Public Proof of Retrievability Scheme against Active Attack in Cloud Storage

Jianhong Zhang, Wenjing Tang
Department of North China University of Technology of Shijingshan District, Bejing 100144, China
jhzhangs@163.com, tangwenjing2011@yeah.net

Abstract - Data integrity is one of the major concerns with cloud data storage for cloud user. Besides, the cloud user's constrained computing capabilities make the task of data integrity auditing expensive and even formidable. In recent years, many public data integrity verification schemes have been proposed, however, most of them are vulnerable to an efficient active attack, which means that the active adversary is able to arbitrarily modify the cloud data without being detected by the auditor. In this work, by utilizing blinding factor, we proposed a new proof of retrievability scheme, which can resist active attack and keep the privacy of the data in the auditing process. Extensive security analysis shows our proposed scheme is provably secure.

Index Terms - Cloud storage, proof of retrievability, active attack.

1. Introduction

By using cloud storage, people can remotely store their data and access them via networks at anytime and from anywhere. Despite the obvious benefits, it also brings new security challenges to the cloud data security. Data integrity and Confidentiality are two biggest concerns. So following the first scheme constructed in [1], a number of new data integrity verification schemes and improvements have been proposed, such as [2],[3],[4]. Most of them consist of five algorithms (*KeyGen, SigGen, Challenge, GenProof, VerifyProof). KeyGen* is a key generation algorithm that is run by the data owner. *SigGen* is used by the data owner to generate authentication tag for each data block m_i. *Challenge* is run by the TPA to produce the challenging message. *GenProof* is run by the cloud server to generate a proof of data integrity, while *VerifyProof* is run by the TPA to audit the proof. For the proof information generated in the *GenProof* algorithm is not provided authentication and it is in linear with m_i, most of the existing scheme are vulnerable to an efficient active attack. That is to say, an active adversary may corrupt or alter the data at his will after data owner stores its data file and the authentication tags in the cloud. Then only with the information how data are modified, the active adversary can change the wrong proof information to a valid one which can pass the *VerifyProof* algorithm.

In this paper, we propose a public proof of Retrievability scheme against active attack. Our work utilizes the technique of public key based homomorphic linear authenticator (or HLA for short) and constant size polynomial commitment technique. What's more, we introduce blinding factor into the *GenProof* algorithm, so that the active adversary cannot compute the valid proof information, that is to say, if the data file stored in the cloud is modified by an active adversary, there is on way to produce a valid proof information which can pass the *VerifyProof* algorithm except for the situation that the cloud server colludes with the active adversary, however, this is no good for the cloud sever, so we can assume this situation does not exist.

The rest of this paper is organized as follows. We provide the detailed description of our scheme in Section 2. Section 3 gives the Security analysis. Finally, We conclude our whole paper in section 4.

2. The Proposed Scheme

This section presents our PoR scheme which can resist active attack and keep the privacy of the data files in the auditing process. After introducing notations, we describe the construction and show the correctness of our scheme.

Let G and G_T be two multiplicative cyclic groups of the same prime order p. g and h are generators of G and $u \xleftarrow{R} G$. $e: G \times G \rightarrow G_T$ is a bilinear map. $H(\cdot)$ is a secure map-to-point hash function: $\{0,1\}^* \rightarrow Z_p^*$. F' is the erasure coded file consisting of n blocks, each of which has s elements: $\{m_{ij}\}, 1 \le i \le n, 0 \le j \le s-1$. $f_{\vec{a}(x)}$ is a polynomial with coefficient vector $\vec{a} = (a_0, a_1, \ldots, a_{s-1})$.

