Volume 4, Issue 10, October 2014 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

Privacy-Preserving Public Verifying and Data Dynamics for Secure Cloud Storage Using TPV

Mrs. Vasanthi. R
AP/CSE &Idhaya Engineering College for Women, India

Ms. Seethalakshmi. P
Idhaya Engineering College for Women, India

Prof. Prakash. R
Vivekananda Engineering College for Women, India

Abstract*— Cloud computing is mainly used for storing client information in the server as the cloud storage without overhead of storage and maintenance, which is very helpful for clients because it is pay per use process. Here security is main problem for both client and server. In This paper we describe secure cloud storage with privacy preserving and data dynamics by using Third Party Verifier (TPV). The TPV is used for providing security for both client and server but the third party verifier does not know the data processed by the client and it does not have local copy of content in server. By this it provides privacy for client's data. Our approach provides high performance and security in data dynamics.

***Keywords*— Cloud storage, Data dynamics, Privacy preserving, TPV,Key generation**

I. INTRODUCTION

Cloud computing is the one of the important concept of Information Technology. The cloud computing provides lot of benefits to IT as they are pay per use, reduce the infrastructure technology, stream line process, improve accessibility and improve flexibility. The main aspect of cloud computing is the data has centralized or outsourced to the cloud. From the user's point of view including both individuals and IT environment, they can store the remote data on cloud that has the advantage like without overhead of storage maintenance, global data access with not dependent of geographical locations.

The cloud computing provide these benefits. In this cloud storage is consists of the cloud service provider(CSP) as like as cloud server or server for providing services for the client when they require and also separate the administrative entities from the outsource data. The outsource data is actually user's ultimate control over their data. The cloud storing data integrity is having risk due to the following reasons.

First of all, cloud infrastructure is more powerful and reliable compare than other personal computing devices, now days having both internal and external threats for data integrity. Secondly, there do various motivations for CSP to behave undependably towards the cloud users relating to the standing of their outsourced information. That is the cloud having outsourced data is economically attractive for long term large scale data storage, it doesn't instantly supply any guarantee on data integrity and accessibility. As users not physically possess the storage of their data, cryptographic primitives for the data security protection not directly adopted [11].

Simply downloading the data for its integrity verification is very difficult solution due to expressiveness in I/O and transmission cost over the network. Besides, it's typically insufficient to find the data corruption only if accessing the data, because it doesn't provide users correctness assurance for those un-accessed data and could be too late to recover the data loss or harm. Considering the large size of the outsourced data and also the user's constrained capability of the resource, the tasks of auditing the data correctness in a cloud environment may be formidable and costly for the cloud users [10], [12]. The overhead of using cloud storage could be minimized as possible, such that user not need to perform too many operations for use of data (in additional to retrieving the data). There is also more than one user accesses identical cloud storage, say in an enterprise setting. For that easier management, it has desirable that the cloud server only entertains request of verification from a single designated party.

To ensure the data integrity and save computation resource of cloud users as well as online burden, that is critical importance of enable public verifying and data dynamics as service for cloud data storage, so the user of cloud storage resort independent of third party verifier(TPV) to verify the outsourced data when that outsourced data will be need. The third party verifier (TPV) who has expertise and capabilities that users don't, can periodically checking the integrity of all the data stored within the cloud that data stored by the users, it provides method, more easier and reasonable way for the users to ensure their storage correctness within the cloud. In addition to help users to evaluate the risk of their cloud data service, the verify result from TPV would also be beneficial for the cloud service providers to improve their cloud based service platform, and serve for independent arbitration purposes [9]. Enabling public verifying services will play an important role for this nascent cloud economy to become totally established, wherever users will need ways to assess risk and gain trust in the cloud.

