International Journal of Security and Its Applications
Vol.7, No.6 (2013), pp.11-20
http://dx.doi.org/10.14257/ijsia.2013.7.6.02

Designated-Verifier Provable Data Possession in Public Cloud Storage

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

Cloud storage is now an important development trend in information technology. However, information security has become an important problem to impede it for commercial application, such as data confidentiality, integrity, and availability. In this paper, we propose designated verifier provable data possession (DV-PDP). In public clouds, DV-PDP is a matter of crucial importance when the client cannot perform the remote data possession checking. We study the DV-PDP system security model and use ECC-based homomorphism authenticator to design DV-PDP scheme. The scheme removed expensive bilinear computing. Moreover in DV-PDP scheme, the cloud storage server is stateless and independent from verifier, which is an important secure property in PDP schemes. Through security analysis and performance analysis, our scheme is provable secure and high efficiency.

Keywords: *cloud computing; data storage auditing; provable data possession*

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.

Storing the data in cloud environment becomes natural and also essential. But, 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 storage auditing (DSA) service to assure data are correctly stored in the Cloud.

In order to solve the problem of data auditing service, 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,

ISSN: 1738-9976 IJSIA

stateless verification, unbounded use of queries, etc. Considering the role of the verifier in the model, all the presented before schemes fall into two categories: private verification and public verification. However, public verification is undesirable in many circumstances. For example, the data owners will be restricted to access the Internet, e.g., on the ocean-going vessel, *et al.*,. In the situations, the data owner cannot perform the remote data integrity checking.

In this paper, we propose the concept of Designated-Verifier Provable Data Possession (DV-PDP). Then, we give DV-PDP system model and formal DV-PDP security model. In DV-PDP, data owners can designate a verifier to verify data integrity of his data. The verifier is stateless and independent from CSP, which solves the problem that the verifier can be controlled by the malicious CSP. In our design, we propose to use ECC-based homomorphism authenticator to design PDP scheme, which does not compute expensive bilinear and consume small amount of calculation and Communications. Our scheme is very suitable for mobile clouds.

2. Related Work

In recent years much of growing interest has been pursued in the context of remotely stored data verification. Ateniese, *et al.*, [1] defined the Provable Data Possession (PDP) model to solve the storage problems of static files. They divided the file into blocks, and computed a homomorphic tag [2] 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.*, [3] 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 [3], and had a very complete proof. They used tags similar to [1], and applied to public authentication. Kevin D. Bowers *et al.*, [5] and Yevgeniy Dodis *et al.*, [6] made some theory and application extensions based on [3, 4]. Zheng and Xu also present a dynamic POR model in [7].The PDP model proposed in [1] can be applied to private authentication. In literature [8] Ateniese improved the PDP model to apply to public authentication. They replaced the homomorphic tags in [1] with homomorphic tags supported public authentication [9].

C. Erway[10] were the first to explore constructions for dynamic provable data possession. This scheme is essentially a fully dynamic version of the PDP solution. It maintained a skip-list for tags, and stored the root metadata in Client's hand to prevent replay attack. Qian Wang, *et al.*, [11] use the tags based on [4] to apply the data integrity verification of dynamic files. Its computation and communication were both smaller than the dynamic provable data possession scheme in [10]. Researchers also proposed the concept of the Proofs of Ownership for cloud computing [12]. In 2012, Zhu *et al.*, presented a cooperative PDP (CPDP) scheme based on homomorphic verifiable response and hash index hierarchy [13]. Many researchers proposed other data storage auditing security models and concrete schemes [14-18].

Considering the role of the PDP verifier, the PDP protocols can be classified into two categories: private PDP and public PDP. In the response checking phase of private PDP, some private information is needed. While in the response checking of public PDP, the private information is not needed. Private PDP is necessary in some cases. Recently Shen *et al.*, presented delegable provable data possession scheme [18], in which data owner generates the delegation key for delegated verifier and store the key in CPSs for

verification. A malicious CPS can control the delegation key and lead to the failure of the subsequent validation work. Wang *et al.*,.also proposed a proxy provable data possession (PPDP) model and provided a construction for it [19]. In PPDP data owner can delegate its remote data possession checking capability to the proxy by sending it a warrant. The warrant will be stored both in the proxy and CPS. Before the verification of the data, the both warrant are checked for consistency. If the CPS is malicious, it can reject all queries from the proxy and interrupt the implementation of the scheme. The problem is that the proxy or delegated verifier is not stateless to CSP. While the verifiers should be stateless, and not need to maintain and update state between CPSs, since such state is difficult to maintain if the verifier's machine crashes or if the verifier's role is delegated to third parties or distributed among multiple machine [4, 11].

