Volume 6, Issue 5, May 2016 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

Secure Data Computing in Cloud

[1] Swati Malik, [2] Mayank Singh
[1] M.Tech, [2] H.O.D.
[1, 2] Department of CSE, Krishna Engineering College, Mohan Nagar, Ghaziabad, Dr. A.P.J Abdul Kalam Technical University, Lucknow, Uttar Pradesh, India

Abstract— Cloud computing has emerged as a new way of computing that changes the way of obtaining computing resources and managing and delivering of computing services. It is a movement away from applications needing to be installed on an individual's computer towards the applications being hosted online. Organizations regardless of size are taking cloud computing as a way to reduce the costs and complexities associated with traditional IT approaches. Apart from reduction in storage costs data outsourcing to the cloud also helps in reducing the maintenance. . The other side to this emerging technology is that with cloud services; security issues of confidentiality, data integrity and data availability has been introduced. Cloud storage moves the user's data to large data centers, which are remotely located, on which user does not have any control. However, this unique feature of the cloud poses many new security challenges which need to be clearly understood and resolved. One of the important security concerns that need to be addressed is to assure integrity of the data. As the data is physically not accessible to the user the cloud service provider should provide a way for the user to check if the integrity of his data is maintained and has not compromised. In this paper we provide an approach which gives a proof for verification process to ensure data integrity in the cloud which the customer can employ to check the correctness of his data in the cloud. This demonstration can be agreed upon by both the customer and the cloud and can be incorporated in the Service level agreement (SLA).

Keywords— Cloud Computing, Cloud Infrastructure, Cloud Security, Metadata Generation, Data Integrity, Cloud Service, Encryption.

I. INTRODUCTION

Cloud computing is an emerging computing paradigm where data and application reside in the cyberspace. It received attention as it changes the way of services to customers. Computing resources such as CPUs, databases and storage systems are provided and managed in a changed way. Cloud computing facilitates the development and utilization of highly flexible, elastic services on-demand, and over broadband network access. It brings new business opportunities, driving many organizations to move their businesses to a cloud platform. The term cloud computing refers to performing computer tasks using services delivered entirely over the Internet. Cloud Computing is a movement away from applications needing to be installed on an individual's computer towards the applications being hosted online. The "cloud" refers to the Internet. Data is neither stored on the local hard drive of your computer, nor on servers located at your home or company. Instead it is stored out in the cloud, the infrastructure is outside of your organization and you access the applications, the infrastructure and the services typically through the web based interface. Web-based email services like Gmail and Hotmail deliver a cloud computing service, users can access their email in the cloud from any computer with a browser and Internet connection, regardless of what kind of hardware is on that particular computer. Over the last few years, there is a tremendous growth seen in cloud computing as witnessed by many popular web applications used today including VoIP (e.g., Skype, Google Voice), Social Applications (e.g., Facebook, Twitter, LinkedIn), media services (e.g., Picasa, You Tube, Flickr), content distribution (e.g., Bit Torrent), financial apps(e.g., Mint), and many more. Even traditional Desktop software, such as Microsoft Office, has moved in part to the web, starting with its Office 2010 web Apps. Cloud computing enables ubiquitous, convenient, on-demand network access to a shared pool of computing resources (e.g., networks, servers, storage, applications and services).

The message exchange or data transmission would take place among three identities cloud entities TTP, CSP, Cloud Customer (Client).

TPA- A cloud auditor is an independent party who has expertise and capabilities to examine a cloud service stack and verifies the integrity of outsourced data in cloud and provide an assessment on privacy, security and availability level of the associated cloud services and ensures that the corresponding Service Level Agreement (SLAs) are fulfilled

CSP- A cloud service provider is an entity who is responsible for everything essential for making cloud service available for the users. The CSP, who manages cloud servers and provides a funded storage space on its infrastructure to the users. Servers are geographically hosted on the different locations across the globe but accessible to the cloud users.

CUSTOMER- A customer is either a cloud service consumer or a cloud service owner. Cloud service owner is an individual or an organization who subscribes for a cloud service. If there is any charge associated with this service, the cloud service owner will be responsible for the charges. Cloud service consumer is the subscribers-individual or application who accesses a cloud service through any of the computing devices like Mobile, PDAs and smart phones

A. Characteristic

Cloud services free businesses and consumers from having to invest in hardware or install software on their devices. IT reduces maintenance and hardware upgrading needs, work can be performed from any computing device anywhere as long as there is a access to the web. Cloud computing has following essential characteristics.

1) On-demand Self-Service: is an appealing characteristic for consumers because it provides them the flexibility of provisioning a service exactly when needed.