Recently, the notion of public verifiability has been proposed in the context of ensuring remotely stored data integrity under different system and security models [8], [10], [11], [13]. The Public verifiability allows an external party, in addition to the user also to verify the correctness of remotely stored data. However, no of these schemes [8], [10], [13] don't consider the privacy protection of user's data against external verifiers. They may potentially reveal user information to the auditors, this severe drawback greatly affects the security of these protocols in cloud computing. To overcome this drawback of auditors we introduce the concept of verifiers called Third Party Verifiers (TPV). The perspective of protecting privacy of data, the users, who own the data and rely on TPV just for the storage security of their data, do not want this verifying process new vulnerabilities of unauthorized information leakage towards their data security [14]. Moreover, there are legal regulations, like the USA health insurance portability and accountability Act (HIPAA) [15], further demanding the outsourced data to not be leaked to external parties [9]. Exploiting encryption before outsourcing [11] is a method to mitigate this privacy concern; however it's only complementary to the privacy preserving public auditing scheme to be proposed in this paper. While not a properly designed auditing protocol, encoding itself cannot prevent data from "the flow away" towards external parties during the verifying process. Thus, it doesn't fully solve the problem of protective data privacy but simply reduces it to the key management. Unauthorized leakage of data still remains a problem because of the potential exposure of decryption keys.

Therefore, the way to enable a privacy-preserving third-party verifying protocol, independent to data encryption, is that the drawback we are aiming to tackle in this paper. Our work is among the primary few ones to support privacy-preserving public verifying in Cloud Computing, with attention on data storage. Besides, with the prevalence of Cloud Computing, a foreseeable increase of verifying tasks from totally different users could also be delegated to TPV. because the individual verifying of these growing tasks may be tedious and cumbersome, a natural demand is then how to enable the TPV to with efficiency perform multiple verifying tasks in a batch manner, i.e., simultaneously

To address these issues, our work utilizes the technique of public key primarily based homomorphic linear authenticator (or HLA for short) [8], [10], [13], which enables TPV to perform the verifying without demanding the local copy of data and so drastically reduces the communication and overhead of computation as compared to the easy data auditing approaches. By integration the HLA with random masking, our protocol guarantees that the TPV could not learn any information about the data content stored within the cloud server throughout the efficient verifying process. The aggregation and algebraical properties of the authenticator additional profit our design for the batch auditing. Specifically, our contribution will be summarized because the following 3 aspects:

1) We have a tendency to inspire the public auditing system of data storage security in Cloud Computing and supply a privacy-preserving verifying protocol, i.e., our scheme allows an external verifier to verify user's outsourced data within the cloud without learning the data content.

2) To the most effective of our knowledge, our scheme is that the first to support scalability and more efficient public verifying in the Cloud Computing. Specifically, our scheme achieves batch auditing wherever multiple delegated verifying tasks from totally different users will be performed simultaneously by the TPV.

3) We have a tendency to prove the protection and justify the performance of our proposed schemes through focused on experiments and comparisons with the state-of-the-art.

The rest of the paper is organized as followed as. Section II introduces the system and model, and our design goals. Then we offer the elaborate description of our scheme in Section III. Section IV provides the security analysis and performance analysis, that can be followed by Section V that overviews the related work. Finally, Section VI provides the concluding remark of the whole paper.

II. PROBLEM STATEMENT

A. The System and Threat Model

We contemplate a cloud data storage service involving three different entities, as illustrated in Fig. 1: the cloud user (U), who has great amount of data files to be stored within the cloud; the cloud server (CS), which is managed by the cloud service provider (CSP) to produce data storage service and has important space for storing and computation resources (we won't differentiate CS and CSP hereafter); the third party verifier (TPV), who has experience and capabilities that cloud users do not have and is trusty to assess the cloud storage service reliableness on behalf of the user upon request. Users believe the CS for cloud data storage and maintenance. They will additionally dynamically interact with the CS to access and update their hold on data for various application functions. to save the computation resource in addition because the on-line burden, cloud users may resort to TPV for ensuring the storage integrity of their outsourced data, whereas hoping to keep their data private from TPV.

