PROFICIENT PUBLIC SUBSTANTIATION OF DATA VERACITY FOR CLOUD STORAGE THROUGH DUAL PROTECTION

S. Gowmithra[1], Dr. N. Velvizhi[2]
[1]Computer Science and Engineering, Indira Institute of Engineering and Technology,Chennai, India
[2]Principal & Professor, Computer Science and Engineering, Indira Institute of Engineering and Technology,Chennai,India

Abstract

The data privacy and service availability in cloud computing are the key security issue. In this project we preserve our data in the cloud storage and would make available for multiple users upon various effective verifications done by auditor and the data stored in the cloud can be updated dynamically. When a data is uploaded to the server the original data splits into three parts and stored in three different locations on the server using Erasure Code algorithm, a unique key code is generated to each split data for auditing. After key generation all the three parts of the single file will be encrypted by AES (Advanced Encryption Standard) algorithm. Each users are given some level of access as per the requirement (View , Read only , Read write) To download the stored file Owner authorization is essential. At the time of download key generated (code based key generation) will be sent to file owner , the encrypted data will be decrypted and spliced from three dissimilar locations, If anyone tries to hack the data at cloud end it is not possible to break the two different blocks as the security scheme of our system is Strongest and this system allows the clients to check and monitor whether their outsourced data is kept intact without downloading the whole data using own auditing based Token generation the system ensures more security , when a data is edited or modified the anonymous change will be made in the original key value, by this client can easily find out the data is being changed or modified.

INTRODUCTION

Cloud computing has been imagined as the following creation data innovation design for undertakings, because of its extensive rundown of unparalleled preferences in the IT history: on-request self-benefit, omnipresent system get to, area self-deciding asset pooling, fast asset versatility, utilization based estimating and transference of hazard.

As a disturbing innovation with significant ramifications, cloud computing is changing the very way of how organizations utilize data innovation. One essential part of this outlook changing is that information are being brought together or outsourced. From clients' view, including together people and IT endeavors, putting away information remotely to the in an adaptable on-request technique bring appealing advantages: arrival of the weight for storage room administration, boundless information access with place autonomy, and evasion of assets expenses on equipment, programming, and staff systems of support, and so on

While cloud computing make these remuneration more engaging than any other time in recent memory, it additionally conveys new and testing security dangers to clients' outsourced information. As administration suppliers (CSP) are part regulatory elements, information outsourcing is really surrendering client's last control more than the destiny of their information.

A. Scope Of The Project

As high-speed networks and ubiquitous Internet access become available in recent years, many services are provided on the Internet such that users can use them from anywhere at any time. Data robustness is a major requirement for storage systems. There have been many proposals of storing data over.

B. Need For The Project

The template is used to format your paper and style the text. All margins, column widths, line spaces, and text fonts are prescribed; please do not alter them. You may note peculiarities. For example, the head margin in this template measures proportionately more than is customary. This measurement and others are deliberate, using specifications that anticipate your paper as one part of the entire proceedings, and not as an independent document. Please do not revise any of the current designations.

C. Objective Of The Project

Service providers (CSP) are separate administrative entities, data outsourcing is actually relinquishing user's ultimate control over the fate of their data. As a result, the correctness of the data in that is being put at risk due to the following reasons. First of all, although the infrastructures under that are much more powerful and reliable than personal computing devices, they are still facing the broad range of both internal and external threats for data integrity.

D. Abbreviations and Acronyms

Define abbreviations and acronyms the first time they are used in the text, even after they have been defined in the abstract. Abbreviations such as IEEE, DML, J2EE, JSP, JVM, RDBMS, SQL, PCS, CSP, FMS, TPA, PDP, QOP, QOE, PKI, EC, AES, DES and OTP are used

II. CONTENT OF THE THESIS

1) User Interface

In our Secure System we have a user friendly user interface to interact with our System. Every Act dual role as a data owner and data consumer while uploading file they are the owner of that file if they search other's file than they are the consumer. Users can create the account them self for that we have new pages, in that page we will get the details from the user and we generate the account for the user's. We have authentication system; we only allow authorized users to access our System.

