International Journal of Hybrid Information Technology
Vol.6, No.6 (2013), pp.301-310
http://dx.doi.org/10.14257/ijhit.2013.6.6.27

Server Re-Tag Provable Data Possession in Public Cloud

Yongjun Ren[1,2], Jiang Xu[1], Jin Wang[1] and Jeong-Uk Kim[3]

[1] *School of Computer and Software, Nanjing University of Information Science & Technology, Nanjing 210044, China*
[2]*Institute of Information Network Technology,Nanjing University of Posts and Telecommunications, Nanjing 210003, China*
[3]*Department of Energy Grid, Sangmyung University, Seoul 110-743, Korea*

Abstract

Integrity checking becomes imperative to secure data in a cloud environment. Especially in mobile cloud storage, clients can easily modify and share data as a group, which easily result in the stored data is neither compromised nor corrupted. To ensure data integrity can be audited publicly, clients need to compute signatures on all the blocks in shared data .In mobile cloud storage environment, the clients frequently join or leave groups. Once the client leaves the group, the blocks, which were previously signed, must be re-signed by an existing client. One method to allow other client to download the shared data and re-sign it, but it is inefficient due to the large size of shared data in the cloud. In this paper, we propose server re-tag provable data possession (SRT-PDP). By utilizing proxy re-signatures, we allow the cloud severs as the group manager to re-sign blocks after the client sign the shared data, so that existing clients do not have to do anything even if some clients leave the group. In addition, a public verifier is always able to audit the integrity of shared data without retrieving the entire data from the cloud servers. SRT-PDP enable the cloud server takes the most of the work, and reduce the burden of client, which is suit for mobile device.

Keywords: *cloud computing; data storage auditing; provable data possession; proxy re-signature*

1. Introduction

Cloud Computing has been envisioned as the next generation architecture of IT Enterprise, which is defined as a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources that can be rapidly provisioned and released with minimal management effort or service provider interaction. For example, Amazon Elastic Compute Cloud (Amazon EC2) provides cloud computation and Amazon Simple Storage Service (Amazon S3) provides cloud storage.

Cloud storage is an important service of cloud computing, which allows data owners to move data from their local computing systems to the Cloud. In the cloud paradigm, data owners move the large data files from their local computing systems to the remote servers. Moreover it is of critical importance for the data owners can avoid the initial investment of expensive infrastructure setup, large equipment, and daily maintenance cost, which is particularly true for small and medium-sized businesses. Moreover the data owners can rely on the Cloud to provide more reliable services, so that they can access data from anywhere and at any time.

Storing the data in cloud environment becomes natural and also essential. But,

ISSN: 1738-9968 IJHIT

security becomes one of the major concerns for all entities in cloud services. Firstly, data owners would worry their data could be misused or accessed by unauthorized users. Secondly, the data owners would worry their data could be lost in the Cloud. This is because data loss could happen in any infrastructure. Moreover, the cloud service providers (CSP) may be dishonest and they may discard the data which has not been accessed or rarely accessed to save the storage space or keep fewer replicas than promised. As a result, data owners need to be convinced that their data are correctly stored in the Cloud. It is desirable to have data integrity verification to assure data are correctly stored in the Cloud. In order to solve the problem of data integrity verification, many schemes are proposed under different systems and security models [1-13]. Great efforts of all these works are made to design solutions that meet various requirements: high scheme efficiency, stateless verification, unbounded use of queries, *etc*.

With the widespread use of mobile electronic devices, cloud storage platform has the incomparable advantage. In such environment, mobile devices often join or leave the group of the shared data. When some members leave the group, the shared data blocks signed by they are hard to audit. Moreover there is such scenario: Small companies usually stores their data to the cloud server, over a period of time the company goes bankrupt, the company does not deal with these data. However, these data are likely to be very valuable to the public.

To solve the problems, we propose server re-tag provable data possession (SRT-PDP). By utilizing proxy re-signatures, we allow the cloud severs to re-sign blocks after the client sign the shared data, so that existing clients do not have to do anything even if some clients leave and discarded the stored data. In addition, a public verifier is always able to audit the integrity of the data without retrieving the entire data from the cloud servers. SRT-PDP enable the cloud server takes the most of the work, and reduce the burden of client, which is suit for mobile device.

