Input Inclusive Cryptosystem for Information Sharing

Durgadevi. k
UG Student
Department of Information Technology
SNS College of Technology

Gowthami. G
UG Student
Department of Information Technology
SNS College of Technology

Nandu. S
UG Student
Department of Information Technology
SNS College of Technology

Selvavinayagam. G
Assistant Professor
Department of Information Technology
SNS College of Technology

Abstract

In Cloud computing, Data sharing is the important functionality. Secure distributed data storage was developed to reduce the burden of data owner from managing numerous files, by transferring the files from data owner to proxy servers. Proxy servers can convert encrypted files for the owner to encrypted files for the receiver withoutthe necessity of knowing the content of the original files. In practice, the original files will be removed by the owner for the sake of space efficiency. Therefore, there is no guarantee that the outsourced files are not accessed by the unauthorized users and not modified by proxy servers is an important problem that has been considered. Hence, the issues on confidentiality and integrity of the outsourced data must be addressed carefully. In this paper, there are two Identity-Based Secure Distributed Data Storage (IBSDDS) schemes. The properties of IBSDDS schemes are as follows: The file owner can decide the access permission independently without the help of the private key generator; for one query, a receiver can only access one file, instead of all files of the owner. These schemes are secure against the collusion attacks, namely even if the receiver can compromise the proxy server, but cannot obtain the owner's secret key. Though the first scheme is only secure against the chosen plaintext attacks (CPA), the second scheme is secure against the chosen cipher text attacks (CCA). To the best of our knowledge, it is the firstscheme where access permission is made by the owner for an exact file and collusion attacks can be protected.

Keywords: Cloud Storage, Cipher texts, Data Owner, Data Receiver, Decryption, Encryption, Proxy Server, Secret key

I. INTRODUCTION

Cloud computing is a collection of scalable resources and computing infrastructure which provides services to users. This kind of technology helps users in handling resources effectively on-site. It involves deploying groups of remote servers and software network that allow centralized data storage and online access to computer services or resources. Based on the hosting of environment and the type of users, the architecture can be of four types: Public Cloud, Private Cloud, Hybrid Cloud and Community Cloud. In a public cloud, cloud services are hosted for public usage and anyone can have their data stored and services get done using this kind of cloud. Data security plays a major role here. A private cloud is the one where the data access and service usage restricted to single authority. A hybrid cloud is shared by a limited set of organizations and has the features of both private and public cloud. Community cloud is much like the private cloud but the data is shared among the same entities of the single organization.

It is a class of the next generation highly scalable distributed computing platform in which computing resources are offered 'as a service'. Cloud storage is storing of data off-site to the physical storage which is maintained by third party. Cloud storage is saving of digital data in logical pool and physical storage spans multiple servers which are manage by third party. Third party is responsible for keeping data available and accessible and physical environment should be protected and running at all time. Instead of storing data to the hard drive or any other local storage, we save data to remote storage which is accessible from anywhere and anytime. It reduces efforts of carrying physical storage to everywhere. By using cloud storage we can access information from any computer through internet which omitted limitation of accessing information from same computer where it is stored.

For example, bloggers can let their friends view a subset of their private pictures an enterprise may grant her employees access to a portion of sensitive data. The challenging problem is how to effectively share encrypted data. Of course users can download the encrypted data from the storage, decrypt them, then send them to others for sharing, but it loses the value of cloud storage. Users should be able to delegate the access rights of the sharing data to others so that they can access these data from the server directly. However, finding an efficient and secure way to share partial data in cloud storage is not trivial.

Cloud computing provides users with a convenient mechanism to manage their personal files with the notion called Database-As-A-Service (DAAS). In DAAS schemes, a user can outsource his encrypted files to untrusted proxy servers. Proxy servers can perform some functions on the outsourced ciphertexts without knowing anything about the original files [1]. Unfortunately, this technique has not been employed extensively. The main reason lies in that users are especially concerned on the confidentiality, integrity and query of the outsourced files as cloud computing is a lot more complicated than the local data storage systems, as the cloud is managed by an untrusted third party. After outsourcing the files to proxy servers, the user will remove them from his local machine. Therefore, how to guarantee the outsourced files are not accessed by the unauthorized users and not modified by proxy servers is an important problem that has been considered in the data storage research community.