In our System we providing the easy file searching user's don't want to keep remember all uploaded file's exact name, for that we have given the keywords while uploading the files it will help to search the file easily.

2) File uploading process:

Storing data over storage servers one way to provide data robustness is to replicate a message such that each storage server stores a message. Another way is to encode a message of k symbols into a code word of n symbols by erasure coding. To store a message, each of its code word symbols is stored in a different storage server. A storage server corresponds to an erasure error of the code word symbol. As long as the number of servers is under the tolerance threshold of the erasure code, the message can be recovered from the code word symbols stored in the available storage servers by the decoding process.

3) Secret key generation:

Firstly the secret key will be generated as the initial step while uploading the file, every which is uploaded, will have unique secret key. This key will be taken as an identification of every file. The secret key which we are using is a three digit number we will make it use for both uploading and downloading. If the user want download some file and if he gives the download request the secret key of that file will be sent to the file owner of the file maybe he can share it.

4) File Sharing:

In our application we can share a file to a registered user by providing basic credentials, with the sharing option it is necessary to provide authority to the shared user whether to view or edit the file. A user can view the shared file within the application without downloading it and the same is possible with the edit option.

5) File Auditing:

Auditing is the process of checking the file whether the original contents of the file is changed. This module provides the file owner auditing, this we achieve by generating tokens. The tokens are generated with the ASCII values

of the characters in the file and these characters are stored in the DB while uploading the file. If a shared user edit's the file and saves it, again a new token will be generated and stored in the DB. If the initial token and the current token aren't same then a notification will be sent to the file owner.

6) Mail alert process:

The uploading and downloading process of the user is first get the secret key in the corresponding user email id and then apply the secret key to encrypted data to send the server storage and decrypts it by using his secret key to download the corresponding data file in the server storage system's the secret key conversion using the Share Key Gen (SKA, t, m). This algorithm shares the secret key SKA of a user to a set of key servers.

7) File Downloading process:

File downloading process is to get the corresponding secret key to the corresponding file to the user mail id and then decrypt the file data. The file downloading process re-encryption key to storage servers such that storage servers perform the re-encryption Operation. The length of forwarded message and the computation of re-encryption is taken care of by storage servers. Proxy re-encryption Schemes significantly reduce the overhead of the data Forwarding function in a secure storage system.

III. TECHNOLOGY IMPLEMENTATION

Erasure Coding : Erasure coding (EC) is a method of data protection in which data is broken into fragments, expanded and encoded with redundant data pieces and stored across a set of different locations or storage media

Erasure coding creates a mathematical function to describe a set of numbers so they can be checked for accuracy and recovered if one is lost. Referred to as polynomial interpolation or oversampling, this is the key concept behind erasure codes. In mathematical terms, the protection offered by erasure coding can be represented in simple form by the following equation: n = k + m. The variable "k" is the original amount of data or symbols. The variable "m" stands for the extra or redundant symbols that are added to provide protection from failures. The variable "n" is the total number of symbols created after the erasure coding process. For instance, in a 10 of 16 configurations, or EC 10/16, six extra symbols (m) would be added to the 10 base symbols (k). The 16 data fragments (n) would be spread across 16 drives, nodes or geographic locations. The original file could be reconstructed from 10 verified fragments.

In coding theory, an erasure code is a forward error correction (FEC) code under the assumption of bit erasures (rather than bit errors), which transforms a message of k symbols into a longer message (code word) with n symbols such that the original message can be recovered from a subset of the n symbols.

Nutanix has recently introduced in the latest tech preview releases, something calledErasure coding (EC-X) which will help give capacity back. loss of data, (RF3 allows two multiple node failures within a cluster) this does however mean that storage is impacted and on the simplest calculation

Example :

MDS stands for maximum distance separable. An MDS erasure code is generally represented as (n, k).