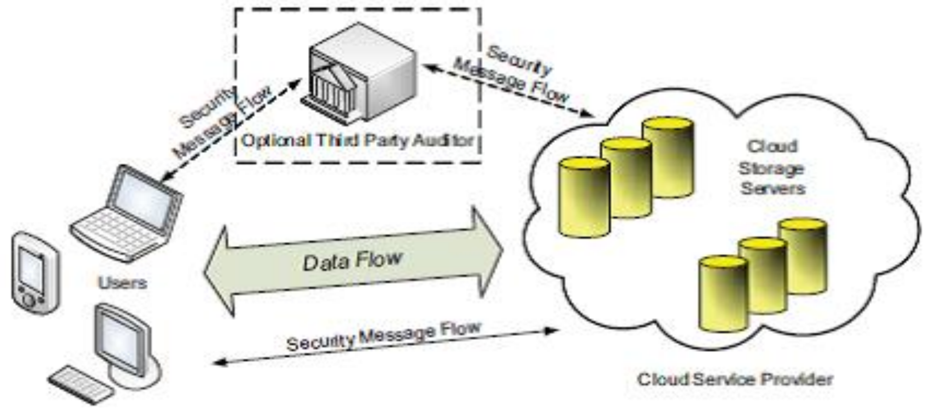

Fig. 1: The architecture of cloud data storage service with TPA

We think about the existence of a semi-trusted CS as [16] does. Namely, in most of your time it behaves properly and doesn't deviate from the prescribed protocol execution. However, for their advantages the CS may neglect to stay or deliberately delete rarely accessed data files that belong to standard cloud users. Moreover, the CS could plan to hide the data corruptions caused by server hacks or failures to keep up reputation. We have a tendency to assume the TPV, who is within the business of verifying, is reliable and independent, and so has no incentive to interact with either the CS or the users during the verifying process. However, it harms the user if the TPV may learn the outsourced data when that can be verify. To authorize the CS to reply to the audit delegated to TPV's, the user will sign a certificate granting verify rights to the TPV's public key, and every one verifies from the TPV are authenticated against such a certificate. These authentication handshakes are avoided in the following presentation.

B. Design Goals

To change privacy-preserving public verifying for cloud data storage below the aforesaid model, our protocol design should reach the subsequent security and performance guarantees.

1) Public verifiability: to permit TPV to verify the correctness of the cloud data on demand while not retrieving a replica of the complete data or introducing additional on-line burden to the users of cloud.

2) Storage correctness: to make sure that there exists no cheating cloud server that may pass the TPV's audit while not indeed storing users' data intact.

3) Privacy-preserving: to ensure that the TPV cannot derive users' data from the information of cloud collected throughout the verifying method.

4) Batch auditing: to enable TPV with secure and efficient verifying capability to address multiple verifying delegations from presumably large number of various users simultaneously.

5) Lightweight: to allow TPV to perform verifying with lowest communication and minimum computation overhead.An easy way to comply with the conference paper formatting requirements is to use this document as a template and simply type your text into it.

III. PROPOSED SCHEMES

This section presents our public verifying scheme which provides a whole outsourcing solution of data – not only the data itself, however additionally its integrity checking. Then we tend to present our main scheme and show a way to extent our main scheme to support batch auditing for the TPV upon delegations from multiple users. Finally, we tend to discuss a way to generalize our privacy-preserving public verifying scheme and its support of data dynamics.

We follow the similar definition of antecedently proposed schemes within the context of remote data integrity checking [8], [11], [13] and adapt the framework for our privacy-preserving public verifying system. Our framework assumes the TPV is the stateless, which can be a planned property achieved by our proposed solution is in Fig 2. It's straightforward to extend the framework above to capture a state full verifying system, basically by splitting the verification metadata into 2 elements that are stored by the TPV and therefore the cloud server respectively. Our design doesn't assume any additional property on the data file.

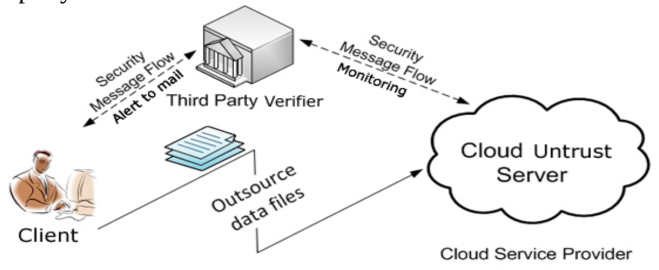

Fig. 2: The architecture of cloud data storage service with TPV

A. Service Provider / Third Party Verifier / Data Owner

Authentication:

If you are the new user going to consume the service then they have to register first by providing necessary details. After successful completion of sign up process, the user has to login into the application by providing username and exact password.

