International Journal of Advance Research in Science and Engineering
Vol. No.6, Special Issue (01), September 2017, BVCNSCS 2017
www.ijarse.com
IJARSE
ISSN 2319 - 8354

ENSURING DATA INTEGRITY IN CLOUD STORAGE USING ECC TECHNIQUE

S.Ramana
Research Scholar,
Dept. of Computer Science,
Osmania University, Hyderabad,Telangana,India

M.V.Ramana Murthy
Professor and B.O.S Chairman,
Dept. of Computer Science & Mathematics,
Osmania University, Hyderabad,Telangana,India

N Bhaskar
Research Scholar,
Dept. of Computer Science, Rayalaseema Univesrity,
Kurnool, A.P, India.

Abstract: The entire world is looking at the future and one of the future components in IT Enterprise is cloud computing which has become the default solution to the increased storage costs of IT Enterprises. Many companies like Amazon, Google, Microsoft, IBM and so on are investing huge amount in cloud storage and development and providing services to many users across the world.
With the increase costs in data storage devices as well as the fast exponential rate at which data is being generated it proves costly for enterprises or individual users to frequently modify their hardware. Apart from reduction in storage costs, data outsourcing to the cloud also helps in reducing the maintenance.
Cloud storage transfers the data of users to large data centers, which are located in different locations across the world, on which user does not have any control. However, this unique feature of the cloud raises many new security issues which need to be understood and resolved clearly.
One of the important concerns that need to be answered is to assure the customer of the integrity i.e. correctness of his data in the cloud. As the data is physically not accessible to the user the cloud should provide a way for the user to check if the integrity of his data is maintained or is compromised. This paper provides a scheme which gives a proof of data integrity in the cloud which the customer can employ to check the correctness of his data in the cloud. This proof can be agreed upon by both the cloud and the customer and can be incorporated in the Service Level Agreement (SLA). This scheme ensures that the storage space used at the client side for running the algorithm is minimal which will be beneficial for thin clients.

Key words: Cloud Storage, data centres, security, integrity, compromised, SLA.

I. INTRODUCTION

Cloud computing is matched with the early production of electricity. Families, trades and municipalities did not want to yield or trust on their own source of power. They began connecting into a bigger power grid, maintained and organized by power utilities. Along with this utility linking came time and cost savings, in addition to bigger access to, and more reliable availability of power.

Similarly, cloud computing signifies substantial chance for service providers and enterprises. Cloud computing has become a progressively popular means of providing valuable IT-enabled business services. Accepting cloud technology can be a reasonable way to get access to a dynamically scalable, virtualized computing environment. Optimal IT hardware, software, expertise and infrastructure management resources that may not otherwise be available from a cost perspective can be quickly installed and easily scaled. Processes, applications and services can be available on request, regardless of the user location or device. The cloud provider is responsible for the setting, so organizations can make use of resources for short periods of time without having to maintain the setting when it is not being used.

In this paper we provide a solution to the Data Integrity problem in Cloud.

II. ISSUES IN CLOUD COMPUTING

While cloud computing models are eye-catching because of their springiness and cost effectiveness, certain tasks must be addressed in order to provide a feasible option to traditional data services. First and foremost is the issue of security. The externalized aspect of outsourcing can make it harder to maintain data integrity and confidentiality, support data and service availability, demonstrate compliance, and secure highly available access to applications and information. In short, cloud computing can contemporary an added level of risk.

The comparative safety of cloud computing services is an argumentative issue that may be postponing its adoption. Issues barring the adoption of cloud computing are due in large part to the private and public sectors.It is the very nature of cloud computing-based services, private or public, that promote external management of provided services. This delivers great motivation to cloud computing service providers to prioritize building and maintaining strong management of secure services. Security issues have been categorized into

1.Availability
2. Confidentiality
3. Data Integrity
4.Control
5. Audit

III. DATA INTEGRITY IN CLOUD COMPUTING

Data Integrity in its widest meaning refers to the honesty of information over its entire life cycle. In more analytic terms, it is "the representational faithfulness of information to the true state of the object that the information represents, where representational faithfulness is composed of four essential qualities or core attributes: completeness, currency/timeliness, accuracy/correctness and validity/authorization."

Data Integrity is very important in database operations in particular and Data Warehousing and Business Intelligence in general, because Data Integrity ensures that data is of high quality, correct, consistent and accessible.

