SECURITY OF DATA AND AUDITING SURVEILLANCES

AkashV.Kulkarni[1], S.Rajaraajeswari[2]

[1] *MCA, MCA, Rajarajeshwari College Of Engineering, Karnataka, India*
[2] *Associate Professor, MCA, Rajarajeshwari College Of Engineering, Karnataka, India*

ABSTRACT

Encryption based complete security has become a issue top to bottom digital guard in various issues in applications. As earlier key concept problem is of distributed storage has been implemented for key upgrades. To address the test, existing arrangements all require the customer to overhaul his mystery enters in each day and age, which may definitely get new nearby weights to the customer, particularly those with restricted solving assets, for example mobile phones. In this paper, we concentrate on the most proficient method to make the security redesigns as straight forward as workable for the customer and In this view, secured updates can be secured and outsourced to some approved and with respect to the key-redesign load on the customer will be secured. Particularly, Defined the outsider valuator (TPA) in many existing open investigating complete outline, let it assume the part of permitted for gathering for our situation, and make it accountable for both the capacity examining and the protected security key for private power-key resistance. In our plan, TPA just needs to hold a scrambled rendition of the customer's master key while doing all these difficult in the interest of the customer. The customer just needs to download the protected security key from the TPA while sending new documents to cloud. The total plan enhances customer with capacity to additionally check the legitimacy of the ***protected keys*** *gave by the TPA. All these notable elements are deliberately intended to make the entire reviewing method with key introduction resistance as maintenance for the customer. We formalize the definition and the security model of this view. The security verification and the execution reproduction demonstrate that our nitty gritty outline instantiations are secure and productive.*

---

1. INTRODUCTION

Cloud Computing now a days is becoming more popular day by day. Mainly in our research we are giving storage and more that is secured transactions which provide protected key that is key based encryption and providing proper security to cloud and performing other operations on third party auditing so that the work is done by third party auditor and all the data is secured .client gives some work to third party auditor through cloud and to provide strong security to data by encryption algorithms. extra capital on deploying and maintaining hardware and software. In recent years, outsourcing computation has attracted much attention and been researched widely. It has been considered in many applications including scientific computations [1], linear algebraic computations [2], linear programming computations [3] and modular exponentiation computations [4], etc. Besides, cloud computing can also provide users with seemingly unlimited storage resource. Cloud storage is universally viewed as one of the most important services of cloud computing. Although cloud store-age provides great benefit to users, it brings new security challenging problems. One important security problem is how to efficiently check the integrity of the data stored in cloud. In recent years, many auditing protocols for cloud storage have been proposed to deal with this problem. These pro-to cols focus on different aspects of cloud storage auditing such as the high efficiency [5]–[17], the privacy protect of data [18], the privacy protection of identities [19], dynamic data operations [13], [15], [16], [20], the data sharing [21], [22], etc. The key security problem, as another important problem in cloud storage auditing, has been considered [23] recently. The problem itself is non-trivial by nature. Once the client's secret key for storage auditing is exposed to cloud, the cloud is able to easily hide the data loss incidents for maintaining its reputation, even discard the client's data rarely accessed for saving the storage. Yu *et al.* [23] constructed a cloud storage auditing protocol with key-exposure resilience by updating the user's secret

keys. In this way, the damage of key exposure in cloud storage auditing can be relaxed. But it also brings in new local burdens for the client because the client has to execute the key update algorithm in each time period to make his secret key move forward. For some clients with limited computation resources, they might not like doing such extra computations by themselves in each time period. It would be obviously more attractive to make key updates as transparent as possible for the client, especially in frequent key update scenarios. In this paper, we consider achieving this goal by out licensed key updates. However, it needs to satisfy several new requirements to achieve this goal. Firstly, the real client's secret keys for

(1) We propose a new paradigm called cloud storage auditing with verifiable outsourcing of key updates key-update operations are not performed by the client, but by an authorized party. The cloud authorized party holds an encrypted secret key of the client for cloud storage auditing and updates it under the encrypted state in each time period. The client downloads the protected encrypted secret key from the authorized party and decrypts it only when he would like to upload new files to cloud. In addition, the client can verify the validity of the encrypted secret key[35].

