Migration Letters

Volume: 21, No: S8 (2024), pp. 713-725

ISSN: 1741-8984 (Print) ISSN: 1741-8992 (Online)

www.migrationletters.com

Artificial Intelligence Driven Dynamic Shuffling (DS) For Cloud Storage Employing Lightweight Techniques

Dr. S. Srisakthi [*], Dr. G. Manju [*], Dr. Tapabrata Roy [*]

Abstract:

From the dawn of cloud computing, the usage of cloud has been growing by leaps and bounds. One of the main usages of cloud is the storage service that it offers. In spite of its many creditable advantages, it also has some disadvantages like data security and data availability. These two are the main issues that a user faces from the time of its inception. Many models have been proposed to solve these issues. These models use cryptographic methods to secure the data. These methods solved the data security issue at the cost of extra storage space and increased time consumption both at the user and at the server side. These models have a limitation when it comes to resource-constrained devices. This paper ensures data security using lightweight techniques which brings in randomization by using the shuffling process. This paper proposes two models that perform shuffling in a dynamic manner. The Artificial Intelligence (AI) component decides the shuffling model that needs to be chosen. The Secure Dynamic Shuffling (SDS) model, executed by the user alone. Secret Shared Dynamic Shuffling (SSDS) model uses a secret sharing scheme, where both the user and the cloud provider share the secret. The proposed model gives a reduction of the storage space by 60%.

KEYWORDS *Artificial Intelligence, Cloud computing, data storage, data security, lightweight techniques.*[1]

1. INTRODUCTION

The cloud computing applications has been increased over the years, and its applications start from high-end applications to day-to-day activities. Software, infrastructure, and platform are some of the services that are provided. Some of the main benefits of cloud can be categorized as: on demand self-service, elasticity, and providing payable services (Mell & Grance 2011). Cloud User (CU), the Cloud Service Provider (CSP) and the Service Instance (SI) are the three main entities involved in a cloud system. The deployment of Cloud has been made possible in many areas including business, government & private organizations, academic institutions, etc. and the technology has attracted non-technical organizations too.

Of the many applications of cloud, cloud data storage is the most prominent. Cloud data storage serves as a service model where data of the user is stored, maintained, managed remotely by the CSP and can be returned back to the user when requested over the internet. Of the many benefits of the cloud data storage, the vital benefits include flexibility, cost savings and accessibility. There is Global management of the data irrespective of any physical boundary has been the vital benefit. The cloud computing can be classified based on their locations as given below:

(a) Public cloud – provides multi-tenant storage and this storage is more suited for unstructured data. Here the cloud vendor supplies the computing infrastructure. The customer will not be able to see or have control over the computing infrastructure.

[*] School of Computer Science and Engineering, VIT Chennai, Tamil Nadu, India.

(b) Private cloud – provides a dedicated storage to the clients. Private cloud is both expensive and secure when compared to public clouds.

(c) Hybrid cloud – this being a combination of both with at least one public cloud and one private cloud.

(d) Community cloud – the computing infrastructure is shared between the cloud organizations of the same community. There may be many government organizations within the boundaries of a state. These organizations may share computing infrastructure on the related to citizens residing in that state.

Many CSPs provide storage service like Amazon S3, Google Cloud, and Microsoft SkyDrive etc. Some provide a free storage up to a certain amount of data and later on charge the CU. Most of the CSPs follow a 'pay-as-you-use' model, whereby the CU needs to pay for storage used. Despite a list of benefits, one of the frequently triggering problems of cloud data storage is maintaining the integrity of the stored data. The client outsources his data to the cloud and do not possess a local copy of it, and therefore the data needs to be intact. However, data integrity faces many challenges internally and externally destroying the intactness of the data as stated by Kui Ren et al. (2012) To design a model, which maintains the data integrity has been a challenging research work. The other challenging design here is to provide a fault tolerant CSP. The client should be able to get his data whenever he needs it, irrespective of any problem that might occur at the provider side. These two are the primary issues that are under research. There has been a lot of work going on to provide a fault tolerant CSP and to ensure high data integrity. Thus, the main concern of the CU is to make sure that the CSP that he uses to store data assures him the three major virtues of Confidentiality, Integrity and Availability (CIA). Confidentiality ensures that the data of the CU can never get into the hands of a third person including the CSP. Integrity refers to the condition where the CUs data has to be maintained intact. Availability requires that data should be available when an authenticated request is made. A good CSP must provide these qualities to the CU. Models were developed to provide these qualities to the user. And most of the models make use of various cryptographic techniques. Our paper proposes two such models that ensures confidentiality, data integrity and data availability. The contributions are summarized as follows:

Design of two model of SDS (Secure Dynamic Shuffling) and SSDS (Secret Shared Dynamic Shuffling) through which a user can store his data in a secure manner. The model has been developed using a CSP and CU. The CU does the process of conversion, splitting, shuffling on the user data. These components work to secure the CU's data. CU plays the role of the front end. The experiments carried out show that our model is able to resist many attacks. The main contribution is the use of lightweight techniques to secure the data. The use of cryptographic techniques increases the overhead both at the CU and at the CSP side and it also consumes time. The problems present due to the use of these techniques has been overcome in the model proposed in this paper.