Cryptography is the art of protecting information by transforming it into an unreadable format, called cipher text. Only those who possess a secret key can decrypt the message into plain text. This technique can be applied in a two major ways one is symmetric key encryption and other is asymmetric key encryption. In symmetric key encryption, same keys are used for encryption and decryption. In asymmetric key encryption different keys are used, public keys for encryption and private keys for decryption. Therefore, to solve the problems of data integrity and confidentiality, the symmetric key encryption technique is used. It is the cryptographic technique, here the same keys are used for encryption and decryption purpose.

The main objective of this paper is to implement the Identity Based Secure Distributed Data Storage schemes, which helps the users to share the information in a secure identity based manner; the access permission can be decided only by the owner, instead of the trusted party (PKG). Therefore, stronger security can be achieved and file based access control will be implemented with the help of IBSDDS schemes.

II. IDENTITY BASED SCHEMES FOR INFORMATION SHARING

In this paper, there are two Identity-Based Secure Distributed Data Storage (IBSDDS) schemes in standard model where, for one query, the receiver can only access one of the owner's files, instead of all files. In other words, access permission (re-encryption key) is bound not only to the identity of the receiver but also the file. The access permission can be decided by the owner, instead of the trusted party (PKG). Furthermore, these schemes are secure against the collusion attacks. There are four entities in an identity based secure distributed data storage (IBSDDS) scheme: the private key generator (PKG), the data owner, the proxy server and the receiver. The PKG validates the users' identities and issues secret keys to them. The data owner encrypts his data and outsources the ciphertexts to the proxy servers. Proxy servers store the encrypted data and transfer the ciphertext for the owner to the ciphertext for the receiver when they obtain the access permission (re-encryption key) from the owner. The receiver authenticates himself to the owner and decrypts the re-encrypted ciphertext to obtain the data.

A. Advantages:

- It has two schemes of security, the first scheme is CPA secure, and the second scheme achieves CCA security.
- To the best of our knowledge, it is the firstIBSDDS schemes where access permission is made by the owner for an exact file and collusion attacks can be protected in the standard model.
- To achieve a stronger security and implement file based access control, the owner must be online to authenticate requesters and also to generate access permissions for them.

III. ALGORITHM

The Data Encryption Standard (DES)algorithmis used here. It was a predominant symmetric key algorithm for the encryption of electronic data. It was highly influential in the advancement of modern cryptography .This algorithm was developed in the early 1970s at IBM. It uses block cipher. It encrypts the data in block size of 64 bits each. Same algorithm and key is used for encryption and decryption. Key is 56 bits of 8, 16,24,32,40,48,56,64 are discarded.

The algorithm uses a 56-bit key to encipher/decipher a 64-bit block of data. The key is always presented as a 64-bit block, every 8th bit that is 8, 16,24,32,40,48,56,64 of which is ignored. Each 8th bit is set,so that each group of 8 bits has an odd number of bits set to 1.The algorithm is best suited to implementation in hardware, probably to discourage implementations in software, which tend to be slow by comparison. Modern computers are so fast that satisfactory software implementations are readily available.

DES is the most widely used symmetric algorithm in the world, despite claims that the key length is too short. Ever since DES was first announced, controversy has raged about whether 56 bits is long enough to guarantee security.Assuming that the only feasible attack on DES is to try each key in turn until the right one is found, then 1,000,000 machines each capable of testing 1,000,000 keys per second would find (on average) one key every 12 hours. Most reasonable people might find this rather comforting and a good measure of the strength of the algorithm.Those who consider the exhaustive key-search attack to be a real possibility and to be fair the technology to do such a search is becoming a reality, can overcome the problem by using double or triple length keys. In fact, double length keys have been recommended for the financial industry for many years.

