HYBRID ENCODING AND DATA TRANSFORMATION FOR CLOUD DATA SECURITY IN POST-QUANTUM ERA

SHAIK MOHAMMAD ILIAS[1], V.CERONMANI SHARMILA[2] †

Research Scholar, Dept. of Computer Science and Engineering, Hindustan Institute of Technology and Science, Chennai, India. 603103 India
Head of Department, Department of Information Technology, Hindustan Institute of Technology and Science Chennai, India. 603103, India

E-mail: illusoft54@gmail.com, † csharmila@hindustanuniv.ac.in.

ABSTRACT

In the wake of emergence of quantum computers, there have been efforts on Post Quantum Cryptography (PQC) schemes. PQC schemes can replace the classical cryptographic primitives for stronger level of security. Demand for such schemes is driven by initiatives and interests from security standardization bodies across the world. In response to the demand, many schemes came into existence. Motivated by the PQC requirements, in this paper, we proposed a security algorithm known as Hybrid Encoding and Data Transformation (HEDT). It has provision for data owners to have more secure outsourcing of their data to cloud and retrieval of the same. It has multiple data transformations with data encoding and decoding to have stronger level of security. It not only provides security to outsourced data but also fulfils needs of features such as data integrity and data availability. It is designed to be a candidate for PQC requirements to safeguard not only data at rest in cloud but also data in transit. An empirical study revealed that the HEDT is capable of providing stronger level of security to data outsourced to cloud.

Keywords –*Encryption, Decryption, Encoding, Decoding, Cloud Data Security, Cloud Computing*

1. INTRODUCTION

Data has become very important for organizations. In fact, it is considered asset as it drives businesses. Traditionally data has been stored in different formats like flat files, databases, relational databases and object relational databases. When data is stored in trusted storage media in local computing devices or local servers, there is protection for the data from unauthorized access. However, with the emergence of cloud computing, the way data is stored is changed. Organizations are outsourcing data to cloud storage instead of making local copy of it. When data is outsourced without having a local copy, security is essential for such outsourced data. As the Cloud Service Provider (CSP) claims secure cloud storage, data owners might depend on it. However, it is not ideal approach to depend on CSP as the cloud is accessed through Internet and considered untrusted area from users' point of view.

Literature review has revealed many contributions towards cloud data security. AES is found to be widely used encryption standard used in cloud platforms as explored in [11], [12], [13], [15]. Information dispersal theory is also found to have wide usage in information systems. It is explored in [1]- [10] and used for security purposes. Hashing is another security approach used in different applications. It is explored in [23] and [25] for security and data integrity. From the literature, it is understood that security or cryptographic primitives have greaterutility in information security. However, in the emergence of quantum computers, there is effort to have Post Quantum Cryptography (PQC) requirements to be fulfilled in future endeavours of security primitives. With cloud computing in place and its affordable business model, many organizations in the real world are attracted towards cloud storage. However, it is to be understood that cloud is access through Internet and it is an untrusted place by default. Unless there is reliable security for outsourced data, there is no guarantee for security and privacy of outsourced data. When the plain data is outsourced, it is completely relying on the security primitives of Cloud Service Provider (CSP). This is not an ideal situation to depend on CSP as there are many instances in the history where data owner lost data due to inherent security issues in cloud infrastructure or internal threats of external threats. In order to overcome this problem,in this paper, we proposed a comprehensive security scheme for data in cloud. Our contributions are as follows.

1. We proposed a PQC driven security algorithm known as Hybrid Encoding and Data Transformation (HEDT) which supports secure data outsourcing and secure data retrieval.
2. A prototype application is built to evaluate the HEDT and underlying mechanisms for secure outsourcing of data to cloud and secure data retrieval.

The remainder of the paper is structured as follows. Section 2 reviews literature on different security schemes and hybrid procedures for stronger level of security. Section 3 presents the proposed algorithm for cloud data security. Section 4 presents experimental results while Section 5 concludes the paper and gives directions for future work.

2. RELATED WORK

This section reviews literature on various aspects of data security. It throws light on IDA approaches, AES improvements and hashing to provide useful insights for further research.

2.1 Information Dispersal Theory

Information dispersal theory is widely used for data security. Wijayanto and Harjito [1] opined that IDA can be used to safely store files. They proposed a methodology to minimize rounding off errors with respect to IDA. Nadendla*et al*. [2] redesigned IDA algorithm of Rabin to be suitable for Cognitive Radio (CR) networks which ensures confidentiality and data integrity in such networks. Mar *et al*. [3] proposed a cloud data security scheme based on IDA. They explored the problem of information leakage and prevention with their proposed method. Baldi*et al*. [4] proposed a unified framework that is used to evaluate IDA with model checking for ascertaining confidentiality. Deryabin*et al*. [5] made a review of various IDA methods available in connection with distributed storage applications. Yand and Lu [6] investigated on IDA with “learning with errors (Ring-LWE)” and “number theoretic transform (NTT)” in order to ascertain its utility in secure storage. MARCELÍN-JIMÉNEZ*et al*. [7] proposed a methodology to ascertain the complexity of IDA and its role in fault tolerance systems. Singh and Sarbjeeth Singh [8] proposed a security system for cloud based on IDA. It is used with erasures coding in order to have reliable storage in cloud. Qian *et al*. [9] proposed an encryption algorithm that makes use of IDA with multiple layers with stronger level of security. Shima and Doi [10] used IDA to develop a secret sharing scheme that is hierarchical in nature. Its implementation is made to achieve information security.

2.2 AES Based Security

AES is found to be widely used security protocol in real world applications. Let *et al*. [11] explored it for cloud data security with HEROKU as cloud platform. They experimented with data security, delay and strength of security. Panda [12] used AES for securing communications in Wireless Sensor Network (WSN). Considering energy constrained devices, they implemented security with AES. Fathurrahmad and Ester [13] explored AES along with Rijndael algorithm for securing web data. They found the hybrid to improve level of security. Kumar and Rana [14] proposed a modified AES by increasing number of rounds for improving strength of security over untrusted channels. Dang and Vo [15] proposed an enhanced AES algorithm in order to protect Internet of Things (IoT) use case with the provision of dynamic key. Akhil*et al*. [16] employed AES algorithm to improve security in cloud computing scenario. Kumar *et al*. [17] explored AES for the implementation of “Field Programmable Gate Arrays (FPGA)” devices. Katkade and Phade [18] implemented AES for serial communication with security in FPGA environments. Zhang and Qunding [19] used AES for securing digital images in order to prevent attacks. Yu *et al*. [20] considered Scan attack and improved AES design to have reliable information security.

2.3 Hashing in Information Security

Hashing techniques are widely used for information security applications. Chen *et al*. [21] explored the context of big data security and evolution of security primitives. Rawat and Agrawal [22] proposed a hash algorithm with message digest for securing information in different applications. Botacin*et al*. [23] explored similarity hashing functions that are used in real world applications. They studied their advantages and disadvantages in the context of malware detection research. Wu and Horng [24] used hash functions to have secure communications in vehicular cloud networks. Topcu*et al*. [25] focused on biometric hashing and the possible privacy and data security attacks on such hashing in image processing applications. Timothy *et al*. [26] used secure hashing technique as part of their proposed hybrid cryptography technique for cloud data security. Maitri and Verma

[27] focused on secure file storage with combination of methods such as "AES, blowfish, RC6 and BRA". Chinnasamy and Deepalakshmi [28] combined hashing and cryptographic primitives to have a hybrid security method for healthcare application in cloud. Ahmad Garko [29] discussed about many hybrid security schemes used for cloud data security. Feng *et al*. [30] proposed a hybrid security scheme using hashing and AES and RSA for securing data. From the literature it is ascertained that, it is important to have hybrid approaches for security of cloud data. Keeping PQC requirements in mind, in this paper we proposed a security algorithm that combines multiple approaches to have more reliable data security.

3. PROPOSED SYSTEM

With cloud computing in place and its affordable business model, many organizations in the real world are attracted towards cloud storage. However, it is to be understood that cloud is access through Internet and it is an untrusted place by default. Unless there is reliable security for outsourced data, there is no guarantee for security and privacy of outsourced data. When the plain data is outsourced, it is completely relying on the security primitives of Cloud Service Provider (CSP). This is not an ideal situation to depend on CSP as there are many instances in the history where data owner lost data due to inherent security issues in cloud infrastructure or internal threats of external threats. In order to overcome this problem, many security frameworks came into existence. Such frameworks are used by data owners for secure outsourcing of data. However, the existing security frameworks need to be strengthened to meet PQC scenarios.

3.2 Hybrid Encoding and Data Transformation Scheme

The HEDT scheme is proposed and implemented in this paper with empirical study. The framework is cloud based and can store different kinds of data. Cloud infrastructure supports storage of structured, semi-structured and unstructured data. The HEDT scheme is illustrated in Figure 1.

