International Journal of Security and Its Applications
Vol.8, No.3 (2014), pp. 377-392
http://dx.doi.org/10.14257/ijsia.2014.8.3.37

Metadata Driven Efficient CRE based Cipher Key Generation and Distribution in Cloud Security

R. Anitha and Saswati Mukherjee

Department of Information Science and Technology,
Anna University, Chennai, India
anitabalajim@yahoo.com

Abstract

The increasing popularity of cloud storage services has lead companies that handle critical data to think about using these services for their storage needs. To keep their sensitive data against untrusted cloud service providers, a natural way is to store only encrypted data in the cloud severs and providing an efficient access control mechanism.. The proposed model involves the attributes of metadata for key generation and distribution techniques. The key problems of this approach includes, the generation of cipher key and establishing an access control mechanism for the encrypted data using cipher key, where keys cannot be revoked without the involvement of user, metadata data server (MDS) and Data Server (DS). From this study, we propose a novel metadata driven cipher key generation and distribution policies by means of exploiting the characteristic of the metadata stored called CRE Scheme, a Cloud Re-Encryption model that improves the confidentiality of the data stored through the cipher-key C_{mxn}. We have implemented our security model using eucalyptus tool and evaluated the performance and scalability of the secured model. We observed that our protocols improved the security of the data stored and compared with existing security models.

Keywords: *cloud security, secured bloom filter, metadata, feistel network*

1. Introduction

Cloud computing has become the most attractive field in industry and in research. The requirement for cloud computing has increased in recent days due to the utilization of the software and the hardware with less investment [5].A recent survey regarding the use of cloud services made by IDC, highlights that the security is the greatest challenge for the adoption of cloud computing technology [6]. The four key components of data security in cloud computing are data availability, data integrity, data confidentiality, and data traceability. Data traceability means that the data transactions and data communication are genuine and that the parties involved are said to be the authorized persons [7]. Several studies shows that data traceability mechanism have been introduced, ranging from data encryption to intrusion detection or role-based access control, doing a great work in protecting sensitive information. However, the majority of these concepts are centrally controlled by administrators, who are one of the major threats to security [8]. In some modern distributed file systems, data is stored on devices that can be accessed through the metadata, which is managed separately by one or more specialized metadata servers [1]. Metadata is a data about data and it is structured information that describes, explains, locates, and makes easier to retrieve, use, or manage an information resource. The metadata file holds the information about a file stored in the data servers. In cloud computing, the users will give up their data to the cloud service provider for storage. The data owners in cloud computing environment want to make sure that their data are kept confidential to outsiders, including the cloud service provider which will be the major data security requirement. In the existing system, an authentication is done using cloud user's identity and must be validated by the central authority, called cloud service providers. When the cloud service provider is malicious unauthorized users can also be impersonated. Hence we are facing a major issue with Key-Generation and Key handling problem. When the secret key is generated in a single space the system can be easily attacked. Hence in order to overcome these issues, in the proposed cipher keys are generated using the metadata attributes and key handling mechanism hence there doesn't have any centralized control over the encryption and decryption technique. The specification of deciding the key is based on the metadata attribute in the metadata server as well as the user key. Most of the existing cloud encryption schemes are constructed on the architecture where a single trusted (TPA) third party authority has the power to secure the secret data stored at the cloud servers. The major drawback of the prevailing system is that the data stored is not much secure because the entire security is taken care by a single space. Hence in the proposed system, the key generation and issuing protocol is handled by User, MDS and DS. The model also makes data owner confident about the complete security of the data stored, since the encryption and decryption keys cannot be compromised without the involvement of data owner and the MDS. Based on the above-mentioned analysis, it is needed to propose a secure data-sharing scheme, which simultaneously achieves high performance, full delegation, and scalable revocation.

The rest of the paper is organized as follows: Section 2 summarizes the related work and the problem statement. Section 3 describes the system architecture model and discusses the detailed design of the system model. Section 4 describes the modified feistel network structure design and issues of the proposed model. Section 5 explains about generation of SBF indexing. Section 6 explains about the data security at the data server location. The performance evaluation based on the prototype implementation is given in Section 7 and Section 8 concludes the paper.

2. Related Work

The related work discusses about the previous work carried out in the area of cloud security and we have also discussed about how metadata is used in cloud computing environment.

2.1. Metadata in Distributed Storage Systems

Recently a large amount of work is being pursued in data analytics in cloud storage [1] [2]. Abhishek Verma *et al.*, [3] have proposed metadata using Ring file system. In this scheme metadata for a file is stored based on hashing its parent location. Replica is stored in its successor metadata server. Yu Hua *et al.*, [4] have proposed a scalable and adaptive metadata management in ultra large scale file systems. Michael Cammert *et al.*, [1] divided into two types: static and dynamic metadata. The author has suggested publish-subscribe architecture, enabled a SSPS to provide metadata on demand and handled metadata dependencies successfully. R.Anitha *et al.*, [5] has described that the data retrieval using metadata in cloud environment is less time consuming when compared to retrieving a data directly from the data server.

2.2. Security Schemes

Chirag Modi *et al.*, [15] proposed a survey paper where they discussed about factors affecting cloud computing storage adoption, vulnerabilities and attacks. The authors have also identified relevant solution directives to strengthen security and privacy in the cloud environment. They further discuss about various threats like abusive use of cloud computing, insecure interfaces, data loss and leakage, identity theft and metadata s spoofing attack. J. Ravi Kumar *et al.*, [14] shows that third party auditor is used to periodically verify the data integrity for the data stored at cloud service provider without retrieving the original data. The security is provided by creating the metadata for the encrypted data. Shizuka Kaneko *et al.*, [19] have proposed a query based hiding schema Information using a Bloom filter. The query given is processed and the attributes of the query is used for key generation. The generated key is used to hide confidential information. Marcos K. Aguilera *et al.*, [20] has proposed a practical and efficient method for adding security to network-attached disks (NADs). The design specifies a protocol for providing access to the remote block-based devices. The security is provided by means of access control mechanism.