3. Preliminaries

DV-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

DV-PDP system consists of three different network entities: Client, PCS, Designated-Verifier (DV). 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) Designated Verifier (DV): an entity, which is trusted and designated to assess and expose risk of cloud storage services on behalf of the Clients.

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. Sometimes, clients do not necessarily have the time, feasibility or resources to monitor the outsourced data, they can designate a certain verifier to perform the monitoring task. The designated verifier is trusted by the clients and is independent from the cloud storage services, which means that the designated verifier is stateless for CSS. In the following, we propose an efficient pairing-based private DV-PDP scheme.

3.2. DV-PDPdefines

The DV-PDP scheme is composed of the following algorithm.

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

It is a key generation algorithm to setup the scheme. It takes input a security parameter k and returns the corresponding private/public key pair. By running it twice, this algorithm can return the Client's private/public key pair(x,X) and the designated verifier's private/public key pair (y,Y).

$TagGen(x, Y, m) \rightarrow T_m$

This algorithm is run by the client. It takes as input its private key *x*, the designated verifier's public key *Y* and a file block *m*, and outputs the checking tag T_m.

$GenChal(k) \rightarrow chal$

It is a probabilistic polynomial time algorithm run by the client or the designated verifier to generate a challenge to CSS for the stored data. It takes as input the public parameter *k*, and outputs the challenge *chal*.

$Genproof(F, \phi, chal) \rightarrow DV$

It is run by the CCS in order to generate the possession proof. It takes as inputs the public parameter, an ordered collection *F* of blocks and tags ϕ, and the challenge *chal*. It returns a data possession proof *pf* for the blocks in *F* that are determined by the challenge *chal*.

$VerfyProof(X, y, pf, chal) \rightarrow \{true, false\}$

This algorithm can be run by either the designated verifier the client or upon receipt of the proof *pf*. It takes as input the public key *X*(*Y*), the challenge *chal*, its own private key *y*(*x*) and the proof *pf* returned from the server, and outputs "true" if the integrity of the file is verified as correct, or "false" otherwise.

3.3. Security Model

Definition 1 (**Unforgeability**): A DV-PDP protocol is secure if for any (probabilistic polynomial)adversary *A* (i.e., malicious PCS) the probability that *A* wins the DV-PDP game is negligible. The DV-PDP game between the challenger *C* and the adversary *A* can be depicted as follows:

(1)SetUp: Suppose the system parameter is *params*. *KeyGen* is a private/public key pair generating algorithm. By running *KeyGen*, *C* can get the client's private/public key pair (*x*,*X*), the designated verifier's private/public key pair (*y*,*Y*).*C* keeps *x*, *y* confidential and sends (*X*,*Y*) to *A*.

(2)Queries: *A* adaptively makes a number of different queries to *C*. Each query can be one of the following.

- Hash query. *A* makes Hash function queries adaptively. *C* responds the Hash values to *A*;
- Proof query. *A* chooses challenge chal and obtains a valid proof with the *chal*.

(3)Challenge: C generates a challenge *chal* which defines a ordered collection. *C* is required to provide a possession proof for the blocks.

(4)Answer: *A* computes a data possession proof *pf* for the blocks indicated by *chal* and returns *pf*.

In the DV-PDP game, we say that the success probability of the adversary A is negligible. i.e.,

$$Adv_A(\Pr[Verifyproof = true]) \leq \varepsilon$$

where ε is negligible.

Hardness problem

Let G be a cyclic multiplicative group on ECC generated by *P*, the related complexity assumptions are as follows.

Computational Diffie-Hellman (CDH) Problem.

Given a randomly chosen $P \in G$, as well as *aP*, *bP*, for unknown $a, b \in Z_p$, compute *abP*.

Elliptic Curve Discrete Logarithm Problem.

Given points *P* and *Q* of the group in elliptic curve, find a number *k* such that $P^k = Q$.

4. Our Designated Verifier Provable Data Possession Scheme

4.1. Our Construction

Now we start to present the main idea behind our scheme. We assume that file *F* (potentially encoded using Reed-Solomon codes [11]) is divided into *n* blocks $\{m_1, m_2, \cdots, m_n\}$, where $m_i \in Z_q$ and q is a large prime. Let G be a cyclic multiplicative group on ECC generated by *g*, two hash functions $H_1, H_2: \{0,1\}^* \to Z_q$, viewed as a random oracle. The procedure of our basic scheme execution is as follows.

