An Efficient Data Integrity Preservation Scheme by SSCE

G Loshma[1], Dr Nagaratna P Hegde[2]
[1]Associate Professor, CSE
Sri Vasavi Engineering College
Tadepalligudem, India

[2]Professor, CSE
Vasavi College of Engineering
Hyderabad, India

Abstract— *Data integrity is the most critical issue in the cloud computing paradigm. This paper proposes a methodology namely "Security Scheme for Cloud Environment with Semi-Trusted Third Party (SSCE)", in order to preserve the integrity of the outsourced data. To achieve the goal, the proposed work is decomposed into five different phases such as key management, construction of auditing scheme, cloud storage server, proxy server and regeneration code for authentication. All these phases work in a flow and the expected outcome is obtained. The performance of the proposed work is evaluated with respect to repair and auditing time. The experimental results prove that the time consumption for both audit and reparation of the proposed work is minimal, when compared to the existing work.*

Keywords— Data integrity; cloud; audit; security.

I. INTRODUCTION

Cloud computing is a new paradigm, which offers computational and storage resources based on demand. This provision is a boon for low and mid-scale industries, as the operational and capital cost can be minimized substantially. Cloud data storage is the mostly exploited service, which comes under the concept of outsourcing. Outsourcing brings in several benefits such as efficient storage management, hassle-free data management and reduced storage cost. However, there are several challenges associated with outsourcing, as the data is submitted to the untrustworthy cloud service provider. Some of the most facing issues are data confidentiality, integrity and privacy.

Cloud confidentiality must ensure the confidentiality of the outsourced data from the cloud service provider and the cloud users. Data integrity means that the data being stored in cloud should not be lost or modified. The data modification or loss must be detected by some mechanism. Privacy is a critical issue because the outsourced data may be highly sensitive, such that the cloud service provider and cloud users may gain access to the data. Among these, integrity is the major issue and this paper strives to provide a solution to the integrity issue.

In this paper, a public auditing scheme for the regenerating-code-based cloud storage is proposed for ensuring the data integrity and to save the users' computation resources. Besides this, online burden of the users are reduced. The proposed public auditing scheme checks integrity and regeneration of failed data blocks and authenticators are implemented by a third-party auditor accompanied by a semi-trusted proxy separately, on behalf of the data owner. A novel authenticator, which is more appropriate for regenerating code, is designed. Additionally, the coefficients are encrypted, in order to protect data privacy against the auditor, which is more lightweight than applying the proof blind technique.

This paper proposes "Security Scheme for Cloud Environment with Semi-Trusted Third Party (SSCE)", a data security system that provides key management, access control, and file assured deletion. This work utilizes Shamir's (k, n) threshold scheme to manage the keys, where *k* out of *n* shares are required to generate the key. Multiple key managers are utilized for each hosting one share of key.

Multiple key managers avoid single point of failure for the cryptographic keys. The work prototype is developed and the performance is evaluated with respect to time consumption during various operations, the execution of SSCE is checked by High Level Petri nets (HLPN) and Satisfiability Modulo. The results reveal that SSCE can be effectively used for security of outsourced data by employing key management, access control, and file assured deletion. This paper relies on five different modules and they are key management, construction of auditing scheme, cloud storage server, proxy server and regeneration code for authentication.

The rest of the paper is organized as follows. The related literature is reviewed in section 2. The proposed approach is presented in section 3. The performance of the system is evaluated in section 4. Finally, the concluding remarks are presented.

II. REVIEW OF LITERATURE

The existing literature with respect to the data integrity is presented in this section.

Compared to traditional systems, scalability and elasticity are key advantages of cloud [1-3]. As such, efficiency in supporting dynamic data is of great importance. Security and privacy protection on dynamic data has been studied extensively in the past [4-7]. Cloud users may also need to split big datasets into smaller datasets and store them

in different physical servers for reliability, privacy-preserving or efficient processing purposes.