2.3. Bloom Filter Schemes

The Bloom filter is a space-efficient probabilistic data structure that supports set membership queries [14]. The data structure was conceived by Burton H. Bloom in 1970. The structure offers a compact probabilistic way to represent a set that can result in false positives (claiming an element to be part of the set when it was not inserted), but never in false negatives (reporting an inserted element to be absent from the set). This makes Bloom filters useful for many different kinds of tasks that involve lists and sets. The basic operations involve adding elements to the set and querying for element membership in the probabilistic set representation. Shizuka Kaneko *et al.*, [19] has discuss about the usage of bloom filter in query processing.

2.4. Steganography Security Schemes

Wawge P.U. *et al.*, [23] describes that steganography comes from the Greek words Steganos (Covered) and Graptos (Writing). The term steganography came into use in 1500‘s after the appearance of Trithemius book on the subject Steganographia. The word steganography technically means covered or hidden writing. The proposed new data hiding scheme byusing matrix matching method.On this basis of matching factor of columns, particularbits may be changed such that change in image quality is minimum. Thus the original content is hidden [24]. Sharon Rose Govada *et al.*, [25] in the year 2012 proposed text steganography with multi level shielding where he proposed a method which is capable of performing text steganography that is more reliable and secure when compared to the existing algorithms. The method is a combination of word shifting, text steganography and synonym text steganography. Nirmalya Chowdhury *et al.*, [26] has proposed an efficient method of steganography using matrix approach. He has discussed that the goal of steganography is to hide messages inside other ‘harmless’ messages in a way that does not allow any enemy to even detect that there is a second message present. Least significant bit (LSB) insertion is a common and simple approach to embed information in a cover object which he has used. The design uses a matrix based steganography which modifies the bit inside the matrix by means of adding and modifying.

3. System Model

The architecture diagram of the proposed system model is shown in Figure 1. Each block in the architecture explains about how the data is encrypted and how the keys are distributed between the user, MDS and the DS.

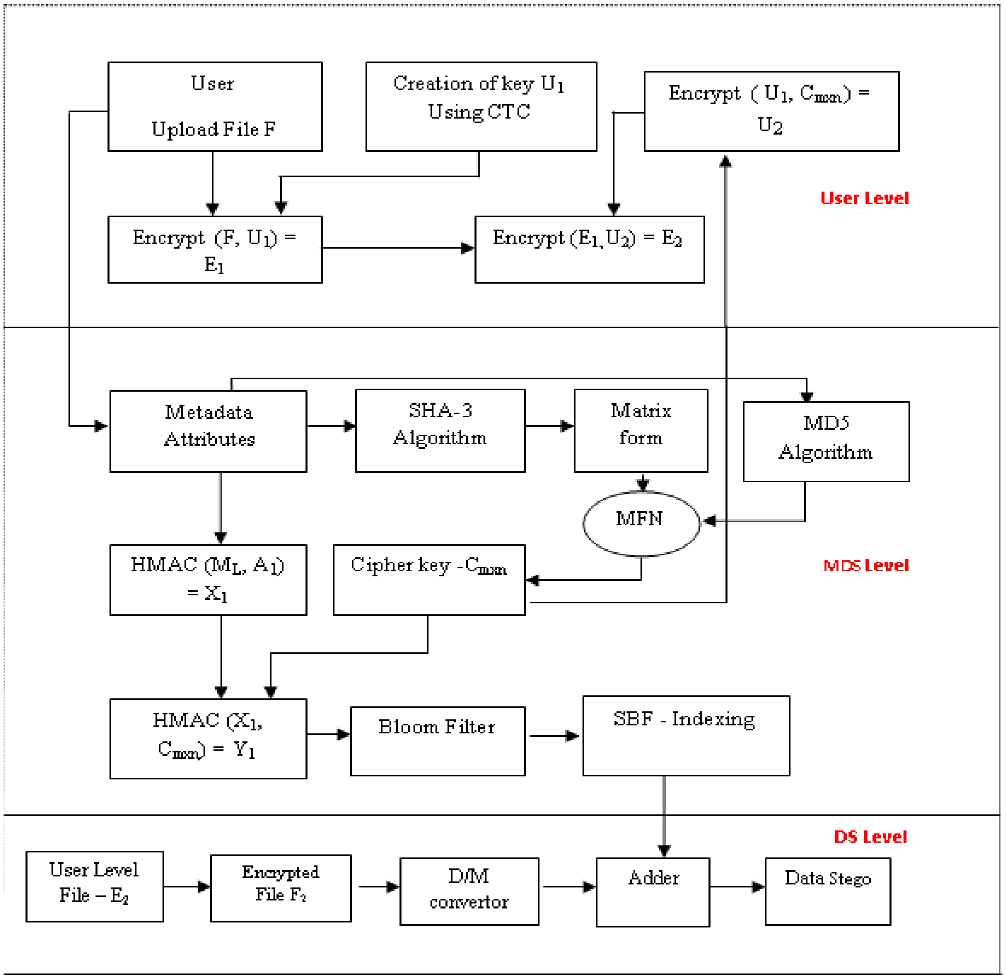


Figure 1. Architecture Diagram

The system model proposes security to the data using modified feistel network where the metadata attributes are taken as input in the form of matrices. In this model the user uploads the encrypted file using the key X_1. The metadata for the file is created and based on the metadata created, attributes of the cipher key C_{mxn} is created. The metadata server sends the cipher key C_{mxn} to the user. Using C_{mxn} as key the user encrypts the key X_1 and generates $X_{2.}$ While downloading the file the key X_2 and C_{mxn} is used to retrieve X_1 and file is decrypted. This model proposes a modified Feistel function F which introduces the matrix operations like transpose, shuffle, addition and multiplication along with the key matrix. The cryptanalysis carried out in this paper clearly indicates that this cipher cannot be broken by the brute force attack. This model provides high strength to the cipher, as the encryption key induces a significant amount of matrix obfuscation into the cipher. The avalanche effect discussed shows the strength of the cipher C_{mxn}. The secured bloom filter indexing is used to generate the stego data in order to prevent the data at the data server location. The data stego is generated using the secured bloom filter index value and the user key K. Thus the original data is made secured at the data server location. The proposed system model also ensures that the data is identically maintained by making use of the cipher key C during any operation like transfer, storage, or retrieval.

