On the Security of a Lightweight Cloud Data Auditing Scheme

Reyhaneh Rabaninejad[a], Maryam Rajabzadeh Asaar[b], Mahmoud Ahmadian Attari[a], Mohammad Reza Aref[c]

[a]*Department of Electrical Engineering, K. N. Toosi University of Technology*
[b]*Department of Electrical and Computer Engineering, Science and Research Branch, Islamic Azad University*
[c]*Department of Electrical Engineering, Sharif University of Technology*

Abstract

In cloud storage service, public auditing mechanisms allow a third party to verify integrity of the outsourced data on behalf of data users without the need to retrieve data from the cloud server. Recently, Shen et al. proposed a new lightweight and privacy preserving cloud data auditing scheme which employs a third party medium to perform time-consuming operations on behalf of users. The authors have claimed that the scheme meets the security requirements of public auditing mechanisms. In this paper, we propose two attacks against Shen et al.'s scheme. In the first attack, an active adversary who is involved in the protocol, can forge a valid authenticator on an arbitrarily modified data block. In the second attack, the dishonest cloud server arbitrarily manipulates the received data blocks, and in both attacks data manipulation is not detected by the auditor in the verification phase. Accordingly, the scheme is insecure for cloud storage auditing.

Keywords: Cloud storage, public auditing, privacy preserving, security analysis.

1. Introduction

The notion of provable data possession (PDP) first proposed by Ateniese et al. [1], is used to ensure integrity of the data remotely stored at a cloud server. In PDP, the users process their data to generate verifiable authenticators which are outsourced along with the data to the cloud. In publicly verifiable PDP schemes, a public verifier who has enough resources and expertize, provides data verification services to users. To audit data integrity, the public verifier

Email addresses: `rabaninejad@ee.kntu.ac.ir` (Reyhaneh Rabaninejad), `asaar@srbiau.ac.ir` (Maryam Rajabzadeh Asaar), `mahmoud@eetd.kntu.ac.ir` (Mahmoud Ahmadian Attari), `aref@sharif.edu` (Mohammad Reza Aref)

Preprint submitted to Journal of LaTeX Templates *October 29, 2017*

challenges the server by randomly choosing a small set of data authenticators and verifies the proof that the server returns [2].

Researchers have explored different important aspects of remote data auditing. As users may frequently update the outsourced data, supporting efficient *dynamic* data operations is an important issue in remote data auditing and several PDP schemes have considered this problem [3, 4, 5, 6, 7, 8]. To make PDP protocols resistant against exposure of user's secret keys, *key exposure resistant* schemes were proposed in recent years [9, 10]. *Data privacy* is also another important aspect in publicly verifiable PDP schemes. That is, the public verifier is only trusted to check data integrity on behalf of users and should not learn any information of the data content, as the users may store sensitive data on the cloud. To address this problem, Wang et al. proposed random masking technique [11]. Yu et al. also designed an ID-based PDP scheme with zero-knowledge data privacy [12]. Wang et al. considered a scenario that a group of users *share* the data outsourced to the cloud [13]. The scheme employs ring signatures to provide group user's identity privacy against public verifier. The same authors in [14], presented another scheme with shared data scenario which exploits proxy re-signatures to support efficient user revocation. However, the scheme is not secure against collusion of the server and revoked users. Yuan and Yu proposed another shared data auditing scheme with user revocation utilizing polynomial-based authentication tags [15]. However, the scheme does not provide identity privacy. Also due to the attack proposed in [16], the scheme in [15] is vulnerable to the collusion of server and revoked users. Recently, an efficient shared data auditing scheme is proposed which provides identity/data privacy and collusion resistant user revocation, simultaneously [17]. It is proved in the paper that the server by colluding to the revoked users can get no extra information and the scheme is collusion resistant. Furthermore, since the exponential term in authenticator generation can be computed *offline*, the scheme provides lightweight computation cost on the users side [17]. The papers [18, 19], also propose two other efficient schemes in which the user's computation is divided in two online/offline phases.

Shen et al. in [20] proposed a new PDP scheme which is *lightweight* and *privacy preserving*. To reduce the computation cost of generating authenticators on the user side, they introduced a third party medium (TPM) who performs time-consuming operations on behalf of users. The users only blind their data and send the blinded data to the TPM. The TPM then completes the task of generating authenticators on blinded data. Therefore, the scheme has lightweight computations on the users side. The TPM in this scheme, also plays the role of public verifier and audits the cloud data integrity on behalf of users. Furthermore, since the TPM only accesses the blinded data, no information of the data content is leaked to the TPM and data privacy is preserved by the scheme. The authors have also claimed that the scheme provides *auditing soundness*. That is, the untrusted cloud server can pass the TPM's verification if and only if he has preserved the stored data intact.