B. Service Provider

Resource Provisioning:

The process of providing resources to customers or clients with accounts, the appropriate access to those accounts, all the rights associated with those accounts, and all of the resources necessary to manage the accounts. When used in reference to a client, provisioning can be thought of as a form of customer service

C. Third Party Verifier

Verifying Process:

In public verifying module, the verifier perceives and recognizes the propositions before him for examination, collects evidence, evaluates the same and on this basis formulates his judgment which is communicated through his verify report.

D. Data Owner

I. *Key Maker:*

In this module, keys are generated according to the user setup in order to generate verification Meta data of uploaded file.

II. *Verifying Request:*

User can send the verifying request to external verifier along with the signatures and Meta data of the file. Then verifier will request for generated proof from service provider in order to do verifying process. Finally data owner will get the verify report.

E. Metadata key Generation:

Let the verifier V needs to the store the file F. Let this file F contains n file blocks. We tend to at the start preprocess the file and make metadata to be appended to the file. Let every data of the n data blocks have m bits in them. A typical file F that has the user needs to store within the cloud. Each of the Meta data from {the data, the info , the information} blocks mi is encrypted by using a appropriate algorithm to provide a new changed Meta data Mi while not loss of generality, we tend to show this method. The encoding technique will be temporary to produce still stronger protection for user's data. All the Meta data bit blocks that are generated exploitation the procedure are to be concatenated along. This concatenated Meta data should be appended to the file F before storing it at the cloud server. Then file F together with the appended Meta data with the cloud.

RSA ALGORITHM:

The RSA scheme is a block cipher scheme. Each plaintext block is an integer between 0 and $n - 1$ for some n, which leads to a block size $\leq \log2$ (n). The typical size for n is 1024 bits. The details of the RSA algorithm are as follows.

• *Key generation*

1) Pick two large prime numbers p and q, $p \neq q$;
2) Calculate $n = p \times q$;
3) Calculate $\varphi (n) = (p - 1) (q - 1)$;
4) Pick e, so that $\gcd (e, \varphi (n)) = 1$, $1 < e < \varphi (n)$;
5) Calculate d, so that $d*e \bmod \varphi (n) = 1$, i.e., d is the multiplication inverse of (e) in mod φ (n);
6) Get public key as KU = {e, n};
7) Get private key as KR = {d, n}.

• *Encryption*

For plaintext block $P < n$, its cipher text $C = P$ (e mod n).

• *Decryption*

For cipher text block C, its plaintext is $P = C$ (d mod n).

IV. Evaluations

A. Performance Analysis

We currently assess the performance of the projected privacy-preserving public verifying schemes to indicate that they're so light-weight. We are going to concentrate on the cost of the efficiency of the privacy-preserving protocol and our projected batch verifying technique. The experiment is conducted using C on Linux system with on Intel Core two processor running at one.86 GHz, 2048 MB of RAM, and a 7200 rpm Western Digital 250 GB Serial ATA drive with an 8 MB buffer. Our code is use the Pairing-Based Cryptography (PBC) library version 0.4.18. The elliptic curve used within the experiment could be a MNT curve, with base field size of 159 bits and therefore the embedding degree 6. The protection level is chosen to be 80 bit, which implies $|vi| = 80$ and $|p|=160$.Allexperimental results representing the mean of 20 trials.

1 Cost of Privacy-Preserving Protocol

We begin by estimating the price in terms of basic cryptographic operations, as notated as in Table 1. Suppose that there are c random blocks per the challenge message chal throughout the Verify section. Below this setting, we have a tendency to quantify the cost introduced of the privacy preserving verifying in terms of server computation, verifier computation furthermore as communication overhead. On that the server side, the generated response includes an aggregative authenticator $\sigma = \Pi_{i\in I} \sigma_i v_i \in G1$, a random factor $R = e (u, v) r \in GT$, and a blind linear combination of sampled blocks $\mu = \gamma\Sigma i \in I v_i m_i + r \in Zp$, wherever $= h(R) \in Zp$. The corresponding computation price is c-MultExp1G1 (|vi|), Exp1GT (|p|), and Hash1Zp + AddcZp + Multc+1Zp, respectively. Compared to the existing HLA primarily based solution for ensuring

Table 1. Notation of Cryptographic operations

Notation	**Cryptographic operations**
$Hash^{t}_{G1}$	Hash t values into the group G1.
$Mult^{t}_{G}$	t multiplications in group G.
$Exp^{t}_{G}(\ell)$	t exponentiations g^{a}_{i}, for g 2 G, $\|a_i\| = \ell$.
$m\text{-}MultExp^{t}_{G}(\ell)$	t m-term exponentiations $\pi^{m}_{i=1}\ g^{a}_{i}$.

