PRIVACY PERSERVING AND ENHANCED DYNAMIC AUDITING PROTOCOL IN CLOUD COMPUTING

AARTHI.T
PG Scholar(CSE)
Sri Ramakrishna Engineering College
Coimbatore
aarthitnila@gmail.com

Mrs.RATHI.G
Assistant professor UG(CSE)
Sri Ramakrishna Engineering College
Coimbatore
rathig@srec.ac.in

ABSTRACT

In cloud computing, data owners host their data on cloud data servers and users can access the data from cloud servers. Due to the data outsourcing, however, this new paradigm of the data hosting service also introduces new security challenges for the data, which requires an independent data auditing service to check the data integrity in the cloud. In existing system is desired to convince data owners that the data are correctly stored in the cloud. In that system there is no security is provided when outsourcing data from TPA to server and does not concern is how to ensure the integrity of the outsourced data. So our proposed system is mainly concentrated on provide security between TPA to server and also user extend our system to check the integrity of failure in that file using Erasure Code technique. The original auditing protocol is vulnerable to the attack from an active adversary since it does not provide authentication of the response, so the user suggest employing a secure digital signature scheme to prevent the proof from being modified. The system present a novel family of erasure codes that are efficiently repairable and offer higher reliability. The proposed design allows users to audit the cloud storage with very lightweight communication and computation cost. The data auditing result not only ensures strong cloud data storage correctness , but also simultaneously achieves fast data error localization, that is the identification of misbehaving server.

Keywords **–** user, cloud server, third party auditor, public auditing

1. INTRODUCTION

Cloud computing is the delivery information as a service, which shares data resources, software, and data information that are provided to computers as a metered service over a network .

Cloud computing provides data access and data storage resources without requiring cloud users.End users access cloud based applications through a web browser or a light weight desktop or mobile app while the data are stored on servers at a remote location. Cloud application providers [1], [2], [3] strive to provide better service and performance on end-user computers.

A. Characteristics

Cloud computing exhibits the following key characteristics:

Empowerment of end-users of computing resources by putting the provisioning of those resources in their own control, as opposed to the control of a centralized IT service Agility improves with users' ability to re-provision technological infrastructure resources.

Application programming interface (API) accessibility to software that enables machines to interact with cloud software in the same way the user interface facilitates interaction between humans and computers. Cloud computing systems typically use REST-based APIs.

Cost is claimed to be reduced and in a public cloud delivery model capital expenditure is converted to operational expenditure. This is purported to lower barriers to entry, as infrastructure is typically provided by a third-party and does not need to be purchased for one-time or infrequent intensive computing tasks. Pricing on a utility computing basis is fine-

grained with usage-based options and fewer IT skills are required for implementation (in-house).

Device and location independence enable users to access systems using a web browser regardless of their location or what device they are using (e.g., PC, mobile phone). As infrastructure is off-site (typically provided by a third-party) and accessed via the Internet, users can connect from anywhere.

Virtualization technology allows servers and storage devices to be shared and utilization be increased. Applications can be easily migrated from one physical server to another.

Multi-tenancy enables sharing of resources and costs across a large pool of users thus allowing for: Centralization of infrastructure in locations with lower costs (such as real estate, electricity, etc.),Peak-load capacity increases (users need not engineer for highest possible load-levels),Utilization and efficiency improvements for systems that are often only 10–20% utilized.

Reliability is improved if multiple redundant sites are used, which makes well-designed cloud computing suitable for business continuity and disaster recovery.

Scalability and Elasticity via dynamic ("on-demand") provisioning of resources on a fine-grained, self-service basis near real-time, without users having to engineer for peak loads.

Performance is monitored and consistent and loosely coupled architectures are constructed using web services as the system interface.

