IJEMD-CSAI, 4 (1) (2025) https://doi.org/10.54938/ijemdcsai.2025.04.1.363

International Journal of Emerging Multidisciplinaries:
Computer Science and Artificial Intelligence

Research Paper
Journal Homepage: www.ojs.ijemd.com
ISSN (print): 2791-0164 ISSN (online): 2957-5036

Secure File Storage on Cloud Using Hybrid Cryptography

Victoria Zevini Sabo [1*] , Jesse Mazadu Ismaila [1]

1. Federal University Wukari, Nigeria.

Abstract

For decades, cloud computing has empowered the widespread and storage of information. Its evolution has provisioned concepts for ubiquitous computing enabling accessibility to individual records without barriers to location. However, the proliferation of cloud computing provides a forum for cybercriminal to experiment. Lately, cybercrimes proliferation has been of great concern to researchers. This have given this study the impetus to evaluate the performance of the Chacha20 and ECC algorithm while hybridizing it with added layer of security. The performance of the algorithms was evaluated against some metric including the file size and encryption and decryption time. The implemented algorithms were further compared with some of the state-of-the-art algorithm. The comparison shows that the implemented ECC and Chacha20 algorithm performed better compared to some of the compared state of the art algorithm.

Keywords: Cybercriminal, Encryption, Cryptography, Cloud, Algorithm

INTRODUCTION

Over the past decades, security has been a major concern in a wide range of applications and cloud storage applications are no exception to these security threats [1]. Furthermore, cybercrimes are proliferating at their peak with the high demand for an immediate and effective solution. Cybercrime, also called computer crime, involves the use of a computer system as an instrument to perpetrate illegal acts, such as bank fraud, intellectual property violation, stealing identities or violating individual privacy, and ransom demands over the internet [2]. Cybercrime perpetrators also referred to as cybercriminals harness information meant to be confidential with the aim of demanding ransom or blackmailing individuals, firms, and users[3]. Moreover, cybercriminals extend their activities to the denial of services for authorized users which is appalling to cloud users.

Email Addresses: victoriasabo@fuwukari.edu.ng **(Victoria) ,** jesse@fuwukari.edu.ng **(Jesse)**

However, advances in technology are proliferating at an exceptionally quick speed while conveying organizations, institutions, etc., with numerous alluring security measures to enhance data security and confidentiality when transmitting information over the cloud [4]. This growth has portrayed the field of cloud computing as the subject that ensures information security and confidentiality. It is important to note that this information might be any classified data, which is needed by a client to be protected from any noxious transactions like-medical services data, bank exchange, Visa subtleties, and so on. The need for information security has prompted the field of cloud computing for security and insurance of information from any unapproved client as spillage of private data can result in service denial to a genuine client [3] One of the significant procedures in cloud computing to accomplish this prerequisite "information security" is cryptography, otherwise called "code making" or "code age". Cryptography techniques have been widely used to encrypt or decrypt transmittable cloud information.

Cryptography is a science of secret writing, a type of secure communication understood by the sender and intended recipient only [5]. Although the encrypted data can be noticed during transmission, the content of that data should remain unknown to third parties. Data in motion (moving on a network) and data at rest (stored on a device, such as a disk) can also be encrypted for security [5]. Furthermore, cryptography can provide confidentiality (secrets remain secret) and integrity (data is not altered without authorization) with added functionalities to provide authentication, which proves an identity claim [6]. Additionally, cryptography can provide nonrepudiation, which is an assurance that a specific user performed a specific transaction that did not change [6]. Moreover, the concept of cryptography encompasses the application of the approach of encryption and decryption [7]. Encryption transforms a database into an unreadable text whereas decryption is the method of converting the cipher text to plain text as the opposite of encryption. A cipher is a series of two algorithms that are used to construct the encoding and decoding operations. The algorithm and a key are in charge of a cipher's lengthy operation. It's a message, a short string of symbols that would decipher the encrypted data.

Cryptography exists in two forms namely symmetrical and asymmetrical cryptography. The symmetric security algorithm has a generic key shared between the sender and receiver, while asymmetric key algorithms are used with different keys for encryption and decryption [8]. Public key encryption is based on computationally extensive mathematical functions, many different approaches have also been proposed to provide data protection in the cloud, such as AES, DES, and RSA, but existing systems often fail when only a certain form of encoding is utilized, either AES, DES, or RSA depending on a consumer requirement.

However, the major issue with this scheme is that each encryption is done with encryption keys, and if these keys are leaked in some manner, the entire data is destroyed, hence, the need for a solution that can have additional security is essential. As a result, a hybrid cryptography is proposed in this study, in which the Chacha20 and Elliptic Curve Cryptography encryption and decryption algorithms are combined. The study conducted revealed that a symmetric cryptosystem uses only one private key for both encryption and decryption of the data. Hence, in a symmetric cryptosystem, the encrypted message is sent over without any public keys attached to it after the private key has been transmitted [9]. Taking cognizance of the symmetric techniques, there exists a problem of key transportation as the secret key is to be transmitted to the receiving system before the actual message is to be transmitted. Added to the problem of its secret key transmission is the fact that the usage of one secret key for both encryption and decryption is vulnerable to detection because every means of electronic communication is insecure as it is impossible to guarantee that no one will be able to tap communication channels [10].

Therefore, the only secure way of exchanging keys would be exchanging them personally. Although the symmetric cryptosystem is faster the asymmetric cryptosystem portrays its viability of not exchanging keys, thus eliminating the key distribution problem via the application of public and private keys [11]. The keys are generated in such a way that it is impossible to derive the private key from the public key. The transmitter and the receiver both have keys in the asymmetric system. However, the private key is kept private by the transmitter and not sent over with the message to the receiver, although the public key is sent. A major drawback of the asymmetric system is speed. Considering the merit of both the asymmetric and symmetric systems, this study proposed the provision of security by integrating both an asymmetric and symmetric cryptographic algorithm, namely the Chacha20 and Elliptic Curve Cryptography algorithms respectively.

The aim of this study is to develop secure file storage on the cloud using hybrid cryptography. The specific objectives are to:

i. Apply the Chacha20 and Elliptic Curve Cryptography algorithm.

ii. Evaluate the performance of the Chacha20 and Elliptic Curve Cryptography encryption.

iii. Compute the throughput of the applied algorithms.

iv. Compare the performance of the implemented algorithms against some state-of-the-art algorithms.

LITERATURE REVIEW

Current Implementation of Hybrid Cryptography in Securing File Storage on Cloud Infrastructure

Users are slowly shifting away from traditional storage devices such as this problem by citing that a not insignificant 32% of SMBs expressed that finding the detection of unauthorized access to data and information stored on the Cloud is more difficult after migrating their data from on-site storage infrastructure. Cloud customers endorse the convenience of cloud storage, they are yet careful in trusting thumb drives, hard disks, and other physical storage devices that are becoming obsolete. This change has risen due to the globalization of business that has necessitated sharing data for collaborative working and using multiple personal devices. However, cloud storage technologies are yet to introduce various data storage security risks such as leakage, unwarranted access, and illegal modification. Such risks have necessitated the implementation of hybrid cryptography and other techniques of ensuring data on cloud storage facilities is secure [12].

