JKAU: Comp. IT. Sci., **Vol**. **8** No. 2, pp: 69 – 92 (1440 A.H. / 2019 A.D.)
Doi: 10.4197/Comp. 8-2.6

A Framework and Cryptography Algorithm for Protecting Sensitive Data on Cloud Service Providers

Mariam O. Alrashidi and **Maher Khemakhem**

Faculty of Computing and Information Technology, King Abdulaziz University
Jeddah, Saudi Arabia
mariamuoh23@yahoo.com

Abstract. Most companies are sceptical about the security and insurance measures offered by cloud services and are reluctant to store sensitive data, such as employee records, in the cloud. Thus, more effort is needed to support the security of information in cloud computing. This paper proposes a cryptography algorithm called "random algorithm" because it is built on the idea of randomising the encryption of uploaded files among four encryption algorithms. The proposed fragmentation technique helps add security and privacy to cloud storage applications. Based on earlier studies, we have created a file-level fragmentation technique that does not work at the database level, in contrast to commonly employed approaches in the case of fragmentation techniques such as horizontal fragmentation, vertical fragmentation, and hybrid fragmentation, which do work at the database level. The proposed encryption algorithm and fragmentation technique work within an integrated security framework that includes a user authentication gateway that encrypts user registration data through a cryptography algorithm called the Rivest-Shamir-Adleman algorithm (RSA). The results of the proposed security framework were positive, as it contributed to reducing the encryption time and decoding time by approximately 99%, compared to earlier studies

Keywords: Data security, Cloud computing, Fragmentation, User privacy, Encryption algorithms.

1. Introduction

Cloud computing has become part of users' lifestyles because sources such as data storage and computer systems are accessible as needed through networks that offer various computing services that do not require resources stored on customers' personal computers. Cloud computing also simplifies the sharing infrastructure among multiple users, who share platforms and services [1]. Cloud computing architecture is composed of four cloud-deployment models: private clouds, public clouds, hybrid clouds, and three-cloud services that include infrastructure as a service, platform as a service, and software as a service [1]. The history of cloud computing [2] began in the 1960s as John McCarthy, a scientist in the computing field, expressed the idea by saying, "computing might be organized to become a public service one day [2]."

We can summarise the weaknesses of cloud computing in two key points [2]:

1) The problem of internet availability is one of the issues faced by developing countries because cloud services require constant internet access, not only to make the service useful but also to ensure service quality while using it.

2) The concern over information security and privacy has some users worried about the

possibility that others could read or steal their files and information.

The main goal of this research is to build a framework for protecting sensitive data on a cloud service provider. This objective is achieved via the following sub-objectives: First: identifying sensitive data; second, creating a method for protecting the identified sensitive data; and third: building the framework.

According to [3], there is a hurdle to the spread of cloud computing technology, as the vast majority of companies have doubts regarding the security measures used in such technology and, therefore, refuse to store their confidential files using such technology. Accordingly, in this paper, a proposal is presented for a security framework for cloud computing that includes a random encryption algorithm and a novel fragmentation technique at the level of the encrypted files rather than using fragmentation at the database level as done with already familiar fragmentation techniques. There have been a number of algorithms and techniques proposed to solve the problem of data security in cloud computing, as discussed in the Earlier Studies section, and their performance is compared with the security framework proposed in this paper.

Although this research has achieved its goals, there were some unavoidable limitations. The first was time: this research was concerned only with data security and the files stored in cloud warehouses, so Infrastructure as a Service (IaaS), Platform as a Service (PaaS), and the extent of their security is beyond the scope of this research due to the time limits imposed for its completion. Second, the need for the privacy of certain files is determined by the user who uploads the files, and there is no algorithm to recognise which files must be protected. The research methodology of this study includes a historical approach based on analysing sensitive data fragmentation techniques and studying a set of current technologies designed for data security within the cloud. Also, the programming and testing approach followed for implementing and testing the proposed security framework has as its main goal the protection of data within cloud storage. The steps followed to achieve the objectives of the research were as follows: (1) Enhancing sensitive data fragmentation technique and (2) Designing a framework for protecting sensitive data on cloud service providers. Section 2 of this paper will discuss the theoretical background of the concept of encryption, the concept of cloud computing, the process of fragmentation, and specific encryption algorithms used in cloud computing. Section 3 presents earlier studies and a critical review of these studies. Section 4 is concerned with describing the proposed framework to protect sensitive data in a cloud service by identifying such data and implementing a technique to protect it by enhancing a sensitive data-fragmentation technique. Section 5 presents the results and analyses them through comparison with earlier studies and academic discussions and ensures the framework's effectiveness in protecting confidential data by preventing illegal users-so-called snipers-from spying on user files by browsing or downloading them. Finally, Section 6, the conclusion, gives a brief summary of the paper and discusses the findings.

2. Background

A. Encryption Concept

Encryption is the process of using an algorithm, called a cryptography algorithm, to convert plain text into an incomprehensible code and later convert it back to plain text [4]. The text relies mainly on the use of a numerical value, called the key, which is part of the

encryption algorithm [4]. In [5], encryption science is attributed to Arabs and is defined as the conversion of ordinary text to text not understood by those who do not know the particular encryption method used; *i.e.*, only a person familiar with the encryption method and its rules would understand it. The English word "cryptography" is derived from the Latin "cryptographia" and can simply be defined as a way to hide data. The science as described in [5] circulated in the Arab world uses the term "blindness", while the deciphering concept is known by the term "blind extraction".

B. Fragmentation Process

Fragmentation, according to Pinal Dave, is warehousing data in a non-contiguous way that is classified into two types [6]:

1) Internal fragmentation, which occurs when records are stored in a single worksheet in a non-consecutive fashion. This often occurs as a result of modifying data through various processes applied to a database, such as the deletion or insertion of data.

2) External fragmentation, which happens when data is stored in a non-contiguous way in physical storage or a disk, which can cause multiple switching.

The second type of fragmentation is also known as "logical fragmentation" [7]. This takes place when a server produces a binary tree data structure for an index and links whole data pages at the leaf level of the constructed tree in a logical order by applying a doubly linked list; the logical fragmentation then happens "when the pages in this doubly linked list are not contiguous in the index" [7].

In [8] defined fragmentation as the breakdown of a relation into smaller parts that are treated as separate units. Implementing a fragmentation technique improves the performance of distributed databases and reduces query execution time. In addition, Verma offers an overview of three types of fragmentation techniques, as follows:

1) Vertical fragmentation splits a relation into portions which include subsets of its attributes. The subsets are placed vertically beside the primary key of the relation.

2) Horizontal fragmentation breaks up a relation into portions along its tuples/rows horizontally.

3) Hybrid fragmentation is, as the name suggests, a hybrid of horizontal fragmentation and vertical fragmentation.

C. Encryption Algorithms in Cloud Computing

In [9], classified cryptographic algorithms into three types:

1) Symmetric key algorithms, including Data Encryption Standard (DES), Advanced Encryption Standard (AES), Ron's Code, triple DES, and Blowfish.

2) Asymmetric key algorithms, including RSA, the digital signature algorithm (DSA), Elliptic-curve algorithm (EC), Diffi-Hillman, and ElGamal.

3) Hash functions, including the message digest and the secure hash algorithm.

i. Advanced Encryption Standard (AES) Algorithm [10]

The AES algorithm is a symmetric encryption algorithm that uses a strategy incorporating encryption and decryption operations. The number of times the rounds are repeated is determined by the size of the selected key. Stages within rounds include key expansion, the first round involving Add-Round-Key, rounds involving repeated operations of Sub-Byte, Shift-Row, Mix-Column, and Add-Round-Key, and the last round involving Sub-Byte, Shift-Row, and Add-Round-Key. There are some important

algorithm specifications of AES as written in [10]; first, AES encryption is easily implemented and is the fastest method for 128-bit keys, but it slows if the key length is increased; second, the memory needs of the AES algorithm are less than those of the Blowfish algorithm; third, it becomes more secure if the key length is increased; i.e., a 256-bit key is slower than a 192-bit key but is more secure; finally, AES is very strong against square attacks, key attacks, key recovery attacks, differential attacks, and smash attacks.

ii. The Rivest–Shamir–Adleman (RSA) Algorithm

The RSA algorithm is based on a mathematical approach producing two keys—a public key to encrypt text and a private key to decrypt text—where the length of the keys can reach 1024 bits [11].

iii. Data Encryption Standard (DES) and Triple DES (3DES) Algorithms

A set of characteristics indicated in [12] for the DES algorithm is as follows:

1) DES has a 64-bit key size and 64-bit block size.

2) Numerous attacks had broken the DES, resulting in the 3DES upgrade.

3) 3DES has an encryption process which is applied three times, similar to the steps applied in the DES encryption process and has a 64-bit block size and 192-bit key size.

4) 3DES takes more time to implement compared to DES.

iv. Blowfish Algorithm

Blowfish is a symmetric cryptographic algorithm established by Bruce Schneier in 1993 [13]. As stated in [13], Blowfish is faster than the DES algorithm. Blowfish receives a 64-bit block as input to the encryption process. Also, the algorithm uses a key ranging from 32 bits to 448 bits [13]. This makes it faster compared to the DES algorithm. In addition, it has 16 to 17 rounds during text encryption.

v. Twofish Algorithm

In [14], the authors clarify the scalability of different Symmetric Key Cryptography algorithms, and these SKC algorithms are analysed relying on encryption execution and memory usage. The main goal of the memory usage test in [14] is to quantify the effectiveness and execution of SKC algorithms. The encryption rate ought to be as low as feasibly possible for better performance. Figure 1 show memory usage and encryption performance of the following algorithms: DES, 3DES or T-DES, Blowfish, International Data Encryption Algorithm (IDEA), Tiny Encryption Algorithm (TEA), Carlisle Adams and Stafford Taveres Algorithm (CAST), Rijndael, Rivest cipher 6 (RC6), Serpent, Twofish, and MARS. The results of this test in terms of encryption performance, from best to worst, were as follows: Serpent, IDEA, MARS, DES, RC6, CAST, 3DES and Twofish had the same encryption performance, followed by Blowfish, TEA and, finally, Rijndael. The results of this test in terms of memory usage, from best to worst, were as follows: first was Twofish, second was DES, TEA, and Rijndael, third was 3DES, CAST and RC6, fourth was Blowfish and Serpent, fifth was IDEA, and, finally, the MARS algorithm.