The process flow diagram of the proposed system model is shown in Figure 2a and Figure 2b. The process flow diagram explains about how the keys are shared between the user, MDS and the DS.

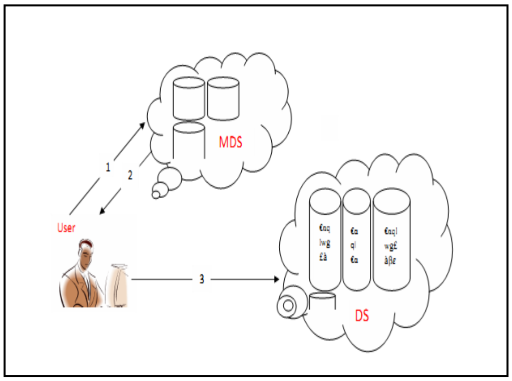


Figure 2a. Process Flow Diagram for Uploading User to Cloud Servers

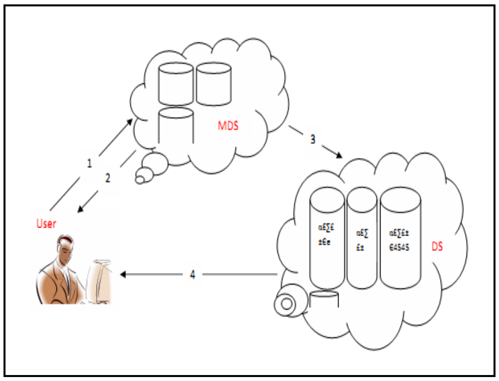


Figure 2b. Process Flow Diagram for File from Downloading File from Cloud to User

The functionalities of the process flow diagram of while uploading is as described below.1. Uploads the Metadata attributes 2. Mechanism for Key Generation and Cipher Key are transferred from cloud to user. 3. Uploads the Re-Encrypted File to cloud server. The main functionalities of the process flow diagram of while downloading the file is illustrated as 1.Uploads the File name and C_{mxn} 2. MDS authorize the user and sends SBF indexing 3. Transfer's Request from MDS to DS 4. Transfer Re-Encrypted file to User.

3.1. File Access

When a user sends request for data stored on the cloud environment, the request is given to the metadata server which provides the recent cipher key C_{mxn} to the user. Using C_{mxn} user decrypts the key X_2 and gets X_1. Using X_1 the encrypted file from the cloud storage is decrypted to get the original data. By providing the recent cipher key C_{mxn} the data integrity is also verified. Our system methodology uses the functionalities

1. File Uploading.
2. Data Pre processing.
3. Construction of modified feistel network
4. Generation of Cipher key C.
5. Generation of Secured Bloom filter Index.
6. Creation of Data stego.

4. Modified Feistel Network

Feistel ciphers are a special class of iterated block ciphers where the cipher text is calculated from the attributes of metadata by repeated application of the same transformation or round function.

4.1. Development of the Cipher Key "C_{mxn}" using Modified Feistel Function

In this paper we propose a complex procedure for generating the cipher key "C_{mxn}" based on matrix manipulations, which could be introduced in symmetric ciphers. The proposed cipher key generation model offers two advantages. First, the procedure is simple to implement and has complexity in determining the key through crypt analysis. Secondly, the procedure produces a strong avalanche effect making many values in the output block of a cipher to undergo changes with one value change in the secret key. As a case study, matrix based cipher key generation procedure has been introduced in this cloud security model and key avalanche have been observed. Thus the cloud security model is improved by providing a novel mechanism using modified Feistel network where the cipher key C_{mxn} is generated with the matrix based cipher key generation procedure. The Cipher key generation procedure is based on a matrix initialized using secret key and the modified feistel function F. The input values used in various feistel rounds are taken from the previous round. The selection of rows and columns for the creation of matrix is based on the number of attributes of the metadata and the secret key matrix " K_{mxn}" and the other functional logic as explained in the following subsections.

4.1.1. Data Preprocessing: Data preprocessing is a model for converting the metadata attributes into matrix form using the SHA-3 cryptographic algorithm, containing m rows and n columns, where m is the number of attributes of the metadata and n takes the size of the SHA-3 output. The matrix is splitted into 4 equal matrix say m_1, m_2, m_3 and m_4. The matrix obfuscation is carried out in order to make the hacker opaque. The matrices m_1, m_3 and m_2, m_4 are concatenated. This obfuscated matrix is fed as input to the feistel network structure where concatenated value of m_1, m_3 will be the left value and m_2, m_4 be the right value of the feistel network.

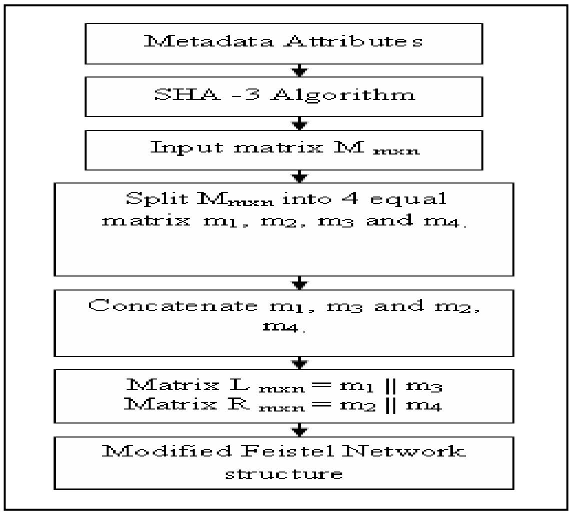

Metadata Attributes
SHA -3 Algorithm
Input matrix M mxn
Split Mmxn into 4 equal matrix m1, m2, m3 and m4.
Concatenate m1, m3 and m2, m4.
Matrix L mxn = m1 || m3
Matrix R mxn = m2 || m4
Modified Feistel Network structure

