GIAP

International Journal of Students' Research in Technology & Management
Vol 3 (03), March 2015, ISSN 2321-2543, pg 313-316

FORTIFYING SECURITY AND INTEGRITY IN CLOUD

Srinivasan S[1], Yokeswaran M[2], Srinivas PR[3]

Department of CSE in Sri Ventakeswara College of engineering, Chennai

[1]Srini.sudars@gmail.com
[2]yokeshmohan95@yahoo.com
[3]gupta.nivas@gmail.com

***ABSTRACT*—Data integrity and security are the main concerns in cloud..In this paper we propose an adequate and impressive scheme by using dynamic data support to ensure that the data in cloud is correct and errorless. The new scheme proposed in this paper supports modification, deletion and updating the data blocks present in the cloud server. The main problem is the efficiency and security to verify that the storage server is storing the client's outsourced data.**

The necessary security and efficiency has to be provided to the server which can be small computing device with limited resources.

Keywords:* *Outsourcing cloud storage, homomorphism token, cloud data.

I INTRODUCTION

Cloud storage is a networked storage model that represents the data of an enterprise or individual user data in which data is stored and maintained in virtualized pools of storage which are generally hosted by third parties. The cloud service providers operate on large volumes of data handled by data centres. The resource may be stored across multiple servers. File safety depends upon the hosting websites. The increasing network bandwidth and flexible network connections make it possible that users can subscribe high quality services from remote data centres. Early work on Cloud computing concentrated on data authentication and integrity, i.e., how to efficiently and securely ensure that the server returns correct and complete results in response to its clients' queries. Further the cloud service providers offer data integrity **by Provable Data Possession (PDP).** By this the service provider ensures that the client is not being cheated. In this context, cheating means that the server might delete some of the data or it might not store all data in fast storage recently, the importance of ensuring the remote data integrity has been highlighted by the following research works.

II. CLOUD ARCHITECTURE

At present the cloud storage technology involves virtualized infrastructure along with characteristics of agility, scalability, elasticity and multi-tenancy in handling the user data. Cloud storage services like Open Stack, cloud storage products like EMC Atmos and Hitachi Cloud Services, and distributed storage research projects like VISION Cloud are all examples of object storage and infer the following guidelines.

Cloud storage is:

- Made up of many distributed resources.
- Highly fault tolerant through redundancy and distribution of data
- Highly durable

The two most significant components of cloud computing architecture are known as the front end and the back end. The front end is the part seen by the client. The web browser acts an API for the user. The back end of the cloud computing architecture is the cloud itself, comprising servers and data storage devices managed by cloud service providers.

Representative network architecture for cloud data storage is illustrated in Figure1. Three different network entities form the cloud service: Users, who store data in the cloud and rely on the cloud for data computation. The user includes both individual consumers and organizations. A CSP has significant resources and expertise in managing distributed cloud storage servers, owns and operates live Cloud Computing system. The CSP is responsible for providing co-operation between user and the data retrieval. An optional TPA, is trustworthy and expertise in providing link between user and CSP and is capable to assess cloud storage services on behalf of the users upon request. Our scheme is based entirely on symmetric-key cryptography. The main idea is that, client uses a unique password and the hardware pre-computes a certain number of short possession verification tokens, each token covering some set of data blocks. The actual data is then handed over to SRV. Subsequently, when client wants to obtain a proof of data possession, it challenges SRV with a set of random-looking block indices. In turn, SRV must compute a short integrity check over the specified blocks (corresponding to the indices) and return it to client. For the proof to hold, the returned integrity check must match the corresponding value pre-computed on user's side network entities form the cloud service: Users, who store data in the cloud and rely on the cloud for data computation. The user includes both individual consumers and organizations. A CSP has significant resources and expertise in managing distributed cloud storage servers, owns and operates live Cloud Computing system. The

CSP is responsible for providing co-operation between user and the data retrieval. An optional TPA, is trustworthy and expertise in providing link between user and CSP and is capable to assess cloud storage services on behalf of the users upon request. Our scheme is based entirely on symmetric-key cryptography. The main idea is that, client uses a unique password and the hardware pre-computes a certain number of short possession verification tokens, each token covering some set of data blocks. The actual data is then handed over to SRV. Subsequently, when client wants to obtain a proof of data possession, it challenges SRV with a set of random-looking block indices. In turn, SRV must compute a short integrity check over the specified blocks (corresponding to the indices) and return it to client. For the proof to hold, the returned integrity check must match the corresponding value pre-computed on user's side.

