Cryptanalysis and Countermeasures on Privacy-Preserving Public Auditing for Regenerating-Code-Based Cloud Storage

Mei-Ping Liu, Rui Jiang [a] and Hua-Feng Kong [b]
School of Information Science and Engineering, Southeast University, Nanjing, China
Key Lab of Information Network Security Ministry of Public Security, Shanghai, China
E-mail: 741027796@qq.com, [a]R.Jiang@seu.edu.cn, [b]robin_kong@126.com

As an on-demand of computational resource, cloud computing appears to be part of a larger upward trend. Data owners can store outsourced data in the cloud and enjoy the high quality service. However, it brings new challenges, including the potential risks of the integrity of data, confidentiality of data, and mutual authentication. Recently, Liu et al. proposed a privacy-preserving public auditing for regenerating-code-based cloud storage and claimed that their scheme was secure and highly efficient. Unfortunately, we find out that there are risks of two attacks and a shortage in the scheme . In this paper, we first point out that Liu et al.'s scheme suffers from two attacks. The first attack is that it cannot resist against forgery attack, which makes cloud servers store the wrong data without being detected. The second attack is that the efficiency of data recovery is easily reduced by attackers since cloud servers may store many useless redundant data. In addition, there is a drawback in Liu et al.'s scheme . The drawback is that the data in the cloud cannot be shared if the data owner uses a private key to encrypt it.

Keywords: Public Auditing; Integrity; Privacy-Preserving; Regenerating-Code.

1. Introduction

There are challenges of guaranteeing the confidentiality and the integrity of the outsourced data on the cloud computing system. In recent years, many mechanisms have been proposed to protect the cloud data from being corrupted. Ateniese et al. proposed a provable data possession model to prove the intactness of data [1]. Meanwhile, Juels and Kaliski proposed a model called Proof of Retrievability where there is a verifier via a concise proof proving the availability of users' file. However, the data cannot be recovered when corruption happens. Based on a solution that data is stored redundantly across multi-servers, many mechanisms are explored with different redundancy schemes, such as replication [3], erasure codes [4] and regenerating codes.

To reduce the storage expense of replication and repair bandwidths of erasure codes, Chen et al. proposed the scheme with regenerating-code-based cloud storage [5], transforming the single-server CPOR scheme to the regenerating-code-scenario. Meanwhile, Chen and Lee [6] proposed a scheme to implement a data integrity protection scheme for FMSR [7]-based cloud storage. Nonetheless, these schemes are only fit for private audit and repair. To overcome these restrictions, Shacham et al. proposed a public auditing scheme suitable for multi-server setting [8]. However, the data privacy cannot be guaranteed. Hence, Yang and Jia presented a public PDP scheme to provide data privacy through combining the cryptography method with the bilinearity property of bilinear pairing [9]. To achieve both public audit and privacy preserving using regenerating codes with functional repair strategy [10], Liu et al. proposed the privacy-preserving public auditing for regenerating- code-based cloud storage [11]. The authors claimed that the scheme was highly efficient and secure.

However, through our security cryptanalysis, we find out that Liu et al.'s scheme is vulnerable to attack. There are two attacks and a drawback. At the first attack, the adversary can launch the forgery attack by adding any data to the regular data blocks which results in destroying the correction of blocks without being detected. At the second attack, the adversary can make redundant blocks, which greatly reduce the efficiency of reconstructing the original file. In addition, the drawback of Liu et al.'s scheme is that the data in cloud cannot be shared if data owners encrypt the data with their own private keys. In order to solve the above security problems, we put forward some countermeasures. To withstand above two attacks, we make signatures for all coded blocks in cloud servers instead of just for the native blocks. To support data sharing, we can encrypt the data at the beginning. Thus, the data owners can realize data sharing between authorized data users.