In this paper, we propose two attacks against Shen et al.'s scheme. In the first attack, an *active adversary* who intercepts the TPM–Cloud line, can forge a valid authenticator on an arbitrarily modified data block. Therefore, as oppose to what claimed in [20], all people (not just the authorized TPM) can upload data to the cloud. In the second attack, the *dishonest cloud server* arbitrarily manipulates the received data blocks and deceives the TPM to beleive that the data is kept intact. Accordingly, the scheme fails to achieve the property of soundness as a basic security requirement of PDP schemes.

The paper is organized as follows. In Section 2, we review Shen et al.'s public auditing protocol. Section 3 proposes two attacks against Shen et al.'s scheme and finally Section 4 concludes the paper.

2. Review of Shen et al.'s Public Auditing Protocol

In this section, we first review the system model used in [20]. Next, we will get a glimpse of the lightweight public auditing scheme proposed in [20].

2.1. System Model

As illustrated in Figure 1, three entities are involved in the protocol: cloud server, third party medium (TPM) and group users. The group users, due to lack of local storage, outsource their data to the cloud who provides low cost storage services for users. The TPM has two roles in the system. He generates authenticators on blinded data blocks and also verifies the integrity of cloud data on behalf of users. The data owner (also known as the group manager) creates shared data and uploads it to the cloud. All users in the group can access shared data stored at the cloud server. The data owner, in order to upload the data, blinds data blocks and sends them to the TPM. The TPM generates authenticators on blinded data blocks and sends the (blinded-block,authenticator) pair to the cloud. The cloud recovers the real data and the related real authenticator and stores them in his storage.

To audit the integrity of cloud data, the TPM sends a challenge to the cloud server. Based on the challenged blocks and their authenticators, the cloud generates a proof and sends it to the TPM. Finally, the TPM verifies the received proof to check the data correctness.

2.2. Protocol Review

Let G_1, G_2 be multiplicative cyclic groups of prime order p, with g_1 and $u_1, ..., u_s$ as generators of G_1. Let $e : G_1 \times G_1 \rightarrow G_2$ be a bilinear map. Also three hash functions $H_1 : \{0,1\}^* \times G_1 \rightarrow Z_p^*$, $H_2 : \{0,1\}^* \rightarrow G_1$ and $h : G_1 \rightarrow Z_p^*$ and a pseudo-random function $f : Z_p^* \times Z_p^* \rightarrow Z_p^*$ are used in the scheme. The data file F is divided into n blocks $(m_1, m_2, ..., m_n)$ and each block m_i is also divided into s sectors $(m_{i,1}, m_{i,2}, ..., m_{i,s})$. Shen et al.'s protocol consists of seven algorithms which are reviewed in the following:

Setup(1^k)**.** In this algorithm, the group manager generates the public-private key pairs, the authorization and the secret seed.

