Chapter - 2

Review of Literature
CHAPTER – 2

REVIEW OF LITERATURE

2.1. Introduction

Storage and maintenance of data are the most important task in all kinds of enterprises. Data storage in an enterprise involves high costs because it requires physical IT resources like servers, storage, operating system and network etc., and frequent management procedures like backup, tuning, etc. and skilled administrative experts for monitoring the whole data storage administration process [Dim, 12]. This would increase the enterprises uncertainly with respect to storage and maintenance of the data [Ahm, 12]. The best solution for data storage and maintenance is data outsourcing in cloud environment [Pie, 10]. The basic operations of cloud are explained in the Chapter 1. The data are outsourced to CSPs, and then CSPs take care of the data within their own specialized structure to offer high availability and disaster protection. Cloud data outsourcing faces many issues especially in security aspects [Dey, 12]. Many research works are being carried out to provide a better security to protect the data in the cloud storage.

Cloud computing brings out a wide range of benefits including configurable computing resources, economic savings, and service flexibility [Vaq, 09]. However, security concerns are shown to be the primary obstacles to a wide adoption of clouds [Kei, 13]. The new concepts that clouds introduce, such as multi-tenancy, resource sharing and outsourcing, create new challenges to the cloud data security. To address these challenges, it is necessary to tune the security measures developed for traditional computing systems and proposing a new security policies, models and
protocols [Dan, 14] for cloud. Khalil et al., [Kha, 14] provides a comprehensive study of cloud computing security concerns and identify cloud vulnerabilities. Vulnerabilities are classified into security threats and attacks. It is important to control the vulnerabilities, neutralize the threats, and calibrate the attacks.

2.2. Security Issues and Solutions

Security risks are the biggest concerns when users want to apply outsourcing computing in cloud storage. There are various security problems involved in the cloud computing. They are data security, network security, data locality, data integrity, data segregation, data access, authentication and authorization, availability, backup, identity management and sign-on process, etc [Sub, 11]. Especially, among these security issues, confidentiality is the most important parameter to secure the data in cloud. Confidentiality for cloud storage ensures that the cloud providers do not learn any information about the users’ data [Cyr, 10].

The main concern around data storage is the protection of information from unauthorized access [Gon, 12]. In several usage scenarios, the risk of data being disclosed, lost, corrupted or stolen is unacceptable [Isa, 11]. Until data are stored on resources owned, controlled and maintained by the data owners, the possibility of unauthorized access is reduced by any physical countermeasure or trust in authentication and authorization mechanism [Jin, 13]. Things radically change when moving from resources fully controlled by the data owner to resources administrated by third party entity like public clouds. Resources that remain outside the users’ domain are not owned and controlled by the users. The risk that someone (an employee of the CSP) can access and disclose or corrupt data is considerable. This risk is usually known as
insider abuse or insider threat or insider attack [Min, 14]. Confidentiality of data in the cloud is to be ensured from insider attack. This is the major risk that, presently, is preventing the large adoption of cloud based solutions by the enterprises. Before companies move their data to the cloud, benefitting from the cloud storage advantages, all issues deriving from storing data on un-owned and un-trusted resources must be addressed by the legal security frameworks.

To protect data in cloud storage, currently a standard approach is to apply cryptographic techniques into users’ data [Pen, 12]. Cryptographic techniques have been widely used in the area of cloud storage, and it plays an important role in the data security. In the cloud environment, security attacks are protected by using cryptographic techniques. In cryptography, the message which is to be kept secret is called plaintext [Wil, 05]. The process of hiding its content is called encryption and the encrypted message is called ciphertext. The process of receiving the content of plaintext from the ciphertext is called decryption. A cryptographic algorithm or cipher is a mathematical function used in the encryption and decryption processes. A modern cryptographic algorithm always includes a key. Cryptographic algorithms, plaintexts, ciphertexts, and keys are collectively called cryptosystem. It works with combination of keys and algorithm to encrypt the plaintext and to decrypt the ciphertext.

The core concept of securing the data in cloud storage is using encryption techniques. The data are encrypted in the trusted environment before sending it to untrusted cloud storage providers. Theoretically both symmetric and asymmetric algorithms could be used, but, since the asymmetric are much slower than the symmetric, symmetric algorithms are preferred for performance reasons in cloud
environment. The usage of encryption is a technique to secure data guarantees and the confidentiality of data in the cloud storage.

