Journal of Computer Engineering & Information Technology

Research Article A SCITECHNOL JOURNAL

A McEliece Cryptosystem for Securing Cloud Data

Henry Chima Ukwuoma*, Arome Gabriel, Aderonke Thompson and Boniface K Alese

Department of Cyber Security, Federal University of Technology, Akure, Nigeria

***Corresponding author:** Henry Chima Ukwuoma, Department of Cyber Security, Federal University of Technology, Akure, Nigeria; E-mail: henry@nipsskuru.gov.ng

Received date: 16 November, 2021, Manuscript No. JCEIT-21-47548;

Editor assigned date: 19 November, 2021, PreQC No. JCEIT-21-47548 (PQ);

Reviewed date: 03 December, 2021, QC No. JCEIT-21-47548;

Revised date: 03 March 2023, Manuscript No. JCEIT-21-47548 (R);

Published date: 31 March 2023, DOI: 10.4172/2324-9307.1000260

Abstract

With the evolvement of quantum computers, most crypto-systems will be rendered susceptible to attack and obsolete since, the safety and security of some cryptosystems is subject to the hardness of the integer factorization problem and discrete logarithm problem. This paper reviews the code-based cryptography (McEliece cryptosystem) and proposes a variant of the McEliece cryptosystem for data security in the cloud. The research simulates the proposed variant alongside other cryptosystems to determine the execution time in terms of key generation, encryption and decryption processes, and with a view to present the pros and cons of the proposed cryptosystem. A variant of McEliece cryptosystem is proposed with a goal to enhance the security of the cryptosystem that can resist attack amidst classical and quantum computing. The simulation revealed that the proposed McEliece cryptosystem has a better time complexity compared to the existing McEliece cryptosystem. The novel distortion of the parameters S and P also strengthens the security of the proposed system.

Keywords: Code based cryptography; McEliece cryptosystem; Public key cryptosystems; Private key cryptography; Quantum computing; Cloud computing

Introduction

Recently, there has been a progressive acceptance of cloud services both in the private and public sectors around the world. This can be attributed to the gains of cloud computing, which is presented to cloud users by cloud providers, they include scalability, elasticity, powerful computations and reduced cost of services. Although, the security of data is one of the obstacles hindering the widespread acceptance of cloud services. Data security can be defined as a means of safeguarding digital data for example, data stored in a database, from attackers and uninvited actions of unauthorized users [1].

Consequently, in this present technological age, storing, distribution and transmission of data have increased tremendously. Generally, information exchange is carried out by means of open channels, which can make it susceptible to interception. The threat of a trespasser accessing surreptitious information has been a continuing concern for the Information Technology (IT) industry, which has resulted to cloud providers and clients adopting various techniques such as encryption in securing cloud data. Encryption can be traced to ancient times but was then perceived as secret communication. However, has evolved in practice in areas such as electronic voting, digital currency and digital signatures. Although, encryption is used to secure data but theft of confidential data still remains prevalent [2].

The most dominant worrisome issue of cloud computing is secrecy and privacy; which specifically involves the safeguarding of sensitive data from uninvited access/users. Privacy as having prior knowledge of an attack/data breach that is bound to occur and being able to use controls to prevent it/control it from happening [3].

Cryptography plays a vital role in times of data attack; when an intruder intercepts data, the data is protected because it has been encoded, so the intruder is unable to understand what he/she has intercepted. It is pertinent to note that with the advent of quantum computing, some of the cryptosystems will no longer be safe since the technology behind quantum computing will render some of the cryptosystems unsafe/obsolete. More so, advancements have been made in developing quantum-computing technologies. Consequently, the emergence of quantum computers will change the world once again due to its high construction costs and maintenance costs, the first quantum computer is likely to be only owned by a small number of organizations. Fortunately, with the use cloud service, users can apply to use quantum computers for test running of their applications or for any other similar use. The threat of quantum computer assisted cryptanalysis is compelling the security community to develop novel types of security protocols/algorithms. These solutions must be protected against classical and post-quantum cryptanalysis techniques as well as adaptable for all kinds of devices. Post-quantum cryptography research is predominantly focused on different approaches, which include lattice based cryptography, multivariate cryptography, hash-based cryptography, code-based cryptography and singular elliptic curve isogeny cryptography. This paper will focus on code based cryptography (McEliece) [4].

Similarly, the McEliece cryptosystem has proven to be the outright replacement for RSA, since RSA will be rendered obsolete with the adoption of quantum computing. Most importantly, McEliece cryptography is not known to be broken by a quantum computer and even with its large key size, it has proven to be most efficient. However, research has shown that the key sizes of the McEliece cryptosystem is larger than that for RSA, it is because of this reason that the McEliece cryptosystem has not been practicable in securing the cloud. This might change since quantum computers are yet to dominate the data computing era and may also result to the cryptanalysis of the McEliece cryptosystem, thus revealing its weak points [5].

With the increased dominance of cloud computing, cloud data encryption has become an important machinery to safeguard data against attackers (hackers). Usually, the information exchange is carried out through channels that are not secured enough, which can make it vulnerable to interception. Cloud computing promises security, reduced cost, flexibility, improved time to market and higher availability. Although, studies carried out and still on going in the area of cloud computing data security shows that various types of frameworks have been developed and deployed for modeling the security of cloud data from the initial stage of data upload, data processing to the final stage of data storage/download. Cryptosystems

such as symmetry and asymmetric encryptions have been deployed, yet data theft prevails. This paper proffers an enhanced way of securing and accessing data in the cloud by proposing and applying a variant of the McEliece cryptosystem [6].