2. Brief Review of Liu et al.'s Scheme

2.1. *Setup*

The Setup phase has four processes, KeyGen, Delegation, SigAndBlockGen and Uploading. At KeyGen process, The data owner keeps the (x, y, ssk) secret while makes (pk_x, pk_y, spk) public where $pk_x = g^x, pk_y = g^y$. At Delegation process, data owner sends x to the proxy using proxy's public key. At Sig And BlockGen process, the owner chooses $\{ID, u, \omega_1, \omega_2, \cdots, \omega_m\}$to generate a file tag t. The data owner properly augments the native m blocks as Eq.(1),

combines the augmented native blocks to generate coded blocks Ψ as Eq.(2) and generates authenticators Φ as Eq.(3). At Uploading process, the data owner uploads $\{\Psi, \Phi, t\}$ to the cloud server.

$$w_i = (\bar{\omega}_{i1}, \bar{\omega}_{i2}, ..., \bar{\omega}_{is}, \overbrace{0, \cdots, 0, 1, 0, \cdots, 0}^{m}) \in \mathrm{GF(p)}^{s+m} \tag{1}$$

$$v_{ij} = \sum_{\lambda=1}^{m} \varepsilon_{ij\lambda} w_\lambda \in GF(p)^{s+m} \tag{2}$$

$$\sigma_{ijk} = H(ID||i||j||k)^x \left(u^{v_{ijk}} \prod_{\lambda=1}^{m} \omega_\lambda^{\varepsilon_{ij\lambda}}\right)^y \tag{3}$$

2.2. *Audit*

For each of the n servers, TPA verifies the possession of α coded blocks by randomly checking samples of segments of every block and performing a batch verification. The Audit phase has three processes, Challenge, ProofGen and Verify, which are illustrated as Figure 1.

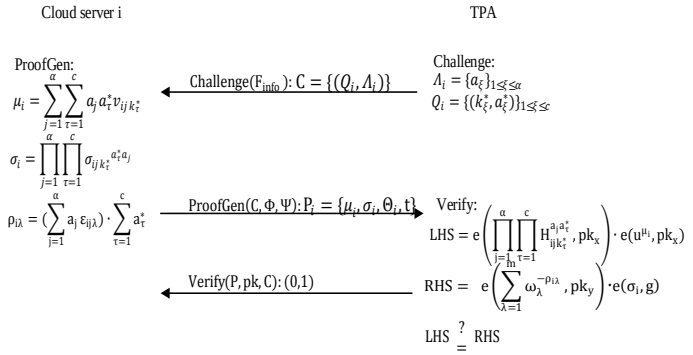

Fig. 1 Audit phase

At Challenge process, the TPA generates challenge C and sends it to the cloud server i. Receiving the challenge C from TPA, cloud server i starts the ProofGen process, computes proof P_i and responses it to TPA. At Verify process, with the receiving proof P_i, the TPA checks whether or not $LHS = RHS$. If the equation holds, the TPA outputs 1 and believes that the data in server i is intact. Otherwise, the TPA outputs 0 and sends an alert to proxy to start a Repair phase.

3. Attacks and a Drawback

3.1. *Attack I: data forgery attack*

On the basis of Dolev-Yao model [13], the outside attacker is able to intercept the coded data during the communication between the cloud servers and the data

owners, and produce a forged data to pollute the correct data. The details of the attack are as follows.

At Uploading process of Setup phase, the attacker interrupts the data $\{v_{ij}, \sigma_{ijk}, t\}$ that the data owner sends to the cloud server i. Then the attacker forges data v'_{ij}, computes $v^*_{ij} = v'_{ij} + v_{ij}$ and uploads $\{v^*_{ij}, \sigma_{ijk}, t\}$ to the server i. The adversary saves the v'_{ij} in his local storage. At Challenge process of Audit phase, the attacker eavesdrops on the $C = \{(Q_i, \Lambda_i)\}$, and computes $\mu_i', \rho_{i\lambda}'$ as ProofGen process in Figure 1 using v'_{ij}. When server i responses to the TPA with the proof P_i, the adversary modifies μ_i^* into $\mu_i^* - \mu_i'$, modifies Θ_i^* into $\Theta_i^* - \Theta_i'$ which is illustrated as Figure 2.