2.3. Cryptographic Technique for Cloud storage

Process of encryption involves using a cryptographic algorithm and a cryptographic key to transform a plaintext into a ciphertext [Ble, 13]. In the field of cryptography, several techniques are available for encryption and decryption. These techniques can be generally classified into two major groups, i.e. conventional cryptography and public key cryptography [Wil, 05].

Conventional cryptography is referred to as symmetric encryption or single key encryption. Same key is used for encryption and decryption. This means that the encryption key is equal to the decryption key. Figure 2.1 represents the simplified model for conventional encryption technique. In general, there are two types of the symmetric ciphers, namely, stream ciphers and block ciphers.

Stream ciphers take the plaintext as streams of characters with size of 1 bit or n-bit word. In cipher, the plaintext is encrypted (and decrypted) one character at a time. According to Alfred et al. [Alf, 97], stream ciphers are used in real-time applications such as pay TV and communications. This is because they are able to run in high speed.

In Block ciphers, the plaintext is encrypted (and decrypted) one block at a time. The block size is commonly 64-bit or 128-bit.
Public key cryptography is referred to as asymmetric encryption or public key encryption in which there are two keys used, meaning that the encryption key is not equal to the decryption key. In this cipher, sender and receiver need to have two keys; a public key (which is made public) and a private key (which is kept secret). Figure 2.2 represents public key cryptosystem. The public key is used for the encryption process and the private key is used for the decryption process. Separate keys are used for encryption and decryption.
The original intelligible message, referred to as plaintext, is converted into apparently random ambiguous message, called ciphertext. The keys are independent on the specific of the plaintext. The algorithm will produce a different output depending on the specific key being used at that time. Changing the key changes the output of the algorithm.

Once the ciphertext is produced, it may be transmitted to cloud storage. Upon reception, the ciphertext can be transformed back to the original plaintext by using a decryption algorithm with the same key that was used in encryption.

Only symmetric encryption has the speed and computational efficiency to handle encryption of large volumes of data. It would be highly unusual to use an asymmetric algorithm for this encryption use case [Tim, 09]. Symmetric encryption algorithm is suitable to use for cloud data storage security, [Ema, 13].

2.3.1. Types of Attacks on Cryptographic Techniques

It is noted that the security of a cryptosystem must be entirely based on the keys. Attacks on the secrecy of encryption schemes try to recover plaintexts from the ciphertext, or even more drastically to recover the secret key [Alf, 97]. The possible attacks depend on the resources available in the adversary. Cryptographic attacks are designed to subvert the security of cryptographic algorithms and they are used to decrypt data without access to a key. The next subsection presents various kinds of attacks normally encountered in cryptographic algorithms [Okt, 13].

2.3.1.1. Ciphertext-Only Attack

In this type of attack, the cryptanalyst has the ciphertext of several messages and they have been encrypted using the same encryption algorithm. The job of
cryptanalyst is to recover the plaintext as possible or could deduce the key(s) which is used to encrypt and decrypt the message.

2.3.1.2. Known-Plaintext Attack

In this type of attack, the cryptanalyst knows the encryption algorithm and ciphertext to be deduced. Cryptanalyst’s role is to deduce the key(s) used to encrypt the message or an algorithm to decrypt the new message encrypted with the same key(s).

2.3.1.3. Chosen-Plaintext Attack

In this type, the cryptanalyst has access not only to ciphertext and associated plaintext for several data but also chooses the specific plaintext blocks to encrypt which yield more information about the key. Cryptanalyst job is to deduce the key(s) used to encrypt the messages or an algorithm to decrypt any new message encrypted with the same key(s).

2.3.1.4. Chosen-Ciphertext Attack

In this attack, the cryptanalyst knows different ciphertexts to be decrypted and has access to the decrypted plaintext. Cryptanalyst’s job is to deduce the key.

