ISSN No. 0976-5697

Volume 4, No. 2, Jan-Feb 2013

International Journal of Advanced Research in Computer Science

RESEARCH PAPER

Available Online at www.ijarcs.info

A Quasi group Encryption based Privacy-Preserving Public Auditing for Data Storage Security in Cloud Computing

K.Suganthi*
Research Scholar Computer Science
Gobi Arts and Science College
Erode, India
sugann@ymail.com

Dr.B.Srinivasan
Associate Professor Computer Science
Gobi Arts and Science College
Erode, India
Srinivasan_gasc@yahoo.com

Mr.P.Naredran
Head of Department Computer Science
Gobi Arts and Science College
Erode, India
naredranp@gmail.com

Abstract: Cloud Computing has been envisioned as the nextgeneration architecture of IT Enterprise. In contrast to traditional solutions, where the IT services are under proper physical, logical and personnel controls, Cloud Computing moves the application software and databases to the large data centers, where the management of the data and services may not be fully trustworthy. This unique attribute, however, poses many new security challenges which have not been well understood. From the perspective of data security, which has always been an important aspect of quality of service, Cloud Computing inevitably poses new challenging security threats for number of reasons. Firstly, traditional cryptographic primitives for the purpose of data security protection can not be directly adopted due to the users 'loss control of data under Cloud Computing. However, the fact that users no longer have physical possession of the possibly large size of outsourced data makes the data integrity protection in Cloud Computing a very challenging and potentially formidable task, especially for users with constrained computing resources and capabilities. Thus, enabling public auditability for cloud data storage security is of critical importance so that users can resort to an external audit party to check the integrity of outsourced data when needed. In this research work, a Third Party Auditor is introduced and an effective quasi group algorithm is proposed in this work.

Keywords: Cloud Computing, privacy-preserving public auditing, public key based homomorphic, Third Party Auditor.

I. INTRODUCTION

Several trends are opening up the era of Cloud Computing, which is an Internet-based development and use of computer technology. The ever cheaper and more powerful processors, together with the software as a service (SaaS) computing architecture, are transforming data centers into pools of computing service on a huge scale. The increasing network bandwidth and reliable yet flexible network connections make it even possible that users can now subscribe high quality services from data and software that reside solely on remote data centers.

Moving data into the cloud offers great convenience to users since they don't have to care about the complexities of direct hardware management. The pioneer of Cloud Computing vendors, Amazon Simple Storage Service (S3) and Amazon Elastic Compute Cloud (EC2) are both well-known examples. While these internet- based online services do provide huge amounts of storage space and customizable computing resources, this computing platform shift, however, is eliminating the responsibility of local machines for data maintenance at the same time. As a result, users are at the mercy of their cloud service providers for the availability and integrity of their data [1].

The data stored in the cloud may be frequently updated by the users, including insertion, deletion, modification, appending, reordering, etc. To ensure storage correctness under dynamic data update is hence of paramount importance.

However, this dynamic feature also makes traditional integrity insurance techniques futile and entails new solutions. Last but not the least, the deployment of Cloud Computing is powered by data centers running in a simultaneous, cooperated and distributed manner. Individual user's data is redundantly stored in multiple physical locations to further reduce the data integrity threats. Therefore, distributed protocols for storage correctness assurance will be of most importance in achieving a robust and secure cloud data storage system in the real world. However, such important area remains to be fully explored in the literature. Recently, the importance of ensuring the remote data integrity has been highlighted by the following research works. These techniques, while can be useful to ensure the storage correctness without having users possessing data, cannot address all the security threats in cloud data storage, since they are all focusing on single server scenario and most of them do not consider dynamic data operations. As a complementary approach, researchers have also proposed distributed protocols for ensuring

storage correctness across multiple servers or peers. Again, none of these distributed schemes is aware of dynamic data operations [2].

Therefore, to fully ensure the data security and save the cloud users' computation resources, it is of critical importance to enable public auditability for cloud data storage so that the users may resort to a third party auditor (TPA), who has expertise and capabilities that the users do not, to audit the outsourced data when needed. Based on the audit result, TPA could release an audit report, which would not only help users to evaluate the risk of their subscribed cloud data services, but also be beneficial for the cloud service provider to improve their cloud based service platform [3]. In a word, enabling public risk auditing protocols will play an important role for this nascent cloud economy to become fully established, where users will need ways to assess risk and gain trust in Cloud.