2. Related work

Cloud computing is a promising computing model that enables on-demand network access to a shared pool of configurable computing resources. The first offered cloud service is cloud storage: data owners let cloud service providers host their data on cloud servers and data consumers can access the data from the cloud servers. Because data owners and data servers have different business interests, this new paradigm of data storage service also introduces new security challenges. Therefore, independent auditing service is required to make sure that the data is correctly hosted in the Cloud. Recently, much of growing interest has been pursued in the context of cloud stored data audit.

The message authentication code (MAC) is a kind of hash function which has been used for checking the data integrity for a long time. Some solutions using MAC for storage auditing service have been proposed. Based on the pre-computed MACs stored on the verifier, the protocols proposed by Lillibridge *et al.*, [2] and Naor *et al.*, [3]can detect any data loss or corruption with high probability. Shacham et al. [4] proposed a MAC-based batch verification for multiple data blocks.

In 2007 Ateniese, *et al.*, [5] proposed a PDP model to solve the storage problems of files. They divided the file into blocks, and computed a homomorphic tag [6] for each block, completed the proof of the data integrity by sampling and verifying the correspondence of the tags and blocks randomly. A. Juels, *et al.*, [7] proposed a provable data recovery (POR) model. Instead of tagging file blocks, they inserted some sentinel blocks, and verified the integrity of the file by checking the correctness of

sentinel blocks. For the sentinel blocks are one-time labels, the number of times that the file can do integrity verification is limited, related to the number of sentinel blocks. Havav Shacham and Brent Waters [4] proposed an improved POR model under the security model defined in [7], and had a very complete proof. They used tags similar to [5], and applied to public authentication. Kevin D. Bowers *et al.*, [8] and Yevgeniy Dodis *et al.*, [9] made some theory and application extensions based on [4, 7]. Zheng and Xu also present a dynamic POR model in [10].

PDP model proposed in [5] only applied to private authentication. It meant that only the person who has the private key can verify the integrity of the file. Ateniese improved PDP model to apply to public authentication in [11]. They replaced the homomorphic tags in [5] with homomorphic tags supported public authentication [12]. C. Erway [13] proposed dynamic PDP model based on PDP model. It maintained a skip-list for tags, and stored the root metadata in Client's hand to prevent replay attack. Qian Wang, *et al.*, [14] use the tags based on [4] to apply the data integrity verification of dynamic files. Its computation and communication were both smaller than DPDP model. Researchers also proposed the concept of the Proofs of Ownership for cloud computing environment [15]. Zhu *et al.*, present a cooperative PDP (CPDP) scheme based on homomorphic verifiable response and hash index hierarchy [16]. Recently Wang e.al proposes a public auditing scheme with client revocation [17], but the scheme assumes that there is secure secret channel prior to execution of the scheme between the every communicated pairs, which is too high and difficult to achieve in the real environment.

3. Preliminaries

SRT-PDP system model and security model are given in this section. At the same time, bilinear pairings and some corresponding difficult problems are also depicted below.

3.1. System Model

SRT-PDP system consists of three different network entities: Client, CCS, and Third Party Auditor (TPA). They can be identified below.

1) Client: an entity, which has massive data which will be moved to CPS for maintenance and computation, can be either individual consumer or organization;

2) Cloud Storage Server (CSS): an entity, which is managed by cloud service provider, has significant storage space and computation resource to maintain the clients' data;

3) Third Party Auditor: an entity, which has expertise and capabilities that clients do not have, is trusted to assess and expose risk of cloud storage services on behalf of the clients upon request.

By hosting their data in the Cloud, data owners can avoid the initial investment of expensive infrastructure setup, large equipment, and daily maintenance cost. Since the clients no longer possess their data locally, it is necessary for the clients to ensure that their data are being correctly stored and maintained. That is, clients should be equipped with certain storage auditing services so that they can periodically check the integrity of the remote data even without the existence of local copies. In such environment, mobile devices often join or leave the group of the shared data. There is a scenario: Small

companies usually stores their data to the cloud server, over a period of time the company goes bankrupt, the company does not deal with these data. However, these data are likely to be very valuable to the public. To solve the problems, we propose server re-tag provable data possession (SRT-PDP).