The paper is organized as follows: Section 2 discusses related works. Section 3 presents the proposed work along with the analysis. Section 4 presents the conclusion.

2. RELATED WORK

CIA has been one of the main properties for preserving the security of a system. Many models have been developed to meet this property. The CU or the CSP, or an external Third Party Auditor (TPA) can maintain the CIA measure. Many researchers are working to find a feasible and optimal solution. The solution that needs to be developed depends on the application for which the model has been developed to be deployed. One of the methods to guarantee this CIA property has been to use a Service Level Agreement (SLA) (Badger 2011). This agreement is a legal document that specifies a set of rules, and it acts as an exact legal contract between the user and the provider. Both the user and the provider sign the legal agreement. This contract forbids the CSP from disclosing any of the CU's information to any other entity, and this includes the fact that the CSP itself should not read the data. Here, the user would receive compensation in the event of the CSP misbehaviour,

or the CSP can be charged with penalty as the user pays for a desired level of security. The main limitation present here is that the CU signs for the contract with the CSP where its data are stored. But the same CSP may have sub contract arrangements with other CSPs. This situation might lead to a point where indirectly the CU might be asked to sign a contract with the other CSPs too. This might not be welcomed by the CU (Mark 2013). The CSP to avoid incurring a bad name which might spoil its reputation, may not disclose the fact that it has been attacked or it has failed to meet the SLA.

During some situations, the TPA has also been used to verify the integrity of the data, through the auditing approach. Ateniese (2007) have proposed a model where a client verifies the integrity of the data. A PDP (Provable Data Possession) protocol has been proposed that permits a public verifier, to audit the data that is present over an un-trusted server. The model uses RSA (Rivest-Shamir-Adleman) based homomorphic tags for auditing the stored data. Though these models verify the integrity, these models possess a limitation that it needs to download the data and then do the auditing, due to which the data becomes vulnerable to attack. This drawback has been overcome by Zhu et al. (2013) who have proposed a dynamic audit service model. This model does the audit by doing a random sampling of the data at prefixed audit schedules and at the place of their storage itself.

When TPA is used solely for data verification, the data becomes vulnerable to insider attacks. In some models the TPA cannot be relied upon completely as it may turn malicious and its overhead also gets increased. At these times the CSP works hand in hand with the TPA to maintain the integrity of the data. Syed Rizvi et al. (2014) have proposed a model where the integrity verification is done by combining the CSP and the TPA. The user encrypts the data using the secret key, and this secret key is sent to the TPA using PKI. When the CSP needs to do any computations, it requests for the secret key. The TPA communicates the secret key to the CSP after verifying the authenticity of the CSP.

The CU maintains the data integrity by making use of encryption and decryption algorithms. XiaoChun YIN et al. (2014) have proposed a model that uses Elliptic Curve Cryptography (ECC). This model was based on Public Key Infrastructure (PKI), a certificate procedure, this procedure ensured that a user shares data with authenticated multiple users in a secure manner. The user needs to authenticates himself and need to obtain a certificate. The certificate needs to be obtained from the Certificate Authority (CA), once the authentication is completed the storage and retrieval of the data can be done. But the management of these certificates creates an overhead. To overcome the use of certificates, a method of access control policy has been proposed (Sushmita 2014) It points out that a person who wants to read or modify the data stored in the cloud needs to authenticate himself. Based on the access permissions that have already been granted to the user, the access is restricted. Srisakthi and Shanthi (2015) have proposed a model where the user's data are encrypted using the ECC algorithm. This encrypted file is split into a fixed number of parts. These parts are taken as the input and two models have been developed. In the first model, a backup is created for each part of the file and is stripped over multi cloud servers along with the original part of the file. Here, each cloud server has two different parts of the file. In the second model, a second back up is created and is also stripped over the cloud. Here, each cloud server has three different parts of the file. The aim of creating backup copies is to reconstruct the data from the remaining cloud servers when one of them goes for a failure. The storage details like the number of parts the file is divided into and the order by which the parts are stored over the multi clouds are written in a file. This file is again encrypted using the secret sharing scheme and the key is held by a set of users.

Access Control Lists (ACLs) has been one of the methods through which the user access can be granted. Access to data in the cloud was provided through this (ACLs), but this model has a problem of scalability (Jin Li, 2010). The role of cryptography in maintaining security of the data has been enormous. Attribute-Based Encryption (ABE) has been one of the techniques that has been used for controlling the data access. Goyal et al. (2006) have proposed the ABE model. This model provides a more scalable and fine-

grained access control to the data when compared to the ACLs method. Proxy Re-encryption is another technique that has been used for ensuring a secure and confidential data storage in the cloud (Wang 2012). This model translates the cipher text under the data owner's public key into another cipher text. This cipher text that could be decrypted by another user's secret key and this is done by the semi-trusted proxy that has the re-encryption key. The assumption here is that at no time the proxy would be able to access the plaintext.