The following literatures were reviewed: The paper presents literatures reviewed on various algorithms adopted/applied for the security of cloud data. Literatures were sourced from IEEE articles web search, springer, science direct and google scholar [7].

The users have limited control over cloud stored data, thus require strong and substantial defense techniques to protect their cloud data. The authors recommended the encryption of cloud data before cloud storage and suggested the application of a hybrid technique; they adopted a lattice-based security technique. Furthermore, they recommended and experimented a framework for the examination of roles and responsibilities using a lattice model. The proposed model was applied using RSA and AES. They proffered a novel application of a hybrid lattice-based access control in safeguarding data. However, the idea of the application of RSA cryptosystem is not quantum safe.

A data security framework with the aim of proffering a double layer security for multimedia data communication and storage. The authors proffered the usage of AES cryptosystem for the first level of security and its encrypted data as and input to the bluefish cryptosystem as a second layer, which uses a 94 bit converter and a random key generator produced the cipher text that is registered in the cloud. The process for decryption was a complete reversal of the encryption process. The framework produced a double layer security framework; however, the AES algorithm has easy-to-guess key operations while the blowfish cryptosystem is susceptible to plaintext attacks. Furthermore, the application of two different symmetric algorithms in a system for encryption and decryption implies that more time is taken for the execution of encryption and decryption.

The academia and industry were concerned with the privacy and security issues that emanated for the increasingly prominent evolvement and development of cloud computing. The author reviewed various literatures on privacy security in cloud computing with the sole aim of identifying challenges on proffered protection techniques and thereafter proposed a novel privacy security protection model. The proposed framework was categorized into three segments, which consist of principle and model, methodology; and design and application. Although, the author proffered a multi-layer data security framework and suggested the building of a secure network infrastructure environment as a foundation for a computing environment, the proposed framework was not experimented, thus cannot be regarded as most efficient.

The use of the McEliece protocol as a means to derive a secure Key Encapsulation Mechanism (KEM). This brings about maximum flexibility for the use of coding theory for data security purposes. This approach provides a strong security as is derived in Niederreiter KEM obtained in Persichetti. The novel construction indicates an alternative for designing quantum cryptographic designs.

The availability of quantum computers and Key Exchange (KE) mechanisms that will be susceptible to quantum attacks. The authors invented the Code-based Algorithm for Key Encapsulation-CAKE; a key encapsulation framework that relies on the McEliece cryptosystem based on Quasi-Cyclic Medium Density Parity Code (QC-MDPC). This methodology ensures an efficient key generation for McEliece QC-MDPC schemes and the use of temporarily keys for its operation. Then, they presented CAKE-an authenticated key exchange protocol, which can be said to be of Internet Key Exchange (IKE) standard. The practical parameters suggested by the framework can be used to attest that code-based Algorithm for key encapsulation is IND-CPA secure and the proposed protocol SK-secure. The limitation of the proposed framework is that there is cost in the key generation process and there is also a possibility of decoding failure [8].

A model using role-based security model as a guide, which the authors referred to as Onibere Odunayo Security-4 (OOS-4). The model was intended for Human Resource Information System (HRIS) in a cloud platform and expected to enhance the interrelationship between varying components of the human resource management organization with adequate security. The authors adopted the cloud computing architecture (platform as a service) and implemented the model through the google app engine. The proposed application encrypted its data in storage (google cloud platform). The model applied an algorithm (NAF encryption algorithm), which is based on simple substitution ciphers an enhancement of ROT-13. The limitation of the approach is the that simple substitution ciphers is an example of weak encryption and practically provides no cryptographic security.

The Quasi-Cyclic Moderate-Density Parity-Check (QC-MDPC) codes to find out if it is a good family of error-correcting codes to replace the family of goppa codes in the McEliece cryptosystem. The author proposed a cryptosystem based on techniques used in coding theory. The limitation of this framework is that despite allowing for the rapid encryption and decryption of messages, storage of the users' cryptographic key requires a large amount of memory, because McEliece cryptosystem generates large keys.

A modification of the McEliece cryptosystem that ensures that the coding used for the public key and public key generation cannot be deduced from the secret code. The rationale behind this methodology is to ensure that the security level of the public key is maximized thereby creating a research gap for the use of classical families of codes such as the reed-solomon codes, that have been exempted from the McEliece cryptosystem for security reasons. Research has shown that codes of these classes produce a decrease in the key size and an increased level of information set decoding, which are the major advantages of the proposed mechanism. They also proposed and described possible susceptibilities/attacks related to the proposed mechanism and proferred design choices that could counter such attacks. The limitation of the approach is that research has shown that goppa codes are the most secure when it comes to McEliece cryptosystem any other could lead to a security breach of the entire system [9].

The McEliece Cryptosystem (MECS) and suggested it as one of the oldest cryptosystems that is post-quantum secure. The authors further surveyed proposed modifications, implementation issues and the security of McEliece cryptosystems and its modifications by various authors. Their research involved a review of the application of implementation challenges of MECS, structural attacks, the decoding problems and parameter selection of the MECS and was based on the irreducible binary goppa codes. They also reviewed numerous suggestions that use alternative codes for MECS and pointed out some attacks on the modified systems. Lastly, they reviewed existing implementations on how MECS can be applied on low-resource platforms.