3.2. Design Goals

To correctly verify the integrity of shared data with efficient user revocation, our public auditing mechanism should achieve the following properties: (1) Correctness: The TPA is able to correctly check the integrity of shared data. (2) Efficient Re-Tag: On one hand, once a client is discarded his data in the cloud server, the data blocks signed by the user can be efficiently re-signed. On the other hand, other existing clients can generate valid signatures on their data and can verify the re-signed data blocks. (3) Public Auditing: The TPA can audit the integrity of the signed data without retrieving the entire data from the cloud, even if some blocks in the cloud have been re-signed by the cloud server.

3.3. Complexity Assumptions

3.3.1. Bilinear Map: Let G_1 and G_2 be two multiplicative cyclic groups of prime order$p$, g is a generator of G_1, Bilinear is a map $e: G_1 \times G_1 \rightarrow G_2$with the following properties: 1)Computability: there exists an efficient algorithm for computingmape. 2) Bilinearity: for all $u, v \in G_1$ and $a, b \in Z_p$, $e(u^a, v^b) = e(u, v)^{ab}$. 3) Non-degeneracy:$e(g, g) \neq 1$.

3.3.2. Complexity Assumption: ***Define1***.Computational Diffie-Hellman (CDH)Problem.For$a, b \in Z_p$, given$g$,g^a,$g^b \in G_1$ as input ,output $g^{ab} \in G_1$.

The CDH assumption holds in G_1if it is computationally infeasible to solve the CDH problem inG_1.

Define 2.Discrete Logarithm (DL) Problem. For$a \in Z_p$, given$g$,$g^a \in G_1$ as input, out a.

The DL assumption holds in G_1if it is computationally infeasible to solve the DL problem in G_1.

3.4. Proxy Re-signatures

Proxy re-signatures, first proposed by Blaze et al. [18], allow a semi-trusted proxy to act as a translator of signatures between two users, for example, Alice and Bob. More specifically, the proxy is able to convert a signature of Alice into a signature of Bob on the same block. Meanwhile, the proxy is not able to learn any private keys of the two users, which means it cannot sign any block on behalf of either Alice or Bob. In this paper, to improve the efficiency of user revocation, we propose to let the cloud to act as the proxy and convert signatures for users.

4. Our Server Re-tag Provable Data Possession Scheme

4.1. Our Construction

Let G_1 and G_2 be two groups of order$q$,g is a generator of G_1, $e: G_1 \times G_1 \to G_2$ is a bilinear map, u is a different random element from g in G_1 ,two hash functions $H_1: \{0,1\}^* \to G_1$, $H_2: \{0,1\}^* \to Z_q$. The public parameters are $(e, q, g, G_1, G_2, u, H_1, H_2)$. We assume that shared data file *F* (potentially encoded using Reed-Solomon codes [11]) is divided into *n* blocks$\{m_1, m_2, \cdots, m_n\}$. The procedure of our basic scheme execution is as follows:

$KeyGen(1^k) \to (sk, pk)$

The client *A* chooses a random $a \in Z_q$and compute $pk_A = g^a$.The secret key is a and the public key is g^a. The cloud server randomly selects $s \in Z_q$ as private keyssk, and compute $spk = g^s$ as public key.

$ReKey \to \mathrm{rk}$

The cloud server generates a re-signing key as follows: The cloud server generates a random $r \in Z_q$ and sends it to the client *A*. The client *A* sends r/a to cloud server. The cloud server computes $rk_{A \to S} = s/a \in Z_q$, where s is the private key of the cloud server.

