Data Security Strategy Based On Multilevel File Encryption for Cloud Computing

Saem Ahmer
Department of CSE
SRM Institute of Science and Technology
Vadapalani,Chennai

Rahul Subhagan
Department of CSE
SRM Institute of Science and Technology
Vadapalani,Chennai

Dr. Bharathi
Department of CSE
SRM Institute of Science and Technology
Vadapalani,Chennai

***Abstract*— With the growing number of applications and data, secure storage of information becomes an essential part of our lives. A strong encryption system is required to protect the data stored. Encryption schemes have a significant disadvantage that the communication overheads and the computation costs are too high. And there is also a possibility that the data is lost after encryption. Hence, it is necessary to implement a "Data Encryption and Decryption" scheme with "Cloud Backup" feature which is more efficient in order to provide high level of data security.**

Properly segregating and maintaining this huge amount of data will require a reliable and structured system in which once the data enters should immediately be secure and stable. Most common methodology for storing files in a database is simple uploading or encrypted uploading. Rather than uploading a single encrypted file, it would be a safer option if the file is uploaded in multiple parts with separate individual encryption. So when the primary user uploads a file, it goes through a process of splitting and encryption before it gets stored into the database.

When the secondary user requests for the uploaded file, the access to that file is in the hands of the primary user who, based on his wish will grant or deny the access. Once the request is accepted, the decryption keys will be provided and the data parts will be merged to form the original file.

Keywords: DES, 3DES, AES, encryption, decryption, private key encryption, public key encryption, cryptography.

I. Introduction

In modern times, when information is available everywhere and it is more important than ever to make sure that the sensitive information does not fall into the wrong hands. Possessing data or a file is uncomplicated job, as a physical or a virtual storage system has the capacity to complete it. The complicated part is maintaining and accessing it in a secure way and finding out what is necessity. For that to be achieved, we need a system which can help us protect our data from various threats and vulnerabilities. Encryption and decryption are pretty common methodologies used these days for this purpose. Nowadays, these features are being utilized in almost all of the cloud related products. In this scenario, a system which supports user registration and a portal for file uploading is required. This when combined with the methodology of file splitting and encryption will give us the desired result. Even if a user with a malicious intent gets his hands on a part of the file, it's merely impossible for him to decrypt it as the algorithm and the keys which are completely unknown to him. But if a valid user wants to access the file, the permission can be granted by the uploader then the decrypting mechanism integrated in the system will automatically decrypt the split parts and the file merger will merge it into the required file while retaining its original structure.

II. Related Work

In the current era of modernization and connectivity where every kind of organization has deployed their work on cloud and at a time when security matters the most:

*Eng. Hashem H. Ramadan, Moussa Adamou Dja*milou in their paper described a similar concept by implementing more than one sort of encryption to the files where AES 256 and RSA were used. AES 256 is a strong encryption algorithm but the downflow is time complexity. To tackle that we have decided to stay with AES 128 and also AES 192 on some occasions along with multiple other algorithms such as DES and 3DES.

Bhaskar Prasad Rimal [1] delivered a classification of cloud computing and expand the current and new cloud systems. The objective of this paper was to create a disciplined procedure of scattered resources with least expenditure in command to acquire great throughput with comfort in cloud computing.

Yu-Sung Wu et.al. [7] discussed the concept of data storage in cloud system and purpose of this paper was to provide integrity to the cloud storage area and securely store or manage the data. It included some important security services like key generation, Encryption and Decryption in Cloud Computing system.

III. Proposed System

To tackle some of the common occurring vulnerabilities such as data theft, we have come up with a novel method to secure data storing on cloud.

In this project we will be uploading a file on the cloud platform the security of which will be provided by encrypting the data within the file. The required file will be encrypted and a key would be provided to the user for decryption. The file is be divided into 3 different parts of equal length using three different encryption algorithms to provide secure outline while maintaining the structural integrity of the file itself. The process can be classified into the following modules:

A. Data Encryption using Cryptography

Data encryption technology consults cryptography and other relevant technologies to replace or shift the clear text information by the encryption key and encryption function. It gets converted into the worthless cipher text, and is impossible to comprehend.

B. Proposed Users :

There are two types of users in our proposed system. First is the primary user i.e. the user uploading the file and the second is the secondary user who requests the file from the primary user. The primary user has access to the file and uploads the required file via login into the system. Once the file gets uploaded, the secondary user can view the file listed in the directory, available for request. If the secondary user wishes to have access to the file listed in the directory, he must send the request to the primary user. This request is sent to the primary user for approval, the user can either grant the request by approving the pending request or decline the proposed request.

This setup helps provide a direct approach for upload and downloading of specific files between the users with minimal interaction between the two.