Among the most pressing problems related to cloud is data security/privacy [8-10]. It has been one of the most frequently raised concerns [9,11]. There is a lot of work trying to enhance cloud data security/privacy with technological approaches on CSP side, such as [12,13].

Integrity verification for outsourced data storage has attracted extensive research interest. The concept of proofs of retrievability (POR) and its first model was proposed by Jules et al. [14]. Unfortunately, their scheme can only be applied to static data storage such as archive or library. In the same year, Ateniese, et al. proposed a similar model named 'provable data possession' (PDP) [16]. Their schemes offer 'blockless verification' which means the verifier can verify the integrity of a proportion of the outsourced file through verifying a combination of pre-computed file tags which they call homomorphic verifiable tags (HVTs) or homomorphic linear authenticators (HLAs).

Work by Shacham, et al. [16] provided an improved POR model with stateless verification. They also proposed a MAC-based private verification scheme and the first public verification scheme in the literature that based on BLS signature scheme [17]. In their second scheme, the generation and verification of integrity proofs are similar to signing and verification of BLS signatures. When wielding the same security strength (say, 80-bit security), a BLS signature (160 bit) is much shorter than an RSA signature (1024 bit), which is a desired benefit for a POR scheme. They also proved the security of both their schemes and the PDP scheme by Ateniese, et al. [15, 18].

From then on, the concepts of PDP and POR were in fact unified under this new compact POR model. Ateniese, et al. extended their scheme for enhanced scalability [5], but only partial data dynamics and a predefined number of challenges is supported.

In 2009, Erway, et al. proposed the first PDP scheme based on skip list that can support full dynamic data updates [6]. However, public auditability and variable-sized file blocks are not supported by default. Wang, et al. [4] proposed a scheme based on BLS signature that can support public auditing (especially from a thirdparty auditor, TPA) and full data dynamics, which is one of the latest works on public data auditing with dynamics support. However, their scheme lacks support for finegrained update and authorized auditing which are the main focuses of our work.

Latest work by Wang et al. [19] added a random masking technology on top of [4] to ensure the TPA cannot infer the raw data file from a series of integrity proofs. In their scheme, they also incorporated a strategy first proposed in [16] to segment file blocks into multiple 'sectors'. However, the use of this strategy was limited to trading-off storage cost with communication cost.

Other lines of research in this area include the work of Ateniese, et al. [20] on how to transform a mutual identification protocol to a PDP scheme; scheme by Zhu, et al. [21] that allows different service providers in a hybrid cloud to cooperatively prove data integrity to data owner; and the MR-PDP Scheme based on PDP [15] proposed by Curtmola, et al. [22] that can efficiently prove the integrity of multiple replicas along with the original data file.

III. Proposed Approach

The proposed approach employs five different modules such as key management, construction of auditing scheme, cloud storage server, proxy server and regeneration code for authentication. The overall flow of the work is presented in figure 1. All the phases of the proposed work are explained in the following subsections.

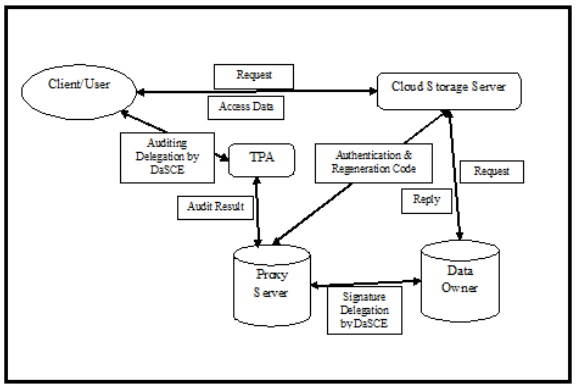

Fig. 1. Overall flow of the system

A. Key management

The proposed SSCE makes use of both symmetric and asymmetric keys. The confidentiality and integrity services for data are provided through symmetric keys that are secured by using asymmetric keys. Asymmetric key pairs are generated by third party *KMs*. Out of the key pair, only public key is transmitted to the client. For secure transmission of keys, a session key is established between client and *KM* through STS protocol. To avoid man-in-the-middle attack, both client and *KM* are authenticated by using digital signatures.