(1) We design the first cloud storage auditing protocol with verifiable outsourcing of key updates. In our design, the third-party auditor (TPA) plays the role of the authorized party who is in charge of key updates. In addition, similar to traditional public auditing protocols [11]–[17]-[35]

(2) We optimize the definition and the security model of the cloud storage auditing protocol with verifiable outsourcing of key updates. We also prove the security of our protocol in the formalized security model and justify its performance and computation by concrete implementation.

Existing System

- Based on distributed storage examining convention with key-introduction flexibility by redesigning the client's mystery keys intermittently. Along these lines, the harm of key introduction in distributed storage evaluating can be lessened. Be that as it may, it additionally acquires new nearby weights for the customer on the grounds that the customer needs to execute the key upgrade calculation in every day and age to make his mystery key advance.

- For a few customers with restricted calculation assets, they dislike doing such additional calculations without anyone else in every day and age. It would be clearly more appealing to make key overhauls as straightforward as workable for the customer, particularly in successive key redesign situations.

- Wang et al. proposed an open security protecting examining convention. They utilized the arbitrary covering procedure to make the convention accomplish security saving property.

Proposed System

The primary commitments are as per the following:

(1) We propose another worldview called distributed storage reviewing with undeniable outsourcing of key redesigns. In this new worldview, key-upgrade operations are not performed by the customer, but rather by an approved gathering. The approved party holds an encoded mystery key of the customer for distributed storage reviewing and overhauls it under the scrambled state in every day and age. The customer downloads the scrambled mystery key from the approved party and decodes it just when he might want to transfer new documents to cloud. Also, the customer can check the legitimacy of the encoded mystery key.

(2) We plan the main distributed storage examining convention with undeniable outsourcing of key upgrades. In our outline, the third party reviewer (TPA) assumes the part of the approved party who is responsible for key upgrades.

(3) We formalize the definition and the security model of the distributed storage examining convention with certain outsourcing of key upgrades. We likewise demonstrate the security of our convention in the formalized security display and legitimize its execution by solid usage.

1.1 MODEL, DEFINITIONS AND PRELIMINARIES

We show the system model for cloud storage auditing with verifiable outsourcing of key updates in Fig. 1. There are three parties in the model: the client, the cloud and the third-party auditor (TPA). The client is the owner of the files that are uploaded to cloud. The total size of these files is not fixed, that is, the client can upload the growing files to cloud in different time points. The cloud stores the client's files and provides download service for the client. The TPA plays two important roles: the first is to audit the data files stored in cloud for the client; the second is to update the encrypted secret keys of the client in each time period. The TPA can be considered as a party with powerful computational capability or a service in another independent cloud. Similar to [23], the whole lifetime of the files stored in cloud is divided into $T + 1$ time periods (from 0-th to T -th time periods). Each file is assumed to be divided into multiple blocks. In order to simplify the description, we do not furthermore divide each block into multiple sectors [7] in the description of our protocol. In the end of each time period, the TPA updates the encrypted client's secret key for cloud storage auditing according to the next time stamp. But the public key keeps unchanged in the whole time periods. The client sends the key requirement to the TPA only when he wants to upload new files to cloud. And then the TPA sends the encrypted secret key to the client. After that, the client decrypts it to get his real secret key, generates authenticators for files, and uploads these files along with authenticators to cloud. In addition, the TPA will audit whether the files in cloud are stored correctly by a challenge-response protocol between it and the cloud at regular time[35].

1.2 Explanations

(1)The definition of cloud storage auditing protocol with verifiable outsourcing of key updates.