Figure 3. Model for Data Preprocessing

4.1.2. Modified Feistel Network Structure: The Matrix L_{mxn} which is a concatenated value of m1 || m3 is considered as the left value of the feistel network structure and Matrix R_{mxn} = m2 || m4 is considered as the right value of the feistel network structure. Using MD5 cryptographic hash algorithm the key matrix Kmxn is generated whose size is m x n where "m" is the number of attributes of metadata and "n" is the size of the MD5 algorithm. The development of the cipher key in the feistel network is done through the number of rounds until the condition is satisfied. In this symmetric block ciphers, matrix obfuscation operations are performed in multiple rounds using the key matrix and the right side value of the feistel network structure. The function F plays a very important role in deciding the security of block ciphers. The concatenated value of L_{mxn} and R_{mxn} in the last round will be the cipher key C_{mxn}. Figure 4 below represents the one round modified feistel network structure.

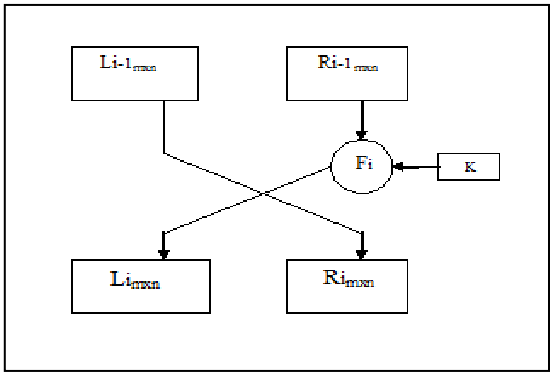


Figure 4. One Round of Modified Feistel Network

Definition of feistel function F: Let R be a function variable and let K be a hidden random seed, then the function f is defined as, F(R, K) = $F_K(R)$ where F is a modified feistel function. The procedure for developing the function is described below. Each round has its own feistel function F. The function f is considered to be varied based on the right side value of the feistel network *i.e.*, the function F is indexed by the matrix R_{mxn} for that round. In this modified feistel network structure, the function for each round depends on the previous round *i.e.*,

$$\text{Round}_i (L_i, R_i) = (R_{i-1}, F(K, L_{i-2}))$$

The above formula shows that a small change in one round affects the entire feistel network. For each round as the value of R of the network gets compressed at some point in time the feistel round automatically stops based on the size of the attributes.

The procedure for deriving function F is explained in steps as follows:

Algorithm 1: Creation of cipher key

```
Begin
 1.  Read Metadata attribute
 2.  Apply SHA-3
 3.  Generate Matrix $M_{mxn}$, split the matrix, and generate $L_{mxn}$ and $R_{mxn}$
 4.  Left value of Feistel = $L_{mxn}$ and Right value of Feistel = $R_{mxn}$
          For i = 1 to n Repeat till n / 2 = 1
   Begin
    4.1 Split $R_{mxn}$ into equal matrix, $R_{1mxn}$, $R_{2mxn}$
    4.2 Transpose $R_{1mxn}$, $R_{2mxn}$ as $R_{1nxm}$, $R_{2nxm}$
    4.3 Apply matrix addition of $R_{1nxm}$, $R_{2nxm}$ = $T_{mxn}$
    4.4 Transpose $T_{mxn}$
    4.5 Matrix multiplication of $T_{nxm} * K_{mxn}$ = $RV_{mxn}$
          / *condition for multiplication is verified */
    4.6 New $L_{mxn}$ = $RV_{mxn}$
    4.7 New $R_{mxn}$ = Old value of $L_{mxn}$
  End
5.  Repeat the step till n takes odd value
6. Write(C) Cipher key C = $L_{mxn}$ || $R_{mxn}$ / *|| represents concatenation */
End
```

Algorithm 2: Creation of modified feistel function-MFN

```
Begin
1) Read Matrices $L_{mxn}$ and $R_{mxn}$
                a. Assign Left value = $L_{mxn}$
                b. Right value = $R_{mxn}$
2) Split $R_{mxn}$ = $R_{1 mxn}$ and $R_{2mxn}$.
3) Transpose ($R_{1mxn}$) = $R_{1 nxm}$
4) Transpose ($R_{2mxn}$) = $R_{2 nxm}$
5) $T_{nxm}$ = $R_{1nxm}$ + $R_{2 nxm}$.
6) $R_{V mxn}$. = $T_{nxm}$ * $K_{mxn}$
7) Re - Assign
        a. $L_{mxn}$ = $R_{V mxn}$
        b. $R_{mxn}$ = $L_{mxn}$
8       Go to Step 2 till n = odd value.
End
```

The above formula shows that a small change in one round affects the entire feistel network. For each round as the value of R of the network gets compressed at some point in time the feistel round automatically stops based on the size of the attributes. The feistel network proposed in this chapter is a four round feistel function which holds good for cipher strength as discussed by avalanche effect. The feistel network is a mapping of (f_1, f_2, f_3, f_4) : $\{x_i\}^{2n} \rightarrow \{x_i\}^{2n}$, "$i$" is the value of the row value consisting of four consecutive rounds. $\varphi(f_1, f_2, f_3, f_4) = \varphi_1 . \varphi_2 . \varphi_3 . \varphi_4$ where each round is presented as $\varphi_i : \{x_i\}^{2n}$ and hence the function $\varphi_i(L_i, R_i) = (R_i, L_i \oplus f_i(R_i))$ where L_i, R_i are the left and right value of the matrix.

