Privacy Presering To Identify De-Duplicate Data With Multi-Pupils Management In Cloud Stroage

T.Ananthi[1] , S. Divya[2], V. Hemalatha[3], S.Janarthanan[4]

[1,2,]3Student Members, Department of Computer Science and Engineering
[4]Assistant Professor, Departmetn of Computer Science and Engineering
Arasu Engineering College, Kumbakonam

Abstract *- With the rapidly increasing amounts of data produced worldwide, networked and multi-user storage systems are becoming very popular. However, concerns over data security still prevent many users from migrating data to remote storage. The conventional solution is to encrypt the data before it leaves the owner's premises. While sound from a security perspective, this approach prevents the storage provider from effectively applying storage efficiency functions, such as compression and deduplication, which would allow optimal usage of the resources and consequently lower service cost. Client-side data deduplication in particular ensures that multiple uploads of the same content only consume network bandwidth and storage space of a single upload. Deduplication is actively used by a number of cloud backup providers as well as various cloud services. Unfortunately, encrypted data is pseudorandom and thus cannot be deduplicated: as a consequence, current schemes have to entirely sacrifice either security or storage efficiency.*

we present a scheme that permits a more fine-grained trade-off. The intuition is that outsourced data may require different levels of protection, depending on how popular it is: content shared by many users. We present a novel idea that differentiates data according to their popularity. Based on this idea, we design an encryption scheme that guarantees semantic security for unpopular dataand provides weaker security and better storage and bandwidth benefits for popular data. This way, data de-duplication can be effective for popular data, whilst semantically secure encryption protects unpopular content. We can use the backup recover system at the time of blocking and also analyze frequent login access system

I. INTROUDCTION

CLOUD COMPUTING

Cloud computing is a type of Internet-based computing that provides shared computer processing resources and data to computers and other devices on demand. It is a model for enabling ubiquitous, on-demand access to a shared pool of configurable computing resources (e.g., computer networks, servers, storage, applications and services),which can be rapidly provisioned and released with minimal management effort. Cloud computing and storage solutions provide users and enterprises with various capabilities to store and process their data in either privately owned, or third-party data centers that may be located far from the user–ranging in distance from across a city to across the world. Cloud computing relies on sharing of resources to achieve coherence and economy of scale, similar to a utility (like the electricity grid) over an electricity network.

II. SYSTEM ANALYSIS

Existing system

Many systems have been developed to provide secure storage but traditional encryption techniques are not suitable for deduplication purposes. Deterministic encryption, in particular convergent encryption, is a good candidate to achieve both confidentiality and deduplication but it suffers from well-known weaknesses which do not ensure protection of predictable files against dictionary attacks. And also existing system makes use of proxy re-encryption, has been proposed but information on performance and overhead were not provided. Unfortunately, deduplication loses its effectiveness in conjunction with end-to- end encryption. End-to-end encryption in a storage system is the process by which data is encrypted at its source prior to ingress into the storage system. It is becoming an increasingly prominent requirement due to both the number of security incidents linked to leakage of unencrypted data and the tightening of sector-specific laws and regulations. Clearly, if semantically secure encryption is used, file deduplication is impossible, as no one apart from the owner of the decryption key can decide whether two cipher texts correspond to the same plaintext. Trivial solutions, such as forcing users to share encryption keys or using deterministic encryption, fall short of providing acceptable levels of security. As a consequence, storage systems are expected to undergo major restructuring to maintain the current disk/customer ratio in the presence of end-to-end encryption. The design of storage efficiency functions in general and of deduplication functions in particular that do not lose their effectiveness in presence of end-to-end security is therefore still an open problem.

Disadvantages

- Deduplication check only with file name and not file content
- Could not achieve secure access control under a dynamic ownership changing environment
- Security degradation of the cloud service

Proposed system

Storage efficiency functions such as compression and deduplication afford storage providers better utilization of their storage back ends and the ability to serve more customers with the same infrastructure. Data deduplication is the process by which a storage provider only stores a single copy of a file owned by several of its users. There are four different deduplication strategies, depending on whether deduplication happens at the client side (i.e. before the upload) or at the server side, and whether deduplication happens at a block level or at a file level. Deduplication is most rewarding when it is triggered at the client side, as it also saves upload bandwidth. For these reasons, deduplication is a critical enabler for a number of popular and successful storage services that offer cheap, remote storage to the broad public by performing client-side deduplication, thus saving both the network bandwidth and storage costs.