Definition 1: A cloud storage auditing protocol with secure outsourcing of key updates is composed by seven algorithms (SysSetup, EkeyUpdate, VerESK, DecESK, AuthGen, Proof-Gen, Proof Verify), shown below:

1)Sys Set u p: the system setup algorithm is run by the client. It takes as input a security parameter k and The total number of time periods T , and generates an Encrypted initial client's secret key $E S K_0$, a decryption key D K and a public key P K . Finally, the client holds D K , and sends $E S K_0$ to the TPA.

2)E key Update: the encrypted key update algorithm is run by the TPA. It takes as input an encrypted client's secret key $E S K_j$, the current period j and the public keyand generates a new encrypted secret key $E S K_{j+1}$ for period j + 1.

2) Ver ESK : the encrypted key verifying algorithm is run

   by the client. It takes as input an encrypted client's secret key $E S K_j$, the current period j and the public key P K , if $E S K_j$ is a well-formed encrypted client's secret key, returns 1; otherwise, returns 0.

4) DecESK : the secret key decryption algorithm is run by
   the client. It takes as input an encrypted client's secret key $E S K_j$, a decryption key D K , the current period j and the public key P K , returns the real client's secret key $S K_j$ in this time period.

5) AuthGen: the authenticator generation algorithm is run by the client. It takes as input a file F, a client's secret key $S K_j$, the current period j and the public key P K , and generates the set of authenticators for F in time period j .

6) Proof Gen: the proof generation algorithm is run by the cloud. It takes as input a file F, a set of authenticators , a challenge Chal, a time period j and the public key P K , and generates a proof P which proves the cloud stores F correctly.

7) Proof Verify: the proof verifying algorithm is run by the TPA. It takes as input a proof P, a challenge Chal, a time period j , and the public key P K , and returns “T r ue” if P is valid; or “F alse”, otherwise.

(2) Definition of Security

As same as other cloud storage auditing protocols [5]–[7], [9]–[13], [15]–[18], [20], [35]the malicious cloud is viewed as the adversary in our security model. We use three games (Game 1, Game 2 and Game 3) to describe the adversaries with different compromising abilities who are against the security of the proposed protocol. Specifically, Game 1 describes an adversary, who fully compromises the TPA to get all encrypted secret keys $E S K_j$ (periods j = 0, . . . , T), tries to forge a valid authenticator in any time period. This game, in fact, shows the security should satisfy that the TPA cannot help the cloud to forge any authenticator in any time period.

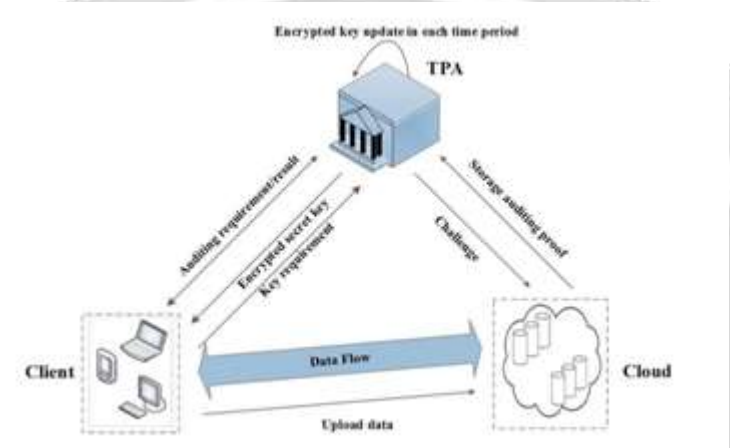

Fig-1 : System Model storage and data protection[35]

.

