Future Generation Computer Systems 52 (2015) 77–85

Contents lists available at ScienceDirect

Future Generation Computer Systems

journal homepage: www.elsevier.com/locate/fgcs

Remote data possession checking with enhanced security for cloud storage☆

Yong Yu [a,b,*], Yafang Zhang [a], Jianbing Ni [a], Man Ho Au [c], Lanxiang Chen [d], Hongyu Liu [a]

[a] School of Computer Science and Engineering, University of Electronic Science and Technology of China, Chengdu, China
[b] State Key Laboratory of Information Security, Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China
[c] Department of Computing, The Hong Kong Polytechnic University, Hong Kong
[d] School of Mathematics and Computer Science, Fujian Normal University, China

HIGHLIGHTS

- Analyze the security of a remote data possession checking protocol.
- Show the protocol is vulnerable to replay attack and deletion attack.
- Propose an improvement to resist the attacks.
- Prove the security of the improvement.
- Report the performance of the improvement by implementing it.

ARTICLE INFO

Article history:
Received 22 June 2014
Received in revised form
26 August 2014
Accepted 1 October 2014
Available online 13 October 2014

Keywords:
Cloud storage
Data integrity
Algebraic signature
Provable security

ABSTRACT

Cloud storage allows users to enjoy the on-demand and high quality data storage services without the load of local data maintenance. However, the cloud server providers are not fully trusted. Whether the data over cloud servers are intact becomes a major concern of data owners. To offer cloud users with the capacity of data integrity verification, recently, Chen proposed a remote data possession checking (RDPC) protocol from algebraic signatures which achieves many desirable features such as high efficiency, short length of challenges and responses, non-block verification. Unfortunately, in this paper, we find that the protocol is vulnerable to replay attack and deletion attack launched by a dishonest server. Specifically, the server can deceive the users to believe that their data are well hold by replaying a previous evidence or re-constructing the deleted data blocks from the corresponding tags in the integrity checking process, while their data have been partially discarded in fact. Then, we present an improved scheme to fix the security flaws of the original protocol. Both the theoretical analysis and the implementation results show that the improvement is secure and practical.

1. Introduction

Cloud storage provides a novel service model wherein data are maintained, managed and backed up remotely and accessed by the users over the network at anytime and from anywhere [1]. Currently, an increasing number of individuals and corporations outsource their data to the cloud servers to free themselves from the workload of data storage and management. However, once the data are uploaded to the cloud, their fate is out of the data owners' control. Although a legitimate cloud storage provider will take care of the outsourced data, it will do good to itself in case some accidents happen because the cloud service providers are not fully trusted [2]. For example, since frequent data access increases the probability of disk corruption, the loss of data may occur constantly, but cloud servers may try to hide data loss incidents in order to maintain their reputation. What is more serious is that cloud servers might discard the data that have not been or are rarely accessed for monetary reasons. Indeed, data breaches and data loss are the two most frequent incidents among the "9 Worst Cloud Security Threats" listed by InformationWeek.[1] Therefore, the cloud users should have a strong evidence to smooth away the

☆ A preliminary version of this paper was presented at ISPEC 2014.

* Corresponding author at: School of Computer Science and Engineering, University of Electronic Science and Technology of China, Chengdu, China.
E-mail address: yyucd2012@gmail.com (Y. Yu).

[1] http://www.informationweek.com/cloud/infrastructure-as-a-service/9-worst-cloud-security-threats/d/d-id/1114085.

http://dx.doi.org/10.1016/j.future.2014.10.006
0167-739X/

worry about the integrity of the data which are being tempered with and partially discarded.

To check the data integrity over untrusted stores, in 2007, Ateniese et al. [3,4] proposed the notion of provable data possession (PDP) for the first time and presented two efficient and provably-secure PDP schemes based on homomorphic verifiable tags. In their protocols, users are allowed to verify data integrity without accessing the entire file. At the same time, Juels et al. [5] formalized the model of proof of retrievability (PoR) which enables the server to produce a concise proof that a user can retrieve data, and then, presented a sentinel-based PoR scheme using error-correcting code. In 2008, Shacham and Waters [6,7] described two efficient and compact PoR schemes. In 2009, Ateniese et al. [8] provided a framework for building public-key homomorphic linear authenticators from any identification protocol and then described how to turn any public-key homomorphic linear authenticator into a publicly-verifiable PDP scheme with an unbounded number of verifications. Subsequently, a number of data auditing protocols [9–18] from some efficient PDP and PoR schemes [5–7,19–21] were proposed to ensure that the integrity of users' data. In particular, a notion of remote data possession checking (RDPC), which is a similar concept inherited from PDP, proposed by Chen [22] and a basic construction was presented using an algebraic signature. But in this basic protocol, the number of verifications is limited. To overcome this drawback, an improved scheme supporting to refresh tags after t verifications was also proposed in [22]. Both protocols provide a number of desirable features such as high efficiency, short length of challenges and responses, no-block verification and were suggested to be adopted to the cloud storage scenario.