Security could improve due to centralization of data, increased security-focused resources, etc., but concerns can persist about loss of control over certain sensitive data, and the lack of security for stored kernels. Security is often as good as or better than other traditional systems, in part because providers [3], [4] are able to devote resources to solving security issues that many customers cannot afford. However, the complexity of security is greatly increased when data is distributed over a wider area or greater number of devices and in multi-tenant systems that are being shared by unrelated users. In addition, user access to security audit logs may be difficult or impossible. Private cloud installations are in part motivated by users' desire to retain control over the infrastructure and avoid losing control of information security.

Maintenance of cloud computing applications is easier, because they do not need to be installed on each user's computer and can be accessed from different places.

In order to achieve the assurances of cloud data integrity and availability and enforce the quality of cloud storage service, efficient methods that enable on-demand data correctness verification on behalf of cloud users have to be designed. However, while providing efficient cross server storage verification and data availability insurance, these schemes are all focusing on static or archival data. In this work, the system propose an effective and flexible distributed storage verification scheme with explicit dynamic data support to ensure the correctness and availability of users' data in the cloud.

2.EXISTING SYSTEM

Previous work presents an efficient and secure dynamic auditing protocol [5], [6], [7] that can meet the above listed requirements. To solve the data privacy problem, the method is to generate an encrypted proof with the challenge stamp by using the Bilinearity property of the bilinear pairing, such that the third party auditor cannot decrypt it but can verify the correctness of proof. Without using mask technique, this method does not require any trusted organizer during the batch auditing for multiple clouds. On the other hand, in this method, user let the server [8], [9], [10] compute the proof as an intermediate value of the verification, such that the third party auditor can directly use this intermediate value to verify the correctness of the proof. Therefore, this method can

greatly reduce the computing loads of the auditor by moving it to the cloud server.

The original contributions can be summarized as follows:

1. user design an auditing framework for cloud storage systems and propose a privacy-preserving and efficient storage auditing protocol. The auditing protocol ensures the data privacy by using cryptography method and the Bilinearity property of the bilinear pairing, instead of using mask technique. The auditing protocol incurs less communication cost between the auditor and server. It also reduces the computing loads of the auditor by moving it to the server.
2. The system extend the auditing protocol to support the data dynamic operations, which is efficient and provably secure in the random oracle model.
3. The system further extend our auditing protocol to support batch auditing for not only multiple clouds but also multiple owners. The multicloud batch auditing does not require any additional trusted organizer. The multiowner batch auditing protocol can greatly improve the third party auditing performances, especially in large-scale cloud storage systems.

3.PROPOSED SYSTEM

Firstly, an adversary cannot choose another valid and uncorrupted pair of data block and data tag (*mk, tk*) to replace a challenged pair of data block and data tag (*mi, ti*), when it has already discarded *mi* or *ti*. Secondly, an adversary cannot forge the data tag for a data block to deceive the auditor. Finally, an adversary cannot generate a valid proof from the previous proofs or other information, without retrieving the outsourced data. The original auditing protocol is vulnerable to the attack from an active adversary since it does not provide authentication of the response, so user suggest employing a secure digital signature scheme to prevent the proof from being modified. It is easy to verify that the fixed protocol still preserves the properties of the original protocol such as dynamic auditing and batch auditing. For the performance of the fixed protocol, it is slightly heavier in computation and communication than the original protocol, since the server needs to compute a signature σ and forward it to the auditor additionally, and the auditor will perform extra signature verification.

Fully homomorphic encryption has numerous applications. For example, the system enables private queries to a search engine - the user submits an encrypted query and the search engine computes a succinct encrypted answer without ever looking at the query in the clear. It enables searching on encrypted data a user stores encrypted files on a remote file server and then can have the server retrieve only files that (when decrypted) satisfy some boolean constraint, even though the server cannot decrypt the files on its own.

The system present a novel family of erasure codes that are efficiently repairable and offer higher reliability compared to Reed-Solomon codes. It show analytically that our codes are optimal on a recently identified tradeoff between locality and minimum distance. The system introduce a new family of erasure codes called Locally Repairable Codes (LRCs) that are efficiently repairable both in terms of network bandwidth and disk I/O. The system analytically show that our codes are information theoretically optimal in terms of their locality, that is the number of other blocks needed to repair single block failures. It present both randomized and explicit LRC constructions starting from generalized Reed-Solomon parities. It also design and implement DFS-Xorbas, a module that replaces Reed-Solomon codes with LRCs in DFS.