$Pair^{t}_{G1,G2}$	t pairings $e(u_i, g_i)$, where u_i Є G1, g_i Є G2.
m-$MultPair^{t}_{G1,G2}$	t m-term pairings $\pi^{m}_{i=1}$ $e(u_i, g_i)$.

remote data integrity [13]1, the extra price for protecting the privacy of user, resulted from the random mask R, is only a constant: Exp1GT (|p|) + Mult1Zp + Hash Zp + Add1 Zp , that has nothing to try and do with the number of sampled blocks c. is set c is ready to be 300 to 460 for high assurance of verifying, as mentioned in Section 3.4, the additional price for privacy-preserving guarantee on the server side would be negligible against the whole server computation for response generation.

Similarly, on the verifier side, upon receiving the response, the corresponding computation price for response validation is Hash1Zp + c- MultExp1G1 (|vi|) + HashcG1 + Mult1G1 + Mult1GT + Exp3G1 (|p|) + Pair2 G1, G2, among that only Hash1Zp + Exp2G1 (|p|) + Mult1GT account for the extra constant computation price. For c = 460 or 300, and considering the comparatively expensive pairing operations, this additional price imposes very little overhead on the general price of response validation, and so is neglected

V. RELATED WORKS

Ateniese et al. [8] are the primary to contemplate public auditability in their outlined "provable data possession" (PDP) model for ensuring possession of data files on untrusted storages. Their scheme utilizes the RSA primarily based Homomorphic linear authenticators for auditing outsourced data and suggests randomly sampling many blocks of the file. However, the general public auditability in their theme demands the linear combination of sampled blocks exposed to external auditor. Once used directly, their protocol isn't demonstrably privacy preserving, and so might leak user data information to the auditor. Juels et al. [11] describe a "proof of retrievability" (PoR) model, wherever spot-checking and error-correcting codes are used to guarantee each "possession" and "retrievability" of data files on remote archive service systems. However, the number of audit challenges a user will perform is fixed a priori, and public auditability isn't supported in their main scheme. Though they describe a simple Merkle-tree construction for public PoR, this approach only works with encrypted data.

Dodis et al. [25] provides a study on completely different variants of PoR with private auditability. Shacham et al. [13] design an improved PoR scheme designed from BLS signatures [17] with full proofs of security within the security model outlined in [11]. Just like the development in [8], they use publically verifiable homomorphic linear authenticators that are designed from demonstrably secure BLS signatures. Supported the elegant BLS construction, a compact and public verifiable scheme is obtained. Again, their approach doesn't support privacy-preserving auditing for the same reason as [8]. Shah et al. [9], [14] propose permitting a TPA to stay on-line storage honest by first encrypting {the data/the info/the information} then sending a number of pre-computed symmetric-keyed hashes over the encrypted data to the auditor. The auditor verifies each the integrity of the data file and therefore the server's possession of a previously committed decoding key. This scheme only works for encrypted files and it suffers from the auditor statefulness and bounded usage, which can probably bring in on-line burden to users once the keyed hashes are spent. In different related work, Ateniese et al. [19] propose a partially dynamic version of the previous PDP scheme, using only symmetrical key cryptography however with a finite number of audits. In [20], Wang et al. think about a similar support for partial dynamic data storage during a distributed situation with extra feature of data error localization. During a subsequent work, Wang et al. [10] propose to combine BLS-based HLA with MHT to support each public auditability and full data dynamics. Virtually at the same time, Erway et al. [21] developed a skip lists primarily based scheme to enable demonstrable data possession with full dynamics support. However, the verification in these 2 protocols needs the linear combination of sampled blocks even as [8], [13], and so doesn't support privacy preserving auditing. whereas all the higher than schemes offer strategies for economical auditing and demonstrable assurance on the correctness of remotely hold on data, none of them meet all the wants for privacy preserving public auditing in cloud computing. A lot of significantly, none of those schemes consider batch auditing, which might greatly reduce the computation price on the TPA once coping with a large number of audit delegations.