2. PRELIMINARIES

Let G_1 and G_2 be two multiplicative cyclic group with the same prime order q. A bilinear pairing is a map $e\hat{} : G_1 \times G_1 \rightarrow G_2$ which satisfies:

Notations[35]

Notation	Meaning
G_1, G_2	Two multiplicative groups with some prime order q
g	A generator of G_1
$\hat{e}$	A bilinear pairing $\hat{e}: G_1 \times G_1 \rightarrow G_2$
H_1, H_2, H_3	Three cryptographic hash functions $H_1: G_1 \rightarrow G_1$, $H_2: \{0,1\}^* \times G_1 \rightarrow Z_q^*$ and $H_3: \{0,1\}^* \times G_1 \rightarrow G_1$
T	The total periods number of the whole lifetime for the files stored in the cloud
$w_1 ... w_t$	The binary string of the node w^j associated with period j
$w^j\|_k (k \leq t)$	The k-prefix of w^j
$w^j 0 (w^j 1)$	The left child node and the right child node of w^j
$w^j\|_k$	The sibling node of $w^j\|_k$
ϵ	Empty binary string
PK	The public key which is unchanged in the whole lifetime
ES_{w^j}	The encrypted node secret key
R_{w^j}	The verification value which is used to verify the validity of authenticators
ESK_j	The client's encrypted secret key in period j
X_j	A set composed by the key pairs
Ω_j	A set composed by the verification values
(ES, R)	The key pair of the root node
F	A file which the client wants to store in cloud
$m_i (i = 1, ..., n)$	n blocks of file F
DK	The decryption key to recover the encrypted secret key for cloud storage auditing

1) Bilinearity: $\forall g_1, g_2 \in G_1$ and $a, b \in_R Z_q^*$, there is $e^{\wedge}(g_1^a, g_2^b) = e^{\wedge}(g_1, g_2)^{ab}$.
2) Non-degeneracy: For some $g_1, g_2 \in G_1$, $e^{\wedge}(g_1, g_2) = 1$.
3) Computability: There is an efficient algorithm to com-pute this map.Assume g is a generator of a multiplicativegroup with the prime order q. The CDH problem is that compute $g^{ab}$when given g^a and g^b $(a, b \in_R Z_q^*)$.

2.1 Protocols

A. High-Level Technique Explanation

Our design is based on the structure of the protocol proposed in [23]. So we use the same binary tree structure as [23] to evolve keys, which has been used to design several crypto-graphic schemes [34], [35]. This tree structure can make the protocol achieve fast key updates and short key size. One important difference between the proposed protocol and the protocol in [23] is that the proposed protocol uses the binary tree to update the encrypted secret keys rather than the real secret keys. One problem we need to resolve is that the TPA should perform the outsourcing computations for key updates under the condition that the TPA does not know the real secret key of the client. Traditional encryption technique is not suit-able because it makes the key update difficult to be completed under the encrypted condition. Besides, it will be even more difficult to enable the client with the verification capability to ensure the validity of the encrypted secret keys. To address these challenges, we propose to explore the blinding technique with basic property to efficiently "encrypt" the secret keys. It allows key updates to be vastly performed under the blinded version, and further makes verifying the validity of the encrypted secret keys possible. Our security analysis later on shows that such blinding technique with basic property can sufficiently prevent adversaries from forging any authenticator of valid messages. Therefore, it helps to ensure our design goal that the key updates are as visible as possible for the client. In the designed *Sys Set u p* algorithm, the TPA only holds an initial encrypted secret key and the client holds a decryption key which is used to decrypt the encrypted secret key. In the designed *K eyU pdat e* algorithm, property makes the secret key able to be updated under encrypted state and makes verifying the encrypted secret key possible. The *V er E S K* algorithm can make the client check the validity of the encrypted secret keys immediately[35]. In the end of this section, we will discuss the technique about how to make this check done by the cloud if the client is not in urgent need to know whether the encrypted secret keys are correct or not.

As computation using notations structure paper used for how fast the algorithm runs

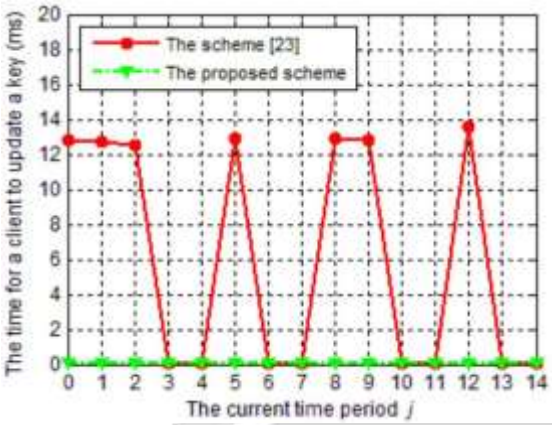

Fig- 2: The key update time on client side in the proposed scheme and the scheme [23].