However, all these models cause an extra burden to the CU, and involve complexity in terms of time and space. The overhead caused due to these encryption algorithms have made many researchers to work on lightweight cryptography techniques. The lightweight approach uses techniques that reduce the computations. Al-Souly et al. (Al-Souly 2013) have proposed lightweight symmetric encryption algorithm that can be used for ensuring a secure database. The model encrypts only the sensitive data and a comparison is made with the Advanced Encryption Standard (AES) and Data Encryption Standard (DES) algorithms. Comparison is done based on the encryption time and size of the encrypted data. To perform the encryption and decryption process keys are used and the keys incur certain management procedures. The key management involves techniques involving generation, distribution, storage, revoking, and verifying the keys. Buchade A. R et al. (2014) discusses the management of keys involved during the encryption and decryption process. Separate keys are involved at the CU side and at the CSP side or a key is shared among them or a key split among them. The paper compares the different key management methods. Thomas Eisenbarth et al. (2007) presented a survey on the state-of-the-art cryptographic algorithms and compared them with lightweight cryptographic algorithms. The comparison is made over the hardware and software implementations of the symmetric and asymmetric ciphers.

All the models made use of cryptographic algorithms and provided a reliable security measure. But the limitations of these models have been that they are not suitable for resource constrained devices like mobile phones. These cryptographic algorithms incur overhead in terms of time and size. In certain scenarios, the security requirements are not high and the importance needs to be given on to the execution time, storage space. At these scenarios the proposed model can be used.

3. PROPOSED MODEL

The main goal of this model is to ensure data security using lightweight techniques. Literature presented many models that were developed, to achieve data security. However, most of these models used the concept of cryptography. In the proposed model the participating entities are denoted by CSP – (Cloud Service Provider) and the CU – (Cloud User). The work of maintaining data integrity is carried out by the CU. The proposed model consists of two phases: system initialization and secure data manipulations. The operations performed by the DS in these phases are as follows:

3.1 System Initialization

This involves the procedure by which the system gets initialized to the CU actions. It follows a two-step process, where a user first registers and then authenticates himself to log on to it. The two steps are as follows:

Registration Process (RP): The CU registers himself to the CSP by giving the user name and password. The CSP after getting this information stores it and uses it later. The process is depicted in figure 1. A Unique IDentification (UID) is computed for each user and is withheld with the CSP

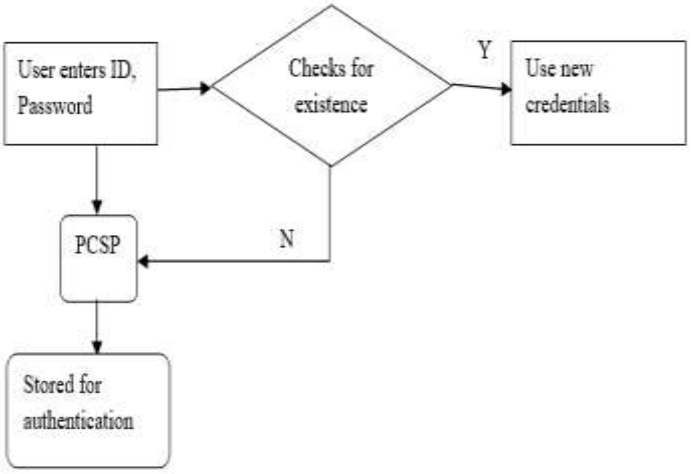

Figure 1. RP done by the CU

Authentication Phase (AP): Once the registration is complete, the user needs to be authenticated each time the user logs in. The user enters the login information to the CSP. Once the authenticity of the user is verified, the user will be able to login in.

3.2 Secure data manipulations

This phase performs manipulations over the data, before being transmitted to the cloud storage. The process that are involved in this phase are conversion, splitting, shuffling and storing. The working of these processes and their benefits are discussed in detail

3.2.1 File conversion and splitting

The user file is converted into a sequence of byte streams. This file goes through the splitting process, where the file is split in to an arbitrary number of 'n' packets of equal size. The number of packets that the file is being split into can be user defined or a number can be set as default. The split packets are shuffled in an order that is known only to the CU. DS uses the shuffling method to shuffle the user data so as to increase the properties of confusion and diffusion. After shuffling the packets, CU uploads it in the CSP. The workflow of DS is shown in figure 2.