VI. CONCLUSIONS

In this paper, we tend to propose a privacy-preserving public verifying system for data storage security in Cloud Computing. we tend to utilize the homomorphic linear appraiser and random masking to ensure that the TPV wouldn't learn any knowledge concerning the data content hold on the cloud server throughout the efficient verifying method, that not only eliminates the burden of cloud user from the tedious and possibly costly verifying task, however additionally alleviates the users' concern of their outsourced data leakage. Considering TPV might at the same time handle multiple verify sessions from completely different users for his or her outsourced information files, we tend to additional extend our privacy-preserving public verifying protocol into a multi-user setting, wherever the TPV will perform multiple verifying tasks during a batch manner for higher efficiency. Intensive analysis shows that our schemes are provably secure and extremely efficient.

APPENDIX A
ZERO KNOWLEDGE PUBLIC AUDITING

Here we present a public verifying scheme with provably zero knowledge leakage. The setup phase is similar to our main scheme presented in Section 3.4. The secret parameters are sk = (x, ssk) and the public parameters are pk = (spk, v, g, u, e (u, v), g1), where g1 Є G1 is an additional public group element. In the audit phase, upon receiving challenge chal = $\{(i, v_i)\}_{i \in I}$, the server chooses three random elements r_m, r_σ, $\rho \leftarrow Zp$, and calculates

$R = e(g1, g)^{r}_{\sigma} \cdot e(u, v)rm$ Є GT and = h(R) Є Zp. Let μ′ denote the linear combination of sampled blocks

$\mu' = \Sigma_i \in {}_I v_i m_i + r \in Zp$, and σ denote the aggregated authenticator $\sigma = \Pi_i \in_I \sigma_i{}^{v}{}_{i}$ Є G1. To make the auditing scheme with zero knowledge leakage, the server has to blind both μ′ and σ. specifically, the server computes: $\mu = r_m + \mu'$ mod p and $\sum = \sigma^* g^{\rho}{}_{1}$. It then sends {ς, μ, ∑, R} as the response proof of storage correctness to the TPA, where ς = rσ +γρ mod p. With the response from the server, the TPA runs VerifyProof to validate the response by first computing = h(R) and then checking the verification equation

$R \cdot e(\Sigma^{\gamma}, g) = e((\Pi^{sc}{}_{i=s1} H(Wi)^{vi})^{\gamma} \cdot u^{\mu}, v) \cdot e(g1, g)^{\varsigma}$ (5)

The correctness of the above verification equation can be elaborated as follows:

$R \cdot e(\Sigma^{\gamma}, g) = e(g1, g)^{r}_{\sigma} \cdot e(u, v)^{r}_{m} \cdot e((\sigma \cdot g^{\rho}{}_{1}), g)$

$= e(g1, g)^{r}_{\sigma} \cdot e(u, v)^{r}_{m} \cdot e((\sigma^{\gamma}, g) \cdot e(g^{\rho\gamma}{}_{1}, g)$

$= e(u, v)^{r}_{m} \cdot e((\sigma^{\gamma}, g) \cdot e(g1, g)^{r\sigma+\rho\gamma}$

$= e((\Pi^{sc}{}_{i=s1} H(Wi)^{vi})^{\gamma} \cdot u^{\mu}, v) \cdot e(g1, g)^{\varsigma}$

The last equality follows from the elaboration of

Equation 1. ***Theorem:*** The above auditing protocol achieves Zero-knowledge information leakage to the TPV and it also ensures the storage correctness guarantee.

Proof: Zero-knowledge is easy to see. Randomly pick γ, μ, ς from Zp and Σ from G1, set $R \leftarrow e((\Pi^{sc}{}_{i=s1} H(Wi)^{vi})^{\gamma} * u^{\mu}, v) * e(g1, g)^{\varsigma} / e(\Sigma^{\gamma}, g)$ and back patch = h(R). For proof of storage correctness, we can extract ρ similar to the extraction of μ′ as in the proof of Theorem 1. With ρ,σ can be recovered from Σ. To conclude, a valid pair of σ and μ′ can be extracted.