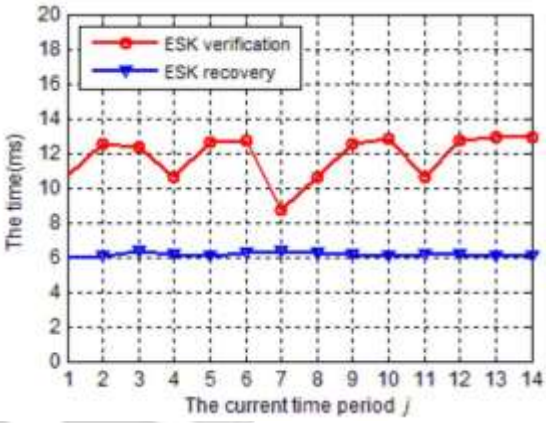

Fig-3:The time to verify and recover an encrypted Secret key (ESK) in different timeperiods[35]

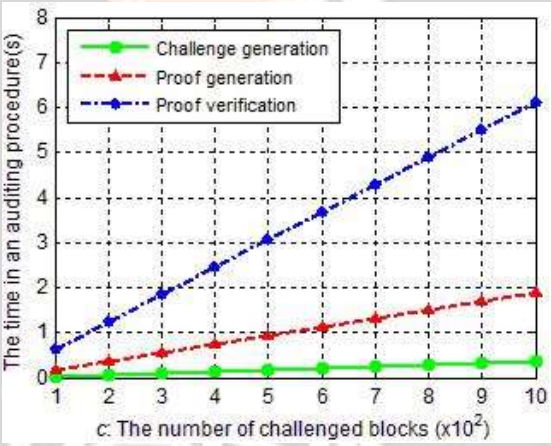

Fig-4: The time of auditing procedures with different number of checked blocks[35]

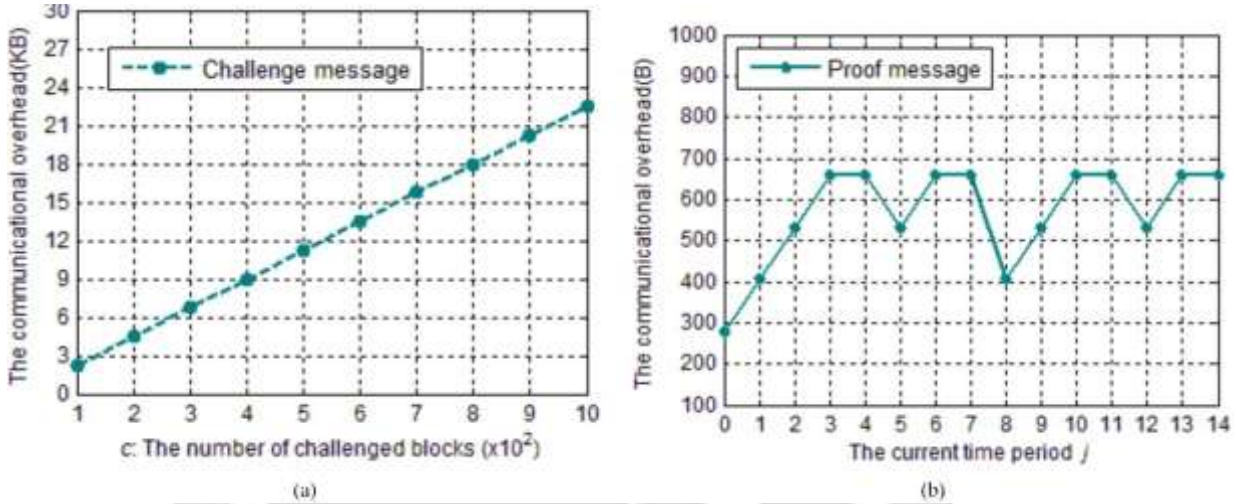

Fig-5:Communicational Cost. (a) The size of the challenge message with different number of checked blocks[35]. (b) The size of the proof message in different time periods[35].