Use of multiple length keys leads us to the Triple-DES algorithm, in which DES is applied three times. In each round key and data bits are shifted, permuted, XORed and sent through, 8 round 64 bit plaintext is handed to initial permutation. Decryption is

same process perform rounds in reverse order. If we consider a triple length key to consist of three 56-bit keys K1, K2, K3 then encryption is as follows,

- Encrypt with K1
- Decrypt with K2
- Encrypt with K3

Decryption is the reverse process, which follows

- Decrypt with K3
- Encrypt with K2
- Decrypt with K1

Setting K3 equal to K1 in these processes gives us a double length key K1, K2.Setting K1, K2 and K3 all equal to K has the same effect as using a single-length (56-bit key). Thus it is possible for a system using triple-DES to be compatible with a system using single-DES.

The plaintext and key are processed by,

1) The Key is split into two 28-bit halves.
2) Each half of the key is shifted(rotated) by one or two bits, depending on the round.
3) The halves are recombined and subject to a compression permutation to reduce the key from 56 bits to 48 bits. This compressed key is used to encrypt this round's plaintext block.
4) The compressed key value is used for next round.
5) Then the data block are also split into two 32-bit halves.
6) Right side half is subject to an Expansion Permutation to increase its size to 48 bits.
7) Output of right side half 48 bits and compressed key value of 48 bits or XORed.
8) The XORed value is fed into an S-box, which substitutes key bits and reduces the 48-bit block back down to 32-bits.S-box it contain 8 boxes each box will accept 6 bit as input and gives 4 bit as output.
9) Then the 32 bits value is subject to a P-box to permute the bits
10) The output from the P-box is exclusive- OR'ed with other half of the data block(i.e)left side half.
11) The two data halves are swapped and become the next round's input.

A. Triple DES:

It is enhancement version of DES. In this 3 times iterations of DES encryption on each block is performed. In Triple DES the 3times iteration is applied to increase the encryption level and average time. Common method of 3DES is Minus Encrypt-Decrypt-Encrypt (EDE). This is equivalent to using a 168-bit encryption key method that is Encryption- Encryption -Encryption (EEE).

Decryption is performed using same algorithm in reverse order. Triple DES uses three DES keys, K1, K2 and K3, each of 56 bits excluding parity bits. k1is used for encryption , k2 is used for decryption, k3 is used for encryption.

In encryption plaintext can be converted into ciphertext .

Ciphertext = EK3(DK2(EK1(plaintext)))

In decryption ciphertext can be converted into plaintext.

Plaintext = DK1(EK2(DK3(ciphertext)))

B. Advantages of algorithm used:

- Same key is used for encryption and decryption purpose.
- Throughput is Very High.
- Confidentiality is High.

IV. IMPLEMENTATION

The Input-Inclusive Cryptosystem for Information Sharing is represented by figure 1. The system can be implemented by integrating the different modules such as data owner, data receiver, proxy server and private key generator. Each module is different from each other and has its own functionalities towards the system. The modules can be implemented by as follows.

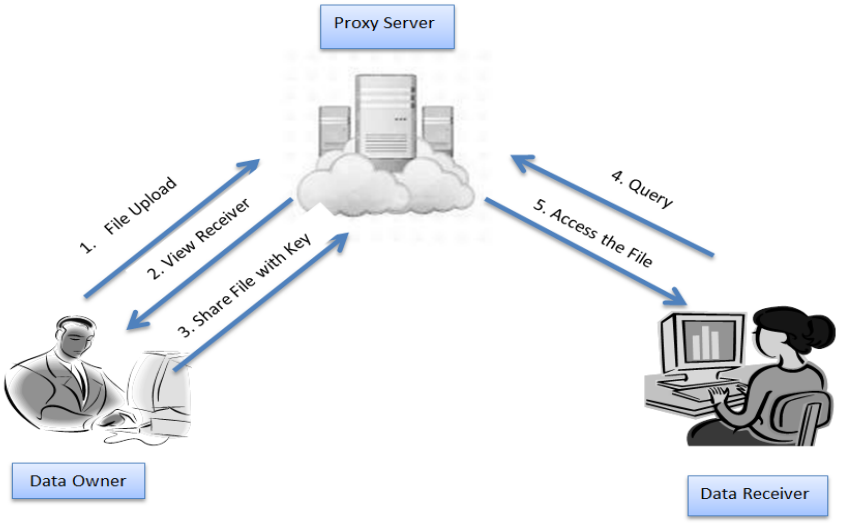

Fig. 1: System Architecture

A. Data Owner:

The new data owner registers with the proxy server and the server will allocate a part of cloud storage. The data owner is the one who owns the data or a particular file. Consider in an organization, top level employees want to share the sensitive data or files to bottom level employees. If the top level employee tries to share their sensitive data in the third party cloud server, anyone related to the organization can download the data, decrypt it and share it with others for their benefit. Therefore there is no security in the third party cloud storage. To avoid the misuse of data, the data owner will registers in our application and gets the login credentials.

