Review on Secure Storage Systems

[1]J. R. Pansare, [2]A. D. Ambade
[1,2]Dept. of Computer Engineering, Savitribai Phule Pune University, India

Abstract
In the last few years, the idea of connecting existing computing devices has given birth to a new concept called "connecting things". The thing includes sensors, actuators, RFID tags, any computing device which can sense the environment and act upon. Due to advances in sensor data collection technology, such as ubiquitous, embedded devices and RFID technology has led to a large number of devices connected to the net and that continuously transmit their data over time. This data is precious to many enterprises, so there is need of secure mass storage system for this data. This paper includes a survey of various storage systems and methodologies to store data securely on the storage system.

Keywords
Storage System, Security

I. Introduction

The term "Big Data" is used for data with three V's. These are Volume, Velocity and Variety. Today, with rapid development of technology, the data is also increasing exponentially. This large and growing data is important to many enterprises to perform data mining and analytics activities. So, there is requirement for mass storage system which can store this ever increasing data.
When designing a storage system the four things to consider are availability, reliability, flexibility and security. Availability ensures that data is available throughout. Reliability ensures that data integrity is maintained. Flexibility is that devices can be added and removed as and when required. The major concern today is security which is ensures that data stored is secure.
Securing stored data involves preventing unauthorized access along with preventing destruction of data (accidental or intentional), corruption or infection of information. This paper studies different storage systems and infrastructure frameworks to ensure security.

II. Different Secure Storage Systems

A. POTSHARDS

POTSHARDS (Protection Over Time, Securely Harbouring And Reliably Distributing Stuff) is a long term secure storage system without using encryption [1]. Encryption is unsuited in many situations for data storage which requires indefinitely long periods of time. As key management is difficult and cryptosystems should be updated to provide secrecy through encryption over periods of decades. The goal of the system is provide security to relatively static data with indefinite long lifetime. The three primary security properties that are provided by this archive are:

- The stored data must be viewable to authorized readers only.
- The data must be accessible and available to authorized users for a reasonable amount of time.
- The data integrity should be maintained.

POTSHARDS uses three primary techniques to provide security for long term:

- Secret splitting
- Approximate pointers
- Use of secure distributed RAID techniques across multiple independent archives

In secret splitting a block is broken into n pieces, of which m must be used to reconstitute the original block, any set of pieces fewer than m does not contain information about the original block. Approximate pointers are used to reconstitute the data in a reasonable time even if all indices over a user's data have been lost. To achieve this there is no need to sacrifice any secrecy property provided by the secret splitting. POTSHARDS first splits users data into secure shards before storing. These shards are then distributed to a number of archives.

B. Secure Storage System for Aggregate Data Storage

This storage system is used to store aggregate IoT data obtained from various devices. Here again secret sharing is used. Shamir's secret sharing algorithm is used with added internal padding [2]. Before storing the data it is split into shares and this data is stored to different storage devices. In the share generation procedure the original data submitted by the client is divided into blocks according to the threshold. Assume that there is M bytes of data and threshold is T. Each block will contain (T-1) bytes of data and if any block contains data lesser than the threshold then padding is added, which is always 1. For each block of data a (T-1)th degree polynomial is generated. Each coefficient is assigned one byte of data and a0 always contains padding size. To retrieve the original data T equations are generated for T shares and are aligned to form multiplication of two matrices.
There are four main components of the system: client, dispatcher, peer manager and peers. Client is the data owner who is responsible for share generation. Dispatcher maintains the peer managers IP address and tells client where to store the data blocks. Peer manager manages peer groups, which are storage devices. After the client receives data, it is divided to blocks using scaled secret sharing scheme. Then client will ask dispatcher IP address of the peer managers to store data blocks. The data is then stored to peer devices via peer managers. To retrieve the original data client will directly contact to peer manager.
Shield
Shield is a stackable secure storage system for sharing file in public storage [3]. As the amount of personal data stored to public data storage is increasing, user have become vulnerable towards losing their data and the data is at risk as I it is stored at third party sites. Traditionally users can either completely trust the storage provider or users have manage their files. Such systems are practically inapplicable in cloud environment. Shield addresses the above challenges by a new proposed secure system architecture and implements stackable secure storage system, where a proxy server is in charge of authentication and access control. A new variant of Merkel Hash Tree is proposed which support file content update and efficient integrity checking. Also a hierarchical key organization is designed for convenient key management and efficient permission revocation.
The system mainly considers cloud service of data sharing which includes of the four entities, the cloud server, the proxy server group, many file owners, and many file users. as illustrated in fig. 1