As a new session key is used for every communication session between client and *KM*, the session key is exchanged through key exchange process and is not randomly generated. This also avoids weakness of randomly generated keys. The symmetric keys are generated once for data encryption by client and encrypted by another symmetric key named *Si*. The *Si* is finally protected by the public key received from *KM*. The encrypted keys are stored at cloud and client deletes the local copies of the keys.

For decryption purpose, client establishes a session with *KM* and sends *Si* to *KM* after masking with random number R. The *KM* decrypts *Si* and sends back to client. The client unmasks *Si* to get the symmetric keys.

B. Construction of auditing scheme

The third party auditor (TPA), who has expertise and capabilities to conduct public audits on the coded data in the cloud, the TPA is trusted and its audit result is unbiased for both data owners and cloud servers; and a proxy agent, who is

semi-trusted and acts on behalf of the data owner to regenerate authenticators and data blocks on the failed servers during the repair procedure.

Our auditing scheme consists of three procedures:

- *Setup*
- *Audit*
- *Repair*

1. Setup:

KeyGen (1^{κ}) → *(pk,sk):*

This polynomial-time algorithm is run by the data owner to initialize its public and secret parameters by taking a security parameter κ as input.

Degelation (sk) → (x):

This algorithm represents the interaction between the data owner and proxy. The data owner delivers partial secret key *x* to the proxy through a secure approach.

Sig And BlockGen (sk, F) →(Φ, ψ ,*t):*

This polynomial time algorithm is run by the data owner and takes the secret parameter *sk* and the original file *F* as input, and then outputs a coded block set ψ , an authenticator set Φ and a file tag *t*.

2. Audit:

The cloud servers and TPA interact with one another to take a random sample on the blocks and check the data intactness in this procedure.

Challenge (F_{info}) → *(C):*

This algorithm is performed by the TPA with the information of the file F_{info} as input and a challenge *C* as output.

*Proof Gen (C,*Φ, ψ*)* → *(P):*

This algorithm is run by each cloud server with input challenge *C*, coded block set ψ and authenticator set Φ, then it outputs a proof *P*.

Verify (P,pk,C) → *(0,1):*

This algorithm is run by TPA immediately after a proof is received. Taking the proof *P*, public parameter *pk* and the corresponding challenge *C* as input, it outputs 1 if the verification passed and 0 otherwise.

3. Repair:

In the absence of the data owner, the proxy interacts with the cloud servers during this procedure to repair the wrong server detected by the auditing process.

Claim For Rep (F_{info}) → (C_r)*:*

This algorithm is similar with the Challenge () algorithm in the Audit phase, but outputs a claim for repair C_r.

Gen For Rep (Cr, Φ*,* ψ*)* → *(BA):*

The cloud servers run this algorithm upon receiving the Cr and finally output the block and authenticators set *BA* with another two inputs Φ, ψ.

Block And Sig ReGen (C_r, *BA)* →(Φ', ψ', ⊥*):*

The proxy implements this algorithm with the claim C_r and responses *BA* from each server as input, and outputs a new coded block set ψ' and authenticator set Φ' if successful, outputting ⊥ if otherwise.

C. Cloud storage server

Our scheme is the first to allow privacy-preserving public auditing for regenerating code-based cloud storage. The coefficients are masked by a PRF (Pseudorandom Function) during the Setup phase to avoid leakage of the original data. Cloud storage to be a collection of n storage servers, data file F is encoded and stored redundantly across these servers. Then F can be retrieved by connecting to any *k*-out-of-*n* servers, which is termed the MDS 2 -property. When a server is detected with data corruption, then the client must contact ℓ healthy servers. This is followed by the downloading of β bits from each server. Thus, the corrupted blocks are regenerated without recovering the entire original file.