$TagGen(sk_A, F) \to T_A$

Given *F*=$\{\mathrm{m}_1, \mathrm{m}_2, \cdots, \mathrm{m}_n\}$, the client *A* with the private key a generates the tag T_A of the block m_i:$\sigma_i = (H_1(f_i)u^{m_i})^a$, where block identifier of m_i is f_i, then denotes the set by$\Phi = \{\sigma_i\}$,$1 \le i \le n$ as the tag for block m_i.The client sendsT_A =$\{F, \Phi\}$ to the cloud server and deletes them from its local storage.

$ConvertTag(T_A, rk_{A \to s}) \to T_s$

When the cloud server receives T_A , it first checks the following formula:$e(\sigma_i, g) = e(H_1(f_i)u^{m_i}, pk_A)$. If the verification fails, reject by the cloud server; otherwise it convert tag of the client *A* into its tag on the same block as follows. The cloud server computes $\sigma_i' = \sigma_i^{rk_{A \to S}} = (H_1(f_i)u^{m_i})^{a \cdot s/a} = (H_1(f_i)u^{m_i})^s$.Let $\Phi' = \{\sigma_i'\}$.Then Φ' and *F* are stored.

$GenChal(k) \to chal$

The TPA can verify the integrity of the shared data. It picks a random *c*-element subset *I* of the set [1,*n*]. For $i \in I$ ($1 \le i \le c$), the TPA chooses a random element $v_i \in Z_q$ and sends the message *chal*=$\{(i, v_i)\}_{i \in I}$ to the cloud server.

$Genproof(F, \phi', chal) \to TPA$

Upon receiving the challenge, the cloud server computes:

$$\sigma = \prod_{i=1}^{c} \sigma_i'^{v_i} \quad , \quad \mu = \sum_{i=1}^{c} v_i m_i.$$

The cloud server outputs *pf*=$\{\sigma, \mu, f_i\}$and sends it to the verifier.

$Verify\ \mathrm{Proof}(pk, chal, pf) \to \{true, false\}$

Upon receiving the response *pf* from the CSS, the TPA checks the correctness of the auditing proof as

$$e(\sigma, g) = e(\prod_{i=1}^{c} H_1(f_i)^{v_i} \cdot u^{\mu}, spk) \quad (1)$$

If so, output "true"; otherwise "false".

4.2. Security Analysis

The correctness analysis and security analysis of our SRT-PDP scheme can be given by the following theorems.

Theorem 1. If the proposed procedures follow as above, any TPA is able to correctly check the integrity of shared data.

Proof: According to our scheme procedures, we know that

$$e(\sigma, g) = e(\prod_{i=1}^{c} \sigma_i'^{v_i}, g)$$

$$= e(\prod_{i=1}^{c} (H_1(f_i) u^{m_i})^{sv_i}, g)$$

$$= e(\prod_{i=1}^{c} (H_1(f_i) u^{m_i})^{v_i}, g^s)$$

$$= e(\prod_{i=1}^{c} H_1(f_i)^{v_i} \cdot \prod_{i=1}^{c} u^{m_i v_i}, spk)$$

$$= e(\prod_{i=1}^{c} H_1(f_i)^{v_i} \cdot u^{\mu}, spk)$$

Theorem 2. The proposed PDP scheme is computational infeasible to generate a forgery of an auditing proof under our mechanism.

Proof: Following the security model and security game defined in [14], we can prove that, if the cloud server could win a security game, named Game 1, by forging an auditing proof on incorrect shared data, and then we can find a solution to the Discrete Logarithm problem in G_1 with a probability of $1-1/q$. We define Game 1 as follows:

Game 1: The TPA chooses a challenge $chal=\{(i, v_i)\}_{i \in I}$ and sends it to the cloud server, cloud server outputs $pf=\{\sigma, \mu, f_i\}$ as the proof on correct shared data F, which should pass the verification with Equation (1). However, the cloud generates a proof on false shared data F' as $\{\sigma, \mu', f_i\}$, where $\mu = \sum_{i=1}^{c} v_i m_i'$, for $i \in I$ $(1 \le i \le c)$ and $F \neq F'$.Define$\Delta\mu = \mu' - \mu$. If this proof still passes the verification performed by the TPA, then the cloud server wins this game. Otherwise, it fails.