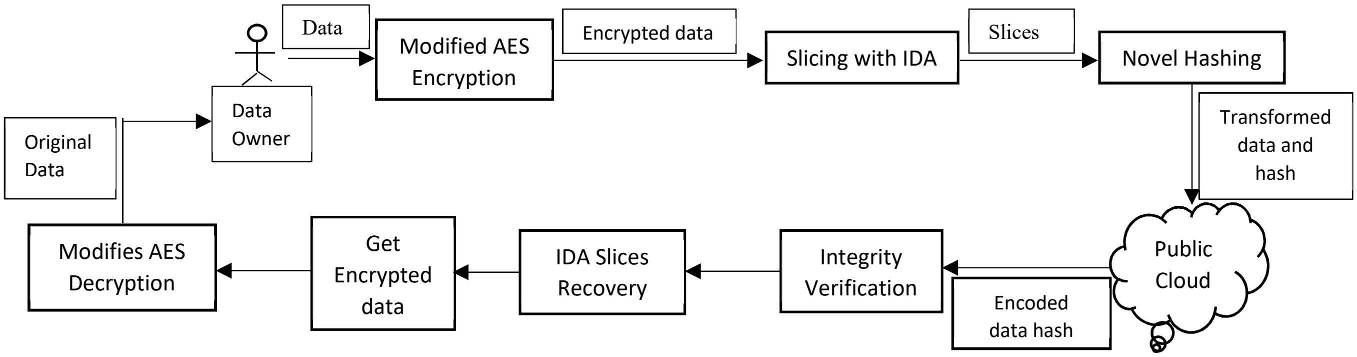


Figure 1: Proposed Hybrid Encoding and Data Transformation (HEDT) scheme

The HEDT scheme is realized with many components such as modified Advanced Encryption Standard (AES), Information Dispersal Algorithm (IDA) and novel hashing mechanism. With these components there is hybrid encoding and multiple data transformation. The outsourcing process involves many different transformations to be compatible with PQC requirements. The data of data owner is subjected to modified AES encryption. This process generates encrypted data or cipher text. The cipher text is further subjected to IDA which breaks a given file F into number of slices. These prices have their utility in reconstruction of F even when some pieces are lost for any reason. IDA has plenty of advantages such as transmission efficiency, fault-tolerance, security and reliability. Once the slices are made, they are subjected to novel hashing mechanism which helps in efficient data integrity verification. After application of hashing, the data is outsourced to public cloud along with hash value.

The decoding process is the reverse process of encoding. When data owner needs to access data from public cloud, he makes a request and gets the data is a secure and reliable fashion. First, encoded data is taken from public cloud and it is subjected to integrity verification. Afterwards, the hashing process is applied IDA slices are recovered. Once slices are recovered, the encrypted file F is reconstructed. Then the file is subjected to the modified AES decryption to arrive at the original data. The entire process is user-friendly and transparent to the data owner.

3.3 Modified AES

AES is NIST recommended standard for data security. It has support for 128-bit data length and 128/192/256-bit key length. It needs 10/12/14 rounds of computations for 128/192/256-bit key length respectively. As explored in [25], AES operations need AES S-Box (Substitution-Box) that is a matrix of Hex values which is used as lookup table. The S-Box generation process is expressed in Eq. 1.

GF (28) = GF (2) [x]/(x8+x4+x3+x+1) (1)

The multiplicative inverse is transformed using affine transformation which involves an XOR operation. Each round in AES has 4 different transformations and the flow is as illustrated in Figure 2. First, each byte of data block is transformed into block using S-Box. Second, the rows in state matrix is left shifted as per row position. Third, columns in state matrix are multiplied with columns in fixed matrix. Fourth, XOR operation takes place between round key matrix and new state matrix.

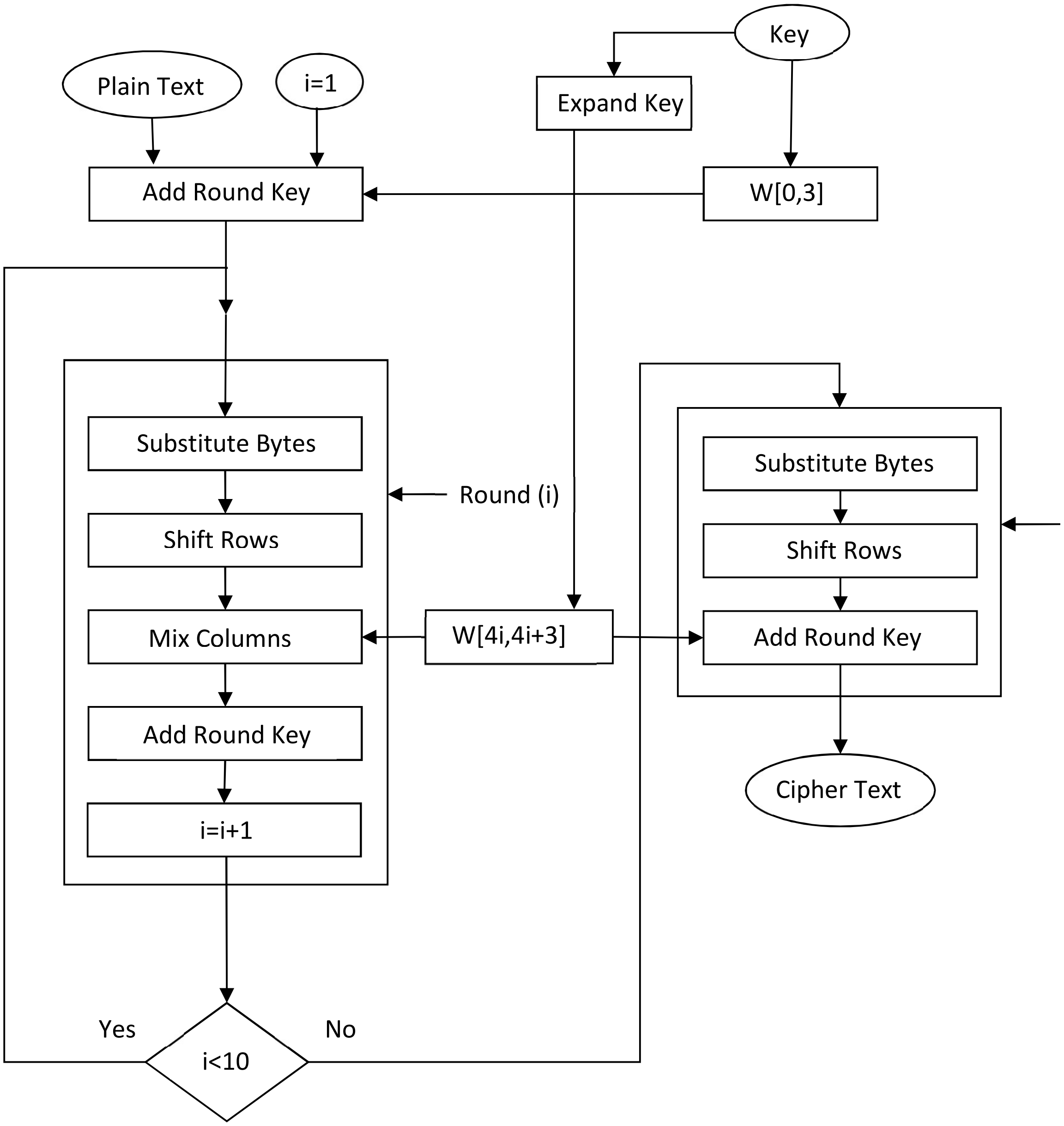


Figure 2: Functional overview of AES

There are certain drawbacks in the traditional AES. First, cyber-attacks are growing continuously to make the security professionals alert and strive to define new schemes as explored in [26]. Second, AES may be broken with different kinds of attacks such as linear attack, algebraic attack and differential attack as studied in [27], [29] and [29]. Third, the breaking of the algorithm leads to interception, impersonating and stealing sensitive information. In order to overcome these drawbacks, D'souza and Panchal [1] introduced certain enhancements to AES. The enhancements include dynamic key generation and dynamic S-Box generation. The function of time is used to generate dynamic keys. The static S-Box is transformed into dynamic S-Box for better security. The process of making dynamic S-Box is shown in Figure 3.