5. Generation of Secured Bloom Filter Index

The second level of security in the metadata layer which is provided using the secured bloom filter look up table. The generation of the look up table is as shown in Figure 5. A Secured bloom filter index is created based on the value of cipher key C_{mxn} and key K which is X_2 of the user using the attribute of metadata. The $HMAC_1$ is applied for every attribute A of the metadata created using the key from the user and the output of the first level is again applied for $HMAC_2$ using the cipher key C_{mxn} hence the index creation cannot be compromised without the involvement of the user and the metadata attributes.

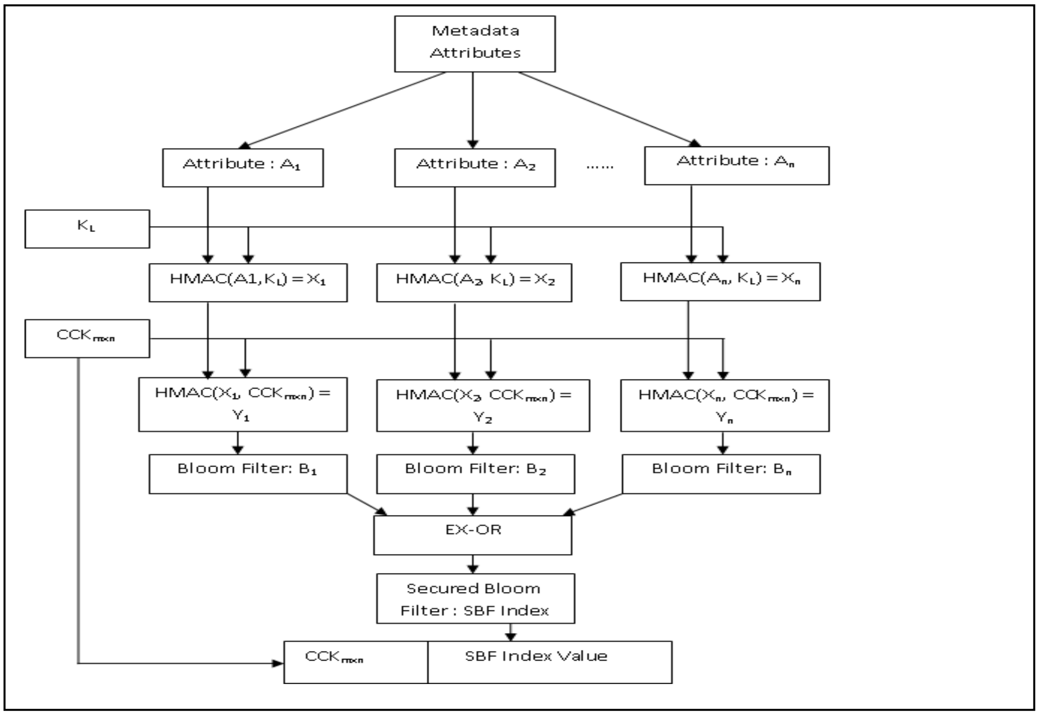


Figure 5. Secured Bloom Filter Index Generation

6. Data Steganography at Data Server Location

This section explains about the data security at the data server location. As the data in cloud is kept in the data server which is away from the user the security of data at rest plays a major role. The generation of stego data is as shown in the Figure 6. The original data is converted into data stego at the time of storing the data. The conversion process is carried out in 4 steps. 1. Data to Matrix converter 2. Matrix is added with SBF value 3. The output is added with the key from the user 4. Matrix to Data convertor. Thus the Original data is divided into data stego and uploaded to the data server location. Certain mathematical operations like converting the data block into matrix form and by using SBF, the original matrix is modified. To hide the original information, straight message insertion may transform every bit value of original information *i.e.*, embedding some bit values to the original value. Each of these techniques is applied to provide security to the data, by hiding the original data. For steganography, we have used matrix operations in order to hide the original information.

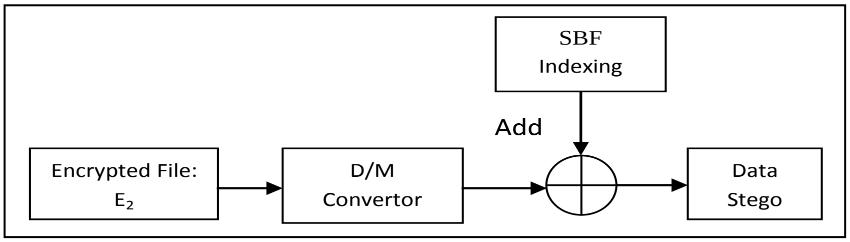


Figure 6. Generation of Stego Data

Algorithm to generate Data Stego

```
Begin
        1. Get the encrypted file.
        2. Convert the file into matrix form.
        3. Add SBF Indexing to the matrix.
        4. Resultant file = Data stego
End
```

7. Analysis of proposed Cloud based Re-Encryption (CRE) Scheme

Let us illustrate the motivation of the CRE model as the data is stored under semi trusted cloud environment.

Case 1: Data Owner Offline

The user in the cloud encrypts the data with their own cipher key U_1. When the user remains offline, there is a chance that the CSP may share the sensitive data to other malicious users without the data owner's knowledge. Hence in order to handle this situation the data owner encrypts the data with their own cipher key.

Case 2: Malicious User Online

As the CSP is unaware of the data owner's information, there is a chance that malicious user may enter with an exact cipher key so that the data is shared by the cloud service provider to the malicious user. Therefore, in order to handle this situation the proposed model has introduced a concept of hidden process where the mechanism of retrieving the plain text from the cipher text is unknown to the unauthorized person. In order handle the above mentioned situations the cloud re-encryption scheme (CRE) scheme proposes an efficient key distribution mechanism so that without the active involvement of data owner and the cloud service provider the keys cannot be compromised. The CRE scheme consists of the following algorithms:

Key Distribution: Based on the file uploaded, the data owner generates a user level cipher key U_1. $\mathrm{KeyGen(CTC)} \longrightarrow \mathrm{U_1}$. Making use of choosing U_1 the original file is encrypted. $\mathrm{Encrypt}\ (\mathrm{F}, \mathrm{U_1}) \longrightarrow \mathrm{E_1}$. In the same way, the MDS at the cloud service providers location generates a cipher key C_{mxn}. $\mathrm{KeyGen(MDS)} \longrightarrow \mathrm{C_{mxn}}$ and is distributed to the user level where the encrypted file is re-encrypted using C_{mxn}. The C_{mxn} is used to generate U_2 by using a master key k_1. $\mathrm{Encrypt}\ (\mathrm{E_1}, \mathrm{U_2}) \longrightarrow \mathrm{E_2}$.