The paper also mentioned some limitations of SKC algorithms; the limitations of a range of algorithms used in random encryption algorithms proposed in our paper are as follow [14]:

• DES: More vulnerable to systematic attacks.

• 3DES: Is vulnerable to three types of attacks: differential attack, related-key attack, and meet-in-the-middle attack.

• Blowfish: With feeble keys in 4 rounds if it is exposed to differential attacks.

• Twofish: It is prone to chosen-key attacks that influence a decrease in security when utilised for hash functions.

The Twofish algorithm is a 16-round Feistel network with a function made up of four key dependent 8*8-bit S-boxes, a fixed 4*4 distance separable matrix, a pseudo Hadamard transform, bitwise rotations, and a cautiously structured key schedule [15].

The work strategy of the Twofish algorithm consists of 6 components as follows [15]:

1. Feistel Network

A Feistel network is a general strategy for changing function F into an equivalent permutation. The principal building block of a Feistel network is the F function: a key dependent mapping of an input string onto a yield string.

In each round, a block named the source block is the input to F, and the yield of F is XORed with a block named the target block, after which these two blocks swap places for the following round.

• The Feistel cipher divides the input block into two halves: the left half L_i and the right half R_i, which are processed through different rounds to produce the ciphertext block.

• Each round portion of the input block is sent through an F function and afterward XORed with the other portion of the input block, at which point the two halves are swapped.

• For each round i, the inputs of round i are L_i and R_i, obtained from the previous round, and a subkey K_i, which is obtained from the key K.

• The substitution and the permutation originate from the handling inside the rounds.

• The output of the round function F is XORed with the left half of the data in each round.

• At the end of each round, the Feistel cipher swaps the left half and the right half for permutation.

• For each round i, the operation, which is iterative, can be mathematically expressed as follows:

• In the ith round:

$$L_i = R_{i-1} \quad (1)$$

$$R_i = L_{i-1} \oplus F(R_{i-1}, K_i) \quad (2)$$

The left half of the round output is nearly the right half of the previous round output, so L_i is equal to R_{i-1}. For the right half of the round output, R_i is actually the result of XOR between the left half of the previous round and the F function output. The F function inputs are the outputs of the previous round and the subkey K_i.

A comparison of the decryption process with the encryption process in the Feistel cipher revealed the following facts:

• The structure of the encryption and decryption processes is the same, except for the parameter values.

• The hardware and software are used for both encryption and decryption, with just a slight change in how the keys are used. For decryption, the ciphertext is the input and the subkeys K_i are used in reverse order; i.e., it uses K_n in the first round, K_(n-1)in the second round and so on until K_1 is used in the last round for decryption. Accordingly, one implementation can be used for both encryption and decryption, but with changes in the subkey inputs and the data inputs.

The Feistel cipher is a structure that many symmetric block ciphers use. There are similar design parameters for the Feistel cipher that can vary according to block cipher design. These design parameters are the block size, key size, number of rounds, the subkey generation algorithm and the round function design.

2. S-Boxes

An S-box is a table-driven non-straight substitution process utilised in most ciphering operations. S-boxes can be made either haphazardly or algorithmically.

3. MDS Matrices

A code over a field, named Maximum Distance Separable (MDS), is a direct mapping from a field component to other field components, delivering a composite vector of a+b components.

4. Whitening

Whitening is the strategy of XORing key material before the first round and after the last round.

5. Key schedule

The key schedule is the method by which the key bits are transformed into round keys that the cipher can utilise.

6. Pseudo-Hadamard transform

A pseudo-Hadamard transform (PHT) is a straightforward blending operation that runs rapidly in programming, giving two inputs, a and b.

Given two inputs, a and b, the 32-bit PHT is characterised as:

$$a' = a + b \bmod 2^{32} \quad (3)$$

$$b' = a + 2b \bmod 2^{32} \quad (4)$$

In Fig. 2, the two words on the left are used as the input to the g function after the rotation by 8 bits of one of them. The g function comprises of four byte-wide key-dependent S-boxes, trailed by a direct mixing step dependent on the MDS matrix. The result of the two g functions is consolidated using a Pseudo Hadamard Transform (PHT), and two key words are included. One of the words on the right is rotated by bit and, after that, the two are XORed into the outcome on the left. The left and right parts are then swapped for the following round. After 16 rounds, the swap of the last round is held, and the four words are XORed with four key words to obtain the ciphertext.

Table 1 summarises the most important features of the cryptographic algorithms discussed, such as algorithm classification, block size, number of rounds, and the key length for each algorithm.

Table 1. Important characteristics of encryption algorithms des, 3des, aes, rsa, blowfish, and twofish [16-21].

Algorithm	Algorithm classification	Block size	Key length	Number of rounds
DES	Symmetric encryption algorithm [16].	64 bits [16]	56 bits [16]	16 rounds [16]
3DES	Symmetric encryption algorithm [17]	64 bits [17]	168, 112 or 56 bits [17]	48 rounds [17]
AES	Symmetric encryption algorithm [18]	128 bits [18]	128, 192 or 256 bits [18]	10, 12 or 14 rounds depending on key size [18]
RSA	Asymmetric encryption algorithm [19]	At least 512 bits [19]	Greater than 1024 bits [19]	One round [19]
Blowfish	A symmetric encryption algorithm [20]	64 bits [20]	32 – 448 bits [20]	Fourteen rounds [20]
Twofish	Symmetric encryption algorithm [21]	128 bits [21]	128,192 or 256 bits [21]	16 rounds [21]

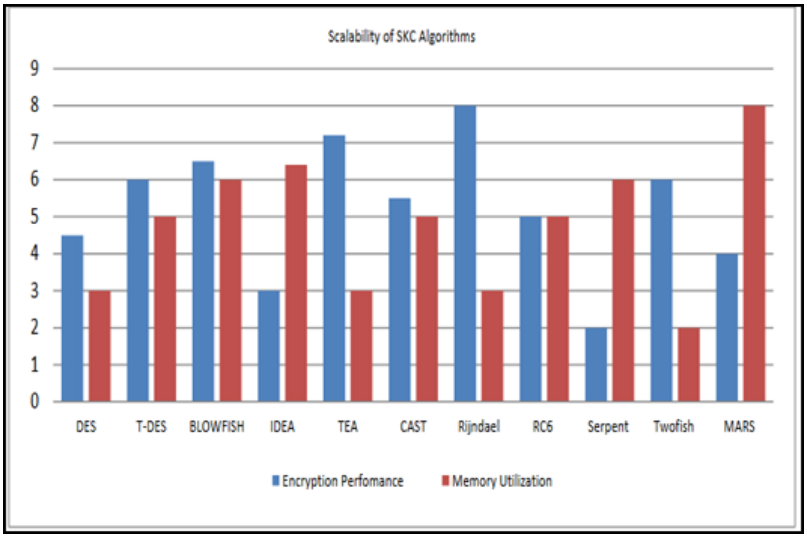


Fig. 1. Memory usage and encryption performance of SKC algorithms [14].

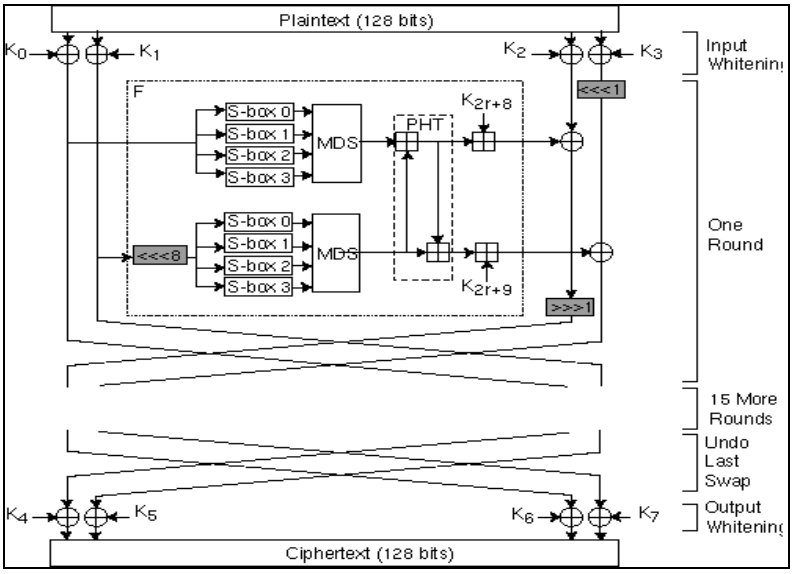


Fig. 2. Block diagram of Twofish algorithm [15].

3. Related Work

To ensure the security of their information in a cloud environment, a user must make sure that the connection is secure through high-security techniques. The user must also be sure that their service provider applies a high level of information security when data are transferred from a user device to the cloud.

In [22], the authors were interested in studying security issues based on the service models SaaS, PaaS or IaaS, and ways to solve these issues. They began by defining each model as follows and clarifying their respective issues:

1) Software as a Service (SaaS): defined as providing software as a service to users, such as storing user data. Security issues associated with this model include data

security, network security, and data confidentiality. The authors raised several questions: How will data be stored and where? What security steps and processes will be taken to protect and save data?