D. Proxy server

Our scheme completely releases data owners from online burden for the regeneration of blocks and authenticators at faulty servers and it provides the privilege to a proxy for the reparation. A proxy agent, who is semi-trusted and acts on behalf of the data owner to regenerate authenticators and data blocks on the failed servers during the repair procedure. The proxy, who would always be online, is supposed to be much more powerful than the data owner but less than the cloud servers in terms of computation and memory capacity. The proxy agent in our system model is assumed to be semi-trusted. It will not collude with the servers but might attempt to forge authenticators for some specified invalid blocks to pass the following verification.

IV. Performance Evaluation

The performance of the proposed work is evaluated in terms of auditing time by varying the value of α, repair time, audit time with respect to count of sample segments. The proposed work is implemented in Java and the experimental results are presented from figure 2 to figure 4.

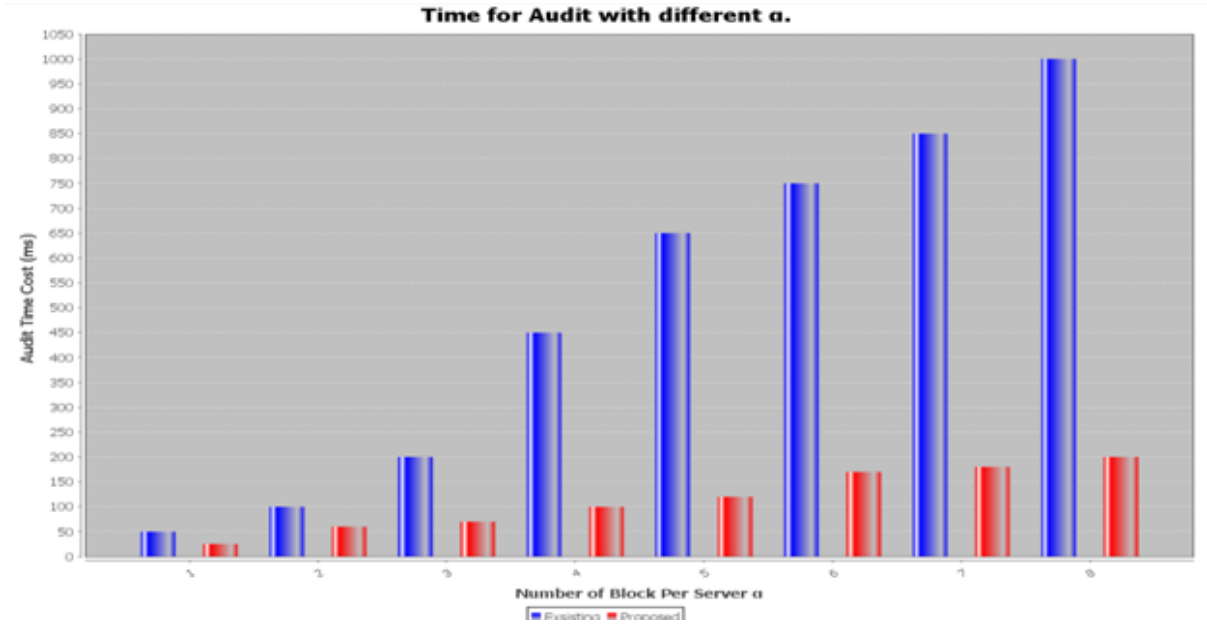

Fig. 2. Auditing time w.r.t α

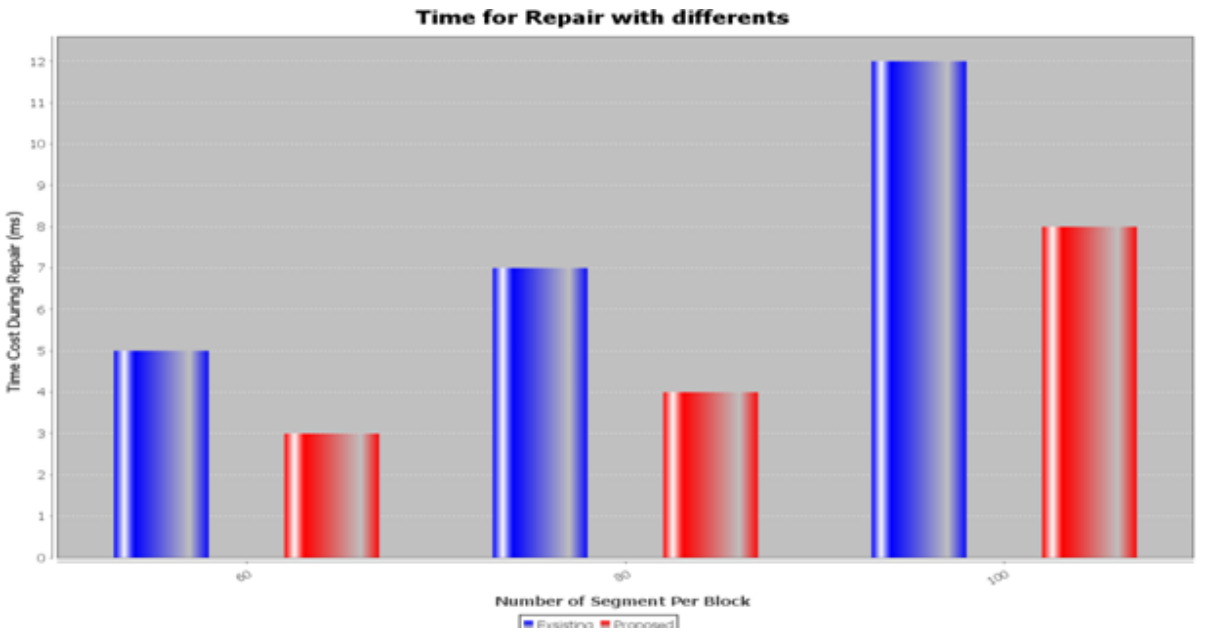

Fig. 3. Repair time w.r.t segment count

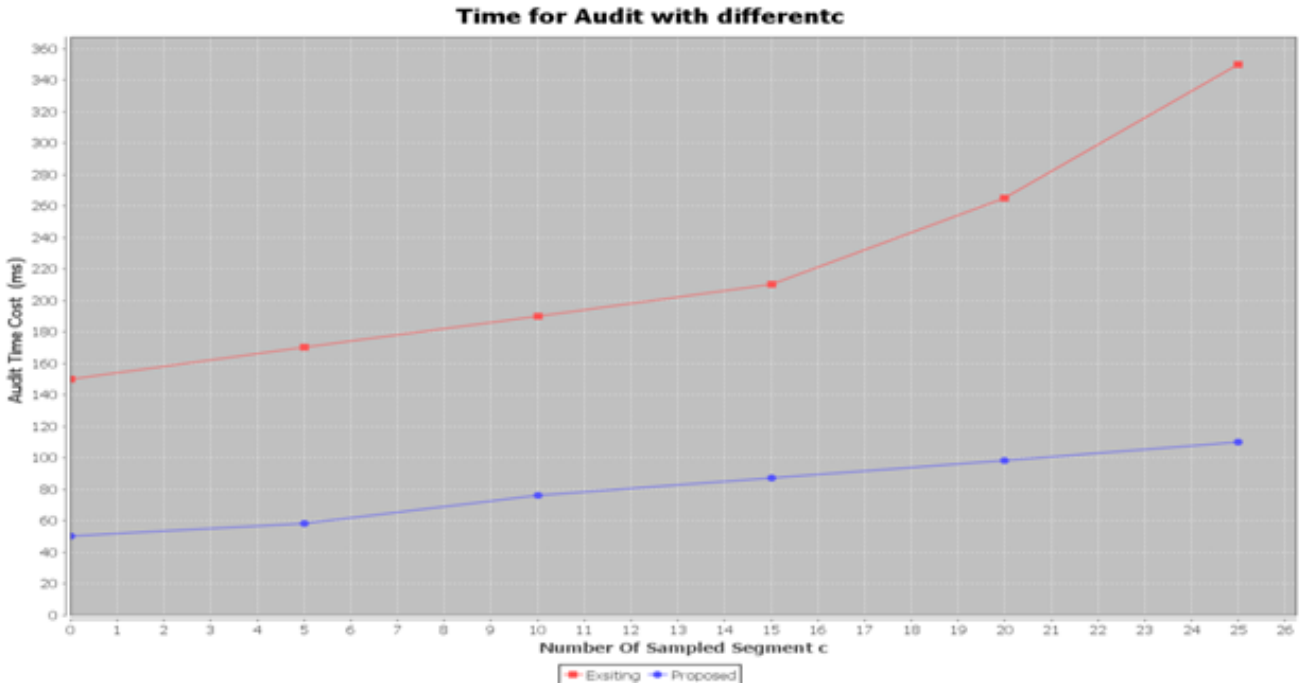

Fig.4. Audit time w.r.t segment count