2) Rapid Elasticity: allows end users to easily and rapidly provision new services and release them, enabling them to pay for what they utilize and how much they use it.

3) Measured Services : The services provided over the cloud are measured services, which means that consumers only pay for how much service they consume; thus eliminating the need for investing in redundant computing resources. Cloud computing has benefits at the providers' end as well.

4) Multi – Tenant: A cloud computing provider pools its computing resources in order to serve multiple consumers by means of a multi-tenant provisioning model.

B. Service Delivery Models

There are several service delivery models of cloud computing services and are described in this paper.

1) Software as a Service (SaaS): delivery model provides the users with access to software applications/services already deployed over cloud infrastructure for consumers. These applications are accessible from various platforms through an easy-to-use client interface such as a web browser. Some examples are Google Doc and Gmail.

2) Platform as a Service (PaaS): delivery model provides the user with development environment services which enables consumers to create and run home grown application and deploy their solutions to the cloud by means of platforms such as application servers and database services provided by the Cloud Platform Provider. Some examples are AppEngine and Bungee Connect.

3) Infrastructure as a Service (IaaS): delivery model provides the user with virtual infrastructure, such as servers, processing, networking and data storage space. In IaaS consumers acquire computing services and can deploy their own custom configured systems in these resources. Amazon S3.

C. Deployment Models

The cloud infrastructure deployment models are of following four types, important to consider when delivering on-line services. A cloud is a combination of infrastructure, software and services which are not local to users.

1) Public Cloud Infrastructures: are used by general public cloud consumers or we can say provisioned for use by any consumer. Infrastructure exists in the premise of the provider. Cloud provider has full ownership of the cloud with its own policy, profit, value, costing, and charging model.

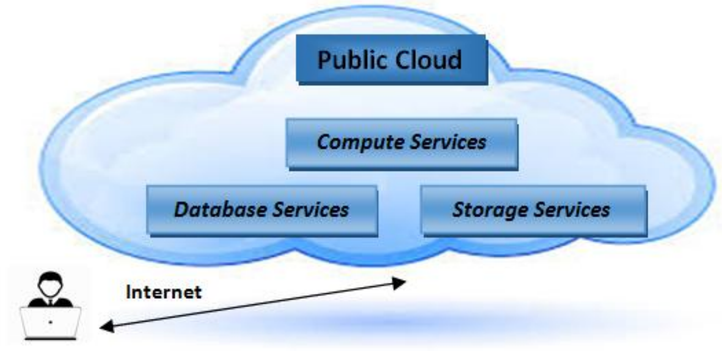

Fig. 1 Public Cloud

2) Private Cloud Infrastructure: is operated solely within a single organization. Cloud can be owned and managed by a single organization, a third party regardless of whether it is located off-premise or on-premise.

3) Community Cloud Infrastructure: is provisioned for exclusive use by a particular community of users from organizations that constitute the specific community.Several organizations jointly construct and share the same cloud infrastructure. Cloud is hosted by a third-party vendor or within one organizations in the community.

4) Hybrid Cloud Infrastructure: is a composition of two or more clouds including private, public or community cloud which remain discrete from another but, offers data and application portability between each cloud using standardized or proprietary technology (e.g., for cloud bursting for load balancing between clouds).

Storing of user data in the cloud despite its advantages has many interesting security concerns which need to be extensively investigated for making it a reliable solution to the problem of avoiding local storage of data. Many problems like data authentication and integrity (i.e., how to efficiently and securely ensure that the cloud storage server returns correct and complete results in response to its clients' queries [1]), outsourcing encrypted data and associated difficult problems dealing with querying over encrypted domain [2] were discussed in research literature.

In this paper we deal with the problem of implementing a protocol to obtain and verify a proof that the data that is stored by a user at a remote data storage in the cloud (called cloud storage archives) is not modified by the archive and thereby the integrity of the data is assured. Such kinds of proofs are very helpful in peer-to-peer storage systems, network file systems, long-term archives, web-service object stores, and database systems. Such verification systems prevent the cloud storage archives from misrepresenting or modifying the data stored at it without the consent of the data owner by using frequent checks on the storage archives. Such checks must allow the data owner to efficiently, frequently, quickly and securely verify that the cloud archive is not cheating the owner. While developing proofs for data possession at untrusted cloud storage servers we are often limited by the resources at the cloud server as well as at the client. Given that the data sizes are large and are stored at remote servers, accessing the entire file can be expensive in I/O costs to the storage server. Also transmitting the file across the network to the client can consume heavy bandwidths. Since growth in storage capacity has far outpaced the growth in data access as well as network bandwidth, accessing and transmitting the entire archive even occasionally greatly limits the scalability of the network resources. Furthermore, the I/O to establish the data proof interferes with the on-demand bandwidth of the server used for normal storage and retrieving purpose. The problem is further complicated by the fact that the owner of the data may be a small device, like a PDA (personal digital assist) or a mobile phone, which have limited CPU power, battery power and communication bandwidth. Hence a data integrity proof that has to be developed needs to take the above limitations into consideration. The scheme should be able to produce a proof without the need for the server to access the entire file or the client retrieving the entire file from the server. Also the scheme should minimize the local computation at the client as well as the bandwidth consumed at the client.

