v_i}, g^{-rt})$$

$$= m \cdot e(\prod\nolimits_{i \in I} X^{m_i v_i}, g^t) = m \cdot (e(X, g)^t)^{\sum_{i \in I} m_i v_i} = m \cdot B$$

5. Performance and Security

In this section, we give the performance and security analysis of our new zero-knowledge privacy preserving public auditing scheme, and we show that our scheme can provide soundness and zero-knowledge privacy requirements.

5.1. Performance Analysis

Because the schemes presented by Wang et.al.[16] are based on Shacham and Waters[4], we compare new proposed scheme with the original scheme [4] which does not consider privacy problem. The comparison consists in storage cost, communication cost and computation cost. To make the comparison convenient, we suppose in both schemes, CS stores the same file F and TPA chooses the same $I = \{s_1, s_2, ..., s_c\} \in [1, n]$, and v_i for each $i \in I$.

Storage Cost. The number of the authentication tag in both schemes is the same. The public key of scheme [4] has 2 group numbers and the secret key has 1 integer in Z_p^*; while the public key of our scheme has 2 group numbers and the secret key has 1 group number and 1 integer.

Communication Cost. In scheme [4], what the TPA sends is $\{(i, v_i)\}_{i\in I}$ and CS needs to send back 1 group numbers and 1 integer. In our scheme with zero knowledge privacy, TPA sends 3 extra group numbers $\{c_1, c_3, c\}$ and messages generated by POK (including 2 group numbers and 1 integer according to [19]) except for $\{(i, v_i)\}_{i\in I}$, and CS only needs to send back 1 group number, i.e. the plaintext.

TABLE I. COMPARISON OF NEW CONSTRUCTION WITH SCHEME IN [4]

		New Scheme	Scheme in [4]
Storage	PK	2G	2G
	SK	1G+1 I	1 I
Communication	TPA	$\{(i, v_i)\}_{i\in I}$ + 5G + 1 I	$\{(i, v_i)\}_{i\in I}$
	CS	1G	1G + 1 I
Computation	TPA	1 BM + 5GE (Off-Line)	2BM + 2GE
	CS	1 BM + 6 GE	1GE

Computation Cost. Here we only count the number of most expensive cost computation including bilinear map and group exponentiation. In scheme[4], the computation of TPA includes 2 bilinear map, 2 group exponentiation computation in the Step3 of auditing phase, and CS needs to compute 1 group exponentiation; while in our scheme, TPA needs to compute 1 bilinear map, 5 group exponentiation in the Step1 of the auditing phase, and CS should proceed 1 bilinear map and 6 group exponentiation computation.

The comparisons are listed in the table 1, where G and I mean group number and integer; GE and BM mean group exponentiation and bilinear map computation respectively. From the comparison, we can see new scheme has a little heavier communication overheads, and has more group exponentiation computation than scheme [4]. In fact, most of the group exponentiation computation of our scheme lies in the knowledge proof scheme POK .

However, we can see the computation of TPA in our scheme happens in Step1 to generate the challenge, so these computation can be pre-compute off-line which does not affect the auditing scheme. Thus the time consumed in our scheme only includes about 1 bilinear map and 6 group exponentiation computation. Because computing a bilinear map can be significantly slower than computing an exponentiation, we can conclude that our scheme is as efficient as Shacham and Waters' scheme without privacy consideration.

5.2. Security Analysis

We show the new proposed scheme can provide soundness and zero knowledge privacy from the following two theorems.

Theorem 5.1. If the signature scheme used for file tags is existentially unforgeable and the computational Diffie-Hellman problem and Bilinear Pairing assumption are hard, then if the cloud server does not possess the specific data intact as it is, he can not pass the audit phase with non-negligible probability.

Proof. After receiving TPA's challenge message $\{(i,v_i)_{i\in I}, c_1, c_3, c, POK\}$, CS has to compute $e(\prod_{i\in I} H(id \| i)^{v_i}, c_2)$ to decrypt the ciphertext c as the response. From the computational Diffie-Hellman assumption, we know it is hard to compute c_2 from c_1 and c_3. So CS can not deduce $e(\prod_{i\in I} H(id \| i)^{v_i}, c_2)$ directly.

Now we prove if the adversary can cheat the TPA in the auditing scheme, we can break the Bilinear Pairing assumption.