2.3.1.5. Brute Force Attack

A brute force attack is a trial-and-error method used to obtain information such as a user password or Personal Identification Number (PIN). In a brute force attack, automated software is used to generate a large number of consecutive guesses as to the value of the desired data. Brute force attacks may be used by criminals to crack encrypted data, or by security analysts to test an organization's network security.
2.3.1.6. Dictionary Attack

In this attack, every word in the dictionary is tried as a possible password for an encrypted message. A dictionary attack is generally more efficient than brute force attack.

Though different kinds of attacks are available to measure the security level of cryptographic algorithms, dictionary and brute force attacks are considered in this research to measure the security level with a security analysis tool.

2.4. Existing Cloud Security Frameworks

Security enables cloud as a safe and secured environment to the users; it restricts many of the attacks on the data from hackers. Kamara et al., [Kam, 10] proposed a cloud data storage framework for public cloud. The framework consisted of four components, namely, a Data Processor (DP), that processes data before they are sent to the cloud; a Data Verifier (DV), that checks whether the data in the cloud have been tampered with; a Token Generator (TG), that generates tokens which would enable the cloud storage provider to retrieve segments of customer data; and a Credential Generator (CG) that implements an access control policy by issuing credentials to the various users.

This framework is designed in the scenario for both general and enterprise users. In the case of general users’ scenario, they install the components of the framework that consists of a DP, a DV and a TG into their local machine. In the case of enterprise users’ scenario, Medium-sized enterprises deploy dedicated machines within their network including a DP, a DV, a TG and a DG. If enterprises are very large, the prospect of running and maintaining dedicated machines to process all
employees’ data are infeasible. More precisely, in this case the dedicated machines only run data verifiers, token generators and credential generators while the data processing is distributed to each employee. In this framework, users have to maintain the components like DP, DV, TG and CG. Cloud is used only for storing the data. Users have the maximum responsibilities to execute this framework.

Yau et al., [Yau, 10] presented an approach to secure the users' data from service providers. The approach contained three main parts, 1) separating software service providers, and infrastructure service providers, 2) hiding data owners’ information in cloud and 3) data obfuscation. The approach consisted of seven entities namely, Software Cloud, Infrastructure Cloud, Software Service Broker, Infrastructure Service Broker, Software Service Attestation Authority, Data Obfuscator and Data De-obfuscator. The Software Cloud and Infrastructure Cloud have the same features of the software layer in ordinary cloud computing architecture. However, the software layer and infrastructure layer are not managed by the same service provider. The Software Service Brokers and Infrastructure Service Brokers have the same functionality of the service brokers in Service Oriented Architecture (SOA), but they have the additional function for identity Anonymization. The Software Service Attestation Authority, Data Obfuscator and Data De-obfuscator are additional entities in this approach.

Atiq ur Rehman et al., [Ati, 11] proposed a framework to preserve confidentiality of data stored in Cloud Database as a Service (DaaS) model. The proposed framework stores sensitive data with a combination of encryption and obfuscation techniques. The framework consists of four modules namely, Encryption,
Obfuscation, Metadata and Query optimizer. Encryption and obfuscation are used to encrypt and obfuscate the data respectively. Encryption and obfuscation are done before sending the data to the cloud DaaS. Metadata is maintained by cloud users for storing details of keys and for encryption and obfuscation techniques. Query optimizer is used to enable users’ query to run on the encrypted and obfuscated data in the cloud storage. The four modules in the framework are executed from the users’ side. Cloud users have more responsibility to generate the key and also to keep the key secured.

Basescu et al., [Bas, 11] proposed a generic security management framework allowing providers of cloud data management systems to define and enforce complex security policies. They have designed the framework to detect and stop a large number of attacks defined through an expressive policy description language and to be easily interfaced with various data management systems. They have showed that they could efficiently protect a data storage system by evaluating their security framework on top of the BlobSeer data management platform [Nic, 09]. The benefits of preventing a DoS attack targeted towards BlobSeer were evaluated through experiments performed on the Grid5000 test bed [Jeg, 06].