4.ARCHITECTURE DIAGRAM

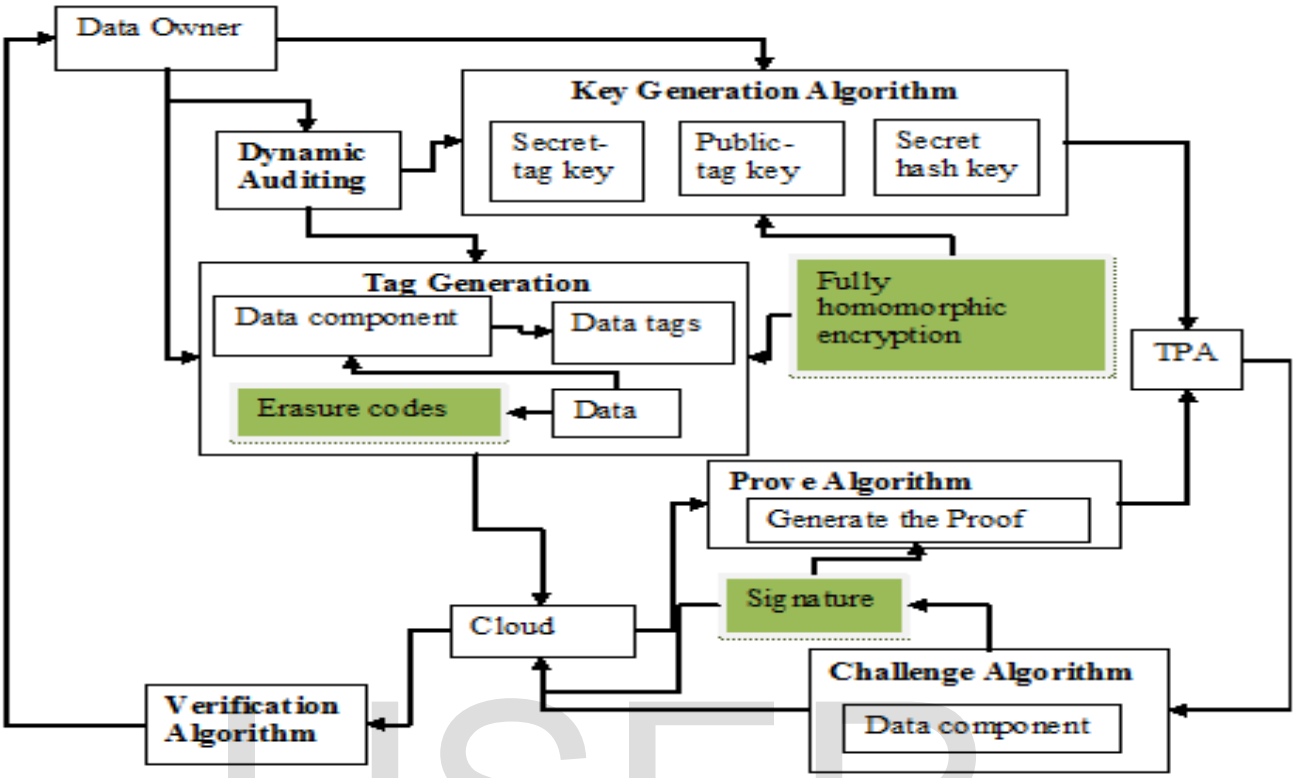

Fig.1.architechture diagram

5.SYSTEM IMPLEMENTATION

A. Create Cloud Setup

Initially the basic network model for the cloud data storage is developed in this module using java. Three different network entities can be identified as follows: User: an entity, who has data to be stored in the cloud and relies on the cloud for data storage and computation, can be either enterprise or individual customers. Cloud Server (CS): an entity, which is managed by cloud service provider (CSP) to provide data storage service and has significant storage space and computation resources Third-Party Auditor: an optional TPA, who has expertise and capabilities that users may not have, is trusted to assess and expose risk of cloud storage services on behalf of the users upon request.

