DISTRIBUTED SCHEME TO AUTHENTICATE DATA STORAGE SECURITY IN CLOUD COMPUTING

ABSTRACT

Cloud Computing is the next-generation IT enterprise's revolution. Cloud computing displaces database and application software to large data centres, where service and data management may be unpredictable, whereas traditional IT services are subject to strict logical, physical, and personal controls. This aspect attribute, however contains a variety of security challenges. which have not been fully understood. It focuses on cloud data storage security, which has always been a critical aspect of quality of service (QOS). We designed and simulated an adaptable and efficient scheme to ensure the correctness of user data stored in the cloud, as well as some notable features, in this paper. For distributed erasure verification, a homomorphic token is used. We can identify misbehaving servers using this scheme. Despite previous efforts, our scheme allows for efficient and secure dynamic data block operations such as data insertion, deletion, and modification. Cloud computing in contrast to traditional solutions that keep IT services under strict physical, logical, and personnel controls, moves application software and databases to the cloud. Data management and services in large data centres may not be completely accurate. The proposed scheme is extremely flexible against malicious data modification, convoluted failures, and server clouding attacks, according to this effective security and performance analysis.

1. INTRODUCTION

Cloud Computing uses computing resources, such as hardware and software, to provide services over the internet. Users can access software and databases in the business model by using software as a service. Cloud providers are used to manage infrastructure and platform management. On-demand software is another term for software as a service. As a result, cloud providers charged on a pay-per-use basis. So this causes proponents to get the computing resources in Outsourcing manner. As a result, users hardware and software maintenance costs are being reduced. The introduction of Cloud Computing eliminates the need for direct hardware management and provides a great deal of convenience to users. The best examples for Cloud Computing initiators are Amazon Elastic Compute Cloud (EC2) and Amazon Simple Storage Service (S3). Because of the shift in computing platforms, the responsibility of local machines is being phased out in Cloud Computing. Here in the Cloud Computing, the data is integrated so that's why the users are showing kindness towards the cloud service providers.

Providing data security is a important factor that has always been an important aspect of service quality. For a variety of reasons, cloud computing is exhibiting new challenges. The most important aspect of cloud computing is that data should be under the user's control. But by using the traditional cryptographic techniques the users lose the control of data. As a result, traditional cryptographic primitives cannot be directly adopted. If we use traditional cryptographic primitives directly, we will have to conduct correctness checks on data stored in the cloud without having a complete understanding of the data. If we store different categories of data stored by different users, it becomes even more difficult to provide continuous assurance of data security. The data in the cloud is updated on a regular basis by the users. Insertion, deletion, modification, appending, reordering, and other data updates are all examples of data updation. During the dynamic data updation it is very important task that to assure the storage correctness of data. Because they all focus on single server situations and a large portion of them do not consider dynamic operations on data, the techniques that are used can be useful to give assurance to storage correctness without possessing user data. However, they cannot address all of the security threats of data storage in the cloud. Researchers have also proposed distributed protocols for ensuring storage correctness data across multiple servers or peers as a balancing approach. Again, no distributed scheme is aware of operations on dynamic data in these cases. As a result, their use in cloud data storage may be limited.

In this paper, we present a viable and flexible distributed scheme that explicitly verifies the user's data correctness in the cloud using dynamic data support. To provide redundancies and ensure data dependability, we use erasure correcting code in the file distribution preparation. When compared to traditional replication-based file distribution techniques, this development drastically reduces communication and storage overhead. This scheme achieves storage correctness insurance as well as data error localization by combining homomorphic tokens with distributed verification of erasure coded data: whenever data corruption is identified during storage correctness verification, this scheme can concurrently localize the data errors, i.e., the identification of the misbehaving server (s).

- Unlike many of its predecessors, which only provide binary results about the storage state across distributed servers, our challenge-response protocol further allows for data error localization.
- Unlike previous efforts to make sure data integrity remotely, the new scheme supports efficient and secure dynamic operations on data blocks, such as update, delete, and append.