C. File System

The file type which would be supported by the proposed system is ".txt" files. If a user wants to upload a file of some other type, conversion of the file into .txt format must be done. Only then would he be able to upload and send the required file.

D. File Splitting

The uploaded file will be split on the basis of file length, i.e. the entire file is read through a file buffer which will store the data in bits and thereby length of the text message is calculated. In this proposed system we will split the file into three parts to keep the operation feasible and quick. While the file is split, the system will make sure to retain the originality of the actual document. The structural integrity of the file remains the same.

E. Encryption

After the required file is divided into three segments, the next action in queue is encryption where there are three parts of the file now to get encrypted. For the system not to become too predictable, we have come up with some set of options for the primary user to choose from. One way would be to choose a standard set of encryption algorithms, i.e. the split parts will sequentially get encrypted through encrypting algorithms in the order of AES (128 bit), DES and 3DES.

The user has another option, where the user determines the type and the level of security for the file. This provides a variable level of data security based upon the preference of the user. Based upon the choice of the user, the algorithm is determined.

Five encryption algorithms will be available to opt from. They are AES(128 bit), AES(192 bit), AES(256 bit), DES and 3DES. The cost of encryption is higher for the latter as the keys are much stronger for AES(192 bit) and the very popular AES(256 bit).

Algorithms Used:

1. Advanced Encryption Standard :

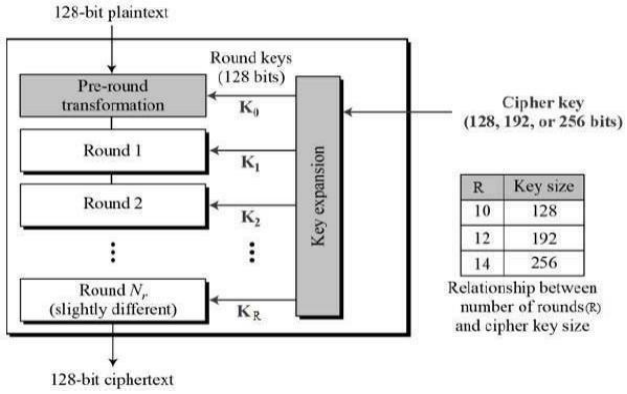
AES is an iterative rather than Feistel cipher. It is based on ‘substitution–permutation network’.

It comprises of a series of linked operations, some of which involve replacing inputs by specific outputs (substitutions) and others involve shuffling bits around (permutations).

Interestingly, AES performs all its computations on bytes rather than bits. Hence, AES treats the 128 bits of a plaintext block as 16 bytes. These 16 bytes are arranged in four columns and four rows for processing as a matrix –

Unlike DES, the number of rounds in AES is variable and depends on the length of the key. AES uses 10 rounds for 128-bit keys, 12 rounds for 192-bit keys and 14 rounds for 256-bit keys. Each of these rounds uses a different 128-bit round key, which is calculated from the original AES key.

The schematic of AES structure is given in the following illustration –

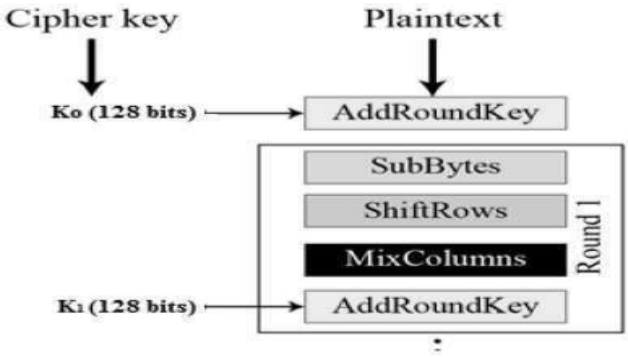


R	Key size
10	128
12	192
14	256

Relationship between number of rounds(R) and cipher key size

Encryption

Here, we restrict to description of a typical round of AES encryption. Each round comprise of four sub-processes. The first round process is depicted below –



Byte Substitution

The 16 input bytes are substituted by looking up a fixed table (S-box) given in design. The result is in a matrix of four rows and four columns.

Shiftrows

Each of the four rows of the matrix is shifted to the left. Any entries that ‘fall off’ are re-inserted on the right side of row. Shift is carried out as follows −

- First row is not shifted.
- Second row is shifted one (byte) position to the left.
- Third row is shifted two positions to the left.
- Fourth row is shifted three positions to the left.

- The result is a new matrix consisting of the same 16 bytes but shifted with respect to each other.

MixColumns

Each column of four bytes is now transformed using a special mathematical function. This function takes as input the four bytes of one column and outputs four completely new bytes, which replace the original column. The result is another new matrix consisting of 16 new bytes. It should be noted that this step is not performed in the last round.