The remainder of the paper is organized as follows; In Section II , Related Work; Section III, Approach ; Section IV, Techniques and algorithm used; Section V, Conclusion ; Section VI, Future Scope.

II. RELATED WORK

The simplest scheme can be made using a keyed hash function $h_k(F)$. In this scheme the verifier, before archiving the data file F in the cloud storage, pre-computes the cryptographic hash of F using $h_k(F)$ and stores this hash as well as the secret key K. To check if the integrity of the file F is lost the verifier releases the secret key K to the cloud archive and asks it to compute and return the value of $h_k(F)$. By storing multiple hash values for different keys the verifier can check for the integrity of the file F for multiple times, each one being an independent proof.

Though this scheme is very simple and easily implementable the main drawback of this scheme are the high re-source costs it requires for the implementation. At the verifier side this involves storing as many keys as the number of checks it want to perform as well as the hash value of the data file F with each hash key. Also computing hash value for even a moderately large data files can be computationally burdensome for some clients(PDAs, mobile phones, etc). As the archive side, each invocation of the protocol requires the archive to process the entire file F . This can be computationally burdensome for the archive even for a lightweight operation like hashing. Furthermore, it requires that each proof requires the prover to read the entire file F - a significant overhead for an archive whose intended load is only an occasional read per file, were every file to be tested frequently [3].

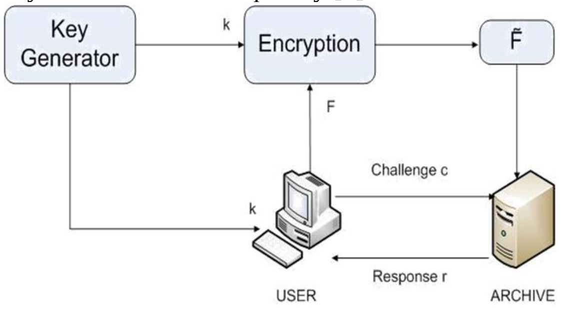

Fig. 2 Schematic view of a proof of retrievability based on inserting random sentinels in the data file F.

Ari Juels and Burton S. Kaliski Jr proposed a scheme called Proof of retrievability for large files using "sentinels"[3]. In this scheme, unlike in the key-hash approach scheme, only a single key can be used irrespective of the size of the file or the number of files whose retrievability it wants to verify. Also the archive needs to access only a small portion of the file F unlike in the key-has scheme which required the archive to process the entire file F for each protocol verification. This small portion of the file F is in fact independent of the length of F . The schematic view of this approach is shown in Figure 1.

In this scheme special blocks (called sentinels) are hidden among other blocks in the data file F . In the setup phase, the verifier randomly embeds these sentinels among the data blocks. During the verification phase, to check the integrity of the data file F , the verifier challenges the prover (cloud archive) by specifying the positions of a collection of sentinels and asking the prover to return the associated sentinel values. If the prover has modified or deleted a substantial portion of F , then with high probability it will also have suppressed a number of sentinels. It is therefore unlikely to respond correctly to the verifier. To make the sentinels indistinguishable from the data blocks, the whole modified file is encrypted and stored at the archive. The use of encryption here renders the sentinels indistinguishable from other file blocks. This scheme is best suited for storing encrypted files.

As this scheme involves the encryption of the file F using a secret key it becomes computationally cumbersome especially when the data to be encrypted is large. Hence, this scheme proves disadvantages to small users with limited computational power (PDAs, mobile phones etc.). There will also be a storage overhead at the server, partly due to the newly inserted sentinels and partly due to the error correcting codes that are inserted. Also the client needs to store all the sentinels with it, which may be a storage overhead to thin clients (PDAs, low power devices etc.).

III. APPROACH

We are providing a scheme which does not involve the whole data's encryption. It encrypts only few bits of data per data blocks, reducing the computational overhead on the clients. The client storage overhead is also minimized as it does not store any data with it. This scheme suits well for thin clients.