- The proposed scheme is extremely efficient and rigid against Byzantine failure, server collusion attacks, and even malicious data modification attacks, and according to highly efficient security and performance analysis.

1.1 Cloud Data Storage Architecture

Figure 1 shows the network architecture for cloud data storage. As shown below, the architecture is divided into three different network modules:

Users: Users are divided into different groups. There are both organizations and individual users among all the users. The data is available for users to store in the cloud. The users must have the trust on the cloud computing.

Cloud Service Provider (CSP): A cloud service provider has the knowledge and important resources in building and managing distributed cloud storage servers. A live Cloud Computing system is identified and operated by a cloud service provider.

Third Party Auditor (TPA): An optional TPA, who has expertise and capabilities that users may not have, ills trusted to assess and expose risk of cloud storage services on behalf of the users upon request. A user stores his data in the cloud through a CSP into a set of cloud servers that are running in a parallel, cooperative, and distributed manner. Data redundancy is important. As the importance and size of the user's data grows, redundancy can be used with the erasure-correcting code technique to tolerate server failures or faults. The user then interacts with cloud servers via Cloud Service Provider (CSP) to retrieve or access his data for the purposes of the application. In some cases, the user has performed block level operations on his data.

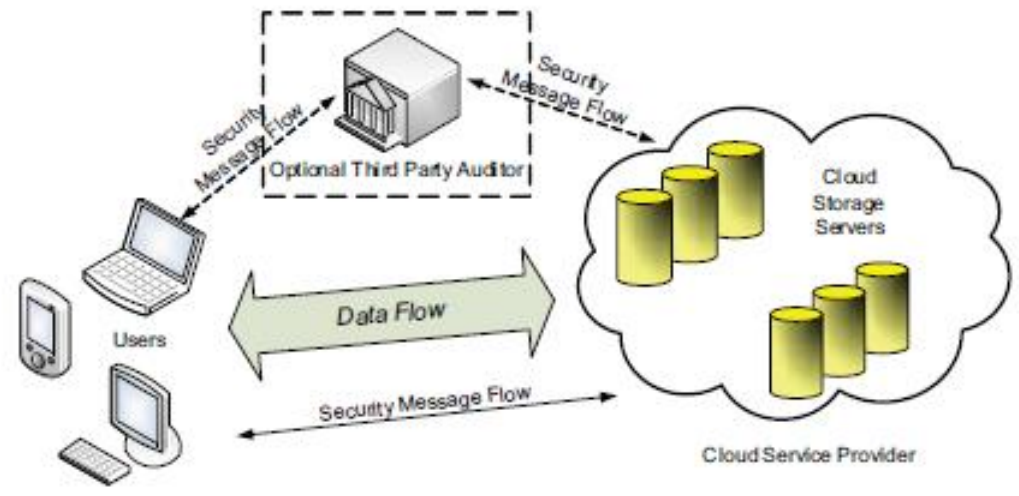

Fig 1: Cloud System Architecture

The most common styles are block insert, delete, update, and append, which are all separate operations. Because users no longer have knowledge of data location, it's critical to inform them that their data is being properly stored and maintained. That is, users should be provided with security tools so that they can ensure the continuous correctness of their data stored in the cloud, even if there is no local copy of the data.