2) Platform as a Service (PaaS): defined as providing a platform as a service to users, such as Google Apps, which is now called G Suite, and Microsoft Azure. The problems with this model are typically the same as the issues suffered by service-oriented architectures (SOA) because it is based on the SOA model; issues include DoS attacks, dictionary attacks, man-in-the-middle attacks, *etc*.

3) Infrastructure as a Service (IaaS): defined as providing an infrastructure (physical and virtual computers) as a service to users, including space for storage and the operating system. The difficulties regarding the IaaS model concern the possibility of erroneous data being routed through the network or by means of infrastructure to the network of a hacker, as the infrastructure provides basic storage services such as storage systems.

The authors in [22] also mentioned some tricks hackers use to hijack data, such as remote access via bad SQL commands, denial of service (DoS) attacks and distributed denial of service (DDoS) attacks. The authors also offered a set of solutions and suggestions to solve the above issues to provide a secure cloud server, which included data security and control, network security, access control, data confidentiality and integrity, and other security precautions. One suggestion given was to provide methods that prevent or reveal the different types of snooping or data manipulation. They recommended several methods for network protection, including the use of secure sockets layer (SSL), managing user sessions, analysing uploaded or downloaded packets, and using firewalls, etc. Moreover, they referred to the importance of having access control techniques (e.g., identity management), access security techniques and auditable reports about user access. They also highlighted that the integrity and confidentiality of data can be maintained by selecting appropriate authentication and authorisation methods as well as implementing cryptographic algorithms and performing data replication.

In [23], studied algorithms employed for data-storage security in both cloud and desktop environments. They also discussed the advantages and disadvantages of algorithms such as Elliptic curve cryptography (ECC), RSA, Rivest Cipher 4 (RC4) and ElGamal. They mentioned the following data security issues in cloud computing.

• Data Integrity: data integrity is very important for data centres and is the responsibility of companies that manage and control data rather than of service providers.

• Data Theft: numerous copies are created from data stored in the cloud. Data theft is an increasingly important issue.

• Privacy Issues: data privacy and confidentiality are concerns that can easily be solved using encryption techniques.

• Malicious Application: occasionally, data storage applications are influenced by viruses, which, in turn, affect the rest of the data stored within infected applications.

• Data Loss: data loss can occur during storage when data are transferred from one platform to another within the cloud. Damage to data can also occur within servers.

The authors in [23] also classified data security into two levels: a user level that includes authentication of authorised users for data access and a vendor level that includes privacy and confidentiality, protecting data from loss during transmission and managing user access.

In [23], proposed a new technique called "quad security layers for cloud trust" for transmitting data in cloud computing by using a scheme with four layers to create a secure cloud environment. These layers include secure transmission of data, encrypted data and processing, a secure database shell, and internal and external log auditing. In [25], proposed an algorithm to ensure data protection and confidentiality while trading within the cloud. They improved two traditional encryption algorithms named Substituting a Cipher and Transposition Cipher. The steps of the proposed algorithm were the following: first, convert plain text to American standard code for information interchange (ASCII) to obtain a square matrix with a length greater or equal to the number of characters in the plain text; second, divide the previous constructed matrix into three parts: upper, diagonal and lower; third, read values in each matrix from right to left, where every matrix has three keys for encryption; fourth, substitute the original values with encryption values in the three matrices; fifth, read the encrypted message column; finally, transform the ASCII into its corresponding character values. Another study in this field that discussed issues related to data security in cloud computing was conducted in [26], which enumerated issues related to data warehousing and data sharing processes in a cloud environment, such as integrity, confidentiality, availability, and security. They proposed a technique to provide data confidentiality and integrity in the cloud using AES-256, the metadata of the uploaded file, and a hash algorithm (SHA-256).

Study Critique for [26]:

In [26], the method was explained clearly and in detail and was also supported by a diagram that described the process of verification of data integrity and the process of pre-encoding. Also, they referred to the experimental aspect used to carry out the proposed approach using the Amazon S3 cloud, as well as the device used during the practical experiments such as the processor type, space on the hard disc, *etc*. The programming language used to make the technical solution was Java. However, the paper would have been more effective if the authors had added a table displaying the ratio of validity of the proposed method in checking file integrity by testing the proposed method on a set of files, thus indicating the effectiveness and performance of the proposed algorithm. Finally, authors in [26] did not mention any limitations in the study, such as the type of files that could not be secured or verified its integrity by using the proposed technique.

In [27], Anitha and Vijayakumar experimented with a system aiming to guarantee the security of data stored within a cloud environment in a federated network. In a federated network, there are two or more clouds used to store and process data. The system model includes a user who uploads data in encrypted form, a key-generation process using an enhanced modified Feistel algorithm, and storage of data in an encrypted form using a Bloom filter-based data arrangement algorithm.

The encryption structure is described by the following equation:

$$c = pq + 2r + m. \quad (5)$$

Where m refers to the plain text, p refers to the secret key, q refers to multiple parameters which were unfortunately not described by the authors, and r is noise against brute force attacks. There are four steps included in the encryption structure in [27] based on the concept of homomorphism, including key generation, encryption, evaluation, and decryption. The encryption procedure in one round is summarised in the following steps:

1) Generating the cipher key based on a previously prepared matrix with a value called a secret key and the Feistel function F.

2) Obtaining the encrypted text by (1) taking the encrypted plain text as a matrix and (2) dividing the input matrix into two halves, left and right.

3) Applying the shifting-row operation to both halves of the matrix, followed by mix-columns, also on both halves.

4) Applying an XOR operation to the output results of the matrices to obtain the cipher text.

Study Critique for [27]:

One limitation of this research is its specific focus on federated clouds. That aside, however, the authors clearly defined the study target and specified the technique used, homomorphic encryption, with the aim of adding security to stored data and improving the performance of the federated cloud system using metadata. The study content was ideal, and it covered all aspects such as the contributions of the study, the proposed system model, and how each element in the system model worked (e.g., the suggested encryption scheme's method). The results of the study were clear and included both the encryption and decryption times for the proposed algorithm, thus making it one of the algorithms which will be used for comparison with the cloud storage system proposed in this paper. In conclusion, the authors summarised the proposed technique and referred to the system analysis and positive results compared to the other algorithms (*i.e.*, AES, DES, and Blowfish). The results of Anitha and Vijay's study are also presented in the results section and will be compared with the results of the security framework proposed in this paper.

The innovative method proposed in [28] includes two aspects of data security: authentication and a proposed cryptography algorithm. The authentication uses an image-sequencing password, for which an exact sequence of images must be remembered by a cloud user to ensure legitimacy. The cryptography algorithm is implemented by hiding data inside the images using randomised and anonymised privacy-preserving techniques and the steganography technique. Data encryption is performed using the following steps:

1) Entering an image as input, followed by finding the edges using a canny edge detection algorithm.

2) Transforming the collected dataset into a binary representation.

3) Preparing an array to store edges and their positions; then, randomly selecting one index from this array.

4) Using the selected index for message hiding and sending the hidden text stored in the file to the cloud and storing the array in the local end.

5) Performing decryption by matching the file to the array stored in the local end.

Study Critique for [28]:

The study can lead to confusion by readers, as in the analysis section the author compared the proposed technology with what was referred to as 'the existing system'. This leads to the following question:

In [28], which system is being compared with the proposed system?

The author further explains that the method used was divided into two parts by using image sequence authentication in lieu of a password and then using a cryptographic algorithm to maintain confidentiality. A detailed explanation of the proposed cryptographic algorithm follows: the purpose was to hide data in an image by using a set of

privacy technologies which combine a randomising privacy-preserving technique with a steganography technique. However, the author did not explain how these techniques worked theoretically. The analytical tool used in the simulations was CloudSim. In the Future Work section in [28], it suggested data classification to reduce the encryption time. The author's suggestion was, therefore, implemented in this paper, by classifying data into sensitive and non-sensitive categories for the same purpose and to reduce the burden on the system. Also, the author referred to many advantages of cloud computing, such as less user effort required for file management in addition to the security issues that limit the success of such a technique. The results of Deepika's study are presented in the results section of this paper and compared with the results of the proposed security framework.

While SCHSs have numerous applications in the field of human services, securing gigantic amounts of significant and confidential patient information and maintaining its security are important issues. Authors in [29], follow some primary steps to reach the actual solution; first, user validation dependent on public-key cryptographic systems performs an essential job in SCHSs by ensuring the patients' security; however, quantum PCs will destroy the security of such techniques; second, the rainbow signature is an important cryptographic algorithms that can resist hacks/ attacks on quantum PCs; however, it is defenseless against Differential Power Analysis (DPA) assaults. Finally, authors proposed strategies to exploit the countermeasures to protect Rainbow against DPA attacks.

The authors in [29], suggested a safe authentication scheme primarily based on a variant of Rainbow signature with a strong defense against DPA attacks. First, they offer techniques to make the most countermeasures to protect Rainbow signature from DPA attacks. They use a random vector to randomize the energy intake of private keys at some stage of calculating the first affine transformation; then, random variables are adopted at some stage of calculating central map transformation; finally, they take two random vectors by calculating the second affine transformation to randomize the power utilization of private keys. They ensure that the random variables are used in computations all through the primary invertible affine transformation, valuable map transformation and the second invertible affine transformation. Second, they proposed a secure authentication scheme based on the proposed algorithm for acquiring patient records. The proposed structure can be summarized briefly through the following steps:

1. Doctor A wants to gain the record of patient V.

2. Doctor A generates a random value R1 and makes use of his AES' key KA to convert V and R1 to the encrypted form, this process can be described by the following equation:

$$E = AES(R_1 + V, K_A) \quad (6)$$

3. Then, he makes use of his private Rainbow key to generate a signature of E and his identification ID_A, this process can be described by the following equation:

$$S = Rainbow(E + ID_A, P_A) \quad (7)$$

4. Then he sends S, ID_A to the center of the medical system.

5. The center uses the public key of A to affirm S, this process can be described by the following equation:

$$M = Rainbow^{-1}(S, P'_A) \quad (8)$$

Where the center makes use of the private key K_A to decrypt E', this process can be described by the following equation:

$$M' = AES^{-1}(E', K_A) \quad (9)$$

Then, the medical center obtains the document of patient V, in other words RecV.