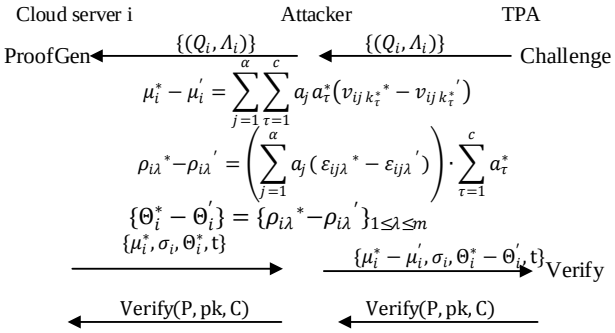

Fig. 2 Attack I at Audit phase

At Verify process of Audit phase, TPA receives the modified proof P and checks if the equation $LHS = RHS$ in Figure 1 holds.

$$\mu_i^* - \mu_i' = \sum_{j=1}^{\alpha}\sum_{\tau=1}^{c} a_j a_\tau^* \left(v_{ijk_\tau^*}^* - v_{ijk_\tau^*}'\right) = \sum_{j=1}^{\alpha}\sum_{\tau=1}^{c} a_j a_\tau^* v_{ijk_\tau^*} = \mu_i$$

$$\rho_{i\lambda}^* - \rho_{i\lambda}' = \left(\sum_{j=1}^{\alpha} a_j \left(\varepsilon_{ij\lambda}^* - \varepsilon_{ij\lambda}'\right)\right) \cdot \sum_{\tau=1}^{c} a_\tau^* = \left(\sum_{j=1}^{\alpha} a_j \varepsilon_{ij\lambda}\right) \cdot \sum_{\tau=1}^{c} a_\tau^* = \rho_{i\lambda}$$

$$LHS = e\left(\prod_{j=1}^{\alpha}\prod_{\tau=1}^{c} H_{ijk_\tau^*}^{a_j a_\tau^*}, pk_x\right) \cdot e\left(u^{\mu_i^* - \mu_i'}, pk_y\right)$$

$$= e\left(\prod_{j=1}^{\alpha}\prod_{\tau=1}^{c} H_{ijk_\tau^*}^{a_j a_\tau^*}, pk_x\right) \cdot e\left(u^{\mu_i}, pk_y\right)$$

$$RHS = e\left(\sum_{\lambda=1}^{m} \omega_\lambda^{-\left(\rho_{i\lambda}^* - \rho_{i\lambda}'\right)}, pk_y\right) \cdot e(\sigma_i, g) = e\left(\sum_{\lambda=1}^{m} \omega_\lambda^{-\rho_{i\lambda}}, pk_y\right) \cdot e(\sigma_i, g)$$

$$= e\left(\sum_{\lambda=1}^{m} \omega_\lambda^{-\rho_{i\lambda}}, pk_y\right) \cdot e\left(\prod_{j=1}^{\alpha}\prod_{\tau=1}^{c} H_{ijk_\tau^*}^{a_\tau^* a_j}, pk_x\right) \cdot e\left(u^{\sum_{j=1}^{\alpha} a_j \mu_{ij}}, pk_y\right)$$

$$\cdot e\left(\sum_{\lambda=1}^{m} \omega_\lambda^{\rho_{i\lambda}}, pk_y\right)$$

$$= e\left(\prod_{j=1}^{\alpha}\prod_{\tau=1}^{c} H_{ijk_\tau^*}^{a_j a_\tau^*}, pk_x\right) \cdot e\left(u^{\mu_i}, pk_y\right)$$

Obviously, $LHS = RHS$, so the TPA will pass the proof and the data owner believes that the data is maintained correctly. However, the coded block has been modified viciously. The reconstructed file is different from the original one if the data owner chooses the attacked block to construct it.