In order to avoid the unauthorized usage of cloud resources, the login credentials of data owner must be same for both the proxy server and the application. Once the data owner attempts to login in the application, the proxy server validates the login credentials to check whether the data owner is authorized user for cloud storage or not. After logging in, the data owner has the permission to upload their file into proxy server. When the data owner is making the upload process our application will directly communicate with the proxy server and stores the file. Once the file is successfully uploaded, the data owner will see the list of employees who already registered in the application.

B. Proxy Server:

Proxy servers store the encrypted data and transfer the cipher text for the owner to the cipher text for the receiver when they obtain access permission (re-encryption key) from the owner. In these systems, proxy servers are assumed to be trusted. They authenticate receivers and validate access permissions. The interactions between the proxy servers and receivers are executed in a secure channel. Therefore, these systems cannot provide an end-to-end data security, namely they cannot ensure the confidentiality of the data stored at the proxy server. In these schemes, a receiver authenticates himself to the proxy server using his password. Then, the proxy server passes the authentication result to the file owner. The owner will make access permission according to the received information.

C. Data Receiver:

The receiver authenticates himself to the owner and decrypts the re-encrypted Ciphertext to obtain the data. In these systems, an end to-end security is provided by cryptographic protocols which are executed by the file owner to prevent proxy servers and unauthorized users from modifying and accessing the sensitive files. These systems can be divided into two types: shared file system and non-shared system. In shared file systems the owner can share his files with a group of users. Cryptographic techniques deployed in these systems are key sharing, key agreement and key revocation. In non-shared file systems in order to share a file with another user, the owner can compute an access key for the user using his secret key. In these two systems, the integrity of the sensitive files is provided by digital signature schemes and message authentication codes (MAC).

D. Private Key Generator:

The private key generator (PKG) validates the user's identities and issues secret keys to them. The key is generated and sent to their respective mail ids with the file name and the corresponding key values. To Create the Secret key 3DES algorithm is used.

V. RESULT

The table1 represents the seven different sizes of files and corresponding encryption execution time and decryption execution time taken by Triple-DES and RSA algorithms in milli seconds. By analyzing the table 1, it is concluded that the encryption time

taken by Triple-DES is relatively small as compared to RSA. RSA has taken the large encryption time as compared to Triple-DES. Similarly, the decryption time taken by Triple-DES is also relatively small as compared to RSA. And RSA has taken the large decryption time as compared to Triple-DES. Therefore, the symmetric key algorithm Triple-DES is more efficient by time when compared to the asymmetric key algorithm RSA

Table-1
Encryption time and Decryption time of different data packet size.

Input File Size (KB)	*Encryption Time(milli seconds)*		*Decryption Time(milli seconds)*	
	Symmetric Key Algorithm (Triple DES)	*Asymmetric Key Algorithm (RSA)*	*Symmetric Key Algorithm (Triple DES)*	*Asymmetric Key Algorithm (RSA)*
45	*50*	*55*	*49*	*61*
106	*76*	*89*	*63*	*57*
240	*113*	*119*	*67*	*64*
317	*155*	*157*	*85*	*154*
572	*171*	*169*	*161*	*163*
903	*299*	*309*	*171*	*183*
5417	*1166*	*1441*	*835*	*827*

VI. CONCLUSION

Distributed data storage schemes provide the users with convenience to outsource their files to untrusted proxy servers. Identity-based secure distributed data storage (IBSDDS) schemes are a special kind of distributed data storage schemes where users are identified by their identities and can communicate without the need of verifying the public key certificates. In this paper, we proposed two new IBSDDS schemes in standard model where, for one query, the receiver can only access one file, instead of all files. Furthermore, the access permission can be made by the owner, instead of the trusted party. Notably, our schemes are secure against the collusion attacks. The first scheme is CPA secure, while the second one is CCA secure. The proposed solution is aimed at designing a secured system against the collusion attacks. By this cryptosystem, we can achieve a stronger security and implement file based access control, the owner must be online to authenticate requesters and also to generate access permissions for them. The system can be further enhanced in future, by identifying whether the data receiver downloaded the file or not.