Our contribution. The contributions of this paper can be summarized as follows:

(1) We identify several security flaws in the RDPC protocols in [22]. First, neither the basic protocol nor the improved one is secure against the replay attack, in which the server is able to generate a valid proof from the previous proofs, without accessing the actual data. Consequently, the server needs only to store a previous proof and replay it as a valid response when required. Second, the improved protocol is vulnerable to a malicious server's deletion attack; namely, the server can generate a valid response in the integrity checking process after deleting the original data file.
(2) To fix these security problems, we propose a new RDPC protocol by utilizing some techniques including involving the file name and the block sequence numbers in generating each tag, using pseudo-random functions to modify the algebraic signature algorithm, and the random sampling trick to provide probabilistic auditing.
(3) We prove that the fixed protocol is secure based on the security model due to Ateniese et al. [3] and maintains the desirable features of the original protocol on performance. The efficiency of our enhanced protocol is comparable to that of the state-of-art and the implementation results show that the improvement is practical.

Portions of the work presented in this paper have previously appeared as an extended abstract at [23]. We revised the paper a lot and added more technical details and experimental results in this version as compared to [23]. Specially, we added Section 6 to analyze the performance of our new scheme, which is lacking in [23]. We also provided a complete security proof in Section 5 in this paper.

Organization: Section 2 gives some preliminaries used in this paper. Section 3 reviews the RDPC protocols in [22] and discusses the security of the protocols. Section 4 describes our new RPDC protocol. Section 5 provides security proofs for the new RDPC protocol. Section 6 offers performance analysis of our protocol and Section 7 concludes the paper.

2. Preliminaries

In this section, we review basic knowledge of the RDPC protocols, including security model, components of an RDPC protocol and its security requirements.

2.1. System model

The remote data possession checking architecture for cloud storage involves two entities: a cloud server and its users. The cloud server, which has significant storage space and computation resources, stores users' data and provides data access service. The users have large amount of data to be stored on the cloud in order to eliminate the overhead of local storage. As users no longer possess the entire data locally and the cloud server is not fully-trusted, it is of critical importance for users to ensure their data are correctly stored and maintained in the cloud. Therefore, the users should be able to efficiently check the integrity and correctness of their outsourced data.

2.2. Components of an RDPC protocol

A remote data possession checking protocol, which can be used to verify the integrity of the users' data, consists of five phases: **Setup**, **TagBlock**, **Challenge**, **ProofGen** and **ProofVerify** [3,4].

- **Setup** is a probabilistic algorithm that is run by the user to setup the protocol. It takes a security parameter κ as input and returns k as the secret key of the user.
- **TagBlock** is a probabilistic algorithm that is run by the user to generate tags for a file. It takes the secret key k and a file F as input and returns the set of tags T for file F.
- **Challenge** is a probabilistic algorithm that is run by the user to generate a challenge. It takes the security parameter κ as input and returns the challenge *chal*.
- **ProofGen** is a deterministic algorithm that is run by the cloud server in order to generate a proof of possession. It takes the blocks of file F and the set of tags T as input and returns a proof of possession R for the challenged blocks in F.
- **ProofVerify** is a deterministic algorithm that is run by the user in order to evaluate a proof of possession. It takes his secret key k, the challenge *chal* and the proof of possession R as input, and returns whether the proof is a correct proof of possession for the blocks challenged by *chal*.

2.3. Security requirements

In cloud storage, the cloud server is not fully-trusted since it is self-interested and might hide data corruption incidents to maintain its reputation. So a practical RDPC protocol should be secure against the internal attacks a cloud server can launch, namely replace attack, forge attack, replay attack and deletion attack [14].

- Replace attack: the server can replace a challenged and corrupted pair of data block and tag (F_j, T_j) with another valid pair of data block and its corresponding tag (F_i, T_i), if F_j or T_j has been deleted.
- Forge attack: the server enables us to forge a valid tag on some data block to deceive the users.
- Replay attack: the server is able to generate a valid proof R from previous proofs, without accessing the stored data.
- Deletion attack: the server may generate a valid proof R making use of the tags T or other information, even the user's entire file has been deleted.