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

The client choose a random $x \in Z_q$ and compute $X = g^x$.The secret key is *x* and the public key is *X*. The client designates a trust verifier DV. DV run the *KeyGen* and randomly choose $y \in Z_q$ as his private key and computes $Y = g^y$ as his public key.

$TagGen(x, Y, m) \to T_m$

Given $F=\{m_1, m_2, \cdots, m_n\}$, the client generates the tag T_m of the block m_i. Let k_{i1} and k_{i2} are random integer in Z_q .The client computes them as follows: $k_{i1} \| k_{i2} = H_1(Y^x, m_i)$ .And Client compute $\sigma_{i,1} = (Y^{H_2(m_i)k_{i1}+k_{i2}})^x, \sigma_{i,2} = X^{k_{i1}}$, $\sigma_{i,3} = X^{k_{i2}}$, then denote the set by $\Phi = \{\sigma_{i,1}, \sigma_{i,2}, \sigma_{i,3}\}, 1 \le i \le n$ as the tag for block m_i.The client sends $T_m = \{F, \Phi\}$ to the CSS and deletes them from its local storage.

$GenChal(k) \to chal$

The client or the designated verifier can verify the integrity of the outsourced data by challenging the server. Verifier picks a random subset *I* of the set [1,*n*], For $i \in I (1 \le i \le c)$,the verifier chooses a random element $v_i \in Z_q$. The verifier sends the message $chal=\{(i, v_i)\}_{i \in I}$ to the CSS.

$Genproof(F, \phi, chal) \to DV$

Upon receiving the challenge, the CSS computes

$$\sigma = \prod_{i=1}^{c} \sigma_{i,1}^{v_i} \ , \ \delta = \prod_{i=1}^{c} \sigma_{i,2}^{H_2(m_i)v_i}, \quad \eta = \prod_{i=1}^{c} \sigma_{i,3}^{v_i}$$

Moreover the CSS will also provide the verifier with a small amount of metadata information. The CSS outputs $pf=\{\sigma, \delta, \eta\}$ and sends *pf* to the verifier as the response.

$VerifyProof(X, y, pf, chal) \to \{true, false\}$

Upon receiving the response *pf* from the CSS, the designated verifier checks whether the following formula holds.

$$\sigma = (\delta\eta)^y$$

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

4.2. Security Analysis

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

Theorem 1.If Client and CSS are honest and follow the proposed procedures, then any challenge-response can pass verifier's checking, i.e., DV-PDP satisfies the correctness.

Proof: According to our scheme procedures, we know that

$$\sigma = \prod_{i=1}^{c} \sigma_{i,1}^{v_i}$$
$$= \prod_{i=1}^{c} (Y^{(H_2(m_i)k_{i1}+k_{i2})x})^{v_i}$$
$$= \prod_{i=1}^{c} (g^{(H_2(m_i)k_{i1}+k_{i2})xy})^{v_i}$$
$$= \prod_{i=1}^{c} (X^{(H_2(m_i)k_{i1}+k_{i2})y})^{v_i}$$
$$= \prod_{i=1}^{c} (X^{H_2(m_i)k_{i1}y})^{v_i} \prod_{i=1}^{c} (X^{k_{i2}y})^{v_i}$$
$$= \prod_{i=1}^{c} (\sigma_{i,2}{}^{H_2(m_i)y})^{v_i} \prod_{i=1}^{c} (\sigma_{i,3}{}^{y})^{v_i}$$
$$= (\delta\eta)^y$$

Theorem 2.If a (t', ε')-algorithm *A*, operated by an adversary, can generate a forgery tag under our DV-PDP scheme after making at most H_1 hash queries, at most q_T tag queries and requesting q_K setup, then there exists a (t, ε) –algorithm *B* that can solve the CDH problem in G with $t \le t' + q_{H_1}T_G + q_T T_G$ and $\varepsilon \ge \varepsilon'/q_K q_{H_2}$, where one exponentiation on G takes time T_G.

Proof: Let *A* be a probabilistic black-box adversary who wins the tag unforgeability game with advantage ε'in time t'. On input (g, g^a, g^b) the CDH algorithm B simulates *A* as follows:

Setup: Given an instance (g, g^a, g^b) of the CDH problem. *B* sets the public parameter (G, g, q).As *A* requests the creation of system users, *B* guesses which one *A* will attempt a forgery against. Without loss of generality, we assume the target public key as pk_v and set it as $pk_v = g^a$. For all other public keys, we set $pk_i = g^{x_i}$ for a random $x_i \in Z_q$. Then *B* can invoke *A* to query. The total number requested is q_K.