RSA (Rivest-Shamir-Adleman) algorithm is one of the commonly used efficient cryptosystems. They suggested that RSA offers data confidentiality, data integrity and privacy required to stand the test of

pre-quantum era. In other to reduce intrusion and maximize security, they proposed the use of the RSA cryptosystem and the round-robin priority scheduling scheme with the sole aim of attaining the smallest overhead and improved throughput and privacy. In the proposed methodology, the RSA cryptosystem generates encrypted messages that are sorted in a priority-wise manner and then sent. The receiver, then decrypts the messages on reception using the RSA algorithm and its message priority. This method arrests the risk of man-in-middle attacks and timing attacks since both the encrypted and decrypted messages are sent and received in a random order with their priority. The proposed methodology reduces the risk of power monitoring attack during information exchange. This ensures efficiency and information security. The limitation of this method is that it is complex and round robin scheduling algorithm yields greater average waiting time, which implies that the approach will consume more time in execution (that is during encryption and decryption).

A new methodology, which highlights on enhanced encryption and protocol security for public key encryption. The proposed approach provides relevant solution for enhanced encryption algorithms and also provides a better security to email services and web services. With the advent and evolvement of the internet, this method proposes an effective way of sending messages *via* the internet and any other network with security as its major consideration. The approach enhances security with the addition of security codes to the Diffie-Hellman cryptosystems. The limitation of the work is that DH uses the same key for the entirety of encryption and decryption process. This has a security flaw because once the hacker is able to hijack one of the keys, the entire system is comprised.

A hybrid approach by using the combination DSA, RSA and MD5 cryptosystems as a hybrid link for wireless devices. The mobiles are more secured with the use of the proposed approach and tend to be very efficient. The proposed hybrid approach was subjected to various test environments with results of improved response time, less network delay and enhanced throughput. The hybrid approach showed better results than other algorithms/approach. This approach can be applied to the routing of packets with lesser loads on servers and mobile nodes for privacy purposes. The limitation of the approach is that it is not a practicable approach (that is not applicable if there is much load on servers).

The McEliece cryptosystem with the view to address the problem of parameter selection for the McEliece cryptosystem centered on goppa codes by the year 2050. This study sought to find secure instances of McEliece cryptosystem in the years between with their equivalent minute key sizes and parameter recommendation. This informs potential users of the McEliece cryptosystem to improve the parameter choice, thus enhancing the applicability and adoption of code-based cryptography. The analysis presented provides additional perceptions into the strengths and limitations of McEliece Cryptosystem (code-based cryptography), thereby suggesting areas for future research [10].

Materials and Methods

The proposed McEliece will be applied to the data that will be stored in the cloud. Below is a conceptual architecture for the security of data stored in the cloud (Figure 1).

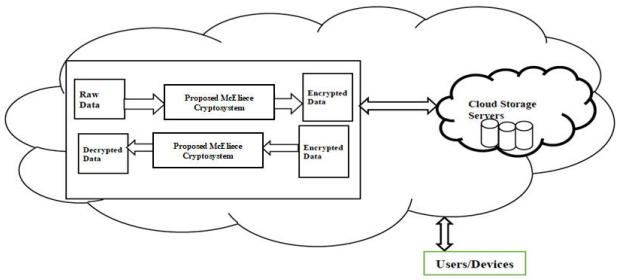

Figure 1: Conceptual architecture for data security in the cloud.

The McEliece algorithm is based on the hardness of deciphering a general linear code. While the public key is generated from the private key, the goppa code conceals the selected code. Two randomly selected invertible matrices S and P; are hidden in the code's generator matrix. Variants of the McEliece cryptosystem have been explored using various types of codes. Majority of them have shown to have loopholes which is a security flaw; structural decoding broke them. Research has shown that the McEliece cryptosystem deployed with goppa codes has resisted cryptanalysis to this point. This study will therefore adopt the existing McEliece cryptosystem algorithm with modifications. These modifications include processes in the key generation algorithm, which produces a public and a private key, the encryption and decryption algorithm. More so, in the process of deploying McEliece cryptosystem, a set of parameters is considered which include: n, k and t. Below is a diagram showing the processes of the modified McEliece cryptosystem (Figure 2).

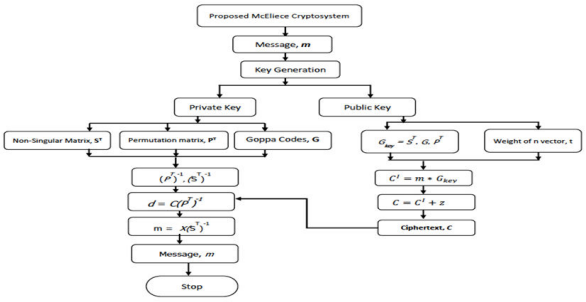

Figure 2: Proposed McEliece cryptosystem.

The following processes are involved in the proposed McEliece cryptosystem:

Proposed key generation

Choose a binary goppa code C; specify the parameters (n; k; t) in particular, according to the security parameter. Obtain a k × n generator matrix G, randomly choose a binary non-singular (k × k) matrix S and a permutation (n × n) matrix P. S must not be a symmetric matrix that is, $S^T \neq S$, S must not be an orthogonal matrix *i.e.* $S^T \neq S^{-1}$ and $P^T \neq P$. These novel criteria ensure the security of the cryptosystem is guaranteed and difficult to break.

$$\text{Compute key} = S^T . G . P^T \quad (1)$$

Where T represents transposes

The public key is set to $P_k=(G_{key}, t)$ and private key $S_K=(S^T, G, P^T)$

Proposed encryption

The message m is encoded as a binary string of length k.

$$\text{Compute } C^I = mG_{key} \quad (2)$$

A random n-bit vector z comprising exactly t ones (a vector of length k and weight t) is generated.