3. Performance Analyses

We evaluate the performance of the proposed scheme through several experiments that are implemented with the help of the Pairing-Based Cryptography (PBC) library [36]. We carry out these experiments on a Linux server with Intel processor running at 2.70 GHz and 4 GB memory. The elliptic curve picked to realize bilinear mapping is a super singular curve with fast pairing operation, and the order of the curve period T 14 which means we can use a full binary tree with depth 2 to associate with these time periods. All results are taken as the mean values from multiple executions

In the proposed scheme, the key update workload is out-sourced to the TPA. In contrast, the client has to update the secret key by itself in each time period in scheme [23]. We compare the key update time on client side between the both schemes in Fig. 3. In scheme [23], the key update time on the client is related to the depth of the node corresponding to the current time period in binary tree. When the depth of node corresponding to the current time period is 0 or 1 (the node is internal node), the update time is about 12.6ms; when it is 2 (the node is leaf node), the update time is almost zero. In our scheme, the key update time on client side is zero in all time periods[35].

When the client wants to upload new files to the cloud, it needs to verify the validity of the encrypted secret key from the TPA and recover the real secret key. We show the time for these two processes happened in different time periods in Fig. 4. In practice, these processes do not happen in most of time periods. They only happen in the time periods when the client needs to upload new files to the cloud. Furthermore, the work for verifying the correctness of the encrypted secret key can fully be done by the cloud[35] .

We demonstrate the time of the challenge generation process, the proof generation process, and the proof verification process with different number of checked data blocks in Fig. 5. In our evaluation, the number of checked block varies from 100 to 1,000. All the time of these processes increase linearly with the number of checked blocks. The challenge generation process spends the least time, which is 0.35s at most; the proof generation process spends more time, varying from 0.19s to 1.89s; the proof verification process spends the most time varying from 0.62s to 6.12s[35].

In our scheme, the communicational messages comprise the challenge message and the proof message. From Fig. 5 (a), we can see that the challenge message is linear with the number of checked blocks. The size of challenge message is 2.25 KB when the checked blocks are 100, and increases to 22.5 KB when the checked blocks are 1,000. As analyzed in [5], when the number of checked blocks is 460, the TPA can detect the data abnormity in the cloud with a probability at least 99%. In this case, the challenge message would be 10.35 KB. From Fig. 5 (b), we can see that the size of proof message varies with the depths of nodes corresponding to time periods. In period 0, the proof

message is the shortest, which is 276.5 bytes, since the depth of the corresponding node is 0. And the longest proof messages appear at the leaves of the tree, which is 0.66 KB[35].

4. CONCLUSIONS

In this paper, we study on how to outsource key updates and performance for cloud storage auditing with key-exposure resilience. We implemented the first cloud storage auditing protocol with verifiable outsourcing of key updates. In this protocol, key updates are outsourced to the TPA and are visible for the client. In addition, the TPA only sees the encrypted version of the client's secret key, while the client can further verify the validity of the encrypted secret keys when downloading them from the TPA. We give the formal security proof and the performance simulation of the proposed scheme .and mainly it is used for distributed storage system and providing strong security for data and then outsourced the key to third party hence the secured work is done and work is also carried by TPA with data security and cloud provides storage and using above algorithms the secured data exchange and work is done

6. REFERENCES

[1] M. J. Atallah, K. N. Pantazopoulos, J. R. Rice, and E. E. Spafford, "Secure outsourcing of scientific computations," *Adv. Comput.*, vol. 54, pp.215–272, 2002.