We first assume that the cloud wins the game. Then, according to Equation (1), we have

$$e(\sigma, g) = e(\prod_{i=1}^{c} H_1(f_i)^{v_i} \cdot u^{\mu'}, spk)$$

Because $\{\sigma,\mu, f_i\}$is a correct auditing proof, we also have:

$$e(\sigma, g) = e(\prod_{i=1}^{c} H_1(f_i)^{v_i} \cdot u^{\mu}, spk)$$

Based on the properties of bilinear maps, we can learn that

$$\prod_{i=1}^{c} u^{\mu x} = \prod_{i=1}^{c} u^{\mu' x}, \prod_{i=1}^{c} u^{x\Delta\mu} = 1.$$

Because G_1 is a cyclic group, then for two elements $\alpha, \beta \in G_1$, there exists $x \in Z_q$ that $\beta = \alpha^x$. Without loss of generality, given α, β, u^x can be generated as $u^x = \alpha^{\xi}\beta^{\varsigma} \in G_1$, where ξ and ς are random values of Z_q. Then, we have

$$1 = \prod_{i=1}^{c}\left(\alpha^{\xi}\beta^{\varsigma}\right)^{\Delta\mu} = \alpha^{\sum_{i=1}^{c}\xi\Delta\mu} \cdot \beta^{\sum_{i=1}^{c}\varsigma\Delta\mu}.$$

Clearly, we can find a solution to the Discrete Logarithm problem. Given$\alpha, \beta = \alpha^x \in G_1$, we can output

$$\beta = \alpha^{-\frac{\sum_{i=1}^{c}\xi\Delta\mu}{\sum_{i=1}^{c}\varsigma\Delta\mu}}, x = -\frac{\sum_{i=1}^{c}\xi\Delta\mu}{\sum_{i=1}^{c}\varsigma\Delta\mu},$$

unless the denominator is zero. However, as we defined in Game 1,at least one of element in $\Delta\mu$ is nonzero, and ς is a random element ofZ_q, therefore, the denominator is zero with a probability of$1/q$, which is negligible because$q$ is a large prime. Then, we can find a solution to the Discrete Logarithm problem with a probability of$1 - 1/q$, which contradicts to the assumption that Discrete Logarithm problem is computationally infeasible in G_1.

4.3. Batch Auditing for Data in Multi-client and cloud server

As cloud servers may concurrently handle multiple verification sessions from different clients, given*K* signatures on *K* distinct data files from *K* clients, it is more advantageous to aggregate all these signatures into a single short one and verify it at one time. To achieve this goal, we extend our scheme to allow for provable data updates and verification in a multi-client and multi- server system. The key idea is to use the bilinear aggregate signature scheme. As in the BLS-based construction, the aggregate signature scheme allows the creation of signatures on arbitrary distinct messages. Moreover, it supports the aggregation of multiple signatures by distinct signers on distinct messages into a single short signature, and thus greatly reduces the communication cost while providing efficient verification for the authenticity of all messages.

Assume there are *K* clients or cloud servers in the system, and each client or cloud server *k* has data files$F_i = \{m_{k,1}, m_{k,2}, \cdots, m_{k,n}\}$, where $k \in \{1, \cdots K\}$.In *TagGen* and *ConvertTag* phase the tag for blocks $\sigma_{k,i}$ is generated. In *GenGhal* phase the verifier sends the query $\{(i, v_i)\}_{i\in I}$to the server for verification of all *K* clients. In the *GenProof* phase, upon receiving the *chal*, the server computes

$$\sigma = \prod_{k=1}^{K}(\prod_{i=1}^{c}\sigma_{k,i}^{v_i}) \quad , \quad \mu_k = \sum_{i=1}^{c} v_i m_{k,i}$$

The server then responses the verifier with $\{\sigma,\mu_k,f_{k,i}\}$.In the *VerifyProof* phase, similar as the single-client case, using the properties of the bilinear map, the verifier can check if the following equation holds:

$$e(\sigma, g) = \prod_{k=1}^{K} e(\prod_{i=1}^{c} H_1(f_{ki})^{v_i} \cdot u^{\mu_k}, spk)$$

4.4. Performance Evaluation

In this section, we will discuss the communication and computation cost of our mechanism.