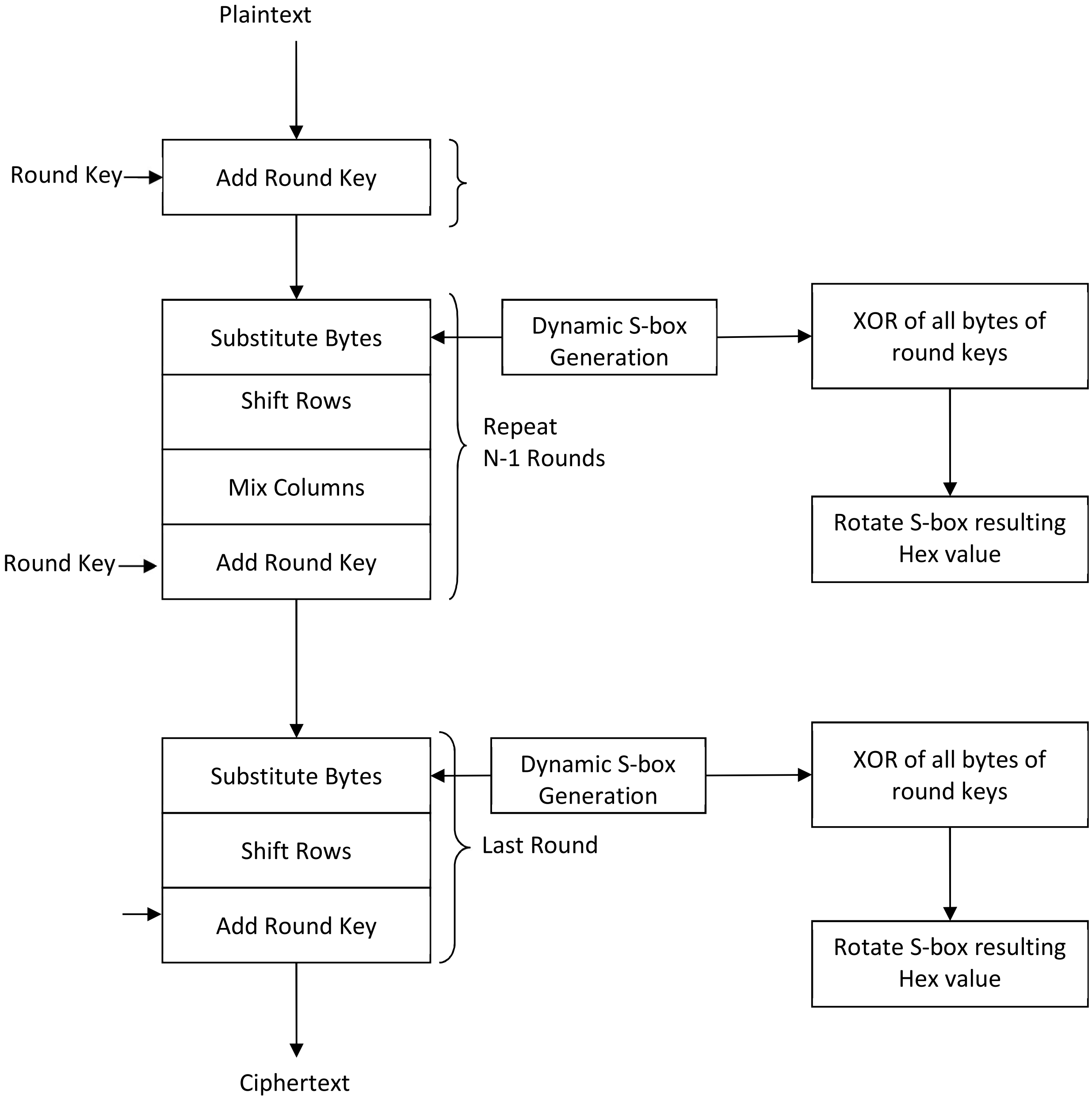


Figure 3: Flowchart of dynamic S-box generation.

In the post-quantum era, this enhanced AES is also needs to be combined with other transformations further for stronger cryptographic security. In this paper, we use Information Dispersal Algorithm (IDA) and hashing to have a novel cryptographic scheme.

3.4 Information Dispersal Algorithm

This algorithm is meant for efficient information dispersal. Let the data in a file is denoted as F = $b_1, b_2, \ldots\ldots, b_N$. Dispersal of F to cloud or any other storage medium is assumed and it is expected to lose less than k pieces due to node failure or communication path issues. The b_i may be set of characters (bytes) that satisfy 0≤ b_i ≤255. When data is in bytes p=257 is sufficient and prime is taken such that B<p. For a string of elements in the finite field Z_p, the computations are made in Z_p that is with mod p. An integer m is chosen such that m = m + k satisfies n/m ≤ 1 + ∈ for a specified ∈> o. A set of vectors denoted as n is chosen as a_i = { a_{i1}, …., a_{im}} ∈ Z_p^m,1 ≤ i ≤n in such a way that there is linear dependency among m vectors in {a_1, …., a_n}. The given data file F is divided into number of sequences of length m.

F= (b_1, …., b_m), (b_{m+1}, …., b_{2m}), ….

denote S_1 =(b_1, …., b_m), etc. For i = 1, ... , n,

F_i= c_{i1}, c_{i2}, …., $c_{iN/m}$

where

$$c_{ik}= a_i . S_k= a_{i1} . b_{(k-1)m+1} + \cdots + a_{i.m} . b_{k.m} \qquad (2)$$

Let A = $(a_{ij})_{1\le i,j\le m}$be the m x m matrix whose i-th row is a_i. It is readily seen that

$$A.\begin{bmatrix} b_1 \\ . \\ b_m \end{bmatrix}=\begin{bmatrix} c_{11} \\ . \\ c_{m1} \end{bmatrix}$$

and hence

$$\begin{bmatrix} b_1 \\ . \\ b_m \end{bmatrix} = A^{-1}. \begin{bmatrix} c_{11} \\ . \\ c_{m1} \end{bmatrix}$$

Denote the i-th row of $A^{-1}$by ($\alpha_{i1}, \ldots., \alpha_{im}$), then in general, for 1 ≤ k ≤ N/m,

$$b_j= \alpha_{i1}c_{1k}+\ldots.+\alpha_{im}c_{mk},\ 1\le j \le N, \qquad (2)$$

where i = j mod m, k = $\lceil j/m \rceil$ (here we take the residues to be 1, ... , m).

Thus A is inverted and F is reconstructed involving operations (2m mod p) on each character of F. In case of large files that satisfy $m^2 \le$ |F| the operation cost is increased due to the cost of reconstruction cost. Instead of Z_p, it is possible to make use of other fields. For instance, field E = GF(2^8) with 256 elements and characteristic 2. In such as there is need for an irreducible polynomial p(x) ∈ Z_2[x] of degree 8 in order to compute in E effectively.

3.5 Novel Hashing

ArshPartow combined many hash functions to have "hybrid relative and additive hash function algorithm". This algorithm is used in this paper for efficiency. It is also known as APHash.

3.6 The Proposed Algorithm

The proposed algorithm known as Hybrid Encoding and Data Transformation (HEDT) is presented in this section. It has two procedures known as encoding and decoding for strong data transformations to leverage security.

Algorithm: Hybrid Encoding and Data Transformation (HEDT)
Encoding Procedure

```
Start
Data owner inputs a file F
C←ModifedAESEncrypt(F, sk)
S←IDA(C, m, n)
For each slice s in S
    s←NovelHashing(s)
End For
Outsource S,hash and id to cloud
End
```

Decoding Procedure

```
Start
S←GetFromCloud(id)
Data integrity verification
IF there is integrity THEN
C←IDAReconstruction(S, m, n)
  F←ModifedAESDecrypt(C, sk)
  Return F to data owner
Else
  Recover data
End If
End
```

Algorithm 1: Hybrid Encoding And Data Transformation (HEDT)

As presented in Algorithm 1, it has two procedures involved. They are known as encoding procedure and decoding procedure. The former is meant for secure and reliable data outsourcing while the latter is meant for secure and reliable retrieval of data from cloud. In the process of encoding, the data owner provides a data file to be outsourced. The file F is subjected to modified AES encryption prior to other mechanisms. The F is converted into cipher text C. In the process of encryption, a secret key sk is used. The C is then subjected to IDA to generate slices that bring about data reliability, availability and fault tolerance. The rationale behind this is that few slices can help in reconstruction of C. On the slices, novel hashing procedure is applied and the final data along with hash value is outsourced to public cloud. Such data is said to have application of hybrid encoding and multiple transformations in PQC driven approach. The decoding on the other hand is a reverse process of encoding. It takes S from the cloud and it is subjected data integrity verification. Since hashing is used, data integrity verification is possible. Then the C is reconstructed and decrypted to have the original data F to be given back to data owner. If there is not data integrity, the data is recovered.

4. RESULTS AND DISCUSSION

This section presents experimental results. In addition to performance in terms of encryption/encoding, decryption/decoding, upload time and download time, it also provides security and reliability analysis of the proposed security scheme.

4.1 Execution Results

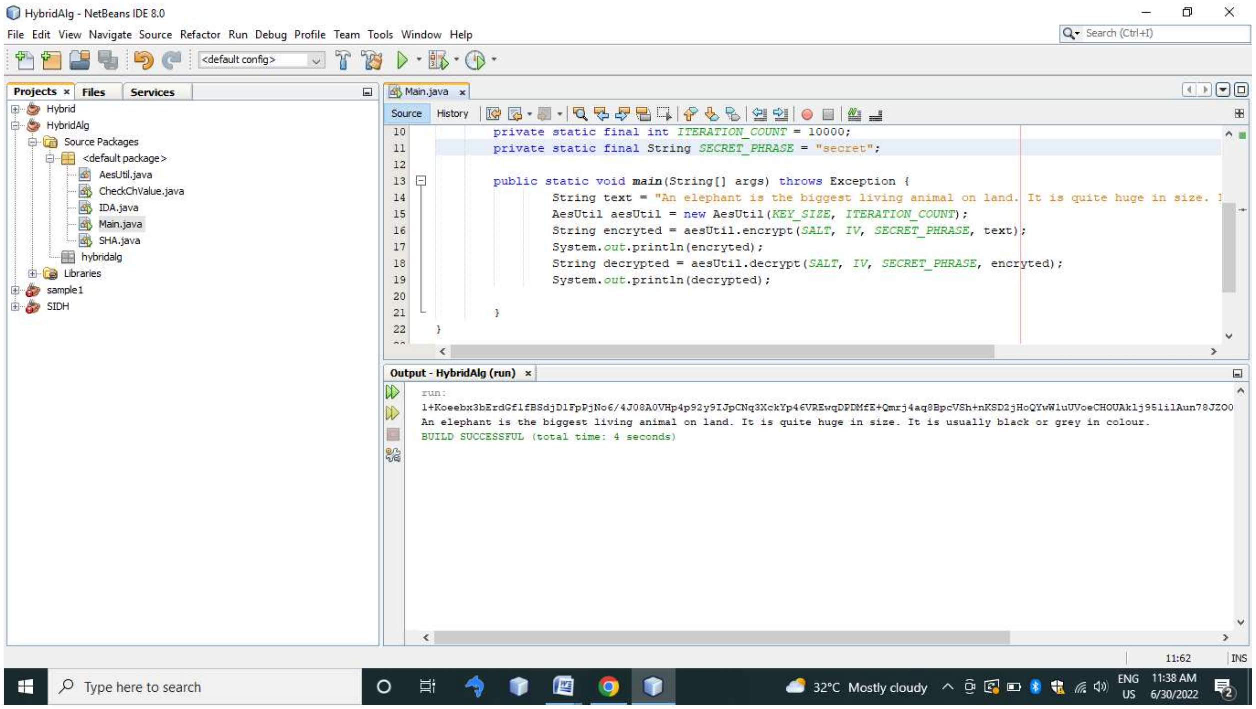
AES

Input

An elephant is the biggest living animal on land. It is quite huge in size. It is usually black or grey in colour.