6. After that, the medical center uses AES' key K_A of doctor A to encrypt RecV and R1, this process can be described by the following equation:

$$E = AES(RecV + R_1, K_A) \quad (10)$$

7. Next, The medical center uses his private key P_C to generate a Rainbow signature of E and his identification ID_C, this process can be described by the following equation:

$$S = Rainbow(E + ID_C, P_C) \quad (11)$$

8. At that point he sends S, ID_C, to A. Where, A uses the public key of the health center to check S, this process can be described by the following equation:

$$M = Rainbow^{-1}(S, P_C') \quad (12)$$

When the doctor needs to decrypt the data (E'), he uses the secret key K_A this process can be described by the following equation:

$$M' = AES^{-1}(E', K_A) \quad (13)$$

Table 2. Abbreviates the performance in [29], which shows that it takes just 4840 ns and 242 clock cycles for every signature generation.

The Signature scheme utilized in execution Testing	Message Size	Signature Size	Time Frequency	Clock Cycle	Execution Time
Rainbow(17,13,13)	26 Bytes	43 Bytes	50 MHz	242 clock cycle	4840 ns

The proposed scheme in [29] is a lot of efficient than usage of RSA and ECC algorithms. Compared with the previous work in the same field, the implementation in [29] with resistance to specific attack is 20% slower, because of anti-attacks against DPA attack.

In [30], they propose SecureLR, innovative framework which performs learning and forecasts on biomedical data without damaging the security and data privacy. The model created in [30] depends on homomorphic encryption procedures with hardware security supported through Software Guard Extensions (SGX). Its implementation demonstrates a sensible hybrid cryptographic resolution to deal with vital issues in conducting machine learning with public clouds. The paper outlined the research problem, which was the security and privacy of biomedical data and the need to resist attacks on such data. The study focused on safe predictive modeling in the public cloud environment, which includes the logistic regression algorithm. This algorithm (logistic regression algorithm) is used to predict diseases, for example, to predict the early diagnosis of myocardial thrombosis. The study discussed a set of existing technologies that sought to solve the security problem in the logistic regression algorithm, including differential privacy, federated data analysis and cryptographic methods. The study stated the defects of some techniques, such as homomorphic encryption (HME), which requires a high storage capacity and large computational expenses. Consequently, using this technique to analyze biomedical data is not practical. The authors defined Software Guard Extensions (SGX) as a security feature of Intel's (sixth era or later) processor design, which gives an equipment upheld area in order to provide secure and private computing, called an enclave. SGX utilizes an "inverse sandbox" structure strategy, which seals private codes, sensitive information, and other selected "insider facts" in the enclave's memory.

The study also discussed the defects of Software Guard Extensions (SGX) technology, including the possibility of exposure to a range of security attacks, such as controlled side channel attacks and cash timing attacks. The authors also referred to the objective of merging the best features of HME and SGX to assist biomedical academics make completely outsourced protected logistic regression evaluation effectively. They also defined logistic regression as a characterization calculation with wide applications in the biomedical informatics, including clinical choice support, hazard evaluation, and sickness categorization. The proposed model in [30] is comprised of four essential elements, with each one performing a task in the SecureLR protocol and performing operations contributing to its functionality and safety. Those elements were Information Proprietors, Cloud Service Providers (CSPs), Authorized Researchers (ARs), and Authentication Service Provider (ASP), which will be presented in more detail below:

• Information Proprietors: an organization that holds the biomedical information.

• Cloud Service Providers (CSPs): confided elements that give: (i) the re-appropriated storage of information (encoded, from DOs), (ii) the interface for Approved Scientists (ARs) to perform secure logistic regression over such datasets, and (iii) return the prepared model parameters in ciphertext to the ARs.

• Approved Specialists (ARs): people who get approval from an authentication service provider (ASP) to demand SecureLR benefits through the CSP.

• Authentication Service Provider (ASP): a confided entity which performs approval and authorization.

The SecureLR system is comprised of five key procedures as follows:

Procedure 1: Adjust and Circulate HME Parameters where these parameters sway numerous parts of the structure, including its degree of security, computational limit, and the size of encrypted information. These parameters should be chosen cautiously.

Procedure 2: Remote Validation and Key Conveyance; the remote confirmation procedure guarantees that the code executed in the enclave is trusted, and a protected channel between the enclave and the enclave maker (for example ASP) has been set up.

Procedure 3: Information provisioning and encryption through encryption and transferring preparing information (DOs→CSP1) in the proposed SecureLR protocol necessitates that the DOs encode all passages in their dataset (for example X and Y) utilizing the k1 provided by the pub confirmed enclave (facilitated by the CSP2).

Procedure 4: SecureLR Training Model which incorporates SLR algorithm under HME (CSP1, CSP2, ARs).

The authors in [30] stated that the size of the dataset determines the space complexity of the proposed protocol. Therefore, in particular the space complexity of SecureLR the protocol is given by the following equation:

$$O(m) + O(N * L) + O(N) \quad (14)$$

Where, m is the number of attributes, N the number of times the gradient increased and L the number of repetitions of the AISR algorithm.

In this section, a review of various studies has been presented. The first study, by S. Kaur and A. Singh, in [22], concerned the security issues based on the type of service model and the view of methods that might solve those issues. In addition, it discussed the

tricks applied by hackers to read data as unauthorised users. S. Bollavarapu and B. Gupta in [23] studied algorithms employed for data-storage security in both cloud and desktop environments. They also discussed the advantages and disadvantages of a set of encryption algorithms, among them were Elliptic Curve Cryptography (ECC), RSA, Rivest Cipher 4 (RC 4) and ElGamal, in addition to discussing a range of data security issues in cloud computing. M. Mushtaq *et al*. in [24] proposed "Quad Security Layers for Cloud Trust" that included 4 layers for the secure transmission of data, encrypted data and processing, a secure database shell, and internal and external log auditing. M. Shinde *et al*. in [25] proposed an algorithm to ensure data protection and confidentiality while trading within the cloud. They improved two traditional encryption algorithms named Substituting a Cipher and Transposition Cipher. Vyas and Modi [26] proposed a technique to provide data confidentiality and integrity in the cloud using AES-256, the metadata of the uploaded file, and a hash algorithm (SHA-256). Anitha and Vijaya Kumar [27] proposed a system model which included user-uploaded data in encrypted form, a key generation process using an enhanced modified Feistel algorithm, and storage of data in encrypted form using a Bloom filter-based data arrangement algorithm. Deepika [28] proposed an innovative method which included two aspects of data security: authentication and cryptography algorithms. The authentication used an image-sequencing password to ensure legitimacy. The proposed cryptography algorithm was implemented by hiding data inside the images using randomised and anonymised privacy-preserving techniques and the steganography technique. The examination of all of these previous studies will allow a thorough comparison to be made regarding the performance results previously obtained, in the mentioned studies, and the framework proposed in this paper. The proposed security framework differs from those proposed in previous studies in that it includes a random encryption algorithm combined with the binary fragmentation of files; a different approach from those attempted in the previous studies.

4. The Proposed Framework

This section presents the structure of the proposed secure cloud storage system framework. The implementation of the research method includes five parts:

1) Enhancement of sensitive data fragmentation technique.

2) Creation of an algorithm that contributes to protecting sensitive data.

3) Design of a framework for protecting sensitive data.

4) Implementation and testing of the proposed framework.

5) Evaluation of the proposed framework.

The programming language used to carry out cloud storage was PHP, which is an easy-to-learn programming language that allows web developers to create web pages that relate to databases for certain tasks such as storing user files [31].

Proposed Framework for Protecting Sensitive Data on a Cloud Service Provider

The framework shown in Fig. 3 shows all processes involved in the system to protect sensitive data, including user access to the system, data entry and encryption, storage in the cloud, and retrieval of data.

The range of specifications within the proposed framework includes:

1) Random encryption algorithms classified as a mix of symmetric key block ciphers and asymmetric key block ciphers.

2) RSA algorithm, which is classified as an asymmetric key encryption algorithm, used for password encryption.

3) A secret question and answer (written by user) used to add a high-security login option if the password is lost by the user. In addition, the secret answer is secured using the Twofish algorithm.

4) Finally, the proposed binary fragmentation technique used to cover the two aspects of system performance and data security used for file-level fragmentation.

All of the previous security features are security elements of the proposed security framework and contribute to making the security framework difficult to break by attackers. In the proposed cloud storage system framework, at least 8 characters are required for a password in order to provide the minimum amount of security. Also, the proposed storage system framework enables the user to change the password when they wish.

In the secure user authentication layer, shown in Fig. 3, the login data for authenticated users will be secured as follows:

1) The user will type the password in unencrypted form (during the registration process in the cloud).

2) The system will convert it into an encrypted format using the RSA algorithm (during the registration process in the cloud).

3) The password is stored in an encrypted form immediately after registration in a separate table within the system that has user_id, user_password, and user_email attributes.

4) A secret question and answer are created during the cloud registration process, in case the password is forgotten by the user, and then encrypted using the Twofish algorithm.

5) At the user login, the system will compare the password entered with the password stored; only a matched username and password will allow a login.