Compute the cipher text as

$$C = C^I + z \quad (3)$$

Proposed decryption

$$\text{Compute the inverse of PT } i.e.\ (P^T)^{-1} \quad (4)$$

$$\text{Compute } d = C\,(P^T)^{-1} \quad (5)$$

Use the decoding algorithm for the code C to decode d to X.

$$\text{Compute } m = X\,(S^T)^{-1} \quad (6)$$

Example of implementation for the proposed McEliece cryptosystem

- Let $S=\begin{pmatrix}1 & 1\\ 0 & 1\end{pmatrix}$; $P=\begin{pmatrix}0&0&0&1&0\\0&0&0&0&1\\1&0&0&0&0\\0&1&0&0&0\\0&0&1&0&0\end{pmatrix}$; $G=\begin{pmatrix}1&0&1&1&1\\0&0&0&1&1\end{pmatrix}$ and $H=\begin{pmatrix}1&1&1&0&0\\1&1&0&1&0\\1&0&0&0&1\end{pmatrix}$

$S^T=\begin{pmatrix}1&0\\1&1\end{pmatrix}$ and $P^T=\begin{pmatrix}0&0&1&0&0\\0&0&0&1&0\\0&0&0&0&1\\1&0&0&0&0\\0&1&0&0&0\end{pmatrix}$ *and* $t = 1$

Process $G_{key} = S^T.G.P^T$ where T represents transpose

So $G_{key}=\begin{pmatrix}1&0\\1&1\end{pmatrix}\begin{pmatrix}1&0&1&1&1\\0&0&0&1&1\end{pmatrix}\begin{pmatrix}0&0&1&0&0\\0&0&0&1&0\\0&0&0&0&1\\1&0&0&0&0\\0&1&0&0&0\end{pmatrix}=\begin{pmatrix}0&1&1&0&1\\1&1&1&1&1\end{pmatrix}$

The Public key is set to $P_k = (G_{key}, t)$ and private key $sk = (S^T, G, P^T)$

Proposed encryption

Message to be encrypted; m= (0 1)

Process $C^I = mG_{key}$

$$C^I = (0\ 1)\begin{pmatrix}0&1&1&0&1\\1&1&1&1&1\end{pmatrix} = (1\ 1\ 1\ 0\ 1)$$

A random *n*-bit vector *z* containing exactly *t* ones (a vector of length n and weight t is generated and then

z = (0 0 0 1 0)

$C = C^I + z$;

$= (1\ 1\ 1\ 0\ 1) + (0\ 0\ 0\ 1\ 0)$

$C = (1\ 1\ 1\ 1\ 1)$ --------- Encrypted Data

Proposed decryption

Process the inverse of P^T i.e. $(P^T)^{-1}$

$$(P^T)^{-1} = \begin{pmatrix}0&0&0&1&0\\0&0&0&0&1\\1&0&0&0&0\\0&1&0&0&0\\0&0&1&0&0\end{pmatrix} \qquad C = (1\ 1\ 1\ 1\ 1)$$

Process $d = C(P^T)^{-1}$

$$d = (1\ 1\ 1\ 1\ 1)\begin{pmatrix}0&0&0&1&0\\0&0&0&0&1\\1&0&0&0&0\\0&1&0&0&0\\0&0&1&0&0\end{pmatrix} = (1\ 1\ 1\ 1\ 1)$$

Using the decoding algorithm for the code *G* to decode *d* to *X*

Syndrome

To calculate the error position;

According to Vaidya and Dutta (2018), the syndrome is computed as follows:

$S = Hy^T = H(c+e) = He^T$ ----------6

$$He^T = \begin{pmatrix}1&1&1&0&0\\1&1&0&1&0\\1&0&0&0&1\end{pmatrix}\begin{pmatrix}0\\0\\0\\1\\0\end{pmatrix} = \begin{pmatrix}0\\1\\0\end{pmatrix}$$; the error is at the fourth column

thus e= (0 0 0 1 0)

Hence codeword= $y - e$

$= (1\ 1\ 1\ 1\ 1) - (0\ 0\ 0\ 1\ 0) = (1\ 1\ 1\ 0\ 1)$; *thus,* $X = (1\ 1)$

But $m = X(S^T)^{-1}$

$m = (1\ 1)\begin{pmatrix}1&0\\-1&1\end{pmatrix} = (0\ 1)$, hence the message

Proposed McEliece algorithm

The steps involved in the key generation, encryption and decryption of the proposed algorithm are depicted below.

Algorithm 1: Proposed McEliece- Key Generation	Algorithm 2: Proposed McEliece - Encryption	Algorithm 3: Proposed McEliece- Decryption
Input: Parameters for Key Generation *(t, S, P, G)*	Input: Parameters for encryption (m, t, G_{key})	Input: Parameters for encryption *(S, P, G)*
Output: Encryption Keys (G_{key}, P_k)	Output: Plain Text (C)	Output: Plain Text *(m)*
Begin	Begin	Begin
i. Compute $G_{key} = S^T.G.P^T$	i. Compute $C^I = m * G_{key}$	i. computes the inverse of P^T i.e. $(P^T)^{-1}$
ii. $P_k = (G_{key}, t)$	ii. A random *n*-bit vector *z*, containing exactly *t* ones (a vector of length *n* and weight *t*, is generated)	ii. Use the decoding algorithm for the code *G* to decode *d* to *X*
iii. Return (G_{key}, P_k)	iii. $C = C^I + z$	iii. $m = X(S^T)^{-1}$
End	iv. Return (C)	iv. Return *(m)*
	End	End

Simulation of the proposed algorithm

Symmetric and asymmetric cryptosystems were selected in order to test run the efficiency of the proposed algorithms. Various data sizes were simulated against four algorithms (RSA, ECC, AES and McEliece) and the proposed algorithm, with the sole aim of computing the throughput of the encryption and decryption execution time of the data used and draw conclusion if the proposed algorithm is a better way of safeguarding cloud data. The simulation was carried out with Mathlab on an intel core i5 MacBook Pro with a RAM size of 8GB and hard disk space of 250GB on MacOS.