Output

```
1+Koeebx3bErdGf1fBSdjD1FpPjNo6/4J08A0VH
p4p92y9IJpCNq3XckYp46VREwqDPDMfE+Qmr
j4aq8BpcVSh+nKSD2jHoQYwW1uUVoeCHOU
Aklj951ilAun78JZO0DdVVO4n2yboPCcYEsaZ9
2jqKbIwMvOz6Awyxlfl4gWpA=
```



```
    private static final int ITERATION_COUNT = 10000;
    private static final String SECRET_PHRASE = "secret";

    public static void main(String[] args) throws Exception {
        String text = "An elephant is the biggest living animal on land. It is quite huge in size. I
        AesUtil aesUtil = new AesUtil(KEY_SIZE, ITERATION_COUNT);
        String encryted = aesUtil.encrypt(SALT, IV, SECRET_PHRASE, text);
        System.out.println(encryted);
        String decrypted = aesUtil.decrypt(SALT, IV, SECRET_PHRASE, encryted);
        System.out.println(decrypted);

    }
}
```

```
run:
1+Koeebx3bErdGf1fBSdjD1FpPjNo6/4J08A0VHp4p92y9IJpCNq3XckYp46VREwqDPDMfE+Qmrj4aq8BpcVSh+nKSD2jHoQYwW1uUVoeCHOUAklj951ilAun78JZO0
An elephant is the biggest living animal on land. It is quite huge in size. It is usually black or grey in colour.
BUILD SUCCESSFUL (total time: 4 seconds)
```

Figure 4: Shows Outcome Of Encryption

IDA

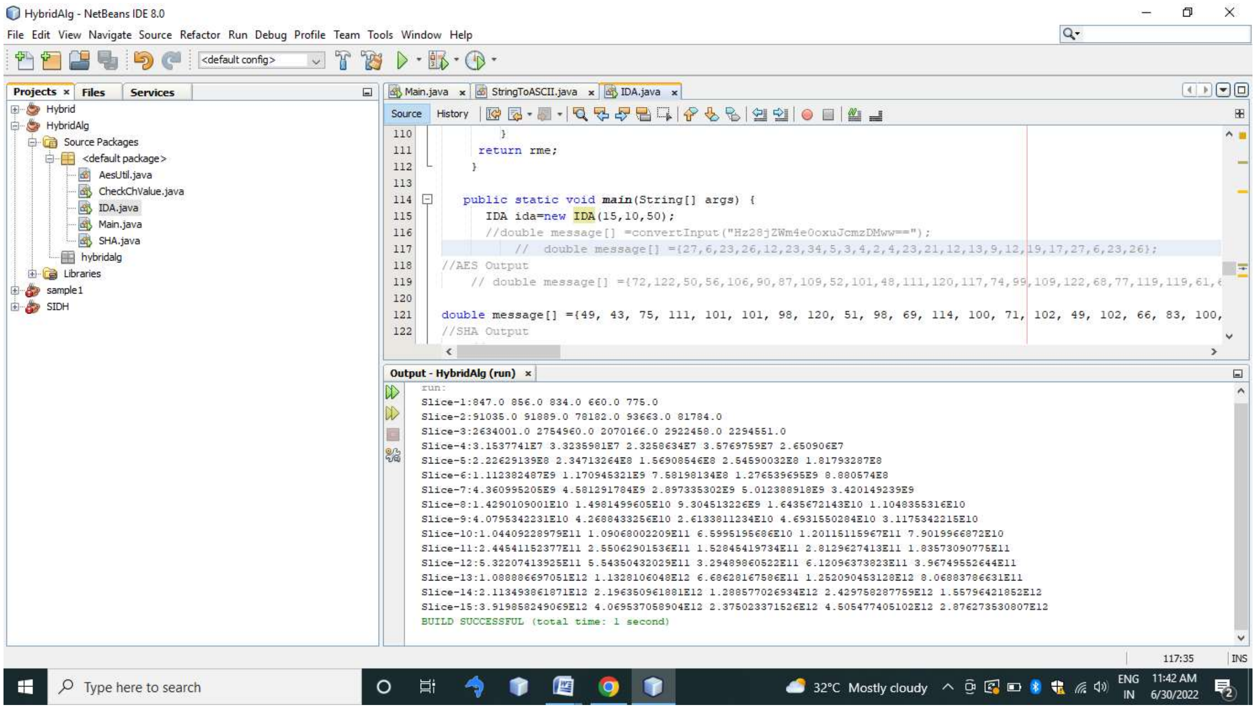
Input

1+Koeebx3bErdGf1fBSdjD1FpPjNo6/4J08A0VHp4p92y9IJpCNq3XckYp46VREwqDPDMfE+Qmrj4aq8BpcVSh+nKSD2jHoQYwW1uUVoeCHOUAklj951ilAun78JZO0DdVVO4n2yboPCcYEsaZ92jqKbIwMvOz6Awyxlfl4gWpA=

[49, 43, 75, 111, 101, 101, 98, 120, 51, 98, 69, 114, 100, 71, 102, 49, 102, 66, 83, 100, 106, 68, 49, 70, 112, 80, 106, 78, 111, 54, 47, 52, 74, 48, 56, 65, 48, 86, 72, 112, 52, 112, 57, 50, 121, 57, 73, 74, 112, 67, 78, 113, 51, 88, 99, 107, 89, 112, 52, 54, 86, 82, 69, 119, 113, 68, 80, 68, 77, 102, 69, 43, 81, 109, 114, 106, 52, 97, 113, 56, 66, 112, 99, 86, 83, 104, 43, 110, 75, 83, 68, 50, 106, 72, 111, 81, 89, 119, 87, 49, 117, 85, 86, 111, 101, 67, 72, 79, 85, 65, 107, 108, 106, 57, 53, 49, 105, 108, 65, 117, 110, 55, 56, 74, 90, 79, 48, 68, 100, 86, 86, 79, 52, 110, 50, 121, 98, 111, 80, 67, 99, 89, 69, 115, 97, 90, 57, 50, 106, 113, 75, 98, 73, 119, 77, 118, 79, 122, 54, 65, 119, 121, 120, 108, 102, 108, 52, 103, 87, 112, 65, 61]

Output

Slice-1:847.0 856.0 834.0 660.0 775.0
Slice-2:91035.0 91889.0 78182.0 93663.0 81784.0
Slice-3:2634001.0 2754960.0 2070166.0 2922458.0 2294551.0
Slice-4:3.1537741E7 3.3235981E7 2.3258634E7 3.5769759E7 2.650906E7
Slice-5:2.22629139E8 2.34713264E8 1.56908546E8 2.54590032E8 1.81793287E8
Slice-6:1.112382487E9 1.170945321E9 7.58198134E8 1.276539695E9 8.880574E8
Slice-7:4.360995205E9 4.581291784E9 2.897335302E9 5.012388918E9 3.420149239E9
Slice-8:1.4290109001E10 1.4981499605E10 9.304513226E9 1.6435672143E10 1.1048355316E10
Slice-9:4.0795342231E10 4.2688433256E10 2.6133811234E10 4.6931550284E10 3.1175342215E10
Slice-10:1.04409228979E11 1.09068002209E11 6.5995195686E10 1.20115115967E11 7.9019966872E10
Slice-11:2.44541152377E11 2.55062901536E11 1.52845419734E11 2.8129627413E11 1.83573090775E11
Slice-12:5.32207413925E11 5.54350432029E11 3.29489860522E11 6.12096373823E11 3.96749552644E11
Slice-13:1.088886697051E12 1.1328106048E12 6.68628167586E11 1.252090453128E12 8.06883786631E11
Slice-14:2.113493861871E12 2.196350961881E12 1.288577026934E12 2.429758287759E12 1.55796421852E12
Slice-15:3.919858249069E12 4.069537058904E12 2.375023371526E12 4.505477405102E12 2.876273530807E12