Addroundkey

The 16 bytes of the matrix are now considered as 128 bits and are XORed to the 128 bits of the round key. If this is the last round then the output is the cipher text. Otherwise, the resulting 128 bits are interpreted as 16 bytes and we begin another similar round.

Decryption Process

The process of decryption of an AES cipher text is similar to the encryption process in the reverse order. Each round consists of the four processes conducted in the reverse order –

- Add round key
- Mix columns
- Shift rows
- Byte substitution

Since sub-processes in each round are in reverse manner, unlike for a Feistel Cipher, the encryption and decryption algorithms need to be separately implemented, although they are very closely related.

2. Data Encryption Standard:

A 16-round Feistel cipher with block size of 64 bits. DES stands for Data Encryption Standard.

DES was developed by IBM in 1974 in response to a federal government public invitation for data encryption algorithms. In 977, DES was published as a federal standard, FIPS PUB 46.

DES algorithm:

```
Input:
        T: 64 bits of clear text
        k1, k2, ..., k16: 16 round keys
        IP: Initial permutation
        FP: Final permutation
        f(): Round function

Output:
        C: 64 bits of cipher text

Algorithm:
        T' = IP(T), applying initial permutation
        (L0, R0) = T', dividing T' into two 32-bit
        parts
        (L1, R1) = (R0, L0 ^ f(R0, k1))
        (L2, R2) = (R1, L1 ^ f(R1, k2))
        ......
        C' = (R16, L16), swapping the two parts C
        = FP(C'), applying final permutation
```

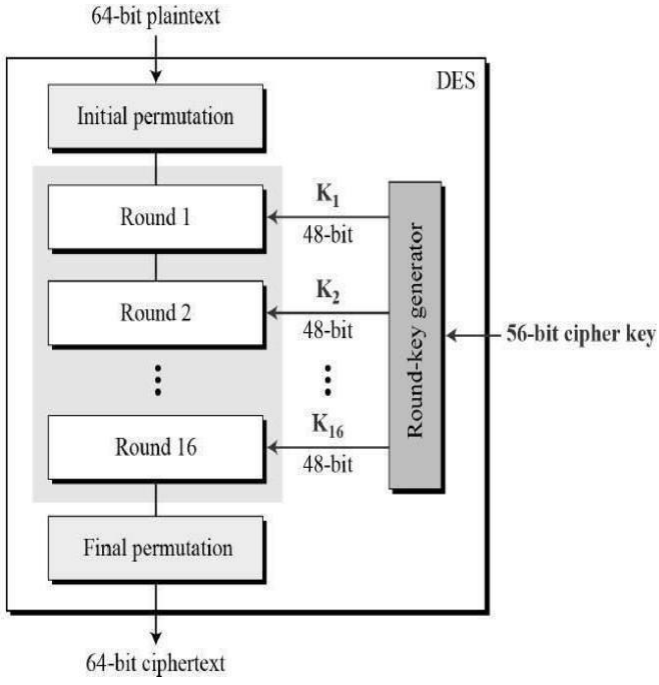
where ^ is the XOR operation.

The round function f(R,k) is defined as:

```
Input:
        R: 32-bit input data
 k: 48-bit round key
 E: Expansion permutation
 P: Round permutation
 s(): S boxes function

Output
        R' = f(R,k): 32-bit output data

Algorithm
 X = E(R), applying expansion permutation
 X' = X ^ k, XOR with the round key
 X" = s(X'), applying S boxes function and
 returning 32-Bit data
 R' = P(X"), applying the round permutation
```

The S boxes function s(X) is defined as:

```
Input:
 X: 48-bit input data
 S1, S2, ..., S8: 8 S boxes - 4 x 16 tables

Output:
  X' = s(X): 32-bit output data

Algorithm:
  (X1, X2, ..., X8) = X
   X' = (S1(X1), S2(X2), ..., S8(X8))
     r = 2*b1 + b6
     c = 8*b2 + 4*b3 + 2*b3 + b4
     b1, b2, b3, b4, b5, b6 are the 6 bits of the Xi.
```



General Structure of DES

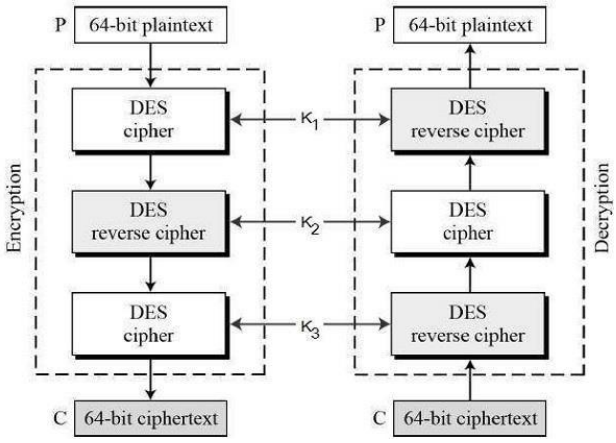
3. Triple Data Encryption Standard:

3DES is an encryption cipher that was derived from the original Data Encryption Standard (DES). It became prominent in the late nineties, but has since fallen out of favor due to the rise of more secure algorithms.
Although it will be deprecated in 2023, it's still implemented in some situations.