Cloud storage can be an eye-catching means of outsourcing the day-to-day management of data, but ultimately the accountability and liability for that data falls on the company that owns the data, not the hosting provider. With this in mind, it is important to understand some of the causes of data corruption, how much responsibility a cloud service provider holds, some basic best practices for utilizing cloud storage safely, and some methods and standards for monitoring the integrity of data regardless of whether that data resides locally or in the cloud.

Integrity checking is essential in cloud storage for the same reasons that data integrity is serious for any data center. Data corruption can happen at any level of storage and with any type of media. Here are some of the examples of different media types causing corruption, Bit rot (the weakening or loss of bits of data on storage media), controller failures, deduplication, metadata corruption, and tape failures. Metadata corruption can be the result of any of the vulnerabilities listed above, such as bit rot, but are also vulnerable to software glitches outside of hardware error rates. Unfortunately, a side effect of deduplication is that a corrupted file, block, or byte affects every related piece of data tied to that metadata. The truth is that data corruption can happen anywhere within a storage environment. Data can become corrupted simply by migrating it to a different platform, i.e., sending your data to the cloud. Cloud storage systems are still data centers, with hardware and software, and are still vulnerable to data corruption. One needs to look no further than the recent highly publicized Amazon failure. Not only did many companies suffer from prolonged downtime, but 0.07 percent of their customers actually lost data. It was reported that this data loss was caused by 'recovering an inconsistent data snapshot of Amazon ESB volumes. What this translates to is that data in Amazon's system became corrupted, and as a result, customers lost data.

Whenever data is lost, especially valuable data, there is a tendency to scramble and assign blame. Often in the IT world, this can result in loss of jobs, downfall in company revenue, and, in severe cases, business expiry. As such, it is serious to understand how much legal responsibility the cloud service provider, as per the service level agreement (SLA), has and to ensure that every possible step has been taken to prevent data loss. As with many legal documents, SLAs are often written to the benefit of the provider, not to the customer. Many cloud service providers offer varying tiers of protection, but as with any storage provider they do not assume responsibility for the integrity of your data.

Cloud SLA language that contains explicit statements protecting the cloud provider if data is lost or corrupted is common practice. An example of this language is found in the Amazon Customer Web Services agreement, which states, "WE… MAKE NO REPRESENTATIONS OR WARRANTIES OF ANY KIND … THAT THE SERVICE OFFERINGS OR THIRD PARTY CONTENT WILL BE UNINTERRUPTED, ERROR FREE OR FREE OF HARMFUL COMPONENTS, OR THAT ANY CONTENT … WILL BE SECURE OR NOT OTHERWISE LOST OR DAMAGED." In fact this agreement even goes as far as to suggest that a customer make "frequent archives" of their data. As mentioned before, the responsibility for managing the integrity of data, whether in a data center, private cloud, hybrid cloud or public cloud always falls on the company that owns the data.

There are some common sense best methods that will allow a company to take benefit of the springiness and approachability of the cloud, without putting its data at risk. The principle of data protection is to distribute the risk so that the likelihood of data loss is reduced. Even when storing data in the cloud, it makes sense to keep a main copy and a backup copy of the data onsite so that access to the data is not dependent upon network performance or connectivity. By following these basic best methods and knowing the details of the cloud provider's SLA, the building blocks are in place to implement a method for proactively observing the integrity of data regardless of the storage platform or location.

It's hard to argument that the cloud industry has taken a few punches in the media recently, especially with large vendors like Iron Mountain withdrawing their basic cloud storage services and the previously discussed data loss at Amazon S3. However, the moral of this story is not that the cloud is an risky storage platform, but rather that when examining and employing cloud strategies, there are more factors to consider than simply cost per gigabyte stored. Cloud storage offers many advantages to companies of any size when properly employed. What cloud doesn't do is eliminate the need for intelligent data management strategies. Regardless of how or where data is stored, it is absolutely crucial to make certain it will be accessible and restorable when needed. This assurance is at the very heart of data integrity monitoring and verification.

Many algorithms have been proposed to verify whether the data present in the cloud has been modified or not and they showed best results for the static data.

A. Hash Function: The simplest Proof of retrievability (POR) scheme can be made using a keyed hash function hk(F). In this scheme the verifier, before archiving the data file F in the cloud storage, pre-computes the cryptographic hash of F using hk(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 hk(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 resource 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). At 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.