The Table 1 depicts the time taken for the generation of private keys for ECC, AES, RSA, existing McEliece and proposed McEliece. Varying data sizes of 20 kb, 77 kb, 153 kb, 283 kb and 305 kb were simulated.

S/no	File size (KB)	ECC (s)	AES (s)	RSA (s)	Existing McEliece	Proposed McEliece
a.	20	0.0001914	0.0001092	0.00000048	0.0017290	0.1088750
b.	77	0.0001933	0.0000970	0.00000095	0.0016661	0.0014391
c.	153	0.0001888	0.0001001	0.00000072	0.0012071	0.0010478
d.	283	0.0001852	0.0000937	0.00000072	0.0013011	0.0015271
e.	305	0.0001800	0.0001142	0.00000072	0.0014410	0.0014920

Table 1: Time taken for the generation of private key.

The Table 2 depicts the time taken for the generation of public keys for ECC, AES, RSA, existing McEliece and proposed McEliece. Varying data sizes of 20 kb, 77 kb, 153 kb, 283 kb and 305 kb were simulated. It should observed that since the AES is a symmetric cryptosystem, it uses a single key (same key) for the entire cryptographic process, thus the timing is same for AES in Tables 1 and 2.

S/no	File size (KB)	ECC (s)	AES (s)	RSA (s)	Existing McEliece	Proposed McEliece
a.	20	0.0002201	0.0001092	0.000002	0.0000839	0.0000173
b.	77	0.0002222	0.0000970	0.0000024	0.0000839	0.0000830
c.	153	0.0002193	0.0001001	0.0000026	0.0000830	0.0000811
d.	283	0.0002136	0.0000937	0.0000024	0.0000929	0.0000829
e.	305	0.0002093	0.0001142	0.0000019	0.0000849	0.0000849

Table 2: Time taken for the generation of public key.

Table 3 above depicts the result of the simulation for the proposed McEliece algorithm, existing McEliece, ECC, AES and RSA. The encryption times of the various algorithms are shown above.

S/no	File size (KB)	ECC (s)	AES (s)	RSA (s)	Existing McEliece	Proposed McEliece
a.	20	0.0097306	0.3122094	0.0752760	2.4642720	2.53989196
b.	77	0.0080342	0.2489905	0.7298180	9.9002440	9.98193884
c.	153	0.0119049	0.2537041	0.5513860	19.662984	20.6195660
d.	283	0.0228049	0.2471297	0.0228050	36.125769	35.9424701
e.	305	0.0181229	0.2490108	0.7931773	40.566775	39.8396039

Table 3: Time taken for the encryption process of various cryptosystems.

Table 4 above depicts the decryption times of the various algorithms as shown above. The result of the simulation for the proposed McEliece algorithm, existing McEliece, ECC, AES and RSA are also shown above.

S/no	File size (KB)	ECC (s)	AES (s)	RSA (s)	McEliece	Proposed McEliece
a.	20	0.003159285	0.2414517	0.0820222	15.0428459	15.0795140
b.	77	0.005183935	0.2383301	0.4337738	54.5491571	56.0398781
c.	153	0.009607315	0.2506232	0.7057719	122.292833	112.9349358
d.	283	0.016005754	0.2514098	1.4745889	214.856751	211.8841350
e.	305	0.016952276	0.2779884	1.5585389	228.275308	226.4901719

Table 4: Time taken for the decryption process of various cryptosystems.

Table 5 above depicts the throughput of the various algorithms. The result of the simulation for the proposed McEliece algorithm, existing McEliece, ECC, AES and RSA are also shown above.

	ECC	AES	RSA	Existing McEliece	Proposed McEliece
Private key execution time	0.00018774	0.00010284	7.18E-07	0.00131896	0.00146886
Public key execution time	0.0002169	0.0001029	0.00000226	0.00008572	0.00041218
Encryption execution time	0.0141195	0.2622089	0.43449246	21.7440088	21.7846942
Decryption execution time	0.010181713	0.25196064	0.85093914	127.00379	124.485727
Total execution time	0.02470585	0.51437528	1.28543458	148.748792	146.272302
Throughput (KB/S)	33919.0879	1629.16072	651.919603	5.63365918	5.72906017

Table 5: Total average execution time of various algorithms.

Results and Discussion

The deductions from the simulation carried out are arranged based on the time taken to generate private key, public key, encryption and decryption.

Generation of private key for cryptosystems

The figure below depicts the time taken for the private key generation for the ECC, AES, RSA, the existing McEliece and proposed McEliece with varying data sizes. Figure 3 below shows that the existing McEliece algorithm takes the most time in the generation of a private key.

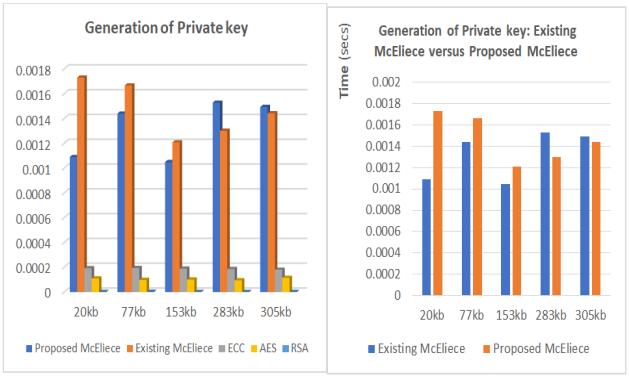

Figure 3: Time taken for the generation of private key.