Once the weaknesses of normal DES became more apparent, 3DES was adopted in a wide range of applications. It was one of the more commonly used encryption schemes before the rise of AES.

As the security weaknesses of DES became more apparent, 3DES was proposed as a way of extending its key size without having to build an entirely new algorithm. Rather than using a single key as in DES, 3DES runs the DES algorithm three times, with three 56-bit keys:

- Key one is used to encrypt the plaintext.
- Key two is used to decrypt the text that had been encrypted by key one.
- Key three is used to encrypt the text that was decrypted by key three.



Once the weaknesses of normal DES became more apparent, 3DES was adopted in a wide range of applications. It was one of the more commonly used encryption schemes before the rise of AES.
Some examples of its implementations included Microsoft Office, Firefox and EMV payment systems. Many of these platforms no longer use 3DES because there are better alternatives.

F. Cloud Database

This encrypted data is now stored in the cloud database, the required database can be accessed and modified from within. This file is stored in an encrypted format i.e. rending the whole file useless at that moment.

The database contains the user login credentials as well as stores the parts of the file using AWS RDS.

G. Decryption

When the secondary user makes a request for the file, upon approval from the primary user the file can be accessed directly from the directory via the secondary user's login.

The file is divided into three parts and present in a form that is cryptic and not understandable. Using the decryption process one can view the data divided into three different parts systematically. This decryption process is different for each algorithm, thus providing the secondary user with the access to the required files.

H. Merging

Now the secondary user can view the entire file in front of him in three different parts, these parts must be merged in order to form a relevant file.

For this part to be accomplished, a simple file operation must be performed on the pre-existing decrypted parts. The file operation helps merge these parts of text into a single text file and lets the secondary user download this file.

This helps provided a bridge between the primary user and users who wishes to attain the files uploaded by him.

I. File Integrity

Hashing algorithms such as SHA-1 could be used to hash the file content before uploading. So that, before providing the access of the encrypted file to the secondary user, the final merged content could be hashed again to compare the values. If the hash matches, then the file can be termed as genuine.

IV. CONCLUSION AND FUTURE SCOPE

Security is the main challenge, requirement and aspect defined for a Cloud System. This security system provides the file management, forwarding and storage in encoded form. The authorization, authentication and encoded storage are also provided in distributed Cloud storage. In this present research, a more functional, secure and reliable security system is provided.

As the security is the main concern in distributed cloud environment, in this present research a secure Cloud System is provided. This proposed security system environment provides the multiple storage and security features. The following way may enhance the future scope of the project:

- The present system is defined specifically for public and private cloud access in hybrid and generalized environment, in the future the proposed system can be applied to an organization.
- The present work is implemented on a generic secure cloud system without specification of any attack, in the future the work model can be implemented on some attack specific security in a cloud environment.
- The encryption model can be extended and applied for more complex cloud system architectures such as mobile cloud or green cloud environment.
- For now the file type is limited to .txt format, while there are many more formats which can be looked into for future extensions.

V. REFERENCES

[1] Bhaskar Prasad Rimal; Eunmi Choi; Ian Lumb, (2009) taxonomy and Survey of Cloud Scheming Systems", IEEE Fifth International "A Joint Conference on INC, IMS, and DC,vol.10, No.2, pp. 44-51.

[2] Amichai Shulman; Top Ten Database Security Threats, 2006 White Paper.

[3] Tanya Bacca; Making Database Security an IT Security priority A SANS Whitepaper – November 2009.

[4] Kadhem, H.; Amagasa, T.; Kitagawa, H.; A Novel Framework for Database Security based on Mixed Cryptography; Internet and Web Applications and Services, 2009. ICIW '09. Fourth International Conference on; Publication Year: 2009.

[5] Zhifeng Xiao and Yang Xiao, (2013) "Security and Privacy in Cloud Computing", IEEE Communications Surveys & Tutorials, Vol. 15, No. 2, pp.843- 859 .

[7] Yu-Sung Wu; Bingrui Foo , (2013) "Amazon Web Services: Overview of Security Processes", International Journal of Network Security, Vo.12, No.1, pp.822-866 .

[5] Rajkumar Buyya, (2013), "Introduction to the IEEE on Cloud Computing", IEEE Transactions On Cloud Computing, Vol. Transactions 1, No.1, ppt. 3-5.