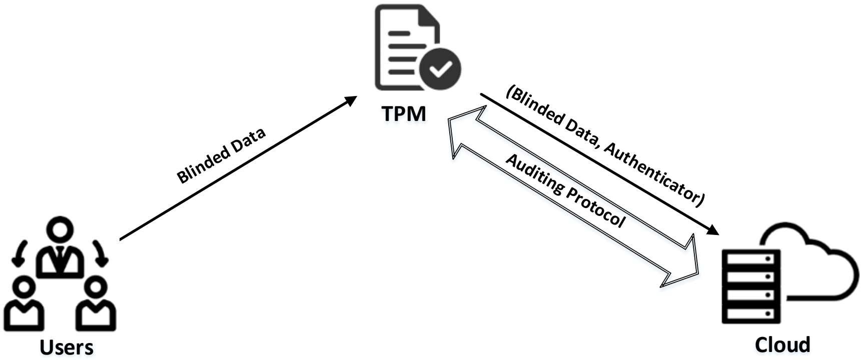

Figure 1: The system model

1. *Group's and TPM's public-private key pairs:* The Group's public-private key pair is $(pk_g = g^x, sk_g = x)$, where x is chosen randomly from Z_p. Also, the group manager picks a random element $r_0 \in Z_p$ to compute $Y_0 = g^{r_0}$ and $\beta_0 = r_0 + x.H_1(ID_{group} \parallel time_1 \parallel time_2, Y_0) mod p$, where ID_{group}, $time_1$ and $time_2$ are the group manager's identity, the start time and the end time, respectively. Finally, β_0 is set as the TPM's private key and his public key is published as $pk_{TPM} = (g^{\beta_0}, u_1^{\beta_0}, u_2^{\beta_0}, ..., u_s^{\beta_0})$.
2. *TPM's authorization:* The group manager picks random $r_1 \in Z_p$ and computes $Y_1 = g^{r_1}$ and $\beta_1 = r_1 + x.H_1(ID_{group} \parallel ID_{TPM} \parallel time_1 \parallel time_2, Y_1) mod p$, where ID_{TPM} is the TPM's identity. The TPM's authorization is set as $\{(ID_{group}, ID_{TPM}, time_1, time_2), Y_1, \beta_1)\}$. The private key β_0 and the authorization $\{(ID_{group}, ID_{TPM}, time_1, time_2), Y_1, \beta_1)\}$ are sent to the TPM.
3. *Secret seed:* The group manager chooses random seed $k_1 \in Z_p$ as the input secret key of the pseudo-random function and sends it to the cloud and the group users.

DataBlind(m_i)**.** To blind data block m_i, the user first employs the secret seed k_1 to compute the blinding factor $\alpha_i = f_{k_1}(i, name)$, where $name \in_r Z_p$ is the file identifier. The blinded sectors are computed as $m'_{i,j} = m_{i,j} + \alpha_i$ for $j \in [1, s]$. Finally, the blinded block $m'_i = (m'_{i,1}, m'_{i,2}, ..., m'_{i,s})$ is sent to the TPM.

AuthGen(β_0, m'_i)**.** The TPM generates the authenticator σ'_i of the blinded data block m'_i with his private key β_0 as follows:

$$\sigma'_i = \left(H_2(i) \prod_{j=1}^{s} u_j^{m'_{i,j}} \right)^{\beta_0} \tag{1}$$

Then the TPM sends the pair (m'_i, σ'_i) along with his authorization to the cloud. Upon receiving the authorization, the cloud first checks whether the current time is between $time_1$ and $time_2$ and then checks the correctness of authorization using Equation 2.

$$g^{\beta_1} = Y_1.pk_g^{H_1(ID_{group}\|ID_{TPM}\|time_1\|time_2,Y_1)} \tag{2}$$

If the equation holds, the cloud runs AuthVerify. Otherwise, he deletes the pair (m'_i, σ'_i) and aborts.

AuthVerify$(pk_{TPM}, \sigma'_i, m'_i)$**.** The cloud verifies the correctness of the authenticator σ'_i via the following equation:

$$e(\sigma'_i, g) = e(H_2(i)\prod_{j=1}^{s} u_j^{m'_{i,j}}, g^{\beta_0}) \tag{3}$$

If the equation holds, the cloud performs the algorithm Recovery. Otherwise, he tells the user that σ'_i is incorrect.

Recovery$(k_1, pk_{TPM}, \sigma'_i, m'_i)$**.** To un-blind the data block and its related authenticator, the cloud first computes the blinding factor $\alpha_i = f_{k_1}(i, name)$ and computes the real data sectors as $m_{i,j} = m'_{i,j} - \alpha_i$ for $j \in [1, s]$. He then recovers the real authenticator σ_i using Equation 4.

$$\sigma_i = \sigma'_i. \prod_{j=1}^{s} (u_j^{\beta_0})^{-\alpha_i} \tag{4}$$

where $u_1^{\beta_0}, u_2^{\beta_0}, ..., u_s^{\beta_0}$ are parts of the TPM's public key. Finally, the cloud stores the real pair (m_i, σ_i) in his storage.

ProofGen$(F, \Phi, chal)$**.** In order to audit the data integrity, the TPM chooses a random c-element subset $I \subset [1, n]$ as the block indices to be challenged in the auditing process. Then for each $i \in I$, the TPM chooses a random value $v_i \in Z_p$ and sends the auditing challenge $chal = \{(i, v_i)\}_{i \in I}$ along with his authorization to the cloud.