Information systems security experts implement hybrid cryptography by combining at least two varying cryptographic algorithms. The first approach uses RSA and AES algorithms, whereas the second uses AES and Blowfish algorithms [13]. The three main functions in Rivest-Shamir-Adleman (RSA) are key generation, encryption, and decryption and the best example of symmetric cryptography is Advanced Encryption Standard (AES), which uses various bit length keys such as 128, 192, and 256. It is a combination of Exclusive-OR operation, octet substitution with s-box, row, and column rotations, and a mixed column in the flow of the algorithm [13].

In the first approach, RSA algorithms are used for key encryption, while AES is used to encrypt text or data. Data uploads on the Cloud require that an AES secret key and RSA public key be present. When a user attempts to upload to the Cloud, the file being uploaded is stored in a directory temporarily as it awaits encryption. During encryption, the RSA algorithm is applied to encrypted data; then, the AES algorithm is applied to the file. The RSA key is then applied to convert the file into an encoded form.

The reverse occurs during decryption [14]. Studies by [15] on the first approach show that the combined implementation of AES alongside RSA ensures efficiency and guarantees cloud storage servers' consistency and trustworthiness.

The study sought to apply various cryptographic techniques during data communication while harnessing cloud computing power to improve the security of ciphertext and encrypted data while simultaneously minimizing time, cost, and memory consumption during the encryption and decryption phases. Research findings revealed that hybrid encryption with RSA and AES consumed significantly less time than the original RSA [15]. The second approach for hybrid cryptography involves implementing AES and Blowfish to provide double encryption over keys and data. This double encryption effectively ensures a higher level of security than the first approach [15]. Another study observes that this hybrid of AES and Blowfish guarantees better security by increasing complexity .

AES is considered the best symmetric encryption algorithm and is considered more secure than Blowfish. However, this combination's downside is that it has a lower throughput and fails to achieve optimal memory usage because Blowfish itself has utilized high quantity of memory [15].

However, another hybridization technique involves the combination of Blowfish and ECC (Elliptic Curve Cryptography), which is an emerging alternative for traditional public-key cryptosystems, such as RSA, and which a study argues is the best substitute for asymmetric encryption [15]. ECC is in itself founded on the "toughness of the discrete logarithm problem (DLP), whose network bandwidth is little, and the public key is short. These characteristics make it difficult to guess the keys of the encryption technique and hence render it resistant to attacks [16]. With ECC, encrypts each file and stores it on more than one Cloud. File information is stored on the cloud server for decryption. Storing files on more than one Cloud achieves security because it ensures that an attacker attempting to acquire the original file can only get a part of it [17].

Furthermore, even when an attacker somehow finds access to any of the techniques' keys, they may not decrypt it in a finite number of life-years [16]. This characteristic is attributable to the fact that ECC algorithms for the encoding and decoding processes require maximum time. ECC is also beneficial in that it offers less overhead and executes encryption better than RSA. On the other hand, Blowfish offers higher throughput than other algorithms [16]. Hybrid cryptography consisting of Blowfish and ECC guarantees less overhead, better execution of encryption processes, and higher throughputs.

Hybrid cryptography is also implemented by combining the Krishna and Triple DES algorithms. This hybrid cryptography system allows users of a cloud storage facility to choose an encryption algorithm that they consider most suitable to the type of data they intend to upload to the Cloud. The system also determines time-efficient and secure encryption algorithms that facilitate data protection as it migrates from a mobile to a cloud platform [18]. During the encryption process, a plain-text file is first encrypted using the Krishna algorithm that uses a secret key merged with public random bits and shared between sender and receiver. This strategy facilitates encryption and decryption [18].

The encryption of the file using Krishna produces a ciphertext, C1k. C1k then undergoes a further sequence of three encryption processes using Triple DES. Triple DES key 1 creates ciphertext C2ktd1, Triple DES key 2 creates cipher text C3ktd2, and Triple DES key 3 creates the final cipher text C4ktd3. During the decryption phase, Triple DES key three decrypts C4ktd3 into C4ktd2, Triple DES key 2 decrypts C4ktd2 into C4ktd1, and Triple DES key 1 decrypts C4ktd1 into C1k. The Krishna algorithm then decrypts C1k into the original plain-text file [18]. The study shows that a combination of Krishna and Triple DES algorithms offers the best ratio of file size to encryption time and is suitable for securing large files in the least amount of time.

Hybrid cryptography is also achieved through a combination of Krishna and AES algorithms. During the encryption phase, the Krishna algorithm is applied to convert the plain-text file into a ciphertext, C1k. The AES algorithm is then utilized to further encrypt C1k into ciphertext C2kA that is the final cipher [19]. The reverse occurs during the decryption phase. The AES key decrypts ciphertext C2kA to C1k. Krishna then decrypts C1k further to reproduce the original plain-text file.

Overall, the implementation of hybrid cryptographic techniques is better than implementing either symmetric or asymmetric cryptography. In their analysis of cloud storage security, [20] discovered that hybrid cryptography is better poised to ensure the attainment of security techniques for data protection that have been accepted universally in the field of information security. These techniques are achieved through mechanisms of access control, authorization, authentication, and confidentiality. A 2016 study appreciates that cloud storage services subscribers can only trust the infrastructure’s data protection capabilities when the prevailing data protection system have the mechanisms above into account [20].

Hybrid Cryptography Scheme

Hybrid cryptosystem work on cloud to secure data. Secluded servers are presumed to be trustworthy, server-side encryption is used for files, and then after files are encrypted on the server, they are stored there. Hybrid cryptography combines few of the hybrid cryptosystem algorithms that are discussed below along with their advantages and disadvantages for secure cloud storage.

A hybrid data encryption system that would use both RSA and Blowfish was implemented in [21]. In this, they used a mathematical methodology to implement the Field Programmable Gate Array (FPGA). This strategy is very effective given its low cost and high level of protection. But key size (448 bits) is the primary issue. [22] suggested the use of a hybrid cryptographic technique to protect cloud file storage. They used steganography with least significant bit (LSB) by which the encryption key is covered into a picture header for key information integrity.

[23], an innovative technique of hybrid cryptography was developed for health records. In that, they used Blowfish and enhanced RSA algorithms to improve patient data security and prevent false requests. [24] they presented a hybrid technique (AES-RSA) for lightweight data. However, it cannot be applied to multimedia data as it provides security for lightweight data only. In addition to Order Preserving Symmetric Encryption (OPSE), symmetric searchable encryptions were employed. System analysis has shown its usefulness in the case of a graded keyword search, but attacks, integrity and confidentiality are not relevant information. So, it might not be appropriate to provide security. Incremental encryption enables data to be encrypted and exchanged with other authorized users with a different encryption key before being stored in the cloud.

[25] proposed to exchange data in the cloud using RSA and they have used the MD5 algorithm for data integrity. They utilized the RSA algorithm to encrypt large data files to enhance data security in the cloud. Sarkar and Kumar, (2016)[26] recommended a method for ensuring cloud data protection using hybrid encryption. This strategy would also boost data protection at a high overhead communication rate in the cloud. [23] introduced a novel technique which produces access control as a service using multilabel (SMBACaaS). They have used an improved key generation scheme of RSA (IKGSR) for generating key and signature to achieve better confidentiality and security. The different types of cryptographic algorithms are analyzed in [27] and are used in modern cloud storage. This study gave a quick summary of various security concerns, and how we can use cryptographic methods to create stable cloud storage systems.