The goal of the system is to guarantee data confidentiality without losing the advantage of deduplication. Confidentiality must be guaranteed for all files, including the predictable ones. The security of the whole system should not rely on the security of a single component (single point of failure), and the security level should not collapse when a single component is compromised. We consider the server as a trusted component with respect to user authentication, access control and additional encryption. The server is not trusted with respect to the confidentiality of data stored at the cloud storage provider.

Therefore, the server is not able to perform offline dictionary attacks. Anyone who has access to the storage is considered as a potential attacker, including employees at the cloud storage provider and the cloud storage provider itself. In our threat model, the cloud storage provider is honest but curious, meaning that it carries out its tasks but might attempt to decrypt data stored by users. And also implement back up recover scheme to recover data at the time of infrequent access. Admin can be sent alert to every 3 days, one week, two weeks and three weeks. If the users not login to the system means, automatically recover the data and forward to alternate storage with mobile intimation.

Advantages:

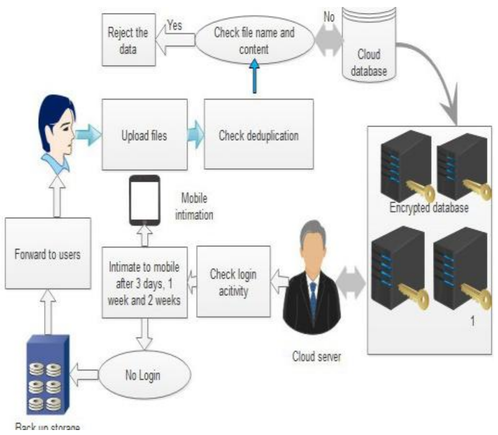
- Dynamic updation can be implemented in cloud storage
- File and file content analyzed
- Security is high and to provide data integrity to all data owners

III. SYSTEM DESIGN

SYSTEM ARCHITECTURE DIAGRAM

In proposed system, we can secure deduplication scheme to check the storage based on file name and file content. Then eliminate the files with same content. After that, store the files into cloud using symmetric encryption algorithm. The login activity can be monitoring the user accounts. If the users can't be login frequently means, automatically transfer the files into alternate mail id. Mobile intimation can be send to user registered number.

The .NET Framework (pronounced dot net) is a software framework developed by Microsoft that runs primarily on Microsoft Windows. It includes a large library and provides language interoperability(each language can use code written in other languages) across several programming languages. Programs written for the .NET Framework execute in a software environment (as contrasted to hardware environment), known as the Common LanguageRuntime (CLR), an applicationvirtual machine that provides services such as security, memorymanagement, and exception handling. The class library activity can be monitoring the user accounts. If the users can't be login frequently means, automatically transfer the files into alternate mail id. Mobile intimation can be send to user registered number.



IV. IMPLEMENTATION STAGES

Block cipher algorithm

In cryptography, a block cipher is a deterministic algorithm operating on fixed-length groups of bits, called blocks, with an unvarying transformation that is specified by a symmetric key. Block ciphers operate as important elementary components in the design of many cryptographic protocols, and are widely used to implement encryption of bulk data. Iterated product ciphers carry out encryption in multiple rounds, each of which uses a different subkey derived from the original key. One widespread implementation of such ciphers, named a Feistel network after Horst Feistel, is notably implemented in the DES cipher. Many other realizations of block ciphers, such as the AES, are classified as substitution-permutation networks. The publication of the DES cipher by the United States National Bureau of Standards (subsequently the U.S. National Institute of Standards and Technology, NIST) in 1977 was fundamental in the public understanding of modern block cipher design. It also influenced the academic development of cryptanalytic attacks.