B. Initialization Process

The storage auditing protocol consists of three phases: owner initialization, confirmation auditing, and sampling auditing.

Owner Initialization: The owner runs the key generation algorithm KeyGen to generate the secret hash key sk_h, the pair of secret-public tag key (sk_t, pk_t). Then, it runs the tag generation algorithm TagGen to compute data tags. After all data tags are generated, the owner sends each data component $M = \{m_i\}_{i\in[1,n]}$ and its corresponding data tags $T = \{t_i\}_{i\in[1,n]}$ to the server together with

the set of parameters $\{u_j\}_{j\in[1,s]}$. The owner then sends the public tag key pkt, the secret hash key sk_h, and the abstract information of the data M_{info} to the auditor, which includes the data identifier FID, the total number of data blocks n.

Confirmation auditing In the auditing construction, the auditing protocol only involves two-way communication:

Challenge and Proof. During the confirmation auditing phase, the owner requires the auditor to check whether the owner's data are correctly stored on the server. The auditor conducts the confirmation auditing phase as

1. The auditor runs the challenge algorithm Chall to generate the challenge C for all the data blocks in the data component and sends the $\mathcal{C} = (\{i, v_i\}_{i\in Q}, R)$ to the server.

2. Upon receiving the challenge C from the third party auditor, server runs the prove algorithm Prove to generate the proof P = (TP,DP) and sends it back to the auditor.

3. When the auditor receives the proof P from server, it runs the verification algorithm to check the correctness of P and extract the auditing result.

The auditor then sends the auditing result to the owner. If the result is true, then the owner is convinced that its data are correctly stored on server, and it may choose to delete the local version of the data.

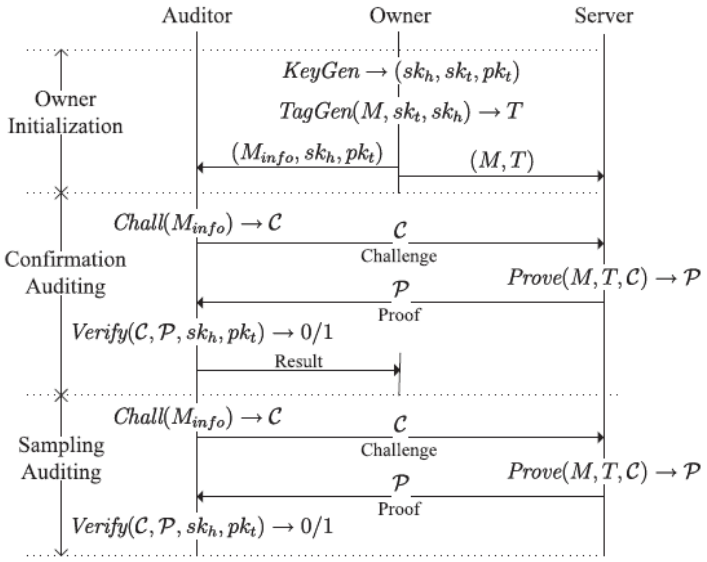

Fig.2.frame work of privacy preserving protocol

Sampling auditing: The auditor will carry out the sampling auditing periodically by challenging a sample set of data blocks. The frequency of taking auditing operation depends on the service agreement between the data owner and the auditor (and also depends on how much trust the data owner has over the server).

C. Dynamic operation for cloud user

Which contains 3 steps: data update, index update, and update confirmation.

Step 1: Data update. There are three types of data update operations that can be used by the owner: data modification, data insertion, and deletion. For each data update operation, there is a corresponding auditing algorithm in the dynamic auditing to process the operation and facilitate the future auditing.