In the binary fragmentation layer, shown in Fig. 3, the fragmentation concept is used to fragment uploaded files (instead of fragmenting the databases as in earlier studies) into two parts, frag_A and frag_B, resulting in a strategy referred to as binary fragmentation at file-level. A database on the created cloud website is then used to store those parts of the file. This fragmentation strategy is combined with a random encryption algorithm selected by the system which is applied to the file entered by the user under a sensitive file classification.

In the file encryption layer, during the upload process, the previous binary fragmentation technique is applied only to user-specified confidential data. As a result, each sensitive uploaded file is divided into two parts, frag_A and frag_B, which are then encrypted in the file encryption layer. The encryption algorithm is selected randomly by means of generating a number from 1– 4. The numbers represent five types of encryption algorithm: AES (1), DES (2), 3DES (3), and Blowfish (4). Once the data are encrypted using one of the random algorithms, a table is populated with information about the file (*i.e.*, file owner, file number, and algorithm used for encryption) and is stored as background information to be used in decryption and defragmentation in the final layer. Figure 4 shows a detailed diagram of the proposed security framework.

After registration, the user can log into the system to upload or view their files, which are either confidential or not confidential (*i.e.*, sensitive files or non-sensitive files). Figure 5(a) shows the user interface view where non-sensitive data is saved in non-encrypted format

and sensitive files are fragmented and encrypted using random encryption algorithms, as shown in Fig. 5(b), which displays the form of the files in the database.

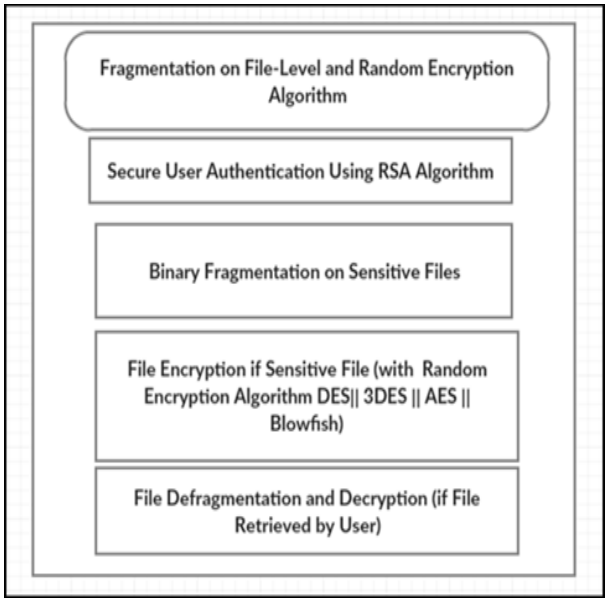


Fig. 3. Proposed framework to protect sensitive data on a cloud service provider (CSP).

5. Results and Discussions

Table 3 lists the equipment used in the testing and analysis of the proposed system.

Table 3. Hardware configuration for implementing and testing the cloud storage system framework.

Operating system	Microsoft® Windows® 8
Processor	Intel® Core™ i3-4030U (3M Cache, 1.90 GHz)
Memory	4 GB
HDD	500 GB
System type	64-bit

In order to implement the cloud framework, we created a cloud site that includes the following information in terms of hosting company, internet address, *etc*.

Domain: safe-storage.net

Web Hosting Service Provider: BlueHost

IP: 198.1.81.38

Location: United States of America

The names of the servers: ns1.benaahost.com and ns2.benaahost.com

Web server: Apache

Speed of internet connection: 1793 KB/sec

In [32], system analysis was defined as a process of troubleshooting or monioring development. The components of system analysis were also defined in [32] as three elements: input, processing, and output. Accordingly, this strategy will be followed in analysing the proposed system and testing its effectiveness.

The hardware configuration and the network configuration used to check the performance of the proposed system where, each test was performed five times to confirm the results and the mean was taken for each test. The proposed framework has been implemented using the same hardware configuration in the previous studies to enable us to compare all solutions.

1. Phase 1: Login System Analysis and Evaluation

The proposed cloud storage system provides high-security access from many perspectives:

1) The RSA algorithm works well because of its difficulty to decipher, which is why it is used to protect user passwords.

Table 4. Login times for the proposed cloud storage.

User ID#	Login Time
1	0.006
2	0.005
3	0.002
4	0.003
5	0.003
Average Login Time	0.0038

The average time taken to enter the system, based on the data in Table 3, was 0.004 milliseconds.

2) Forget password property step (1:3) was as follows:

First, the user is asked to enter their email address to reveal the secret question associated with his/her account. Through system observation, the page load time after clicking on the secret question link was 0.08 milliseconds. Then, the secret question associated with the secret answer appears. After that, the system waits for the user to enter the secret answer, and then checks if the secret answer matches the stored secret answer to allow a password reset; this process took 0.08 milliseconds for the page to load after the user enters the correct answer to their secret question. Finally, the storage system asks the user to enter a new password and confirms it for the purpose of resetting the password, which took 0.05 milliseconds. Actually, in the created storage system, the secret answer is stored in the database in encrypted form by using the Twofish algorithm.

2. Phase 2: Analysis and Evaluation of the Storing and Viewing Processes in the Cloud Storage System for Non-sensitive Files

In testing the proposed framework on non-confidential files, excellent results were obtained for both storage time and file display time due to the immediate storage in the database, as shown in Fig. 6. This is due to the non-sensitive data not being encrypted or fragmented, as shown in Table 5.

Table 5. Storage time (st) and viewing time (vt) in millisecond for the proposed cloud storage for non-sensitive files.gin times for the proposed cloud storage.

File size	Storing Time non-sen file	View Time non-sen file
1 KB	0.01	0.004
2 KB	0.0008	0.006
3 KB	0.1	0.006
4 KB	0.01	0.02
5 KB	0.07	0.01
6 KB	0.09	0.004
7 KB	0.007	0.01
8 KB	0.008	0.008
9 KB	0.01	0.006
10 KB	0.002	0.006
11 KB	0.09	0.008
28 KB	0.02	0.02
78 KB	0.04	0.01
2000 KB	0.1	3.3
4000 KB	0.08	3.8
6000 KB	0.2	4.2
8000 KB	0.1	5
10000 KB	0.2	5.9
12000 KB	0.2	6.5
14000 KB	0.3	9.9

3. Phase 3: Comparison of the Proposed System with Previous Studies

In this section, the performance of the proposed system is compared with a set of earlier studies. In [27], the authors selected files from a relatively small size range, from 1 kB to 11 kB; the proposed storage system was also tested on files of the same size. The encryption and decryption times were much lower compared to those obtained in the Anitha study, where they ranged from 0.01 to 0.06 in terms of encryption time and from 0.01 to 0.02 in terms of decryption time, as shown in Fig.8.

It can also be noted that the times for the improved algorithm in this research for encryption and decryption became shorter due to the proposed binary fragmentation. For example, the AES algorithm's encryption time was approximately 65 milliseconds, while the improved version of the AES algorithm used in this study took 0.04 milliseconds for an 11 kB file. Furthermore, the decryption times with the proposed storage system in this study were 0.02 milliseconds and 52 milliseconds for the non-enhanced AES algorithm in [27].

From an implementation viewpoint regarding the created cloud storage, it is possible to calculate the estimated duration of both storing time and file viewing time for sensitive files using the following equations, Eq. (15) and Eq. (16):

$$storing\_time_{senf} = encr\_time_{senf} + frag\_time_{senf} + storage\_time_{senf} \quad \textbf{(15)}$$

$$Viewing\_time_{senf} = dec\_time_{senf} + frag\_time_{senf} + view\_time_{senf} \quad \textbf{(16)}$$

Storing time and file viewing time for non-sensitive files can be calculated using the following equations, Eq. (17) and Eq. (18):

$$Storing\_time_{non-senf} = storing\_time_{non-senf} \quad (17)$$

$$Viewing\_time_{non-senf} = Viewing\_time_{non-senf} \quad (18)$$

Likewise, the proposed system was compared with [28], which produced a cryptography algorithm resulting from the integration of both randomised privacy and steganography techniques. The encryption and decryption times in [28] were tested on files with the following sizes: 3, 28 and 78 kB (Fig. 7).

From Fig. 7 and [28], it can be noted that 3 KB, 28 KB and 78 KB files took 30, 50 and 96 milliseconds, respectively, to encrypt, while in the proposed system, the storage process (including both encryption and binary fragmentation) of files of the same size took 0.01, 0.01 and 0.02 milliseconds, respectively. As is clear from Fig. 7, the decryption time for a 3 KB, 28 KB and 78 KB file in [28] took 154, 186 and 232 milliseconds, respectively, while in the proposed system, files of the same sizes went through the file viewing process (decryption and defragmentation) in 0.01, 0.02 and 0.02 milliseconds, respectively. This clearly demonstrates the contribution of the proposed method in this study to reducing both storage and file viewing time. Finally, the test of the proposed system vs. secure self-destructing scheme for electronic data (SSDD), an IBE-based secure self-destruction (ISS) scheme and FullPP scheme) as shown in Fig. 9, showed that the proposed framework is more efficient than the previous schemes.

1. Discussion of Phase 1 Results

In this proposed framework, two large prime numbers are used in the RSA algorithm selected, thus ensuring stronger protection of encrypted text (in this case, passwords and user login data). The RSA algorithm, based on a mathematical approach, produces two keys - a public key to encrypt text and private key to decrypt text - where the length of the key can reach 1024 bits [9]. The private key in the implemented RSA algorithm is composed of 904 bytes ≈ 2048 bits. The algorithm will encrypt the user's password with the public key, while the private key is used for decryption purposes. In addition, the forgotten password feature and the secret answer lock used to change forgotten passwords are more secure than in other systems. From the study [14] it was found that the Twofish algorithm has mean encryption performance compared to the other algorithms in the study and it uses less memory; two factors considered in selecting that algorithm to encrypt the questions and answers for resetting the password in this current study.