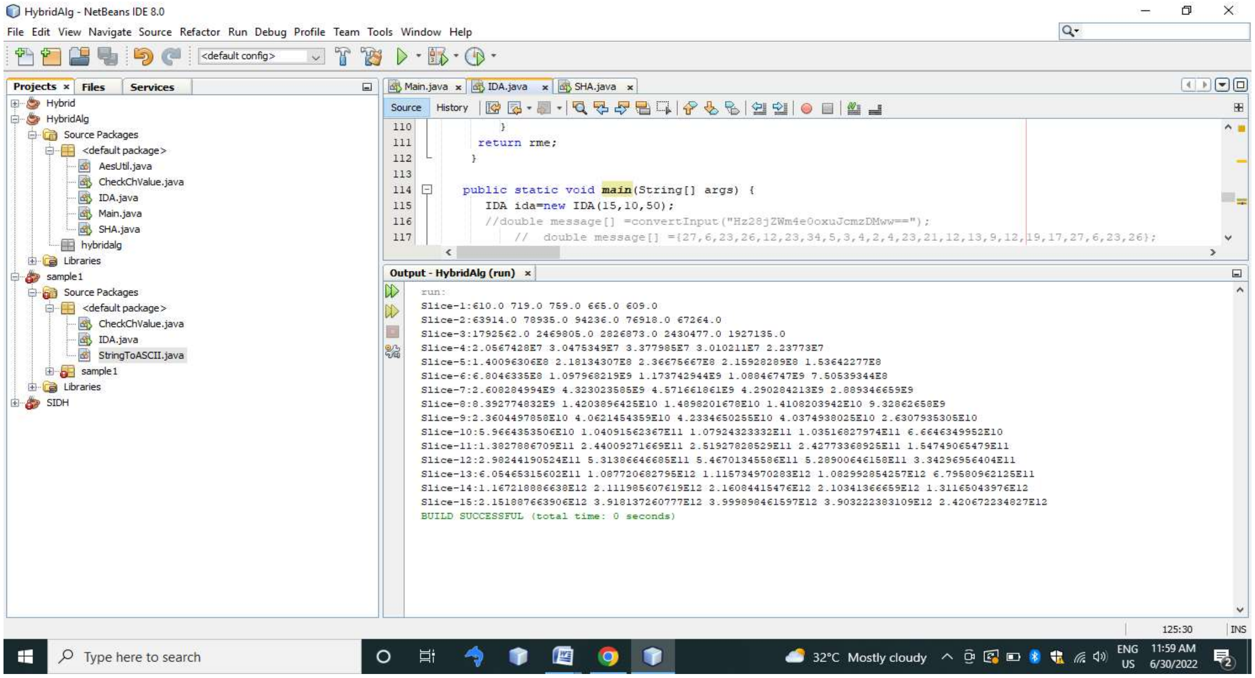
```
             }
         return rme;
        }

    public static void main(String[] args) {
       IDA ida=new IDA(15,10,50);
       //double message[] =convertInput("Hz28jZWm4e0oxuJcmzDMww==");
           //  double message[] ={27,6,23,26,12,23,34,5,3,4,2,4,23,21,12,13,9,12,19,17,27,6,23,26};
//AES Output
      // double message[] ={72,122,50,56,106,90,87,109,52,101,48,111,120,117,74,99,109,122,68,77,119,119,61,6

double message[] ={49, 43, 75, 111, 101, 101, 98, 120, 51, 98, 69, 114, 100, 71, 102, 49, 102, 66, 83, 100,
//SHA Output
```

```
run:
Slice-1:847.0 856.0 834.0 660.0 775.0
Slice-2:91035.0 91889.0 78182.0 93663.0 81784.0
Slice-3:2634001.0 2754960.0 2070166.0 2922458.0 2294551.0
Slice-4:3.1537741E7 3.3235981E7 2.3258634E7 3.5769759E7 2.650906E7
Slice-5:2.22629139E8 2.34713264E8 1.56908546E8 2.54590032E8 1.81793287E8
Slice-6:1.112382487E9 1.170945321E9 7.58198134E8 1.276539695E9 8.880574E8
Slice-7:4.360995205E9 4.581291784E9 2.897335302E9 5.012388918E9 3.420149239E9
Slice-8:1.4290109001E10 1.4981499605E10 9.304513226E9 1.6435672143E10 1.1048355316E10
Slice-9:4.0795342231E10 4.2688433256E10 2.6133811234E10 4.6931550284E10 3.1175342215E10
Slice-10:1.04409228979E11 1.09068002209E11 6.5995195686E10 1.20115115967E11 7.9019966872E10
Slice-11:2.44541152377E11 2.55062901536E11 1.52845419734E11 2.8129627413E11 1.83573090775E11
Slice-12:5.32207413925E11 5.54350432029E11 3.29489860522E11 6.12096373823E11 3.96749552644E11
Slice-13:1.088886697051E12 1.1328106048E12 6.68628167586E11 1.252090453128E12 8.06883786631E11
Slice-14:2.113493861871E12 2.196350961881E12 1.288577026934E12 2.429758287759E12 1.55796421852E12
Slice-15:3.919858249069E12 4.069537058904E12 2.375023371526E12 4.505477405102E12 2.876273530807E12
BUILD SUCCESSFUL (total time: 1 second)
```

Figure 5: Shows information slices after application of IDA

SHA
Input
Slice-1:847.0 856.0 834.0 660.0 775.0
Slice-2:91035.0 91889.0 78182.0 93663.0 81784.0
Slice-3:2634001.0 2754960.0 2070166.0 2922458.0 2294551.0
Slice-4:3.1537741E7 3.3235981E7 2.3258634E7 3.5769759E7 2.650906E7
Slice-5:2.22629139E8 2.34713264E8 1.56908546E8 2.54590032E8 1.81793287E8
Slice-6:1.112382487E9 1.170945321E9 7.58198134E8 1.276539695E9 8.880574E8
Slice-7:4.360995205E9 4.581291784E9 2.897335302E9 5.012388918E9 3.420149239E9
Slice-8:1.4290109001E10 1.4981499605E10 9.304513226E9 1.6435672143E10 1.1048355316E10
Slice-9:4.0795342231E10 4.2688433256E10 2.6133811234E10 4.6931550284E10 3.1175342215E10
Slice-10:1.04409228979E11 1.09068002209E11 6.5995195686E10 1.20115115967E11 7.9019966872E10
Slice-11:2.44541152377E11 2.55062901536E11 1.52845419734E11 2.8129627413E11 1.83573090775E11
Slice-12:5.32207413925E11 5.54350432029E11 3.29489860522E11 6.12096373823E11 3.96749552644E11
Slice-13:1.088886697051E12 1.1328106048E12 6.68628167586E11 1.252090453128E12 8.06883786631E11
Slice-14:2.113493861871E12 2.196350961881E12 1.288577026934E12 2.429758287759E12 1.55796421852E12
Slice-15:3.919858249069E12 4.069537058904E12 2.375023371526E12 4.505477405102E12 2.876273530807E12
Output
87c51013e1a9a4d7757bf5a5890aca6f1f44381b4d161031e881d5fe2b62282c
Reverse Process
SHA
Output
87c51013e1a9a4d7757bf5a5890aca6f1f44381b4d161031e881d5fe2b62282c
IDA
Output
Slice-1:610.0 719.0 759.0 665.0 609.0
Slice-2:63914.0 78935.0 94236.0 76918.0 67264.0
Slice-3:1792562.0 2469805.0 2826873.0 2430477.0 1927135.0
Slice-4:2.0567428E7 3.0475349E7 3.377985E7 3.010211E7 2.23773E7
Slice-5:1.40096306E8 2.18134307E8 2.36675667E8 2.15928289E8 1.53642277E8
Slice-6:6.8046335E8 1.097968219E9 1.173742944E9 1.08846747E9 7.50539344E8

Slice-7:2.608284994E9 4.323023585E9 4.571661861E9 4.290284213E9 2.889346659E9
Slice-8:8.392774832E9 1.4203896425E10 1.4898201678E10 1.4108203942E10 9.32862658E9
Slice-9:2.3604497858E10 4.0621454359E10 4.2334650255E10 4.0374938025E10 2.6307935305E10
Slice-10:5.9664353506E10 1.04091562367E11 1.07924323332E11 1.03516827974E11 6.6646349952E10
Slice-11:1.3827886709E11 2.44009271669E11 2.51927828529E11 2.42773368925E11 1.54749065479E11
Slice-12:2.98244190524E11 5.31386646685E11 5.46701345586E11 5.28900646158E11 3.34296956404E11
Slice-13:6.05465315602E11 1.087720682795E12 1.115734970283E12 1.082992854257E12 6.79580962125E11
Slice-14:1.167218886638E12 2.111985607619E12 2.16084415476E12 2.10341366659E12 1.31165043976E12
Slice-15:2.151887663906E12 3.918137260777E12 3.999898461597E12 3.903222383109E12 2.420672234827E12



```
            }
          return rme;
        }

    public static void main(String[] args) {
       IDA ida=new IDA(15,10,50);
       //double message[] =convertInput("Hz28jZWm4e0oxuJcmzDMww==");
         //  double message[] ={27,6,23,26,12,23,34,5,3,4,2,4,23,21,12,13,9,12,19,17,27,6,23,26};
```

```
run:
Slice-1:610.0 719.0 759.0 665.0 609.0
Slice-2:63914.0 78935.0 94236.0 76918.0 67264.0
Slice-3:1792562.0 2469805.0 2826873.0 2430477.0 1927135.0
Slice-4:2.0567428E7 3.0475349E7 3.377985E7 3.010211E7 2.23773E7
Slice-5:1.40096306E8 2.18134307E8 2.36675667E8 2.15928289E8 1.53642277E8
Slice-6:6.8046335E8 1.097968219E9 1.173742944E9 1.08846747E9 7.50539344E8
Slice-7:2.608284994E9 4.323023585E9 4.571661861E9 4.290284213E9 2.889346659E9
Slice-8:8.392774832E9 1.4203896425E10 1.4898201678E10 1.4108203942E10 9.32862658E9
Slice-9:2.3604497858E10 4.0621454359E10 4.2334650255E10 4.0374938025E10 2.6307935305E10
Slice-10:5.9664353506E10 1.04091562367E11 1.07924323332E11 1.03516827974E11 6.6646349952E10
Slice-11:1.3827886709E11 2.44009271669E11 2.51927828529E11 2.42773368925E11 1.54749065479E11
Slice-12:2.98244190524E11 5.31386646685E11 5.46701345586E11 5.28900646158E11 3.34296956404E11
Slice-13:6.05465315602E11 1.087720682795E12 1.115734970283E12 1.082992854257E12 6.79580962125E11
Slice-14:1.167218886638E12 2.111985607619E12 2.16084415476E12 2.10341366659E12 1.31165043976E12
Slice-15:2.151887663906E12 3.918137260777E12 3.999898461597E12 3.903222383109E12 2.420672234827E12
BUILD SUCCESSFUL (total time: 0 seconds)
```

Figure 6: Shows IDA information slices

Encrypted Form

1+Koeebx3bErdGf1fBSdjD1FpPjNo6/4J08A0VH p4p92y9IJpCNq3XckYp46VREwqDPDMfE+Qmr j4aq8BpcVSh+nKSD2jHoQYwW1uUVoeCHOU Aklj951ilAun78JZO0DdVVO4n2yboPCcYEsaZ9 2jqKbIwMvOz6Awyxlfl4gWpA=

Original Input

An elephant is the biggest living animal on land. It is quite huge in size. It is usually black or grey in colour.