The basic premise of erasure coding goes as follows:

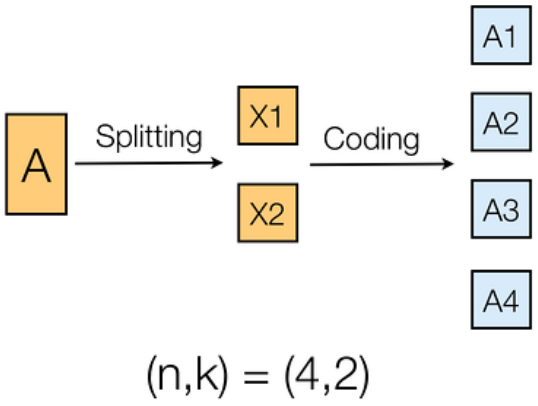
Take a file and split into k pieces and encode into n pieces. Now, any k pieces can be used to get back the file

So here is the recipe:

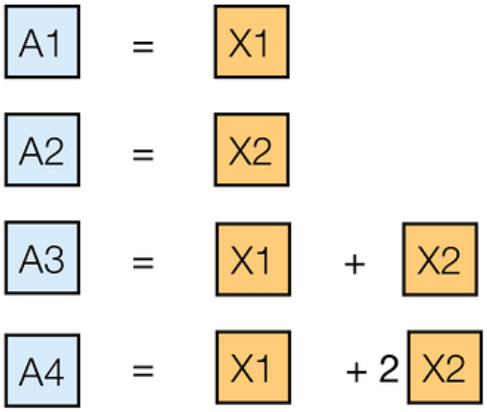
- Take a file of size M.
- Split the file into k chunks, each of the same size M/k.
- Now, apply the (n, k) code on these k chunks to get n chunks, each of the same size M/k.
- Now the effective size is nM/k. Thus the file is expanded n/k times. We need n to be greater than or equal to k, so n/k is at least 1. If n equals k, you have just split the file and there is no coding performed.
- Any k chunks out of the n chunks can be used to get back the file.

So this also means that the code can tolerate upto (n - k) erasures. The following figure

shows this recipe being followed for a (4, 2) code.



Without really wondering how do actually add files, the following example illustrates one particular case of designing a (4, 2) code.



.

Erasure codes were first designed to assist in detecting and correcting "problems" when sending data (through an unreliable channel). One of the famous examples of erasure codes are Reed Solomon codes. While erasure codes are also called as error correcting codes, there is a crucial difference between an error and an erasure. If I send ten bits and one bit flipped, an error has occurred, and I do not know where it has occurred. However, if I store ten blocks of a file into different nodes and one node dies, I know exactly which block I lost, and so I know where the erasure has happened. See the difference?

So thats all you need to know about erasure coding if you want to get started. Hopefully in a separate post, I will introduce other tools such as Jerasure, which can be used to implement erasure codes

Advanced Encryption Standard: The more popular and widely adopted symmetric encryption algorithm likely to be encountered nowadays is the Advanced Encryption Standard. It is found at least six times faster than triple DES. A replacement for DES was needed as its key size was too small. With increasing computing power, it was considered vulnerable against exhaustive key search attack. Triple DES was designed to overcome this drawback but it was found slow.

a) The Features Of Aes

- Symmetric key symmetric block cipher
- 128-bit data, 128/192/256-bit keys
- Stronger and faster than Triple-DES
- Provide full specification and design details

b) Operation Of Aes

AES is an iterative rather than Feistel cipher. It is based on 'substitution–permutation network'. It comprises of a series of linked operations, some of which involve replacing inputs by specific outputs (substitutions) and others involve shuffling bits around (permutations). Interestingly, AES performs all its computations on bytes rather than bits. Hence, AES treats the 128 bits of a plaintext block as 16 bytes. These 16 bytes are arranged in four columns and four rows for processing as a matrix

Unlike DES, the number of rounds in AES is variable and depends on the length of the key. AES uses 10 rounds for 128-bit keys, 12 rounds for 192-bit keys and 14 rounds for 256-bit keys. Each of these rounds uses a different 128-bit round key, which is calculated from the original AES key.

A block cipher processes the data blocks of fixed size. Usually, the size of a message is larger than the block size. Hence, the long message is divided into a series of sequential message blocks, and the cipher operates on these blocks one at a time.