Govinda et al., [Gov, 12] proposed an agent-based security framework for ensuring security. It helps users to control their sensitive information, and also ensures that the users have fewer burdens at their side. It assists the users by communicating their security related preferences to the service providers and assists the service providers in compliance with security law and regulations. An essential feature of agent-based security is obfuscation, used by users to protect the security of the data. Agent
should control two entities namely; Obfuscator that obfuscates the data sent by user to the cloud, Data Retriever that retrieves the data sent by cloud to users. Agent could automatically obfuscate some or all the fields in a data structure before they are sent off to the cloud for processing, and translates the output from the cloud back into de-obfuscated form. The obfuscation and data retrieval are done using a key which is chosen by the agent and not revealed to CSP. Simple obfuscation technique can easily be broken. There is a need for proper SLA among users, agent and cloud providers.

Munir et al., [Mun, 13] proposed a cloud security framework that identifies security challenges in cloud computing. The framework contains the following components. 1) *Client*: Users could access the client side with Multi-Factors Authentication (MFA) provided by End-User Service Portal (EUSP). 2) *End-User Service Portal*: When clearance is granted, a Single Sign-on Access Token (SSAT) could be issued using certification of user. Then the access control component shares the user information related with security policy and verification with other components in EUSP and CSPs. 3) *Single Sign-on (SSO)*: It enables user to access multiple applications and services in the cloud computing environment through a single login. 4) *Service Configuration*: The service enabler makes provision for personalized cloud service using user’s profile. 5) *Service Gateway and Service Broker*: A service gateway manages network resources and VPN on the information lifecycle of service broker. 6) *Security Control*: It provides significant protection for access control, security policy and key management against security threats. 7) *Security Management*: It provides the security specification and enforcement functionality. 8) *Trust Management*: It is a challenging need of integrating requirements driven trust negotiation techniques with fine-grained access control
mechanisms. 9) *Service Monitoring*: An automated service monitoring system guarantees a high level of service performance and availability.

Hamdan Al-Sabri et al., [Ham, 13] proposed Cloud Storage Encryption (CSE) architecture by using encryption techniques to provide a high level of data protection to cloud storage. The CSE architecture allows to encrypt and to index data in a manner that ensures the protection of data. The proposed architecture is composed of seven components. 1) *Director generated Keys and privileges*: A center within the organization to generate public and private keys for data users, as well as granting special privileges to the suitable roles inside the organization. 2) *Data users*: Clients or employees within the organization. 3) *User Roles*: It determines the characteristics and privileges for users. 4) *Encryption Point*: It is used to encode and index the data and divide data into several packages. Each package is stored in different cloud servers. A specific code is included in the divided packets, so that it can be assembled during the retrieval. 5) *Searchable Encryption*: It is a technique to search for the encrypted data during the retrieval of data from cloud storage without decryption. 6) *Decryption Point*: It is used to decode the encrypted data retrieved from cloud storage. 7) *Cloud Data Storage*: Databases for data storage. Users should maintain this architecture with all components. It increases the user encumbrance.

### 2.5. Data Protection Techniques

To protect the data in the cloud storage, the following data protection techniques are reported in the literature:

1. Data encryption [Sas, 12] [Moh, 13]
2. Data obfuscation [Gov, 12] [Sia, 09] [Yau, 10] [Mah, 12]
Chapter – 2

Review of Literature

The goal of these techniques is to store data on the cloud servers in an inaccessible format using encryption and obfuscation. Encryption is the procedure of transforming the data from readable form into unreadable form using a cryptographic algorithm and a key. Obfuscation is a method that masks the users’ data from illegitimate users by implementing a specific mathematical function or using programming methods. The major difference between encryption and obfuscation is that encrypted data cannot be processed until they are decrypted, whereas obfuscated data can be processed without de-obfuscation.

2.6. Related Works on Data Encryption

Data encryption is a traditional technique for ensuring the confidentiality of data in transit or data at rest [Ham, 10]. Cloud providers should use the data encryption based on their own internal structure. It is not sure that all the providers should absolutely encrypt customer’s data. For example, EMC’s Mozy Enterprise does encrypt a customer’s data. However, Amazon Web Service (AWS) S3 does not encrypt a customer’s data. Customers are able to encrypt their own data prior to uploading, but S3 does not provide encryption [Tim, 09]. This leads to the data confidentiality issue in cloud storage. Users do not know whether the data are encrypted in the cloud storage or not. Users should encrypt their data before they are uploaded to the cloud storage [Ati, 11]. Many researchers have proposed different encryption algorithms for security of data in cloud. From the literature review, the symmetric encryption algorithm is suitable for cloud storage, because asymmetric encryption techniques are about 1000 times slower than symmetric encryption which makes it impractical when trying to encrypt large amounts of data [Gur, 13]. The next
sub section discusses several symmetric encryption algorithms proposed by different researchers.