The cloud after receiving the challenge and the authorization from the TPM, first verifies the authorization's correctness as explained before. If the authorization is valid, he generates a proof of data possession as follows:

1. For $j \in [1, s]$ the cloud calculates $R_j = u_j^r \in G_1$, where $r \in_r Z_p$. He then computes $\mu_j = \sum_{i \in I} v_i m_{i,j} + rh(R_j) \in Z_p$, for $j \in [1, s]$ as the linear combination of the challenged blocks.
2. Aggregates the related authenticators as $\sigma = \prod_{i \in I} \sigma_i^{v_i}$.
3. Sends back the auditing proof $P = \{R, \mu, \sigma\}$ to the TPM, where $R = (R_1, ..., R_s)$ and $\mu = (\mu_1, ..., \mu_s)$.

ProofVerify$(pk_{TPM}, chal, P)$**.** The TPM verifies the server's proof $P = \{R, \mu, \sigma\}$ through Equation 5. If the equation holds the verification passes, otherwise it fails.

$$e(\sigma, g) = e\Bigg(\prod_{i \in I} H_2(i)^{v_i}. \prod_{j=1}^{s} (u_j^{\mu_j}.R_j^{-h(R_j)}), g^{\beta_0}\Bigg) \tag{5}$$

3. Analysis of Shen et al.'s Protocol

In this section we propose two attacks which demonstrate that Shen et al.'s protocol does not achieve two security properties as oppose to what claimed in their paper [20]. In the first attack, an active adversary who does not possess the signing private key, forges valid authenticator on an arbitrarily modified data block and outsources the forged pair (block*,authenticator*) to the cloud. In the second attack, the untrusted cloud server manipulates the received data blocks and deceives the TPM to believe that the data is intact. In the following, we explain these two attacks in detail.

3.1. Active Adversary Attack

Shen et al. have claimed that in their protocol only an authorized TPM can upload data to the cloud [20]. Here, we show that if an active adversary intercepts the message (m'_i, σ'_i) and the TPM's authorization which is sent from the TPM to the cloud, then he is able to forge a valid authenticator σ^*_i on modified block $m^*_i = m'_i + \Delta m_i$ and outsource (m^*_i, σ^*_i) along with the TPM's authorization to the cloud. More precisely, the active adversary $\mathcal{A}$ utilizes (m'_i, σ'_i) to forge the new pair (m^*_i, σ^*_i) as below:

$$\forall j \in [1, s] : m^*_{i,j} = m'_{i,j} + \Delta m_{i,j} \tag{6}$$

where $\Delta m_{i,j} \in Z_p$ is arbitrarily chosen by adversary $\mathcal{A}$ to modify each blinded data sector $m'_{i,j}$. Therefore, $m^*_{i,j}$ can be written as $m^*_{i,j} = m'_{i,j} + \Delta m_{i,j} = (m_{i,j} + \alpha_i) + \Delta m_{i,j} = (m_{i,j} + \Delta m_{i,j}) + \alpha_i$. Finally, the modified block is set to $m^*_i = (m^*_{i,1}, m^*_{i,2}, ..., m^*_{i,s})$. Furthermore, authenticator σ^*_i on block m^*_i is produced through Equation 7

$$\sigma^*_i = \sigma'_i \times \prod_{j=1}^{s} (u_j^{\beta_0})^{\Delta m_{i,j}} \tag{7}$$

where $u_1^{\beta_0}, u_2^{\beta_0}, ..., u_s^{\beta_0}$ are parts of the TPM's public key. The validity of authenticator σ^*_i on block m^*_i can be easily shown as below:

$$\begin{aligned}
\sigma^*_i &= \sigma'_i \times \prod_{j=1}^{s} (u_j^{\beta_0})^{\Delta m_{i,j}} \\
&= \left(H_2(i) \prod_{j=1}^{s} u_j^{m'_{i,j}}\right)^{\beta_0} \times \prod_{j=1}^{s} (u_j^{\beta_0})^{\Delta m_{i,j}} \\
&= \left(H_2(i) \prod_{j=1}^{s} u_j^{m'_{i,j}}\right)^{\beta_0} \times \left(\prod_{j=1}^{s} u_j^{\Delta m_{i,j}}\right)^{\beta_0} \\
&= \left(H_2(i) \prod_{j=1}^{s} u_j^{(m'_{i,j} + \Delta m_{i,j})}\right)^{\beta_0} \\
&= \left(H_2(i) \prod_{j=1}^{s} u_j^{m^*_{i,j}}\right)^{\beta_0} \qquad (8)
\end{aligned}$$