By the correctness of the scheme, we know the expected response (σ, B) that CS generates must satisfy:

$$B = e(X, g^t)^{\mu} = e(\sigma, c_1)\cdot e(\prod_{i\in I} H(id \| i)^{v_i}, c_2) \qquad (4)$$

If the file that CS stores has been changed, we know there exist $(\sigma^* \neq \sigma, B^*)$ that can pass the verification equation:

$$B^* = e(X, g^t)^{\mu'} = e(\sigma^*, c_1)\cdot e(\prod_{i\in I} H(id \| i)^{v_i}, c_2) \qquad (5)$$

It follows from the verification equation that $\mu \neq \mu'$, or it should contradict our assumption above. From equation (4) and (5), we can get:

$$B / B^* = e(X, g^t)^{\mu-\mu'} = e(\sigma/\sigma^*, c_1) = e(\sigma/\sigma^*, g^t)$$

which means $X^{\mu-\mu'} = \sigma/\sigma^*$, i.e. $X = (\sigma/\sigma^*)^{(\mu-\mu')^{-1}}$. This outcome contradict the Bilinear Pairing assumption. ■

Theorem 5.2. If the knowledge proof scheme for equality of discrete logarithm in [19] is correct, the new public auditing scheme can provide zero knowledge privacy.

Proof: As to any challenge $\{(i,v_i)_{i\in I}, c_1, c_3, c, POK\}$, messages c_1 and c_3 should have the same discrete logarithm corresponding to g and A because of the correctness of POK scheme. So the response CS sending back is:

$$c\cdot e(\sigma, c_1)/c_3^{\mu} = c\cdot e(\prod_{i\in I}(X^{m_i} H_i^{-r})^{v_i}, c_1)/c_3^{\mu} = c\cdot e(X, c_1)^{\mu} e(\prod_{i\in I}(H_i^{-r})^{v_i}, c_1)/c_3^{\mu}$$

where $\mu = \sum_{i\in I} v_i m_i$. It is easy to see if c_1 and c_3 have the same discrete logarithm corresponding to g and A, $e(X, c_1)^{\mu}$ must equal to c_3^{μ}, and the response can be simplified as $c\cdot e(\prod_{i\in I}(H_i^{-r})^{v_i}, c_1)$, which can be determined by the adversary alone. So the interactive auditing communication does not leak any information of the data file stored.

In fact, we can see any adversary can choose randomly $\{(i,v_i)\}_{i\in I}$, and picks random element $t \in Z_p^*$, and computes:

$$c_1 = g^t,\ c_2 = R^t,\ c_3 = A^t,\ c = m\cdot e(\prod_{i\in I} H(id \| i)^{v_i}, c_2)$$

The manuscript that the adversary outputs is $\{(i,v_i)_{i\in I}, c_1, c_3, c, m, POK\}$, which has the identical distribution of the real interaction between TPA and CS. That is to say, the adversary can not get any information from the auditing proof and he can output the manuscripts all by himself. ■

6. Conclusion

Security and integrity of data are the major concerns of client in the cloud storage network. In this paper, we tackle the privacy problem caused by the public auditing scheme. After presenting a new construction of *ASBB* scheme, we propose an efficient zero knowledge privacy preserving public auditing scheme for data storage security in cloud computing, i.e. the adversary can not deduce any information of the file stored through the auditing interaction between CS and TPA.

The new public auditing scheme not only eliminates the burden of cloud user from the tedious and possibly expensive auditing task, but also alleviates the users' fear of their outsourced data leakage. The new scheme has roughly the same efficiency as the Shacham and Waters' scheme without considering privacy problem.

In this paper, we only consider the situation that cloud users do not change data file stored; while in practice, cloud users may modify, insert, add or delete their outsourced data. How to construct zero-knowledge public auditing scheme for dynamic data storage will be considered in the future.

Acknowledgment

This work is supported by the Priority Academic Program Development of Jiangsu Higher Education Institutions (PAPD), National Natural Science Funds (Grant No.60903181) and Nanjing University of Posts and Telecommunications Funds (Grant No.NY208072).

References

[1] M. Einar, N. Maithili, T. Gene. Authentication and integrity in outsourced databases [J]. ACM Transactions On Storage. 2006, 2(2): 107-138.

