An efficient public auditing scheme for cloud storage server

Haiyang Yu[a], Yongquan Cai and Shanshan Kong

College of Computer Science, Beijing University of Technology, Beijing, 100022, China

Abstract. Storing data in cloud storage servers is a new trend with the development of cloud computing. Cloud storage brings lots of benefits and allows cloud users to access their data from any device. However, it also brings security challenges. Cloud user loses its control on its data because all data is stored remotely on the cloud servers. Data auditing is desired to help cloud user ensure the safety of its data. This technology can verify the integrity of cloud user's data without downloading any data. In this paper, we proposed an efficient public auditing scheme, which can preserve the privacy of cloud user's data by using the property of bilinear map and random masking technique. The proposed scheme uses BLS signature to improve the efficiency. Security analysis shows that the proposed scheme is provably secure.
Keywords: cloud storage; public auditing; data integrity checking; provable data possession.

1 Introduction

Nowadays, moving data to cloud storage servers becomes more and more popular among cloud users [1]. Cloud storage is becoming a key technology in cloud computing [2]. There is abundant storage resource in the cloud. When comparing with local storage, cloud storage offers flexible outsourcing service and brings lots of benefits such as low cost on software and hardware and convenient data access on any devices. However, even though cloud storage has all these advantages, it also brings serious security challenges [3]. System failures caused by famous Cloud Service Providers such as Google and Amazon [4] make the public worried about the stability of the cloud storage and the security of their remote data. Security of data in the cloud has become the key issue of limiting the use of cloud storage service [5-7]. After moving data to the cloud, cloud users lose their control and can't protect their data any more. Their data may be damaged by CSP or malicious attackers. Users should have the ability to know that if their data stored in cloud is intact. Thus, auditing for data integrity in the cloud becomes a hot area in both academia and industry.

Related works. Provable Data Possession (PDP) is considered as an essential technology in cloud auditing. It was first proposed by Ateniese et al. [8] in 2007. Then PDP has been well studied in several aspects [9-12].

Recently, Wang et al. [13] introduces a third party auditor to perform public auditing and relieves the computation burden of cloud users. Zhu et al. [14] proposed a cooperative provable data possession scheme which introduces index hash hierarchy. Liu and Jiang [15] proposed an auditing protocol to support batch auditing for multi-cloud environment. However, similar as

[a] Corresponding author : billyukiwi@163.com

[14], their scheme requires an organizer, which is not practical in reality. Yang et al. [16] proposed an auditing scheme to prevent the cloud servers from suffering DDOS attack, but it only supports constrained auditing number.

Our contributions. In this paper, we focus on the problem of integrity checking in cloud storage. We propose a public auditing protocol, which can efficiently verify the data integrity without leaking any privacy.

Our contribution can be illustrated as follows.

1. We design a new auditing scheme for cloud storage server and propose an efficient and privacy-preserving public auditing protocol.
2. We prove the security of our auditing scheme.

Organization. The remaining of this paper is organized as follows.

In Section 1, we discuss the related work of the proposed scheme. Section 2 introduces the framework, threat model design goals and some definitions of our scheme. In Section 3, we describe our detailed auditing scheme. Section 4 gives the security analysis of the proposed auditing scheme. In the end, we give the conclusion of the whole paper in Section 5.

2 Problem statement

2.1 System framework

The system framework consists of three entities: cloud user, auditor and cloud storage server, which are shown in Fig 1. Cloud user uses cloud storage and stores data remotely in cloud server. Cloud storage server provides the cloud storage service and has abundant storage and computation resource. Auditor has the capability and expertise to check the integrity of cloud user's data in cloud server.

