Secure Data Storage And Sharing Techniques For Data Protection In Cloud Environments: A Systematic Review, Analysis, And Future Directions

Parthiban.s[1], Karthi.p[2], Sanjay kumar.e[3], Ragul.k[4]
[1]Assistant Professor, Dept of Computer Science
[2, 3, 4]Dept of Computer Science
[1, 2, 3, 4] Computer Science, Mahendra Institute of Engineering and Technology, Tamil Nadu, Namakkal DT – 637 503

Abstract- *High-speed networks and ubiquitous Internet access become available to users for access anywhere at any time. Cloud computing is a concept that treats the resources on the Internet as a unified entity, a cloud. Cloud storage is a model of networked online storage where datais stored in virtualized pools of storage which are generally hosted by third parties. Hosting companies operate large data centers, and people who require their data to be hosted buy or lease storage capacity from them. The data center operators, in the background, virtualize the resources according to the requirements of the customer and expose them as storage pools, which the customers can themselves use to store files or data objects. Physically, the resource may span across multiple servers.*

Data robustness is a major requirement for storage systems. There have been many proposals of storing data over storage servers. One way to provide data robustness is to replicate a message such that each storage server stores a copy of the message. A decentralized erasure code is suitable for use in a distributed storage system.

We construct a secure cloud storage system that supports the function of secure data forwarding by using an AES and Proxy re encryption. In this model initial phase owner will upload the data with AES Encryption. Next phase, inside of cloud again the data has divided into small pieces, for this process we will apply a dividing key. Data will place in different storage lactations. The information of data storage will monitor by a unique data distributors. If the valid user accessing the data cloud will retrieve the data as reversible manner..

I. INTRODUCTION

Objective

We propose a novel method called secured erasure code based algorithm for cloud data security in distributed storage system. Rapid organizations and universal Internet access become accessible to clients for access anyplace whenever. Distributed computing is an idea that treats the assets on the Internet as a brought together element, a cloud. Distributed storage is a model of arranged online stockpiling where information is put away in virtualized pools of capacity which are by and large facilitated by third gatherings. Facilitating organizations work huge server farms, and individuals who require their information to be facilitated purchase or rent stockpiling limit from them. The server farm administrators, behind the scenes, virtualize the assets as indicated by the necessities of the client and uncover them as capacity pools, which the clients would themselves be able to use to store records or information objects. Genuinely, the asset may range across different workers. Information vigor is a significant necessity for capacity frameworks. There have been numerous recommendations of putting away information over capacity workers. One approach to give information power is torecreate a message to such an extent that every capacity worker stores a duplicate of the message.A decentralized eradication code is reasonable for use in a circulated stockpiling framework.

We build a protected distributed storage framework that upholds the capacity of secure information sending by utilizing an AES and Proxy re encryption. In this model beginning stage proprietor will transfer the information with AES Encryption. Next stage, within cloud again the information has separated into little pieces, for this cycle we will apply a partitioningkey. Information will put in various capacity lactations. The data of information stockpiling will screen by an extraordinary information wholesalers. On the off chance that the legitimate client getting to the information cloud will recover the information as reversible way.

II. LITERATURE SURVEY

QoS Support for End Users of I/O-intensive Applications Using Shared Storage Systems.

We consider the problem of constructing an erasure code for storage over a network when the data sources are distributed. Specifically, we assume that there are n storage nodes with limited memory and k < n sources generating the data. We want a data collector, who can appear anywhere in the network, to query any k storage nodes and be able to retrieve the data. We introduce Decentralized Erasure Codes, which are linear codes with a specific randomized structure inspired by network coding on random bipartite graphs. We show that decentralized erasure codes are optimally sparse, and lead to reduced communication, storage and computation cost over random linear coding.

Repair Locality from a Combinatorial Perspective.

Plutus is a cryptographic storage system that enables secure file sharing without placing much trust on the file servers. In particular, it makes novel use of cryptographic primitives to protect and share files. Plutus features highly scalable key management while allowing individual users to retain direct control over who gets access to their files. We explain the mechanisms in Plutusto reduce the number of cryptographic keys exchanged between users by using file groups, distinguish file read and write access, handle user revocation efficiently, and allow an untrusted server to authorize file writes. We have built a prototype of Plutus on OpenAFS. Measurementsof this prototype show that Plutus achieves strong security with overhead comparable tosystems that encrypt all network traffic.

On the Effective Parallel Programming of Multi-core Processors.

