An Enhanced Cloud Computing Security mechanism From Single to Multi-Clouds

Mrs Manisha kumari[1], Pooja Soni[2]
[1,2]Chandravati Group Of Institutions,Bharatpur(Raj.)

Abstract

The cloud computing has increased rapidly in many organizations. Cloud computing have many benefits in terms of low cost and accessibility of data but Ensuring the security of cloud computing are major factor in the cloud computing environment, as users often store sensitive information with cloud storage providers but these providers may be untrusted. Dealing with "single cloud" providers is predicted to become less popular with customers due to risks of service availability failure and the possibility of malicious insiders in the single cloud. A movement towards "multi-clouds", or in other words, "interclouds" or "cloud-of-clouds" has emerged recently.

This paper implement multiclouds with extra security mechanism. It is found that the research into the use of multi-cloud providers to maintain security has received less attention from the research community than has the use of single clouds. This work aims to promote the use of multi-clouds due to its ability to reduce security risks that affect the cloud computing user.

Keywords:cloud computing,interclouds,security.

I. Introduction

The use of cloud computing is increasingly popular due to the potential cost savings from outsourcing data to the cloud service provider (CSP). One technique to protect the data from a possible untrusted CSP is for the data owner to encrypt the outsourced data. Flexible encryption schemes such as attribute based encryption (ABE) can be adopted to provide fine grained access control.

ABE allows data to be encrypted using an access structure comprised of different attributes. Instead of specific decryption keys for specific files, users are issued attribute keys. Users must have the necessary attributes that satisfy the access structure in order to decrypt a file.

The key problem of storing encrypted data in the cloud lies in revoking access rights from users. A user whose permission is revoked will still retain the keys issued earlier, and thus can still decrypt data in the cloud. A naive solution is to let the data owner immediately re-encrypt the data, so that the revoked users cannot decrypt the data using their old keys, while distributing the new keys to the remaining authorized users. This solution will lead to a performance bottleneck, especially when there are frequent user revocations.

In this paper, we solve this problem by proposing a time based re-encryption scheme, which enables the cloud servers to automatically re-encrypt data based on their internal clocks.

Flexibility: We have assumptions about the underlying network as possible and can be used in the majority of deployments to detect compromises.

Robustness: Wireless Sensor network should be robust. Compromised nodes may attempt to undermine the detection system through malicious behavior, such as slander attacks, where they send false information that implicates legitimate nodes as compromised.

Scalability: Since motes have limited resources, applications with high overheads will interfere with other applications and decrease the lifespan of the mote. So it should be scalable. i.e life spam should be more.

II. Basic idea

Many researchers have proposed storing encrypted data in the cloud to defend against CSP.**S.Kamara** in his work **"Cryptographic cloud storage"** considered the problem of building a secure cloud storage service on top of a public cloud infrastructure where the service provider is not completely trusted by the customer. He described, at a high level, several architectures that combine recent and non-standard cryptographic primitives in order to achieve his goal. At its core, the architecture consists of three components: a data processor (DP), that processes data before it is sent to the cloud; a data verifier (DV), that checks whether the data in the cloud has been tampered with; and a token generator (TG), that generates tokens that enable the cloud storage provider to retrieve segments of customer data; and a credential generator that implements an access control policy by issuing credentials to the various parties in the system (these credentials will enable the parties to decrypt encrypted files according to the policy). Under this

approach, users are revoked by having a third party to re encrypt data such that previous keys can no longer decrypt any data

Mahesh in his work "Plutus: Scalable secure file sharing on untrusted storage" introduces a new secure file system which strives to provide strong security even with an untrusted server. The main feature of Plutus is that all data is stored encrypted and all key distribution is handled in a decentralized manner. All cryptographic and key management operations are performed by the clients, and the server incurs very little cryptographic overhead. In this paper author concentrate on the mechanisms that Plutus uses to provide basic file system security features — (1) to detect and prevent unauthorized data modifications, (2) to differentiate between read and write access to files, and (3) to change users' access privileges.