[28] suggested a user data encryption system before being transmitted to the cloud. AES is used to encrypt user data, and the RSA algorithm encodes the secret key. The same operation for decryption is followed, too. The hybrid strategy had been used to combat cloud DOS attacks. Similar to other methods, the only downside of this approach is more time-consuming. 29] created a new data storage architecture using the cryptographic hybrid model. Secure data storage is obtained by using the AES, Blake2b and Schnorr Signature algorithms. The service provider is unknown about the personal encryption method to provide a high level of security because data encryption is performed on the client-side before uploading to the cloud.

The method is nonetheless incompatible with multimedia files.

[30] proposed the use of both symmetric key (OneTime-Pad) and Asymmetric-key algorithm (RSA) to provide strong security. The product of this approach offered better security. The time taken to encipher data is also faster than the process already in use. proposed a new method for cloud services with XaaS architecture. The authors suggested Cloud Encryption as a Service (EaaS) by which the service provider encryption security risk is reduced and client-side protection is enhanced [31].

METHODOLOGY

The study conducted has identified the ever-growing need to secure data transmission over the cloud distinctively. Hence, this study is the adaptation of the Chacha20 for two-layer security for the encryption and decryption of user information over the cloud environment. Furthermore, this study intended to harness the power of symmetric and asymmetric key cryptographic techniques for providing more security. Multilevel encryption and decryption are proposed to enhance the system's security.

From symmetric key cryptography, the advanced encryption standard algorithm was identified for the first level of encryption meanwhile, the Elliptic Curve Cryptography (ECC) algorithm was identified from the Asymmetric key cryptography for the second level of encryption. Symmetric key encryption can use either stream ciphers or block ciphers. Stream ciphers encrypt one digit at a time of a certain text. Block ciphers take several bits and encrypt them as a single unit. Blocks of 64 bits have been commonly used. Whereas the asymmetric encryption model is used to solve key distribution where it focuses on generating two types of keys [32].

To harness the power of cryptography for the encryption of users' sensitive information, this study devised a two-step methodological approach for the encryption and decryption of information between the receiver and sender. The first phase entails the encryption of plain users' information captured by the sender. In summary, the step includes passing the captured user's information via the proposed web platform which then encrypts the user's information before transmitting this information to the intended users, using the aforementioned and discussed two-layer security (namely the Chacha20 and ECC encryption techniques) before proceeding to the second phase of the methodology. The second phase involves the decryption of the transmitted and encrypted information at the receiver's end. The process is conducted in reverse order with ECC being the first decryption layer followed by the Chacha20.

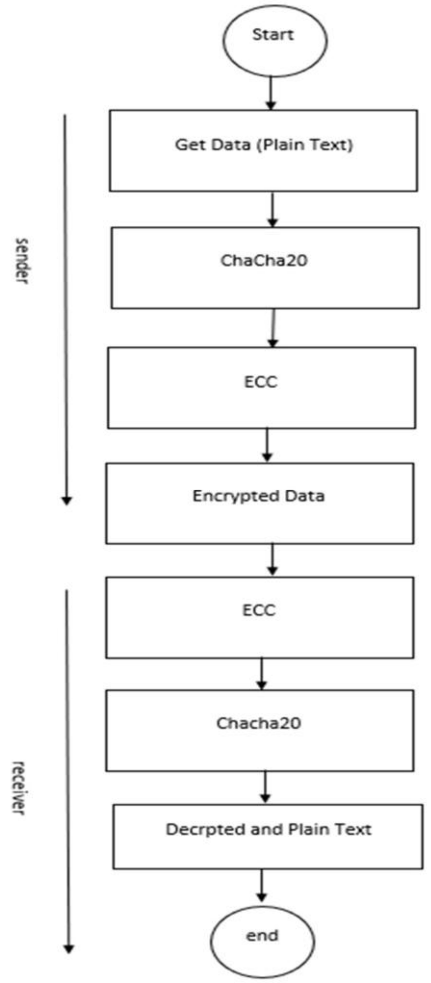


Figure 1: Research Methodology Material and Tools To implement the hybrid cryptographic system, this study proposes the utilization of the below utilities:

i. Python Software Development Kit (SDK).

ii. Visual Studio is an integrated development environment for the implementation of the proposed system.

Description of the Proposed Cryptographic Method

Chacha20

Chacha20 is a secure and fast symmetric cryptographic algorithm that uses a 512-bit key to both encrypt and decrypt data. The algorithm is a refinement of the initial Salsa 20. It was developed by Daniel J. Bernstein, a well-known cryptographer, in 2008 as a stream cipher, and later revised in 2014 as a block cipher [33]. The Chacha20 encryption algorithm is designed to provide both speed and security. Furthermore, the algorithm is designed to be resistant to known attacks, including differential cryptanalysis and linear cryptanalysis.

As aforementioned, the Chacha20 cipher algorithm generates a 512-bit key stream from 4 × 4 initial matrixes composed of 16-32-bit basic operational units [33]. The Chacha20 function takes as input 4 constants {C , C , C , C } 8 keys {K , K , … , K }, 1 counter { t }, and 3 nonces {n , n , n } arranged as follows:

$$X[16] = \begin{matrix} x & x & x & x \\ x & x & x & x \\ x & x & x & x \\ x & x & x & x \end{matrix} = \begin{matrix} c & c & c & c \\ k & k & k & k \\ k & k & k & k \\ t & n & n & n \end{matrix} \qquad 1$$

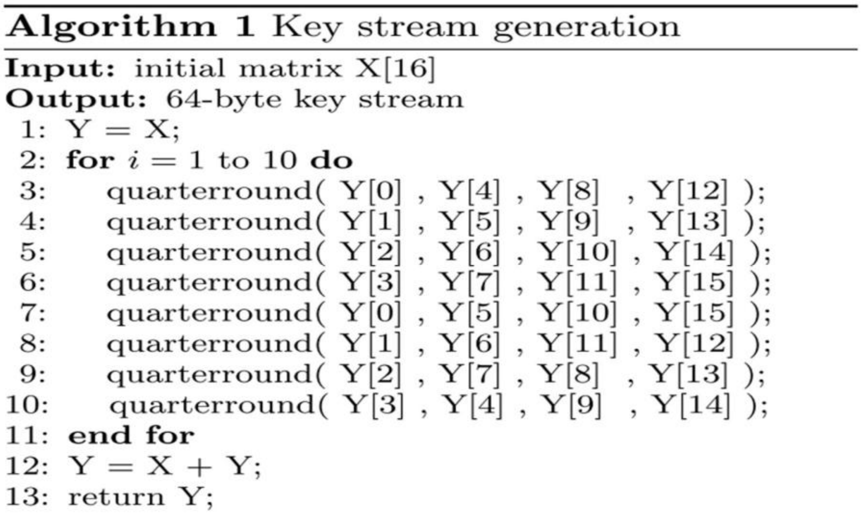
With Chacha20, the input used to generate the key stream is independent of the ciphertext. At each time or iteration, the algorithm encrypts 512-bit plaintext using 512-bit key stream generated by an initial matrix. When the plaintext length exceeds 512-bit, encrypting all plaintext requires repeated use of key stream generation and encryption. In the initial matrix of each encryption, the 256-bit keys and the 128-bit constants remain the same, the 32-bit counter starts at 0 and increments by 1 for each encryption, and the 96-bit nonces need to be guaranteed to be different in each encryption. The quarter round, which is a series of addition" + ", XOR , "⊕" and left rotation " ≪ " operations on four 32-bit unsigned integers, is the most basic and important operation in generating key stream. The quarter round (a, b,c,d) operation is as follows:

$$a{+}{=}\, b;\ d = (d \oplus a) \ll 16 \qquad 2$$

$$c{+}{=}\, d;\ b = (b \oplus c) \ll 12 \qquad 3$$

$$a{+}{=}\, b;\ d = (d \oplus a) \lll 8 \qquad 4$$

$$c{+}{=}\, d;\ b = (b \oplus c) \lll 7 \qquad 5$$



Algorithm 1 Key stream generation

```
Input: initial matrix X[16]
Output: 64-byte key stream
Y = X;
for $i = 1$ to 10 do
    quarterround( Y[0] , Y[4] , Y[8]  , Y[12] );
    quarterround( Y[1] , Y[5] , Y[9]  , Y[13] );
    quarterround( Y[2] , Y[6] , Y[10] , Y[14] );
    quarterround( Y[3] , Y[7] , Y[11] , Y[15] );
    quarterround( Y[0] , Y[5] , Y[10] , Y[15] );
    quarterround( Y[1] , Y[6] , Y[11] , Y[12] );
    quarterround( Y[2] , Y[7] , Y[8]  , Y[13] );
    quarterround( Y[3] , Y[4] , Y[9]  , Y[14] );
end for
Y = X + Y;
return Y;
```

Chacha20 algorithm runs 20 rounds and is mostly completed in 10 iterations as shown by Algorithm 1. Each iteration encompasses of a “column” round and a “diagonal” round with round having four quarter rounds. In each iteration, quarter rounds 1–4 are part of a “column” round, while 5-8 are part of a “diagonal” round. At the end of the 20 rounds, the calculated matrix is added to the initial matrix to obtain the output key stream. After the key stream is generated, the ciphertext can be obtained by XOR

operation between the 512-bit plaintext and the 512-bit key stream. And decryption is to obtain plaintext through ciphertext and key stream in the same way.

Elliptic Curve Cryptography (ECC)

Elliptic Curve Cryptography (ECC) is an asymmetric cryptography that is gaining massive acceptance by many security experts as an alternative to the RSA cryptographic algorithm. ECC is a public-key encryption technique based on elliptic curve theory and can thus create faster, smaller, and more efficient cryptographic keys through the properties of the elliptic curve equation [34]. Hence, to break down an ECC algorithm, an attacker must compute an elliptic curve discrete logarithm, which is significantly more difficult than factoring. As a result, ECC key sizes can be significantly smaller than those required by RSA while still delivering equivalent security with lower computing power and battery resource usage. The elliptic curve can be defined by a Weierstrass equation:

$$Y \quad = X \quad + ax + b(\mathrm{mod}\ p) \qquad 6$$

Here, p is a prime number p ≠2 and 3, and a and b are the constants with respect to the curve satisfying 4a + 27b ¬≡ 0(mob p). The (mod p) indicates that the algorithm will be dealing with curve in a finite field. The curve also consists of the point at infinity O. The curve also has a point G known as the curve generator whose point multiplication can generate all points of the field F. In summation, the curve can be written as: E F = {a, b, p, G}. The properties of Elliptic Curve include the point addition, doubling, and multiplication.

Point Addition: assuming, there are 2 points say P = (x , y) and P = (x , y) where P is not equal to P . Adding these 2 points result to R:= (x , y) which relies on the same curve where:

$$y \; - y$$

$$\overline{\qquad\qquad} \quad S = \ (\mathrm{mod}\ p)$$

$$7\ x \qquad\quad - x$$

$$x \;\; = s \qquad - (x \quad + x\ )(\mathrm{mod}\ p) \qquad 8$$

$$y \ = s(x \quad - x\ ) - y \ (\mathrm{mod}\ p) \qquad 9$$

Point Doubling: When P = P , the condition is known as Point Doubling. Hence, R can be defined as:

$$S = \frac{3x \ + a}{} \ (\mathrm{mod}\ p) \qquad 10\ 2y$$

The formula for x and y co-ordinates remains the same as the point addition.

Point Multiplication: The point multiplication (performing k times P) is defined by the repetitive addition of point P with itself for k times. The algorithm computes the point multiplication in log(k) time complexity, efficiently allowing computation for large values of k.

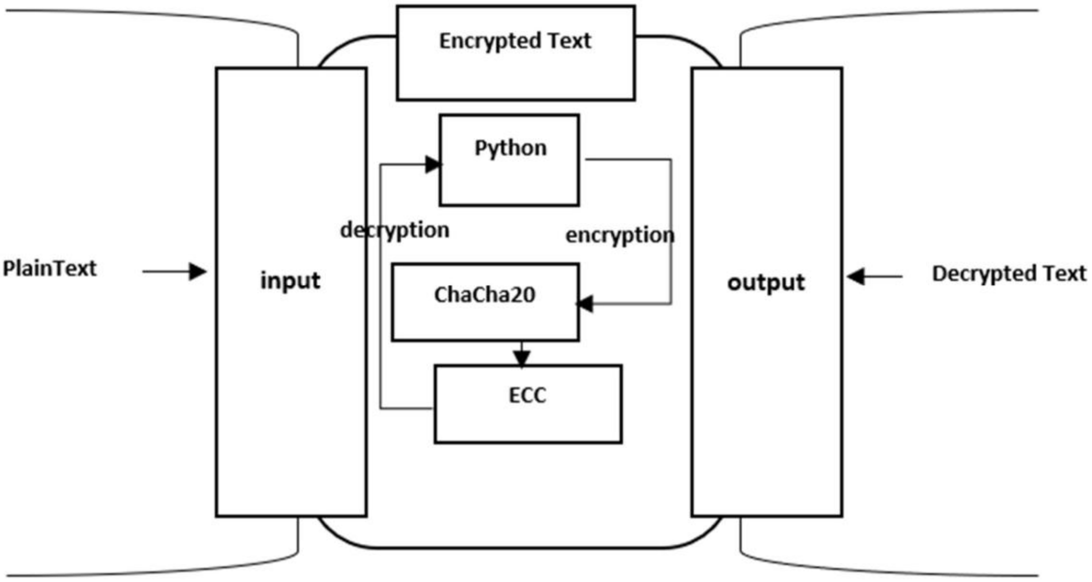
$$R = kP = P + P + P + \cdots + P \text{ (k times)} \qquad 11$$

An effective algorithm to solve point multiplication can be shown as an example:

$$R = 15P = 2(2(2P + P) + P) + P \qquad 12$$

System Architecture

The system architecture as shown on Figure 3.2 depicts how the input data considered as the transmittable information are being processed at the middle layer via the two-layer encryption and decryption techniques using the ECC and Chacha20 cryptography before the output (as an encrypted information) is transmitted over the cloud to the intended user. On the other end (i.e., the receiver end), the reverse process of encryption (information decryption) is performed via the proposed platform.



Evaluation Metric

To evaluate the performance of the cryptographic algorithms particularly, Chacha20, and ECC. The study uses following metric:

i. Time: the duration of taken to encrypt and decrypt some information.

ii. File-Size: the size of the file utilized to encrypt and decrypt using the two cryptographic algorithms iii. Throughput: is a measure of the file's sizes per unit execution time of the algorithms for
both encryption and decryption

RESULTS AND DISCUSSION

Experimental Setup

This chapter presents the result and discussion of the hybrid cryptographic algorithm. It is important to note that the result of the cryptographic algorithm (ECC and Chacha20) is presented in tabular and graphical form and are thus evaluated using some performance evaluation metrics namely time, file size, and throughput. The graphics are meant to depict the convergence behavior of the presented cryptographic algorithm.