B. Proof of Retrievability: Ari Juels and Burton S. Kaliski Jr [4] proposed a scheme called Proof of retrievability for large files using "sentinels". 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 the fig 1

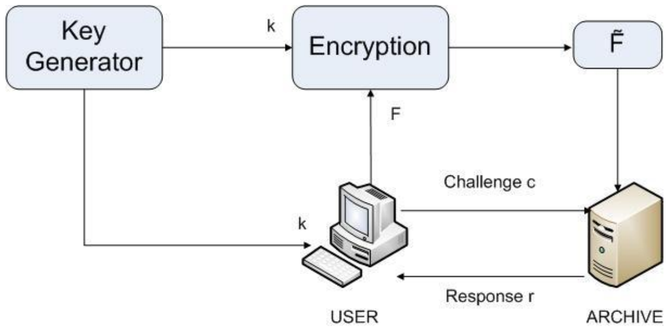

Fig. 1 Schematic view of a proof of retrievability based on inserting random sentinels in the data file

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.).

C. Data integrity Proofs in cloud storage: R Sravan Kumar and Ashuthosh Saxena [5] of Infosys Technologies Ltd, Hyderabad presented a scheme which does not involve the encryption of the whole data. We encrypt only few bits of data per data block thus reducing the computational overhead on the clients as shown in fig 2. The client storage overhead is also minimized as it does not store any data with it. Hence our scheme suits well for thin clients. In this data integrity protocol the verifier needs to store only a single cryptographic key - irrespective of the size of the data file F- and two functions which generate a random sequence. The verifier does not store any data with it. The verifier before storing the file at the archive, preprocesses the file and appends some meta data to the file and stores at the archive as shown in fig 3. At the time of verification the 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 the 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 other schemes like redundant storing etc, can be implemented.

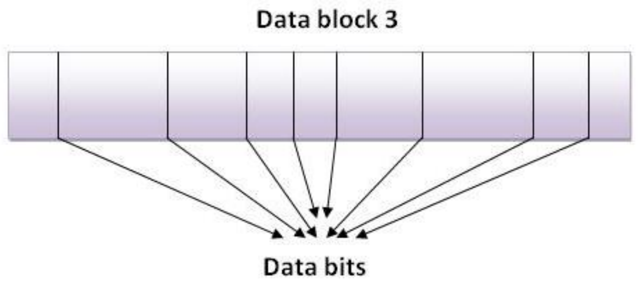

Fig. 2 A data block of the file F with random bits selected in it

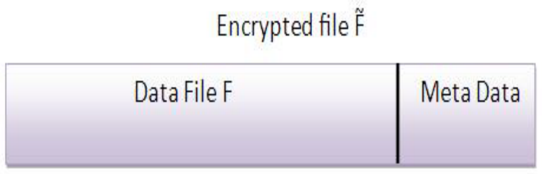

Fig.3 The encrypted file F which will be stored in the cloud.

It should be noted that this scheme applies only to static storage of data. It cannot handle to case when the data need to be dynamically changed. Hence developing on this will be a future challenge. Also the number of queries that can be asked by the client is fixed apriori. But this number is quite large and can be sufficient if the period of data storage is short.

D. Data integrity Proofs in cloud storage: In this paper, Wenjun Luo and Guojing Bai [7] have proposed a remote data possession checking protocol with the support public verifiability. They used HLAs and RSA construction to complete the protocol. The support of public verifiability makes the protocol very flexible, since the user can commit the data possession to check the TPA. And since the protocol based on the RSA problem with large public exponent, so the security of the data storage is enhanced.

This scheme is consisted of four algorithms
KeyGen SigGen GenProof and Verify Proof.
KeyGen : a key generation algorithm that is run by the user to setup the scheme.
SigGen : used by the user to generate verification metadata, which may consist of MAC, signatures, or other related information that will be used for auditing.
GenProof: run by the cloud server to generate a proof of data storage correctness.
Verify Proof: run by the TPA to verify the proof from the cloud server.

Our public verify system can be constructed from the above auditing scheme in two phases:

Setup: The user initializes the public and secret parameters of the system by executing KeyGen, and preprocesses the data file F by using SigGen to generate the verification metadata. The user then stores the data file F at the cloud server, then deletes its local copy, and publishes the verification metadata to TPA for later audit. As part of pre-processing, the user may alter the data file F by expanding it or including additional metadata to be stored at server.

Audit: The TPA issues an audit message or challenge to the cloud server to make sure that the cloud server has retained the data file F properly at the time of the audit. The cloud server will derive a response message from a function of the stored data file F by executing GenProof. Using the verification metadata, the TPA verifies the response via Verify Proof .