In encrypt-on-disk file systems, the clients encrypt all directories and their contents. The original work in this area is the Cryptographic File System (CFS), which used a single key to encrypt an entire directory of files and depended on the underlying file system for authorization of writes. Later variants on this approach include TCFS, which uses a lockbox to protect only the keys. Cepheus uses group-managed lockboxes with a centralized key server and authorization at the trusted server. SNAD also uses lockboxes and introduces several alternatives for verifying writes. The SiRiUS file system layers a cryptographic storage file system over heterogenous insecure storage such as NFS and Yahoo! Briefcase.

Kevin Fu in his work **"Improved Proxy Re-encryption Schemes with Applications to Secure Distributed Storage "**discussed several efficient proxy re-encryption schemes that offer security improvements over earlier approaches. The primary advantage of his scheme is that they are unidirectional (i.e., Alice can delegate to Bob without Bob having to delegate to her) and do not require delegators to reveal all of their secret key to anyone—or even interact with the delegatee—in order to allow a proxy to re-encrypt their cipher texts. In our schemes, only a limited amount of trust is placed in the proxy. For example, it is not able to decrypt the cipher texts it re-encrypts and we prove our schemes secure even when the proxy publishes all the re-encryption information it knows. This enables a number of applications that would not be practical if the proxy needed to be fully trusted. The solution by author for instance, lets the data owner issue a re-encryption key to an untrusted server to re encrypt the data. Their solution utilizes PRE "Divertible protocols and atomic proxy cryptography ", which allows the server to re-encrypt the stored cipher text to a different cipher text that can only be decrypted using a different key. During the process, the server does not learn the contents of the cipher text or the decryption keys.

ABE is a new cryptographic technique that efficiently supports fine grained access control. The combination of PRE and ABE was first introduced by "Achieving secure, scalable, and fine-grained data access control in cloud computing ", and extended by G.Wang in Hierarchical attribute-based encryption for fine-grained access control in cloud storage services. In this, a hierarchical attribute-based encryption (HABE) scheme is proposed to achieve high performance and full delegation. The main difference between prior work and ours is that we do not require the underlying cloud infrastructure to be reliable in order to ensure correctness.

Our scheme relies on time to re-encrypt data. However, in a cloud, the internal clock of each cloud server may differ. There have been several solutions to this problem. For instance, **F. Cristian** in his work "**Probabilistic clock synchronization**" proposed a probabilistic synchronization scheme, which exchanges messages to get remote servers' accurate clocks with high probability. Work by **K. Romer** in " **Time synchronization in ad hoc networks** " used message delay to estimate the maximal difference between two communicating nodes to synchronize the clocks. Work by **L. Gillam**, "**Cloud Computing: Principles, Systems and Applications**" proposed a clock synchronization scheme for cloud environments, which uses an authoritative time source shared by all participants in a transaction to achieve clock synchronization between virtual cloud policy enforcement points. By applying these techniques to achieve loose synchronization in the cloud, and to determine the maximal time difference between the data owner and each cloud server, our R3 scheme can always achieve correct access control in unreliable clouds.

Problem Statement

The key problem of storing encrypted data in the cloud lies in revoking access rights from users. A user whose permission is revoked will still retain the keys issued earlier, and thus can still decrypt data in the cloud. A naive solution is to let the data owner immediately re-encrypt the data, so that the revoked users cannot decrypt the data using their old keys, while distributing the new keys to the remaining authorized users. This solution will lead to a performance bottleneck, especially when there are frequent user revocations.

Data Integrity

One of the most important issues related to cloud security risks is data integrity. The data stored in the cloud may suffer from damage during transition operations from or to the cloud storage provider. Cachinet al.[12] give examples of the risk of attacks from both inside and outside the cloud provider, such as the recently attacked Red Hat Linux's distribution servers.