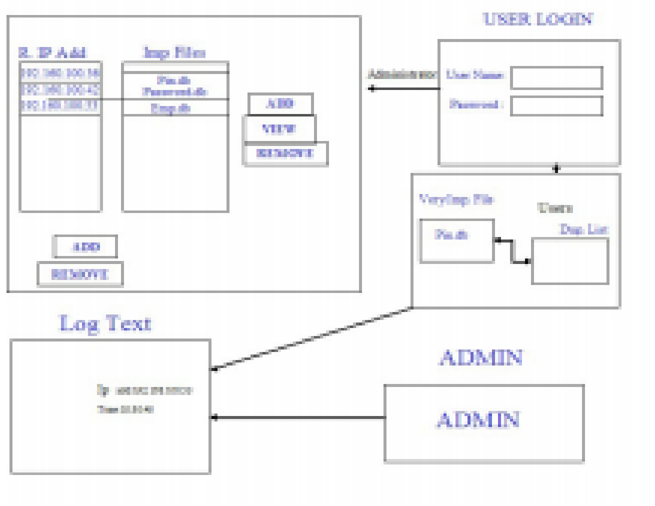

Fig 2: Cloud Server Data Security Architecture

If users don't have the time or resources to check their data, they can delegate the task to a trusted Third Party Auditor (TPA) of their choice. We assume in this model that the point-to-point communication channels between each user and the cloud server are authenticated and trustworthy, which can be achieved with little overhead in practice. Note that this type of data privacy issue is not addressed here because, in Cloud Computing, data privacy is unrelated to the problem we're looking at.

2. KEY GENERATION ALGORITHM

2.1. Key generation (μ)

```
Step 1. Start
    2. Obtain parameters (g,h,p).
    3. G1 can be generated by g and h.
    4. G2 can be generated by prime number p.
    5. Set μ=(g,h,ẽ,G1,G2,p,f)
    6. f:Zp*{0,1} Zp*Is a one-way hash function
    7. User A selects three parameters $a_1;a_2;\ a_3\ Z_p*$
        8. $PK_A$=( ),ska=($a_1,a_2,a_3$).
    9. Stop.
```

2.2. Share Key Generation Algorithm

Saharekeygen(SKA,t,m)

```
Step 1. Start
    2. Select m key servers
    3. Share secret key $SK_A$ to key server $ks_i$
    4. $SK_{A,i}=(f_{A,1}(i),f_{A,2}(i))$ where $1\leq i \leq m$.
    5. Stop.
```

2.3. Encryption Algorithm

Encryption(PKA,T,m1,m2,…..mk)

```
Step 1: Start
    2. Divide message in to m1,m2,…..mk
    3. Calculate cipher text c1,c2,…..ck by using
)
    4. Stop.
```

2.4. Key Recovery Algorithm

Key Recover(SKA,i1 , SKA,i2 , SKA,i3 , SKA,i4…………… , SKA,in)

```
Step 1. Start
    2. User searches for first component a1.
    3. if (a1 found)
            No need of key recovery.
    4. if (a1 not found)
        Recovered by using
    5. Stop
```

2.5. Re Key Generation Algorithm

ReKeyGen(PKA,SKA,ID,PKB)

```
Step 1. Start
    2. Encrypt the data by using SKA.
    3. Select a random number e from Zp*.
    4.Compute rekey for proxy re-encryption by using
    5. Stop.
```

2.6. Proxy Re-Encryption Algorithm

ProxyReEncryption()

```
Step 1. Start
    2. Calculate RK, C’.
```

3. Re-Encrypted symbol can be computed as

$$C'' = (1, \alpha, h^{b_{2}a_{1}}(\mathrm{f}(a_{3},\mathrm{ID})+\mathrm{e}), \gamma . \tilde{e}\,(\alpha, h^{a1e}))$$

$$= (1, g^{r'}, h^{b_{2}a_{1}}(\mathrm{f}(a_{3},\mathrm{ID})+\mathrm{e}), \mathrm{W}\tilde{e}\,(\mathrm{g},\mathrm{h})^{a1r'((\mathrm{f}(a_{3},\mathrm{ID})+\mathrm{e})})$$

3. Stop

3 IMPLEMENTATION

3.1 Client Module

The server receives a query from the client in this client module. It sends the corresponding file to the client based on the query sent by the user to the server. The client should first give his or her permission before proceeding. For security purposes, the server verifies the client's name and password. If it has been successfully verified and the client's queries have been received, it will search the database for the corresponding files. Finally, it locates and sends the files to the client. If the server discovers the intruder's means, it generates an alternative Path for them.

3.2. System Module

Figure 1 shows a representative network architecture for cloud data storage. There are three network entities to be identified, as shown below.

3.2.1. User

Individual consumers and organisations both are users who have data to store in the cloud and rely on the cloud for data computation.