2. Discussion of Phase 2 Results

In the second stage, the proposed framework was tested on non-confidential files, which do not need the proposed framework to be encrypted and segmented according to the proposed algorithm. Thus, both the storage time and display time of previously stored and non-confidential files will be short due to the direct storage.

3. Discussion of Phase 3 Results

Differences in cloud environments necessarily result in differences in the results for data being read, such as encryption and decryption times, due to the difference in the number of clients for a cloud service, whether it is public or private, among other factors. As a result of the previous factors, the pressure on a cloud site and any congestion in the cloud-supporting network will affect the performance of the cloud. Also, differences in how the data protection algorithms work and the time spent by the user to authenticate login data have a significant impact on the performance of the cloud used for the algorithm.

In the end, the advantages can be seen of the algorithms used to protect the data in [28] and in the framework proposed in this paper. The technique of steganography contributes to making hackers unaware of the existence of information because it is hidden in the image. Also, random encryption leads to difficulty in detecting the encryption technology used to protect the data. All these differences in [27] and [28] lead to different results for the two studies, and the hardware used by each client will also lead to different results. It can be deduced from the random algorithm that there is no guarantee of performance, but rather the performance depends on the encryption algorithms included in the random circuit specified by the algorithm designer. Also, the partitioning of a file into parts will, in all cases, contribute to improved performance of the cloud site as a result of the reduction in load on the processors.

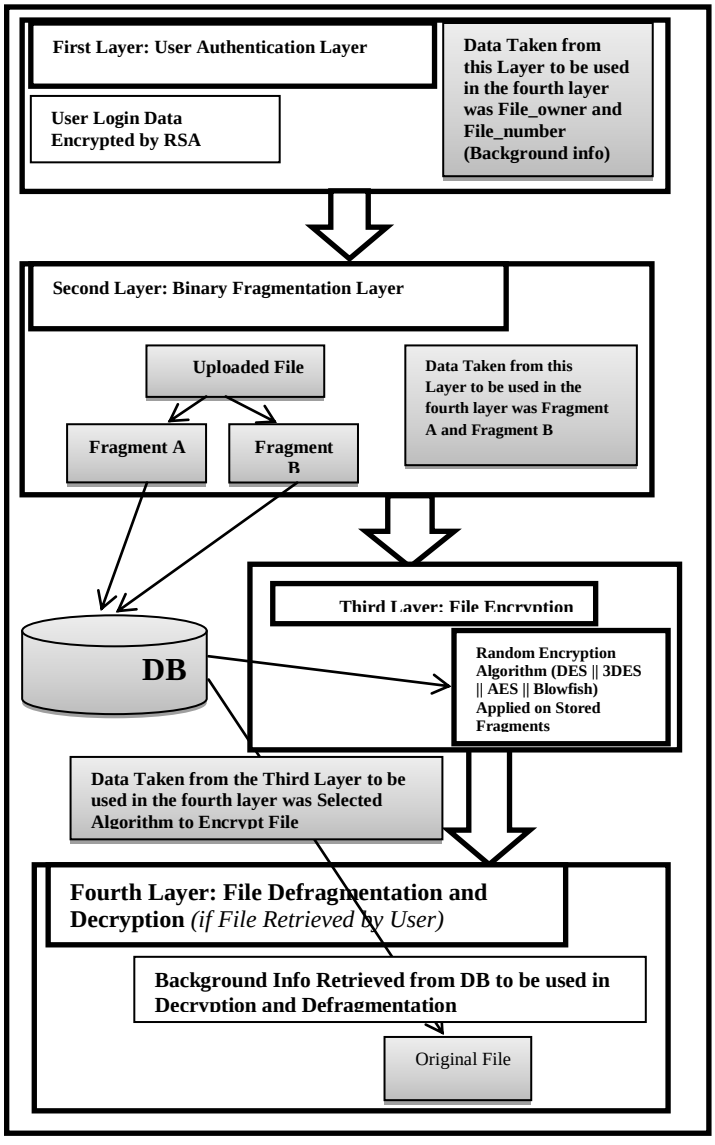
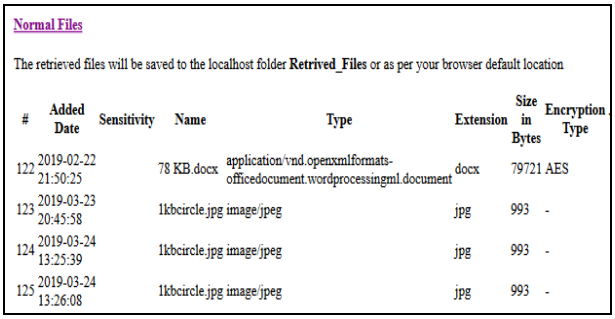


Fig. 4. Detailed architecture of the proposed framework.

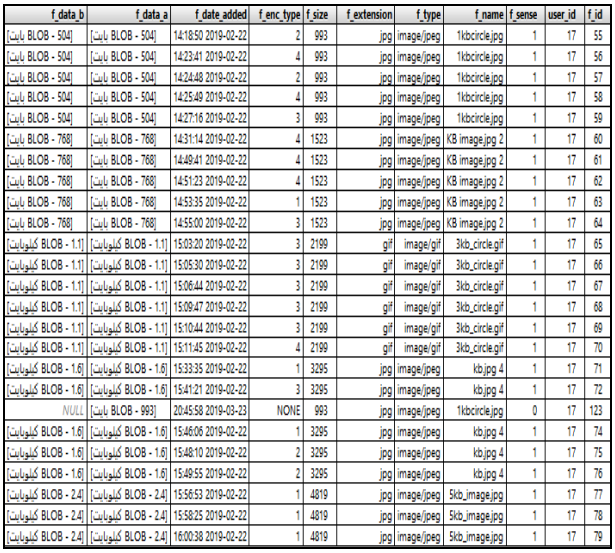


Normal Files

The retrieved files will be saved to the localhost folder **Retrived_Files** or as per your browser default location

#	Added Date	Sensitivity	Name	Type	Extension	Size in Bytes	Encryption Type
122	2019-02-22 21:50:25		78 KB.docx	application/vnd.openxmlformats-officedocument.wordprocessingml.document	docx	79721	AES
123	2019-03-23 20:45:58		1kbcircle.jpg	image/jpeg	jpg	993	-
124	2019-03-24 13:25:39		1kbcircle.jpg	image/jpeg	jpg	993	-
125	2019-03-24 13:26:08		1kbcircle.jpg	image/jpeg	jpg	993	-

(a)



f_data_b	f_data_a	f_date_added	f_enc_type	f_size	f_extension	f_type	f_name	f_sense	user_id	f_id
[بايت BLOB - 504]	[بايت BLOB - 504]	14:18:50 2019-02-22	2	993	jpg	image/jpeg	1kbcircle.jpg	1	17	55
[بايت BLOB - 504]	[بايت BLOB - 504]	14:23:41 2019-02-22	4	993	jpg	image/jpeg	1kbcircle.jpg	1	17	56
[بايت BLOB - 504]	[بايت BLOB - 504]	14:24:48 2019-02-22	2	993	jpg	image/jpeg	1kbcircle.jpg	1	17	57
[بايت BLOB - 504]	[بايت BLOB - 504]	14:25:49 2019-02-22	4	993	jpg	image/jpeg	1kbcircle.jpg	1	17	58
[بايت BLOB - 504]	[بايت BLOB - 504]	14:27:16 2019-02-22	3	993	jpg	image/jpeg	1kbcircle.jpg	1	17	59
[بايت BLOB - 768]	[بايت BLOB - 768]	14:31:14 2019-02-22	4	1523	jpg	image/jpeg	KB image.jpg 2	1	17	60
[بايت BLOB - 768]	[بايت BLOB - 768]	14:49:41 2019-02-22	4	1523	jpg	image/jpeg	KB image.jpg 2	1	17	61
[بايت BLOB - 768]	[بايت BLOB - 768]	14:51:23 2019-02-22	4	1523	jpg	image/jpeg	KB image.jpg 2	1	17	62
[بايت BLOB - 768]	[بايت BLOB - 768]	14:53:35 2019-02-22	1	1523	jpg	image/jpeg	KB image.jpg 2	1	17	63
[بايت BLOB - 768]	[بايت BLOB - 768]	14:55:00 2019-02-22	3	1523	jpg	image/jpeg	KB image.jpg 2	1	17	64
[كيلوبايت BLOB - 1.1]	[كيلوبايت BLOB - 1.1]	15:03:20 2019-02-22	3	2199	gif	image/gif	3kb_circle.gif	1	17	65
[كيلوبايت BLOB - 1.1]	[كيلوبايت BLOB - 1.1]	15:05:30 2019-02-22	3	2199	gif	image/gif	3kb_circle.gif	1	17	66
[كيلوبايت BLOB - 1.1]	[كيلوبايت BLOB - 1.1]	15:06:44 2019-02-22	3	2199	gif	image/gif	3kb_circle.gif	1	17	67
[كيلوبايت BLOB - 1.1]	[كيلوبايت BLOB - 1.1]	15:09:47 2019-02-22	3	2199	gif	image/gif	3kb_circle.gif	1	17	68
[كيلوبايت BLOB - 1.1]	[كيلوبايت BLOB - 1.1]	15:10:44 2019-02-22	3	2199	gif	image/gif	3kb_circle.gif	1	17	69
[كيلوبايت BLOB - 1.1]	[كيلوبايت BLOB - 1.1]	15:11:45 2019-02-22	4	2199	gif	image/gif	3kb_circle.gif	1	17	70
[كيلوبايت BLOB - 1.6]	[كيلوبايت BLOB - 1.6]	15:33:35 2019-02-22	1	3295	jpg	image/jpeg	kb.jpg 4	1	17	71
[كيلوبايت BLOB - 1.6]	[كيلوبايت BLOB - 1.6]	15:41:21 2019-02-22	3	3295	jpg	image/jpeg	kb.jpg 4	1	17	72
NULL	[بايت BLOB - 993]	20:45:58 2019-03-23	NONE	993	jpg	image/jpeg	1kbcircle.jpg	0	17	123
[كيلوبايت BLOB - 1.6]	[كيلوبايت BLOB - 1.6]	15:46:06 2019-02-22	1	3295	jpg	image/jpeg	kb.jpg 4	1	17	74
[كيلوبايت BLOB - 1.6]	[كيلوبايت BLOB - 1.6]	15:48:10 2019-02-22	2	3295	jpg	image/jpeg	kb.jpg 4	1	17	75
[كيلوبايت BLOB - 1.6]	[كيلوبايت BLOB - 1.6]	15:49:55 2019-02-22	2	3295	jpg	image/jpeg	kb.jpg 4	1	17	76
[كيلوبايت BLOB - 2.4]	[كيلوبايت BLOB - 2.4]	15:56:53 2019-02-22	1	4819	jpg	image/jpeg	5kb_image.jpg	1	17	77
[كيلوبايت BLOB - 2.4]	[كيلوبايت BLOB - 2.4]	15:58:25 2019-02-22	1	4819	jpg	image/jpeg	5kb_image.jpg	1	17	78
[كيلوبايت BLOB - 2.4]	[كيلوبايت BLOB - 2.4]	16:00:38 2019-02-22	1	4819	jpg	image/jpeg	5kb_image.jpg	1	17	79