Data Intrusion

According to Garfinkel[19], another security risk that may occur with a cloud provider, such as the Amazon cloud service, is a hacked password or data intrusion. If someone gains access to an Amazon account password, they will be able to access all of the account's instances and resources.

Service Availability

Another major concern in cloud services is service availability. Amazon [6] mentions in its licensing agreement that it is possible that the service might be unavailable from time to time. The user's web service may terminate for any reason at any time if any user's files break the cloud storage policy. In addition, if any damage occurs to any Amazon web service and the service fails, in this case there will be no charge to the Amazon Company for this failure.

III.Software Design

Data owner generates the keys using the key manager. The files uploaded by Data owner, are encrypted by the Encryption module and stored in the cloud storage.

The User who wants to access file gets the key from the data owner and decrypts the file using decryption. Key manager module will keep track of keys and frequently changes the keys , so that user cannot access file always. Also the file in cloud is re – encrypted with new keys so that user holding previous keys cannot decrypt the file with old keys.

Solution strategy:

Our solution is to allow each cloud server to independently re-encrypt data without receiving any command from the data owner. In this paper, we propose a reliable re-encryption scheme in unreliable clouds (R3 scheme for short). R3 is a time-based re-encryption scheme, which allows each cloud server to automatically re-encrypt data based on its internal clock. The basic idea of the R3 scheme is to associate the data with an access control and an access time.

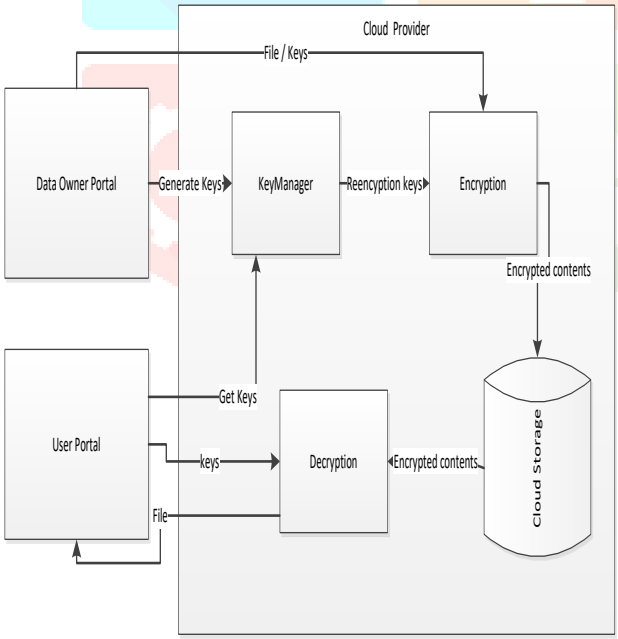


Fig 1 Architectural Diagram

Each user is issued keys associated with attributes and attribute effective times. The data can be decrypted by the users using the keys with attributes satisfying the access control, and attribute effective times satisfying the access time. Unlike the command-driven re-encryption scheme, the data owner and the CSP share a secret key, with which each cloud server can re encrypt data by updating the data access time according to its own internal clock. Even through the R3 scheme relies on time, it does not require perfect clock synchronization among cloud servers. Classical clock synchronization techniques that ensure loose clock synchronized in the cloud are sufficient.