Furthermore, it can be deduced from Figure 4 below that RSA cryptosystem has the average least time ($7.18e^{-07}$ s) for the generation of the private key, while ECC has the average most time (0.00278781 s) in generating the private key as shown in Figure 4 below. Thus, it can be stated that RSA cryptosystem has the best time for private key generation when compared to the above stated cryptosystems. Additionally, comparing the existing McEliece cryptosystem and the proposed McEliece (Figure 4), it can be deduced from the simulation results that the proposed McEliece cryptosystem takes lesser time in generation of the private key, making it more efficient in time consumption when generating private keys and compared to the existing McEliece.

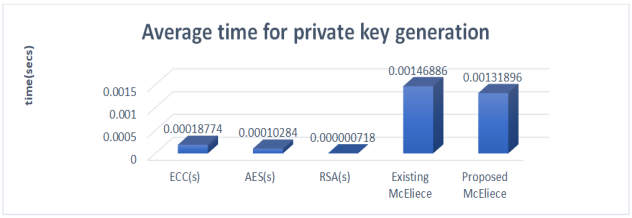

Figure 4: Average time taken for private key generation.

Generation of public keys for cryptosystems

The time taken in the generation of public key for ECC, AES, RSA, the existing McEliece and proposed McEliece with varying data sizes are shown below. Figure 5 below shows that the existing McEliece algorithm takes the most time in the generation of a public key.

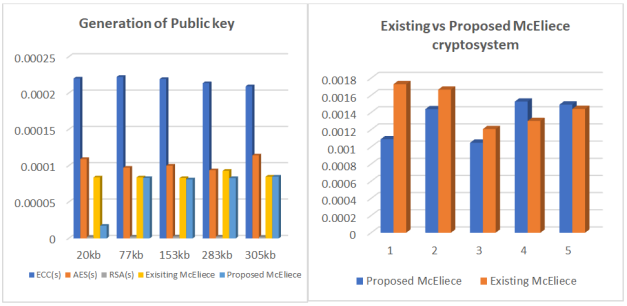

Figure 5: Time taken for the generation of public keys.

More so, it can be inferred from Figure 6 below that the RSA cryptosystem has the least time (average, 0.00000226 s) for the generation of the public key while ECC has the most time (average, 0.0002169 s) in generating the public key as shown in Figure 5 below. Thus, it can be deduced that RSA cryptosystem has the best time for public key generation when compared to the other cryptosystems.

Also, comparing the existing McEliece cryptosystem and the proposed McEliece (Figure 6), it can be deduced from the simulation results that the proposed McEliece cryptosystem takes lesser time (0.00006984 s) in generation of the public key, making it more efficient in terms of time consumption when generating public keys as to when compared to the existing McEliece [11].

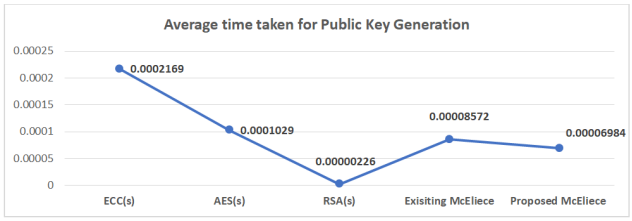

Figure 6: Average time taken for the generation of public key.

Time taken for the encryption of data

The Figure below illustrates the time taken in the encryption process of ECC, AES, RSA, the existing McEliece and proposed McEliece with varying data sizes. Figure 7 above shows that the existing McEliece algorithm takes the most time in the encryption of data.

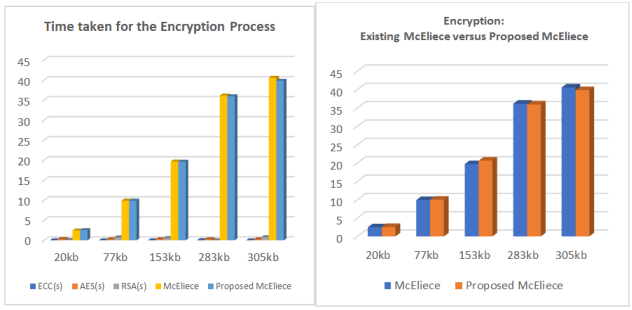

Figure 7: Time taken for the encryption process of various algorithms.

Additionally, it can be gathered from figure 7 that the ECC cryptosystem has the least time (average, 0.0141195 s) for the encryption of data, while the proposed McEliece algorithm has the average most time (21.78 s) in the encryption process as shown in Figure 6 below. Thus, it can be deduced that ECC cryptosystem has the best time for data encryption when compared to the other cryptosystems. Also, comparing the existing McEliece cryptosystem and the proposed McEliece (Figure 8), it can be deduced from the simulation results that the existing McEliece cryptosystem takes lesser time (21.74 s) in encryption of data, making it more efficient in terms of time consumption when compared to the proposed McEliece.

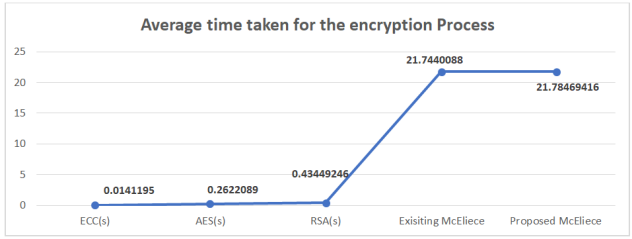

Figure 8: Average time taken for the encryption process of various algorithms.