IV. PROPOSED SOLUTION

The proposed algorithm is used to verify whether the user data in the cloud is altered by some unauthorized party or not.

Here, initially the client will generate the checksum for the data to be placed in the cloud and stores the checksum in his local site and upload the data into the cloud. Then, he will use the stored checksum to verify the data integrity for the data being placed in the cloud.

The algorithm consists of two phases whose working is as follows.

(Phase 1): Generating the Checksum

The Generation of checksum is explained pictorially through activity diagram in fig(5)
Step 1: Generate a 128 bit random vector consisting of the positions in the data (i.e, the locations of the data) (Call it as A).
Step 2: Read the data from the cloud, based on the locations present in the random vector. Let the read data be called as ‘B’.
Step 3: Generate a 128 bit key (Random Number) and call it as ‘C’.
Step 4: Perform X-OR operation between B and C.
Step 5: The result of step 4 is called as the valuator, say I (First Level Encryption).
Step 6: The valuator is now again encrypting using any of the standard encryption algorithms(both conventional and public key cryptosystems).
(Second Level Encryption).
Step 7: The result of step 6 is an encrypted value called as checksum.
Step 8: Place the checksum (Result of Step 7) along with the Data in the Cloud (Optional).
Step 9: The Random Vector ‘A’ and the 128 bit key should be kept secret, so as to protect from the attacks and used for verification of the data later by the user.

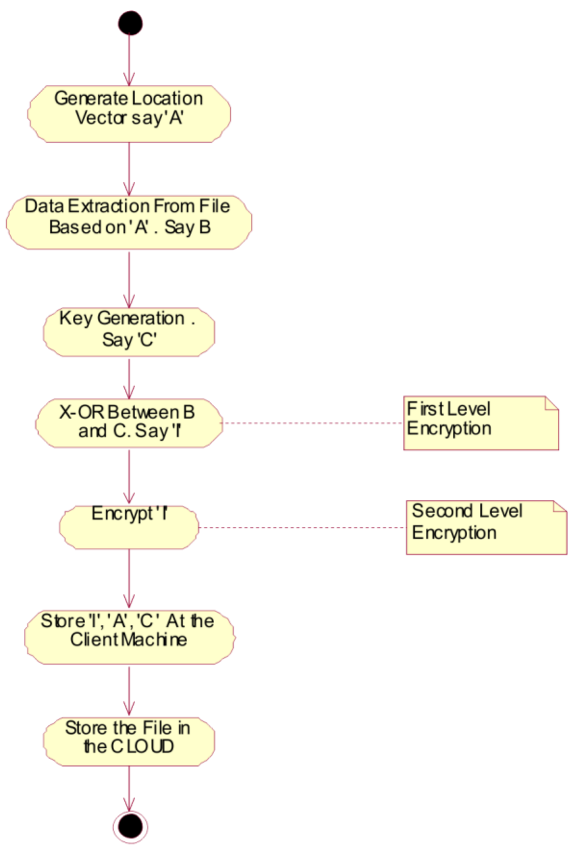
Generate Location Vector say 'A'
Data Extraction From File Based on 'A'. Say B
Key Generation . Say 'C'
X-OR Between B and C. Say 'I'
First Level Encryption
Encrypt 'I'
Second Level Encryption
Store 'I', 'A', 'C' At the Client Machine
Store the File in the CLOUD

Fig. 5 Activity Diagram describing the generation of Checksum.

(Phase 2): Verifying the Checksum

Note: Here the data only is placed in the cloud and the checksum, the 128 bit key(‘C’) and the location vector (‘A’) are kept with the user.

The verification of checksum is explained pictorially through activity diagram in fig(6)

Step 1: The user will read the 128-bit data from the cloud based on the location vector (‘A’) and names it as ‘B’.
Step 2: The generated ‘B’ is X-OR ed with the already stored 128 bit key (‘C’) at the user.
(First Level Encryption)
Step 3: The resultant of step 2 is now again encrypted using the same standard encryption algorithm that was used at the time of generating the checksum.
(Second Level Encryption)
Step 4: The result of step 3 is compared with the already stored checksum at the user. If they are matched, then Data Integrity is not lost. If they are not matched then Data integrity is lost.