Level 1 Data Flow Diagram

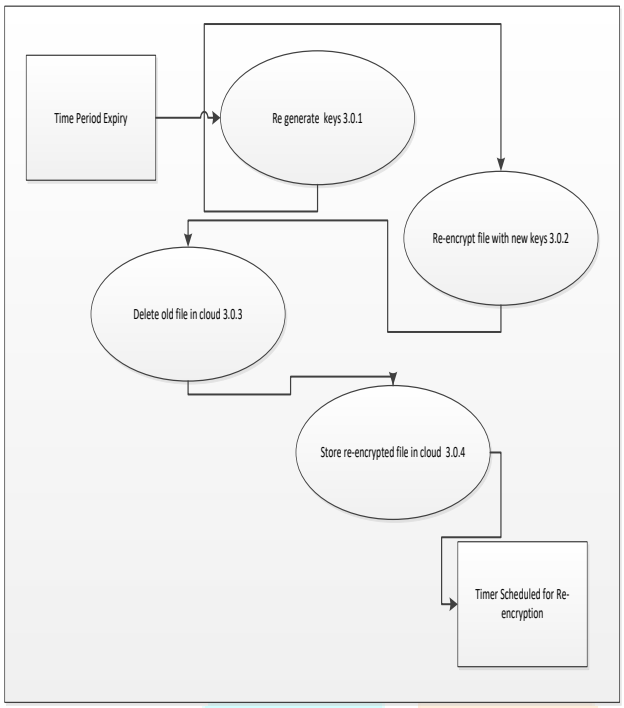


Fig 2 Data level diagram

As mentioned earlier, the loss of availability of service is considered one of the main limitations in cloud computing and it has been addressed by storing the data on several clouds. The loss of customer data has caused many problems for many users such as the problem that occurred in October 2009 when the contacts, photos,etc. of many users of the Sidekick service in Microsoft were lost for several days [44].

As the DepSky system deals with different cloud providers, the DepSky library deals with different cloud interface providers and consequently, the data format is accepted by each cloud. The DepSky data model consists of three abstraction levels: the conceptual data unit, a generic data unit, and the data unit implementation.

System model. The DepSky system model contains three parts: readers, writers, and four cloud storage providers, where readers and writers are the client's tasks. Bessani et al. [8] explain the difference between readers and writers for cloud storage. Readers can fail arbitrarily (for example, they can fail by crashing, they can fail from time to time and then display any behavior) whereas, writers only fail by crashing.

Cloud storage providers in the DepSky system model. The Byzantine protocols involve a set of storage clouds (n) where n = 3 f +1, and f is maximum number of clouds which could be faulty. In addition, any subset of (n – f) storage cloud creates byzantine quorum protocols [8].

Analysis of Multi-Cloud Research

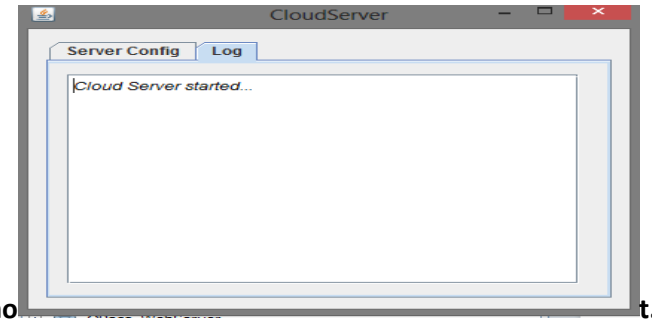
Moving from single clouds or inner-clouds to multiclouds is reasonable and important for many reasons.According to Cachinet al. [12] "Services of single clouds are still subject to outage". In addition, Bowers et al. [10] showed that over 80% of company management "fear security threats and loss of control
of data and systems". Vukolic [54] assumes that the main purpose of moving to interclouds is to improve what was offered in single clouds by distributing reliability, trust, and security among multiple cloud providers. In addition, reliable distributed storage [15] which utilizes a subset of BFT techniques was suggested by Vukolic [54] to be used in multi-clouds. A number of recent studies in this area have built protocols for interclouds. RACS (Redundant Array of Cloud Storage) [3] for instance, utilizes RAID-like techniques that are normally used by disks and file systems, but for multiple cloud storage. Abu-Libdeh et al. [3] assume that to avoid "vender lock-in", distributing a user's data among multiple clouds is a helpful solution. This replication also decreases the cost of switching providers and offers better fault tolerance. Therefore, the storage load will be spread among several providers as a result of the RACS proxy [3]. HAIL (High Availability and Integrity Layer) [10] is another example of a protocol that controls multiple clouds. HAIL is a distributed cryptographic system that
permits a set of servers to ensure that the client's stored data is retrievable and integral. HAIL provides a software layer to address availability and integrity of the stored data in an intercloud [10].