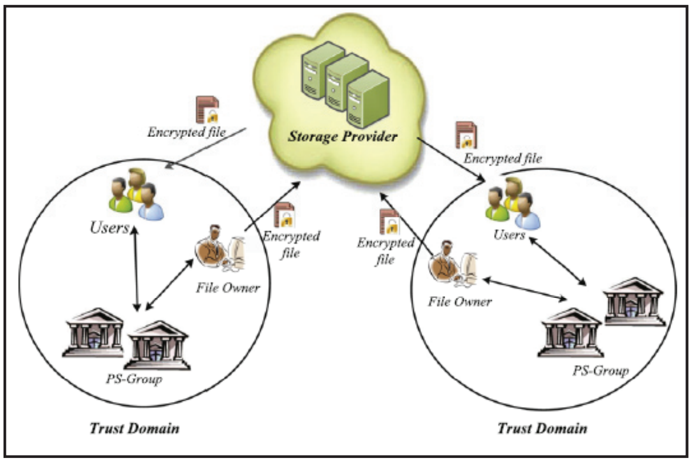

Fig. 1: System Model of Shield

1. Trust Domain

It organizes the massive number of users who are assigned various attributes in the cloud environment efficiently. The trust domain specifies the granularity of file sharing and users in same domain are treated as same.

2. Cloud Server

The cloud server performs two tasks, of reliably storing the files and using existing methods (e.g., Access Control List) to enforce access control of the ciphertext.

3. Proxy Server

Proxy server is responsible for responsible for processing access requests of user by distributing the corresponding secret keys according to their access permissions.

4. File Owners and File Users

File owners create the data file and apart from this they grant access permissions to their files.
Shield was designed for secure data storage and sharing in the trust domain under the shared network and storage environments and to save data owners from tedious key management.

D. Cryptonite

Cryptonite [4] is secure storage repository for sharing scientific datasets in public cloud. Data sharing is the key concept in scientific computing, as we are moving toward data intensive sciences and engineering. The sharing is not just to scientific results but it also refers to the raw and intermediate data that is required in the scientific process. The architecture integrates the client side libraries with the repository service and existing cloud storage service operations like, PUT, GET, DELETE, GRANT, REVOKE and SEARCH.
The fig. 2 illustrates kryptonite architecture. The client side library is responsible for cryptographic operations like encryption and pre-processing of the text file while uploading, and decrypting it on the receipt. The Data Repository Service runs within the Cloud virtual machine that has three sub-components: File Manager (FM), Secure Index Manager (SIM) and Audit Manager (AM). File Manager interacts with the cloud storage to store and retrieve files requested by the user, and also restricts unauthorised updates to data. Secure audit log for each file access is maintained by the Audit Manager. Secure index per user for all the files owned by that user is maintained by the Secure Index Manager. The index is stored in the Secure Index Storage. The user's request is accepted by Index Manager which then executes a SEARCH query over the index and returns matching results. The Cryptonite Secure Storage (CSS) is the storage account with public Cloud which is used to store the files and metadata. It uses standard authentication mechanisms that are provided by Cloud data services.