[2] G. Ateniese, R. Burns, R. Curtmola, et al. Provable data possession at untrusted stores.Cryptology ePrint Archive, Report 2007/202, 2007. Online: http://eprint.iacr.org/. Version of 7 Dec. 2007; visited 10 Feb. 2008.

[3] A. Juels, B. Kaliski. Pors: proofs of retrievability for large files[C]. Proceedings of CCS 2007. Alexandria, VA, USA, 2007. 584-597.

[4] H. Shacham, B. Waters. Compact proofs of retrievability [C]. Proceedings of ASIACRYPT 2008. Melbourne, Australia, 2008. 90-107.

[5] F. Sebe, J. Domingo-Ferrer, A. Martinez-Balleste, et al. Efficient Remote Data Possession Checking in Critical Information Infrastructures. IEEE Trans. Knowledge and Data Eng., 2008. 1034-1038.

[6] G. Ateniese, R.D. Pietro, L.V. Mancini, et al. Scalable and efficient provable data possession [C]. Proceedings of the 4th international conference on security and privacy in Communication networks. Istanbul, Turkey: ACM, 2008: 90-99.

[7] C. Wang, Q. Wang, K. Ren, et al.. Ensuring data storage security in cloud computing[C]. Proceedings of IW QoS 2009, Charleston, South Carolina, USA, 2009.

[8] C. Erway, A. Kupcu, C. Papamanthou, et al. Dynamic provable data possession [EB/OL] Cryptology ePrint Archive, Report 2008/432, 2008.

[9] Q. Wang, C. Wang, J. Li, et al. Enabling public verifiability and data dynamics for storage security in cloud computing[C]. In Proc. of ESORICS'09, Saint Malo, France, Sep. 2009. Lecture Notes in Computer Science, 2009, Volume 5789/2009: 355-370.

[10] K. D. Bowers, A. Juels, A. Oprea. Proofs of retrievability: Theory and implementation[C]. In: Proc. of the 2009 ACM Workshop on Cloud Computing Security, CCSW 2009, Co-Located with the 16th ACM Computer and Communications Security Conf., CCS 2009. New York: Association for Computing Machinery, 2009. 43-54.

[11] S. Kamara, K. Lauter. Cryptographic Cloud Storage[C]. Lecture Notes in Computer Science. Financial Cryptography and Data Security. Berlin: Springer, 2010: 136-149.

[12] Z. Hao, N. Yu. A multiple-replica remote data possession checking protocol with public verifiability[A]. Proc. Second Int'1 Data, Privacy and E-Commerce Symp. (ISDPE '10), 2010.

[13] Q. Wang, C. Wang, K. Ren, W. Lou, and J. Li. Enabling public auditability and data dynamics for storage security in cloud computing[J]. IEEE Transactions on Parallel and Distributed Systems, vol. 22, no. 5, pp. 847–859, 2011.

[14] Q. Zheng and S. Xu. Fair and dynamic proofs of retrievability[C]. In Proc. 1st ACMM Conference on Data and Application Security and Privacy (CODASPY), 2011.

[15] M. A. Shah, M. Baker, J. C. Mogul, and R. Swaminathan. Auditing to keep online storage services honest[C]. In Proc. of HotOS'07, 2007, pp. 1-6.

[16] C. Wang, S.S.-M. Chow, Q. Wang, K. Ren and W.J. Lou. Privacy-Preserving Public Auditing for Secure Cloud Storage. http://eprint.iacr.org/2009/579.pdf.

[17] Wu, Q., Mu, Y., Susilo, W., Qin, B., Domingo-Ferrer, J.: Asymmetric Group Key Agreement[C]. In: Joux, A. (ed.) EUROCRYPT 2009. LNCS, vol. 5479, pp. 153-170. Springer, Heidelberg (2009).

[18] S. Goldwasser, S. Micali, R. Rivest. A Digital Signature Scheme Secure against Adaptive Chosen-message Attacks [J]. SIAM J. Computing 17(2), 281-308 (1988).

[19] D. Chaum, P.T. Pederson. Wallet databases with observers [C]. E.F. Brichell(Ed.): Advances in Cryptology- CRYPTO'92, LNCS 740, pp.89-105, 1993.