In this data integrity protocol verifier needs to store only a single cryptographic key - irrespective of the size of data file F - and two functions which generate a random sequence. Verifier does not store any data with it. The verifier before storing the file at archive, preprocesses file and appends some meta data to the file and stores at the archive. At the time of verification verifier uses this meta data to verify the integrity of the data. It is important to note that our proof of data integrity protocol just checks the integrity of data i.e. if data has been illegally modified or deleted. It does not prevent the archive from modifying the data. In order to prevent such modifications or deletions schemes like redundant storing etc, can be implemented which is not a scope of discussion in this paper.

IV. TECHNIQUES AND ALGORITHM USED

We are generating proof for data integrity based on random bit selection here. The client before storing its data file F at the cloud should process it and create suitable meta data which is used in the later stage of verification the data integrity at the cloud storage. When checking for data integrity the client queries the cloud storage for suitable replies based on which it concludes the integrity of its data stored in the cloud.

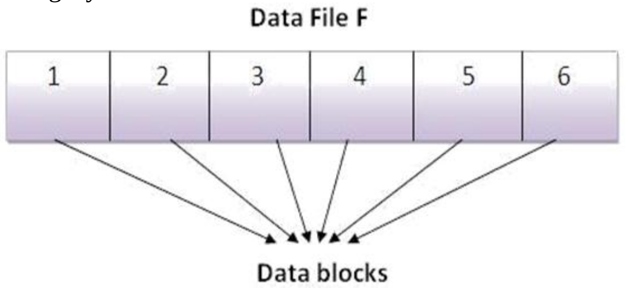

Fig. 3 A data file F with 6 data blocks.

A. Setup Phase

Let the verifier V wishes to the store the file F with the archive. Let this file F consist of n file blocks. We initially preprocess the file and create metadata to be appended to the file. Let each of the n data blocks have m bits in them. A typical data file F which the client wishes to store in the cloud is shown in Figure 2. The initial setup phase can be described in the following steps

1) *Generation of Meta-Data:* Let g be a function defined as follows

$$g(i, j) \rightarrow \{1..m\},\ i \in \{1..n\},\ j \in \{1..k\} \qquad (1)$$

Where k is the number of bits per data block which we wish to read as meta data. The function g generates for each data block a set of k bit positions within the m bits that are in the data block. Hence g(i, j) gives the j^{th} bit in the i^{th} data block. The value of k is in the choice of the verifier and is a secret known only to him. Therefore for each data block we get a set of k bits and in total for all the n blocks we get n* k bits. Let m_i represent the k bits of meta data for the i^{th} block. Figure 3 shows a data block of the file F with random bits selected using the function g.

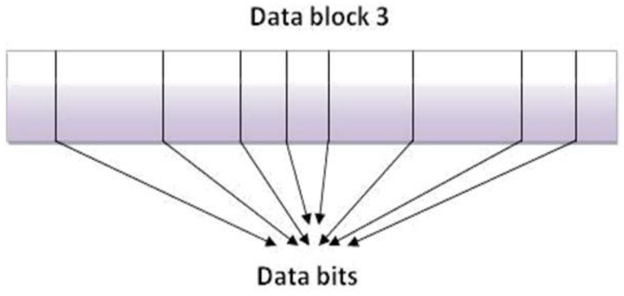

Fig.4 A data block of the file F with random bits selected in it.

2) *Encrypting the Meta Data:* Each of the meta data from the data blocks m_i is encrypted by using a suitable algorithm to give a new modified meta data M_i.

Without loss of generality we show this process by using a simple XOR operation. Let h be a function which generates a k bit integer α_i for each i. This function is a secret and is known only to the verifier V .

$$h : i \rightarrow \alpha_i,\ \alpha_i \in \{0..2^n\} \qquad (2)$$

For the meta data (m_i) of each data block the number α_i is added to get a new k bit number M_i.

$$M_i = m_i + \alpha_i \qquad (3)$$

$M_i = m_i + \alpha_i$ In this way we get a set of n new meta data bit blocks. The encryption method can be improvised to provide still stronger protection for verifiers data.

3) Appending of Meta Data: All the meta data bit blocks that are generated using the above procedure are to be concatenated together. This concatenated meta data should be appended to the file F before storing it at the cloud server. The file F along with the appended meta data F^e is archived with the cloud.

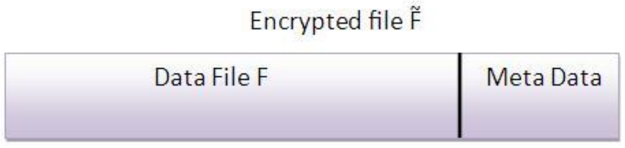

Fig.5 The encrypted file which will be stored in cloud.