4.2 Performance Analysis

This section presents the HEDT performance and its comparison with other existing schemes such as RSA, DES and AES.

Table 1: Encryption/encoding time Comparison

Data Size (MB)	Encryption/Encoding Time (seconds)			
	RSA	DES	AES	HEDT
10	3.14349	1.001595	0.845985	0.888284
50	4.83441	2.951235	2.649885	2.782379
100	5.661075	3.24891	2.933385	3.080054
500	26.45538	14.90013	14.33639	15.0532

Table 1 presents the performance comparison of the proposed HEDT and existing security schemes in terms of encryption/encoding time against workloads of varied size.

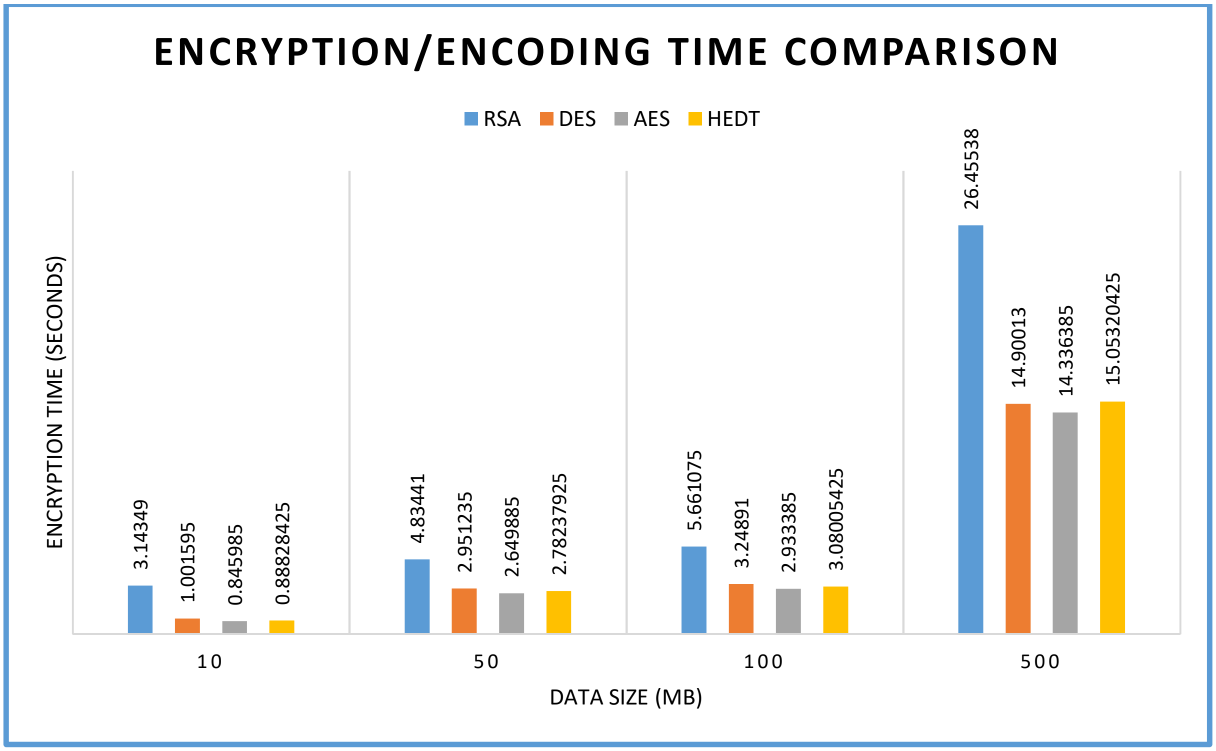


Figure 7:Impact Of Data Size On Security Schemes In Terms Of Encryption/Encoding Time

As presented in Figure 7, the performance of HEDT is compared against other schemes like RSA, DES and AES in terms of encryption/encoding time. The workload is found to have its impact on the execution time. It is evident in the time taken for the schemes to complete encryption/encoding. The results revealed that the RSA took more time when compared to any other scheme. The proposed scheme HEDT is found better than other schemes though it takes more time when compared with that of AES. The overhead is acceptable provided higher level of security.

	Decryption/Decoding Time (seconds)			
Data Size (MB)	**RSA**	**DES**	**AES**	**HEDT**
10	2.771055	1.138935	0.834225	0.875936
50	4.03158	2.410485	2.087295	2.19166
100	5.06037	3.41334	2.67456	2.808288
500	19.90065	10.80524	10.15707	10.66492

Table 2: Decryption/Decoding Time Comparison

Table 2 presents the performance comparison of the proposed HEDT and existing security schemes in terms of decryption/decoding time against workloads of varied size.

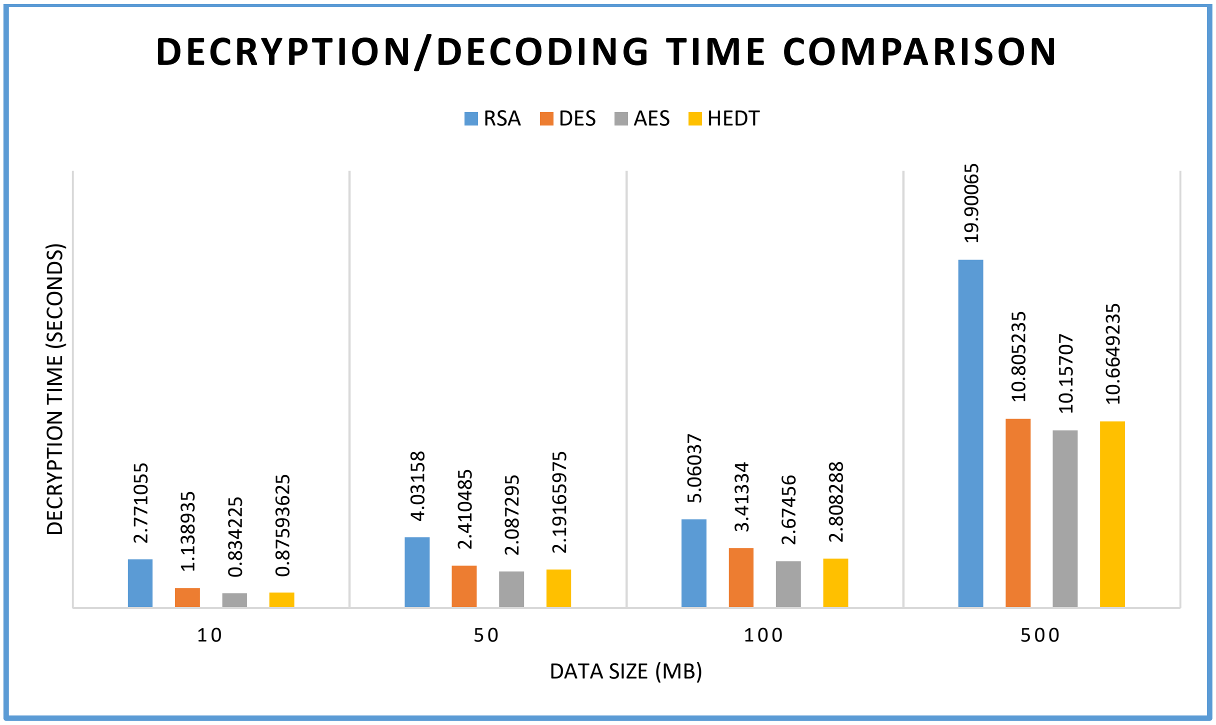


Figure 8: Impact Of Data Size On Security Schemes In Terms Of Decryption/Decoding Time

As presented in Figure 8, the performance of HEDT is compared against other schemes like RSA, DES and AES in terms of decryption/decoding time. The workload is found to have its impact on the execution time. It is evident in the time taken for the schemes to complete decryption/decoding. The results revealed that the RSA took more time when compared to any other scheme. The proposed scheme HEDT is found better than other schemes though it takes more time when compared with that of AES. The overhead is acceptable provided higher level of security.

	Upload Time (seconds)			
Data Size (MB)	**RSA**	**DES**	**AES**	**HEDT**
10	1.805055	0.676515	0.61551	0.58464
50	4.263945	3.127005	2.85201	2.697345
100	9.200205	4.81971	4.70001	4.54209
500	33.68873	19.920705	18.29751	17.83488

Table 3: Upload Time Comparison

Table 3 presents the performance comparison of the proposed HEDT and existing security schemes in terms of upload time against workloads of varied size.