Recently, the notion of public auditability has been proposed in the context of ensuring remotely stored data integrity under different systems and security models [4, 5, 6, 7]. Public auditability allows an external party, in addition to the user himself, to verify the correctness of remotely stored data. However, most of these schemes do not support the privacy protection of users' data against external auditors, i.e., they may potentially reveal user data information to the auditors.

Therefore, how to enable a privacy-preserving third-party auditing protocol, independent to data encryption, is the problem we are going to tackle in this paper. Our work is among the first few ones to support privacy-preserving public auditing in Cloud Computing, with a focus on data storage. Besides, with the prevalence of Cloud Computing, a foreseeable increase of auditing tasks from different users may be delegated to TPA.

II.LITERATURE SURVEY

Using Cloud Storage, users can remotely store their data and enjoy the on-demand high quality applications and services from a shared pool of configurable computing resources, without the burden of local data storage and maintenance. However, the fact that users no longer have physical possession of the outsourced data makes the data integrity protection in Cloud Computing a formidable task, especially for users with constrained computing resources. Moreover, users should be able to just use the cloud storage as if it is local, without worrying about the need to verify its integrity. Thus, enabling public auditability for cloud storage is of critical importance so that users can resort to a third party auditor (TPA) to check the integrity of outsourced data and be worry-free. To securely introduce an effective TPA, the auditing process should bring in no new vulnerabilities towards user data privacy, and introduce no additional online burden to user. In this paper, we propose a secure cloud storage system supporting privacy-preserving public auditing. We further extend our result to enable the TPA to perform audits for multiple users simultaneously and efficiently. Extensive security and performance analysis show the proposed schemes are provably secure and highly efficient.

Cloud computing is the next stage in evolution of the internet, which provides large amount of computing and storage to customers provisioned as a service over the internet. However, cloud computing facing so many security challenges due to the possible compromise or byzantine failures. In this paper, we focus on Ensuring data storage security in cloud computing, which is an important aspect of Quality of Service (QoS). We propose an effective and flexible distribution verification protocol to address data storage security in cloud computing. In this protocol, we rely on erasure code for the availability, reliability of data and utilize token pre-computation using Sobol Sequence to verify the integrity of erasure coded data rather than Pseudorandom Data in existing system. Unlike prior works, our scheme provides more security to user data stored in cloud computing. The performance analysis shows that our scheme is more secure than existing system against Byzantine failure, unauthorized data modification attacks, and even cloud server colluding attacks.

Ateniese et al. [8] are the first to consider public auditability in their defined "provable data possession" (PDP) model for ensuring possession of data files on untrusted storages. Their scheme utilizes the RSA-based homomorphic authenticators for auditing outsourced data and suggests randomly sampling a few blocks of the file. However, the public auditability in their scheme demands the linear combination of sampled blocks exposed to external auditor. When used directly, their protocol is not provably privacy preserving, and thus may leak user data information to the auditor.

Juels et al. [9] describe a "proof of retrievability" (PoR) model, where spot-checking and error-correcting codes are used to ensure both "possession" and "retrievability" of data files on remote archive service systems. However, the number of audit challenges a user can perform is a fixed priori, and public auditability is not supported in their main scheme. Although they describe a straightforwardMerkle-tree construction for public PoRs, this approach only works with encrypted data.

III. METHODOLOGY

Cloud computing is a general term for anything that involves delivering hosted services over the internet. These services are broadly divided into three categories: Infrastructure-as-a-Service (IaaS), Platform-as-a-Service (PaaS) and Software-as-a-Service (SaaS). A cloud service has three distinct characteristics that differentiate it from traditional hosting. It is sold on demand, typically by the minute or the hour; it is elastic - a user can have as much or as little of a service as they want at any given time and the service is fully managed by the cloud service provider (the consumer needs nothing but a personal computer and Internet access).The advantage of cloud is cost savings. The prime disadvantage is security. Cloud computing is used by many software industries nowadays. Since the security is not provided in cloud, many companies adopt their unique

security structure [3]. For eg) Amazon has its own security .structure. Introducing a new and uniform security structure for all types of cloud is the problem we are going to tackle in this paper. Since the data placed in the cloud is accessible to everyone, security is not guaranteed .I propose a method to build a trusted computing environment for Cloud Computing system by providing Secure cross platform in to Cloud Computing system. In this method some important security services including authentication, encryption and decryption and compression are provided in Cloud Computing system.