REFERENCES

[1] Jinguang Han,Willy Susilo and Yi Mu : Identity-Based Secure Distributed Data Storage Schemes, IEEE 2013.
[2] Cheng-Kang Chu, Sherman S.M. Chow, Wen-GueyTzeng, Jianying Zhou, and Robert H. Deng : Key- Aggregate Cryptosystem for Scalable Data Sharing in Cloud Storage, IEEE Transactions on Parallel and Disturbuted Systems, Volume 25, Number 2, Feb 2014.
[3] Hsiao-Ying Lin, Wen-GueyTzeng : A Secure Erasure Code-Based Cloud Storage System with Secure Data Forwarding, IEEE Transactions on Parallel and Disturbuted Systems, Volume 23, Number 6, June 2012.
[4] AdiShamir : Identity-based cryptosystems and signature schemes,Advances in Cryptology, CRYTO'84, LNCS 196, pp.47-53, 1985.
[5] NesrineKaaniche, AymenBoudguiga, MarylineLaurent : Identity-Based Cryptography for Secure Cloud Data Storage, June 2013.
[6] Hongwei Li, Yuanshun Dai, Bo Yang : Identity-Based Cryptography for Security, 2011.
[7] JinguangHan ,WillySusilo ,Yi Mu : Identity-based data storage in cloud computing, 2013.
[8] VarshaS.Agme, Prof ArchanaC.Lomte : Cloud Data Storage Security Enhancement Using Identity Based Encryption (IJAIEM)Volume 3, Issue 4, April 2014 ISSN 2319 – 4847.
[9] S.Komal Kate, S.D.Potdukhe : Data sharing in cloud storage with key-aggregate cryptosystem, International Journal of Engineering Research and General Science, Volume 2, Issue 6, October-November, 2014 ISSN 2091-2730 882 .
[10] Sowmiya Murthy : Cryptographic Secure Cloud Storage Model with Anonymous Authentication and Automatic File Recovery, October 2014.
[11] DanBoneh and Matt Franklin: Identity-Based Encryption from the Weil Pairing, CRYPTO'01, (2001) 213–229.
[12] Joye.M., Neven.G : Identity-Based Cryptography , Volume 2, Cryptology and Information Security Series.
[13] Aniket Kate and Ian Goldberg : Distributed Private-Key Generators for Identity-Based Cryptography.
[14] C.Cocks : An identity based encryption scheme based on quadratic residues, Eighth IMA International Conference on Cryptography and Coding, Dec 2001.
[15] DanBoneh, Xavier Boyen : Efficient Selective Identity-Based Encryption Without Random Oracles , March 21, 2011.
[16] Vishwagupta, Gajendra Singh, Ravindra Gupta : Advance cryptography algorithm for improving data security, January 2012.
[17] [AlexandraBoldyreva, VipulGoyal, Virendra Kumar : Identity-based Encryption with Efficient Revocation, 2008.
[18] Tang Q, Hartel PH, Jonker W : Inter-domain identity-based proxy re-encryption, 2008.
[19] Wang L, Wang L, Mambo M, Okamoto E : New identity-based proxy re-encryption schemes to prevent collusion attacks, 2010.
[20] SongLuo, Jianbin Hu, Zhong Chen : New Construction of Identity-based Proxy Re-encryption, 2010.
[21] Anca Ivan, YevgeniyDodis : Identity-based Proxy Re-encryption.
[22] G. Ateniese, K. Fu, M. Green ,and S. Hohenberger, "Improved proxy re-encryption schemes with applications to secure distributed storage",Inproc Network and Distributed System Security Symposium , pp. 1-15, The Internet Society, Feb. 2005.