KeyGen: The TA chooses a random number $\alpha \xleftarrow{R} Z_p^*$ and generates the public keys for the system as $\{g^{\alpha^j}\}_{j=0}^{s-1}$. α is the master key of the system only known to the TA. Given a security parameter λ, the data owner generates a signing key-pair $((spk, ssk) \xleftarrow{R} Sign())$. The data owner also chooses a random number $\grave{o} \xleftarrow{R} Z_p^*$ and computes $v \leftarrow h^{蝌}, k \leftarrow h^{\alpha}$.
Then the cloud server chooses a random number $\xi \xleftarrow{R} Z_p^*$, and computes $\theta = v^{\xi}$ based on the public key v. So the public key, private key and master key are:

$$PK = \{p, k, v, \theta, spk, h, u, \{g^{\alpha^j}\}_{j=0}^{s-1}\}$$

$$SK_1 = \{\grave{o}, ssk\} \quad SK_2 = \{\xi\} \quad MK = \{\alpha\}.$$

SigGen: To outsource a file F, the data owner first obtain F' by applying erasure code. Then the owner randomly chooses a file name $name \in Z_p^*$ and generates the file tag τ

under ssk as $\tau \leftarrow name \,||\, n \,||\, sign_{ssk}(name \,||\, n)$. For each data block $m_i, 1 \le i \le n$, the owner produces an authentication tag as:

$$\sigma_i = (u^{H(name||i)} \cdot \prod_{j=0}^{s-1} g^{m_{ij}\alpha^j})^{\text{蝌}} = (u^{H(name||i)} \cdot g^{f_{\vec{\beta}_i}(\alpha)})$$

where $\vec{\beta}_i = \{m_{i,0}, m_{i,1}, \ldots, m_{i,s-1}\}$. The data owner stores F', file tag τ and corresponding authentication tags σ_i in the cloud.

Challenge: To verify the integrity of F', a user first gets the file tag τ from the cloud server and verifies the signature on τ with spk. If the signature is invalid, the user rejects and halts; otherwise, the user recovers file name $name$ and n. Then the user randomly chooses a $k-$elements subset K of $[1,n]$ and one random number $r \xleftarrow{R} Z_p^*$. Finally, the user produces the challenging message $CM = \{K, r\}$ and sends it to the cloud server.

GenProve: Based on the challenging message $CM = \{K, r\}$, the cloud server first computes $\{p_i = r^i \bmod p\}$ and $\{v_i = \xi \cdot p_i\}, i \in K$. Then the cloud server generates $y = f_{\vec{A}}(r)$, where $\vec{A} = \left\{\sum_{i\in I} v_i m_{i,0}, \cdots, \sum_{i \in I} v_i m_{i,s-1}\right\}$. As polynomials $f(x) \in Z[x]$ have the algebraic property that $(x-r)$ perfectly divides the polynomial $f(x) - f(r)$. The server divides the polynomial $f(x) - f(r)$ with $(x-r)$ using polynomial long division, and denotes the coefficients vector of the resulting quotient polynomial as $\vec{\omega} = (\omega_0, \omega_1, \ldots, \omega_{s-1})$, that is, $f_{\vec{\omega}}(x) \equiv \dfrac{f_{\vec{A}}(x) - f_{\vec{A}}(r)}{x-r}$. The cloud server generates

$$\psi = \prod_{j=0}^{s-1} (g^{\alpha^j})^{\omega_j} = g^{f_{\vec{\omega}}(\alpha)}.$$

Then cloud server computes $\sigma = \prod_{i \in I} \sigma_i^{v_i}$ and sends the proof information $Prf = \{\sigma, \psi, y\}$ to the user.

VerifyProof: On receiving the Prf, the user firstly computes $p_i = r^i \bmod p, i \in K$ and $\eta = u^{\sum_{i\in K} H(name||i) p_i}$. Based on η, the user verifies the integrity of F together with $Prf = \{\sigma, \psi, y\}$ as:

$$e(\eta, \theta) \cdot e(\psi, k \cdot v^{-r}) \overset{?}{=} e(\sigma, h) \cdot e(g^{-y}, v) \qquad (1)$$

If Eq.1 holds, the user outputs $AuditRst$ as accept; otherwise, outputs $AuditRst$ as reject.

