Cryptanalysis of auditing protocol proposed by Wang et al. for data storage security in Cloud Computing

XU Chun-xiang, HE Xiao-hu, Daniel Abraha
School of Computer Science and Engineering, University of Electronics Science and Technology of China
2006 Xi Yuan Avenue, West High-tech Zone, Chengdu, China
email: hh2870714@163.com

Abstract. Cloud Computing as the on-demand and remote provision of computational resources has been eagerly waited for a long time as a computing utility. It helps users to store their data in the cloud and enjoy the high quality service. However, users do not have physical possession on their own data, hence it is indispensable to create mechanisms on how to protect the security of the data stored. Thus, some auditing protocols are introduced to ensure authenticity and integrity of the outsourced data. Wang et al. proposed a public auditing protocol in 2010 and argued that it can resist against various known attacks. In this paper, we analyze the protocol and find serious security flaws in their protocol. Our analysis shows that the public auditing scheme proposed by Wang et al. can not resist against existential forgery using a known message attack. Moreover, we show that the protocol is vulnerable to attacks by a malicious cloud server and an outside attacker through four specific attacking schemes. The results show that the protocol can not provide secure data storage for users.

Key words: Cloud Computing, public auditing, secure data storage

1. Introduction

Cloud Computing is emerging as the next evolution of computing for its numerous contribution to the IT enterprise. In contrast to traditional solutions, Cloud Computing has a number of essential characteristics, such as: on-demand self-service, ubiquitous network access, location independent resource pooling, rapid elasticity, and measured service [1]. Despite these distinct advantages it brings, Cloud Computing has introduced some new security threats and challenges. If we could not solve this problem, it will seriously hinder the development of Cloud Computing.

Data outsourcing, one of the fundamental components of Cloud Computing, centralizes users' data to the cloud server (CS). Users including both individuals and enterprises can remotely store their data in the cloud and enjoy benefits, such as: release the pressure from storage management, universal data access with independent geographical locations, and reducing expenditure on hardware, software, personnel maintenance, and so on [2]. Data outsourcing has been applied to commercial application in pay-as-you-use fashion at relatively low prices for users. For example, Amazon's S3 data storage service just charges $0.12/GB to $0.15/GB monthly. As a result, with the increasing development of Cloud Computing technologies, it is believed that more and more users will prefer to store their data in the cloud.

As mentioned above, Cloud Computing faces a lot of security threats and challenges. While data outsourcing brings advantages and convenience to users, there are also many security issues. When users outsource their data in the CS, they no longer possess their data physically. As a result, data outsourcing confronts the following security problems. Firstly, although the cloud service provider (CSP) can provide more powerful and reliable infrastructures than users, a huge mass of data storing in the CS makes it more vulnerable to active attack. Moreover, the CSP may also lead to data loss because of some irresistible reasons [3]-[5]. Secondly, towards the cloud users, the CSP may deliberately distort the status of users' outsourced data for some benefits. For example, the CSP may discard the data that users rarely or never access to save costs, or even hide data loss

incidents for the sake of reputation [6]-[7]. So we can see, although data outsourcing can bring advantages and convenience to users, it can not ensure the authenticity and integrity of data.

Generally speaking, users like to outsource a huge mass of data in the CS, so simply downloading the data to verify the authenticity and integrity is not a feasible solution. To solve the security problems of data outsourcing mentioned above, researchers proposed auditing protocols to ensure the correctness of the outsourced data. The authenticity and integrity of data should be guaranteed in a relatively low computation and communication complexity through an efficient auditing protocol.

We divide auditing protocol into self auditing protocol and public auditing protocol according to different types of auditors. Juels and Kaliski proposed a self auditing protocol [8] which successfully realizes secure auditing of the outsourced data. Furthermore, there are also some other self auditing protocols been proposed [6][9]. Considering the users' limited computing or communication capability in practice, people proposed public auditing protocols. It is of great importance to enable public auditing so that users can recourse to a third party auditor (TPA) who has expertise and capabilities and users do not have to audit the outsourced data by themselves. Shah et al. proposed some public auditing protocols [7][10]. However, their protocols can only be applied to encrypted data, which limits their scope of application. Ateniese et al. proposed an efficient public auditing protocol [6], but it does not support the privacy protection of users' data against the TPA. Wang et al. proposed a privacy-preserving public auditing protocol [11] and argued that it can resist against various known attacks. However, we find that this public auditing protocol is vulnerable to attacks from a malicious cloud server and an outside attacker.

In this paper, we firstly review the public auditing protocol proposed by Wang et al. [11]. Then, we demonstrate that the protocol can not ensure the authenticity and integrity of the outsourced data through our four concrete attack schemes.