Finally, $\mathcal{A}$ sends (m_i^*, σ_i^*) along with the TPM's authorization to the cloud. The cloud, first verifies correctness of the authenticator σ_i^* via Equation 3. Since σ_i^* is a valid authenticator due to the above equalities, it passes the verification equation. Next, the server un-blinds the pair (m_i^*, σ_i^*) as below:

$$\begin{aligned} \forall j \in [1, s] : m_{i,j}^* - \alpha_i &= (m'_{i,j} + \Delta m_{i,j}) - \alpha_i \\ &= ((m_{i,j} + \Delta m_{i,j}) + \alpha_i) - \alpha_i \\ &= m_{i,j} + \Delta m_{i,j} \end{aligned} \tag{9}$$

$$\begin{aligned} \sigma_i &= \sigma_i^* . \prod_{j=1}^{s} (u_j^{\beta_0})^{-\alpha_i} \\ &= \left(H_2(i) \prod_{j=1}^{s} u_j^{m_{i,j}^*} \right)^{\beta_0} . \prod_{j=1}^{s} (u_j^{\beta_0})^{-\alpha_i} \\ &= \left(H_2(i) \prod_{j=1}^{s} u_j^{m_{i,j}^* - \alpha_i} \right)^{\beta_0} \\ &= \left(H_2(i) \prod_{j=1}^{s} u_j^{m_{i,j} + \Delta m_{i,j}} \right)^{\beta_0} \end{aligned} \tag{10}$$

Therefore, instead of $(m_{i,1}, ..., m_{i,s})$, the modified block $(m_{i,1} + \Delta m_{i,1}, ..., m_{i,s} + \Delta m_{i,s})$ and its authenticator σ_i are stored in the cloud without any problem.

3.2. Cloud Server Attack

In this subsection, we show that the cloud server can make the protocol lose the property of soundness. Specifically, the dishonest cloud server can arbitrarily manipulate the data and still pass the TPM's data integrity verification. Assume that the cloud has received the message (m'_i, σ'_i) from the TPM. The dishonest cloud can employ (m'_i, σ'_i) to generate a valid authenticator σ_i^* on an arbitrary block m_i^* using Equations 11 and 12.

$$(H_2(i))^{\beta_0} = \frac{\sigma'_i}{\prod_{j=1}^{s} (u_j^{\beta_0})^{m'_{i,j}}} = \frac{\left(H_2(i) \prod_{j=1}^{s} u_j^{m'_{i,j}} \right)^{\beta_0}}{\left(\prod_{j=1}^{s} u_j^{m'_{i,j}} \right)^{\beta_0}} \tag{11}$$

$$\begin{aligned} \sigma_i^* &= (H_2(i))^{\beta_0} \prod_{j=1}^{s} (u_j^{\beta_0})^{m_{i,j}^*} \\ &= \left(H_2(i) \prod_{j=1}^{s} u_j^{m_{i,j}^*} \right)^{\beta_0} \end{aligned} \tag{12}$$

where $u_1^{\beta_0}, u_2^{\beta_0}, ..., u_s^{\beta_0}$ are parts of the TPM's public key. First, due to Equation 11, the term $(H_2(i))^{\beta_0}$ is calculated and then it is used to generate σ_i^* in Equation 12.

It can be easily seen that σ_i^* is a valid authenticator on block m_i^*. Therefore, instead of un-blinding and storing the received pair (m_i', σ_i') as the ith data block, the cloud stores the pair (m_i^*, σ_i^*) in his storage. Since σ_i^* is a valid authenticator on block m_i^*, the cloud can produce a valid proof in response to the TPM's challenge and pass the verification; although the data integrity has been broken. Accordingly, the scheme fails to achieve the property of soundness.

4. Conclusion

In this paper, we reviewed a recently proposed lightweight and privacy preserving cloud auditing scheme by Shen et al. and analyzed its security. By presenting active adversary attack and cloud server attack, we demonstrated the vulnerability of the scheme. Specifically, both the active adversary and the dishonest cloud server can arbitrarily modify data blocks without being detected by the auditor. These attacks show that the scheme is insecure and does not meet auditing soundness as a basic security requirement of PDP schemes.

Acknowledgements

This research is supported by Iran National Science Foundation (INSF) under Grant No. 92.32575.