Correctness: we analyse the correctness of our construction based on Eq.1 as:

$$\begin{aligned}
& e(\eta, \theta) \cdot e(\psi, k \cdot v^{-r}) \\
&= e(u^{\sum_{i\in K} H(name||i) p_i}, h^{\text{蝌}\xi}) \cdot e(g^{f_{\vec{\omega}}(\alpha)}, h^{(\alpha - r)}) \\
&= e(u, h^{\sum_{i\in K} H(name||i) p_i \text{ò}\cdot\xi}) \cdot e(g, h^{\frac{f_{\vec{A}}(\alpha) - f_{\vec{A}}(r)}{\alpha - r} \cdot (\alpha - r)\text{ò}}) \\
&= e(u^{\text{ò}\sum_{i\in K} H(name||i) p_i \,\xi}, h) \cdot e(g, h^{(f_{\vec{A}}(\alpha) - f_{\vec{A}}(r))\text{ò}}) \\
&= e(u^{\text{ò}\sum_{i\in K} H(name||i) v_i}, h) \cdot e(g^{\text{蝌}f_{\vec{A}}(\alpha) - f_{\vec{A}}(r)}, h) \\
&= e(u^{\text{ò}\sum_{i\in K} H(name||i) v_i} \cdot g^{\text{ò}f_{\vec{A}}(\alpha)}, h) \cdot e(g^{-f_{\vec{A}}(r)}, h^{\text{ò}}) \\
&= e(\sigma, h) \cdot e(g^{-y}, v)
\end{aligned}$$

3. Security Analysis

In this section, we start from giving the assumptions used in our scheme, then we evaluate the security of our scheme by analysing its fulfillment of the security guarantee, namely, the soundness, confidentiality and Active Attack Resistance property.

Definition 1. Discrete Logarithm Problem

Let $g, h \in G$, where G is a cyclic group. Given (g, h), it is computationally intractable to compute the value of x such that $h = g^x$.

Definition 2. Computational Diffie-Hellman Problem (CDH)

Let $x, y \xleftarrow{R} Z_p^*$. Given (g, g^x, g^y), it is computationally intractable to compute the value of g^{xy}, where G is a cyclic group of order p and g is a generator of G.

Definition 3. Static Diffie-Hellman Problem

Let $\alpha \xleftarrow{R} Z_p^*$. Given input as (g, g^α) and $h \in G$, where g is a generator of a cyclic group G of order p. it is computationally intractable to compute the value h^α.

Definition 4. t-Strong Diffie-Hellman (t-SDH) Problem

Let $\alpha \xleftarrow{R} Z_p^*$. Given input as a $(t+1)-tuple$ $(g, g^\alpha, \cdots, g^{\alpha^t}) \in G^{t+1}$, where g is the generator of a cyclic group G of order q. For any probabilistic polynomial time adversary, the probability $Pr[Adv(g, g^\alpha, \cdots, g^{\alpha^t}) = (c, g^{\frac{1}{\alpha + c}})]$ is negligible for any value of $\alpha \in Z_p^* / -\alpha$.

Definition 5. Knowledge of Exponent Assumption

Let $\alpha \xleftarrow{R} Z_p^*$. Given input as $(g, g^\alpha) \in G$, where g is a generator of a cyclic group G of order p. The only way to output a pair $(C, C^\alpha) \in G$ in polynomial time is to extract the exponent c, such that $g^c = C$.

A. Soundness Guarantee

We need to prove that the cloud server cannot generate valid proof information for the user without faithfully storing the data, as captured by *Theorem 1* and *Theorem 2.*

Theorem 1. If $g^{f_{\vec{A}}(\alpha)}$ can be forged by an existed probabilistic polynomial time adversary Adv, we can construct an algorithm B to efficiently compute the solution to the *t-SDH* problem based on the Adv.

Using the similar idea of [5], we prove Theorem 1 as:

Proof: Suppose there exists a probabilistic polynomial time adversary Adv that can generate $f_{\vec{A_1}}(\alpha)$ such that $g^{f_{\vec{A_1}}(\alpha)} = g^{f_{\vec{A}}(\alpha)}$, where $f_{\vec{A}(\alpha)}$ and $f_{\vec{A_1}(\alpha)}$ are known to the Adv. The Adv can construct another polynomial $f_{\vec{A_2}}(x) = f_{\vec{A}}(x) - f_{\vec{A_1}}(x)$. And we can get $g^{f_{\vec{A_2}}(\alpha)} = g^{f_{\vec{A}}(\alpha) - f_{\vec{A_1}}(\alpha)} = g^{f_{\vec{A}}(\alpha)} / g^{f_{\vec{A_1}}(\alpha)} = 1$, so $f_{\vec{A_2}}(\alpha) = 0$, which means α is a root of polynomial $f_{\vec{A_2}}(x)$. By factoring $f_{\vec{A_2}}(x)$, B can find α and easily find a number c to get $(c, g^{1/\alpha+c})$ as solution to the instance of the *t-SDH* problem given by the system parameters.