3.2.2. Cloud Service Provider (CSP)

A live cloud computing system is controlled by a Cloud Service Provider who has major resources and expertise in managing and building distributed servers for cloud storage.

3.2.3. Third Party Auditor (TPA)

On request, an optional TPA with expertise and capabilities that users may lack is trusted to assess and expose risks involved with cloud storage services on behalf of the users.

3.3. Cloud data storage Module

This is when a user uses CSP to store data in a group of cloud servers that are all running at the same time, and then interacts with the cloud servers via CSP to retrieve or access his data. The user may need to perform block level operations on his own data in some cases. As a result, users should be equipped with security tools in order to build a continuous correctness verification of their data stored in the cloud, even if there is no local copy of the data. If the user does not have the time or resources to monitor their data, they can take the decision to a trusted Third Party Auditor (TPA) of their choice.

In this model, we assume that the user and the cloud server have point-to-point communication channels, which can be achieved with little overhead in practise if the user is authenticated and reliable.

3.4. Cloud Authentication Server

The Authentication Server (AS) performs the same functions as any other Authentication Server, but with a few additional behaviours added to the standard client-authentication protocol. First, send client authentication information to the masquerading router. On the application network, the Authentication Server also provides as an authorising permission control and checking tickets server. Some other optional function that the AS should support is the updating of client lists, which can result in a reduction in authentication time or even the removal of a valid client depending on the request.

3.5. Unauthorized data modification and corruption module

One of the most important issues is to effectively detect any unauthorised data modification or corruption, which could occur as a result of server conciliation and/or random Byzantine failures. Also, in the distributed case, when that type of inconsistency is successfully detected, determining which server the data error is placed on is critical.

3.6. Adversary Module

Data storage in the cloud is highly secured from two sources. For First, a CSP can be self-interested, untruthful, and even malicious. Its objective is to not only move data that hasn't been used or is only used rarely to a lower tier of storage than is acceptable for monitoring purposes, but there's also a chance it'll try to hide a data loss incident caused by Byzantine failures or management errors, for eg.

Second, there is a possibility of an adversary who is financially motivated, has the ability to compromise a number of cloud data storage servers in a set of time intervals, and can modify or delete user data for a period of time while remaining undetected by CSPs. Consider two different types of adversary models, each with various different levels of capability.

3.6.1. Weak Adversary

This adversary is interested in destroying user data files stored on individual servers. If a server is compromised, the adversary can corrupt the original data files by introducing or modifying deceptive data to prevent the user from retrieving the original data.

3.6.2. Strong Adversary

This adversary represents the worst-case scenario, in which we expect the adversary to compromise all other storage servers so that he or she can change data files on purpose as long as they are internally reliable. In fact, this is analogous to a situation in which all of the servers work together to hide a data loss or corruption event.

4. CONCLUSION AND FUTURE WORK

We look at the data security problem of data storage in the cloud, which is essentially a distributed storage system, in this paper. To ensure the accuracy of users' data stored in the cloud, we proposed an efficient and flexible distributed scheme that explicitly supports dynamic data, including block update, append, and delete. For the preparation of file distribution, we depended on erasure-correcting code to provide a parity vector of redundancy and ensure data dependability. Our system achieves that integration of storage correctness assurance and localization of data errors by using the homomorphic token of erasure coded data with distributed verification, that is whenever data corruption is detected during the correctness verification of data storage in the cloud across distributed servers. Now, we can almost guarantee that the misbehaving servers will be identified simultaneously. We illustrated that our scheme is highly resilient and efficient to malicious data modification attacks, Byzantine Failures, and server collusion attacks through this detailed performance and security analysis.

We believe that data storage security in Cloud Computing is an area that is full of challenges and vital importance, but it is still in its early life, and there are many research problems in this area that have yet to be identified. Future research in this area could go in a number of directions, according to us. We believe that a model in which public verifiability is required is the most exciting. The Third Party Auditor (TPA) can audit the data storage in the cloud without requiring users' time, resources, or feasibility, thanks to public verifiability. A difficult question here is whether we can implement a scheme that ensures both storage correctness and public verifiability of dynamic data. We also intend to investigate the problem of fine-grained data error localization as part of this research on dynamic data storage in the cloud.