This work is among the first few ones to support privacy-preserving public auditing in Cloud Computing, with a focus on data storage. Besides, with the prevalence of Cloud Computing, a foreseeable increase of auditing tasks from different users may be delegated to TPA. As the individual auditing of these growing tasks can be tedious and cumbersome, a natural demand is then how to enable TPA to efficiently perform the multiple auditing tasks in a batch manner, i.e., simultaneously. To address these problems, our work utilizes the technique of public key based homomorphic authenticator which enables TPA to perform the auditing without demanding the local copy of data and thus drastically reduces the communication and computation overhead as compared to the straightforward data auditing approaches. By integrating the homomorphic authenticator with random mask technique our protocol guarantees that TPA could not learn any knowledge about the data content stored in the cloud server during the efficient auditing process. The aggregation and algebraic properties of the authenticator further benefit our design for the batch auditing.

A. The System and Threat Model:

We consider a cloud data storage service involving three different entities, as illustrated in Fig. 1: the cloud user (U), who has large amount of data files to be stored in the cloud; the cloud server (CS), which is managed by cloud service provider (CSP) to provide data storage service and has significant storage space and computation resources (we will not differentiate CS and CSP hereafter.); the third party auditor (TPA), who has expertise and capabilities that cloud users do not have and is trusted to assess the cloud storage service security on behalf of the user upon request.

Users rely on the CS for cloud data storage and maintenance. They may also dynamically interact with the CS to access and update their stored data for various application purposes.

The users may resort to TPA for ensuring the storage security of their outsourced data, while hoping to keep their data private from TPA. We consider the existence of a semi-trusted CS in the sense that in most of time it behaves properly and does not deviate from the prescribed protocol execution. While providing the cloud data storage based services, for their own benefits the CS might neglect to keep or deliberately delete rarely accessed data files which belong to ordinary cloud users. Moreover, the CS may decide to hide the data corruptions caused by server hacks or Byzantine failures to maintain reputation. We assume the TPA, who is in the business of auditing, is reliable and independent, and thus has no incentive to collude with either the CS or the users during the auditing process. TPA should be able to efficiently audit the cloud data storage without local copy of data and without bringing in additional on-line burden to cloud users. However, any possible leakage of user's outsourced data towards TPA through the auditing protocol should be prohibited.

Note that to achieve the audit delegation and authorize CS to respond to TPA's audits, the user can sign a certificate granting audit rights to the TPA's public key, and all audits from the TPA are authenticated against such a certificate. These authentication handshakes are omitted in the following presentation.

B. Design Goals:

To enable privacy-preserving public auditing for cloud data storage under the aforementioned model, our protocol design should achieve the following security and performance guarantee: 1) Public auditability: to allow TPA to verify the correctness of the cloud data on demand without retrieving a copy of the whole data or introducing additional on-line burden to the cloud users; 2) Storage correctness: to ensure that there exists no cheating cloud server that can pass the audit from TPA without indeed storing users' data intact; 3) Privacy-preserving: to ensure that there exists no way for TPA to derive users' data content from the information collected during the auditing process; 4) Batch auditing: to enable TPA with secure and efficient auditing capability to cope with multiple auditing delegations from possibly large number of different users simultaneously; 5) Lightweight: to allow TPA to perform auditing with minimum communication and computation overhead.

a. Public Auditability Scheme:

In the introduction we motivated the public auditability with achieving economies of scale for cloud computing. This section presents our public auditing scheme for cloud data storage security.

We start from the overview of our public auditing system and discuss two straightforward schemes and their demerits. Then we present our main result for privacy-preserving public auditing to achieve the aforementioned design goals. We also show how to extent our main scheme to support batch auditing for TPA upon delegations from multi-users. Finally, we discuss how to adapt our main result to support data dynamics.

A. Definitions and Framework of Public Auditing System:

We follow the similar definition of previously proposed schemes in the context of remote data integrity checking [10] and adapt the framework for our privacy-preserving public auditing system.

A public auditing scheme consists of four algorithms (KeyGen, SigGen, GenProof, VerifyProof).

KeyGen is a key generation algorithm that is run by the user to setup the scheme. SigGen is used by the user to generate verification metadata, which may consist of MAC,

signatures, or other related information that will be used for auditing. GenProof is run by the cloud server to generate a proof of data storage correctness, while VerifyProof is run by the TPA to audit the proof from the cloud server.