3.2. *Attack II: data recovery attack*

There are α efficient blocks that can be used to reconstruct the original file in a cloud server. However, through our attack, the efficient blocks reduce to one and there are $\alpha - 1$ redundant blocks. The attack procedures are shown as follows.

At Uploading process of Setup phase, the attacker interrupts the data $\{v_{ij}, \sigma_{ijk}, t\}$, and computes v_{ij}^*. Then the attacker sends $\{v_{ij}^*, \sigma_{ijk}, t\}$ to the cloud server i without being detected. As illustrated as Figure 3, when cloud server i responses to the TPA at ProofGen process of Audit phase, the attacker computes $(1/\alpha)\mu_i{}^*$, $(1/\alpha)\cdot\rho_{i\lambda}{}^*$, and sends the proof $\{(1/\alpha)\mu_i^*, \sigma_i, (1/\alpha)\Theta_i{}^*, \mathrm{t}\}$ to cloud sever i. Receiving the proof from server i, the TPA verifies whether $LHS = RHS$ or not.

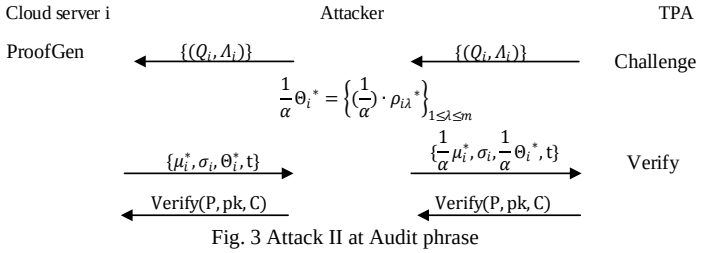

Fig. 3 Attack II at Audit phrase

$$
\begin{aligned}
(1/\alpha)\mu_i{}^* &= (1/\alpha)\textstyle\sum_{j=1}^{\alpha} a_j\left(\mu_{ij}{}^*\right) = (1/\alpha)\sum_{j=1}^{\alpha}\sum_{\tau=1}^{c} a_j a_\tau^*\left(v_{ijk_\tau^*}{}^*\right)\\
&= (1/\alpha)\textstyle\sum_{j=1}^{\alpha}\sum_{\tau=1}^{c} a_j a_\tau^*\left(v_{ijk_\tau^*} + \sum_{b=1,b\neq j}^{\alpha} v_{ibk_\tau^*}\right)\\
&= (1/\alpha)\cdot\alpha\textstyle\sum_{j=1}^{\alpha}\sum_{\tau=1}^{c} a_j a_\tau^*\left(v_{ijk_\tau^*}\right) = \mu_i\\
(1/\alpha)\cdot\rho_{i\lambda}{}^* &= (1/\alpha)\cdot\left[\textstyle\sum_{j=1}^{\alpha}(a_j\,\varepsilon_{ij\lambda} + \sum_{b=1,b\neq j}^{\alpha}\varepsilon_{ib\lambda})\right]\cdot\sum_{\tau=1}^{c} a_\tau^*\\
&= (1/\alpha)\cdot\left[\textstyle\sum_{j=1}^{\alpha}(a_j\,\varepsilon_{ij\lambda}) + (\alpha-1)\sum_{j=1}^{\alpha}(a_j\,\varepsilon_{ij\lambda})\right]\cdot\sum_{\tau=1}^{c} a_\tau^*\\
&= (1/\alpha)\cdot\alpha\cdot\textstyle\sum_{j=1}^{\alpha}(a_j\,\varepsilon_{ij\lambda})\cdot\sum_{\tau=1}^{c} a_\tau^* = \rho_{i\lambda}\\
LHS &= e\left(\textstyle\prod_{j=1}^{\alpha}\prod_{\tau=1}^{c} H_{ijk_\tau^*}^{a_j a_\tau^*}, pk_x\right)\cdot e\left(u^{(1/\alpha)\mu_i^*}, pk_x\right)\\
&= e\left(\textstyle\prod_{j=1}^{\alpha}\prod_{\tau=1}^{c} H_{ijk_\tau^*}^{a_j a_\tau^*}, pk_x\right)\cdot e\left(u^{\mu_i}, pk_x\right)\\
RHS &= e\left(\textstyle\sum_{\lambda=1}^{m}\omega_\lambda^{-(1/\alpha)\cdot\rho_{i\lambda}{}^*}, pk_y\right)\cdot e(\sigma_i, g) = e\left(\textstyle\sum_{\lambda=1}^{m}\omega_\lambda^{-\rho_{i\lambda}}, pk_y\right)\cdot e(\sigma_i, g)\\
&= e\left(\textstyle\sum_{\lambda=1}^{m}\omega_\lambda^{-\rho_{i\lambda}}, pk_y\right)\cdot e\left(\textstyle\prod_{j=1}^{\alpha}\prod_{\tau=1}^{c} H_{ijk_\tau^*}^{a_\tau^* a_j}, pk_x\right)\cdot e\left(u^{\sum_{j=1}^{\alpha} a_j\mu_{ij}}, pk_y\right)\\
&\quad\cdot e\left(\textstyle\sum_{\lambda=1}^{m}\omega_\lambda^{\rho_{i\lambda}}, pk_y\right)\\
&= e\left(\textstyle\prod_{j=1}^{\alpha}\prod_{\tau=1}^{c} H_{ijk_\tau^*}^{a_j a_\tau^*}, pk_x\right)\cdot e\left(u^{\mu_i}, pk_y\right)
\end{aligned}
$$