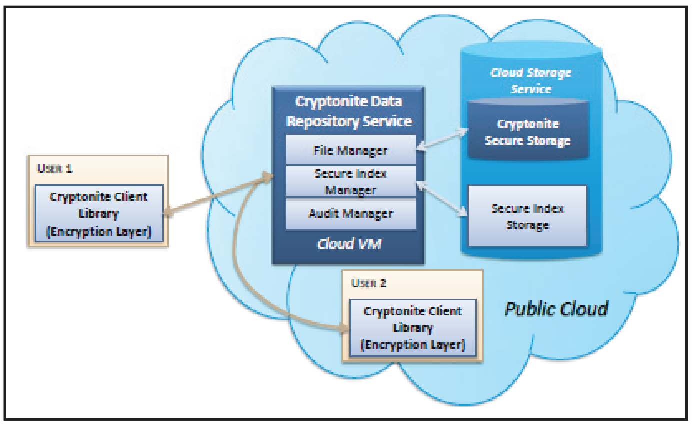
Cryptonite Data Repository Service
File Manager
Secure Index Manager
Audit Manager
Cloud VM
Cloud Storage Service
Cryptonite Secure Storage
Secure Index Storage
USER 1
Cryptonite Client Library (Encryption Layer)
USER 2
Cryptonite Client Library (Encryption Layer)
Public Cloud

Fig. 2: Cryptonite Architecture

The cryptonite use various cryptographic and security techniques in its design like Public Key Infrastructure (PKI), Digital Signatures, Broadcast Encryption, Lazy Revocation, Key rotation, Searchable Encryption. By performing client side encryption before data is stored in the repository the Cryptonite service ensures data confidentiality.

E. Secure Distributed Repository

The secure distributed repository[5] is the data repository service for Grid environments supporting secure sharing of confidential data by members of ad-hoc created groups. This repository is designed to make it easier for ad-hoc collaborations which require sharing of restricted-access data between members of dynamically created groups. For this the repository must support creating and managing users group and provide access to data based on user's membership. Each repository entry is associated with a metadata record of variable length vector attribute of name-value pair. Both generic and application specific properties are captured by attributes. Each metadata record is identified by an URI (Unique Resource Identifier). A replica locator maintains the relationship between URI and physical location (URL) of the corresponding repository entry. The relationship between metadata URI and URL for corresponding entry is one-to-many. Instead of storage entry it rather feasible to associate collaborative group access privileges with metadata record. The access control mechanisms of are based on groups privileges to perform specified operations on the entries. A user may be the member of different groups, having different privileges for different groups. It will be tedious to modify the flag permissions individually for each resource to be share. For this reason folders are used to arrange view of data collection.

F. HASS

HASS[6] is a Highly Available, Scalable and Secure distributed data storage systems. To achieve scalability in terms of performance and key management, file systems such as Object based Storage Devices (OSD) and Identity Based Encryption (IBE) are integrated. High performance I/O is provided by OSD while IBE simplifies key distribution and eliminates pre shared secret keys. For privacy and integrity both static and dynamic data are protected with different encryption strategies. IBE is used to protect data in transit while secret sharing is used to protect data at rest. HASS is a high performance and fault resilient infrastructure of distributed storage systems, which is deployed over object based storage devices. It uses large number of servers/devices for parallel I/O operations. High availability of data is obtained by replication. To achieve flexible and scalable key management Identity based encryption is adopted. Large data files are stripped into thousands of OSDs for parallel I/O. Each OSD is a cluster of a root node and 2D device/server matrix. At this level secret sharing is used to protect static data, without using key based encryption or decryption. The root node or server splits the data object into c shares based on (b,c) scheme and distributes to in a row of the matrix. Then these shares are duplicated for high availability and fault resilience along the same columns. Encryption is done when data objects are transferred between client and root, and when shares are being transferred between root and device matrix. IBE is used for protection of data in transit as it allows to be arbitrary strings and private keys can be derived from public keys. For private keys IBE uses centralized key generator. There is no need to transfer private keys between devices. Each client and server maintains its own IBE infrastructure. HASS is a reconfigurable system, in which OSDs can join and leave.