Our public auditing system can be constructed from the above auditing scheme in two phases, Setup and Audit:

Setup: The user initializes the public and secret parameters of the system by executing KeyGen, and pre-processes the data file F by using SigGen to generate the verification metadata. The user then stores the data file F at the cloud server, deletes its local copy, and publishes the verification metadata to TPA for later audit. As part of pre-processing, the user may alter the data file F by expanding it or including additional metadata to be stored at server.

Audit: The TPA issues an audit message or challenge to the cloud server to make sure that the cloud server has retained the data file F properly at the time of the audit. The cloud server will derive a response message from a function of the stored data file F by executing

GenProof. Using the verification metadata, the TPA verifies the response via VerifyProof.

Note that in our design, we do not assume any additional property on the data file, and thus regard error-correcting codes as orthogonal to our system. If the user wants to have more error- resiliency, he/she can first redundantly encode the data file and then provide us with the data file that has error-correcting codes integrated.

B. The Basic Schemes:

Before giving our main result, we first start with two warmup schemes. The first one does not ensure privacy-preserving guarantee and is not as lightweight as we would like. The second one overcomes the first one, but suffers from other undesirable systematic demerits for public auditing: bounded usage and auditor statefulness, which may pose additional on-line burden to users as will be elaborated shortly. We believe the analysis of these basic schemes will lead us o our main result, which overcomes all these drawbacks.

Basic Scheme I The cloud user pre-computes MACs _i = MACsk(i||mi) of each block mi (i ∈ {1, . . . , n}), sends both the data file F and the MACs {_i}1≤i≤n onto the clou d server, and releases the secret key sk to TPA. During the Audit phase, the TPA requests from the cloud server a number of randomly selected blocks and their corresponding MACs to verify the correctness of the data file. The insight behind this approach is that auditing most of the file is much easier than the whole of it. However, this simple solution suffers from the following severe drawbacks: 1) The audit from TPA demands retrieval of users' data, which should be prohibitive because it violates the privacy-preserving guarantee; 2) Its communication and computation complexity are both linear with respect to the sampled data size, which may result in large communication overhead and time delay, especially when the bandwidth available between the TPA and the cloud server is limited.

Basic Scheme II To avoid retrieving data from the cloud server, one may improve the above solution as follows: Before data outsourcing, the cloud user chooses s random message authentication code keys {sk_}1≤_≤s pre - computes s MACs, {MACsk_ (F)}1≤_≤s for the whole data file F, and publishes these verification metadata to TPA. The TPA can each time reveal a secret key sk_ to the cloud server and ask for a fresh keyed MAC for comparison, thus achieving privacy-preserving auditing. However, in this method: 1) the number of times a particular data file can be audited is limited by the number of secret keys that must be a fixed priori. Once all possible secret keys are exhausted, cloud user then has to retrieve data from the server in order to re-compute and re-publish new MACs to TPA. 2) The TPA has to maintain and update state between audits, i.e., keep a track on the possessed MAC keys. Considering the potentially large number of audit delegations from multiple users, maintaining such states for TPA can be difficult and error prone.

C. The Privacy-Preserving Public Auditing Scheme:

To effectively support public auditability without having to retrieve the data blocks themselves, we resort to the homomorphic authenticator technique [7,9,11]. Homomorphic authenticators are unforgeable verification metadata generated from individual data blocks, which can be securely aggregated in such a way to assure an auditor that a linear combination of data blocks is correctly computed by verifying only the aggregated authenticator. However, the direct adoption of these techniques is not suitable for our purposes, since the linear cobination of blocks may potentially reveal user data information, thus violating the privacy-preserving guarantee. Specifically, if enough number of the linear combinations of the same blocks are collected, the TPA can simply derive the user's data content by solving a system of linear equations.

To achieve privacy-preserving public auditing, we propose to uniquely integrate the homomorphic authenticator with random mask technique. In our protocol, the linear combination of sampled blocks in the server's response is masked with randomness generated by a pseudo random function (PRF). With random mask, the TPA no longer has all the necessary information to build up a correct group of linear equations and therefore cannot derive the user's data content, no matter how many linear combinations of the same set of file blocks can be collected. Meanwhile, due to the algebraic property of the homomorphic authenticator, the correctness validation of the block-authenticator pairs will not be affected by the randomness generated from a PRF, which will be shown shortly. Note that in our design, we use public key based homomorphic authenticator, specifically, the one in [10] which is based on BLS signature [11], to equip the auditing protocol with public auditability. Its flexibility in signature aggregation will further benefit us for the multi-task auditing.