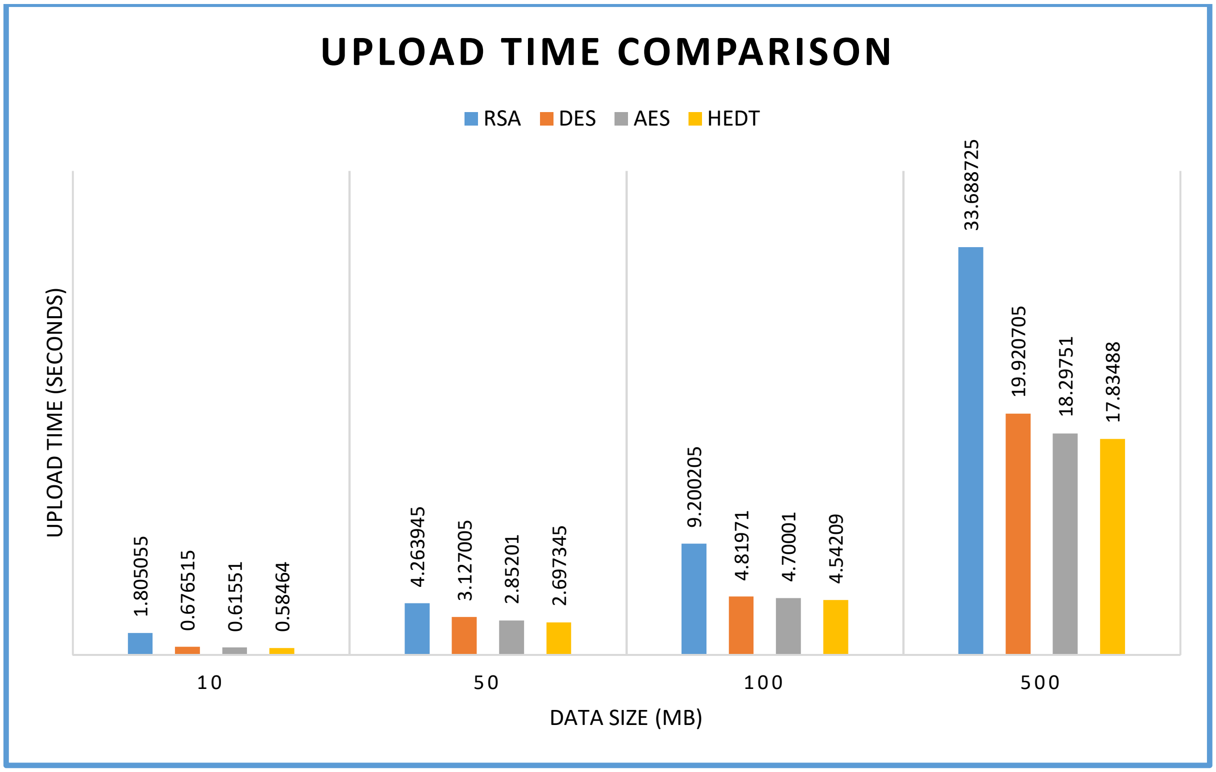


Figure 9: Impact Of Data Size On Security Schemes In Terms Of Upload Time

As presented in Figure 9, the performance of HEDT is compared against other schemes like RSA, DES and AES in terms of upload time. The workload is found to have its impact on the execution time. It is evident in the time taken for the schemes to complete uploading data to cloud. The results revealed that the RSA took more time when compared to any other scheme. The proposed scheme HEDT is found better than other schemes besides providing PQC driven security and reliability.

	Download Time (seconds)			
Data Size (MB)	**RSA**	**DES**	**AES**	**HEDT**
10	2.056635	1.0206	0.84609	0.74529
50	4.924605	3.45807	2.649885	2.08488
100	4.73361	3.351705	2.933385	2.547825
500	24.00615	14.91998	14.33639	12.97464

Table 4: Download Time Comparison

Table 4 presents the performance comparison of the proposed HEDT and existing security schemes in terms of download time against workloads of varied size.

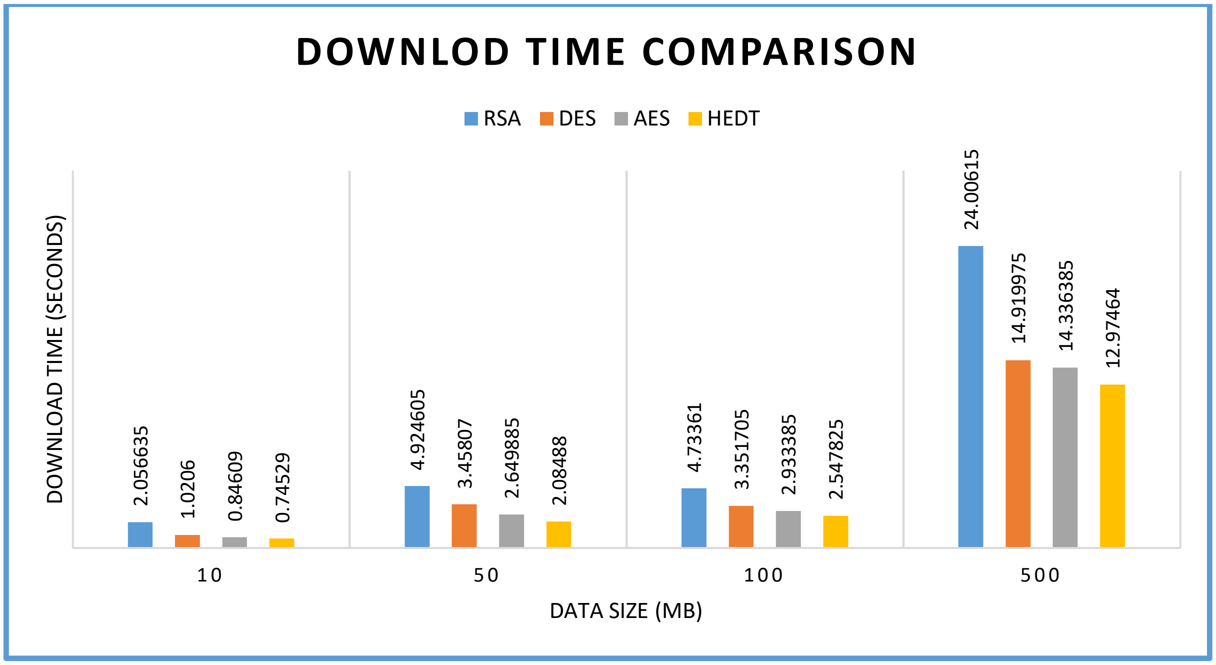


Figure 10: Impact Of Data Size On Security Schemes In Terms Of Download Time

As presented in Figure 10, the performance of HEDT is compared against other schemes like RSA, DES and AES in terms of download time. The workload is found to have its impact on the execution time. It is evident in the time taken for the schemes to complete downloading data from cloud. The results revealed that the RSA took more time when compared to any other scheme. The proposed scheme HEDT is found better than other schemes besides providing PQC driven security and reliability.

4.3 Security Analysis

Unlike other schemes, the proposed HEDT scheme has many benefits. First, its level of security is very high as it is PQC driven scheme. It has encoding and decoding procedures with multiple data transformations. It promotes data availability as the data is not only stored in cloud but also the scheme has provision to have IDA slices that guarantee reconstruction of original data. Even if some data is lost, the slices can help in data recovery. This feature is known as fault tolerance. Fault tolerance can help in data recovery in presence of internal and external attacks. The scheme also provides data integrity as it uses novel hashing that helps in data integrity verification. The scheme also promotes data transmission efficiency.

5. CONCLUSION AND FUTURE WORK

In this paper, we proposed a security algorithm known as Hybrid Encoding and Data Transformation (HEDT)for stronger security in untrusted cloud environments. HEDT has hybrid approach with multiple data transformation to fit into the requirements of QPC. It has multiple transformation of data in order to have higher level of security. With its underlying mechanisms, the proposed algorithm ensures data security, data integrity and availability. The algorithm also supports fault tolerance. With all its features, HEDT is a candidate for PQC security requirements. An empirical study revealed that the HEDT is capable of providing stronger level of security to data outsourced to cloud. The results also show the efficiency of HEDT scheme in data transmission. In future, we intend to build a comprehensive security framework for cloud data security which includes a hybrid key sharing scheme made up of Supersingular Isogeny Diffie-Hellman (SIDH) and Hybrid Encoding and Data Transformation (HEDT) schemes. Besides the HEDT is integrated with the security framework for having holistic approach in cloud data security.

REFERENCES

[1] Wijayanto, Ardhi; Harjito, Bambang (2019). [IEEE 2019 International Conference on Computer, Control, Informatics and its Applications (IC3INA) - Tangerang, Indonesia (2019.10.23-2019.10.24)] 2019 International Conference on Computer, Control, Informatics and its Applications (IC3INA) - Reduce Rounding Off Errors in Information Dispersal Algorithm. , p36–40.

[2]Nadendla, V. SriramSiddhardh; Han, Yunghsiang S.; Varshney, Pramod K. (2015). [IEEE 2015 IEEE International Symposium on Information Theory (ISIT) - Hong Kong, Hong Kong (2015.6.14-2015.6.19)] 2015 IEEE International Symposium on Information Theory (ISIT) - Information-dispersal games for security in cognitive-radio networks. , p1600–1604.