[2] D. Benjamin and M. J. Atallah, "Private and cheating-free outsourcing of algebraic computations," in *Proc. 6th Annu. Conf. Privacy, Secur. Trust*, 2008, pp. 240–245.

[3] C. Wang, K. Ren, and J. Wang, "Secure and practical outsourcing of linear programming in cloud computing," in *Proc. IEEE INFOCOM*, Apr. 2011, pp. 820–828.

[4] X. Chen, J. Li, J. Ma, Q. Tang, and W. Lou, "New algorithms for secure outsourcing of modular exponentiations," in *Proc. 17th Eur. Symp. Res. Comput. Secur.*, 2012, pp. 541–556.

[5] G. Ateniese *et al.*, "Provable data possession at untrusted stores," in *Proc. 14th ACM Conf. Comput. Commun. Secur.*, 2007, pp. 598–609.

[6] A. Juels and B. S. Kaliski, Jr., "PORs: Proofs of retrievability for large files," in *Proc. 14th ACM Conf. Comput. Commun. Secur.*, 2007, pp.584–597.

[7] H. Shacham and B. Waters, "Compact proofs of retrievability," in *Advances in Cryptology*. Berlin, Germany: Springer-Verlag, 2008

[8] G. Ateniese, R. Di Pietro, L. V. Mancini, and G. Tsudik, "Scalable and efficient provable data possession," in *Proc. 4th Int. Conf. Secur. Privacy Commun. Netw.*, 2008, Art. ID 9.

[9] F. Sebe, J. Domingo-Ferrer, A. Martinez-balleste, Y. Deswarte, and J. Quisquater, "Efficient remote data possession checking in critical information infrastructures," *IEEE Trans. Knowl. Data Eng.*, vol. 20, no. 8, pp. 1034–1038, Aug. 2008.

[10] R. Curtmola, O. Khan, R. Burns, and G. Ateniese, "MR-PDP: Multiple-replica provable data possession," in *Proc. 28th IEEE Int. Conf. Distrib. Comput. Syst.*, Jun. 2008, pp. 411–420.

[11] Y. Zhu, H. Wang, Z. Hu, G.-J. Ahn, H. Hu, and S. S. Yau, "Efficient provable data possession for hybrid clouds," in *Proc. 17th ACM Conf. Comput. Commun. Secur.*, 2010, pp. 756–758.

[12] C. Wang, K. Ren, W. Lou, and J. Li, "Toward publicly auditable secure cloud data storage services," *IEEE Netw.*, vol. 24, no. 4, pp. 19–24, Jul./Aug. 2010.

[13] Q. Wang, C. Wang, K. Ren, W. Lou, and J. Li, "Enabling public auditability and data dynamics for storage security in cloud comput-ing," *IEEE Trans. Parallel Distrib. Syst.*, vol. 22, no. 5, pp. 847–859, May 2011.

[14] K. Yang and X. Jia, "Data storage auditing service in cloud computing: Challenges, methods and opportunities," *World Wide Web*, vol. 15, no. 4, pp.409–428, 2012.

[15] Y. Zhu, G.-J. Ahn, H. Hu, S. S. Yau, H. G. An, and C.-J. Hu, "Dynamic audit services for outsourced storages in clouds," *IEEE Trans. Services Comput.*, vol. 6, no. 2, pp. 227–238, Apr./Jun. 2013.

[16] K. Yang and X. Jia, "An efficient and secure dynamic auditing protocol for data storage in cloud computing," *IEEE Trans. Parallel Distrib. Syst.*, vol. 24, no. 9, pp. 1717–1726, Sep. 2013.

[17] H. Wang, "Proxy provable data possession in public clouds," *IEEE Trans. Services Comput.*, vol. 6, no. 4, pp. 551–559, Oct./Dec. 2013.