However, the pre-computed tokens locally or outsourcing them – in encrypted form – to SRV

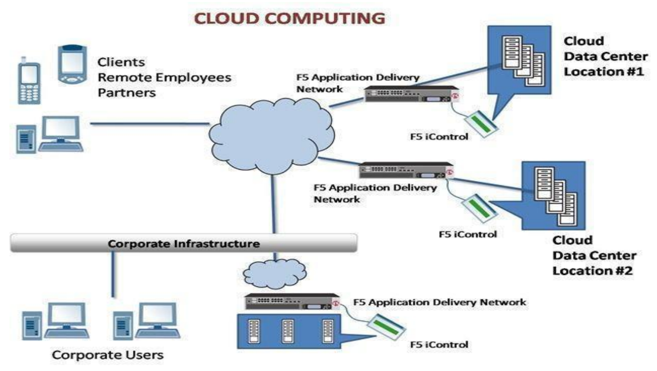

CLOUD COMPUTING
Clients
Remote Employees
Partners
F5 Application Delivery Network
Cloud Data Center Location #1
F5 iControl
F5 Application Delivery Network
F5 iControl
Cloud Data Center Location #2
Corporate Infrastructure
F5 Application Delivery Network
F5 iControl
Corporate Users

Figure 1

A. The Cloud Ecosystem

Virtual infrastructure management tools for data centres have been around. These tools are fully functional with providing cloud-like Interfaces for security, Thus, an ecosystem of cloud tools is starting to from (Figure 2) where cloud toolkits attempt to span both cloud management and VI management. The focus of our work is, therefore, to produce a VI management solution with a flexible and open architecture that can be used to build private/hybrid clouds.

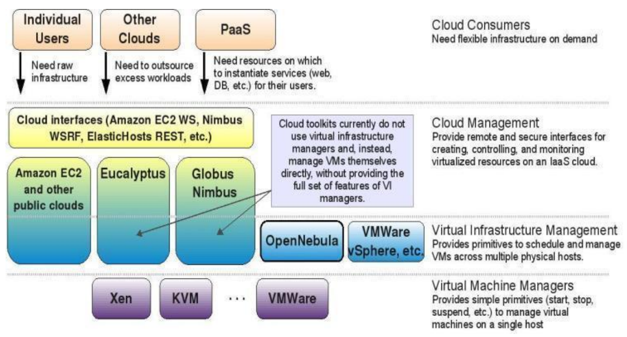

Individual Users
Other Clouds
PaaS
Need raw infrastructure
Need to outsource excess workloads
Need resources on which to instantiate services (web, DB, etc.) for their users.
Cloud Consumers
Need flexible infrastructure on demand
Cloud interfaces (Amazon EC2 WS, Nimbus WSRF, ElasticHosts REST, etc.)
Cloud toolkits currently do not use virtual infrastructure managers and, instead, manage VMs themselves directly, without providing the full set of features of VI managers.
Cloud Management
Provide remote and secure interfaces for creating, controlling, and monitoring virtualized resources on an IaaS cloud.
Amazon EC2 and other public clouds
Eucalyptus
Globus Nimbus
OpenNebula
VMWare vSphere, etc.
Virtual Infrastructure Management
Provides primitives to schedule and manage VMs across multiple physical hosts.
Xen
KVM
...
VMWare
Virtual Machine Managers
Provides simple primitives (start, stop, suspend, etc.) to manage virtual machines on a single host

Figure 2

B. Verification Framework for Multi-Cloud

DDR techniques are incapable of satisfying the requirements of multiple clouds in terms of communication and computation costs. To address this problem, we consider a multi-cloud storage service (Figure 1). In this architecture, a data storage service involves three different entities: Clients who have a large amount of data to be stored in multiple clouds , Cloud Service Providers (CSPs) who work together to provide data storage services and have enough storages and computation resources; and Trusted Third Party (TTP) who is trusted to store verification parameters. In this architecture, we consider the existence of multiple CSPs to maintain the clients data. The verification procedure is described as follows:

Firstly, a client uses the secret key to pre-process a file which consists of a collection of blocks, generates a set of public verification information that is stored in TTP, transmits the file and some verification tags to CSPs, and may delete its local copy; Then, by using a verification protocol, the clients can issue a challenge for one CSP to check the integrity and availability of outsourced data with respect to public information stored in TTP. We neither assume that CSP is trust to guarantee the security of the stored data. To achieve this goal, a TTP server is constructed as a core trust base on the cloud for the sake of security we assume the TTP is reliable and independent and having the following functions:

- *To setup and maintain the CDDR cryptosystem.*
- *To generate and store data owner's public key.*
- *To store the public parameters.*

C. Division of Data Blocks

*Let us consider a large database D***:** Let it be divided into d blocks.
Let't' be the number of times the storage server is being accessed: It is similar to the encryption key generation used in **AES, DES** and **BF**. This algorithm uses two main function namely, pseudo-random function (PRF) f, and a pseudo-random permutation g defined as follows,

$F=\{0,1\}^{C} X\{0,1\}^{K}$

So $f\text{->}\{0,1\}^{L}$

$g=\{0, 1\}^{l} X \{0, 1\}^{L} -! \{0, 1\}^{l}$

In our case l = log d, since we use g to permute indices. The output of 'f' is used to generate the key for g and c = log t.