The experimental setup for hybrid cryptography encompasses the tools and materials utilized for the development of the hybridized cryptographic system. In detail, the programming language utilized is Python programming language. The programming environment used is visual studio code that is built with rich plugins which support the adapted programming language and thus enable the installation of the Py-crypto libraries required for the incorporation of the cryptographic algorithm. The hardware specification utilized includes a window operating system with 4GB of RAM and 2.4GH processor speeds. Table 1 enumerates the hardware and software utilities utilized.

Table 1: Specifications definition

Utilities	Values/Size
Operating System (OS)	Window 11
Programming Language	Python
Processor	Intel Core i5
Processor speed	2.4GHz
Disk	Solid State Disk (SSD)
Programming environment	Visual Studio Code

Parameter Setting

The module parameter settings define the variables for the developed hybrid cryptographic system. The parameters for the Chacha20 and Elliptic Curve Cryptography (ECC) algorithm are presented in Table 2. The nonce as a variable is a random or pseudo-random number used to protect and authenticate communication protocols. The secret key is a randomized stream of characters used to encrypt plaintext a thus decrypt encrypted bytes at the other end. Considering that the blocks of data might not be completed after being segmented, the study used the PKCS7 (Public Key Cryptography Standard) techniques to add streams of bytes to accommodate for the uncompleted block. The PKCS7 pads with equal numbers of padding bytes. Padding is used in a block cipher to fill up the blocks with padding bytes. Unpadding in the other way removes the padded blocks during the course of decryption. To generate the private key for the Elliptic Curve Cryptography (ECC) the "SECP256R1" was utilized. The SECP256R1 is a specific elliptic curve and associated domain parameters selected and recommended by SECG (Standards for Efficient Cryptography Group) and hence uses 256 bits to generate private key characters.

Table 2: parameter setting

Parameters	Value
Nonce	32 bytes string
Secret key	16 bytes string
Pad	PKCS7-128-bit block
Un-Pad	PKCS7-128-bit block
Private-Key	SPECP256R1

Model Architectural Discussion

As aforementioned in chapter three of this research, the architecture of hybridized cryptographic system encompasses the stacking of the cryptographic algorithm such that the result of encryption or decryption from the Chacha20 algorithm becomes an input to the Elliptic Curve Cryptography. Essentially, the concept of layered security entails the encryption of a file using the Chacha20 algorithm and thus encrypting the secret key using the Elliptic Curve Cryptography, the private key from the sender end is kept secret and the encrypted key and file are sent to the recipient. The recipient on the other end uses the public key from the Elliptic Curve Cryptography algorithm to decrypt the secret encrypted key and the decrypted key is used to decrypt the file.

Result Discussion

This module compares the results of the observed (ECC and Chacha20 algorithm) against each other. The metrics used (as proposed in chapter three) for the comparison are the encryption and decryption execution times, file size, and throughput as shown in tables 3 – 4. The comparison of the execution time for both encryption and decryption of sample data sizes for the ECC and Chacha20 algorithm is analyzed in Table 3 below. The file sizes utilized were measured in megabytes and ranged between 1 MB – 1024MB. The file size includes 1MB, 2MB, 5MB, 10MB, 25MB, 50MB, 100 MB, 200MB, 500MB, and 1G. It is important to note that the encryption and decryption time are both measured in seconds.

Furthermore, a hybrid cryptography of the two proposed algorithms was implemented. In this hybridization technique, the plaintext is encrypted using both Chacha20 and ECC algorithms. First, the plaintext is encrypted using Chacha20 encryption, and the resulting ciphertext is then encrypted using ECC encryption. Similarly, during decryption, the ciphertext is decrypted using ECC decryption, and the resulting plaintext is decrypted using Chacha20 decryption. Essentially, the hybrid encryption and decryption algorithm combine the strengths of both Chacha20 and ECC encryption, providing an added layer of security.

Table 3: Execution time comparison

File Size	ECC		Chacha 20		ECC-CHACHA	
(MB)	EN*	DE*	EN*	DE*	EN*	DE*
1	0.016	0.006	0.116	0.015	0.014	0.013
2	0.017	0.008	0.120	0.021	0.016	0.016
5	0.024	0.032	0.119	0.016	0.021	0.029
10	0.064	0.056	0.141	0.034	0.060	0.050
25	0.144	0.126	0.181	0.074	0.133	0.129
50	0.246	0.295	0.235	0.103	0.235	0.133

100	0.627	0.658	0.302	0.223	0.321	0.223
200	1.573	1.430	0.981	0.632	0.625	0.502
500	1.311	1.320	0.845	0.549	1.042	1.028
1024	1.756	1.679	1.549	1.676	2.195	1.775

Observation

Chacha20 which is a symmetric algorithm, is fast in encryption and decryption time while Elliptic Curve Cryptography, takes a little more time in encrypting and decrypting time because it is more secure compared to Chacha20. Combining the efficiency of Chacha20 and the security of Elliptic Curve Cryptography have made the result of hybridizing ECC and CHACHA efficacious. Figure 12 shows the performance of the ECC algorithm with the encryption and decryption time displayed on the y-axis and the file size on the x-axis. The blue bar chart shows the encryption for the ranging file size and time. Furthermore, Figure 13 shows the line graph of the ECC encryption and decryption time in seconds, the x-axis also shows the file size in MB with an orange line representing the encryption and the blue identifying the decryption time against the file size.