REFERENCES

[1] Cong Wang, Qian Wang, and Kui Ren, "Ensuring Data Storage Security in Cloud Computing " in Proc. of IWQoS'09, July 2009, pp. 1–9

[2] G. Ateniese, R. D. Pietro, L. V. Mancini, and G. Tsudik, "Scalable and Efficient Provable Data Possession," Proc. of SecureComm '08, pp. 1–10, 2008.

[3] Amazon.com, "Amazon Web Services (AWS)," Online at http://aws.amazon.com, 2008. [4] A. Juels and J. Burton S. Kaliski, "PORs: Proofs of Retrievability for Large Files," Proc. of CCS '07, pp. 584–597, 2007.

[5] H. Shacham and B. Waters, "Compact Proofs of Retrievability," Proc. of Asiacrypt '08, Dec. 2008. [6] K. D. Bowers, A. Juels, and A. Oprea, "Proofs of Retrievability: Theory and Implementation," Cryptology ePrint Archive, Report 2008/175, 2008, http://eprint.iacr.org/.

[7] 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/. [8] G. Ateniese, R. Burns, R. Curtmola, J. Herring, L. Kissner, Z. Peterson, and D. Song, "Provable Data Possession at Untrusted Stores," Proc. Of CCS '07, pp. 598–609, 2007.

[9] R. Curtmola, O. Khan, R. Burns, and G. Ateniese, "MR-PDP: Multiple- Replica Provable Data Possession," Proc. of ICDCS '08, pp. 411–420, 2008.

[10] M. Lillibridge, S. Elnikety, A. Birrell, M. Burrows, and M. Isard, "A Cooperative Internet Backup Scheme," Proc. of the 2003 USENIX Annual Technical Conference (General Track), pp. 29–41, 2003.

[11] D. L. G. Filho and P. S. L. M. Barreto, "Demonstrating Data Possession and Uncheatable Data Transfer," Cryptology ePrint Archive, Report 2006/150, 2006, http://eprint.iacr.org/. [12] M. A. Shah, M. Baker, J. C. Mogul, and R. Swaminathan, "Auditing to Keep Online Storage Services Honest," Proc. 11th

USENIX Workshop on Hot Topics in Operating Systems (HOTOS '07), pp. 1–6, 2007.

[13] T. S. J. Schwarz and E. L. Miller, "Store, Forget, and Check: Using Algebraic Signatures to Check Remotely Administered Storage," Proc. of ICDCS '06, pp. 12–12, 2006.

[14] N. Gohring, "Amazon's S3 down for several hours," Online at http://www.pcworld.com/businesscenter/article/142549/amazons s3 down for several hours.html, 2008.

[15] 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/.

[16] L. Carter and M. Wegman, "Universal Hash Functions," Journal of Computer and System Sciences, vol. 18, no. 2, pp. 143–154, 1979.

[17] J. Hendricks, G. Ganger, and M. Reiter, "Verifying Distributed Erasurecoded Data," Proc. 26th ACM Symposium on Principles of Distributed Computing, pp. 139–146, 2007.

[18] J. S. Plank and Y. Ding, "Note: Correction to the 1997 Tutorial on Reed-Solomon Coding," University of Tennessee, Tech. Rep. CS-03- 504, 2003.

[19] Q. Wang, K. Ren, W. Lou, and Y. Zhang, "Dependable and Secure Sensor Data Storage with Dynamic Integrity Assurance," Proc. of IEEE INFOCOM, 2009.

[20] R. Curtmola, O. Khan, R. Burns, and G. Ateniese, "MR-PDP: Multiple- Replica Provable Data Possession," Proc. of ICDCS '08, pp. 411–420,2008.

[21] D. L. G. Filho and P. S. L. M. Barreto, "Demonstrating Data Possession and Uncheatable Data Transfer," Cryptology ePrint Archive, Report 2006/150, 2006, http://eprint.iacr.org/.