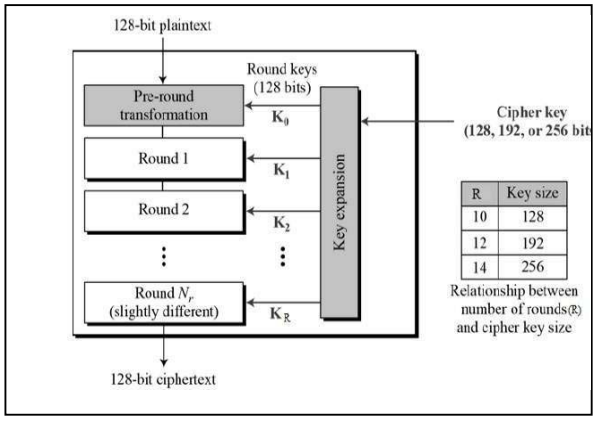


Figure 1. The Schematic of AES Structure

c) Encryption Process:

Here, we restrict to description of a typical round of AES encryption. Each round comprise of four sub-processes. The first round process is depicted below

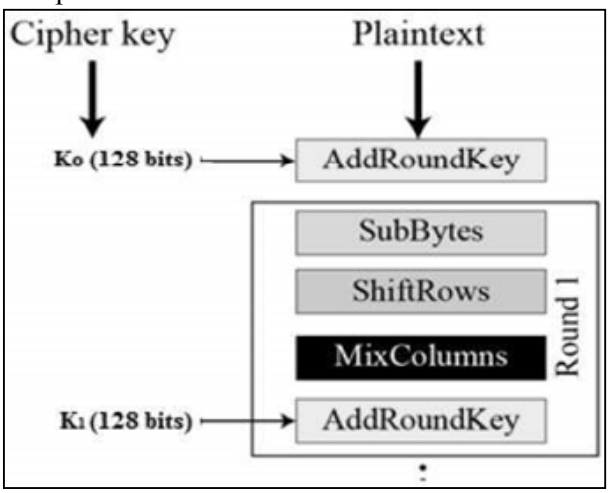


Figure 2. Encryption Process

d) Byte Substitution

The 16 input bytes are substituted by looking up a fixed table (S-box) given in design. The result is in a matrix of four rows and four columns.

e) Shift rows

Each of the four rows of the matrix is shifted to the left. Any entries that 'fall off' are re-inserted on the right side of row. Shift is carried out as follows −

- First row is not shifted.
- Second row is shifted one (byte) position to the left.
- Third row is shifted two positions to the left.
- Fourth row is shifted three positions to the left.
- The result is a new matrix consisting of the same 16 bytes but shifted with respect to each other.

f) Mixcolumns

Each column of four bytes is now transformed using a special mathematical function. This function takes as input the four bytes of one column and outputs four completely new bytes, which replace the original column. The result is another new matrix consisting of 16 new bytes. It should be noted that this step is not performed in the last round.

g) Addroundkey

The 16 bytes of the matrix are now considered as 128 bits and are XORed to the 128 bits of the round key. If this is the last round then the output is the ciphertext. Otherwise, the resulting 128 bits are interpreted as 16 bytes and we begin another similar round.

h) Decryption Process

The process of decryption of an AES ciphertext is similar to the encryption process in the reverse order. Each round consists of the four processes conducted in the reverse order −

- Add round key
- Mix columns
- Shift rows
- Byte substitution

Since sub-processes in each round are in reverse manner, unlike for a Feistel Cipher, the encryption and decryption algorithms needs to be separately implemented, although they are very closely related.

i) Aes Analysis

In present day cryptography, AES is widely adopted and supported in both hardware and software. Till date, no practical cryptanalytic attacks against AES has been discovered. Additionally, AES has built-in flexibility of key length, which allows a degree of 'future-proofing' against progress in the ability to perform exhaustive key searches

IV. PERFORMANCE OF EVALUATION AND RESULTS

A Successful uploading of file to the cloud by splitting it into equal three halves based on its size is done. The files are splits using the erasure code method and successfully encrypted and uploaded to the cloud. The dual protection of data stored over the cloud will be preserved more efficiently than the existing system. The

performance of the existing and proposed system is depicted through the histogram chart.