2.6.1. Symmetric Encryption Used for Cloud

Some researchers have suggested the traditional encryption technique for data security in cloud such as DES, 3DES and Blowfish. This section describes the working procedure of these algorithms. Some researchers have tried to propose new encryption algorithms.

Data Encryption Standard (DES) is based on a feistel block cipher [Wil, 05]. It is developed by the IBM cryptography researcher Horst Feistel [Mic, 07]. It consists of a number of rounds where each round contains bit-shuffling, non-linear substitutions and Exclusive OR operations. DES is a 64-bit block cipher as it uses the same key for both encryption and decryption and only operates on 64-bit blocks of data at a time. The key size used is 56-bits; however a 64-bit (or eight-byte) key is actually input. The least significant bit of each byte is used for parity and does not increase the security in any way. Once a plaintext is received for encryption, it is arranged into 64-bit blocks required as input. If the number of bits in the message is not evenly divisible by 64, then the last block is padded.

DES performs an initial permutation on the entire 64-bit block of data. It is then split into two, 32-bit sub-blocks, namely, $L_i$ and $R_i$ which are then passed into first round, of which there are 16 rounds. At the end of the 16th round, the 32-bit $L_i$ and $R_i$ output quantities are swapped to create the pre-output. The concatenation of $R_{16}$ and $L_{16}$ is permuted using a function which is the exact inverse of the initial permutation. The output of this final permutation is the 64-bit ciphertext [Gar, 14].
There are many attacks and methods that exploit the weaknesses of DES, which shows that DES is an insecure block cipher [Jaw, 11].

Triple DES (3DES) [Kau, 12] is simply a concatenation of three DES algorithm operations. The procedure for encryption is exactly the same as regular DES, except that it is passed through the DES engine three times. The first pass is a DES encryption, the second pass is a DES decryption of the first DES ciphertext result and the third pass is a DES encryption of the second pass result. This produces the resultant 3DES ciphertext. The procedure for decrypting a 3DES ciphertext is the same as 3DES encryption except in reverse order. Note that although the input key for DES is 64-bits long, the actual key used by DES is only 56-bits in length. This means that the effective key strength for TDES is actually 168-bits. Let $E_K(I)$ and $D_K(I)$ represent the DES encryption and decryption of $I$ using DES key $K$ respectively. Each 3DEA encryption and decryption operation is a compound operation of DES encryption and decryption operations [Mon, 12]. The following operations are executed,

1. 3DEA encryption operation: The transformation of a 64-bit block $I$ into a 64-bit block $O$ that is defined as,

   $$O = E_K_3(D_K_2(E_K_1(I)))$$

2. 3DEA decryption operation: The transformation of a 64-bit block $I$ into a 64-bit block $O$ that is defined as,

   $$O = D_K_1(E_K_2(D_K_3(I)))$$

There are several keying methods that 3DES uses. All three keys can be independent of each other, or the first and third keys can be identical, with the second
key being unique [Ras, 13]. All three keys can also be identical, which provides least security and it takes time to encrypt than DES.

Blowfish is a symmetric block cipher that can be effectively used for encryption and safeguarding of data [Shi, 14]. It takes a variable-length key, from 32-bits to 448-bits, making it ideal for securing data. Blowfish was designed in 1993 by Bruce Schneier as a fast, free alternative to existing encryption algorithms. Blowfish is unpatented and license-free, and is available free for all users. Blowfish Algorithm is a Feistel Network [Hoa, 10], iterating a simple encryption function 16 times. A Feistel network is a general method of transforming any function (usually called F function) into a permutation [And, 13] [Sar, 14]. Blowfish uses a large number of sub keys. These keys must be pre computed before any data encryption or decryption. The input is a 64-bit data element and produces the 64-bit cipher text.