2. Review of the protocol proposed by Wang et al.

In this section, some notations and preliminaries are provided and then we describe the public auditing protocol proposed by Wang et al..

2.1 Notations and Preliminaries

Some notations used in the proposed protocol are outlined.

- F: the data file to be outsourced, we assume that F can be denoted as n blocks $m_1,\ldots,m_n \in Z_p$ for some large prime p.
- $f_{key}(\cdot)$: the pseudorandom function (PRF), define as: $\{0,1\}^* \times key \to Z_p$.
- $\pi_{key}(\cdot)$: the pseudorandom permutation (PRP), define as: $\{0,1\}^{\log_2(n)} \times key \to \{0,1\}^{\log_2(n)}$.
- $H(\cdot)$: the one-way hash function, define as: $\{0,1\}^* \to G_1$, where G_1 is a multiplicative cyclic group of prime order p.
- $h(\cdot)$: the one-way hash function, define as: $G_1 \to Z_p$.

Some necessary cryptographic background information for the proposed scheme is introduced below.

Bilinear Map: G_1, G_2, G_T are multiplicative cyclic groups of prime order p. g_1 is a generator of G_1 and g is a generator of G_2. A bilinear map is a map e: $G_1 \times G_2 \to G_T$ with the following properties: (1) Bilinear: for all $u \in G_1$, $v \in G_2$ and a, $b \in Z_p$, $e(u^a, v^b) = e(u,v)^{ab}$. (2) Non-degenerate: $e(g_1, g) \neq 1$. (3) Computable: there exists an efficient computable algorithm for computing e.

2.2 Protocol description

A public auditing protocol is a collection of four polynomial-time algorithms (KeyGen, TagBlock, GenProof, CheckProof). KeyGen is a probabilistic key generation algorithm run by the user to setup the protocol. TagBlock is an algorithm run by the user to generate the verification metadata. GenProof is run by the cloud server in order to generate a proof of possession. CheckProof is run by the TPA in order to validate a proof of possession. The detailed steps of the protocol proposed by Wang et al. are described as follows:

(1) KeyGen:

The cloud user runs KeyGen to generate the public and secret keys used in the protocol. He chooses a random $x \in Z_p$ and a random element $u \in G_1$. Then he computes $v=g^x$, $w=u^x$. The secret key is $sk=(x)$ and the public keys are $pk=(v, w, g, u)$.

(2) TagBlock:

The cloud user is in possession of the data file F and he runs TagBlock to generate the verification metadata for each block m_i by computing $\sigma_i = (H(i) \cdot u^{m_i})^x \in G_1 (1 \le i \le n)$. Then the user sends $\{F, \{\sigma_i\}_{1 \le i \le n}\}$ to the cloud server and deletes them from his local storage.

(3) GenProof:

The TPA selects the number of data blocks c that he needs to audit the cloud server for proof of possession. For each auditing the TPA computes $s_j = \pi_{k_1}(j) (1 \le j \le c)$ which indicate the random blocks the TPA wants to audit, where k_1 is a fresh randomly chosen key by the TPA for each auditing. It is assumed that $I = \{s_j\} (1 \le j \le c)$ and $s_1 \le \ldots \le s_c$. For each element $i \in I$, the TPA chooses a random v_i. Then the TPA sends the $chal = \{(i, v_i)\}_{i \in I}$ as a challenge to the cloud server.

Upon receiving challenge $chal = \{(i, v_i)\}_{i \in I}$, the cloud server runs GenProof to generate a proof of possession. Firstly, the cloud server computes $r = f_{k_2}(chal) \in Z_p$, where k_2 is a fresh randomly chosen key by the cloud server for each auditing. Then he computes $R = (w)^r = (u^x)^r \in G_1$, $\mu' = \sum_{i \in I} v_i m_i$, $\mu = \mu' + rh(R) \in Z_p$ and $\sigma = \prod_{i \in I} \sigma_i^{v_i} \in G_1$. Finally, he sends $\{\mu, \sigma, R\}$ to the TPA as a response to the challenge.

(4) CheckProof:

Upon receiving the response $\{\mu, \sigma, R\}$, the TPA runs CheckProof to validate the response by checking the verification equation $e(\sigma \cdot (R^{h(R)}), g) \stackrel{?}{=} e(\prod_{i \in I} H(i)^{v_i} \cdot u^{\mu}, v)$. If the cloud server verifies the equation equal, it means that the cloud server possesses the outsourced data.