[18] C. Wang, S. S. M. Chow, Q. Wang, K. Ren, and W. Lou, "Privacy-preserving public auditing for secure cloud storage," *IEEE Trans. Comput.*, vol. 62, no. 2, pp. 362–375, Feb. 2013.

[19] B. Wang, B. Li, and H. Li Oruta, "Oruta: Privacy-preserving public auditing for shared data in the cloud," *IEEE Trans. Cloud Comput.*, vol. 2, no. 1, pp. 43–56, Jan./Mar. 2014.

[20] C. Erway, A. Küpçü, C. Papamanthou, and R. Tamassia, "Dynamic provable data possession," in *Proc. 16th ACM Conf. Comput. Commun. Secur.*, 2009, pp. 213–222.

[21] B. Wang, B. Li, and H. Li, "Public auditing for shared data with efficient user revocation in the cloud," in *Proc. IEEE INFOCOM*, Apr. 2013, pp.2904–2912.

[22] J. Yuan and S. Yu, "Public integrity auditing for dynamic data shar-ing with multiuser modification," *IEEE Trans. Inf. Forensics Security*, vol. 10, no. 8, pp. 1717–1726, Aug. 2015.

[23] J. Yu, K. Ren, C. Wang, and V. Varadharajan, "Enabling cloud storage auditing with key-exposure resistance," *IEEE Trans. Inf. Forensics Security*, vol. 10, no. 6, pp. 1167–1179, Jun. 2015.

[24] D. Chaum and T. Pedersen, "Wallet databases with observers," in *Advances in Cryptology*. Berlin, Germany: Springer-Verlag, 1993, pp.89–105.

[25] B. Chevallier-Mames, J.-S. Coron, N. McCullagh, D. Naccache, and M. Scott, "Secure delegation of elliptic-curve pairing," in *Proc. CARDIS*, 2010, pp. 24–35.

[26] S. Hohenberger and A. Lysyanskaya, "How to securely outsource cryptographic computations," in *Proc. TCC*, 2005, pp. 264–282.

[27] M. J. Atallah and J. Li, "Secure outsourcing of sequence comparisons," *Int. J. Inf. Secur.*, vol. 4, no. 4, pp. 277–287, 2005.

[28] M. J. Atallah and K. B. Frikken, "Securely outsourcing linear algebra computations," in *Proc. 5th ACM Symp. Inf., Comput. Commun. Secur.*, 2010, pp. 48–59.

[29] X. Chen, J. Li, X. Huang, J. Li, Y. Xiang, and D. S. Wong, "Secure outsourced attribute-based signatures," *IEEE Trans. Parallel Distrib. Syst.*, vol. 25, no. 12, pp. 3285–3294, Dec. 2014.

[30] F.Zhang efficient *Inf. Sci.*, vol. 286

[31] M. Sookhak, A. Gania, M. K. Khanb, and R. Buyyac, "Dynamic remote data auditing for securing big data storage in cloud computing," *Inf. Sci.*, Sep. 2105, doi: 10.1016/j.ins.2015.09.004.

[32] C. Guan, K. Ren, F. Zhang, K. Florian, and J. Yu, "Symmetric-key based proofs of retrievability supporting public verification," in *Proc. 20th Eur. Symp. Res. Comput. Secur. (ESORICS)*, 2015, pp. 203–223.

[33] G. Yang, J. Yu, W. Shen, Q. Su, Z. Fu, and R. Hao, "Enabling

public auditing for shared data in cloud storage supporting iden-tity privacy and traceability," *J. Syst. Softw.*, vol. 113, pp. 130–139, Mar. 2016.

[34] J. Yu, F. Kong, X. Cheng, R. Hao, and G. Li, "One forward-secure signature scheme using bilinear maps and its applications," *Inf. Sci.*, vol. 279, pp. 60–76, Sep. 2014.
Inf. Sci., vol. 286
[35] Enabling Cloud Storage Auditing With Verifiable Outsourcing of Key Updates Jia Yu, Kui Ren, *Fellow, IEEE*, and Cong Wang, *Member, IEEE*