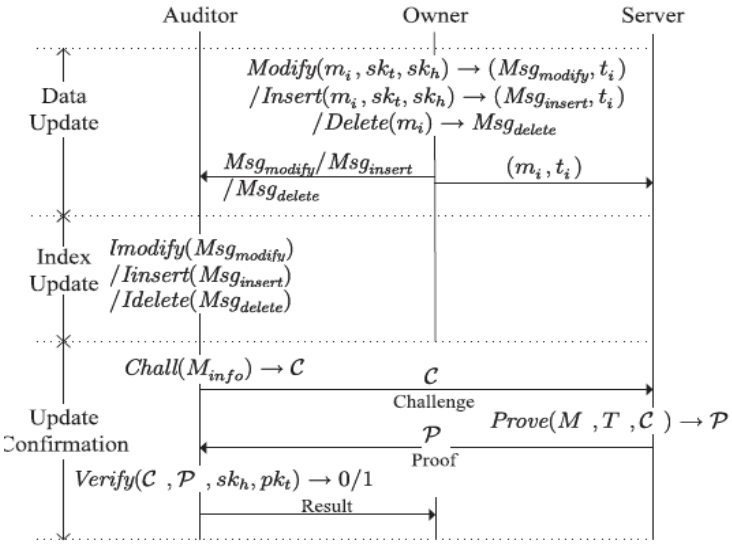

Fig.3.framework of auditing for dynamic operation

Step 2: Index update. Upon receiving the three types of data update messages, the auditor calls three corresponding auditing algorithms to update the ITable.

Step 3: Update confirmation. After the auditor updates the data ITable, it conducts a confirmation auditing for the updated data and sends the result to the data owner. Then, the owner can choose to delete the local version of data according to the update confirmation auditing result.

D. Initialization Process for Batch Auditing for Multiowner and Multicloud

Let O be the set of owners and S be the set of cloud servers. The data batch auditing for multiowner and data multicloud can be constructed as follows:

Owner initialization: Each owner O_k($k \in O$) runs the key generation algorithm KeyGen to generate the pair of secret-public tag key ($sk_{t,k}$, $pk_{t,k}$) and a set of secret hash key $\{sk_{h,kl}\}_{l \in S}$. That is, for different cloud servers, the owner has different secret hash keys. It denote each data component as M_{kl}, which means that this data component is owned by the owner O_k and stored on the cloud server S_l. Suppose the data component M_{kl} is divided into n_{kl} data blocks, and each data block is further split into s sectors. (Here, we assume that each data block is further split into the same number of sectors. The owner O_k runs the tag generation algorithm TagGen to generate the data tags $T_{kl} = \{t_{kl,i}\}_{i \in [1,n_{kl}]}$.

E. Batch auditing phase for multiowner and multicloud

O_{chal} and S_{chal} denote the involved set of owners and cloud servers involved in the batch auditing. The batch auditing also consists of three steps: batch challenge, batch proof, and batch verification.

Step 1: Batch challenge. During this step, the data auditor runs the batch challenge algorithm B_{Chall} to generate a batch challenge C for a set of challenged owners O_{chal} and a set of clouds S_{chal}.

Step 2: Batch proof. Upon receiving the challenge, each server S_l($l \in S_{chal}$) generates a proof $P_l = (TP_l, DP_l)$ by using the following batch prove algorithm BProve and sends the proof P_l to the auditor.

Step 3: Batch verification. Upon receiving all the proofs from the challenged data servers, the auditor runs the batch verification algorithm BVerify to check the correctness of the proofs.

F. Signature generation between TPA and Server

So user suggest employing a secure digital signature scheme to prevent the proof from being modified. Specifically, in KeyGen phase, the algorithm outputs additional two parameters (*skS*, *pkS*) as the cloud server's secret/public key pair. In the auditing process, before sending the proof *P* = (*TP*,*DP*) to the auditor, the server uses its secret key *skS* to generate a signature σ of *P* and sends (*TP*,*DP*,σ) as the response to the challenge. Upon receiving the response, the auditor firstly verifies the signature σ. If it is valid, the auditor performs the Verify phase of the original auditing protocol; Otherwise, discards the response.

