VFAST Transactions on Software Engineering
http://vfast.org/journals/index.php/VTSE@ 2023, ISSN(e): 2309-3978, ISSN(p): 2411-6246
Volume 11, Number 3, July-September 2023 pp:10-24

Blockchain Technology Performance of Asymmetric Algorithms: An Empirical Study

Mujeeb-ur-Rehman Jamali [1*], **Najma Imtiaz Ali** [1], **Aadil Jamali** [1], **Asad Ali Jamali** [2], **Mazher Ali** [1], **Abdul Khalique Baloch** [3]

[1]Institute of Mathematics and Computer Science, University of Sindh, Jamshoro, Sindh, Pakistan; [2]University of Essex Colchester; [3]Department of Artificial Intelligence and Data Science, Istanbul Aydin University, Turkiye

Keywords: *Blockchain Technology, security, asymmetric cryptography.*

Journal Info:
Submitted:
June 17, 2023
Accepted:
September 16, 2023
Published:
September 22, 2023

Abstract

Blockchain technology is a transparent, and unchangeable distributed ledger. It has the potential to transform the way to interact with the digital world by allowing to construct a decentralized database that is tamper-proof. Concerns regarding the security of confidential and sensitive data being outsourced are rising. It is possible that service providers may be dishonest since unscrupulous administrators have access to, may alter, and can misuse private and sensitive data. Security precautions are necessary because sensitive data stored on public clouds has to be safeguarded. There is no mechanism to detect data changes, as data are stored in plaintext. Therefore, maintaining privacy and secrecy is impossible. It is important that data must always be kept secure, even after it has been kept on the server. Data stored on the server must be safeguarded against outsider access and, if the insider cannot be trusted, must also be safeguarded against hostile insiders. Asymmetric algorithms are employed to safeguard data during transmission. Asymmetric cryptography is required in modern security systems, and several algorithms have been devised to provide safe and effective encryption and decryption. Asymmetric algorithms are empirically compared in this study. We evaluated each algorithm's performance by taking into account criteria such as key size, memory utilization, and execution time. Our results show that while all algorithms provide safe encryption and decryption, there are significant performance disparities between them. It is determined, in particular, that ECC required the least amount of memory and had the shortest key size. The findings show that ECC's prime and binary fields created pairs of keys faster and with more security than other asymmetric algorithms with smaller bit sizes. On small, medium, and big datasets, ECC had the fastest execution time for plaintext encryption operations. These findings have important implications for the selection and deployment of asymmetric algorithms in various security systems.

***Correspondence author email address**: mujeebjamali@usindh.edu.pk
DOI: 10.21015/vtse.v11i2.1439

1 Introduction

A rising number of concerned about the security of private and sensitive data that is being outsourced. Because malevolent database administrators may access, alter, and abuse private and sensitive data, service providers may be dishonest. Sensitive data stored in public clouds must be secured, thus security measures are inevitable. Data is retained in plaintext, and there is no method to detect data modifications. As a result, secrecy and privacy cannot be achieved. Data must be safeguarded at all times, including after it has been stored on the server. This is critical.

Data stored on the server must be safeguarded from outsiders and, if the insider cannot be trusted, must also be protected from hostile insiders. Asymmetric algorithms can allow secure communication between devices and networks by encrypting data during transmission. This protects sensitive information, such as personal data and financial transactions, against eavesdropping and interception.

Blockchain technology makes extensive use of asymmetric algorithms to provide security, confidentiality, and authenticity. With the help of blockchain technology, multiple parties can conduct transactions with one another without the need for a reliable middleman. Asymmetric algorithms are essential for maintaining the integrity and security of blockchain transactions. For the purpose of verifying digital signatures, asymmetric algorithms are frequently used in blockchain technology. The sender of each transaction on the blockchain uses their private key to sign the transaction, and anyone can verify the signature using the sender's public key. Using this, the authenticity of transactions on the blockchain can be ensured in a safe and reliable manner. Data encryption is a crucial application of asymmetric algorithms in blockchain technology. Sensitive information can be encrypted on the blockchain using asymmetric encryption, making sure that only authorized parties with the right decryption key can access it. As a result, the blockchain's data is able to remain confidential and accurate.