Theorem 2. If the signature scheme used for authentication tags is existentially unforgeable, the *CDH* problem, the Static Diffie-Hellman problem and the *t-SDH* problem are hard in bilinear groups, then, in the random oracle model, no adversary against the soundness of our public PoR scheme could cause verifier to accept in a proof-of-retrievability protocol instance with non-negligible probability, except by responding with correctly computed values Prf.

Proof: It is easy to prove that the signature scheme is existentially unforgeable based on the *CDH* problem. In concrete, if there is an adversary that can break our signature scheme, we show how to use this adversary to solve the *CDH* problem as follows: set $u = h^{x_0}, g = h^{x_1}$, the signature scheme used for authentication tags can be simulated as

$$\sigma_i = (u^{t_i} \cdot \prod_{j=0}^{s-1} g^{m_{ij}\alpha^j})^{\text{蝌}} = (h^{x_1 \cdot f_{\vec{\beta_i}}(\alpha) + x_0 ?t_i})$$

where $\vec{\beta_i} = \{m_{i,0}, m_{i,1}, \ldots, m_{i,s-1}\}$ and $t_i = H(name \| i)$.

Finally, if there is an adversary can forge a new signature $\sigma_i' = \sigma_i$ on a message m_i, the adversary have found a solution to the *CDH* problem. i.e., given $h^x = u^{t_i} \prod_{j=0}^{s-1} g^{m_{ij}\alpha^j}$ and $h^{\text{ò}} = v$, the adversary can generate $h^{x\,\text{ò}} = \sigma'$. Hence, the signature scheme used for authentication tags is existentially unforgeable.

Next, we prove that any proof information which can pass the ***verifyproof*** algorithm of our scheme must be the correct proof information.

Suppose a probabilistic polynomial time adversary Adv can generate a $Prf' = (\psi', \sigma', y'), (\psi', \sigma', y') \neq (\psi, \sigma, y)$ and pass the verification in our proposed scheme, we can get the following two equations:

$$e(\eta, \theta) \cdot e(\psi, k \cdot v^{-r}) = e(\sigma, h) \cdot e(g^{-y}, v) \quad (2)$$

$$e(\eta, \theta) \cdot e(\psi', k \cdot v^{-r}) = e(\sigma', h) \cdot e(g^{-y'}, v) \quad (3)$$

Dividing Eq.3 with Eq.2, we obtain:

$$e(\psi' \cdot \psi^{-1}, k \cdot v^{-r}) = e(\sigma' \cdot \sigma^{-1}, h) \cdot e(g^{-(y'-y)}, v) \quad (4)$$

Now we do a case analysis for Prf'.

Case 1: $\sigma' \neq \sigma$. We rewrite Eq.4 as

$$e(\sigma' \cdot \sigma^{-1}, h) = e(\psi' \cdot \psi^{-1}, k \cdot v^{-r}) \cdot e(g^{y'-y}, v)$$
$$= e((\psi' \cdot \psi^{-1})^{\alpha - r} \cdot g^{y'-y}, v) \quad (5)$$

Based on the Eq.5, we can get $((\psi' \cdot \psi^{-1})^{\alpha-r} \cdot g^{y'-y})^{\text{ò}} = \sigma' \cdot \sigma^{-1}$ For $Prf' = (\psi', \sigma', y')$ and $Prf = (\psi, \sigma, y)$ are both known to the Adv, the Adv can compute $\pi = (\psi' \cdot \psi^{-1})^{\alpha-r} \cdot g^{y'-y}$ with the mater key α got from the TA.

At first, we prove that $\pi \neq h$.