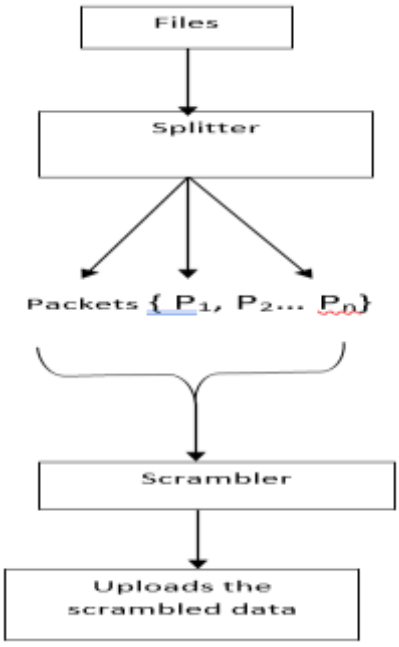

Figure 2 Workflow of SDS

There are many methods or technique by which the shuffling can be done. Out of the various shuffling algorithms, perfect shuffling technique is one of the most famous and widely used algorithms. Initially perfect shuffle was proposed as an interconnection pattern (Stone 1971), being used in parallel processing to establish connections between the registers and processing units on computational speed. It was stated that the perfect shuffle algorithm has a wide variety of applications. Graham et al. (1983) have proposed a model which has been termed as Perfect Shuffle Crypto Algorithm (PSCA) that holds many interesting properties (John 2002). In PSCA perfect shuffle algorithm is used as a security algorithm to secure the messages during transmission. A minor modification has been tried over the PCSA model, to increase its security further more. Initially a little description of PSCA and its working have been explained in the subsequent section. In the next section, the enhancement over it has been explained.

3.2.2 Shuffle exchange network model

The shuffle exchange network model has been built based on two routing functions of shuffle and exchange (Ernastuti 2009). Here, the shuffling process is alone taken for study. The working of shuffle exchange network, is illustrated through an example. The example considers the scenario where 8 processors or nodes need to be shuffled and the value of N is 8. The perfect shuffling is done among the 8 processors or nodes and are given in the figure 3 (a). The inverse perfect shuffle does the opposite, to restore the original ordering and is given in figure 3 (b).

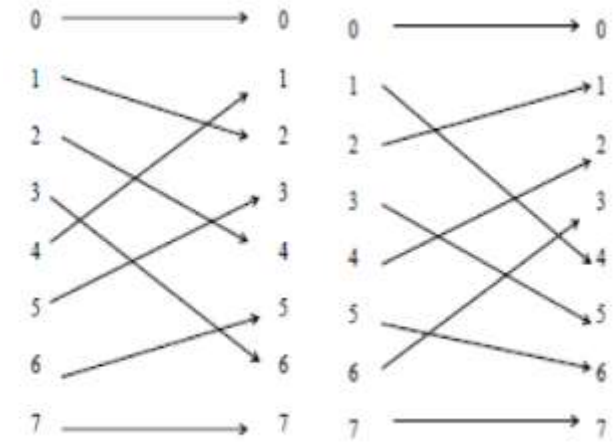

Figure 3 (a) Perfect shuffle (b) Inverse perfect shuffle

The shuffling and the exchange function can be implemented as either a recirculating network or a multistage network. An example of a shuffling function using the recirculating network is given in figure 4

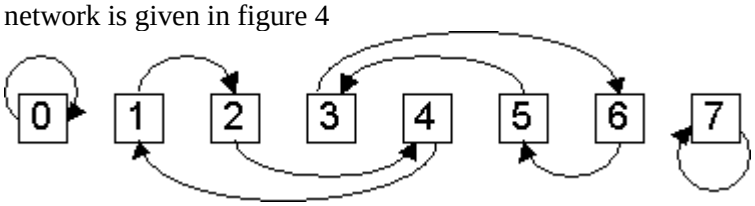

Figure 4 Shuffle exchange recirculating network for N=8

The shuffle exchange network does the processing in two kinds of stages termed as shuffle and exchange. The shuffle component establishes a link between node i to node j. A fixed permutation procedure is used to establish the link as given in the table 1

Table 1 Permutation procedure

p (j) = 2 * i	0<=i<= 2^n/ 2-1
p (j) = 2*i + 1 - 2^n	Other

Values of i and j specify the old and new position values. The computed value of j specifies the position to which the old value is moved.

3.2.3 Procedure followed in DS

Instead of the fixed shuffle as used in perfect shuffle, the proposed model tries to bring in a dynamic shuffle using a key. The procedure can be stated as a permutation cipher which is another form of transposition cipher. Permutation cipher acts on blocks of letters and not on the whole of plain text or cipher text. When stated mathematically, a permutation is a rule that tells of how to rearrange a set of elements. The permutation is performed based on an encryption key that can be calculated in two ways:

i. Secure Dynamic Shuffling (SDS) model – the encryption key is given by the user which is a permutation cipher that rearranges the packets
ii. Secret Shared Dynamic Shuffling (SSDS) model – the key is shared by the CU and CSP.

Both the proposed methods has many benefits when compared to the static shuffling. The benefits are explained through an example.

3.2.3.1 SDS

Based on the number of packets, the CU enters a permutation cipher. This is used as the key, based on which the data packets are shuffled. In the example, there are 8 packets, and hence the CU needs to give a permutation cipher of 8 values. In figure 4.5 (a) the key used as the permutation cipher is 8,4,3,7,6,2,5,1. Inverse shuffling can be calculated by performing the inverse operations. The decryption key or the inverse key can be calculated using the algorithm 1. Here, en_key – specifies the encryption key, len – specifies the length of the encryption key, and de_key – specifies the generated decryption key. To execute the algorithm, the encryption key is given as the input to the algorithm.