4.4.1. Communication Cost: According to the description, the size of an auditing message $\{(i, v_i)\}_{i \in I}$ is $c \cdot (|n| + |q|)$ bits, where *c* is the number of selected blocks, $|n|$is the size of an element [1,*n*] and $|q|$is the size of an element of Z_q. The size of an auditing proof $\{\sigma,\mu, f_i\}$ is $2|p| + c \cdot |f|$ bits, where $|p|$is the size of an element of $Z_q$or G_1,$|f|$is the size of a block identifier. Therefore, the total communication cost of an auditing task is$2|p| + c \cdot (|f| + |n| + |q|)$ bits.

4.4.2. Computation Cost: As shown in *ConvertTag* of our mechanism, the cloud first verifies the correctness of the original signature on a block, and then computes a new signature on the same block with a re-signing key. The computation cost of re-signing a block in the cloud is $2Exp_{G_1} + Mul_{G_1} + 2Pair + Hash_{G_1}$, where Exp_{G_1} denotes one exponentiation in G_1,Mul_{G_1} denotes one multiplication in G_1, Pair denotes one pairing operation, and $Hash_{G_1}$denotes one hashing operation in G_1. The cloud can further reduce the computation cost of the re-signing on a block to Exp_{G_1} by directly re-signing it without verification. The public auditing performed by the TPA ensures that the re-signed blocks are correct. Moreover *ConvertTag* algorithm is performed by cloud server. In generally we assume that the computable ability of the cloud server is unlimited. Thus our scheme reduces the burden of the clients.

5. Conclusions

In this paper, we propose server re-tag provable data possession (SRT-PDP). By utilizing proxy re-signatures, we allow the cloud severs as the group manager to re-sign blocks after the client sign the shared data, so that existing clients do not have to do anything even if some clients leave the group. In addition, a public verifier is always able to audit the integrity of shared data without retrieving the entire data from the cloud servers. SRT-PDP enable the cloud server takes the most of the work, and reduce the burden of client, which is suit for mobile device.

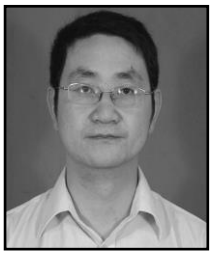
Acknowledgements

This work was supported by the National Natural Science Foundation of China (61232016), the Jiangsu Province Universities Natural Science Research Program (NO.11KJB510010) and Jiangsu Province Research and Innovation Project for College Graduates (CXZZ12_0515). It was also supported by the Industrial Strategic Technology Development Program (10041740) funded by the Ministry of Trade, Industry and Energy (MOTIE Korea), and by the Natural Science Foundation of Jiangsu Province (No. BK2012461). Professor Jeong-Uk Kim is the corresponding author.

