SCIENCE CHINA
Information Sciences

• LETTER •

March 2016, Vol. 59 039101:1–039101:3
doi: 10.1007/s11432-016-5519-7

Public verifiability for shared data in cloud storage with a defense against collusion attacks

Zhonghua WANG*, Zhen HAN & Jiqiang LIU

School of Computer and Information Technology, Beijing Jiaotong University, Beijing 100044*, China*

Received September 29, 2015; accepted November 24, 2015; published online January 18, 2016

Citation Wang Z H, Han Z, Liu J Q. Public verifiability for shared data in cloud storage with a defense against collusion attacks. Sci China Inf Sci, 2016, 59(3): 039101, doi: 10.1007/s11432-016-5519-7

Dear editor,

With the rapid development of cloud computing, data storage and sharing services have been widely used. Users can easily work together as a group by sharing data with each other. To provide shared data integrity and availability of remote cloud storage, a number of schemes [1–3] have been proposed with the support of dynamic data and public verifiability. Nevertheless, these schemes [1–3] rarely consider the problem of group user revocation. Recently, researchers have begun to discuss it and proposed the following schemes. Wang et al. [4] presented a public integrity verification scheme based on proxy re-signatures. Yuan and Yu [5] proposed a scheme with secure group user revocation. Jiang et al. [6] provided a public integrity verification scheme with secure group user revocation to resist collusion attacks.

Unfortunately, these schemes [4–6] are still not secure against collusion between a revoked group user and the cloud server. To achieve secure group user revocation and resist collusion attacks, we present a new public integrity verification scheme. Based on a Merkle hash tree (MHT) structure and proxy re-signatures, our scheme provides enhanced security against collusion attacks.

Methodology. A cloud storage system model is composed of three major entities: a *cloud server, group users* and a *third party auditor* (TPA). Group users include many general users and a *group manager* (i.e., data owner) who is the owner of the shared data and manages the membership of the general users. The TPA is in charge of verifying the integrity of the shared data.

We assume that data owner A's private/public key pair is $sk_A = \pi_A$, $pk_A = g^{\pi_A}$ and the group general user U_j's private/public key pair is $sk_j = \pi_j$, $pk_A = g^{\pi_j}$, where g is a generator of group $\mathbb{G}_1$ with prime order p. At the beginning, given sk_A, secret ϵ_0, and data block m_k, A produces the signature of m_k as: $\delta_k = (\mathrm{H}(m_k)\omega^{m_k})^{\pi_A \epsilon_0}$, where H is a hash function and ω is another $\mathbb{G}_1$ generator. Then, A generates a root R based on the MHT construction, where the leaf nodes of the tree are an ordered set of BLS (Boneh-Lynn-Shacham) [7] hashes of $\mathrm{H}(m_k)$. Afterwards, A signs the root R with sk_A as $sig_{sk_A}(\mathrm{H}(R)) \leftarrow (\mathrm{H}(R))^{\pi_A}$.

When U_j modifies a shared data block, U_j first computes the corresponding signature ${\delta_i}' = (\mathrm{H}({m_i}')\omega^{{m_i}'})^{\pi_j \epsilon_j}$ for ${m_i}'$ with π_j and secret ϵ_j. U_j then sends update information $update = (i, {m_i}', {\delta_i}')$ to the cloud server. Upon receiving the messages, the cloud server firstly replaces m_i and δ_i with ${m_i}'$ and ${\delta_i}'$ respectively, then generates the value of new root R' with ${m_i}'$. Finally, it sends $P_{\mathrm{update}} = \{\mathrm{H}(m_i), \Omega_i, sig_{sk}(\mathrm{H}(R)), R'\}$ to U_j, where Ω_i denotes the node siblings on the

* Corresponding author (email: wangzhonghua@bjtu.edu.cn)
The authors declare that they have no conflict of interest.

path from the leaves $\{\mathrm{h}(\mathrm{H}(m_i))\}$ to the root R of the MHT, where h is a cryptographic hash function. To reduce the computation and communication overhead of the user and the cloud server, we adopt a different method to allow U_j to modify l ($l \geqslant 2$) data blocks simultaneously. The detailed description will be shown in our supplementary file.

When A revokes a group general user U_t, every signature generated by user U_t needs to be recomputed. Firstly, A receives $sig_{sk}(\mathrm{H}(R))$ of current root R from the cloud server, then A confirms the identity of $sig_{sk}(\mathrm{H}(R))$ by checking $e(sig_{sk}(\mathrm{H}(R)), g) \stackrel{?}{=} e(\mathrm{H}(R), g^{\pi_t})$, where g^{π_t} represents U_t's public key. If it is true, A will recompute $sig_{sk_A}(\mathrm{H}(R))$ and send it to the cloud server; otherwise, it indicates the root R is signed by some valid group user. Secondly, A computes $\frac{\pi_A(\epsilon_0+\rho)}{\pi_t\epsilon_t} (\bmod\ p)$ and sends it to the cloud server, $\rho \in \mathbb{Z}_p^*$, where ϵ_t is U_t's secret. Finally, A generates $g^{\pi_A(\epsilon_0+\rho)}$ and $\frac{\epsilon_0+\rho}{\epsilon_0} (\bmod\ p)$, and then sends them to the valid group users and the TPA. After they receive the message, the previous pubic information $g^{\pi_A\epsilon_0}$ will be discarded. When A or the valid group users want to check whether the current shared data is intact, the TPA will perform the verification process as follows:

(1) Challenge message generation. Firstly, the TPA generates $g^{r'}$ and $g^{(\frac{\epsilon_0+\rho}{\epsilon_0})r'}$, where r' is a random element in $\mathbb{Z}_p^*$. Next, the TPA picks a random c-element subset I of set $[1,n]$, where n is the total number of shared data blocks. For each $i \in I$, the TPA chooses a random $v_i \in \mathbb{Z}_q^*$. Finally, the TPA sends $\{i, v_i\}_{i\in I}$, $g^{r'}$ and $g^{(\frac{\epsilon_0+\rho}{\epsilon_0})r'}$ to the cloud server.

(2) Proof generation. Upon receiving the challenge messages from the TPA, the cloud server firstly computes

$$\mu = \sum_{i\in I} v_i m_i \tag{1}$$

for data blocks in the challenged set I modified by the revoked user U_t. It next computes

$$\delta_t{}' = (\delta_t)^{\frac{\pi_A(\epsilon_0+\rho)}{\pi_t\epsilon_t}} = (\mathrm{H}(m_i)\omega^{m_i})^{\pi_A(\epsilon_0+\rho)} \tag{2}$$

for data blocks in the challenged set I modified by the valid group users. It then computes

$$\delta_j{}' = (\delta_j)^{\frac{\pi_A\epsilon_0}{\pi_j\epsilon_j}} = (\mathrm{H}(m_i)\omega^{m_i})^{\pi_A\epsilon_0} \tag{3}$$

for those blocks in the challenged set I created or modified by A. Nothing changes. Finally, the cloud server aggregates the above values as

$$\phi = \prod_{i\in I} \phi_i{}^{v_i}$$

$$= \prod_{i\in rev} e\Big(\delta_i, g^{r'}\Big)^{v_i} \prod_{i\notin rev} e\Big(\delta_i, g^{(\frac{\epsilon_0+\rho}{\epsilon_0})r'}\Big)^{v_i}$$

$$= \prod_{i\in I} e\Big((\mathrm{H}(m_i)\omega^{m_i})^{\pi_A(\epsilon_0+\rho)}, g^{r'}\Big)^{v_i}. \tag{4}$$

Afterwards, the cloud server generates proof $\boldsymbol{P} = \big\{\mu, \phi, \{\mathrm{H}(m_i), \Omega_i\}_{i\in I}, sig_{sk_i}(\mathrm{H}(R))\big\}$, and returns it to the TPA.

(3) Proof verification. Based on the responses from the cloud server, the TPA generates root R with $\{\mathrm{H}(m_i), \Omega_i\}_{i\in I}$ and authenticates $sig_{sk_i}(\mathrm{H}(R))$ by checking

$$e(sig_{sk_i}(\mathrm{H}(R)), g) \stackrel{?}{=} e(\mathrm{H}(R), g^{\pi_i}), \tag{5}$$

where sk_i belongs to the private-key sets of A and the valid group users. It is used for generating the signature of the current root R, where g^{π_i} is the corresponding public key. If it fails, the TPA outputs FALSE and reports it to A and the valid group users. Otherwise, the TPA checks

$$\phi \stackrel{?}{=} e\left(\prod_{i\in I}(\mathrm{H}(m_i))^{v_i}\omega^{\mu}, g^{\pi_A(\epsilon_0+\rho)r'}\right). \tag{6}$$

If it is true, the TPA believes the correctness of the shared data with user revocation; otherwise, the shared data has been corrupted. Finally, the TPA reports the verification result to A and the valid group users.

Experimental results. We evaluate the performance of our proposed scheme and the schemes proposed in [1,4–6]. The size of each block is 4 KB. The sizes of the selected shared files are 64 MB, 128 MB, 256 MB, and 512 MB. To evaluate the influence of the group size, we vary the revoked user number from 50 to 450. Lastly, the selected block number $c = 460$.

The comparison of integrity verification time cost for the TPA and the cloud server in [1,4,5], for when the data owner initially uploads the shared data to the cloud server, is almost equal, as shown in Figure 1(a). In this letter, the experiment results related to [6] are not shown. An explanation is provided in our supplementary file. With regard to our scheme, the verification time is about seven to nine times as long as that of [1, 4, 5]. However, as depicted in Figure 1(b), as the size of the data block increases, the computation cost of our scheme could be decreased significantly.