[3] Mar, KhengKok; Hu, ZhengQing; Law, Chee Yong; Wang, Meifen (2016). [IEEE 2016 14th Annual Conference on Privacy, Security and Trust (PST) - Auckland, New Zealand (2016.12.12-2016.12.14)] 2016 14th Annual Conference on Privacy, Security and Trust (PST) - Securing cloud data using information dispersal. , p445–448.

[4]Baldi, Marco; Cucchiarelli, Alessandro; Senigagliesi, Linda; Spalazzi, Luca; Spegni, Francesco (2016). [IEEE 2016 International Conference on High Performance Computing & Simulation (HPCS) - Innsbruck, Austria (2016.7.18-2016.7.22)] 2016 International Conference on High Performance Computing & Simulation (HPCS) - Parametric and probabilistic model checking of confidentiality in data dispersal algorithms. , p476–483.

[5]Deryabin, Maxim; Chervyakov, Nikolai; Tchernykh, Andrei; Berezhnoy, Viktor; Djurabaev, Anvar; Nazarov, Anton; Babenko, Mikhail (2019). [IEEE 2019 24th Conference of Open Innovations Association (FRUCT) - Moscow, Russia (2019.4.8-2019.4.12)] 2019 24th Conference of Open Innovations Association (FRUCT) - Comparative Performance Analysis of Information Dispersal Methods. , p67–74.

[6] Ling Yang and Xianhui Lu. (2017). An Efficient Dispersal Storage Scheme Based on Ring-LWE and NTT. IEEE, p23-30.

[7] Marcelin-Jimenez, Ricardo; Ramirez-Ortiz, Jorge Luis; De La Colina, Enrique Rodriguez; Pascoe-Chalke, Michael; Gonzalez-Compean, Jose Luis (2020). On the Complexity and Performance of the Information Dispersal Algorithm. IEEE Access, 8, p159284–159290.

[8]Makhan Singh, Sarbjeet Singh. (2019). A Framework for Cloud Storage System Based on Information Dispersal Algorithm. International Journal of Recent Technology and Engineering (IJRTE). 7 (6C), p145-148.

[9] Qian, Quan; Yu, Zhi-ting; Zhang, Rui; Hung, Che-Lun (2018). A multi-layer information dispersal based encryption algorithm and its application for access control. Sustainable Computing: Informatics and Systems, p1-12.

[10] Koji Shima(B) and Hiroshi Doi. (2019). A Hierarchical Secret Sharing Scheme Based on Information Dispersal Techniques. Springer, p217–232.

[11] Lee, Bih-Hwang; Dewi, Ervin Kusuma; Wajdi, Muhammad Farid (2018). [IEEE 2018 27th Wireless and Optical Communication Conference (WOCC) - Hualien, Taiwan (2018.4.30-2018.5.1)] 2018 27th Wireless and Optical Communication Conference (WOCC) - Data security in cloud computing using AES under HEROKU cloud. , p1–5.

[12] Panda, Madhumita (2015). [IEEE 2015 IEEE 9th International Conference on Intelligent Systems and Control (ISCO) - Coimbatore, India (2015.1.9-2015.1.10)] 2015 IEEE 9th International Conference on Intelligent Systems and Control (ISCO) - Data security in wireless sensor networks via AES algorithm. , p1–5.

[13]Fathurrahmad, Ester. (2020). Development And Implementation Of The Rijndael Algorithm And Base-64 Advanced Encryption Standard (AES) For Website Data Security. INTERNATIONAL JOURNAL OF SCIENTIFIC & TECHNOLOGY RESEARCH. 9 (11), p6-9.

[14] Kumar, Puneet; B, Shashi (2015). Development of Modified AES Algorithm for Data Security. Optik - International Journal for Light and Electron Optics, p1-5.

[15] Dang, ThanhNha; Vo, Huan Minh (2019). [IEEE 2019 IEEE 4th International Conference on Computer and Communication Systems (ICCCS) - Singapore (2019.2.23-2019.2.25)] 2019 IEEE 4th International Conference on Computer and Communication Systems (ICCCS) - Advanced AES Algorithm Using Dynamic Key in the Internet of Things System. , p682–686.

[16]Akhil, K. M.; Kumar, M Praveen; Pushpa, B. R. (2017). [IEEE 2017 International Conference on Intelligent Computing and Control (I2C2) - Coimbatore, India (2017.6.23-2017.6.24)] 2017 International Conference on Intelligent Computing and Control (I2C2) - Enhanced cloud data security using AES algorithm. , p1–5.

[17] Kumar, Keshav; Ramkumar, K.R.; Kaur, Amanpreet (2020). [IEEE 2020 8th International Conference on Reliability, Infocom Technologies and Optimization (Trends and Future Directions) (ICRITO) - Noida, India (2020.6.4-2020.6.5)] 2020 8th International Conference on Reliability, Infocom Technologies and Optimization (Trends and Future Directions) (ICRITO) - A Design Implementation and Comparative Analysis of Advanced Encryption Standard (AES) Algorithm on FPGA. , p182–185.

[18]Katkade, Pradnya; Phade, G. M. (2016). [IEEE 2016 International Conference on Inventive Computation Technologies (ICICT) - Coimbatore (2016.8.26-2016.8.27)] 2016 International Conference on Inventive Computation

Technologies (ICICT) - Application of AES algorithm for data security in serial communication. , p1–5.

[19] Zhang, Qi; Ding, Qun (2015). [IEEE 2015 Fifth International Conference on Instrumentation & Measurement, Computer, Communication and Control (IMCCC) - Qinhuangdao, China (2015.9.18-2015.9.20)] 2015 Fifth International Conference on Instrumentation and Measurement, Computer, Communication and Control (IMCCC) - Digital Image Encryption Based on Advanced Encryption Standard (AES). , p1218–1221.

[20] Yu, Liting; Zhang, Dongrong; Wu, Liang; Xie, Shuguo; Su, Donglin; Wang, Xiaoxiao (2018). [IEEE 2018 17th IEEE International Conference On Trust, Security And Privacy In Computing And Communications/ 12th IEEE International Conference On Big Data Science And Engineering (TrustCom/BigDataSE) - New York, NY, USA (2018.8.1-2018.8.3)] 2018 17th IEEE International Conference On Trust, Security And Privacy In Computing And Communications/ 12th IEEE International Conference On Big Data Science And Engineering (TrustCom/BigDataSE) - AES Design Improvements Towards Information Security Considering Scan Attack. , p322–326.

[21] Chen, Gang; Wu, Sai; Wang, Yuan (2015). The Evolvement of Big Data Systems: From the Perspective of an Information Security Application. Big Data Research, 2(2), p65–73.

[22]AkankshaRawat, 2Deepak Agrawal. (2015). An Enhanced Message Digest Hash Algorithm for Information Security. International Journal of Recent Research in Electrical and Electronics Engineering (IJRREEE). 2 (1), p54-62.

[23]Botacin, M., GalhardoMoia, V. H., Ceschin, F., AmaralHenriques, M. A., &Grégio, A. (2021). Understanding uses and misuses of similarity hashing functions for malware detection and family clustering in actual scenarios. Forensic Science International: Digital Investigation, 38, 301220, p1-19.

[24] Wu, Hsin-Te; Homg, Gwo-Jiun (2016). [IEEE 2016 International Conference on Advanced Materials for Science and Engineering (ICAMSE) - Tainan, Taiwan (2016.11.12-2016.11.13)] 2016 International Conference on Advanced Materials for Science and Engineering (ICAMSE) - Vehicular cloud network and information security mechanisms. , p196–199.

[25]Topcu, Berkay; Karabat, Cagatay; Azadmanesh, Matin; Erdogan, Hakan (2016). Practical security and privacy attacks against biometric hashing using sparse recovery. EURASIP Journal on Advances in Signal Processing, 2016(1), p1-20.

[26] Timothy, DivyaPrathana; Santra, Ajit Kumar (2017). [IEEE 2017 International conference on Microelectronic Devices, Circuits and Systems (ICMDCS) - Vellore, India (2017.8.10-2017.8.12)] 2017 International conference on Microelectronic Devices, Circuits and Systems (ICMDCS) - A hybrid cryptography algorithm for cloud computing security. , p1–5.

[27]Punam V. Maitri and ArunaVerma. (2016). Secure file storage in cloud computing using hybrid cryptography algorithm. IEEE, p1635-1638.

[28]Chinnasamy, P.; Deepalakshmi, P. (2018). [IEEE 2018 Second International Conference on Inventive Communication and Computational Technologies (ICICCT) - Coimbatore, India (2018.4.20-2018.4.21)] 2018 Second International Conference on Inventive Communication and Computational Technologies (ICICCT) - Design of Secure Storage for Health-care Cloud using Hybrid Cryptography. , p1717–1720.

[29] Ahmad, SadiqAliyu; Garko, Ahmed Baita (2019). [IEEE 2019 15th International Conference on Electronics, Computer and Computation (ICECCO) - Abuja, Nigeria (2019.12.10-2019.12.12)] 2019 15th International Conference on Electronics, Computer and Computation (ICECCO) - Hybrid Cryptography Algorithms in Cloud Computing: A Review. , p1–6.

[30] Feng, Ruijue; Wang, Zhidong; Li, Zhifeng; Ma, Haixia; Chen, Ruiyuan; Pu, Zhengbin; Chen, Ziqiu; Zeng, Xianyu (2020). A Hybrid Cryptography Scheme for NILM Data Security. Electronics, 9(7), p1-18.