Scheme Details Let G1, G2 and GT be multiplicative cyclic groups of prime order p, and e : G1 × G2 → GT be a bilinear map as introduced in preliminaries. Let g be the generator of G2. H(·) is a secure map-to-point hash function: {0, 1}∗ → G1, which maps strings uniformly to G1. Another hash function h(·) : GT → Zp maps group

element of GT uniformly to Zp. The proposed scheme is as follows:

Setup Phase:

(a). The cloud user runs KeyGen to generate the system's public and secret parameters. He chooses a random x ← Zp, a random element u ← G1, and computes v ← gx. The secret parameter is sk = (x) and the public parameters are pk = (v, g, u, e(u, v)). Given data file F = (m1, . . . ,mn), the user runs SigGen to compute signature _i for each block m_i: $\sigma_i \leftarrow (H(i).u^{mi})^x \in G_1 (i = 1, \ldots, n)$. Denote the set of signatures by $\Phi = \{\sigma_i\}_{1 \le i \le n}$. The user then sends {F, Φ } to the server and deletes them from its local storage.

Audit Phase:

(b). During the auditing process, to generate the audit message "chal", the TPA picks a random c-element subset I = {s1, . . . , sc} of set [1, n], where $s_q = \pi k_{prp}(q) \; for \; 1 \le q \le c$ and k_{prp} is the randomly chosen permutation key by TPA for each auditing.

IV. IMPROVED STORAGE SECURITY MODEL

The system is designed to perform public data integrity analysis on cloud environment. Multi user based data file management is used in the system. Authentication and integrity schemes are improved in the system. Bilinear aggregate signature model is enhanced to manage multi user based data dynamic.

The system includes confidentiality for data security. The system is designed to manage shared storage with security. Data providers share the storage space under the data centers. Third party auditor (TPA) verifies the data integrity for the data providers. The system is divided into five major modules. Data center, data provider, third party auditor, data distribution process and security management. The data center provides the shared storage for the data providers. The data provider module is designed to share data sources.

Third party auditor handles the public auditability process. Data distribution module is designed to manage data update and delivery process. Integrity analysis is performed in security management module.

a. Data Center:

The data center application is designed to allocate storage space for the data providers. One or more data centers can be used in the cloud environment. Different sized storage area is allocated for the data providers. Data files are delivered to the clients.

b. Data Provider:

Shared data files are provided by the data providers. Data providers are managed by multiple users. Each user is assigned with separate authentication code. Data update operations are handled by the users.

c. Third party Auditor:

The third party auditor module is designed to provide security for the cloud. Data files and their signatures are maintained under the TPA. Block based signature model is used in the system. Data dynamics initiates the signature update process.

d. Data Distribution Process:

The data values are requested by the clients. The client requests are processed by the data centers. Shared data file contents are delivered to the clients. All the access information is updated to the data centers.

e. Security Management:

Authentication and integrity analysis are carried out under the security management process. Block signature model is used in the integrity analysis. TPA performs the security process. Bilinear aggregate signature is used for multi user based parallel verification process. The quasi group algorithm is used for security. The encryption technique used in this approach is the quasigroup, which has significant data-scrambling properties and thus it has effectively used in symmetric cryptography. The main aim of the scrambler is to enhance the entropy at the output even in scenario where the input is constant. The massive complexity associated with the assignment of discovering scrambling transformation assures the effectiveness of the encryption process. Quasi-group encryption is a development that has permutation based scrambling at its basis.

The encryption technique used in this approach is the quasi group encryption technique [12]. The quasigroup encryptor has significant data-scrambling properties and thus, it has effectively used in symmetric cryptography. The main aim of the scrambler is to enhance the entropy at the output, even in scenario where the input is constant. The massive complexity associated with the assignment of discovering the scrambling transformation assures the effectiveness of the encryption process. Quasigroup encryption is a development that has permutation based scrambling [13] at its basis.

If Q is a quasi-group such that a1,a2,a3,….,an belong to it, then the encryption operation QE(Quasi-Encryptor) which is defined over the defined elements, maps those elements to another vector b1,b2,b3,….,bn such that the elements of the resultant vector also belong to the same quasi-group.

The mathematical equation used for encryption (basic level) is defined by:

$$E_a(a_1, a_2, a_3, \ldots, a_n) = b_1, b_2, b_3, \ldots b_n \quad (2)$$

where $b_1 = a * a_1, b_i = b_{i-1} * a_i$, i increments from 2 to the number of elements that have to be encrypted and a is the hidden key.