Algorithm 1: DECODE algorithm for performing the inverse shuffle

Input: en_key, len

Output: de_key

```
1.  begin
2.    for i ← 0 to  len do
3.          de_key [en_key[i]] = i
4.        end
5.    return de_key
6.  End
```

Thus, the dynamic shuffling was performed using the SDS model. Here, some of the points that needed to be taken care of are

- CU should give a permutation cipher as input, that needs to be as long as the number of nodes
- CU needs to remember the input sequence so as to use it, to get the decryption key
- Has a single point of vulnerability, as the whole system depends on the CU and on the permutation cipher input.

These issues are taken care of in the next model, where the secret to shuffling and un shuffling is shared.

3.2.3.2 SSDS

To secure the encryption process and to avoid a single point of vulnerability, SSDS model uses the concept of "shared secret". The encryption key is split among the CU and the CSP. Along with shared secret approach, the SSDS model uses the principle of random numbers. To have a dynamic generation of the permutation cipher, encryption keys are used. And to have an unpredictable generation of the permutation cipher, SSDS model follow the Pseudo Random Number Generators (PRNG). Using random numbers, initiates the following issues to be taken care of:

(a) Random number generators take in a seed value, using which the output sequences are generated

(b) Randomness of the sequence is dependent on the seed value and is held by one entity

(c) Seed value needs to be protected, failing which the random numbers become predictable

(d) Same seed value yields the same sequence of random numbers

To overcome these issues, the seed value is calculated using a Pseudo Random Function (PRF). The PRF generates a random number of fixed length and this value is added with the seed value to generate the random sequences. The PRF function takes the UUID value from the CSP. It is added with the seed value given by the CU. Figure 4.5 (b) represents an example when SSDS is followed.

Example: Let there be N=8 nodes that need to be shuffled and connected. The interconnection structure of these nodes using the perfect shuffle were depicted in the figure 4.3 (a). The same set of nodes when shuffled by using the proposed models, would vary based on the encryption key. This is depicted as below in figure 5.

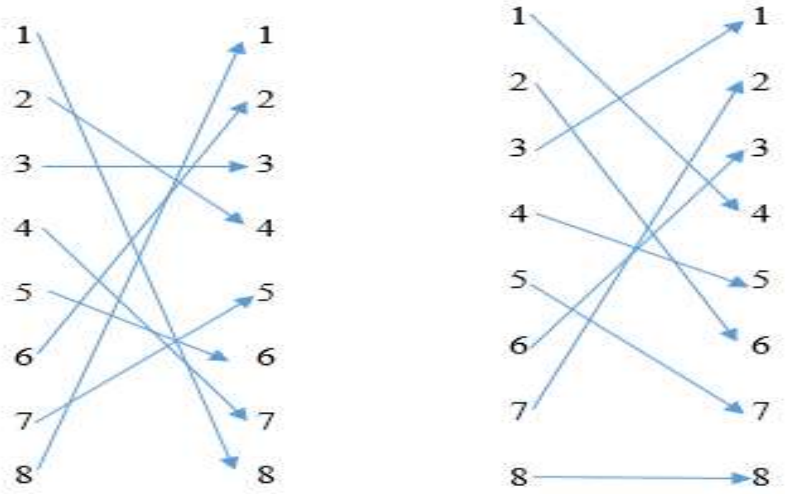

Figure 5 (a) SDS (b) SSDS

When the security is concerned, the randomization increases the security of data. The two main benefits are the:

(a) Dynamic permutation of the packets based on the key

(b) Correlation among the neighbouring data being very less

When the user data are split and shuffled, the individual packets should be dispersed as far as possible. A fixed shuffle will not help in randomization and it may help the attacker in deducing the order of packets, thereby revealing the user data.

3.2.3.3 Use of an AI component

The method of shuffling to be used is decided by an AI component that is present in the network. The component looks on the type of data and the source of the data. The type of data may reveal its sensitivity. But it may cause some privacy and security issues. Hence the data can be checked for their source. If the data has been received from a sensitive source, then the second methos of SSDS is chosen. If the source of the data is from an individual, then it can be considered as less sensitive and the first method of SDS can be applied. In the case where the data is being received from a source like an organization or a government setup, then the data might be considered sensitive.

Hence in such cases the second method of SSDS can be applied. This method uses the random number generator approach so as to randomize the shuffling process. The method also uses the secret sharing concept which overcomes the single point of failure or attack model. Though the overhead of this method maybe more when compared to the SDS method, the security parameter is more efficient in the second method.

3.3 PERFORMANCE ANALYSIS

The performance of the proposed model is done based on three measures. The first is done based on the distance measure. It measures the correlation between the packets when shuffling is done. Next the analysis is done based on the security parameter of the proposed model.

3.3.1 Distance measure

To measure the randomization property of the proposed model, Euclidean distance measure is used. The Euclidean distance is calculated for the perfect shuffle and for the SDS model using the formula as stated in the equation (4.1).