3. Attacks on the protocol proposed by Wang et al.

We consider the cloud data storage service involving three different entities similar to what is explained in Wang et al. paper: the cloud user, the cloud server and the third party auditor (TPA). The cloud server might be a malicious cloud server which might not keep data or might delete data owned by cloud users and might even hide some data corruptions. We also consider an outside attacker which can intercept or eavesdrop on the data cloud users send to the cloud server.

It is claimed in Wang et al.'s paper that their public auditing protocol can resist against various known attacks. Unfortunately, we find that this protocol is vulnerable to attacks from a malicious cloud server and an outside attacker. The malicious cloud server can modify the outsourced data as he wants when he possesses it. Moreover, the malicious cloud server can pass the auditing from the TPA when he loses the outsourced data. Besides the attacks from the malicious cloud server, we find that this protocol is vulnerable to attacks from an outside attacker. Even if the cloud server is trusted, the outside attacker can intercept the data sent from the user to the cloud server in

TagBlock step and modify it arbitrarily. Furthermore, the outside attacker can just eavesdrop on that data and forge a great deal of data.

Scheme I Data modification Tag forging attack

We assume that the malicious cloud server wants to modify m_j in F to m_j^* which is chosen as he wants. The malicious cloud server can modify the data m_j and forge its corresponding tag σ_j so that they will be able to pass the auditing from the TPA. The detailed attack scheme is as follows:

(1) TagBlock:

After the user finishes KeyGen and TagBlock, the malicious cloud server computes $\sigma_j^* = \sigma_j \cdot w^{m_j^* - m_j} = (H(j) \cdot u^{m_j^*})^x$ using the data m_j he possesses, tag σ_j and public key w. Then the malicious cloud server modifies m_j to m_j^* and σ_j to σ_j^*.

(2) GenProof:

Upon receiving challenge $chal = \{(i, v_i)\}_{i \in I}$, the malicious cloud server computes $r = f_{k_2}(chal)$ and $R = (w)^r = (u^x)^r$ as normal. However, he computes $\mu'^* = \sum_{i \in I} v_i m_i^*$, $\sigma^* = \prod_{i \in I} \sigma_i^{*v_i}$, $\mu^* = \mu'^* + rh(R)$ differently. Finally, he sends $\{\mu^*, \sigma^*, R\}$ to the TPA as a new response to the challenge.

(3) CheckProof:

Upon receiving the response $\{\mu^*, \sigma^*, R\}$, it is clear that the TPA verifies equation $e(\sigma^* \cdot (R^{h(R)}), g) = e(\prod_{i \in I} H(i)^{v_i} \cdot u^{\mu^*}, v)$ equal.

From the above attack scheme, we can see that the malicious cloud server can modify the outsourced data as he wants when he possesses the outsourced data and the TPA can't find this modification.

Scheme II Data lost Auditing pass attack

We assume that the malicious cloud server loses the outsourced data for some internal or external reasons. The malicious cloud server also can pass the auditing from the TPA. The detailed attack scheme is as follows:

(1) TagBlock:

After the user finishes KeyGen and TagBlock, the malicious cloud server computes $t(i) = \sigma_i / w^{m_i} = H(i)^x (1 \le i \le n)$ and saves $T = \{t(i)\} (1 \le i \le n)$.

(2) GenProof:

Upon receiving challenge $chal = \{(i, v_i)\}_{i \in I}$, the malicious cloud server computes $r = f_{k_2}(chal)$ and $R = (w)^r = (u^x)^r$ as normal. However, he computes $\mu^* = rh(R)$, $\sigma^* = \prod_{i \in I} t(i)^{v_i}$ differently. Finally, he sends $\{\mu^*, \sigma^*, R\}$ to the TPA as a new response to the challenge.

(3) CheckProof:

Upon receiving the response $\{\mu^*, \sigma^*, R\}$, it is clear that the TPA verifies equation $e(\sigma^* \cdot (R^{h(R)}), g) = e(\prod_{i \in I} H(i)^{v_i} \cdot u^{\mu^*}, v)$.

From the above attack scheme, we can see that the malicious cloud server also can pass the auditing from the TPA although he does not possess the user's data.

The two attacks above are launched by the malicious cloud server. Furthermore, we show that the protocol is vulnerable to attacks from an outside attacker even if the cloud server is trusted.

Scheme III Data interception and modification attack

The outside attacker can intercept the data sent from the user to the cloud server in TagBlock step and modify it arbitrarily. The scheme is explained as follows:

i. After the user finishes KeyGen and TagBlock steps, he sends $\{F, \{\sigma_i\}_{1\le i\le n}\}$ to the server. But the outside attacker intercepts it and modifies m_j to m_j^* and σ_j to σ_j^* and sends modified data to the server. The computation is as that of Scheme I.

ii. The computation of GenProof step by the cloud server and the CheckProof step by the TPA will be done as normal protocol computation.

iii. It is now clear that this modification of data by an outside attacker still passes the TPA.