G. Fully Homomorphic Encryption technique

Our ultimate goal is to construct a fully homomorphic encryption scheme ε. First, let us discuss what it means to be *fully homomorphic*.

At a high-level, the essence of fully homomorphic encryption is simple: given ciphertexts that encrypt $\pi_1, \ldots, \pi_t$, fully homomorphic encryption should allow anyone (not just the key-holder) to output a ciphertext that encrypts $f(\pi_1, \ldots, \pi_t)$ for any desired data function f, as long as that function can be more efficiently computed. No information about $\pi_1, \ldots, \pi_t$ or $f(\pi_1, \ldots, \pi_t)$, or any intermediate user plaintext values, should leak; the data inputs, output and intermediate values are always encrypted.

Formally, there are different ways of defining what it means for the final ciphertext to "encrypt" $f(\pi_1, \ldots, \pi_t)$. The minimal requirement is correctness of the data. A fully homomorphic data encryption scheme ε should have an efficient algorithm $\text{Evaluate}_\varepsilon$ that, for any valid ε key pair (sk, pk), any circuit C, and any ciphertexts $\psi_i \leftarrow \text{Encrypt}_\varepsilon(pk, \pi_i)$, outputs $\psi \leftarrow \text{Encrypt}_\varepsilon(pk, C, \psi_1, \ldots \psi_t)$ such that $\text{Decrypt}_\varepsilon(sk, \psi) = C(\pi_1, \ldots, \pi_t)$

H. Erasure code technique for identify misbehaving server

Error localization is a key prerequisite for eliminating errors in storage systems. It is also of critical importance to identify potential threats from external attacks. Our scheme outperforms those by integrating the correctness verification and error localization (misbehaving server identification) in our challenge-response protocol: the response values from servers for each challenge not only determine the correctness of the distributed storage, but also contain information to locate potential data error(s).

Once the inconsistency among the storage has been successfully detected, user can rely on the precomputed verification tokens to further determine where the potential data error(s) lies in.

DFS-Xorbas computes two extra parities for a total of 16 blocks per stripe as shown in Fig. 1. Similar to the calculation of the RS parities, Xorbas calculates all parity blocks in a distributed manner through TPA encoder jobs. All blocks are spread across the cloud according to configured block placement policy. The default policy randomly places blocks at cloud, avoiding collocating blocks of the same stripe. A special cloud is dispatched to attempt light-decoding: a single task opens parallel streams to the nodes containing the required blocks, downloads them, and performs a simple XOR. In the presence of multiple failures, the 5 required blocks may not be available. In such case the light-decoder fails and the heavy decoder is initiated. The heavy decoder operates in the same way as in Reed-Solomon: streams to all the blocks of the stripe are opened and decoding is equivalent to solving a system of linear equations. The RS linear system has a Vandermonde structure which allows small CPU utilization. The recovered block is finally sent and stored to a Datanode according to the cloud block placement policy.

Cloud starts a decoding process when corrupt files are detected. Xorbas uses two decoders: the light-decoder aimed at single block failures per stripe, and the heavy-decoderemployed when the light-decoder fails. When the BlockFixer detects a missing (or corrupted) block, it determines the 5 blocks required for the reconstruction according to the structure of the LRC.

6.EVALUATION

The result give the communication cost comparison, security comparison and computation complexity comparison between proposed scheme and existing works.

To study the impact of the component failure probability on the system reliability, it compare existing system under failure probability settings of 0.1 to 1 percent with a step value of 0.1 percent. With the increase of component failure probability from 0.1 to 1 percent, calculate the application failure probabilities of two approaches. From the results our proposed system has better than existing approach.