Asymmetric algorithms are susceptible to a number of security problems, which may limit their usefulness in some applications. Asymmetric algorithms are susceptible to certain types of attacks, including timing attacks and side-channel attacks, which can jeopardize their performance and security. Computational complexity can make asymmetric algorithms more susceptible to attacks like brute-force attacks, is one of the common security issues and challenges they face. It's possible that the original could be recovered from the ciphertext as a result of successful brute force attacks that attempt every key algorithm imaginable. Due to a lower security level, a key's shorter length could be easily compromised by an attacker. In side-channels attack, intruder may take advantage of data leaks through physical channels. It is crucial to carefully choose an appropriate algorithm for the particular application and its security requirements in order to address these security issues and challenges. Asymmetric cryptography is crucial to the operation of contemporary security systems. Several asymmetric algorithms have been created over time, each with unique advantages and disadvantages. Four commonly used asymmetric algorithms i.e., RSA, DSA, ECC, and ElGamal were empirically compared in this study. In order to determine the advantages and disadvantages of each algorithm, we assessed each one's performance using metrics like execution time, memory usage, and key size. We then compared the findings. Due to the computational complexity required for asymmetric encryption and decryption, asymmetric algorithms are typically slower than symmetric algorithms. This may cause sluggish performance when handling large amounts of data or when applied to devices with limited resources. In order to provide the same level of security as symmetric algorithms, asymmetric algorithms typically need keys with larger sizes. When processing large amounts of data, this may lead to higher memory utilization and slower performance. Due to their higher computational complexity, asymmetric algorithms can be more energy-intensive than symmetric ones. This can be problematic for devices with limited resources, such as mobile phones and Internet of Things (IoT) devices. Because asymmetric algorithms require more work to implement on average than symmetric ones, using them in some applications may be more challenging. Several methods have been developed to address these performance issues, including the use of hardware accelerators to shorten execution times, the optimization of key sizes, and the creation of more effective algorithms. Furthermore, choosing an algorithm that is appropriate for the particular application and its needs can have a significant impact on performance.

Whitfield Diffie and Martin Hellman first introduced the idea of asymmetric encryption in 1976. In 1977, RSA algorithm, was jointly proposed by three

scientists, Rivest Adi, Shamir, and Leonard Adleman. To create public and private keys for encryption and decryption, RSA makes use of big prime numbers. The Integer Factorization Problem (IFP)-based algorithm. In this algorithm, two large prime numbers that can be divided by one another or by themselves are used to generate pairs of keys of various lengths that are mathematically related. RSA's public key is used to validate digital signatures and encrypt data, while its private key is used to create digital signatures and decrypt ciphertext. The CA signs the user's digital certificate using its private key, and the user's digital certificate is validated using the CA's public key. Taher Elgamal created the ElGamal algorithm in 1985. It is based on the fact that discrete logarithm problems in finite fields are difficult to solve. Elliptic curve cryptography (ECC), a development that replaced conventional asymmetric algorithms, was created in the 1990s. The discrete logarithm problem (ECDLP) over finite fields forms the basis of the elliptic curve cryptography (ECC) that Miller and Koblitz proposed. Over the finite field, binary and prime ECC fields are used. A set or series of points are provided for the equation satisfying in these curves. Weiertrass generalized form is the name given to it. the dimensions of the elliptic curve key with the given parameter. If the curve specified and the key size in bits are larger, the efficiency would be slower and more processing time would be needed for encryption and decryption. Compared to other asymmetric algorithms, ECC requires a key size with curve specification that is much smaller. Secret key algorithms i.e., AES or DES is used in conjunction with ECC. It uses little memory and has small key sizes. ECC is especially well suited for applications like mobile devices, which have constrained computational capacity. ECC can be significantly faster and use less memory than RSA, even though it offers security that is comparable to that of RSA. A popular asymmetric algorithm for digital signature verification is the Digital Signature Algorithm (DSA). Despite having a potential for slower performance, DSA requires less memory than RSA. In some applications, the security of DSA may be constrained by its key size restriction.

The goal of this research work is to compare the performance, security, and memory use of prominent asymmetric algorithms i.e., RSA, ECC, DSA, and ElGamal and answers the following questions.

- In terms of efficiency, security, and memory utilization, compare of RSA, ECC, DSA, and ElGamal algorithms. What are the trade-offs for each algorithm in terms of performance, security, and memory usage?
- How do different key sizes and processing resources affect the results?

The study is anticipated to provide light on the advantages and disadvantages of common asymmetric algorithms, as well as their trade-offs in terms of performance, security, and memory utilization. The findings will help practitioners choose the best algorithm for their unique applications based on speed, security, and memory needs. This research work will also add to the present level of knowledge on asymmetric algorithms and point the way forward for future research in this field.