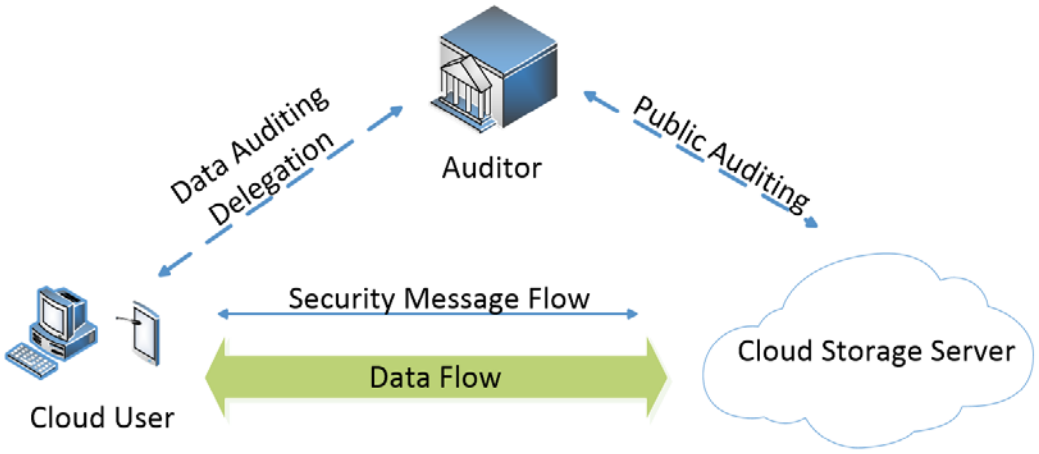

Figure 1. Framework of the Auditing System

Cloud user stores its data in cloud server and has the need to update its data. Cloud server manages data in its cloud storage servers. To check the integrity of data, cloud user delegates auditing tasks to the auditor. The auditor will perform data auditing and give the auditing result to cloud user.

2.2 Threat model and design goals

In the model of the proposed scheme, we consider that auditor is curious but honest. But the cloud storage sever cannot be trusted and may damage cloud user's data. To efficiently check

the data integrity, the proposed scheme should satisfy four properties: public auditability, auditing correctness, privacy preserving and efficiency.

3 The proposed auditing scheme

3.1 Definition of our auditing scheme

The proposed scheme consists of five algorithms (KeyGen, TagGen, Challenge, ProofGen, VerifyProof). KeyGenis a key generation algorithm which outputs secret hash key and a pair of user's public-secret key.TagGenis a tag generation algorithm which generates data tag for each data block and outputs a set of tags. Challenge is used to generate challenge information. ProofGenis a proof generation algorithm that outputs verification proofs.VerifyProofis a verification algorithm outputting the verification result. If the file component is intact, it outputs 1 and 0 otherwise.

3.2 Notations and preliminaries

1) Bilinear Map. Let G_1, G_2 and G_T be multiplicative cyclic groups of prime order p. Let g_1 be the generator of G_1 and g_2 be the generator of G_2. A bilinear map is e:$G_1 \times G_2 \rightarrow G_T$ which has three properties: for all $u \in G_1, v \in G_2$ and a, b$\in Z_p$, $e(u^a,v^b)=e(u,v)^{ab}$; the map is nondegeneracy, which implies $e(g_1,g_2) \neq 1$; there is an efficient algorithm to compute the e.

2) BLS Signature. BLS signature scheme consists of four algorithms: Setup, KeyGen, Sign and Verify.

- Setup. The algorithm creates a bilinear map e:$G_1 \times G_2 \rightarrow G_T$ and a hash function $h:\{0,1\}^* \rightarrow G_1$. Let P be the generator and m be the message to be signed.
- KeyGen. The algorithm randomly chooses secret key $sk \in Z_q$ and chooses generator $P \in G_2$. Then it computes public key $pk=P^{sk}$.
- Sign. The algorithm computes hash value $h(m)$ and signature $sig=h(m)^{sk}$.
- Verify. By comparing $e(sig,P)$ and $e(h(m),pk)$, the algorithm verifies the signature.

3.3 The proposed auditing scheme

In our scheme, to satisfy the property of privacy preserving, we introduce random masking technique, which can efficiently prevent cloud user's data blocks from being derived by the auditor.

Let e be the bilinear map e:$G_1 \times G_2 \rightarrow G_T$, where G_1, G_2 and G_T are multiplicative cyclic groups of large prime order p. Let g_1 and g_2 be the generator of G_1 and G_2, respectively.

The detailed auditing scheme is as follows:

We now describe the detailed proposed scheme, which consists of two phases: Setup and Auditing.

Setup. The cloud user runs the key generation algorithm KeyGen to generate public and private keys. It chooses two random elements sk_u, $sk_h \in Z_p$ to create secret user key and secret hash key respectively and computes public user key $pk_u=g_2^{sku} \in G_2$.