Time taken for the decryption of data

The Figure below illustrates the time taken in the decryption processes of ECC, AES, RSA, the existing McEliece and proposed McEliece against varying data sizes. Figure 9 below shows that the existing McEliece algorithm takes the most time in the decryption of data [12].

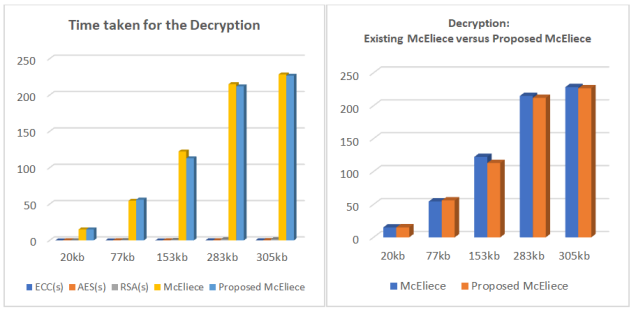

Figure 9: Time taken for decryption process.

In addition, it can be inferred from Figure 8 below that the ECC cryptosystem has the least time (average, 0.01018171 s) for the decryption of data, while the existing McEliece algorithm has the average most time (127.003379 s) in the decryption process as shown in Figure 10 below. Thus, it can be deduced that ECC cryptosystem has the best time for data decryption when compared to the other cryptosystems. Also, comparing the existing McEliece cryptosystem and the proposed McEliece (Figure 10), it can be deduced from the simulation results that the proposed McEliece cryptosystem takes lesser time (124.485727 s) in the decryption of data, making it more efficient in terms of time consumption when decrypting data when compared to the existing McEliece.

Figure 10: Average time taken for the decryption process of various algorithms.

Throughput of the algorithms

The throughput is computed based on the private and public key computation; and also encryption and decryption execution times of the algorithms.

Figure 11 above shows that ECC has the best throughput; however, the proposed McEliece algorithm has a better throughput when compared with the existing.

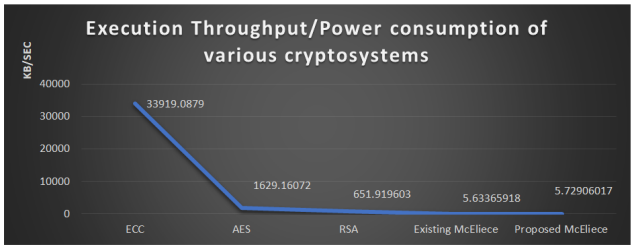

Figure 11: Throughput of algorithms.

Power consumption of various cryptosystems

If the throughput is accurately calculated, the higher the throughput of the time complexity of a cryptosystem, the lesser the power consumption. Thus, the execution throughput is inversely proportional to the power consumption. From the Figure 4.5 above, it can be deduced that ECC has the lowest power consumption. However, the existing NTRU consumes lesser power when compared to the proposed NTRU and the proposed McEliece algorithm consumes lesser power when compared with the existing McEliece algorithm [13].

Time complexity of the proposed McEliece algorithm

Proposed McEliece- Key Generation

Input: Parameters for Key Generation *(S, P, G)*
Output: Keys (G_{key}, P_k)
Begin
i. Calculate the transpose of *S and P*
ii. Compute $G_{key} = S^T.G.P^T$
iii. $P_k = (G_{key}, t)$
iv. Return (G_{key}, P_k)
End

The transpose function

Transpose (int a, int n)

{For (int i=0; i<n; i++)---------- $C_1(n+1)$
For (int j=i+1; j<n; j++)---------- $C_2 n(n+1)/2$

Swap (a[i][j], a[j][i]);---------- $C_3 n(n-1)/2$*}*
$T_{transpose}(n) = C_1(n+1) + C_2 n(n+1)/2 + C_3 n(n-1)/2$

$= (\frac{C_2}{2} + \frac{C_3}{2})\, n^2 + (C_1 + \frac{C_2}{2} - \frac{C_3}{2})n + C_1$. *cancelling all the lesser powers and constant, Thus,* $O(n^2)$

The matrix multiplication algorithm

```
S[A][B], P[G][H]
If B = = G then
        for m=0; m < A; m++ do
                for r=0: r < H; r++ do
                        Q[m][r]=0;
                        for k=0; k<G; k++ do
                                Q[m][r] +=S[m][k]n* P[k][r];
                        End
                End
        End
End
```

Three nested loops make up the matrix multiplication method. The total number of runs in the inner loops would be equal to the length of the matrix for each iteration of the outer loop. Hence the integer operations take O(1) time. In general, if the length of the matrix is N, the total time complexity would be $O(n^*n^*n)=O(n^3)$ [14].

Complexity of the proposed McEliece key generation

- Process 1---Time complexity for the transpose of a matrix=$O(n^2)$
- Process 2--Time complexity for the matrix multiplication=$O(n^3)$
- Process 3--Time complexity for assigning to variable=O(n)
- Process 4--Time complexity of retuning the output=O(n)

Hence, the time complexity for the key generation, considering the highest complexity is $O(n^3)$.

Complexity of the proposed McEliece encryption generation

Below is the computation of the complexity for the proposed encryption of McEliece cryptosystem.