From the experimental analysis, it is evident that the proposed work outperforms the existing work in terms of audit and repair time.

V. Conclusion

This paper employs a semi-trusted proxy into the system model and provides a privilege for the proxy to handle the reparation of the coded blocks and authenticators. The authenticator is designed on the basis of the BLS signature, so as to suit the code regeneration. This authenticator can be efficiently generated by the data owner simultaneously with the encoding procedure. Our scheme completely releases data owners from online burden for the regeneration of blocks and authenticators at faulty servers and it provides the privilege to a proxy for the reparation. The storage overhead of servers, the computational overhead of the data owner and communication overhead during the audit phase are effectively reduced. The performance of the proposed work is found to be satisfactory in terms of audit and repair time.

References

[1] R. Buyya, C.S. Yeo, S. Venugopal, J. Broberg, and I. Brandic, ‘‘Cloud Computing and Emerging IT Platforms: Vision, Hype, Reality for Delivering Computing as the 5th Utility,’’ Future Gen.Comput. Syst., vol. 25, no. 6, pp. 599-616, June 2009.

[2] M.Armbrust, A. Fox, R. Griffith, A.D. Joseph, R.Katz, A.Konwinski, G. Lee, D. Patterson, A. Rabkin, I. Stoica, andM. Zaharia, ‘‘A Viewof Cloud Computing,’’ Commun. ACM, vol. 53, no. 4, pp. 50-58, Apr. 2010.

[3] Customer Presentations on Amazon Summit Australia, Sydney, 2012, accessed on: March 25, 2013.[Online].Available:http://aws.amazon.com/apac/awssummit-au/.

[4] Q. Wang, C.Wang, K. Ren,W. Lou, and J. Li, ‘‘Enabling Public Auditability and Data Dynamics for Storage Security in Cloud Computing,’’ IEEE Trans. Parallel Distrib. Syst., vol. 22, no. 5, pp. 847-859, May 2011.

[5] G. Ateniese, R.D. Pietro, L.V. Mancini, and G. Tsudik, ‘‘Scalable and Efficient Provable Data Possession,’’ in Proc. 4th Int’l Conf. Security and Privacy in Commun. Netw. (SecureComm),2008, pp. 1-10.

[6] C. Erway, A. Ku¨ pc¸u¨ , C. Papamanthou, and R. Tamassia, ‘‘Dynamic Provable Data Possession,’’ in Proc. 16th ACM Conf. on Comput. and Commun. Security (CCS), 2009, pp. 213-222.

[7] Y. He, S. Barman, and J.F. Naughton, ‘‘Preventing Equivalence Attacks in Updated, Anonymized Data,’’ in Proc. 27th IEEE Int’l Conf. on Data Engineering (ICDE), 2011, pp. 529-540.

[8] J. Yao, S. Chen, S.Nepal,D. Levy, and J. Zic, ‘‘TrustStore: Making Amazon S3 Trustworthy With Services Composition,’’ in Proc. 10th IEEE/ACM Int’l Symposium on Cluster, Cloud and Grid Computing (CCGRID), 2010, pp. 600-605.