References

[1] G. Ateniese, R. Burns, R. Curtmola, J. Herring, L. Kissner, Z. Peterson, D. Song, Provable data possession at untrusted stores, in: Proceedings of the 14th ACM conference on Computer and communications security, ACM 2007, ACM, Alexandria, Virginia, USA, 2007, pp. 598–609.

[2] M. Sookhak, H. Talebian, E. Ahmed, A. Gani, M. K. Khan, A review on remote data auditing in single cloud server: Taxonomy and open issues, Journal of Network and Computer Applications 43 (2014) 121–141.

[3] Y. Zhu, G.-J. Ahn, H. Hu, S. S. Yau, H. G. An, C.-J. Hu, Dynamic audit services for outsourced storages in clouds, IEEE Transactions on Services Computing 6 (2) (2013) 227–238.

[4] C. C. Erway, A. Küpçü, C. Papamanthou, R. Tamassia, Dynamic provable data possession, ACM Transactions on Information and System Security (TISSEC) 17 (4) (2015) 213–222.

[5] N. Garg, S. Bawa, Rits-mht: Relative indexed and time stamped merkle hash tree based data auditing protocol for cloud computing, Journal of Network and Computer Applications 84 (2017) 1–13.

[6] M. Sookhak, A. Gani, M. K. Khan, R. Buyya, Dynamic remote data auditing for securing big data storage in cloud computing, Information Sciences 380 (2017) 101–116.

[7] C. Lin, Z. Shen, Q. Chen, F. T. Sheldon, A data integrity verification scheme in mobile cloud computing, Journal of Network and Computer Applications 77 (2017) 146–151.

[8] D. Cash, A. Küpçü, D. Wichs, Dynamic proofs of retrievability via oblivious ram, Journal of Cryptology 30 (1) (2017) 22–57.

[9] J. Yu, K. Ren, C. Wang, V. Varadharajan, Enabling cloud storage auditing with key-exposure resistance, IEEE Transactions on Information forensics and security 10 (6) (2015) 1167–1179.

[10] J. Yu, H.-Q. Wang, Strong key-exposure resilient auditing for secure cloud storage, IEEE Transactions on Information Forensics and Security 12 (8) (2017) 1931 – 1940.

[11] C. Wang, S. S. Chow, Q. Wang, K. Ren, W. Lou, Privacy-preserving public auditing for secure cloud storage, IEEE Transactions on computers 62 (2) (2013) 362–375.

[12] Y. Yu, M. H. Au, G. Ateniese, X. Huang, W. Susilo, Y. Dai, G. Min, Identity-based remote data integrity checking with perfect data privacy preserving for cloud storage, IEEE Transactions on Information Forensics and Security 12 (4) (2017) 767–778.

[13] B. Wang, B. Li, H. Li, Oruta: privacy-preserving public auditing for shared data in the cloud, IEEE transactions on cloud computing 2 (1) (2014) 43–56.

[14] B. Wang, B. Li, H. Li, Panda: public auditing for shared data with efficient user revocation in the cloud, IEEE Transactions on services computing 8 (1) (2015) 92–106.

[15] J. Yuan, S. Yu, Public integrity auditing for dynamic data sharing with multiuser modification, IEEE Transactions on Information Forensics and Security 10 (8) (2015) 1717–1726.

[16] Y. Yu, Y. Li, J. Ni, G. Yang, Y. Mu, W. Susilo, Comments on public integrity auditing for dynamic data sharing with multiuser modification, IEEE Transactions on Information Forensics and Security 11 (3) (2016) 658–659.

[17] R. Rabaninejad, M. Ahmadian Attari, M. Rajabzadeh Asaar, M. R. Aref, Corpa: A novel efficient shared data auditing protocol in cloud storage, http://eprint.iacr.org/2017/941.

[18] J. Li, L. Zhang, J. K. Liu, H. Qian, Z. Dong, Privacy-preserving public auditing protocol for low-performance end devices in cloud, IEEE Transactions on Information Forensics and Security 11 (11) (2016) 2572–2583.

[19] Y. Wang, Q. Wu, B. Qin, S. Tang, W. Susilo, Online/offline provable data possession, IEEE Transactions on Information Forensics and Security 12 (5) (2017) 1182–1194.

[20] W. Shen, J. Yu, H. Xia, H. Zhang, X. Lu, R. Hao, Light-weight and privacy-preserving secure cloud auditing scheme for group users via the third party medium, Journal of Network and Computer Applications 82 (2017) 56–64.