IV Result

1.In this phase user name aand password send to user.

2.Here user can create user name and password and register successfully.

3.Also cloud server is started in the given screensho



t.

V. Future work

For future work, we aim to provide a framework to supply a secure cloud database that will guarantee to prevent security risks facing the cloud computing community. This framework will apply multi-clouds and the secret sharing algorithm to reduce the risk of data intrusion and the loss of service availability in the cloud and ensure data integrity. In relation to data intrusion and data integrity, assume we want to distribute the data into three different cloud providers, and we apply the secret sharing algorithm on the stored data in the cloud provider. An intruder needs to retrieve at least three values to be able to find out the real value that we want to hide from the intruder. This depends on Shamir's secret sharing algorithm with a polynomial function technique which claims that even with full knowledge of (k – 1) clouds, the service provider will not have any knowledge of vs (vs is the secret value) [47]. We have used this technique in previous databases-as-a-serves research [5]. In other words, hackers

need to retrieve all the information from the cloud providers to know the real value of the data in the cloud. Therefore, if the attacker hacked one cloud provider's password or even two cloud provider's passwords, they still need to hack the third cloud provider (in the case where k = 3) to know the secret which is the worst case scenario. Hence, replicating data into multi-clouds by using a multi-share technique [5] may reduce the risk of data intrusion and increase data integrity. In other words, it will decrease the risk of the Hyper-Visor being hacked.

[1] S. Kamara and K. Lauter, "Cryptographic cloud storage," Financial Cryptography and Data Security, 2010.
[2] M. Armbrust, A. Fox, R. Griffith, A. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, A. Rabkin, and I. Stoica, "A view of cloud computing," Communications of the ACM, 2010.
[3] A. Sahai and B. Waters, "Fuzzy identity-based encryption," Advances in Cryptology–EUROCRYPT, 2005.
[4] V. Goyal, O. Pandey, A. Sahai, and B. Waters, "Attribute-based encryption for fine-grained access control of encrypted data," in Proc. of ACM CCS, 2006.
[5] J. Bethencourt, A. Sahai, and B. Waters, "Ciphertext-policy attributebased encryption," in Proc. of IEEE Symposium on S&P, 2007.
[6] M. Blaze, G. Bleumer, and M. Strauss, "Divertible protocols and atomic proxy cryptography," Advances in Cryptology–EUROCRYPT, 1998.
[7] A. Boldyreva, V. Goyal, and V. Kumar, "Identity-based encryption with efficient revocation," in Proc. of ACM CCS, 2008.
[8] G. Wang, Q. Liu, and J. Wu, "Hierarchical attribute-based encryption for fine-grained access control in cloud storage services," in Proc. Of ACM CCS (Poster), 2010.
[9] S. Yu, C. Wang, K. Ren, and W. Lou, "Achieving secure, scalable, and fine- grained data access control in cloud computing," in Proc. of IEEE INFOCOM, 2010.
[10] F. Cristian, "Probabilistic clock synchronization," Distributed Computing, 1989.
[11] K. Romer, "Time synchronization in ad hoc networks," in Proc. of ACM MobiHoc, 2001.
[12] P. Ramanathan, K. Shin, and R. Butler, "Fault-tolerant clock synchronization in distributed systems," Computer, 2002.
[13] N. Antonopoulos and L. Gillam, "Cloud Computing: Principles, Systems and Applications," Springer Publishing Company, 2010.
[14] M. Kallahalla, E. Riedel, R. Swaminathan, Q. Wang, and K. Fu, "Plutus: Scalable secure file sharing on untrusted storage," in Proc. of USENIX
FAST, 2003.ACM Transactions on Information and System Security, 2006.