Availability is a storage system property that is both highly desired and yet minimally engineered. While many systems provide mechanisms to improve availability– such as redundancy and failure recovery – how to best configure these mechanisms is typically left to the system manager. Unfortunately, few individuals have the skills to properly manage the trade-offs involved, let alone the time to adapt these decisions to changing conditions. Instead, most systems are configured statically and with only a cursory understanding of how the configuration will impact overall performance or availability. While this issue can be problematic even for individual storage arrays, it becomes increasingly important as systems are distributed – and absolutely critical for the wide area peer-to-peer storage infrastructures being explored. This paper describes the motivation, architecture and implementation for a newpeer-to-peer storage system, called Total Recall that automates the task of availability management. In particular, the Total Recall system automatically measures and estimates the availability of its constituent host components, predicts their future availability based on past behavior, calculates the appropriate redundancy mechanisms and repair policies, and delivers user-specified availability while maximizing efficiency.

Parallel Reed/Solomon Coding on Multicore Processors.

This paper sketches the design of PAST, a large-scale, Internet-based, global storage utility that provides scalability, high availability, persistence and security. PAST is a peer-to-peer Internet application and is entirely selforgaining. PAST nodes serve as access points for clients, participate in the routing of client requests, and contribute storage to the system. Nodes are not trusted, they may join the system at any time and may silently leave the system without warning. Yet, the system is able to provide strong assurances, efficient storage access, load balancing and scalability.

Privacy-preserving and Secure Distributed Storage Codes

Storage outsourcing is a rising trend which prompts a number of interesting security issues, many of which have been extensively investigated in the past. However, Provable Data Possession (PDP) is a topic that has only recently appeared in the research literature. The main issue is how to frequently, efficiently and securely verify that a storage server is faithfully storing its client's(potentially very large) outsourced data. The storage server is assumed to be untrusted in terms of both security and reliability. (In other words, it might maliciously or accidentally erase hosted data; it might also relegate it to slow or off-line storage.) The problem is exacerbated by the client being a small computing device with limited resources. Prior work has addressed this problem using either public key cryptography or requiring the client to outsource its data in encrypted form. In this paper, we construct a highly efficient and provably secure PDP technique based entirely on symmetric key cryptography, while not requiring any bulk encryption.

III. SYSTEM ANALYSIS

EXISTING SYSTEM

In Existing System we use a straightforward integration method. In straightforward integration method Storing data in a third party's cloud system causes serious concern on data confidentiality. In order to provide strong confidentiality for messages in storage servers, a user can encrypt messages by a cryptographic method before applying an erasure code method to encode and store messages. When he wants to use a message, he needs to retrieve the Codeword symbols from storage servers, decode them, and then decrypt them by using cryptographic keys. General encryption schemes protect data confidentiality, but also limit the functionality of the storage system because a few operations are supported over encrypted data. A decentralized architecture for storage systems offers good scalability, because a storage server can join or leave without control of a central authority.

.

DRAWBACKS:

1. The user can perform more computation and communication traffic between the user and storage servers is high.
2. The user has to manage his cryptographic keys otherwise the security has to be broken.
3. The data storing and retrieving, it is hard for storage servers to directly support other functions

PROPOSED SYSTEM

In our proposed system we address the problem of forwarding data to another user by storage servers directly under the command of the data owner. We consider the system model that consists of distributed storage servers and key servers. Since storing cryptographic keys ina single device is risky, a user distributes his cryptographic key to key servers that shall perform cryptographic functions on behalf of the user.

These key servers are highly protected by security mechanisms.Here Storage system has allocates by different data container. Once owner uploads the data with AES encryption mechanism, system again takes the data and makes Secure Data segregation process. All the data pieces will be save in different location in cloud storage. Here public distributor monitors all the data and corresponding positions where it is saved. When a proper client asking the data, cloud system will provide the data in reversible manner. So our system will prevent our data from both Inside and Outside attackers.

ADVANTAGES

1. Tight integration of encoding, encryption, and forwarding makes the storage system efficiently meet the requirements of data robustness, data confidentiality, and data forwarding.
2. The storage servers independently perform encoding and re-encryption process and the key servers independently perform partial decryption process.
3. More flexible adjustment between the number of storage servers and robustness.