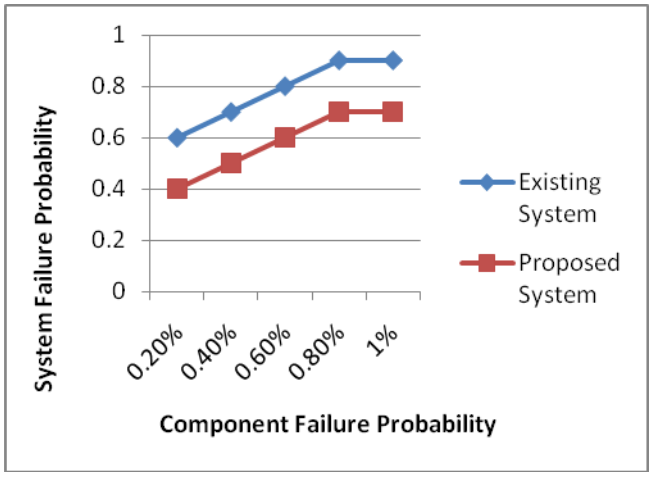

Fig. 4.Impact of component failure probability

7.CONCLUSION AND FUTURE WORK

The system investigate the problem of data security in cloud data storage, which is essentially a distributed data storage system. To achieve the assurances of cloud data integrity and availability and enforce the quality of dependable cloud storage service for users, the system propose an effective and flexible distributed scheme with explicit dynamic data support, including block update, delete, and append. It revisited the dynamic and privacy preserving auditing protocol for the cloud storage proposed and demonstrated that an active adversary can modify the auditing proof to fool the auditor and the owner that the remote cloud files are pristine, while the files have been corrupted. It rely on erasure-correcting code in the file distribution preparation to provide redundancy parity vectors and guarantee the data dependability. it introduced a new family of codes called Locally Repairable Codes (LRCs) that have marginally suboptimal storage. The proposed scheme achieves the integration of storage correctness insurance and data localization, that is, whenever data file corruption has been detected during the storage correctness verification across the distributed servers, it can almost guarantee the simultaneous identification of the misbehaving server(s). The proposed system improves the security and integrity for Dynamic Auditing Protocol.

REFERENCES

[1]CloudSecurityAlliance,"TopThreats to Cloud Computing," http://www.cloudsecurityalliance.org , 2010.

[2]M.Arrington,"GmailDisaster:Reports of Mass EmailDeletions," http://www.techcrunch.com/2006/2/28/gmail-disasterreports of mass email deletions/, 2006.

[3]J.Kincaid,"MediaMax/TheLinkupClosesItsDoors," http://www.techcrunch.c/2008/07/10/mediamaxthelinkup-closesits-doors/, July 2008.

[4]Amazon.com,"Amazons3AvailabilityEvent:July20,2008,"http://status.aws.amazon.com/s3200807 20.html, July 2008.

[5]G.Ateniese,R.Burns,R.Curtmola,J.Herring,L.Kissner,Z.Peterson,andD.Song, "Provable Data Possession at Untrusted Stores," Proc. 14th ACMConf.Computer and Comm. Security(CCS '07), pp. 598-609, 2007.

[6] Cloud Security Alliance, "Security Guidance for Critical Areas of Focus in Cloud Computing," http://www.cloudsecurityalliance.org, 2009.

[7] K.D.Bowers, A. Juels, and A. Oprea, "Proofs of Retrievability:Theory and Implementation," Proc. ACM Workshop Cloud Computing Security (CCSW '09), pp. 43-54, 2009.

[8] C. Wang, Q. Wang, K. Ren, and W. Lou, "Towards Secure andDependable Storage Services in Cloud Computing," IEEE Trans.Service Computing, vol. 5, no. 2, 220-232, Apr.-June 2012.

[9] R.C. Merkle, "Protocols for Public Key Cryptosystems," Proc. IEEE Symp. Security and Privacy, 1980.

[10] Y. Dodis, S.P. Vadhan, and D. Wichs, "Proofs of Retrievability via Hardness Amplification," Proc. Theory of Cryptography Conf. Theory of Cryptography (TCC), pp. 109-127, 2009.