The cloud user divides each file into data blocks and splits each data block into sectors. Then it runs the TagGen algorithm to generate homomorphic verification tags for data blocks. Cloud user selects s random values $\rho_1,\rho_1,\ldots\rho_s \in Z_p$ and computes $\varphi_k=g_1^{\rho k} \in G_1$, *for* $k \in [1,s]$. Verification tag τ_j of each data block d_j is computed as

$$\tau_j = (h(sk_h, \mathrm{T}_j) \cdot \prod_{k=1}^{s} \varphi_k{}^{d_{jk}})^{sk_u} \quad . \tag{1}$$

where $T_j=FID \,//\, j$. is identifier of each file and j is the index of d_j.

After the set of verification tags $\Phi=\{\tau_j\}_{j\in[1,n]}$ is generated, cloud user sends all data blocks $\{d_j\}_{j\in[1,n]}$, the set of verification tags Φ and parameters $\{\varphi_k\}_{k\in[1,s]}$ to the cloud server. The cloud user sends secret hash key sk_h, public user key pk_u and parameters $\{\varphi_k\}_{k\in[1,s]}$ to the auditor together with F_{info}.

Auditing. Cloud user sends checking request to the auditor to check the data integrity in cloud storage server. The auditor generates the challenge C by running the challenge algorithmChallenge. It constructs a challenge set Θ by randomly choosing α data blocks. It then chooses a random value v_j for each selected data block d_j. It finally generates the challenge $C=\{j,v_j\}_{j\in\Theta}$.

After challenge generation, the auditor sends the challenge C to the cloud server. The cloud server generates response proof P by running ProofGen algorithm. It first computes an aggregated tag $\tau=\prod_{j\in\Theta}\tau_j^{vj}\in G_1$. It then selects a random element $\gamma\in Z_p^*$ and s random values $\theta_k\in Z_p$ for $k\in[1,s]$. It computes response proof $\eta=\{\eta_k\}_{k\in[1,s]}$, where $\eta'_k=\sum_{j\in\Theta}v_j\cdot d_{jk}$ for $k\in[1,s]$ and $\eta_k=\gamma\cdot\eta'_k+\theta_k$. Then it calculates $\Psi=e(\prod_{k=1}^{s}\varphi_k^{\theta k}, pk_u)$. The response proof is $P=(\tau,\eta,\Psi,\gamma)$.

The cloud server sends the response proof back to the auditor. The auditor finally runs the verification algorithm VerifyProof to generate the auditing result. It then verify the response proof from the auditor by the following equation:

$$\Psi\cdot e(\tau^{\gamma},g_2)=e\Big(\prod_{j\in\Theta}h\big(sk_h,T_j\big)^{\gamma v_j}\cdot\prod_{k=1}^{s}\varphi_k^{\eta_k},pk_u\Big), \tag{2}$$

if the equation holds, then the algorithm outputs 1. Otherwise, it outputs 0.

The correctness of the equation can be proved as follows:

$$\begin{aligned}
\Psi\cdot e(\tau^{\gamma},g_2) &= e\Big(\prod_{k=1}^{s}\varphi_k^{\theta_k},pk_u\Big)\cdot e\Big(\prod_{j\in\Theta}\tau_j^{v_j\gamma},g_2\Big)\\
&= e\Big(\prod_{k=1}^{s}\varphi_k^{\theta_k},pk_u\Big)\cdot e\Big(\prod_{j\in\Theta}\big(h(sk_h,T_j)\cdot\prod_{k=1}^{s}{\varphi_k}^{d_{jk}}\big)^{sk_u v_j\gamma},g_2\Big)\\
&= e\Big(\prod_{k=1}^{s}\varphi_k^{\theta_k},pk_u\Big)\cdot e\Big(\prod_{j\in\Theta}h(sk_h,T_j)^{\gamma v_j}\cdot\prod_{k=1}^{s}\prod_{j\in\Theta}\varphi_k^{\gamma v_j\cdot d_{jk}},pk_u\Big)\\
&= e\Big(\prod_{k=1}^{s}\varphi_k^{\theta_k},pk_u\Big)\cdot e\Big(\prod_{j\in\Theta}h(sk_h,T_j)^{\gamma v_j}\cdot\prod_{k=1}^{s}\varphi_k^{\gamma\cdot\sum_{j\in\Theta}v_j\cdot d_{jk}},pk_u\Big)\\
&= e\Big(\prod_{j\in\Theta}h(sk_h,T_j)^{\gamma v_j}\cdot\prod_{k=1}^{s}\varphi_k^{\gamma\cdot\eta'_k+\theta_k},pk_u\Big)\\
&= e\Big(\prod_{j\in\Theta}h\big(sk_h,T_j\big)^{\gamma v_j}\cdot\prod_{k=1}^{s}\varphi_k^{\eta_k},pk_u\Big)
\end{aligned}$$

4 Security analysis

We analyze the achievement of security properties in Section 2.2, which are public auditability and privacy preserving property.