It is obvious that the equation $RHS = LHS$ holds and the TPA will pass the auditing. In Liu et al.'s scheme, the authors claimed that any k-out-of-n servers could be used to recover the original data file when the inequality $k\alpha \geq m$ was maintained. However, through our cryptanalysis, the recovery efficiency will be strongly reduced by our Attack II.

3.3. *Failure of data sharing*

In Liu et al.'s scheme, the data owner maintains a secret key k for $f_k(i)$ in the beginning of Setup phase and augments m original data blocks as Eq.(4) in order to preserve privacy.

$$w_i = \left(\bar{\omega}_{i1}, \bar{\omega}_{i2}, \cdots, \bar{\omega}_{is}, \underbrace{\overbrace{0, \cdots, 0, f_k(i)}^{m}, 0, \cdots, 0}_{i} \right) \in GF(p)^{s+m} \quad (4)$$

When data owners want to recover the original file, they randomly choose m blocks as Eq.(2) in the cloud servers and get uncovered coefficients $\varepsilon_{ij\lambda}$ by computing $\varepsilon_{ij\lambda} = \varepsilon_{ij\lambda}{}' / f_k(\lambda)$ using masked coefficients $\varepsilon_{ij\lambda}{}'$ with $1 \le \lambda \le m$.

Then the native file can be reconstructed from m blocks by solving a set of linearly independent m equations $\{V_t\}_{1 \le t \le m}$ as follows:

$$V_t = v_{ij} = \sum_{\lambda=1}^{m} \varepsilon_{ij\lambda} w_\lambda \quad (5)$$

V_t is randomly chosen from αn blocks stored in n cloud servers.

This algorithm does guarantee the privacy of the outsourced data. However, it also becomes impossible for other data users to reconstruct the original file. Since $f_k(\lambda)$, which is a hushing algorithm with a secret key k, is computationally infeasible in mathematics while the secret key remains unknown. Thus, without the uncovered coefficients, other data users can barely share the native file.