Both differential and linear

Cryptanalysis arose out of studies on the DES design. As of 2016 there is a palette of attack techniques against which a block cipher must be secure, in addition to being robust against brute force attacks. Even a secure block cipher is suitable only for the encryption of a single block under a fixed key. A multitude of modes of operation have been designed to allow theirrepeated use in a secure way, commonly to achieve the security goals of confidentiality and authenticity. However, block ciphers may also feature as building-blocks in other cryptographic protocols, such as universal hash functions and pseudo-random number generators.

One important type of iterated block cipher known as a substitution- permutation network (SPN) takes a block of the plaintext and the key as inputs, and applies several alternating rounds consisting of a substitution stage followed by a permutation stage—to produce each block of cipher text output. The non-linear substitution stage mixes the key bits with those of the plaintext, creating Shannon's confusion. The linear permutation stage then dissipates redundancies, creating diffusion. A substitution box (S-box) substitutes a small block of input bits with another block of output bits. This substitution must be one-to-one, to ensure invertibility (hence decryption). A secure S-box will have the property that changing one input bit will change about half of the output bits on average, exhibiting what is known as the avalanche effect—i.e. it has the property that each output bit will depend on every input bit.

Block level deduplication

Block deduplication requires more processing power than the file deduplication, since the number of identifiers that need to be processed increases greatly. Correspondingly, its index for tracking the individual iterations gets also much larger. Using of variable length blocks is even more source- intensive. Moreover, sometimes the same hash number may be generated for two different data fragments, which is called hash collisions. If that happens, the system will not save the new data as it sees that the hash number already exists in the index. The algorithm steps as follows

BlockTag(FileBlock) - It computes hash of the File block as file block Tag;

DupCheckReq(Token) - It requests the Storage Server for Duplicate Check of the file block.

FileUploadReq(FileBlockID, FileBlock, Token) – It uploads the File Data to the Storage Server if the file block is Unique and updates the file block Token stored.

FileBlockEncrypt(Fileblock) - It encrypts the file block with Convergent Encryption, where the convergent key is from SHA Hashing of the file block;

TokenGen(File Block, UserID) – the process loads the associated privilege keys of the user and generate token.

FileBlockStore(FileBlockID, FileBlock, Token) - It stores the FileBlock on Disk and updates the Mapping.

V. CONCLUSION

We proposed the distributed deduplication systems to improve the reliability of data while achieving the confidentiality of the users and also shared authority outsourced data with an encryption mechanism. Four constructions were proposed to support file-level and block-level data deduplication. The security of tag consistency and integrity were achieved. We implemented our deduplication systems using the Ramp secret sharing scheme and demonstrated that it incurs small encoding/decoding overhead compared to the network transmission overhead in regular upload/download operations. In this work, we have identified a new privacy challenge during data accessing in the cloud computing to achieve privacy-preserving access authority sharing for deduplicated files.

Authentication is established to guarantee data confidentiality and data integrity. Data anonymity is achieved since the wrapped values are exchanged during transmission. User privacy is enhanced by access requests to privately inform the cloud server about the users access desires. The backup recovery scheme is to improve the recovered scheme to avoid the blockages and also refund the amount to unused spaces in cloud system.

VI. FUTURE WORK

In future, we can extend the framework to implement various encryption algorithms to improve the security and also implement in real time video and audio deduplication storage systems

References

[1]. D. T. Meyer, and W. J. Bolosky, "A study of practical deduplication,"Proc. USENIX Conference on File and Storage Technologies 2011

[2]. W. K. Ng, W. Wen, and H. Zhu, "Private data deduplicationprotocols in cloud storage," Proc. ACM SAC'12, 2012.

[3]. N. Baracaldo, E. Androulaki, J. Glider, A. Sorniotti, "Reconciling end-to-end confidentiality and data reduction in cloud storage," Proc. ACM Workshop on Cloud Computing Security, pp. 21–32, 2014.

[4]. P. Anderson, L. Zhang, "Fast and secure laptop backups with encrypted de- duplication," Proc. USENIX LISA, 2010. [5] J. Li, X. Chen, M. Li, J. Li, P. Lee, and

[5]. W. Lou, "Secure deduplicationwith efficient and reliable convergent key management," IEEE Transactions on Parallel and Distributed Sytems, Vol. 25, No. 6, 2014.