Decryption: Given the request, the MDS authorize the user based on SBF indexing and sends the request to the data server and the encrypted file is sent back to the user and user decrypts the cipher text and retrieves the original file. Hence the keys are distributed efficiently so that the data cannot be compromised without the involvement of all the three parties.

As such the CRE scheme has the entire control over the security of the data stored at cloud environment based on the user's secret keys and keys generated from CTC and MFN networks. Finally, an implementation is used to validate the performance of the model. The figure 7 represents the key distribution mechanism between the user, MDS and the DS.

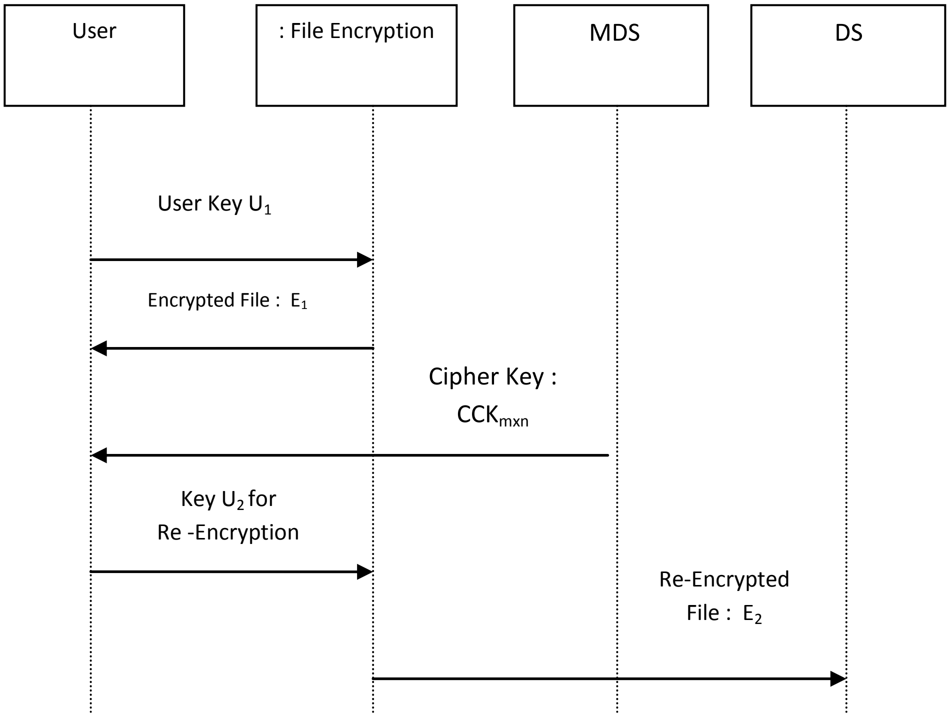


Figure 7. Key Distribution Mechanism using CRE Scheme

The proposed cloud based re-encryption model is efficient and secured enough, as cipher keys are generated and managed by the user and also by the cloud service providers. The concept of re-encryption secures the data under different cases. At the initial stage, the user encrypts the data with their own key and the data is re-encrypted data using CSP cipher key which is generated by the cloud service provider, and key distribution is efficiently handled and also conserves the communication costs of key transfer. The key distribution proposed is a novel solution that entails a key management based on re-encryption by CSP that effectively utilizes the metadata stored at the cloud environment computing the cipher key. As the cipher key generation supports a frequently changing metadata attributes, security is improved.

8. Implementation and Results Discussion

The experiments have been carried out in a cloud setup using eucalyptus which contains cloud controller and walrus as storage controller. These tests were done on 5 node cluster. Each node has two 3.06 GHz Intel (R) Core TM Processors, i-7 2600, CPU @ 3.40GHZ, 4 GB of memory and four 512 GB hard disks, running Eucalyptus. The tests used 500 files of real data set, uploaded into the storage and then downloaded based on the user's requirement. The experimental results show that the model provides a complex cipher key C_{mxn} which adequately strengthens the data stored. Results demonstrate that our design is highly complex in nature and the time taken for generating the cipher key is less compared to the existing algorithms. Performance Analysis metrics is done based on the experimental set up. To the best of the domain knowledge obtained due to a wide literature survey on cloud-based performance analysis methodologies and tools, the performance analysis metrics useful for analyzing the cloud security are listed and the comparison results are given.

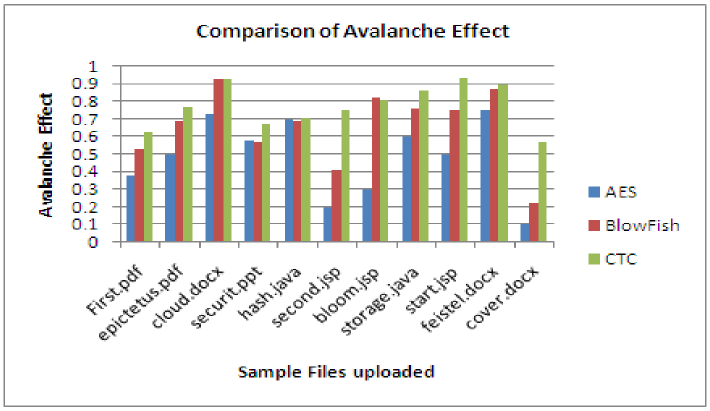
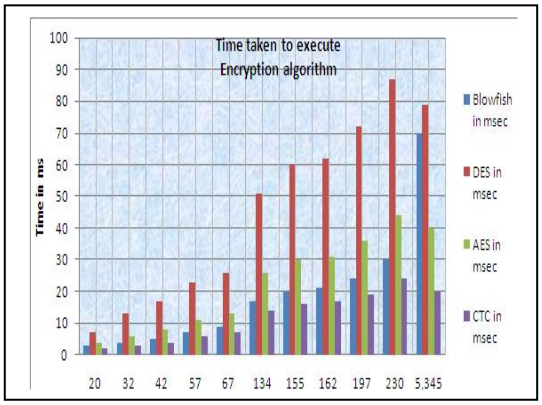