4. Our Countermeasures

4.1. *Countermeasure for two attacks*

At SigAndBlockGen process of Setup phase, in Liu et al.'s scheme, the authors only generated a file tag t and only signed for the native blocks as follows:

$$\mathrm{t} = (\mathrm{ID}||\mathrm{u}||\omega_1||\omega_2||\ldots||\omega_\mathrm{m})||\mathrm{Sig}_{\mathrm{ssk}}(\mathrm{ID}||\mathrm{u}||\omega_1||\omega_2||\ldots||\omega_\mathrm{m}) \quad (6)$$

After the coded blocks generate as Eq.(3), we additionally construct different tags t_{ij} for different blocks in different servers as Eq.(7) to prevent from above two attacks.

$$\mathrm{t_{ij}} = (\mathrm{ID}||\mathrm{u}||\mathrm{v_{ij1}}||\ldots||\mathrm{v_{ij(s+m)}})||\mathrm{Sig}_{\mathrm{ssk}}(\mathrm{ID}||\mathrm{u}||\mathrm{v_{ij1}}||\ldots||\mathrm{v_{ij(s+m)}}). \quad (7)$$

where v_{ijk} denote the kth segment in jth block of ith server.

At the Uploading process of Setup phase, cloud servers perform a checking process before accepting the uploading data from data owners. As soon as cloud servers receive the data, they first use spk to verify the tags t_{ij} and check whether or not uploading blocks are the correct coded ones. If the check is

correct, Uploading process continues. Otherwise, it stops. Cloud servers reject to accept the data and remind the data owner to upload the data again.

4.2. *Countermeasure for failure of data sharing*

To achieve data sharing, instead of encrypting the coefficients with secret key k as Eq.(4), we encrypt the original file with KP-ABE [14]. We can choose a set of attributes γ from a universe of attributes $U = \{1,2,3,\dots,n\}$ ($\{t_i\}_{i\in U}$ are uniformly at random in Z_p) to construct the encryptor. The public parameters PK are $\{\{T_i = g^{t_i}\}_{i\in U}, Y = e(g,g)^y\}$ and master keys MK are $\{\{t_i\}_{i\in U}, y\}$. We encrypt the file F : $E = (\gamma,\ F' = FY^s, \{E_i = T_i{}^s\}_{i\in\gamma})$ where s is randomly chosen in Z_p. After encryption, we divide encrypted file F' into m native blocks which are augmented as Eq.(8).

$$w_i{}' = \left(\bar{\omega}_{i1}{}', \bar{\omega}_{i2}{}', \cdots, \bar{\omega}_{is}{}', \underbrace{0,\cdots,0,\overbrace{1}^{m},0,\cdots,0}_{i}\right) \tag{8}$$

where $1 \le i \le m$.

The authorized data users can randomly download m blocks as Eq.(9), solve a set of linearly independent equations to get the encrypted data $\{\bar{w}_i{}'\}_{i=1}^{m}$ and reconstruct encrypted file F'.

$$v_{ij}{}' = \sum_{\lambda=1}^{m} \varepsilon_{ij\lambda} w_\lambda{}' \tag{9}$$

If and only if the access structures of authorized data users satisfy the data attributes, they can get decryption key $D = \left\{\{D_x = g^{q_x(0)/t_i}\}_{i=att(x)}\right\}$ where q_x is the polynomial of node x. For the root node r of the access tree, $q_r(0) = y$. Then we recursively run the decryption procedure to compute $Y^s = e(g,g)^{ys}$ using decryption key D. Finally, they can decrypt the original file F by computing $F = F'/Y^s$.

Thus, even though the adversary gains the coded blocks and solves the system of equations, they can only get the encrypted data since the data attributes remain unknown, which preserves the privacy. Only the authorized data users whose access structures satisfy the data attributes can decrypt data and get the original file of data owners, which achieves data sharing.