If $\mathbf{p} = (p_1, p_2, \ldots, p_n)$ and $\mathbf{q} = (q_1, q_2, \ldots, q_n)$ are two points in Euclidean n-space, then the distance (d) from **p** to **q**, or from **q** to **p** is given by the Pythagorean formula:

$$\mathrm{d}(\mathbf{p},\mathbf{q}) = \mathrm{d}(\mathbf{q},\mathbf{p}) = \sqrt{(q_1 - p_1)^2 + (q_2 - p_2)^2 + \cdots + (q_n - p_n)^2}$$

$$= \sqrt{\sum_{i=1}^{n} (q_i - p_i)^2}.$$

(1)

Table 2 gives the Euclidean distance for perfect shuffle model and the SDS model. As an instance, the model uses n=8 packets of data that are to be shuffled. In total there are 8! i.e. 40,320 permutations that are possible. The Euclidean measure was calculated for all these 40,320 combinations. The average measure was calculated to be 8.98 and the maximum distance measure was 12.96. The graph in figure 6 depicts the comparison of the Euclidean distance between perfect shuffle model and SDS model for 4 such possible combinations out of the maximum of 40,320 combinations as an example.

The perfect shuffle has a static distance measure of 5.29, and the proposed SDS model has a dynamic distance measure which varies depending on the encryption key. A comparison of the Euclidean distance measure for the perfect shuffle and for the proposed SDS model is given in the figure 6.

Table 2 Sample input and their corresponding Euclidean measure

S. No	Input data	Perfect shuffle	SDS model
1.	1,2,3,4,5,6,7,8	5.29	11.49 (For key 1)
2.		5.29	8.25 (For key 2)
3.		5.29	9.06 (For key 3)
4.		5.29	9.90 (For key 4)

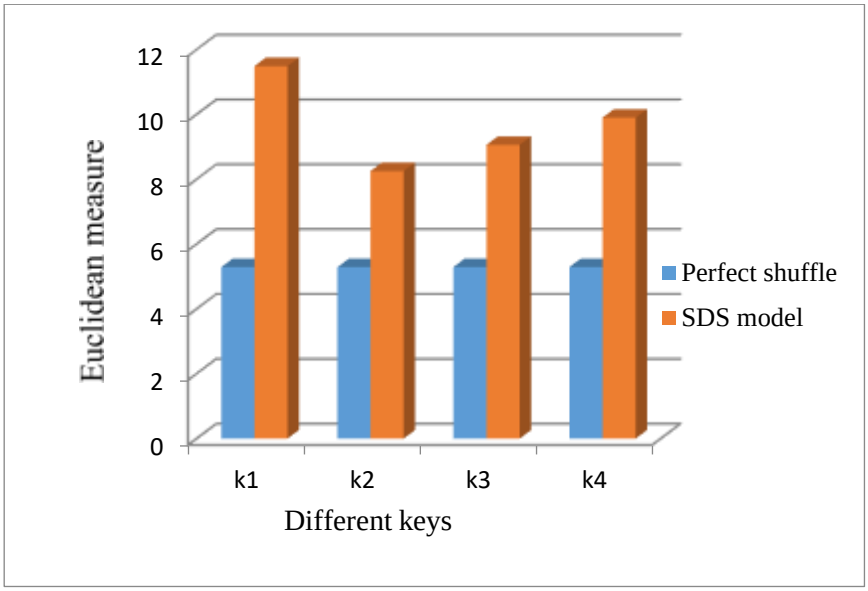

Figure 6 Comparison of the distance measures

3.3.2 Analysis of the security feature

The main aim of our work is to provide a secure storage service to the CU. The user data can be attacked at two places either during the transmission or at rest, in the place of its storage in the cloud itself. The proposed model works is able to overcome these two attacks.

3.3.2.1 Secure data transmission

There are a lot of attacks that take place when the data are in transmission on the internet. Most of the CSPs and the personal clouds in particular make use of REST API to get the data from the customer. These functionalities provide an easy and attractive feature, by which an attacker can attack the storage providers. The usual procedure followed to upload the data to the CSP is through the web browser. These browsers by themselves have a lot of loop holes. There have been many works that were developed to protect the data that goes through the web browser. There have also been a number of attacks taking place in the web service (Roberto 2014). Even when the data are encrypted and sent over the browser, attacks are made on it. The browser needs to be updated often, as they frequently encounter security issues or it may come across a new attack. Hence, a security patch needs to be done frequently. The researchers at the Google team found a bug present in the SSL 3.0 protocol. The exploit could take advantage to intercept the critical data that was believed to be encrypted between client and server. There has been a man-in-the-middle attack where the attacker was able to decrypt the secure HTTP cookies. Google called this the POODLE (Padding Oracle On Downgraded Legacy Encryption) attack (Meiko 2009).

To overcome these limitations, in the proposed model the data upload and download operations are not performed on raw data. The data undergoes a series of operations from which the hacker can never reconstruct the original data. One of the main vantages of the

splitting operation which splits data in to a number of packets after converting them in to byte streams are discussed below:

Let m – be the number of packets to be split

P_1, P_2 … P_m – be the different individual packets after splitting

An attacker needs to get all the packets in order to reconstruct the data. If the size of 'm' is made large, then getting hold of all the packets and reconstructing them become impossible. This solution has a very small computational overhead of O(m) and this becomes negligible when the security parameter is concerned.

3.3.2.2 Secure data storage

The data need to be protected at the place of its storage and from the internal components. In the proposed model, the data is being handled by the two entities.