Blockchain Technology

Blockchain technology is a distributed and decentralized digital ledger that makes it possible to record transactions between multiple parties in a secure, transparent, and immutable way. It is the core technology that underpins cryptocurrencies like Bitcoin, but its uses go far beyond virtual money. A blockchain is fundamentally a chain of blocks, each of which contains a list of transactions. These transactions are compiled in a set and connected in time order. The system is extremely secure and impervious to hacking because once a block is added to the chain, it cannot be changed or deleted without affecting all subsequent blocks. Blockchain technology has several important features, including Blockchain runs on a decentralized network of computers called nodes, in contrast to traditional centralized systems. This makes the network more resilient and less prone to single points of failure by ensuring that no single entity has total control over it. Blockchain transactions are encrypted using cryptography for security and immutability. It is very challenging to change a transaction once it has been added to a block. Any

unauthorized changes are quickly detected and rejected because of the network's distributed structure and the consensus mechanisms employed. Transparency: Since every member of a blockchain network has access to the same data, there is a single, open source of truth. This openness can lessen fraud and increase mutual trust. Blockchains employ consensus techniques like Proof of Work and Proof of Stake) to concur on the ledger's current state and approve transactions. These controls ensure the legitimacy of transactions and stop the double use of digital assets.

Blockchain technology now has the ability to perform more complicated, automated processes in addition to straightforward transactions.

Beyond cryptocurrencies, blockchain technology has a wide range of potential uses, such as supply chain management, which involves tracking the movement and provenance of goods throughout a supply chain to increase transparency and lower fraud. Healthcare: Managing the private information of patients while securely transferring medical records and data among healthcare providers. Voting Systems: Establishing secure, open, and transparent digital voting systems to guarantee the fairness of elections. Identity management: Giving people more control over their personal information and digital identities. Financial Services: Simplifying and automating procedures in financial institutions, such as cross-border payments and settlements. Real estate: facilitating efficient and transparent title management and property transfers. Blockchain technology is still developing and finding new applications across numerous industries. Significant interest and investment have been generated from both the public and private sectors due to its potential to improve security, transparency, and efficiency in transactions and data management.

Issues and Challenges in Blockchain Technology

Blockchain technology has many advantages, but it also has a number of security problems that need to be carefully resolved. There are some significant security issues with blockchain technology that is Blockchain transactions are frequently thought of as pseudonymous, but they are still made available to the public. This issue is addressed by privacy-focused blockchains, but popular blockchains like Bitcoin and Ethereum have issues protecting user privacy. Blockchain users must securely manage their private keys. The assets linked to a lost or stolen private key may be permanently lost or stolen. Finding a balance between usability and security is difficult. Network flaws: Blockchain networks are susceptible to attacks such as Distributed Denial of Service (DDoS) attacks, in which the network is overloaded with traffic, causing disruptions and possibly causing consensus problems. Interoperability: As new blockchains appear, the absence of standardized protocols for data exchange and communication can create interoperability issues as well as security risks when integrating dissimilar systems. Issues with regulations and the law: The decentralized nature of blockchain technology can make it difficult to comply with regulations, particularly in industries with strict regulations. It can be difficult to ensure compliance while upholding blockchain's fundamental ideas. Risks Associated with Centralization: While blockchain aims for decentralization, some networks may end up being essentially centralized as a result of the concentration of mining power or stakeholder control, which may jeopardize security and trust. Quantum Computing: As quantum computing technology develops, it is possible that it will eventually be able to crack the cryptography that underpins many blockchain systems, requiring the creation of quantum-resistant algorithms. It takes a combination of technical advancements, industry best practices in development and deployment, ongoing monitoring, community agreement, and regulatory frameworks that strike a balance between innovation and protection to address these security challenges. As blockchain technology develops, researchers and practitioners are working to identify practical solutions to reduce these security risks and guarantee the sustainability of blockchain applications.

According to the authors of this study [1], Blockchain technologies are experiencing new security issues, defects, and security concerns. As a

result, greater study into security methods is required. To address security concerns, the authors recommended smart contracts and security protections to avoid vulnerabilities.

According to the authors of [2] , Blockchain is a relatively new technology that is a distributed database used for sharing amongst computer network nodes. Blockchain innovation protects the accuracy and security of data records while also generating confidence in the absence of a trusted third party. The authors assess the security risk of blockchain systems, examine the vulnerabilities exploited on the blockchain, and outline recent security difficulties that the blockchain is facing.

The authors evaluated the hazards and basic security considerations associated with the cloud environment. According to the them, Blockchain technologies have security, information management, compliance, and dependability challenges. The authors exposed several cloud security concerns and network difficulties in order to thwart cloud attacks and simplify risk reduction strategies for researchers, end-users, and cloud providers doing threat analysis. By combining blockchain with cloud computing, the communication between blockchain and cloud is shown [3] . According to [4], block chain technology has touched several sectors and enterprises, affecting the lives of countless individuals. While the characteristics of block chain technology have the ability to provide more trustworthy and convenient services, serious security issues remain. The authors explain the concept of block chain, its current applications in business, and the myriad risks and security problems related with block chain technology. The broad deployment of block chain technology has the potential to overcome intractable trust issues in a wide range of businesses.