B. Verification Phase

Let the verifier V want to verify the integrity of the file F . It throws a challenge to the archive and asks it to respond. The challenge and the response are compared and the verifier accepts or rejects the integrity proof.

Suppose the verifier wishes to check the integrity of n^{th} block. The verifier challenges the cloud storage server by specifying the block number i and a bit number j generated by using the function g which only the verifier knows. The verifier also specifies the position at which the meta data corresponding the block i is appended. This meta data will be a k-bit number. Hence the cloud storage server is required to send k+1 bits for verification by the client.

The meta data sent by the cloud is decrypted by using the number α_i and the corresponding bit in this decrypted meta data is compared with the bit that is sent by the cloud. Any mismatch between the two would mean a loss of the integrity of the clients data at the cloud storage.

Let the verifier V wishes to the store the file F with the archive. Let file F consist of n file blocks. We initially preprocess file and create metadata to be appended to the file. Let each of the n data blocks have m bits in them. A typical data file F which client wishes to store in the cloud.

Each of the Meta data from the data blocks m_i is encrypted by using a suitable algorithm to give a new modified Meta data M_i. Without loss of generality we show this process by using a simple XOR operation. All the Meta data bit blocks that are generated using the above procedure are to be concatenated together. This concatenated Meta data should be appended to the file F before storing it at the cloud server. The file F along with the appended Meta data e F is archived with the cloud.

V. CONCLUSION

We have worked to facilitate the verification process to ensure integrity of the data which client wishes to store in the cloud storage servers with efforts and minimum costs.This scheme reduces the computational and storage overhead of the client as well as to minimize the computational overhead of the cloud storage server.It also minimizes the size of the data integrity proof so as to reduce the consumption of network bandwidth. Many schemes which was proposed earlier require the archive to perform tasks that need a lot of computational power to ensure data integrity. But in this scheme the archive just need to fetch and send few bits of data to the client.

VI. FUTURE SCOPE

Our scheme applies to static storage of data only. It cannot handle when the data is changed dynamically. Hence it will be a future challenge to developing on this. The number of queries that can be asked by the client is fixed also. To increase the number of queries using this scheme will be a challenge.

ACKNOWLEDGMENT

I would like to thank my guide Dr. Mayank Singh and faculty members of Department of CSE, Krishna Engineering College for their constant support and assistance.

REFERENCES

[1] E. Mykletun, M. Narasimha, and G. Tsudik, “Authentication and integrity in outsourced databases,” Trans. Storage, vol. 2, no. 2, pp. 107–138, 2006.

[2] D. X. Song, D. Wagner, and A. Perrig, “Practical techniques for searches on encrypted data,” in SP ’00: Proceedings of the 2000 IEEE Symposium on Security and Privacy. Washington, DC, USA: IEEE Computer Society, 2000, p. 44.

[3] A. Juels and B. S. Kaliski, Jr., “Pors: proofs of retrievability for large files,” in CCS ’07: Proceedings of the 14th ACM conference on Computer and communications security. New York, NY, USA: ACM, 2007, pp. 584–597.

[4] G. Ateniese, R. Burns, R. Curtmola, J. Herring, L. Kissner, Z. Peterson, and D. Song, “Provable data possession at untrusted stores,” in CCS ’07: Proceedings of the 14th ACM conference on Computer and communica-tions

security. New York, NY, USA: ACM, 2007.

[5] Dinesh.C, "Data Integrity and Dynamic Storage Way in Cloud Computing ", published in International Journal of Computer Applications (0975 – 8887) Volume 31– No.6, September 2011.

[6] Dr. Nedhal A. Al-Saiyd, Nada Sail, "Data Integrity in Cloud Computing Security , published in Journal of Theoretical and Applied Information Technology (JTATT), Vol. 58 No.3, December 2013.

[7] Adam Swidler, " 7 Security threats in the cloud", published in Campus Technology Magazine- Cloud Security Alliance (CSA) December 2013.

[8] Ting Sang, "A Log Based Approach to Make Digital Forensics Easier on Cloud Computing", published in Third International Conference on Intelligent System Design and Engineering Applications, 2013.

[9] Rohit Ranchal, Bharat Bhargava, Lotfi Ben Othmane, Leszek Lilien, Anya Kim, Myong Kang, Mark Linderman "Protection of Identity Information in Cloud Computing without Trusted Third Party", published in 29th IEEE International Symposium on Reliable Distributed Systems,2010.