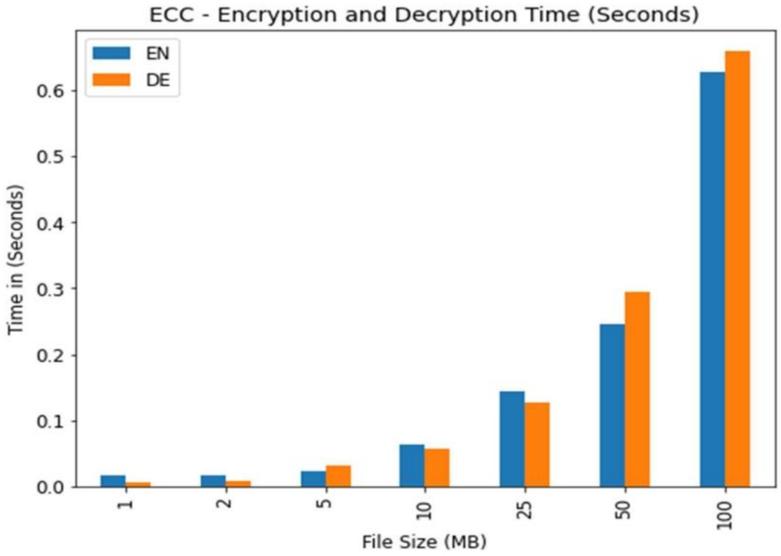


Figure 12: ECC encryption and decryption bar chart

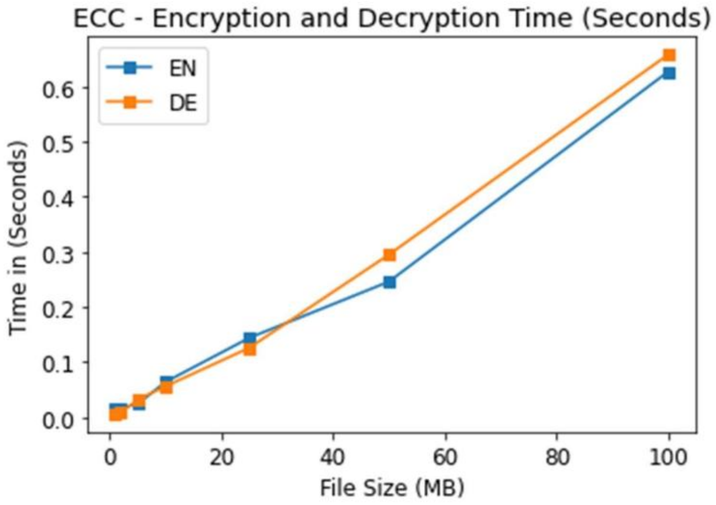


Figure 13: ECC encryption and decryption line chart

Figure 14 and 15 also shows the performance of the ChaCha-20 algorithm with the encryption and decryption time displayed on the y-axis and the file size on the x-axis for the bar chart diagram and the line chart diagram. The blue bar chart shows the encryption for the ranging file size and time while the oranges depict the decryption convergences.

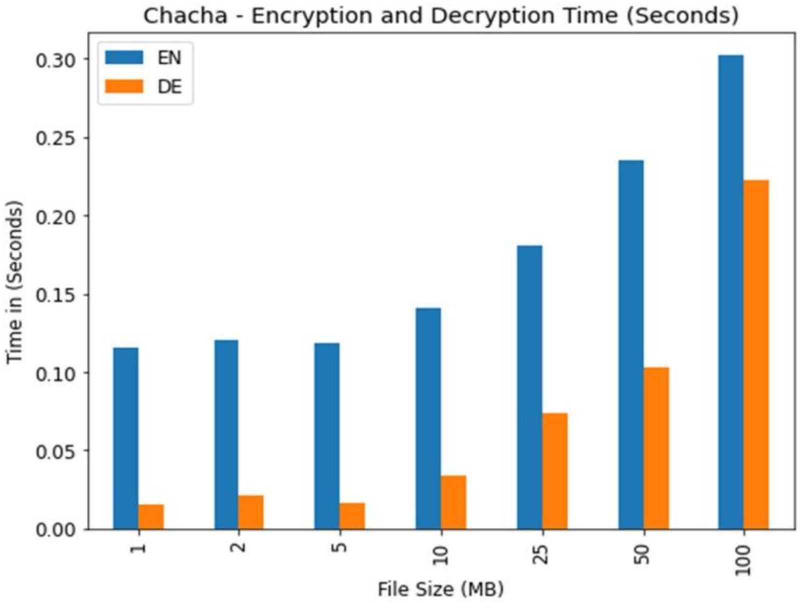


Figure 14: Chacha encryption and decryption bar chart

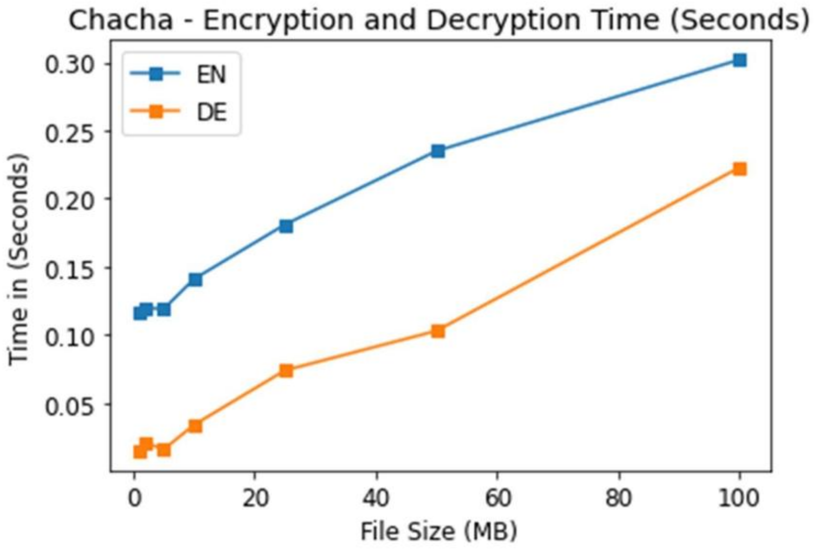


Figure 15: Chacha encryption and decryption line chart

Considering that the model was hybridized, a graphical plot of the hybrid of Chacha20 and ECC is presented in Figures 16 and 17. The performance is also shown based on the encryption and decryption time of the hybridized algorithm. Thus, the encryption and decryption time are displayed on the y-axis and the file size on the x-axis for the bar chart diagram and the line chart diagram just like the ECC and Chacha20 algorithm. The blue bar chart shows the encryption for the ranging file size and time while the oranges depict the decryption convergences.