In this case L = 128. We use the PRF f with two master secret keys W and Z, both of k bits. The key W is used to generate session permutation keys while Z is used to generate challenge nonce.

ALGORITHM:

The algorithm below describes the encrypted key generation from client side when user wants to access the data.

```
Begin
Choose parameters c,k,l and function f and g. w, x, k € {0, 1}^k for (i=0 to t) do
Begin Round i;
Generate $k_i$=fw (i) and $c_i$=fz
Compute
$V_i$=H ($C_i$ , D[$gk_{t(1)}$],.....,D[$gk_{t(1)}$]
$V_i'$ =$AE_k$(i,$v_i$);
End;
Send to SRV: (D, {[i, $v_i$'] for 1 < i < t})
End;
```

III. DESIGN METHODOLOGIES

The data design transforms the information domain model created during analysis into the data structures that will be required to implement the software. The data objects and relationships defined in the entity relationship diagram and the detailed data content depicted in the data dictionary provide the basis for the data design activity. More detailed data design occurs as each software component is designed. The architectural design defines the relationship between major structural elements of the software, the design patterns that can be used to achieve the requirements the system architecture, the interface representation, and the component level detail. During each design activity, we apply basic concepts and principles that lead to high quality.

We aim to design efficient mechanisms for dynamic data verification and operation and achieve the following goals:

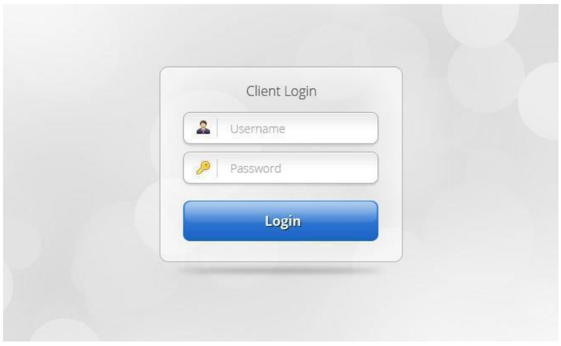
- *Storage correctness*
- *Fast localization of data error*
- *Dynamic data support*
- *Dependability*

IV. SECURITY ANALYSIS AND PERFORMANCE EVALUATION

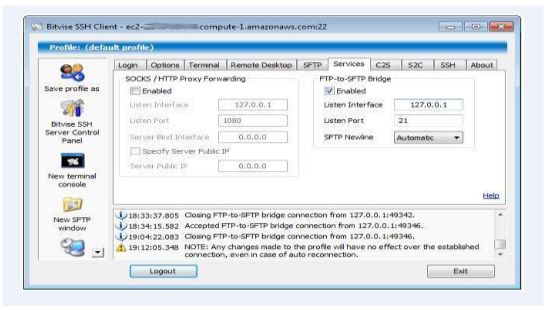
We evaluate the efficiency of this scheme via implementation of both file distribution preparation and verification token pre computation. ***The keys generated are unique and the duplication of keys is not possible.*** This algorithm paves way for the data possession only to the respective client. Thus the encrypted key generated will be true if it contains the data block segment. The service records provide a match key by which user can only access information. Even though along with the user, ***TTP has the security knowledge it can't access the user data because control access primarily depends on the user.*** The DNS in the service records only allow authenticated access and thus other hosts can't claim permission to user space even though multiple records of the users are maintained in the same server.

V. EXPERIMENTAL RESULTS

Client login



Cloud Authentication Server


Bitvise SSH Client - ec2-[redacted].compute-1.amazonaws.com:22

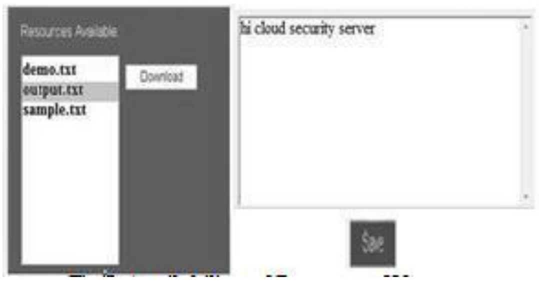
Profile: (default profile)

Login | Options | Terminal | Remote Desktop | SFTP | Services | C2S | S2C | SSH | About