4.1 Public auditability

Theorem 1. The proposed scheme allows not only the cloud user, but all public entities to challenge the cloud storage server to verify the integrity of cloud data.

Proof. In auditing phase, every entity including auditor or other cloud users could obtain the secret hash key, public user key and parameter φ_k from cloud user. It can generate challenge by collecting file information and generating random values. Besides, it can get the response proof

from cloud storage server. Thus every entity could send a challenge to the cloud storage server and verify the response proof from it, which shows the property of public auditability in the proposed scheme.

4.2 Privacy preservation

Theorem 2. The auditor can't learn any content of cloud user's data according to the response proof from the cloud storage server.

Proof. With the random values γ and θ_k, the auditor cannot solve the linear equations in response proofs, which means the auditor cannot derive cloud user's data by collecting linear combination of data blocks [13]. The auditor can't reveal the random parameter θ_k from Ψ because of the bilinear map. Therefore, the auditor cannot retrieve any information of cloud user's data by collecting the verification response.

5 Conclusion

Data auditing for remote cloud storage is an important technology. In this paper, we designed an auditing scheme for verifying the integrity of cloud user's data. The proposed scheme generates the verification tags and proofs by using homomorphic linear authenticators, which is based on BLS signatures. By introducing random mask technique, the proposed scheme can preserve the privacy of cloud user's data. It can reduce the computation load of both auditor and cloud server. In the future, we will consider to extend our work to support dynamic auditing for data update operation and batch auditing for multiple cloud servers and cloud users.

Acknowledgments

This work was funded by the National Natural Science Foundation of China under grant 61170221.

References

1. M. Armbrust, A. Fox, R. Griffith,A. D. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, A. Rabkin, I. Stoica, M. Zaharia, **53**, 4, pp.50-58(2010)
2. P. Mell, T. Grance, Communications of the ACM, **53**, 6(2010),.
3. L. Wei, H. Zhu, Z. Cao, X. Dong, W. Jia,Y. Chen, A. V. Vasilakos,Information Sciences,**258**, pp.371-386(2014)
4. S. Tan, Y. Jia, W. H. Han, Chinese Journal of Computers, **1**, pp.13(2015)
5. W. Jansen, T. Grance, NIST Special Publication, Gaithersburg, MD, United States,(2011).
6. K. Julisch, M. Hall, Information Security Journal: A Global Perspective,**19**, 6, pp.299-309(2010)
7. D. G. Feng, M. Zhang, Y. Zhang, Z. Xu, Journal of Software, **22**, 1, pp.71-83(2011)
8. G. Ateniese, R. Burns, R. Curtmola, J. Herring, L. Kissner, Z. Peterson, D. Song, *Proceedings of the 14th ACM Conference on Computer and Communications Security, Alexandria, USA*,pp.598-609 (2007)
9. C. C. Erway, A. Küpçü, C. Papamanthou, R. Tamassia, ACM Transactions on Information and System Security,**17**, 4, pp. 213-222(2015)
10. Q. Wang, C. Wang, J. Li, K. Ren, W. Lou, *European Symposium on Research in Computer Security, Saint-Malo, France*,(2009)
11. R. Curtmola, O. Khan, R. Burns, G. Ateniese, *The 28th International Conference on Distributed Computing Systems, Beijing, China*,(2008)
12. A. F. Barsoum and M. A. Hasan, IACR Cryptology ePrint Archive 2011, pp. 447-476(2011)
13. C. Wang, S. S. Chow, Q. Wang, K. Ren, W. Lou, IEEE Transactions on Computers, **62**, 2,pp. 362-375(2013)

14. Y. Zhu, H. Hu, G. J. Ahn, M. Yu, IEEE Transactions on Parallel and Distributed Systems, **23**, 12,pp. 2231-2244(2012)
15. X. Liu and Y. Jiang, International Journal of Security and Its Applications, **8**, 6,pp. 197-210(2014)
16. G. Yang, H. Xia, W. Shen, X. Jiang, J. Yu, International Journal of Security and Its Applications,**9**, 9, pp. 21-32(2015)