Blakley's Secret Sharing Scheme for Distributed File Systems: Security, reliability and scalability are the main design goals of Distributed File System (DFS) in cloud storage. Traditionally data duplication and cryptography were used to support these features of DFS. However, key management is an hassle in cryptography and it will be a costly affair. A revised Blakley's secret sharing scheme is applied to the DFS to support security and reliability without sacrificing scalability [7]. The distributed system is deployed with Graphics Processing Unit (GPU). There are two phases of secret sharing scheme: share generation and data restoration. To improve the security level the large data is split into shares and a random number is added. There are six phases of share generation: Padding Addition, Internal Random Bytes (IRB) Insertion and XOR, Random Matrix Generation, Matrix Multiplication, Matrix Concatenation, and Share Division. The following fig. 3 illustrates share generation phase

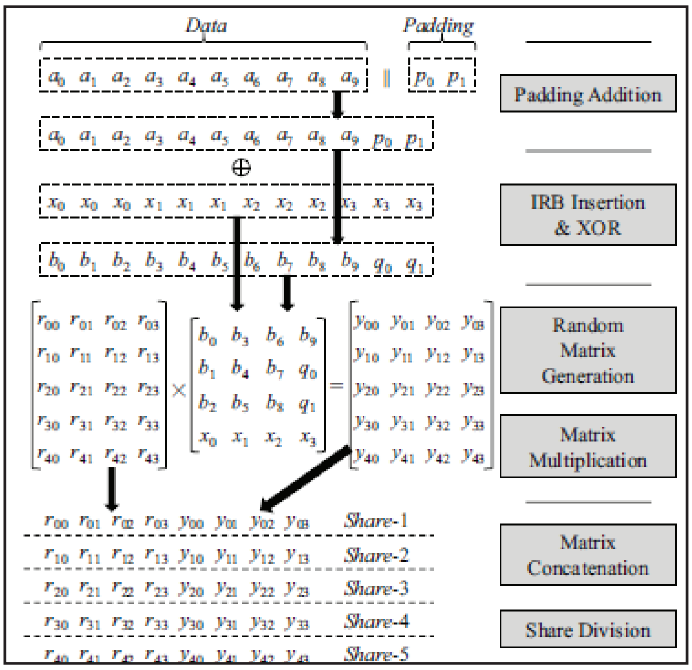

Fig. 3: Share Generation

Data restoration also has six phases: Share Combination, Matrix Extraction, Gauss Elimination, Matrix Multiplication, IRB XOR and Deletion, and Padding Deletion. Data restoration is explained in the following figure 4. The distributed system based on secret sharing consists of an index node and some compute and data nodes. Compute and data nodes status is managed by index node. This is done by positively requesting their status periodically and receiving the status report passively. The index node will phase out after assigning compute nodes for share generation and data restoration as well as after assigning data node for storage. Then client will directly communicate with compute nodes for their later work.

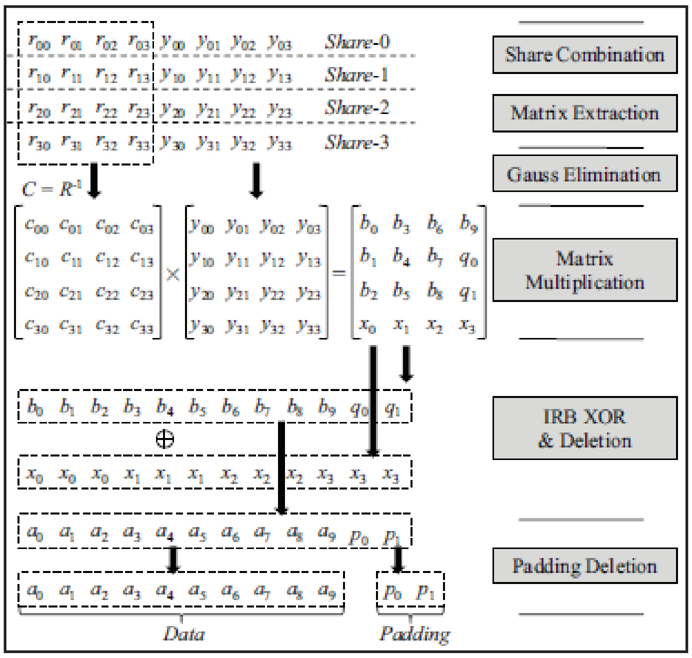

Fig. 4: Data Restoration

H. Key-Aggregate Cryptosystem(KAC)

In cloud storage data sharing is an important functionality. This scheme is proposed to securely, flexibly and efficiently share data with others on cloud storage. KAC is a public key encryption scheme where any set of cipher text is decryptable by a constant size decryption key. This decryption key is generated by the owner of the master key. Using KAC user will encrypt the message with a public key and a ciphertext identifier called class. If two users want to share something on cloud, they should encrypt their messages using KAC scheme, by which an aggregate key is generated and this key is used for decryption. The KAC scheme consist of five polynomial time algorithms:

- Setup: this algorithm is used by data owner to establish the public system parameter.
- KeyGen: this algorithm generates a public/master-secret key pair.
- Encrypt: it is use to encrypt the message by anyone who wants to encrypt.
- Extract: this algorithm is used by the data owner to generate the aggregate key with the help of a master secret key for a set of ciphertext class.
- Decrypt: any user with an aggregate key can decrypt the ciphertext provided it contains the ciphertext class within it.

I. DepSky

Companies that handle critical data are using cloud storage services to store their data due to its increasing popularity. The critical data includes medical record databases, power system historical information and financial data. DepSky is a system that provides encryption, encoding and replication of the data on diverse clouds that form a cloud-of-clouds to improve the availability, integrity and confidentiality of information stored in the cloud. DepSky is a dependable and secure storage system uses the benefits of cloud computing by making use of combination of diverse commercial clouds to build cloud of clouds. It is virtual cloud storage where a user can access it by invoking operations of individual cloud. Depsky addresses four limitations of cloud computing; loss of availability, loss of privacy, loss of data and corruption of data. The asynchronous distributed system consist three types of parties, readers, writers and cloud storage. Here reader and writer are roles of clients.

III. Conclusion

Security is a major concern in today's world. Along with availability, reliability and scalability, security of the system is also important. Security should be provided not only for data in transit but also for data at rest i.e., dynamic and static data respectively. There are many schemes designed to protect the data. They may be cryptographic or non-cryptographic. In this paper various secure systems and schemes are studied.

References

[1] Mark W. Storer, Kevin M. Greenan, Ethan L. Miller, Kaladhar Voruganti,POTSHARDS: secure long-term storage without encryption, In: Proceedings of USENIX Annual Technical Conference, 2007.

[2] Hai Jiang, Feng Shen, Su Chen, Kuan-Ching Li, Young-Sik Jeong, "A secure and scalable storage system for aggregate data in IoT", In Future Generation Computer Systems, Elsevier(2015)

[3] Jiwu Shu, Zhirong Shen, Wei Xue,"Shield: a stackable secure storage system for file sharing in public storage", J. Parallel Distrib. Comput. 74 (9) (2014).

[4] Alok Kumbhare, Yogesh Simmhan, Viktor Prasanna, Designing a secure storage repository for sharing scientific datasets using public clouds, In: Proceedings of the Second International Workshop on Data Intensive Computing in the Clouds, 2011.

[5] Tomasz Haupt, Anand Kalyanasundaram, Igor Zhuk, Architecture for a secure distributed repository, In: Proceedings of the 7th IEEE/ACM International Conference on Grid Computing, 2006.

[6] Zhiqian Xu, Hai Jiang, HASS: Highly available, scalable and secure distributed data storage systems, In: Proceedings of the 2009 IEEE/IFIP International Symposium on Trusted Computing and Communications, 2009.

[7] Su Chen, Yi Chen, Hai Jiang, Laurence T. Yang, Kuan-Ching Li, A secure distributed file system based on revised Blakley's secret sharing scheme, in: Proceedings of the 11th IEEE International Conference on Trust, Security and Privacy in Computing and Communications, Liverpool, UK, 2012.

[8] Cheng-Kang Chu, Sherman S.M. Chow, Wen-Guey Tzeng, Jianying Zhou, Robert H. Deng,"Key-aggregate cryptosystem for scalable data sharing in cloud storage", IEEE Trans. Parallel and Distributed System, Vol. 25, pp. 468-477, 2014

[9] Bessani, Alysson, et al.,"DepSky: dependable and secure storage in a cloud-of-clouds." ACM Transactions on Storage (TOS) 9.4 (2013): 12.