(b)

Fig. 5. Screenshots of uploaded files (fragmented and encrypted): (a) Appearance of stored files in user interface, (b) Appearance of stored files in framework database.

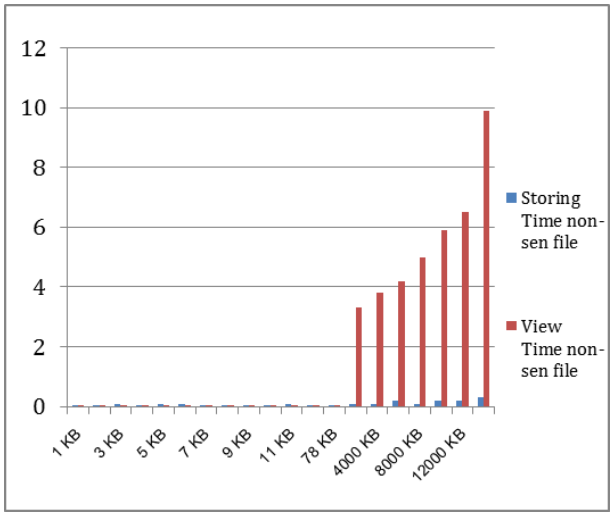


Fig. 6. Storing time (ST) and viewing time (VT) for the proposed cloud storage for non-sensitive files.

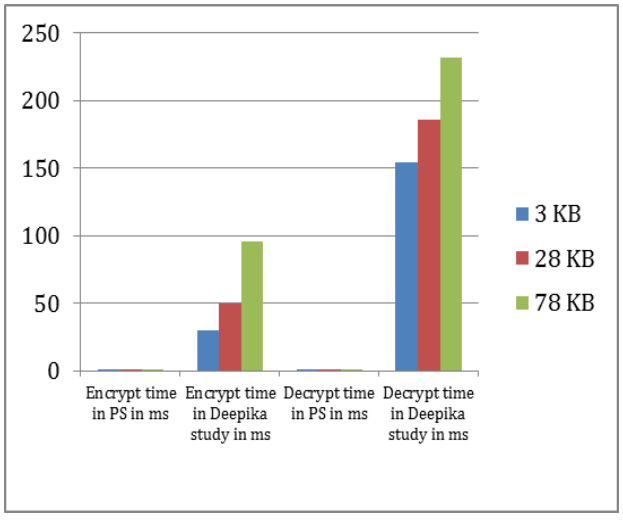


Fig. 7. Encryption time and decryption time in Deepika study vs. new proposed system.

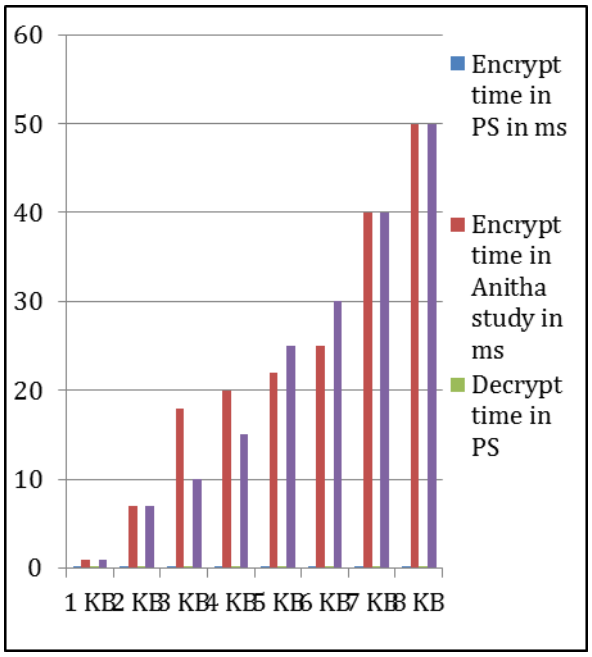


Fig. 8. Encryption time and decryption time in Anitha study vs. new proposed system.

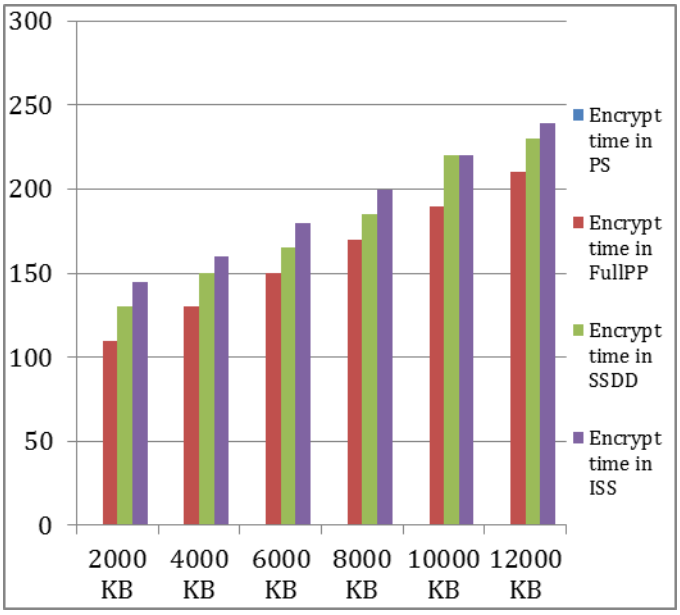


Fig. 9. Encryption time in the proposed system vs. FullPP, SSDD and ISS.

6. Conclusion

This paper proposes a framework that includes a new strategy for file fragmentation within cloud computing repositories and a random encryption algorithm to protect confidential files. The binary fragmentation technique improves performance by up to 99%, compared to previous studies, in two aspects: the storage time and the viewing time of confidential files. The strength of the proposed random encryption algorithm depends on the strength of the selected algorithms in terms of how they work, and the strength of the encryption key used. All of the features included in the framework to protect confidential data, such as using security questions and sharing secure files, greatly contribute to the security of cloud systems by providing a barrier to unauthorised persons trying to access the system data.

There is still a need to find techniques and frameworks to avoid changes in stored data as well as find practical solutions to recover from attacks in the case of occurrence. Initially, this is done by building detailed models of the behaviours of the natural system and behaviours that can lead to the destruction of a cloud system as a result of attacks.

Author Contributions

Both authors contributed to the general idea of this manuscript, including the general subject of the paper, writing goals and the manuscript method. Mariam Alrashidi proposed a random encryption algorithm and its integration with binary fragmentation to build the security framework and conduct experiments. Dr. Maher Khemakhem reviewed and supervised the work of the manuscript after being written by Mariam Alrashidi and provided the necessary advice and recommendations.

Acknowledgment

The authors wish to thank Dr. Fathi Al-borai, who was supportive of this research through suggestions and scientific advice.

Conflicts of Interest: The authors declare no conflict of interest.

References

[1] **Garg, G., Sabharwal, S.** and **Jain, A.** (2016) "Basics of cloud computing," *International Research Journal of Engineering and Technology*, **3** (July.): 1514-1517.

[2] **Khafaja, A.** (2010) "*Cloud computing and its Applications in Libraries*" [Online]: Available: http://www.journal.cybrarians.org/index.php?option=com_content&view=article&id=445:2011-08-10-01-36-53&catid=158:2009-05-20-09-59-42, June 2010 [Accessed: 30-Dec-2017].

[3] **Al Mark, A., al-Shammari, A.,** *et al*. (2017) "Security in Cloud Computing," [Online] Available: https://cloudcomputingksu.wordpress.com, May 2012 [Accessed: 28-Dec-2017].

[4] **Shrivastava, A.** and **Singh, L.** (2016). A new hybrid encryption and steganography technique: a survey. *International Journal of Advanced Technology and Engineering Exploration*, **3**(14): 9–14. http://doi.org/10.19101/IJATEE.2016.314005.

[5] **Ali, H.** (2017) "The Arabs first used encrypted messages and called it "the art of blindness", *Raseef 22 Journal*, 2017. [Online]. Available: https://raseef22.com/life/2017/01/27/. [Accessed: 16- Dec- 2017].