As a result, the outside attacker modifies the data while the TPA couldn't find the modification by the attacker.

Scheme IV Data eavesdropping and forgery

The outside attacker eavesdrops on the data m_j and its tag σ_j. Then he can forge a great deal of data $(m_j^1, \sigma_j^1), (m_j^2, \sigma_j^2), (m_j^3, \sigma_j^3), \ldots$. The detailed attack scheme is as follows:

i. After the user performs TagBlock step, the outside attacker will eavesdrop on the data (m_j, σ_j)and then do the forging as:

$$\sigma_j^k = \sigma_j w^{m_j^k - m_j} \quad \text{where } k=1, 2, 3, \ldots.$$

ii. The outside attacker forges a great deal of data and sends it to the cloud server. The cloud server will receive these plenty of data as if they were normal data from the cloud user and store them.

iii. Clearly, the TPA will not ask for the proof of these data.

Actually these plenty of data were non-sense data and they do not deserve storage in the cloud server. This attack imposes cost on the cloud user for storage. It is obvious that the malicious cloud server can also launch the same attack.

Consequently, the above four attack schemes show that this public auditing protocol is vulnerable to existential forgeries using known message attacks from a malicious cloud server and an outside attacker. Even if the cloud server has passed the auditing from the TPA, the authenticity and integrity of the outsourced data are not guaranteed.

4. Conclusion

In this paper, we have shown the security flaws of the protocol proposed by Wang et al. Although Wang et al. claimed their protocol is more efficient than others and can resist against various attacks [11], we have demonstrated that the protocol is vulnerable to existential forgeries using known message attacks from a malicious cloud server and an outside attacker. The malicious cloud server can modify the outsourced data as for their own purpose and need when he possesses the actual outsourced data. The outside attacker can also modify data as he wants and even forge a great deal of data which asks the user too much cost for the data storage. Therefore, the protocol can not ensure the security of the outsourced data. How to remedy the security flaws of the protocol requires further research.

Acknowledgment

This work is supported by Science and Technology on Communication Security Laboratory Foundation(NO.9140C110301110C1103) and The Weaponry Equipment Pre-Research Foundation, The PLA General Armament Department(NO.9140A04020311DZ02).

REFERENCES

[1] P. Mell and T. Grance, Draft NIST working definition of cloud computing, Referenced on June. 3rd, 2009 Online at http://csrc.nist.gov/groups/SNS/cloud-computing/index.html, 2009.

[2] M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, A. Rabkin, I. Stoica, and M. Zaharia, Above the clouds: A berkeley view of cloud computing. University of California,

Berkeley, Tech, Rep. UCB-EECS-2009-28, Feb 2009.
[3] Amazon.com, Amazon s3 availability event: July 20, 2008, Online at http://status.aws.amazon.com/s3-20080720.html, July 2008.
[4] S.Wilson, Appengine outage, Online at http://www.cio-weblog.com/50226711/ appengine_outage.php, June 2008.
[5] B.Krebs, Payment processor breach may be largest ever, Online at http://voices.washingtonpost.com/securityfix/2009/01/payment_processor_breach_may_b.html, Jan. 2009.
[6] G. Ateniese, R. Burns, R. Curtmola, J. Herring, L. Kissner, Z. Peterson, and D. Song, Provable data possession at untrusted stores, Cryptology ePrint Archive, Report 2007/202, 2007.
[7] M. Shah, R. Swaminathan, and M. Baker, Privacy-preserving audit and extraction of digital contents, Cryptology ePrint Archive, Report 2008/196, 2008.
[8] A. Juels and B. Kaliski, PORs: Proofs of retrievability for large files, In ACM CCS'07, Full paper available on e-print(2007/243), 2007.
[9] C. Wang, Q. Wang, K. Ren, and W. Lou, Ensuring Data Storage Security in Cloud Computing, In Proc. of IWQoS'09, July 2009.
[10] M. Shah, M. Baker, J. Mogul, and R. Swaminathan, Auditing to keep online storage services honest, In Proc. of HotOS'07. Berkeley, CA, USA: USENIX Association, 2007, pp. 1-6.
[11] C. Wang, Q. Wang, K. Ren, and W. Lou, Privacy-preserving public auditing for data Storage Security in cloud computing, In InfoCom2010, IEEE, March 2010.