Researchers have tried to propose new encryption algorithms for data protection in the cloud storage. Avinash Sharma et al., [Avi, 12] have used the ASCII code to encrypt the plaintext. The proposed idea has initially produced ASCII codes for the plaintext. The ASCII codes are reversed and they are called reverse ASCII codes (For example: ASCII Code is 82, Reversed ASCII code is 28). Generate two keys namely $K_1$ and $K_2$. $K_1$ is generated from the sum of reverse ASCII Codes and $K_2$ is generated by the sum of ASCII codes produced from plaintext. The key $K_1$ is applied on the reverse ASCII codes in odd position and key $K_2$ is applied on reverse ASCII codes in position to produce ciphertext. This idea is only applicable to character based plaintext. Ciphertext size is greater than plaintext, because plaintext “RESPECTEVERYONE” is encrypted with ciphertext as “1084 1251 1094 1163 1152..."
Subhasri et al., [Sub, 13] proposed a method for Multi-level Encryption algorithm which is used to secure the data in the cloud. This method has two levels of encryption process; first, the characters of plaintext are rearranged using the rail fence cipher; second, the result of rail fence cipher is applied by improved ceaser cipher algorithm. The ceaser cipher is improved and steps involved in the improved cipher are, splits letters of the plaintext into individual character. Assign the position value $i$ to each character in the plaintext. Convert each character into ASCII code. Assign a key $k$ value for encryption. Apply the values in the following formula:

$$E = (p + k + i) \% 256$$

$p$ – Plaintext, $k$ – Key, $i$ – Character Position.

A decimal value among 0 to 256 is produced for each character in the plaintext. Position value for each character is maintained for decryption process. It takes more storage from users’ side to store the position of each character in plaintext. For example, if plaintext has 10,000 characters, then 10,000 character positions are maintained alone with the key $k$.

Sunitha Rani et al., [Sun, 12] proposed a hybrid encryption algorithm in order to provide security to the data in the cloud. The proposed methodology used three encryption algorithms sequentially to encrypt a message. First, plaintext is encrypted by the ceaser cipher. Second, the encrypted result from ceaser cipher is again
encrypted using RSA substitution algorithm. Third, the result from RSA is again encrypted by the monoalphabetic substitution method. Three algorithms are executed one after another. This technique has taken more time to encrypt the text by three algorithms sequentially. Anshu et al., [Ans, 13] proposed encryption algorithm to make cloud data secure. The authors discuss the security issues and challenges of cloud and compare the existing algorithms like AES, DES, Blowfish and RSA Algorithms. Comparison shows that DES algorithm consumes less encryption time. RSA takes larger memory usage and encryption time. Blowfish has taken the least memory requirement. AES algorithm takes less time to execute cloud data.

Most of the researchers have proposed an encryption algorithm for cloud by integrating any two or three existing algorithms. But this is not a complete solution for cloud environment. Cloud needs a complete security service algorithm to protect users’ data in the cloud storage. The proposed AROcrypt SSA in SEaaS is an encryption algorithm (discussed in Chapter 4) and it is compared with DES, 3DES and Blowfish with respect to security and time.

2.7. Related work on Data Obfuscation

Data Obfuscation (DO) [Ran, 10][Cha, 12] is a form of data masking where data is purposely scrambled to prevent unauthorized access to sensitive data. This form of DO results are in meaningless or confusing. The techniques listed below have two main goals; protect sensitive data from disclosure and create usable test data with the same data shape as the original. It is ideal to use more than one technique to bring added protection.
2.7.1. Masking

Data masking replaces sensitive characters or fields with a meaningless character such as “X”. Masking preserves the data shape for display on screens and reports.

2.7.2. Substitution

Substitution replaces the fields of data with similar content that is unrelated to the original data. An example of substitution is to replace the actual first and last names with names randomly picked up from a large list of valid first and last names that have been created specifically for use in substitution. Substitution preserves the original data shape while hiding the actual sensitive information.

2.7.3. Shuffling Records

Shuffling and substitution are similar except that shuffling uses the source data itself instead of an external list. Shuffling moves data between rows so that the data shape is preserved but the original details of the sensitive information are hidden.

2.7.4. Number and Date Variance

Variance modifies number or date by replacing the field with similar information that is a random percentage of the original. The percent variance is chosen to keep the new data within valid ranges for the field. Variance keeps the data shape while hiding the original sensitive information.