References

[1] K. Yang and X. Jia, "Data storage auditing service in cloud computing: challenges, methods and opportunities", The journal of World Wide Web, vol. 15, **(2012)**, pp. 409.
[2] M. Lillibridge, S. Elnikety, A. Birrell, M. Burrows and M. Isard, "A cooperative internet backup scheme", Proceedings of the annual conference on USENIX Annual Technical Conference, **(2003)** June, pp. 27-36; Berkeley, CA, USA.
[3] M. Naor and G. N. Rothblum, "The complexity of online memory checking", Proceedings of the 46th Annual IEEE Symposium on Foundations of Computer Science, (**2005**) October, pp. 573–584; Pittsburgh, USA.
[4] Shacham and B. Waters, "Compact proofs of retrievability", In ASIACRYPT '2008, **(2008)** December, pp. 90-107; Melbourne, Australia.
[5] G. Ateniese, R. Burns, R. Curtmola, J. Herring, L. Kissner, Z. Peterson and D. Song, "Provable data possession at untrusted stores", In CCS '07, **(2007)** October, pp. 598-609; Alexandria, VA, USA.
[6] R. Johnson, D. Molnar, D. song and D. wagner, "Homomorphic signature schemes", In Proc. of CT-RSA, **(2002)** February, pp. 244-262; San Jose, CA, USA.
[7] A. Juels and B. Kaliski, "PORs: Proofs of retrievability for large files", In CCS '07, **(2007)** October, pp. 584-597; Alexandria, VA, USA.
[8] K. Bowers, A. Juels and A. Oprea, "Proofs of retrievability: Theory and implementation", Technical Report 2008/175, Cryptology ePrint Archive, **(2008)**.
[9] Y. Dodis, S. Vadhan and D. Wichs, "Proofs of retrievability via hardness application", In TCC'09, **(2009)** March, pp. 109-127; San Francisco, CA, USA.
[10] Q. Zheng and S. Xu, "Fair and Dynamic Proofs of Retrievability", CODASPY'11, **(2011)** February 21–23; San Antonio, Texas, USA.
[11] G. Ateniese, S. Kamara and J. Katz, "Proofs of storage from homomorphic identification protocols", ASIACRYPT'09, **(2009)** December, pp. 319-333; Tokyo, Japan.
[12] D. Boneh, B. Lynn and H. Shacham, "Short signatures from the weil pairing", ASIACRYPT 2001, **(2001)** December, pp. 514-532; Gold Coast, Australia.
[13] C. Erway, A. Kupcu, C. Papamanthou and R. Tamassia, "Dynamic provable data possession", In CCS '09, **(2009)** November, pp. 213-222; Chicago, IL, USA.
[14] Q. Wang, C. Wang, J. Li, K. Ren and W. Lou, "Enabling public verifiability and data dynamics for storage security in cloud computing", IEEE Trans. on Parallel and Distributed Systems, vol. 22, **(2011)**, pp. 847.
[15] S. Halevi, D. Harnik, B. Pinkas and A. Shulman-Peleg, "Proofs of Ownership in Remote Storage Systems", The Proceedings of the 18th ACM conference on Computer and communications security, **(2011)** November, pp. 491-500; Chicago, IL, USA.
[16] Y. Zhu, H. Hu, G. -J. Ahn and M. Yu, "Cooperative Provable Data Possession for Integrity Verification in Multicloud Storage", IEEE Transactions on Parallel and Distributed Systems, vol. 23, no. 12, **(2012)**.
[17] B. Wang, B. Li and H. Li, "Public Auditing for Shared Data with Efficient User Revocation in the Cloud", accepted by INFOCOM 2013, **(2013)** July15-19; Turin, Italia.
[18] G. Ateniese and S. Hohenberger, "Proxy Re-signatures: New Definitions, Algorithms and Applications", in the Proceedings of ACM CCS 2005, **(2005)** November, pp. 310–319; Alexandria, VA, USA.

Authors



Yongjun Ren

He obtained his Masters in Computer received the M.S. degree in computer science from HoHai University, China, in 2004 and PhD degree in the computer and science Department at Nanjing University of Aeronautics and Astronautics, China, in 2008.Now he is serving as a full time faculty in the computer and software Department at Nanjing University of Information science and Technology. His research interests include network security and privacy and applied cryptography with current focus on security and privacy in cloud computing, lower layer attack and defense mechanisms for wireless networks, and sensor network security.

Jiang Xu

He graduated as the top student in the Nanjing University of Aeronautics and Astronautics where would obtain his PhD in 2013. At the same time, he is serving as a full time faculty in the School of Computer & Software, Nanjing University of Information Science & Technology (NUIST). His research interest includes wireless communication network, wireless sensor networks and Internet of things.

Jin Wang

He received the B.S. and M.S. degree in the Electronical Engineering from Nanjing University of Posts and Telecommunications, China in 2002 and 2005, respectively. He received Ph.D. degree from the Computer Engineering Department of Kyung Hee University Korea in 2010. Now, he is a professor in the Computer and Software Institute, Nanjing University of Information Science and Technology. He has published more than 120 journal and conference papers. His research interests mainly include routing protocol and algorithm design, performance evaluation and optimization for wireless ad hoc and sensor networks. He is a member of the IEEE and ACM.