CSP

- The CSP may get attacked or it may attack the user data
- Leads to the attacker getting hold of the packets
- But reconstruction of the data in its meaningful form is not possible.
- Due to the fact that the packets are shuffled in an order which is not known to the CSP.

CU

- The packets are shuffled in an order that is known only to the CU.
- To avoid the CU from leaking the permutation cipher, SSDS model uses a shared secret approach

3.3.3 Benefits of the proposed lightweight method

When it comes to the traditional cryptographic techniques, the encryption algorithm should satisfy a set of specifications and conditions to provide high security. Based on the various literatures that were analysed, the following are the criteria that needs to be fulfilled (Fursan 2021) by any of the newly developed algorithm:

- Encryption of the entire character set
- Encrypting each plain-text character into a special sequence
- There should be strong encoding
- The encryption approach should be complex

The proposed method satisfies all the above stated criteria. As stated in the section 3.3.1, as the distance measure increases, it increases the security parameter. This measure indicates of how similar or dissimilar two or more sets of data are. It was stated that (Lo, Owen 2021) when the distance measure is low, then the correlation between the data is more. This gives the attacker an upper hand to attack the data. When the similarity between the plaintext and the encrypted text is more, it helps the attacker to find a pattern between both the texts. Using this pattern, it becomes easier to the attacker to guess the plaintext. Greater or higher the distance value, means that more dissimilar the two sets are. Hence, the proposed method improves the security parameter.

If the shuffling is done in a static manner as given in figure 3, this implies that the system is more predictable. The system might have a biased distribution (Moatsum 2022), with a strong correlation. When the system is predictable, there is a possibility of an estimation attacks. As lightweight techniques are used, the overall overhead is reduced.

3.4 POSSIBLE SHUFFLINGS

There are other possible methods by which the shuffling can be done. In this paper the bit streams are considered as packets and the shuffling is carried out on them. The bit streams can be converted into blocks. And the shuffling can be done within these blocks. If more number of blocks are available then shuffling can be done multiple blocks, that is a inter block shuffling.

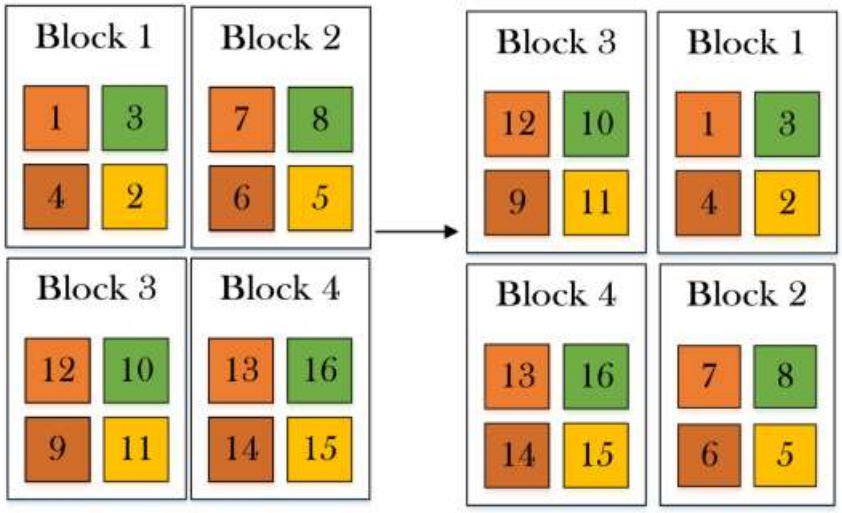

Figure 7 A snapshot of the block shuffling scenario

The figure 7 gives a snapshot of the block shuffling scenario. There are 4 blocks with data packets in it. These 4 blocks can be shuffled as shown. In a similar manner the data packets inside each block can also be shuffled. In a similar manner, the data packets of one block can be interchanged with data packets of other blocks. All the mentioned scenarios can be done so as to increase the randomization. The shuffling method can be chosen as per the user needs and this information can be maintained by the user.

4. CONCLUSION

This work has proposed two new models SDS and SSDS. Both offer a storage service to the user by using lightweight techniques. The method addresses the main problem of security. We have tried to provide a solution to this feature without making use of any of the cryptographic techniques, which incur a high overhead. The security of data is maintained during transmission and at the place of its storage.

REFERENCES

[1] Al-Souly, H. A., Al-Sheddi, A. S., & Kurdi, H. (2013) "A Lightweight Symmetric Encryption Algorithm for Secure Database". International Journal of Advanced Computer Science and Applications, 53-62.

[2] Ateniese, G., Burns, R., Curtmola, R., Herring, J., Kissner, L., Peterson, Z., & Song, D. (2007) "Provable data possession at untrusted stores". In Proceedings of the ACM Conference on Computer and Communications Security. pp. 598-610.

[3] Badger, Lee, Tim Grance, Robert Patt-Corner, and Jeff Voas. (2011) "Draft cloud computing synopsis and recommendations." NIST special publication 800:146.

[4] Buchade, A. R., & Ingle, R, (2014) "Key management for cloud data storage: Methods and comparisons". Fourth International Conference on Advanced Computing & Communication Technologies (ACCT), pp. 263-270, February.