[6] **Dave, P.** (2017) "*SQL SERVER - Fragmentation - Detect Fragmentation and Eliminate Fragmentation*," Journey to SQL Authority with Pinal Dave, 21-Mar-2016. [Online]. Available: https://blog.sqlauthority.com/2010/01/12/sql-server-fragmentation-detect-fragmentation-and-eliminate-fragmentation/. [Accessed: 03-Jun-2017].

[7] "*Notes – SQL Server Index Fragmentation, Types and Solutions*", Pankaj Mittal [MSFT], 2017. [Online]. Available: https://blogs.msdn.microsoft.com/pamitt/2010/12/23/notes-sql-server-index-fragmentation-types-and-solutions/. [Accessed: 03- Jun- 2017].

[8] **Verma, S. K.** (2016). Fragmentation techniques for distribution database: a review. *International Journal of Innovative Computer Science & Engineering*, **3**(2), 47–50.

[9] **Meneses, F., Fuertes, W., Sancho, J., Salvador, S., Flores, D., Aules, H.** and **Nuela, D.** (2016). RSA encryption algorithm optimization to improve performance and security level of network messages. *International Journal of Computer Science and Network Security (IJCSNS)*, **16**(8): 55-62.

[10] **Pancholi, V. R.** and **Patel, B. P.** (2016) "Enhancement of cloud computing security with secure data storage using AES*," IJIRST-International Journal for Innovative Research in Science & Technology*, **2**: 18-21.

[11] **Thirumalai, C.** (2016) "Review on the Memory Efficient RSA Variants," *International Journal of Pharmacy & Technology*, **8**(4) Dec.: 4907-4916.

[12] **Parashar, A.** (2016) "Services & Security of Cloud Computing Using DES," *International Journal of Computer & Mathematical Sciences IJCMS*, **5**: 41–45.

[13] **Sahu, R.** and **Ansari, M. S.** (2017). Securing messages from brute force attack by combined approach of honey encryption and Blowfish. *International Research Journal of Engineering and Technology (IRJET)*, **4**(9): 1019-1023.

[14] **Mewada, S., Sharma, P.** and **Gautam, S.** (2016) "Classification of Efficient Symmetric Key Cryptography Algorithms," *International Journal of Computer Science and Information Security*, **14**(2) Feb.: 105-110.

[15] **Aparna, K., Jyothy Solomon, Harini** and **Indhumathi, M.** (2016) "A study of Twofish Algorithm," *International Journal of Engineering Development and Research*, **4**(2) Nov.: 148-150.

[16] **Basri, M., Mawengkang, H.** and **Zamzami, E.** (2018) "Cloud Computing Security Model with Combination of Data Encryption Standard Algorithm (DES) and Least Significant Bit (LSB)," in: *Journal of Physics: Conference Series: Conf. Series* **970**: 1–7.

[17] **Adhie, R., Hutama, Y., Ahmar, A.** and **Setiawan, M.** (2019) "Implementation Cryptography Data Encryption Standard (DES) and Triple Data Encryption Standard (3DES) Method in Communication System Based Near Field Communication (NFC)," in: *Proc. of Journal of Physics: Conference Series: Conf. Series 954, IOP 2016, 3-4 October 2016, Makassar, Indonesia* [Online]. Available: https://iopscience.iop.org/article/10.1088/1742-6596/954/1/012009/pdf. [Accessed: 10 Feb. 2019].

[18] **Dusane, P., Patil, J., Jain, U.** and **Pandya, R.** (2017) "Security of Data with RGB Color and AES Encryption Techniques," *International Research Journal of Engineering and Technology (IRJET)*, **4**(4) Apr.: 3063–3067.

[19] **Gupta, A.** and **Kaushik, S.** (2017) "A Review: RSA and AES Algorithm," *IITM Journal of Management and IT*, **8**(4) January-June: 82-85.

[20] **Gupta, U., Saluja, S.** and **Tiwari, T.** (2018) "Enhancement of Cloud Security and Removal of Anti-patterns Using Multilevel Encryption Algorithms," *International Journal of Recent Research Aspects*, **5**(1) Mar.: pp. 55-61.

[21] **Rayan, A. M., Abdel-Hafez, A. A.** and **Hafez, I. M.** (2016) "Provably Secure Encryption Algorithm based on Feistel Structure," *International Journal of Computer Applications*, **139**, no. 1: 1–8.

[22] **Kaur, S.** and **Singh, A.** (2017) "Security issues in cloud computing", *International Education & Research Journal (IER)*, **3**(5) May: 118–119.

[23] **Bollavarapu, S.** and **Gupta, B.** (2014) "Data security in cloud computing," *International Journal of Advanced Research in Computer Science and Software Engineering*, **4**(3) March: 1208-1215.

[24] **Mushtaq, M.**, *et al*. (2016) "An efficient framework for information security in cloud computing using auditing algorithm shell (AAS)," *International Journal of Computer Science and Information Security (IJCSIS)*, **14**, (11): 317-331.

[25] **Shinde, M.** and **Taur, R.** (2015) "Encryption algorithm for data security and privacy in Cloud Storage," *American Journal of Computer Science and Engineering Science*, **3**(1): 34-39.

[26] **Vyas, J.** and **Modi, P.** (2017). Providing confidentiality and integrity on data stored in cloud storage by hash and meta-data approach. *International Journal of Advance Research in Engineering, Science & Technology*, **4**(5): 2394-2444.

[27] **Anitha, R.** and **Vijayakumar, V.** (2015) "Holomorphic encryption-based data security on federated cloud computing," *ARPN Journal of Engineering and Applied Sciences*, **10**(March.): 1975–1979.

[28] **Deepika** (2017) "Enhancement of data security for cloud environment using cryptography and steganography technique," *International Journal of Innovative Research in Computer and Communication Engineering*, **5**: 225-230.

[29] **Yi, H., Li, J., Lin, Q.,** *et al*. (2019) "A Rainbow-Based Authentical Scheme for Securing Smart Connected Health Systems," *J Med Syst (United States)*, **43**(8) Jul 06: 276.

[30] **Jiang, Y., Hamer, J., Wang, C.,** *et al*. (2019) "SecureLR: Secure Logistic Regression Model *via* a Hybrid Cryptographic Protocol," *IEEE/ACM Trans Comput Biol Bioinform (United States)*, **16**(1) Jan-Feb: 113-123,.

[31] "*PHP Tutorial*", www.tutorialspoint.com, 2017. [Online]. Available: https://www.tutorialspoint.com/php/. [Accessed: 05- Sep- 2017].

[32] **Shweimi, N.** (2018) "*Introduction to Systems Analysis," Egyptian researchers, Apr-2017*. [Online]. Available: http://www.egyres.com/articles/. [Accessed: 02-Jan-2018].

إطار عمل وخوارزمية التشفير لحماية البيانات الحساسة على موفري الخدمات السحابية

مريم عبيد مسلم الرشيدي و **ماهر خماخم**

كلية الحاسبات وتقنية المعلومات، جامعة الملك عبدالعزيز، جدة، المملكة العربية السعودية
mariamuoh23@yahoo.com

المستخلص. في هذه الورقة، تم تصميم إطار أمني لحماية البيانات السرية في بيئات التخزين السحابية. يتضمن ذلك الإطار تقنية التجزئة الثنائية على مستوى الملفات السرية وخوارزمية أطلقنا عليها خوارزمية التشفير العشوائي، لأنها بنيت على فكرة عشوائية تشفير الملف السري بين أربعة أنواع من خوارزميات التشفير. في هذا الإطار تم تصنيف الملفات المخزنة في السحابة إلى نوعين: ملفات سرية وملفات ليست سرية. تقنية التجزئة الثنائية المقترحة تعمل على مستوى الملفات السرية، على عكس ما هو شائع في حالات تقنيات التجزئة الأخرى، والتي تعمل على مستوى قاعدة البيانات، مثل التجزئة الأفقية، والتجزئة الرأسية، والتجزئة الهجين. مفهوم التجزئة الثنائية، ينص على أن يتم تجزئة الملف السري إلى جزئين أحدهما يرمز له بالرمز (أ) والآخر بالرمز (ب) ثم يتم تشفير كلا الجزئين على حدة باستخدام خوارزمية التشفير العشوائي المقترحة وتخزينهما في أماكن منفصلة داخل قاعدة البيانات. علاوة على ذلك، يتضمن الإطار الأمني المقترح بوابة مصادقة المستخدم التي تشفر بيانات تسجيل المستخدم من خلال خوارزمية التشفير الشهيرة RSA. بالإضافة إلى خوارزمية Twofish المستخدمة لجعل إعادة تعيين كلمة المرور أكثر أمنا باستخدام سؤال سري وأجابة سرية، واللتان يحفظان بشكل مشفر في قاعدة البيانات. كانت نتائج اختبار الإطار الأمني إيجابية، حيث أسهمت في تقليل وقت التشفير ووقت فك التشفير للملفات السرية بنسبة ٩٩٪ مقارنة بالدراسات السابقة. نتيجة للجمع بين كل تلك الخوارزميات والتقنيات في إطار واحد، سيزيد المستوى الأمني لنظام التخزين السحابي. وأخيرًا، نوصي من يسعى لنظام تخزين سحابي آمن أن يأخذ بعين الأعتبار جميع الجوانب الأمنية، ونذكر منها أمن البيانات، وسلامة البيانات، وتبادل الملفات بشكل آمن، وأمن التحكم في الوصول، ومن ذلك تحديد صلاحيات المسؤول عن السحابة وصلاحيات المستخدم ... الخ. أضف إلى ذلك، ينبغي أن يكون هناك نظام للكشف عن أي خطر قد يتعرض له النظام واتخاذ التدابير اللازمة لتجنب الخطر أو التقليل منه.

الكلمات المفتاحية: أمن البيانات، الحوسبة السحابية، التجزئة، خصوصية المستخدم، خوارزميات التشفير.