Query: *A* can query oraclesO_{H_1},O_{H_2},O_{Tag} during his execution. *B* handles these oracles as follows:

--O_{H_1}: *B* maintains a table $T_{H_1}$to look up the O_{H_1} query records. *B* takes m_i as input, if record $(*, m_i)$ exists, then it outputs (k_{i1}, k_{i2}). Otherwise guess if $m_i$is the block m^* that *A* will attempt to use in a forgery. If $m_i = m^*$, output $Y^{H_2(m_i)k_{i1}+k_{i2}}=g^b$; otherwise, select a random $y_i \in Z_q$, let $Y^{H_2(m_i)k_{i1}+k_{i2}}=g^{y_i}$, and inserts (g^{ay_i}, m_i) into T_{H_1} for each $m_i \ne m^*$.

--O_{H_2}: *B* maintains a table $T_{H_2}$to look up the O_{H_2} query records. *B* takes $u \in (0,1)^*$ as input and outputs ρ if record (u, ρ)exists inT_{H_2}. Otherwise, *B* randomly selects l from Z_q and inserts (u, l) into T_{H_2}.

--O_{Tag}: *B* maintains a table T_{Tag} to look up the O_{Tag} query records. *B* takes $F=\{m_1, m_2, \cdots, m_n\}$ as input. For $1 \le i \le n$, if m_i has been queried to oracle O_{H_1}, *B* aborts. Otherwise, *B* randomly choose r_1 , r_2 from Z_q , and let $k_{i1} = r_1/a$, $k_{i2} = r_2/a$. So $\sigma_{i,1} = (Y^{H_2(m_i)k_{i1}+k_{i2}})^a = g^{b(H_2(m_i)r_1+r_2)}$,$\sigma_{i,2} = X^{k_{i1}}$, $\sigma_{i,3} = X^{k_{i2}}$. Then *B* inserts $(m_i, \sigma_{i,1}, \sigma_{i,2}, \sigma_{i,3})$ to T_{Tag}.

Forgery: Eventually *A* outputs a forgery $(pk_j, m', \sigma_{i,1}', \sigma_{i,2}', \sigma_{i,3}')$, *B* responses them as challenge to *A*. If $v \ne j$, then *B* guessed the wrong target user and must abort. If Verify Proof=*false*, or $(m', \sigma_{i,1}', \sigma_{i,2}', \sigma_{i,3}')$ is the result of any O_{Tag}, *B* also aborts. Otherwise *B* let $\sigma_{i,1}' = (Y^{H_2(m_i)k_{i1}+k_{i2}})^a = (g^b)^a = g^{ab}$ as the proposed CDH problem. So $k_{i1}'\|k_{i2}' = H_1(g^{ab}, m')$,$\sigma_{i,2}' = g^{ak_{i1}'}$,$\sigma_{i,3}' = g^{ak_{i2}'}$.

The probability that *B* will guess the target user correctly is $\frac{1}{q_K}$,and the probability that *B*

will guess the forged block m' is $\frac{1}{q_{H_1}}$. Therefore, if *A* generates a forgery tag with probability ε', then *B* solves the CDH problem with probability $\varepsilon'/q_K q_{H_1}$. Algorithm *B* requires one exponentiation on G for each H_1 query, one extra exponentiation on G for each tag query, so its running time is *A*'s running time plus $q_{H_1} c_G + q_T c_G$.

4.3. Performance Evaluation

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

4.3.1. Computation Cost

In our DV-PDP protocol, suppose there exist *n* message blocks. In the *TagGen* phase, the client needs to compute 3integer exponentiation and 1 multiplication. In the *GenProof* phase, the CCS needs to do 3c integer exponentiations and 3c multiplications. In the *VerifyProof* phase, the verifier needs to do 1 multiplication and 1 integer exponentiation. Other operations like hashing and permutation are omitted since they just contribute negligible computation cost. Our scheme does need to compute expensive pairing, which improves its computational efficiency considerably.

4.3.2. Communication Cost

Compared to RSA, elliptic curves cryptography (ECC) has shorter key length based on the same level of security. It has been shown that 160-bit ECC provides comparable security to 1024-bit RSA. At the same time, 224-bit ECC provides comparable security to 2048-bitRSA. The communication overhead incurred mostly comes from the DV-PDP response. In DV-PDP response, the CCS needs to send 3 elements in *G*. The total communication is about 480 bits. This communication overhead is totally tolerable for current communication techniques. We do not consider the communication overhead incurred by storing their data on CCS. For storing processes, the communication overheads are more similar than the schemes in [19]. Thus, the total communication overhead of DV-PDP is more efficient.