IV. SOFTWARE ENVIRONMENTS

SOFTWARE DESCRIPTION

FRONT END

Java The JAVA language was created by James Gosling in June 1991 for use in a set top box project. The language was initially called Oak, after an oak tree that stood outside Gosling's office - and also went by the name Green - and ended up later being renamed to Java, from a list of random words. Gosling's goals were to implement a virtual machine and a language that had a familiar C/C++ style of notation. The first public implementation was Java 1.0 in 1995. It promised "Write Once, Run Anywhere” (WORA), providing no-cost runtimes on popular platforms. It was fairly secure and its security was configurable, allowing network and file access to be restricted. Major web browsers soon incorporated the ability to run secure Java applets within web pages. Java quickly became popular. With the advent of Java 2, new versions had multiple configurations built for different types of platforms. For example, J2EE was for enterprise applications and the greatly stripped down version J2ME was for mobile applications. J2SE was the designation for the Standard Edition. In 2006, for marketing purposes, new J2 versions were renamed Java EE, Java ME, and Java SE, respectively. In 1997, Sun Microsystems approached the ISO/IEC JTC1 standards body and later the Ecma International to formalize Java, but it soon withdrew from the process. Java remains a standard that is controlled through the Java Community Process. At one time, Sun made most of its Java implementations available without charge although they were proprietary software. Sun's revenue from Java was generated by the selling of licenses for specialized products such as the Java Enterprise System. Sun distinguishes between its Software Development Kit (SDK) and Runtime Environment (JRE)which is a subset of the SDK, the primary distinction being that in the JRE, the compiler, utility programs, and many necessary header files are not present.

Primary Goals:

There were five primary goals in the creation of the Java language:

- It should use the object-oriented programming methodology.
- It should allow the same program to be executed on multiple operating systems.
- It should contain built-in support for using computer networks.

- It should be designed to execute code from remote sources securely.
- It should be easy to use by selecting what were considered the good parts other object-oriented languages.

V. MODULE DESCRIPTION

System Modules:

1. Registration
2. Sharing Data
3. Secure Cloud Storage
4. Proxy re-encryption
5. Data retrieval

Registration:

For the registration of user with identity ID the group manager randomly selects a number. Then the group manager adds into the group user list which will be used in the traceability phase. After the registration, user obtains private key which will be used for group signature generation and file decryption.
Registration

Sharing Data:

The canonical application is data sharing. The public auditing property is especially useful when we expect the delegation to be efficient and flexible. The schemes enable a content provider to share her data in a confidential land selective way, with a fixed and small cipher text expansion, by distributing to each authorized user a single and small aggregate key.

Secure Cloud Storage:

Data robustness is a major requirement for storage systems. There have been many proposals of storing data over storage servers. One way to provide data robustness is to replicate a message such that each storage server stores a copy of the message.A decentralized erasure code is suitable for use in a distributed storage system.

Proxy re-encryption:

Proxy re-encryption schemes are crypto systems which allow third parties (proxies) to alter a cipher text which has been encrypted for one user, so that it may be decrypted by another user. By using proxy re-encryption technique the encrypted data (cipher text) in the cloud is again altered by the user. It provides highly secured information stored in the cloud. Every user will have a public key and private key. Public key of every user is known to everyone but private key is known only the particular user.

Data retrieval:

Reports and data are the two primary forms of the retrieved data from servers. There are some overlaps between them, but queries generally select a relatively small portion of the server, while reports show larger amounts of data. Queries also present the data in a standard format and usually display it on the monitor; whereas reports allow formatting of the output however you like and is normally retrieved.

VI. CONCLUSION

We have presented a novel image fusion method based on guided filtering. The proposed method utilizes the average filter to get the two-scale representations, which is simple and effective. More importantly, the guided filter is used in a novel way to make full use of the strong correlations between neighborhood pixels for weight optimization. Experiments show that the proposed method can well preserve the original and complementary information of multiple input images. Encouragingly, the proposed method is very robust to image registration.

VII. FUTURE ENHANCEMENT

The proposed method is computationally efficient, making it quite qualified for real applications. At last, how to improve the performance of the proposed method by adaptively choosing the parameters of the guided filter can be further researched.

REFERENCES

[1] A. A. Goshtasby and S. Nikolov, “Image fusion: Advances in the state of the art,” *Inf. Fusion*, vol. 8, no. 2,pp. 114–118, Apr. 2007.

[2] D. Socolinsky and L. Wolff, “Multispectral image visualization through first-order fusion,” *IEEE Trans.Image Process.*, vol. 11, no. 8,pp. 923–931, Aug. 2002.

[3] R. Shen, I. Cheng, J. Shi, and A. Basu, “Generalized random walks for fusion of multi-exposure images,”*IEEE Trans. Image Process.*, vol. 20,no. 12, pp. 3634–3646, Dec. 2011.

[4] S. Li, J. Kwok, I. Tsang, and Y. Wang, “Fusing images with different focuses using support vectormachines,”

IEEE Trans. Neural Netw.,vol. 15, no. 6, pp. 1555–1561, Nov. 2004.

[5] G. Pajares and J. M. de la Cruz, "A wavelet-based image fusion tutorial, "*Pattern Recognit.*, vol. 37, no. 9,pp. 1855–1872, Sep. 2004.

[6] D. Looney and D. Mandic, "Multiscale image fusion using complex extensions of EMD," *IEEE Trans.Signal Process.*, vol. 57, no. 4,pp. 1626–1630, Apr. 2009.