Figure 8. Comparison of Avalanche Effect of Cipher Key

The need for the discussion of avalanche effect is as shown in figure 8. Avalanche effect is that by changing only one bit in a matrix, leads to a large change in the existing key, hence it is hard to perform an analysis of cipher text, when trying to come up with an attack. Higher the avalanche effect, higher the strength of the cipher key. The avalanche effect is calculated by the formula,

$$\textit{Avalanche Effect} = \frac{\textit{Number of values changed in the cipher Key } C_{m \times n}}{\textit{Total Number of values in the cipher key } C_{m \times n}}$$

The Figure 9 and Figure 10 discusses the time taken for encryption and decryption of files using the proposed modified feistel network function.



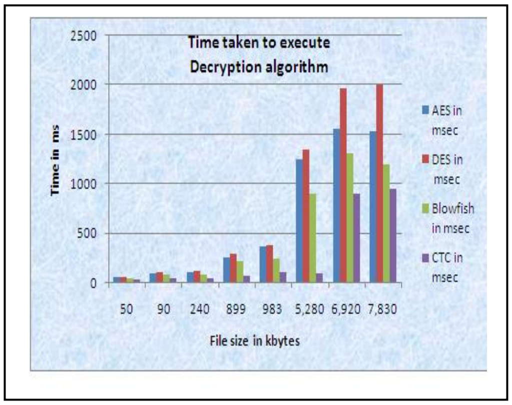


Figure 9. Comparison of Time Taken for Executing using MFN Algorithms

Figure 10. Comparison of Time Taken for Executing Decryption Algorithm using MFN

From Figure 9 and Figure 10, it is observed that the time taken for encryption using MFN is less when compared to the existing encryption algorithms. The number of rounds taken for executing the proposed algorithm is less when compared to existing encryption algorithms.

The Table 1 illustrates the comparison of proposed modified feistel network algorithm with respect to existing algorithm under various features.

Table 1. Comparison of Proposed Algorithm with Existing Algorithms with Respect to Various Features

Features Analyzed	Algorithms				
	DES	AES	Two Fish	Blow Fish	Modified Feistel Network
Created By	IBM in 1975	Joan Daemen & Vincent Rijmen in 1998	Bruce Schneier in 1993	Bruce Schneier in 1993	2013
Algorithm Structure	Feistel Network	Substitution-Permutation Network	Feistel Network	Feistel Network	MFN
Rounds	16	10, 12 or 14	16	16	4
Key Size	56 bits	128 bits, 192 bits, 256 bits	128 bits, 192 bits or 256 bits	32-448 bit in steps of 8 bits. 128 bits by default	256*6 bits
Type	Block cipher	Block cipher	Block cipher	Block cipher	Block cipher
Block Size	64 bits	128 bits	128 bits	64 bits	64 bits
Algorithm Running Time Kbytes/msec	5kb/msec	2kb/ msec	150kb/ msec	190kb/msec	250kb/msec
Key Strength	Low	Low	High	Very High	Very High
Existing Cracks	Brute force attack, differential cryptanalysis, linear cryptanalysis, Davies' attack	Side channel attacks	Truncated differential cryptanalysis	Second-order differential attack	NO
Avalanche Effect	Less	Less	Moderate	Moderate	High

9. Conclusion

This paper presents how metadata based security frameworks can be used in helping to provide security to the data stored at cloud storage servers. Further, it has proposed a security framework which decouples the third party auditor from the cloud security scenario and provides security by its own. The metadata attribute based security system is a promising technique, which can be applied in any remote storage systems. Many details about a novel method of cipher-key generation using modified feistel network is carried out and are presented along with key distribution policies. The proposed security model holds good for data confidentiality as the CSP and malicious users cannot recover data without the data owner's involvement. Hence, the goal of providing security to the data under rest and also during motion is achieved through metadata. In addition to that, the whole of the proposed security system is semantically secured against malicious users and also against the illegal activity of the trusted CSPs.

References

[1] M. Cammert, J. Kramer and B. Seeger, "Dynamic Metadata Management for Scalable Stream Processing Systems", Proc. of IEEE International Conference on Data Engineering Workshop, **(2007)**, pp. 644-653.

[2] J.-J. Wu, P. Liu and Y.-C. Chung, "Metadata Partitioning for Large-scale Distributed Storage Systems", Proc of IEEE International Conference on Cloud Computing, **(2010)**, pp. 212-219.

[3] A. Verma, S. Venkataraman, M. Caesar and R. Campbell, "Efficient Metadata Management for Cloud Computing Applications", Proc of International Conference on Communication Software and Networks, **(2010)**, pp. 514-519.

[4] Y. Hua, Y. Hong Jiang, D. Feng and L. Tian, "Supporting Scalable and Metadata Management in Ultra Large Scale File Systems", IEEE Transactions on Parellel and Distributed Systems, vol. 22, no. 4, **(2011)**, pp 580-593.

[5] R. Anitha and S. Mukherjee, "A Dynamic Metadata Model in Cloud Computing", Proc. of Springer CCIS, vol. 2, **(2011)**, pp. 13-21.

[6] S. O. Kuyoro, F. Ibikunle and O. Awodele, "Cloud Computing Security Issues and Challenges", International Journal of Computer Networks, vol. 3, no. 5, **(2011)**, pp. 247-255.