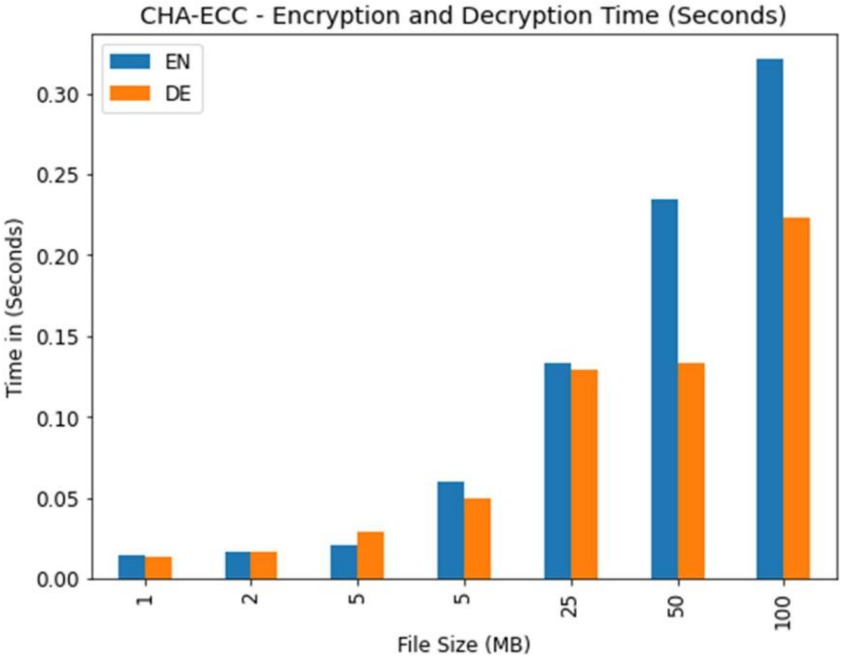


Figure 16: ECC-Chacha encryption and decryption bar chart

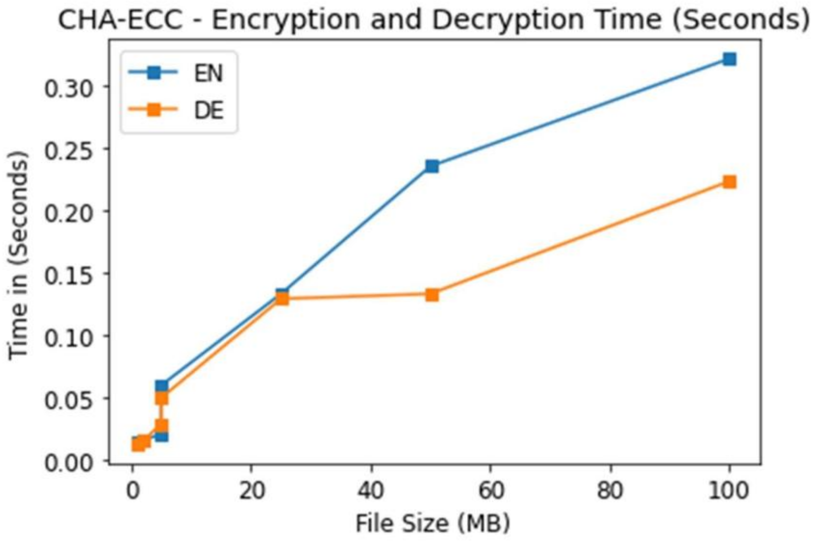


Figure 17: ECC-Chacha encryption and decryption line chart.

Throughput

This module compares the results of the observed (ECC and Chacha20 algorithm) against each other with respect to their throughput. The metrics throughput is a measure of the file's sizes per unit execution time of the algorithms for both encryption and decryption. The file size and the encryption and decryption time are captured from the result of Table 4. the metric for evaluation is defined as:

$$\text{throughput} = t / e$$

Where t is the file size (MB) and e is the execution time.

Table 4: Throughput

		Throughput (MB)	
File Size (MB)	ECC	Chacha 20	ECC-Chacha
1	62.5	8.621	71.43
2	117.65	16.67	125.00
5	208.33	42.02	238.10
10	156.25	70.92	16.67
25	173.61	138.12	187.97

50	203.25	212.77	212.77
100	159.49	331.13311	.53

Comparative Analysis

This section compares the implemented cryptographic algorithms against some of the state-of-the-art algorithms implemented by two distinct authors. During the computational analysis, the computational time of the state-of-the-art cryptography algorithm is also classified as encryption/decryption time. For the comparisons, the length of the file size was the same for all the cryptographic algorithms The stateof-the-art algorithm comparison is shown in Table 4 and Table 5 below.

Current Study Execution Time (Seconds)							(Chinnasammy and Deepalakshimi, 2018) Execution Time (Seconds)					
File Size (MB)	ECC		Chacha 20		ECC-Chacha		AES		BLOWFISH			AES-BLOWFISH
	EN*	DE*	EN*	DE*	EN*	DE*	EN*	DE*	EN*	DE*	EN*	DE*
1 0.006	0.016		0.116	0.015	0.014	0.013	0.294	0.032	0.299	0.041	0.247	0.032
3	0.020	0.023	0.187	0.016	0.016	0.016	0.311	0.047	0.314	0.055	0.278	0.047
5 7	0.024	0.032	0.119	0.016	0.021	0.029	0.320	0.052	0.351	0.094	0.309	0.048
	0.033	0.046	0125	0.019	0.025	0.040	0.326	0.063	0.384	0.119	0.316	0.050
9	0.063	0.065	0.109	0.026	0.058	0.045	0.340	0.078	0.399	0.140	0.325	0.064
10	0.064	0.056	0.114	0.031	0.060	0.050	0.353	0.083	0.435	0.174	0.342	0.072

Table 5 : State of the Art Comparison with the Work of (Chinnasammy and Deepalakshimi, 2018)

To evaluate the performance of the developed cryptographic algorithm. The performance of the implemented ECC, CHACHA20, and the hybrid of the two algorithms was compared with the work of Chinnasammy and Deepalakshimi, (2018) that implemented two based algorithms namely the AES and Blowfish which in their work was compared with the proposed hybrid algorithm. The comparison was made based on some varying input data file sizes (in megabytes) in terms of running time for both the encryption and decryption processes (seconds). The file sizes range from 1, 3, 5, 7, 9, and 10MB.

Observations

From the encryption and decryption time of the different input data sizes measured (1, 3, 5, 7, 9, and 10MB) given in the table above, it is observed that the proposed methods of ECC, CHACHA20, and the hybrid of the two have less encryption and decryption time compared to the implementation of AES and Blowfish and the hybrid (Chinnasammy & Deepalakshimi, 2018).

Table 6: State-of-the-Art Comparison with the Work of (Ali, Tariq, and Zaid, 2021)

	Current Study Execution Time (Seconds)						(Ali, Tariq, and Zaid, 2021) Execution Time (Seconds)							
File Size		ECC	Chacha			ECC-Chacha	AES		BLOWFISH		MD5		Hybrid	
(MB)	EN*	DE*	EN*	DE*	EN*	DE*	EN*	DE*	EN*	DE*	EN*	DE*	EN*	DE*
1	0.016	0.006	0.116	0.015	0.014	0.013	1.263	1.001	1.225	0.967	1.213	0.998	1.24	0.97
3	0.020	0.023	0.187	0.016	0.016	0.016	1.28	1.016	1.24	0.981	1.244	1.013	1.259	0.99
5	0.024	0.032	0.119	0.016	0.021	0.029	1.289	1.021	1.277	1.02	1.275	1.014	1.26	1.001
7	0.033	0.046	0125	0.019	0.025	0.040	1.295	1.032	1.31	1.045	1.282	1.016	1.27	1.012
9	0.063	0.065	0.109	0.026	0.058	0.045	1.309	1.047	1.325	1.066	1.291	1.03	1.29	1.028
10	0.064	0.056	0.114	0.031	0.060	0.050	1.322	1.052	1.361	1.100	1.308	1.038	1.30	1.036

In continuation of the developed cryptographic algorithm comparison to some state-of-the-art algorithm. The performance of the implemented ECC, CHACHA20, and the hybrid of the two algorithms was compared with the work of Ali, Tariq, and Zaid, 2021 that implemented three based algorithms namely the AES, Blowfish, and MD5 which also in their work, the three algorithm was compared with the hybrid of the three proposed algorithm. The comparison was made based on some varying input data file sizes (in megabytes) in terms of running time for both the encryption and decryption processes (seconds). The file sizes range from 1, 3, 5, 7, 9, and 10MB.

Observations

From the encryption and decryption time of the different input data sizes measured (1, 3, 5, 7, 9, and 10MB) given in the table above, it is also observed that the proposed methods of ECC, CHACHA20, and the hybrid (i.e., the algorithm from the current work) of the two have less encryption and decryption time compared to the implementation of AES, Blowfish, MD5 and the hybrid.

References

[1] Kagita, M. K., Thilakarathne, N., Gadekallu, T. R., Maddikunta, P. K. R., & Singh, S. (2021). A review on cyber-crimes on the Internet of Things. Deep Learning for Security and Privacy Preservation in IoT, 83-98.

[2] Javed, M. K., & Javaid, S. J. D. A. (2020). Corona Virus Awareness in Pakistan: A Case Study. International Journal of Medical Science in Clinical Research and Review, 3(03,), 256-2 Jyoti, T., & Pandi, G. (2017). Achieving cloud security using hybrid cryptography algorithm. International Journal of Advance Research and Innovative Ideas in Education (IJARIIE), 3(5), 1518-1523.

[3] Deora, R. S., & Chudasama, D. (2021). A brief study of cybercrime on the internet. Journal of Communication Engineering & Systems, 11(1), 1-6.