REFERENCES

[1] P. Mell and T. Grance, "Draft NIST working definition of cloud computing," Referenced on June. 3rd, 2009 online at http://csrc.nist.gov/groups/SNS/cloudcomputing/index.html ,2009

[2] M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. H. Katz, A. Konwinski, G. Lee, D. A. Patterson, A. Rabkin, I. Stoica, and M. Zaharia, "Above the clouds: A Berkeley view of cloud computing," University of California, Berkeley, Tech. Rep.

[3] M. Arrington, "Gmail disaster: Reports of mass email deletions," Online at http://www.techcrunch.com/2006/12/28/gmaildisasterreports-of-mass-email-deletions/ , December 2006.

[4] J. Kincaid, "MediaMax/The Linkup Closes Its Doors," Online at http://www.techcrunch.com/2008/07/10/mediamaxthelinkp-closes-its-doors/ , July 2008.

[5] Amazon.com, "Amazon s3 availability event: July 20, 2008,"Online at http://status.aws.amazon.com/s320080720.html, 2008.

[6] S. Wilson, "Appengine outage," Online at http://www.cioweblog.com/50226711/appengineoutage.php , June 2008.

[7] B. Krebs, "Payment Processor Breach May Be Largest Ever," Online at http://voices.washingtonpost.com/securityfix/2009/01/payment_processor_breach_ma_ b.html , Jan. 2009.

[8] G. Ateniese, R. Burns, R. Curtmola, J. Herring, L. Kissner, Z. Peterson, and D. Song, "Provable data possession at untrusted stores," in Proc. of CCS'07, pp. 598–609, Alexandria, VA, October 2007.

[9] M. A. Shah, R. Swaminathan, and M. Baker, "Privacy preserving audit and extraction of digital contents," Cryptology ePrint Archive, Report 2008/186, 2008.

[10] 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, volume 5789 of LNCS. Springer-Verlag, pp. 355–370, Sep. 2009.

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

[12] Cloud Security Alliance, "Security guidance for critical areas of focus in cloud computing," 2009, http://www.cloudsecurityalliance.org .

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

[14] M. A. Shah, M. Baker, J. C. Mogul, and R. Swaminathan, "Auditing to keep online storage services honest," in Proc. Of HotOS'07.Berkeley,CA, USA: USENIX Association, ,pp 1-6, 2007.

[15] 104th United States Congress, "Health Insurance Portability and Accountability Act of 1996 (HIPPA)," Online at http://www.aspe.hhs.gov/admnsimp/pl104191.html, 1996.

[16] S. Yu, C. Wang, K. Ren, and W. Lou, "Achieving secure. scalable, and fine-grained access control in cloud computing," in Proc. of IEEE INFOCOM'10, San Diego, CA, USA, March 2010.

[17] D. Boneh, B. Lynn, and H. Shacham, "Short signatures from the Weil pairing," J. Cryptology, vol. 17, no. 4, pp. 297–319, 2004.

[18] A. L. Ferrara, M. Greeny, S. Hohenberger, and M. Pedersen, "Practical short signature batch verification," in Proceedings of CT-RSA, volume 5473 of LNCS. , pp. 309– 324,Springer-Verlag, 2009.

[19] G. Ateniese, R. D. Pietro, L. V. Mancini, and G. Tsudik, "Scalable and efficient provable data possession," in Proc. Of SecureComm'08, , pp. 1–10, 2008.

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

[21] C. Erway, A. Kupcu, C. Papamanthou, and R. Tamassia, "Dynamic provable data possession," in Proc. of CCS'09, pp. 213–222,2009.

[22] R. C.Merkle, "Protocols for public key cryptosystems," in Proc. of IEEE Symposium on Security and Privacy, Los Alamitos, CA, USA, 1980.

[23] G. Ateniese, S. Kamara, and J. Katz, "Proofs of storage from homomorphic identification protocols," in ASIACRYPT, pp. 319–333,2009.

[24] M. Bellare and G. Neven, "Multi-signatures in the plain public key model and a general forking lemma," in ACM Conference on Computer and Communications Security , pp. 390–399,2006.

[25] Y. Dodis, S. P. Vadhan, and D. Wichs, "Proofs of retrievability via hardness amplification," in TCC, 2009, pp. 109–127, Dec 2008.