[7] A. Mathew, "Survey Paper on Security & Privacy Issues in Cloud Storage Systems", Proc. of Electrical Engineering Seminar and Special Problems 571B, **(2012)**, pp. 1-13.

[8] J. Heurix, M. Karlinger and T. Neubauer, "Perimeter – Pseudonymization and Personal Metadata Encryption for Privacy-Preserving Searchable Documents", Proc. of International Conference on Health Systems, vol. 1, no. 1, **(2012)**, pp. 46-57.

[9] Y. Tang, P. P.C. Lee, J. C.S. Lui and R. Perlman, "Secure Overlay Cloud Storage with Access Control and Assured Deletion", IEEE Transacations On Dependable and Secure Computing, vol. 9, no. 6, **(2012)**, pp. 903-916.

[10] Q. Wang, C. Wang, K. Ren, W. Lou and J. Li. "Enabling Public Auditability and Data Dynamics for Storage Security in Cloud Computing", IEEE Transactions On Parallel And Distributed Systems, vol. 22, no. 5, **(2011)**, pp. 1-13.

[11] S. Yu, C. Wang, K. Ren and W. Lou, "Achieving Secure, Scalable, and Fine-grained Data Access Control in Cloud Computing", Proc. of IEEE INFOCOM, **(2010)**, pp 1-9.

[12] Y. Wang and HaiTao Lv, "Efficient Metadata Management in Cloud Computing", Proc. of IEEE International Conference on Communication Software and Networks, **(2011)**, pp 514-519.

[13] K. Kuroiwa and R. Uda, "A Low Cost Privacy Protection Method for SNS by Using Bloom Filter", Proc. of the 6th International Conference on Ubiquitous Information Management and Communication, **(2012)**.

[14] J. Ravi Kumar and M. Revati, "Efficient Data Storage and Security in Cloud", Proc. of International Journal Of Emerging trends In Engineering And Development, vol. 6, no. 2, **(2012)**.

[15] C. Modi, D. Patel, B. Borisaniya, A. Patel and M. Rajarajan, "A survey on security issues and solutions at different layers of Cloud computing", Journal of Super Computers, **(2013)**, pp. 561–592.

[16] Y. Arai and C. Watanabe, "Query Log Perturbation Method for Privacy Preserving Query", Proc. of the 4th International Conference on Ubiquitous Information Management and Communication, **(2010)**, pp. 1-8.

[17] C. Watanabe and Y. Arai, "Privacy-Preserving Queries for a DAAS model using Two-Phase Encrypted Bloom Filter", Proc. of International Conference on Database Systems for Advanced Applications, **(2009)**, pp 491-495.

[18] B. Kumar Dewangan and S. Kumar Baghel, "Verification Of Metadata By Encryption For Data Storage Security in Cloud", International Journal of Research in Computer and Communication technology, vol. 1, no. 6, **(2012)**, pp. 300-305.

[19] S. Kaneko, T. Amagasa and C. Watanabe, "Semi-Shuffled BF: Performance Improvement of a Privacy-Preserving Query Method for a DaaS Model Using a Bloom filter", Proc. International Conference on Parallel and Distributed Processing Techniques and Applications, **(2011)**.

[20] M. K. Aguilera, M. Lillibridge and Maccormick, "Block-Level Security for Network-attached disks", Proc. of the 2nd Usenix conference on File and Storage Technologies, **(2003)**, pp. 159–174.

[21] C. Wang, Q. Wang, K. Ren and W. Lou, "Ensuring Data Storage Security in Cloud Computing", Proc. of 17th International Workshop on Quality of Service, **(2009)**, pp. 1-9.
[22] K. Nyberg and L. R. Knudsen, "Provable security against differential cryptanalysis, in: Advances in Cryptology", Proc. of CRYPTO'92, - LNCS, Springer-Verlag, vol. 740, **(1992)**, pp. 566-574.
[23] P. U. Wawge and A. R. Rathod, "Cloud Computing Security with Steganography and Cryptography AES Algorithm Technology", Proc. of World Research Journal of Computer Architecture, vol. 1, no. 1, **(2012)**, pp. 11-15.
[24] J. Kaur, M. Duhan, A. Kumar and R. Kumar Yadav, "Matrix Matching Method for Secret Communication using Image Steganography", International Journal of Engineering, Hunedoara, no. 3, **(2012)**, pp. 45-48.
[25] S. Rose Govada, B. SatishKumar, M. Devarakonda and M. James Stephen, "Text Steganography with Multi level Shielding", International Journal of Computer Science Issues, vol. 9, no. 4, **(2012)**, pp. 401-404.
[26] N. Chowdhury and P. Manna, "An Efficient Method of Steganography using Matrix Approach", International Journal of Intelligent Systems and Applications, vol. 4, no. 1, **(2012)**, pp. 32-38.

Authors

R. Anitha, has received her B.E degree in Electronics and Communication from Bharathidasan University in 2000 and ME degree in computer science from Anna University in 2008. She has worked in Tagore Engineering College, Chennai as a Senior Lecturer. Currently, she is doing research in the area of cloud computing under the guidance of Dr. Saswati Mukherjee. Anitha is a senior research fellow in University Grant Commission, New Delhi. Her current research interests include cryptography and information security in cloud computing and also in Big Data scenario.

Saswati Mukherjee, is currently a Professor in the Department of Information Science and Technology, CEG, Anna University, Chennai, India. Her fields of research interest are Distributed Systems, Grid and Cloud Computing and Information Retrieval and Natural Language Processing. While some of her students have completed, she is currently guiding many fellows towards their Ph.D. and M.S.(by research) in Anna University. Dr. Mukherjee had completed her UG from North Bengal University and PG from Calcutta University. She obtained her Ph.D. degree from Jadavpur University. After a short stint in ISI, Kolkata as a SRF, she joined the teaching fraternity. She has worked in B.E. College, West Bengal, VJTI, Mumbai before joining CEG, Anna University.