V. CONCLUSION

A privacy-preserving public auditing system for data storage security in computing. We utilize the homomorphism linear authenticator and random masking to guarantee that the TPA would not learn any knowledge about the data content stored on the server during the efficient auditing process, which not only eliminates the burden of user from the tedious and possibly expensive auditing task, but also alleviates the users' fear of their outsourced data leakage. Considering TPA may concurrently handle multiple audit sessions from different users for their outsourced data files,

We further extend our privacy-preserving public auditing protocol into a multiuser setting, where the TPA can perform data stored over the cloud will be preserved more efficiently than the existing system. The performance of the existing

REFERENCES

The template will number citations consecutively within brackets [1]. The sentence punctuation follows the bracket [2]. Refer simply to the reference number, as in [3]—do not use "Ref. [3]" or "reference [3]" except at the beginning of a sentence: "Reference [3] was the first . . ."

Number footnotes separately in superscripts. Place the actual footnote at the bottom of the column in which it was cited. Do not put footnotes in the reference list. Use letters for table footnotes.

Unless there are six authors or more give all authors' names; do not use "et al.". Papers that have not been published, even if they have been submitted for publication, should be cited as "unpublished" [4]. Papers that have been accepted for publication should be cited as "in press" [5]. Capitalize only the first word in a paper title, except for proper nouns and element symbols.

[1] Y. Cui, Z. Lai, X. Wang, N. Dai, and C. Miao, "Quicksync: Improving synchronization efficiency for mobile storage services," in Proceedings of MobiCom. ACM, 2015, pp. 592–603.

[2] H. Li, Y. Yang, T. H. Luan, X. Liang, L. Zhou, and X. Shen, "Enabling fine-grained multi-keyword search supporting classified sub-dictionaries over encrypted data," IEEE Transactions on Dependable and Secure Computing, vol. 13, no. 3, pp. 312–325, 2016.

[3] Q. Wang, C. Wang, K. Ren, W. Lou, and J. Li, "Enabling public auditability and data dynamics for storage security in computing," IEEE Transactions on Parallel and Cloud Systems, vol. 22, no. 5, pp. 847–859, 2011.

[4] M. Sookhak, A. Gani, H. Talebian, A. Akhunzada, S. U. Khan, R. Buyya, and A. Y. Zomaya, "Remote data auditing in computing environments: A survey, taxonomy, and open issues," ACM Computing Surveys, vol. 47, no. 4, 2015.

[5] G. Ateniese, R. Burns, R. Curtmola, J. Herring, L. Kissner, Z. Peterson, and D. Song, "Provable data possession at untrusted stores," in Proceedings of CCS. ACM, 2007, pp. 598–609.

[6] H. Li, D. Liu, Y. Dai, and T. H. Luan, "Engineering searchable encryption of mobile networks: When qoe meets qop," IEEE Wireless Communications, vol. 22, no. 4, pp. 74–80, 2015.

[7] K. Yang and X. Jia, "An efficient and secure dynamic auditing protocol for data storage in computing," IEEE Transactions on Parallel and Cloud Systems, vol. 24, no. 9, pp. 1717–1726, 2013.

[8] C. Wang, Q. Wang, K. Ren, and W. Lou, "Privacy-preserving public auditing for data storage security in computing," in Proceedings of INFOCOM. IEEE, 2010, pp. 19.

[9] G. Ateniese, R. D. Pietro, L. V. Mancini, and G. Tsudik, "Scalable and efficient provable data possession," in Proceedings of SecureComm. ACM, 2008.

[10] S. Garg, C. Gentry, S. Halevi, M. Raykova, A. Sahai, and B. Waters, "Hiding secrets in software: A cryptographic approach to program obfuscation," Communications of The ACM, vol. 59, no. 5, pp. 113– 120, 2016.