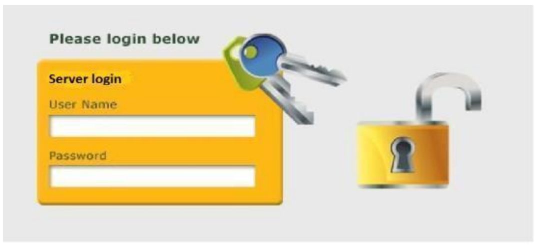
SOCKS / HTTP Proxy Forwarding
[ ] Enabled
Listen Interface: 127.0.0.1
Listen Port: 1080
Server Bind Interface: 0.0.0.0
[ ] Specify Server Public IP
Server Public IP: 0.0.0.0

FTP-to-SFTP Bridge
[✓] Enabled
Listen Interface: 127.0.0.1
Listen Port: 21
SFTP Newline: Automatic

```
18:33:37.805  Closing FTP-to-SFTP bridge connection from 127.0.0.1:49342.
18:34:15.582  Accepted FTP-to-SFTP bridge connection from 127.0.0.1:49346.
19:04:22.083  Closing FTP-to-SFTP bridge connection from 127.0.0.1:49346.
19:12:05.348  NOTE: Any changes made to the profile will have no effect over the established
              connection, even in case of auto reconnection.
```



Resource availability



Server login

VI. CONCLUSION

In this paper, we investigated the problem of data security in cloud data storage, which is essentially a distributed storage system. To ensure the correctness of users' data in cloud data storage, we proposed an effective and flexible distributed scheme with explicit dynamic data support, including block update, delete, and append. We rely on erasure-correcting code in the file distribution preparation to provide redundancy parity vectors and guarantee the data dependability. By utilizing the homomorphic token with distributed verification of erasure coded data, our scheme achieves the integration of storage correctness insurance and data error localization, Through detailed security and performance analysis, we show that our scheme is highly efficient and resilient to Byzantine failure, malicious data modification attack, and even server colluding attacks.

VII. ACKNOWLEDGMENT

Our wholehearted thanks to our guide Mrs. R.Anitha , Asst. Professor, Department of CSE, SVCE for the timely guidance and encouraging us to proceed with the idea.

VII. REFERENCES

[1] M. Lillibridge, S. Elnikety, A. Birrell, M. Burrows, and M.
[1] Isard, "A Cooperative Internet Backup Scheme," Proc. of the 2003 USENIX Annual Technical Conference (General Track), 29–41, 2003.
[2] K. D. Bowers, A. Juels, and A. Oprea, "HAIL: A High-Availability and Integrity Layer for Cloud Storage," Cryptology ePrint Archive, Report 2008/489, 2008, http://eprint.iacr.org/.
[3] L. Carter and M. Wegman, "Universal Hash Functions," Journal of Computer and System Sciences, vol. 18, no. 2, pp. 143–154, 1979.
[4] J. Hendricks, G. Ganger, and M. Reiter, "Verifying Distributed Erasure coded Data," Proc. 26th ACM Symposium on Principles of Distributed Computing, pp. 139–146, 2007.
[5] B. Sotomayor, R. S. Montero, I. M. Llorente, and I. T. Foster, "Virtual infrastructure management in private and hybrid clouds," IEEE Internet omputing, vol. 13, no. 5, pp.14–22, 2009.
[6] G. Ateniese, R. D. Pietro, L. V. Mancini, and G. Tsudik, "Scalable and efficient provable data possession," in Proceedings of the 4th international conference on Security and privacy in communication networks, Secure Comm., 2008, pp. 1–10.
[7] Rajkumar Buyya, Chee Shin Yeo, and Srikumar Venugopal "Market- Oriented Cloud Computing:Vision, Hype, and Reality for Delivering IT Services as Computing
[8] Utilities" The 10th IEEE International Conference on HighPerformance Computing and Communications , pp.10 Table 1
[9] Christopher Moretti, Jared Bulosan, Douglas Thain, and Patrick J.Flynn, "All-Pairs: An Abstraction for Data-Intensive CloudComputing", Parallel and Distributed Processing, 2008. IEEE International Symposium

About the authors:

__Srinivasan S.__ was born in Madurai on November 4, 1994. He is pursuing BE CSE in Sri Venkateswara college of Engineering, Chennai, India. Interested in cloud computing and database.

__Srinivas PR.__ was born in Banglore on November 9,1994. He is pursuing BE CSE in Sri Venkateswara college of Engineering, Chennai ,India. Interested in data analytics and business management.

__Yokeswaran M.__ was born in Cuddalore on July 10, 1995. He is pursuing BE CSE in Sri Venkateswara college of Engineering , Chennai, India. Interested in web development and software development.