[4] Nayyar, A. N. A. N. D., Rameshwar, R. U. D. R. A., & Solanki, A. R. U. N. (2020). Internet of Things (IoT) and the digital business environment: a standpoint inclusive cyber space, cyber crimes, and cybersecurity. The evolution of business in the cyber age, 10, 9780429276484-6.

[5] Subedar, Z., & Araballi, A., (2020). Hybrid cryptography: Performance analysis of various cryptographic combinations for secure communication. International Journal of Mathematical Sciences and Computing (IJMSC), 6(4), 35-41.

[6] Bhat, S., & Kapoor, V. (2019). Secure and efficient data privacy, authentication, and integrity schemes using hybrid cryptography. In International Conference on Advanced Computing Networking and Informatics, 279-285.

[7] Obaid, T. S. (2020). Study a public key in the RSA algorithm. European Journal of Engineering and Technology Research, 5(4), 395-398.

[8] Agarwal, V., Kaushal, A. K., & Chouhan, L. (2020). A survey on cloud computing security issues and cryptographic techniques. In Social Networking and Computational Intelligence, 119-134.

[9] AbdElminaam, D. S. (2018). Improving the security of cloud computing by building new hybrid cryptography algorithms. International Journal of Electronics and Information Engineering, 8(1), 40-48.

[10] Mitali, V. K., & Sharma, A. (2014). A survey on various cryptography techniques. International Journal of Emerging Trends & Technology in Computer Science (IJETTCS), 3(4), 307-312.

[11] Mallouli, F., Hellal, A., Saeed, N. S., & Alzahrani, F. A. (2019). A survey on cryptography: a comparative study between RSA vs ECC algorithms, and RSA vs El-Gamal algorithms. In 2019 6th IEEE International Conference on Cyber Security and Cloud Computing (CSCloud), 173-176.

[12] Sharma, T. (2018, January). Proposed hybrid RSA algorithm for cloud computing. In 2018 2nd international conference on inventive systems and control (ICISC) (pp. 60-64). IEEE.

[13] Kumar, M.A. & Karthikeyan, S., (2016). 2012. Investigating the efficiency of Blowfish and Rejindael (AES) Algorithms. International Journal of Computer Network and Information Security, 4(2), 22.

[14] Mahalle, V.S. & Shahade, A.K., (2014). Enhancing the data security in Cloud by implementing hybrid

(Rsa & Aes) encryption algorithm. In 2014 International Conference on Power, Automation and

Communication (INPAC), 146-149.

[15] Bhandari, A., Gupta, A. & Das, D., (2016), January. Secure algorithm for cloud computing and its applications. In 2016 6th International Conference-Cloud System and Big Data Engineering (Confluence) 188-192.

[16] Timilsina, S., & Gautam, S., (2019). Analysis of Hybrid Cryptosystem Developed Using Blowfish and ECC with Different Key Size. Technical Journal, 1(1), 10-15.

[17]Bala, B., Kamboj, L. & Luthra, P., (2018). Secure File Storage in Cloud Computing Using Hybrid Cryptography Algorithm. International Journal of Advanced Research in Computer Science, 9(2).

[18] Taha, Ali Abdulridha, Diaa Salama AbdElminaam, & Khalid M. Hosny., (2017). "NHCA: Developing New Hybrid Cryptography Algorithm for Cloud Computing Environment." (IJACSA)

[19] Chennam, K.K., Muddana, L. & Aluvalu, R.K., (2017). May. Performance analysis of various encryption algorithms for usage in multistage encryption for securing data in Cloud. In 2017 2nd IEEE

International Conference on Recent Trends in Electronics, Information & Communication Technology (RTEICT), 2030-2033.

[20] Jakimoski, K.(2016). Security techniques for data protection in cloud computing. International Journal of Grid and Distributed Computing, 9(1), 49-56.

[21] Bansal, V. P., & Singh, S. (2015). A hybrid data encryption technique using RSA and Blowfish for cloud computing on FPGAs. In 2015 2nd International Conference on Recent Advances in Engineering & Computational Sciences (RAECS) 1-5.

[22] Maitri PV & Verma A (2016). Secure file storage in cloud computing using hybrid

cryptographyalgorithm. In: 2016 international conference on wireless communications, signal processing and networking (WiSPNET), 1635–1638

[23] Chinnasamy P, & Deepalakshmi P (2018) A scalable multilabel-based access control as a service for the cloud (SMBACaaS). Trans Emerg Telecommun Technol 29(8):e3458. https://doi.org/10.1002/ett.3458,2018

[24] Mazrekaj, A, Shabani, I. & Sejdiu, B (2016). Pricing schemes in cloud computing: an overview.

International Journal of Advanced Computer Science and Applications, 7(2), 80-86.

[25] Dubey AK, Namdev M, & Shrivastava SS (2012) Cloud-user security based on RSA and MD5 algorithm for resource attestation and sharing in Java environment. In: CSI sixth International Conference, Software Engineering (CONSEG)

[26] Sarkar M.K. & Kumar S (2016) Ensuring data storage security in cloud computing based onhybrid encryption schemes. In: Fourth International Conference on Parallel, Distributed and Grid Computing (Pdgc), Waknaghat, 320–325. https://doi.org/10.1109/pdgc.2016.7913169

[27] Yong P, Wei Z, Feng X, Zhong-hua D, Yang G, & Dongqing C (2012) A secure cloud storage-based on cryptographic techniques. J China Univ Posts Telecommun 19:182–189.

cyber crimes, and cybersecurity. In The Evolution of Business in the Cyber Age, 111-152.

[28] Singh N., & Kaur P.D., (2015) A hybrid approach for encrypting data on cloud to prevent DoSattacks. Int J Database Theor Appl 8(3):145–154. http://dx.doi.org/10.14257/ijdta.2015.8.3.12

[29] Akomolafe, O. P., & Abodunrin, M. O. (2017). A hybrid cryptographic model for data storage in mobile cloud computing. International Journal of Computer Network and Information Security, 9(6), 53.

[30] Karthik, Chinnasamy, & Deepalakshmi (2017) Hybrid cryptographic technique using OTP:RSA.In:

2017 IEEE international conference on intelligent techniques in control, optimization and signal processing (INCOS), Srivilliputhur, 1–4.

[31] Rahmani H, Sundararajan E, Zulkarnain Md, & Ali AMZ (2013) Encryption as a service (EaaS) as a solution for cryptography in cloud. Procedia Technol 11:1202–1210

[32] Visconti, P., Capoccia, S., Venere, E., Velázquez, R., & Fazio, R. D. (2020). 10 Clock-Periods Pipelined Implementation of AES-128 Encryption-Decryption Algorithm up to 28 Gbit/s Real Throughput by Xilinx Zynq UltraScale+ MPSoC ZCU102 Platform. Electronics, 9(10), 166.

[33] Cai, W., Chen, H., Wang, Z., & Zhang, X. (2022). Implementation and optimization of ChaCha20 stream cipher on Sunway taihu Light supercomputer. The Journal of Supercomputing, 78(3), 4199-4216.

[34] Mhatre, V. S., & Patel, S. N. S.(2023) Secure File Storage On Cloud Using Elliptic Curve Cryptography (Ecc) Algorithm. International Research Journal of Modernization in Engineering Technology and Science, 5(1). 1182-1184