2 Related Work

For the protection of sensitive data and communications, asymmetric algorithms, also referred to as public-key cryptography, are used to offer secure encryption and decryption. It offers a reliable and effective solution to the security issues present in the online world. It is a type of encryption in which two keys are used: one for encrypting the data and the other for decrypting it. A thorough literature study is carried out to determine the current status of research on asymmetric algorithms, their strengths and limitations, and the factors that influence their performance, security, and memory utilization. The security lifespan of algorithms was reported in [5] . According to the NIST security standard and guidelines on the security of the well-known asymmetric algorithms ElGamal, RSA, and ECC, as presented in Table 1:

Table 1. Security lifespan of an asymmetric algorithm

bits	RSA	ElGamal	Elliptic	Lifetime
80	1024	1024	160	2010
112	2048	2048	224	2030
128	3072	3072	256	Beyond 2030
192	7680	7680	384	Beyond 2030
256	15360	15360	512	Beyond 2030

The authors evaluated the performance of ElGamal, RSA, and ECC for key pair generation. Table 2 shows the encryption and Table 3 depicts the encryption of 159 bits plaintext using a public key, followed by decryption using a private key, with results recorded in milliseconds. The findings suggested that ECC was superior to RSA and ElGamal in terms of encryption. The authors proposed that in future work, data compression on the encrypted plaintext may be used to reduce the size of the ciphertext:

Table 2. ElGamal, RSA and Elliptic Curve Cryptography Key Pair Generation

Key size	RSA	ElGamal	Elliptic
160	951	406	2437
224	1237	830	4449
256	1957	895	6451
512	16863	15946	40317
1024	76498	68773	318574

Table 3. ElGamal, RSA and Elliptic Curve Cryptography Encryption and Decryption

Key size	RSA	ElGamal	Elliptic
160	18 / 10	2816 / 18	2696 / 1292
224	28 / 28	5842 / 27	6268 / 2907
256	37 / 37	7098 / 37	8242 / 3932
512	258 / 259	29388 / 194	57236 / 27034
1024	2013 / 1966	140979 / 1107	411558 / 198823

The authors of [6] suggested that ECC is more secure since it is based on the most difficult to solve discrete logarithmic problem. The ECC algorithm was used to encrypt and decrypt plaintext. The authors suggested a mechanism for translating plaintext to ASCII and then Hex decimal values. ECC algorithm with 192 bit key size with curve specification utilized. The encryption and decryption operations were timed in milliseconds, i.e., 0.08 and 0.06, respectively. The results indicated that the ECC encryption procedure took longer than the decryption operation. The author claims that ECC is more effective than RSA with a smaller key size and an identical degree of security. Data confidentiality during the transition requires protection via electronic mail, a financial transaction in banking and credit details, and so on via an insecure cloud. During data transmission, private and confidential data might be intercepted and read. Ransomware assaults are also on the rise. Because of the lower security and longer time required for encryption and decryption, the author suggested the RSA modified method, SRNN. The SRNN algorithm made advantage of the huge prime integer. The authors believed that changing the RSA algorithm might improve security [7].

In [8] a short-range natural number was proposed as a tweak to the RSA method for key pair creation. The authors contended that altering the RSA algorithm may improve security.

In [9] authors implemented the RSA algorithm for encryption and decryption with data of varying sizes. The duration of the performance was measured in seconds. The authors discovered that for the same size of key and data, RSA took longer to encrypt than it did to decrypt. To increase the efficiency of an existing RSA cryptosystem, two alternative keys are used. In this method, a separate key for tiny data sizes is utilized alongside another key for large data sizes for encryption and decryption. Cryptographic processes were timed in milliseconds. The tiny size of the natural number employed in the procedure in the modified RSA algorithm gave more safety, according to the findings.

The work [10] stated that security is the primary worry for cloud consumers. As a result, privacy and security are important concerns for data kept on the cloud. The authors stated that traditional algorithms may be utilized to accomplish data security. The author presented an upgraded version of RSA, known as the ERSA algorithm, which uses prime integers in addition to the original RSA method. The authors stated that breaking ERSA into numerous blocks to file will allow it to run faster. This strategy may enhance computing complexity while also increasing security.

Authors [11] advocated that unpredictability of the key be utilized for encryption to boost data security, but that this would require more resources. The authors proposed that multimedia content encryption be upgraded so that secure transmission via unprotected connections is achievable.