[9] D. Zissis and D. Lekkas, ‘‘Addressing Cloud Computing Security Issues,’’ Future Gen. Comput. Syst., vol. 28, no. 3, pp. 583-592, Mar. 2011.

[10] X. Zhang, L.T. Yang, C. Liu, and J. Chen, ‘‘A Scalable Two-Phase Top-Down Specialization Approach for Data Anonymization Using MapReduce on Cloud,’’ IEEE Trans. Parallel Distrib. Syst., vol. 25, no. 2, pp. 363-373, Feb. 2014.

[11] S.E. Schmidt, ‘‘Security and Privacy in the AWS Cloud,’’ presented at the Presentation Amazon Summit Australia, Sydney,Australia,May 2012, accessed on: March 25, 2013. [Online].Available: http://aws.amazon.com/apac/awssummit-au/.

[12] C. Liu, X. Zhang, C. Yang, and J. Chen, ‘‘CCBKEV Session Key Negotiation for Fast and Secure Scheduling of Scientific Applications in Cloud Computing,’’ Future Gen. Comput. Syst.,vol. 29, no. 5, pp. 1300-1308, July 2013.

[13] X. Zhang, C. Liu, S. Nepal, S. Panley, and J. Chen, ‘‘A Privacy Leakage Upper-Bound Constraint Based Approach for Cost-Effective Privacy Preserving of Intermediate Datasets in Cloud,’’ IEEE Trans. Parallel Distrib. Syst., vol. 24, no. 6, pp. 1192-1202, June 2013.

[14] Juels and B.S. Kaliski Jr., ‘‘PORs: Proofs of Retrievability for Large Files,’’ in Proc. 14th ACM Conf. on Comput. and Commun. Security (CCS), 2007, pp. 584-597.

[15] G.Ateniese, R.B. Johns,R. Curtmola, J.Herring, L. Kissner,Z. Peterson, and D. Song, ‘‘Provable Data Possession at Untrusted Stores,’’ in Proc. 14th ACM Conf. on Comput. and Commun. Security (CCS), 2007, pp. 598-609.

[16] H. Shacham and B. Waters, ‘‘Compact Proofs of Retrievability,’’ in Proc. 14th Int’l Conf. on Theory and Appl. of Cryptol. and Inf. Security (ASIACRYPT), 2008, pp. 90-107.

[17] D. Boneh, H. Shacham, and B. Lynn, ‘‘Short Signatures From the Weil Pairing,’’ J. Cryptol., vol. 17, no. 4, pp. 297-319, Sept. 2004.

[18] G. Ateniese, R. Burns, R. Curtmola, J. Herring, O. Khan, L. Kissner, Z. Peterson, and D. Song, ‘‘Remote Data Checking Using Provable Data Possession,’’ ACM Trans. Inf. Syst. Security, vol. 14, no. 1, May 2011, Article 12.

[19] Wang, Q. Wang, K. Ren, and W. Lou, ‘‘Privacy-Preserving Public Auditing for Data Storage Security in Cloud Computing,’’ in Proc. 30st IEEE Conf. on Comput. and Commun. (INFOCOM), 2010, pp. 1-9.

[20] G. Ateniese, S. Kamara, and J. Katz, ‘‘Proofs of Storage From Homomorphic Identification Protocols,’’ in Proc. 15th Int’l Conf. on Theory and Appl. of Cryptol. and Inf. Security (ASIACRYPT), 2009, pp. 319-333.

[21] Y. Zhu, H. Hu, G.-J. Ahn, and M. Yu, ‘‘Cooperative Provable Data Possession for Integrity Verification in Multi-Cloud Storage,’’ IEEE Trans. Parallel Distrib. Syst., vol. 23, no. 12, pp. 2231-2244, Dec. 2012.

[22] R. Curtmola, O. Khan, R.C. Burns, and G. Ateniese, ‘‘MR-PDP: Multiple-Replica Provable Data Possession,’’ in Proc. 28th IEEE Conf. on Distrib. Comput. Syst. (ICDCS), 2008, pp. 411-420.