V. EXPERIMENTAL RESULTS

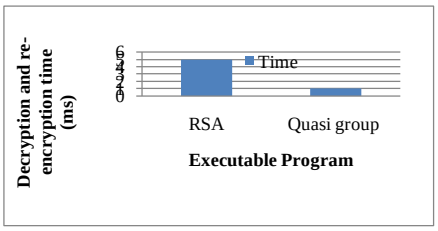

Figure 4.1: Comparison of Decryption and Re-Encryption Time

Figure 4.1 represents the comparison of decryption and re-encryption time. It is observed from the figure 4.1 that the Decryption and re-encryption time by the proposed quasi group encryption scheme is very less when compared with Reed Solomon code encryption scheme.

VI. CONCLUSION

An efficient privacy-preserving public auditing system for data storage security in Cloud Computing is proposed, where TPA can perform the storage auditing without demanding the local copy of data. We utilize the homomorphic authenticator and random mask technique to guarantee that TPA would not learn any knowledge about the data content stored on the cloud server during the efficient auditing process, which not only eliminates the burden of cloud user from the tedious and possibly expensive auditing task, but also alleviates the users' fear of their outsourced data leakage. An efficient cryptographic technique is used in this approach called the quasi group encryption technique for better security. Considering TPA may concurrently handle multiple audit sessions from different users for their outsourced data files, we further extend our privacy-preserving public auditing protocol into a multi-user setting, where TPA can perform the multiple auditing tasks in a batch manner, i.e., simultaneously. Extensive security and performance analysis shows that the proposed schemes are provably secure and highly efficient. We believe all these advantages of the proposed schemes will shed light on economies of scale for Cloud Computing.

VII. REFERENCES

[1] K. Valli Madhavi, R.Tamilkodi and R. BalaDinakar, "Data Storage Security in Cloud Computing for Ensuring Effective and Flexible Distributed System", National Conference on Research Trends in Computer Science and Technology – 2012.

[2] Cong Wang, Qian Wang, and Kui Ren, "Ensuring Data Storage Security in Cloud Computing".

[3] M. A. Shah, R. Swaminathan, and M. Baker, "Privacy-preserving audit and extraction of digital contents," Cryptology ePrint Archive, Report 2008/186, 2008, http://eprint.iacr.org/.

[4] G. Ateniese, R. Burns, R. Curtmola, J. Herring, L. Kissner, Z. Peterson, and D. Song, "Provable data possession at untrusted stores," Cryptology ePrint Archive, Report 2007/202, 2007, http://eprint. iacr.org/.

[5] Q. Wang, C. Wang, J. Li, K. Ren, and W. Lou, "Enabling public verifiability and data dynamics for storage security in cloud computing," in Proc. of ESORICS'09, Saint Malo, France, Sep. 2009.

[6] H. Shacham and B. Waters, "Compact proofs of retrievability," in Proc. of Asiacrypt 2008, vol. 5350, Dec 2008, pp. 90–107.

[7] 12. A. Juels and J. Burton S. Kaliski, "Pors: Proofs of retrievability for large files," in Proc. of CCS'07, Alexandria, VA, October 2007, pp. 584–597.

[8] G. Ateniese, R. Burns, R. Curtmola, J. Herring, L. Kissner, Z. Peterson, and D. Song, "Provable data possession at untrusted stores," Cryptology ePrint Archive, Report 2007/202, 2007, http://eprint. acr.org/.

[9] A. Juels and J. Burton S. Kaliski, "Pors: Proofs of retrievability for large files," in Proc. of CCS'07, Alexandria, VA, October 2007, pp. 584–597.

[10] H. Shacham and B. Waters, "Compact proofs of retrievability," in Proc. of Asiacrypt 2008, vol. 5350, Dec 2008, pp. 90–107.

[11] D. Boneh, B. Lynn, and H. Shacham, "Short signatures from the weil pairing," in Proc. of ASI- ACRYPT'01. London, UK: Springer-Verlag, 2001, pp. 514–532.

[12] C. Wang, Q. Wang, K. Ren, and W. Lou, "Privacy-preserving public auditing for data storage security in cloud computing," in InfoCom2010, IEEE, March 2010.

[13] C. Wang, Q. Wang, K. Ren, and W. Lou, "Ensuring data storage security in cloud computing," in IWQoS'09, pp. 1 –9, july 2009.