[5] Eisenbarth, T., & Kumar, S, (2007) "A survey of lightweight-cryptography implementations", IEEE Design & Test of Computers, 24(6).

[6] Ernastuti and Ravi A.S, (2009) "The Application of ELC Numbers to Golden Cryptography", Proceedings of the 5th International Conference on Information and Communication Technology and Systems (ICTS), pp. 329-334.

[7] Fursan Thabit, Associate Prof Sharaf Alhomdy, Abdulrazzaq H.A. Al-Ahdal, Prof Dr Sudhir Jagtap (2021), "A new lightweight cryptographic algorithm for enhancing data security in cloud computing", Global Transitions Proceedings, Volume 2, Issue 1, 2021, Pages 91-99, ISSN 2666-285X, https://doi.org/10.1016/j.gltp.2021.01.013.

[8] Goyal, V., Pandey, O., Sahai, A., & Waters, B, (2006) “Attribute-based encryption for fine-grained access control of encrypted data”. In Proceedings of the 13th ACM conference on Computer and communications security pp. 89-98, October.
[9] Graham L.R. and William M.K., (1983) “The Mathematics of Perfect Shuffles”, Advances in Applied Mathematics. pp. 175-196.
[10] Jin Li, Gansen Zhao, Xiaofeng Chen, Dongqing Xie, (2010) "Fine-grained access control system with user accountability in cloud computing", 2nd IEEE International Conference on Cloud Computing Technology and Science, pp. 89-96.
[11] John E. and Hongbing. F, (2002) “The Cycles of Multiway Perfect Shuffle” Discrete Mathematics and Theoretical Computer Science. pp. 169-180.
[12] Kui Ren, Cong Wang, Qian Wang, (2012) "Security Challenges for the Public Cloud", IEEE Internet Computing, vol.16, no. 1, pp. 69-73, February.
[13] Lo, Owen, William J. Buchanan, Paul Griffiths, and Richard Macfarlane. (2018): "Distance measurement methods for improved insider threat detection." Security and Communication Networks 2018 1-18.
[14] Mark D. Ryan, “Cloud computing security”, (2013) Journal of Systems and Software, v.86 n.9, p.2263-2268, September.
[15] Meiko Jensen, Nils Gruschka and Ralph Herkenhoener, (2009) “A survey of attacks on web services” Computer Science - Research and Development (CSRD), Volume 24, pp. 185 -197, Nov.
[16] Moatsum Alawida, Je Sen Teh, Abid Mehmood, Abdulhadi Shoufan, Wafa’ Hamdan Alshoura, (2022) “A chaos-based block cipher based on an enhanced logistic map and simultaneous confusion-diffusion operations” , Journal of King Saud University - Computer and Information Sciences, Volume 34, Issue 10, Part A, 2022, Pages 8136-8151, ISSN 1319-1578, https://doi.org/10.1016/j.jksuci.2022.07.025.
[17] Mell. P and T.Grance, (2011) “The NIST definition of cloud computing”(NIST Special Publication 800-145). Gaithersburg, MD: National Institute of Standards and Technology, US Department of Commerce.
[18] Rizvi. S. , K. Cover and C. Gates, (2014) "'A Trusted Third-Party (TTP) Based Encryption Scheme for Ensuring Data Confidentiality in Cloud Environment", In Procedia Computer Science vol. 36, pp. 381-386.
[19] Roberto Baldwin (2014) “A Web encryption vulnerability opens ‘encrypted’ data to hackers Tweet”, http://thenextweb.com/google/2014/10/15/web-encryption-vulnerability-opens-encrypted-data-hackers/#!newthread807995. Accessed on January 2023
[20] Srisakthi.S. , and A. P. Shanthi, (2015) "Towards the Design of a Secure and Fault Tolerant Cloud Storage in a Multi-Cloud Environment." Information Security Journal: A Global Perspective Taylor & Francis pp 1 – 9.
[21] Stone, H. S, “Parallel processing with the perfect shuffle”, (1971) IEEE transactions on computers, 100(2), pp.153-161.
[22] Sushmita Ruj, Milos Stojmenovic, Amiya Nayak, "Decentralized Access Control with Anonymous Authentication of Data Stored in Clouds", IEEE Transactions on Parallel & Distributed Systems, vol.25, no. 2, pp. 384-394, Feb. 2014.
[23] Wang X.A and W. Zhong, (2010) "A New Identity Based Proxy Re-Encryption Scheme," International Conference on Biomedical Engineering and Computer Science, Wuhan,pp.1-4.
[24] Yin X. C., Z. G. Liu and H. J. Lee, (2014) "An efficient and secured data storage scheme in cloud computing using ECC-based PKI," 16th International Conference on Advanced Communication Technology, Pyeongchang, pp.523-527.
[25] Zhu, G. J. Ahn, H. Hu, S. S. Yau, H. G. An and C. J. Hu, (2013) "Dynamic Audit Services for Outsourced Storages in Clouds," in IEEE Transactions on Services Computing, vol. 6, no. 2, pp.227-238.