Proposed McEliece - Encryption

Input: Parameters for encryption (m, G_{key})
Output: Cipher Text (C)
Begin
i. Compute $C^l = m * G_{key}$
ii. a random *n*-bit vector *z*, containing exactly *t* ones (a vector of length *n* and weight *t*, is generated)
iii. $C = C^l + z$
iv. Return (C)
End

For matrix addition and a square matrix

```
for 1:n
        for 1:m
                c[i][j]= a[i][j]+ b[i][j]
        end
end
```

Complexity of the proposed McEliece encryption process

- Process 1-- Time complexity for the matrix multiplication=$O(n^3)$
- Process 2-Time complexity for generating an n-bit vector z=O(n)
- Process 3--Time complexity for the matrix addition=$O(n^2)$

- Process 4--Time complexity of retuning the output=O(n)

Hence, the time complexity for the encryption considering the highest complexity is $O(n^3)$.

Complexity of the proposed McEliece decryption process

Below is the proposed decryption for McEliece cryptosystem

Proposed McEliece- Decryption
Input: Parameters for encryption *(S, P, G)* Output: Plain Text (*m*) Begin i. Computes the inverse of P^T i.e. $(P^T)^{-1}$ ii. Use the decoding algorithm for the code *G* to decode *d* to *X* iii. $m = X(S^T)^{-1}$ iv. Return (*m*) End

- Process 1---Time complexity for the transpose of a matrix=$O(n^2)$
- Process 2---Using the syndrome algorithm and multiplication of matrix=O (n^3)
- Process 3---Multiplication of matrix=$O(n^3)$
- Process 4--Process 4--Time Complexity of retuning the output=O(n)

Hence, the time complexity for the decryption, considering the highest complexity is $O(n^3)$.

Time complexity of the proposed McEliece algorithm

Time complexity for (key generation+encryption +decryption)=$O(n^3)+O(n^3)+ O(n^3)=3O(n^3)$. After eliminating constants, the time complexity of the proposed McEliece algorithm is $O(n^3)$. The McEliece algorithm has a time complexity of $O(N^2)$. Though, the proffered McEliece algorithm by this study requires approximately $O(N^3)$ operations. Hence, it can be stated that the existing McEliece algorithm has a better time complexity when compared with the proposed McEliece algorithm, which is mainly as a result of the transpose of a matrix function that was introduced in the computations [15].

Conclusion

This paper proposes a variant of McEliece cryptosystem with the sole aim to ascertain if it will be faster and enhance data security in the cloud. The proposed variant of McEliece was subjected to simulation Alongside Other Symmetric (AOS) and asymmetric (McEliece, RSA and ECC) cryptosystems to ascertain the time complexity of each of the algorithms in terms of key generation, encryption and decryption. It was revealed that in terms of private and public key generation, the RSA cryptosystem revealed to take the least time in terms of key generation and ECC has the best execution time (shortest). Similarly, juxtaposing the existing and proposed McEliece cryptosystem, in terms of key generation, encryption and decryption, findings revealed that the proposed McEliece cryptosystem proved to have the least processing time for key generation (private and public) and decryption, thus making it most efficient.

Future Research

The study suggests that with the advent of quantum computing, enhanced data security and storage; and faster processing capability, a hybrid approach of McEliece system and any other quantum safe algorithm used to safeguard cloud data.

References

1. A Elnapi NM, Omara FA, Omran NF (2016) A hybrid hashing security algorithm for data storage on cloud computing. Intern J Comp Sci Inform Sec 14.
2. Gabriel AJ, Alese BK, Adetunmbi AO, Adewale OS (2015) Post-quantum crystography based security framework for cloud computing. J Intern Technol Secur Trans 4: 351-357.
3. Garg V, Rishu R (2012) Improved diffie-hellman algorithm for network security enhancement. Intern J Comp Tech Appl 3.
4. Kaur K, Seema E (2012) Hybrid algorithm with DSA, RSA and MD5 encryption algorithm for wireless devices. Intern J Engg Res App 2: 914-917.
5. Kuo MH (2011) Opportunities and challenges of cloud computing to improve health care services. J Med Intern Res 13: e1867.
6. Lizy RF (2021) Improvement of RSA algorithm using euclidean technique. Turk J Comp Math Educ 12: 4694-4700.
7. Shankar M, Akshaya P (2014) Hybrid cryptographic technique using RSA algorithm and scheduling concepts. Intern J Netw Sec App 6: 39.
8. Sun P (2020) Security and privacy protection in cloud computing: Discussions and challenges. J Network Comp App 160: 102642.
9. Ouahmane H, Kartit A, Marwan M (2018) A secured data processing technique for effective utilization of cloud computing. J Data Min Digital Humanities.
10. Ukwuoma HC, Arome G, Thompson A, Alese BK (2022) Post-quantum cryptography-driven security framework for cloud computing. Open Comp Sci 12: 142-153.
11. Zeng P, Chen S, Choo KK (2019) An IND-CCA2 secure post-quantum encryption scheme and a secure cloud storage use case. Human-Centric Comput Informat Sci 9: 1-5.
12. El Balmany C, Tbatou Z, Asimi A, Bamarouf M (2022) Secure virtual machine image storage process into a trusted zone-based cloud storage. Computers and Security 120: 102815.
13. Liu Z, Choo KK, Grossschadl J (2018) Securing edge devices in the post-quantum internet of things using lattice-based cryptography. IEEE Communications Magazine 56: 158-162.
14. Balogh S, Gallo O, Ploszek R, Spacek P, Zajac P (2021) IoT security challenges: Cloud and blockchain, postquantum cryptography and evolutionary techniques. Electronics 10: 2647.
15. Kumar A, Ottaviani C, Gill SS, Buyya R (2022) Securing the future internet of things with post-quantum cryptography. Security and Privacy 5: e200.