The computation and communication costs, for when a single group user modifies a multi-block, are depicted in Figure 1 (c) and (d), respectively. It can be seen that the costs of the user and the cloud server in the data modification process could

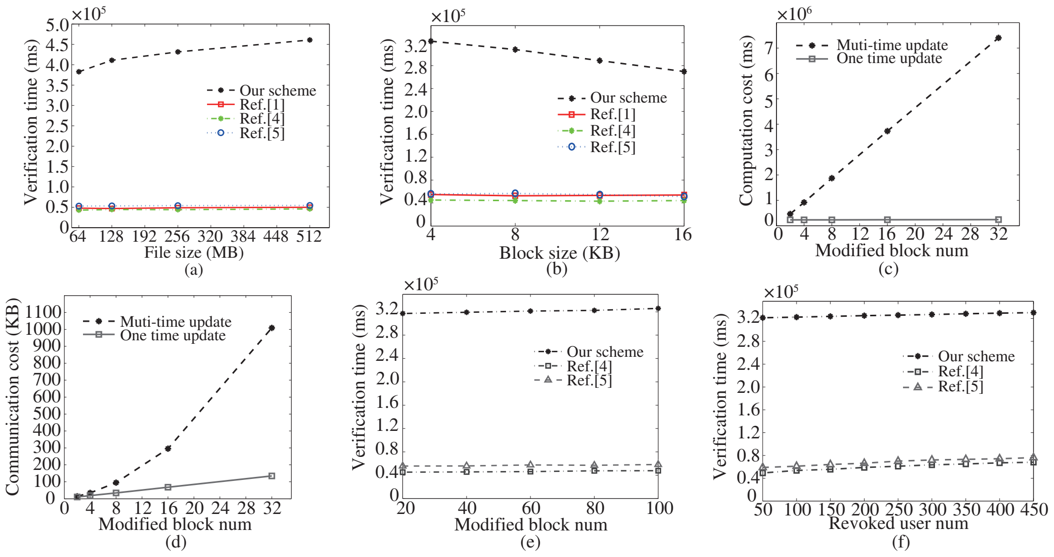

Figure 1 (Color online) Experimental results. (a) Data block of the same size (block size = 4 KB, c = 460); (b) data blocks of different size (file size = 8 MB, c = 460); (c) computation cost (file size = 8 MB, block size = 4 KB); (d) communication cost (file size = 8 MB, block size = 4 KB); (e) single revoked user modifies multi-block (file size = 8 MB, block size = 4 KB, c = 460); (f) multi-user modify multi-block (file size = 8 MB, block size = 4 KB, c = 460).

be reduced significantly with the method proposed in our supplementary file.

The results of the verification cost between [4,5] and our proposed scheme, for when a revoked user modifies a multi-block, are shown in Figure 1(e). With the increase in the amount of shared data modified by the revoked user, it should be noticed that the verification time in our proposed scheme is the longest, which is a problem we have to face.

The run time of the integrity verification process, for when the data owner revokes multi-user and each revoked user is supposed to modify different data blocks, is demonstrated in Figure 1(f). Obviously, the run time of [4,5] and our proposed scheme will increase with the number of revoked users. Nevertheless, the growth rate of our proposed scheme is the least. Therefore, we believe that the verification time in [4,5] will grow rapidly with the increase in the number of revoked users. In conclusion, based on the secure multi-user revocation, compared with [4–6], our proposed scheme is feasible and the total cost is acceptable in the cloud environment.

Conclusion. In this letter, we propose a novel public integrity verification scheme for the dynamic shared data that defends against collusion attacks. Compared with existing schemes, we prove, with the theoretical analysis in our supplementary file, that the proposed scheme provides enhanced security against a collusion attack. Furthermore, based on the secure multi-user revocation, the experimental results demonstrate that our proposed scheme is feasible and the total cost is acceptable in the cloud environment.

Acknowledgements This work was supported by Research Fund for the Doctoral Program of Higher Education of China (Grant No. 20120009110007), Program for Changjiang Scholars and Innovative Research Team in University (Grant No. IRT201206), and Program for New Century Excellent Talents in University (Grant No. NCET-11-0565).

Supporting information The supporting information is available online at info.scichina.com and link.springer.com. The supporting materials are published as submitted, without typesetting or editing. The responsibility for scientific accuracy and content remains entirely with the authors.

References

1 Kwon O, Koo D Y, Shin Y J, et al. A secure and efficient audit mechanism for dynamic shared data in cloud storage. Sci World J, 2014, 2014: 1–16
2 Wang B Y, Li B C, Li H. Oruta: privacy-preserving public auditing for shared data in the cloud. IEEE Trans Cloud Comput, 2014, 2: 43–56
3 Wang B Y, Li B C, Li H. Knox: privacy-preserving auditing for shared data with large groups in the cloud. In: Proceedings of 10th International Conference on Applied Cryptography and Network Security, Singapore, 2012. 507–525
4 Wang B Y, Li B C, Li H. Panda: public auditing for shared data with efficient user revocation in the cloud. IEEE Trans Serv Comput, 2013, 8: 92–106
5 Yuan J W, Yu S C. Public integrity auditing for dynamic data sharing with multi-user modification. In: Proceedings of 33rd Annual IEEE International Conference on Computer Communications, Toronto, 2014. 2121–2129
6 Jiang T, Chen X F, Ma J F. Public integrity auditing for shared dynamic cloud data with group user revocation. IEEE Trans Comput, 2015, PP: 1
7 Boneh D, Lynn B, Shacham H. Short signatures from the weil pairing. J Cryptol, 2004, 17: 297–319