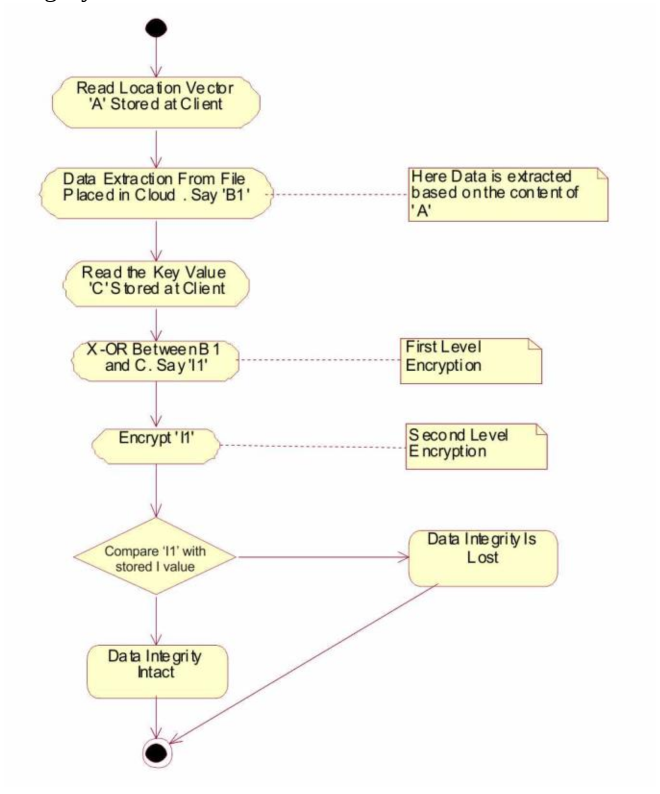
Read Location Vector 'A' Stored at Client
Data Extraction From File Placed in Cloud . Say 'B1'
Here Data is extracted based on the content of 'A'
Read the Key Value 'C' Stored at Client
X-OR Between B1 and C. Say 'I1'
First Level Encryption
Encrypt 'I1'
Second Level Encryption
Compare 'I1' with stored I value
Data Integrity Is Lost
Data Integrity Intact

Fig.6 Activity Diagram Describing The Verification and Checking Data Integrity

Advantages:
1. Simple technique (Old wine in new bottle)
2. Easy to implement.
3. Mobile devices / Thin Clients can also implement this algorithm in a very easy and efficient way.
4. Easy to apply for dynamic changing of data.

V. COMPARATIVE STATEMENT

Below are the details about the performance of the algorithm using two Conventional Encryption Algorithms and one public key cryptosystem algorithm ‘ECC’.

Here a Data of Size 40 KB is considered (4096 Bytes) and following ‘aprior’ time is recorded.

Operation	AES	Blowfish	ECC
CHECKSUM Generation (Time in Milli Seconds)	**3853**	**3915**	**3806**
CHECKSUM Verification (Time in Milli Seconds)	**3838**	**3900**	**3853**

Table 1 .Comparative study of three algorithms.

VI. CONCLUSION

Due to the increasing demand in Cloud Computing (Providing Database as a service), many companies like IBM, Amazon, Google etc are moving to Data Storage in Cloud. The concerned factor about cloud is the security issue. This paper deals with the proof of checking whether data is correct or not in the cloud. The proposed solution uses the checking of correctness of data in cloud for dynamic change and easily accessible by any mobile device. The technique has two levels of Generation and Verification Process
This proposed solution is easily accessible by any Mobile device in an efficient manner, If the attacker corrupts the data, then sufficiently/efficiently he cannot tamper the checksum. The probability of corrupting the checksum accordingly is very very high.

VII. REFERENCES

[1] Enhancing Security through Steganography by using Sudoku puzzle and ECC Algorithm by B.Chitra, Depavath Harinath, M.V. Ramana Murthy, K. Ramesh Babu, IJRSET, ISSN 2394-739X
[2] Encryption techniques for Big Data in a cloud by Depavath Harinath, K. Ramesh Babu, Mrs. B.Chitra, M.V. Ramana Murthy, IJMTER, ISSN (ONLINE) : 2349-9745
[3]Won Kim, “Cloud Computing: Today and Tomorrow”, Journal of Object Technology, January/February 2009.
[4] 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.
[5] Shravan Kumar Rana and AshutoshSaxena, “Data Integrity Proofs in Cloud Storage” in IEEE 2011 Paper.
[6] Jeffrey Naruchitparames, Mehmet HadiGunes¸ “Enhancing Data Privacy and integrity in the cloud” of IEEE 2011 Paper