5. Conclusions

In this paper, we propose the designated verifier provable data possession model that provides authorized verification on remote data. DV-PDP enables a designated trusted third party to check data integrity under data owner's permission. Moreover the DV-PDP is more efficient because expense pairing is not calculated.

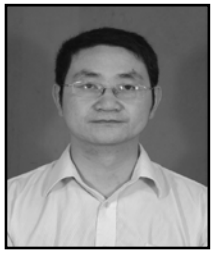
Acknowledgements

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] Q.Wang, C.Wang, J. Li, K. Ren, and W. Lou. Enabling public verifiability and data dynamics for storage security in cloud computing. IEEE Transactions on Parallel and Distributed Systems, 22,847**(2011)**
[2] Yan Zhu, Hongxin Hu,Gail-Joon Ahn and Mengyang Yu. Cooperative Provable Data Possession for Integrity Verification in Multicloud Storage. IEEE Transactions on Parallel and Distributed Systems, 23, 12**(2012)**
[3] Cong Wang, Sherman S.M. Chow, Qian Wang, KuiRen, and Wenjing Lou, Privacy-Preserving Public Auditing for Secure Cloud Storage, IEEE Transactions on Computers (TC), 10, 451**(2012)**
[4] Kan Yang, Xiaohua Jia. Data storage auditing service in cloud computing: challenges, methods and opportunities. The journal of World Wide Web. 15, 409**(2012)**
[5] Huaqun Wang. Proxy Provable Data Possession in Public Clouds. IEEE Transactions on Services Computing, P, P**(2012)**
[6] 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)** October598-609;Alexandria, VA, USA
[7] R.Johnson, D.Molnar, D.song, and D.wagner. Homomorphic signature schemes. In Proc. of CT-RSA, **(2002)** Feb 244-262; San Jose,CA, USA
[8] A.Juels and B. Kaliski. PORs: Proofs of retrievability for large files. In CCS '07, **(2007)** October584-597; Alexandria, VA, USA
[9] Shacham and B. Waters. Compact proofs of retrievability.In ASIACRYPT '2008, **(2008)** December 90-107;Melbourne, Australia
[10] K. D. Bowers, A. Juels, A. Oprea, Proofs of Retrievability: Theory and Implementation, Proc. 2009 ACM Cloud Computing Security Workshop, **(2009)** November 43-54;Chicago, IL, USA
[11] Y. Dodis, S. Vadhan, and D. Wichs.Proofs of retrievability via hardness application. In TCC'09, **(2009)** March. 109-127; San Francisco, CA, USA
[12] Qingji Zheng and Shouhuai Xu. Fair and Dynamic Proofs of Retrievability. CODASPY'11, **(2011)** February 21–23; San Antonio, Texas, USA.
[13] G.Ateniese, S.Kamara, and J.Katz.Proofs of storage from homomorphic identification protocols. ASIACRYPT'09, **(2009)** Dec 319-333;Tokyo,Japan
[14] D Boneh, B Lynn, H Shacham. Short signatures from the weil pairing. ASIACRYPT 2001. **(2001)** Dec 514-532;Gold Coast, Australia
[15] C. Erway, A. Kupcu, C. Papamanthou, and R. Tamassia. Dynamic provable data possession. In CCS '09, **(2009)** November 213-222; Chicago, IL, USA
[16] ShaiHalevi , Danny Harnik, Benny Pinkas, and Alexandra Shulman-Peleg. Proofs of Ownership in Remote Storage Systems.The Proceedings of the 18th ACM conference on Computer and communications security, **(2011)** November 491-500;Chicago, IL, USA
[17] C. Erway, A. Kupcu, C. Papamanthou, and R. Tamassia.Dynamic provable data possession. In CCS '09, **(2009)** November 213-222; Chicago, IL, USA
[18] Boyang Wang, Baochun Li, Hui Li, Public Auditing for Shared Data with Efficient User Revocation in the Cloud, accepted by INFOCOM 2013, **(2013)** July15-19;Turin, Italia
[19] Shiuan-TzuoShen, Wen-GueyTzeng. Delegable Provable Data Possession for Remote Data in the Clouds.InICICS 2011, **(2011)**, December 93–111;Singapore

Authors



Yongjun Ren, 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, 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, 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.

Jeong-Uk Kim obtained his B.S. degree in Control and Instrumentation Engineering from Seoul National University in 1987, M.S. and Ph.D. degrees in Electrical Engineering from Korea Advanced Institute of Science and Technology in 1989, and 1993, respectively. He is a professor in Sangmyung University in Seoul. His research interests include smart grid demand response, building automation system, and renewable energy.