Suppose $\pi = h$, we can get the following equation:

$$(\psi' \cdot \psi^{-1})^{\alpha-r} \cdot g^{y'-y} = h \quad (6)$$

We rewrite Eq.6 as $(\psi' \cdot \psi^{-1})^{\alpha-r} = h \cdot g^{-(y'-y)}$.

According to the Knowledge of Exponent Assumption, given $(g, g^{\alpha-r})$, the only way for the Adv to output $(\psi' \cdot \psi^{-1}, h \cdot g^{-(y'-y)})$ is that the Adv must know the exponent c such that $\psi' \psi^{-1} = g^c$. Then we can get the following equation: $g^{c\,(\alpha-r)} = h \cdot g^{-(y'-y)}$

We see that the Adv have found the solution to the discrete logarithm problem: $h = g^{c\,(\alpha-r)+(y'-y)}$. Therefore, $\pi \neq h$. Using the same method, we can prove that $\pi \neq h^{\alpha}$.

Now, given input (h, v) and π, the Adv have solved the Static Diffie-Hellman Problem: $\pi^{\text{ò}} = \sigma' \sigma^{-1}$.

Therefore, $\sigma' = \sigma$.

Case 2: $y' \neq y$. Here, by Eq.4 and $\sigma' = \sigma$, we can output

$$e(\psi' \cdot \psi^{-1}, k \cdot v^{-r}) = e(g^{-(y'-y)}, v) \quad (7)$$

Based on the Eq.7 we can get $(\psi' \cdot \psi^{-1})^{\alpha-r} = g^{y-y'}$ We rewrite the above equation as $(\psi' \cdot \psi^{-1})^{\frac{1}{y-y'}} = g^{\frac{1}{\alpha-r}}$.

Now, we can output $(-r, g^{\frac{1}{\alpha-r}})$ as a solution for *t-SDH* problem, unless $\alpha = r$. However, α and r are two random numbers chosen from Z_p^*, so the probability of $\alpha = r$ is $1/p$, which is negligible. Therefor, $y' = y$.

Case 3: $\psi' \neq \psi$. Based on the Eq.4 with $\sigma' = \sigma$ and $y' = y$, we can output $(\psi' \cdot \psi^{-1})^{\alpha-r} = 1$. As $\psi' \neq \psi$, the Adv can infer $\alpha = r$. In this case, the Adv can also output $(r, g^{\frac{1}{\alpha+r}})$ as a solution of the *t-SDH* problem. Therefore, $\psi' = \psi$. In addition, as we proved in Theorem 1, $\psi = g^{f_{\vec{\omega}}(\alpha)}$ cannot be forged. That is, when the Adv outputs $\psi' = \psi$, it have to be computed based on actual data blocks according to our scheme.

Based on our above analysis, we prove that there is no Adv that can compute invalid proof information to pass the verification in our scheme with non-negligible probability, i.e., our scheme is equipped with soundness property.

B. *Confidentiality Guarantee*

Theorem 3. From the cloud server's prove message Prf, the user cannot recover the data file $m_{ij}, i \in I, 0 \le j \le s-1$.

Proof: Suppose the user can recover the data file $m_{ij}, i \in I, 0 \le j \le s-1$ in polynomial time, then we show the user can construct a simulator that solves the discrete logarithm problem.

The simulator is given as inputs values $\theta, v \in G$; its goal is to output ξ such that $\theta = v^{\xi}$.

For $\{v_i = \xi \cdot p_i\}, i \in K$, the simulator can get

$$y = f_{\vec{A}}(r) = \xi \cdot f_{\vec{A}'}(r)$$

where $\overrightarrow{A'} = \left\{\sum_{i \in I} p_i m_{i,0}, \cdots, \sum_{i \in I} p_i m_{i,s-1}\right\}$.

Based on the recovered data file $m_{ij}, i \in I, 0 \le j \le s-1$, the simulator computes $y' = f_{\vec{A}'}(r)$.

Finally the simulator computes $\xi = \dfrac{y}{y'}$.

So if the user can recover the data file $m_{ij}, i \in I, 0 \le j \le s-1$ in polynomial time, it can construct a simulator that solves the discrete logarithm problem.Thus the TPA cannot recover the data file $m_{ij}, i \in I, 0 \le j \le s-1$ based on the prove message.