2.7.5. Gibberish Generation

Gibberish generation is used when to hide the sensitive data with associated data such as correspondence that can identify the original data. A practical example
would be bank records. Users could obfuscate the account information of customers in the database tables but the records are linked to images of the monthly statements sent to the customers. The stored statements contain the entire information that customers wish to hide. To prevent the sensitive information from being revealed, gibberish generation replaces the confidential data with random junk of data files with equivalent size.

2.7.6. Data Generation

Data generation creates fictitious or mock data from scratch or other sources which are usable for testing purposes.

There are many more obfuscation techniques available in the literature. The approach for each of the obfuscation techniques can range from very simple to highly mathematical. Recently, researchers have started to use obfuscation techniques for security protection in cloud environment.

Siani Pearson et al., [Sia, 09] described a privacy manager for cloud computing, which reduces the risk to the cloud computing user of their private data being stolen or misused. As a first line of defense, the privacy manager uses a feature called obfuscation. The idea is that instead of being present unencrypted data in the cloud, the users’ sensitive data are sent to the cloud in an obfuscated form. The obfuscation method uses a key which is chosen by the users and known by the privacy manager, but which is not communicated to the CSPs. Thus the CSP is not able to de-obfuscate the users’ data, and the data are not present on the CSPs’ machines. It reduces the risks of theft of the data from the cloud and unauthorized uses of the data [Mir, 12]. Moreover, the obfuscated data are not personally identifiable information,
and so the CSPs are not subject to the legal restrictions that apply to the processing of the de-obfuscated data. However, it is not practical for all cloud applications to work with obfuscated data.

Maheshwari et al., [Mah, 12] introduced a scheme which allows a user to store data in a cloud and perform database style query on the stored data without using standard cryptography schemes while maintaining data confidentiality. Data obfuscation is used for security which makes it hard for the cloud to reconstruct the plaintext. This scheme represents each character by its glyph image. Instead of storing the normal glyph in a given font, which can be readily understood by a machine, add noise and split the glyph image into small portions. Different portions belonging to a glyph image are stored in servers belonging to different cloud providers. To reconstruct the data, the glyph portions from different servers need to be gathered. The overall concept of this scheme is to convert the users’ data into an image and each part of the image is sent to different independent clouds. The system mainly comprises cloud service providers and users. The users’ data consists of any printable ASCII character. A library of images corresponding to each character is created at the users’ side. The library needs to be constructed only once. The BMP image file format is manipulated in the system. Each image is constructed in grayscale, 1-bit depth and two colors, black and white. The resolution is $40 \times 40$, where the size of each image is 382 bytes. As data are read from the users’ document, their equivalent images are obfuscated by adding noise to each image. After adding noise to the bitmap images, each image is split into as many equal parts as the number of cloud providers. The providers are independent and are not aware of each other’s presence. The data held is disjoint, that is, no two providers have any part of the data common between them. Hence, even if
one provider is compromised, the user is still safe since only a small part of the data will be revealed. Each divided part of the original image is essentially a secret image. This scheme needs to maintain log details of data splitting which are used for reconstructing the data into original form.

The proposed MONcrypt SSA in SEaaS is an obfuscation technique (Discussed in Chapter 5) and it is compared with existing obfuscation techniques such as Base64, Base32 and Hexadecimal Encoding with respect to security and time.

2.8. Chapter Summary

A review has been conducted to find the issues and solutions provided related to security of data in the cloud storage. Many researchers and practitioners work on identifying cloud threats, vulnerabilities, attacks, and other security issues, in addition to providing countermeasures in the form of frameworks. Many research works are carried out to solve the security issues especially aimed at concentrating on confidentiality of data in the cloud. From the review, it is clear that many novel research works are needed to strengthen the public cloud storage environment. If the security services are stronger and well established, then it will prevent many security issues either from CSP or other users of the cloud.

The security service should be fast and at the same time security should not be compromised. In order to effectively secure the public cloud storage, data sent to the cloud are masked using prominent security services. This chapter has presented the literature review and listed the research works related to the proposed research work. The next chapter explains the security framework, called AROMO which has security service to ensure the confidentiality of the data in the cloud storage.