5. Conclusion

In this paper, we first reveal that Liu et al.'s scheme suffers from two attacks which are the data forgery attack and data recovery attack. In addition, Liu et al.'s scheme has a drawbacks which is failure of data sharing. At last, we put forward our countermeasures to solve the security problems. To prevent from two attacks, we construct block tags for different blocks in the cloud. To guarantee data sharing, we encrypt the original data rather than the coefficients.

Acknowledgments

This work is supported by the National Natural Science Foundation of China (NO.61202448), Key Project of Chinese National Programs for Fundamental Research and Development (973 program) (NO.2013CB338003) and the Key Laboratory Program of Information Network Security of Ministry of Public Security (No.C16612). Rui Jiang and Hua-Feng Kong are corresponding authors.

References

1. G. Ateniese et al, "Provable Data Possession at Untrusted Stores," in Proceedings of 14th ACM Conference on Computer and Communication Security (CCS), New York, USA, 2007, pp. 598–609.
2. A. Juels and B. S. Kaliski, "PORs: Proofs of Retrievability for Large Files," in Proc. of 14th ACM Conf. on Com. and Com. Sec., 2007, pp. 584–597.
3. R. Curtmola, O. Khan, R. Burns, and G. Ateniese, "MR-PDP: Multiple-replica Provable Data Possession," in Proceedings of 28th International Conf. on Distributed Computing Systems (ICDCS), Jun. 2008, pp. 411–420.
4. K. D. Bowers, A. Juels, and A. Oprea, "HAIL: A high-availability and integrity layer for cloud storage," in Proceedings of 16th ACM Conference on Computer and Communication Security, 2009, pp. 187–198.
5. B. Chen, R. Curtmola, G. Ateniese, and R. Burns, "Remote Data Checking for Network Coding-based Distributed Storage Systems," in Proceedings of ACM Workshop on Cloud Computing Security Workshop, 2010, pp. 31–42.
6. H. C. H. Chen and P. P. C. Lee, "Enabling Data Integrity Protection in Regenerating-coding-based Cloud Storage: Theory and implementation," IEEE Trans. on Para. and Dist. Sys., vol. 25, no. 2, pp. 407–416, Feb. 2014.
7. Y. Hu, H. C. H. Chen, P. P. C. Lee, and Y. Tang, "NCCloud: Applying Network Coding for the Storage Repair," Proc. USENIX FAST, 2012, p. 21.
8. H. Shacham and B. Waters, "Compact Proofs of Retrievability," in Advances in Cryptology. Berlin, Germany: Springer-Verlag, 2008, pp. 90–107.
9. K. Yang and X. Jia, "An efficient and secure dynamic auditing protocol for data storage in cloud computing," IEEE Transactions on Parallel and Distributed Systems, vol. 24, no. 9, pp. 1717–1726, Sep. 2013.
10. A.G. Dimakis, K. Ramch, "A Survey on Network Codes for Distributed Storage," Proc. of the IEEE, vol. 99, no. 3, pp. 476–489, Mar. 2011.
11. J. Liu, K. Huang, H. Rong, H. Wang, and M. Xian, "Privacy-preserving Public Auditing for Regenerating-code-based Cloud Storage," IEEE Trans. on Information Forensics and Security, vol. 10, no. 7, pp. 13-28, Jul. 2015.

12. R.C. Merkle, “Schemes for Public Key Cryptosystems,” Proceedings of the IEEE Symposium on Security and Privacy, pp. 122-133, 1980.
13. D. Dolev, A.C. Yao, “On the Security of Public Key Protocols” , IEEE Transactions on Information Theory, vol. 29, no. 2, pp. 198-208, Mar. 1983.
14. V. Goyal, O. Pandey, A. Sahai, “Attribute-based Encryption for Fine-grained Access Control of Encrypted Data,” in Proc. of the 13th ACM Conference on Computer and Communications Security, Alexandria, USA, 2006, pp. 89-98.