C. *Active Attack Resistance Guarantee*

Now we show that our scheme can resist the active attack, as shown in Theorem 4.

Theorem 4. If an active adversary alters M^* to M', then, except with negligible probability the adversary cannot change the proof information Prf' produced by the cloud server based on the modified data file M' to the valid proof information Prf which can pass the verification algorithm.

Proof: Suppose there is an active adversary who alters M^* to M', where M' consists of $\left\{m'_{ij}\right\}, 1 \le i \le n, 0 \le j \le s-1$, and $m'_{ij} = m_{ij} + m''_{ij}$. $Prf' = \left\{\sigma, \psi', y'\right\}$ is the proof information produced by the cloud server based on the message M'.

Now we show that if the active adversary can produce the valid proof information $Prf = \left\{\sigma, \psi, y\right\}$ with negligible probability, it can construct a simulator to solve the discrete logarithm problem.

The simulator is given as inputs values $\theta, v \in G$; its goal is to output ξ such that $\theta = v^{\xi}$.

Based on the challenging message CM, the simulator computes

$$y'' = f_{\vec{B}}(r)$$

where $\vec{B} = \left\{\sum_{i \in I} p_i m''_{i,0}, \cdots, \sum_{i \in I} p_i m''_{i,s-1}\right\}$.

Based on y and y', the simulator computes

$$y^* = y - y' = f_{\vec{A}}(r) - f_{\vec{A}'}(r)$$

where $\vec{A} = \left\{\sum_{i \in I} v_i m_{i,0}, \cdots, \sum_{i \in I} v_i m_{i,s-1}\right\}$

$$\vec{A}' = \left\{\sum_{i \in I} v_i m'_{i,0}, \cdots, \sum_{i \in I} v_i m'_{i,s-1}\right\}.$$

It is clear that $y^* = \xi \cdot f_{\vec{B}}(r) = \xi \cdot y''$.

At the last, the simulaor can get $\xi = \dfrac{y^*}{y''}$.

So if the active adversary can produce the valid proof information $Prf = \left\{\sigma, \psi, y\right\}$ with negligible probability after it modified the data file, it can construct a simulator to solve the discrete logarithm problem.

Thus, our scheme can resist active attack.

4. Conclusions

In this paper, we propose a new public PoR scheme for data storage security in cloud computing, we utilize blinding factor to guarantee that any active adversary who is able to arbitrarily modify the cloud data cannot produce valid proof information to pass the verification algorithm and the TPA would not learn any knowledge about the data content during the auditing process. Then the following security analysis shows our proposed schemes is provably secure.

Acknowledgment

This work was supported partly by Supported by Beijing Natural Science Foundation (No:4122024,4132056), the New Star Plan Project of Beijing Science and Technology (NO:2007B001)and and The importation and development of High-Caliber Talents project of Beijing Municipal Institutions(NO:CIT$\&$TCD201304004).

References

[1] Y. Deswarte, J. J. Quisquater, A. Sa?dane. Remote integrity checking //Integrity and Internal Control in Information Systems VI. Springer US, 2004: 1-11. H. Simpson, *Dumb Robots*, 3rd ed., Springfield: UOS Press, 2004, pp.6-9.

[2] G. Ateniese, R. Burns, R. Curtmola, et al. Provable data possession at untrusted stores //Proceedings of the 14th ACM conference on Computer and communications security. ACM, 2007: 598-609.

[3] G. Ateniese, R. Di Pietro, L. V. Mancini, et al. Scalable and efficient provable data possession //Proceedings of the 4th international conference on Security and privacy in communication netowrks. ACM, 2008: 9.

[4] C. Erway, A. Küp?ü, C. Papamanthou, et al. Dynamic provable data possession //Proceedings of the 16th ACM conference on Computer and communications security. ACM, 2009: 213-222.

[5] J. Yuan, S. Yu. Proofs of retrievability with public verifiability and constant communication cost in cloud //Proceedings of the 2013 international workshop on Security in cloud computing. ACM, 2013: 19-26.