Cloud architecture presented two secure methods in [12] . In this strategy, the user encrypts data with the RSA algorithm before sending it to a cloud environment. Meanwhile, in the second technique, the administrator executes cryptographic activities by obtaining public and private keys from cloud users. Both encryption and steganography techniques were used in [13] to secure private and sensitive data on the local host. The cryptographic asymmetric technique, RSA, is employed for security, together with two consecutive layer steganography audio-based, which gives the greatest possible security. Using audio steganography and cryptography, authors [13] outlines a method for data security. The authors suggested inserting confidential data that ensures secure data into the audio data. In the first layer, the R-Prime RSA algorithm was used to improve the original RSA algorithm and encrypt the data. A genetic algorithm will be

used to insert data into the audio file in the second layer. The authors claimed that using random layers to thwart attacks was necessary because the data cannot be discovered and, even if it were, it would be challenging for an attacker to extract. The audio file is not distorted as a result, and higher capacity and robustness are attained. Similar to this, authors presented a system that uses audio steganography and cryptography together. By converting data from plaintext to ciphertext and then embedding it in an audio file, the RSA algorithm was used. Insert the ciphertext during steganography. They argued that the findings showed the audio file data is not altered in terms of quality or size. When using a proxy server, the server may store and run a query against an unstructured database. To accelerate cryptographic operations, the RSA algorithm will be modified for encoding and decoding. It was suggested that steganography could be used to secure multimedia contents while also using encryption to protect confidential data.

The proposed work offers an alternative strategy from that which has previously been put forth by other studies. Data at rest encryption was not recommended in this study; instead, application-level security is used. The sensitive data is encrypted at the application-level. It offers total control over data and it is independent. It is discovered through the literature that the systems currently in use are designed for a time when there are greater security concerns, which needed to be addressed with more advanced methods.

3 Proposed Methodology

Asymmetric algorithms are essential to contemporary cryptography and are frequently employed in a wide range of contexts, such as secure communication and data encryption. We implemented each algorithm and carried out experiments to assess the performance of each algorithm. For each algorithm, we kept track of the execution time, memory usage, and key size. To find any notable variations in performance between the algorithms, we statistically analyzed the data. In terms of performance and memory usage, the goal of this study is to compare asymmetric algorithms. When choosing an algorithm for a particular application, it is crucial to be aware of the trade-offs between various asymmetric algorithms' performance characteristics and memory needs. We have implemented and tested RSA, ECC, ElGamal, and DSA, on a computer system with various computational resources and key sizes. Measure each algorithm's performance and memory usage in various contexts. Based on an application's performance and memory needs, the research's findings can be used to guide the choice of asymmetric algorithms. By highlighting potential areas for additional study and advancement, the findings can also aid in the ongoing development of the cryptography field. Each algorithm is put into practice while using various computational resources and key sizes. The information gathered includes the time required to encrypt and decrypt data, memory usage, and security metrics like key strength and attack vulnerability.

System Design

We have used asymmetric cryptography to scramble data into unintelligible ciphertext to protect it. Only the private key can be used to decrypt the ciphertext back into plain text. The discrete logarithm, integer factorization, and elliptic curve serve as the foundation of RSA, ElGamal, ECC, and DSA respectively.

Our design of system use the standardized modeling language called Unified Modeling Language (UML) to visually represent system designs. One of the most popular types of UML diagrams is the class diagram, which gives a graphical representation of the classes, interfaces, and connections between them in a system. Classes are the primary building blocks of a UML class diagram, but there are other elements as well. They represent the individuals or things that comprise the system and frequently contain attributes (data fields) and functions (methods) that specify their behavior. Relationships specify how the system's classes relate to one another. In UML, there are different kinds of relationships, such as association (a connection between two classes), aggregation (a part-to-whole relationship between classes), and inheritance (a hierarchical relationship between classes). Multiplicity

describes the number of instances of a class that can be linked to another class. A range of values, like 0 is used to represent it. A superclass and one or more subclasses are related in a generalization. The subclass represents relationship in which the superclass's attributes and methods are inherited. By outlining the classes and connections that go into a system, UML class diagrams used to design system.

UML class diagrams offer a standardized and visual way to represent the classes, interfaces, and their relationships within a system. They are an effective tool for creating, expressing, and documenting system designs. To determine the variables that affect each algorithm's performance, security, and memory usage, the data is analyzed using the appropriate statistical techniques. The data analysis results are compared to determine the trade-offs between each algorithm's performance, security, and memory usage. Figure 1 depicts the attributes, operations, and parameters of the proposed system's operations using asymmetric algorithms.

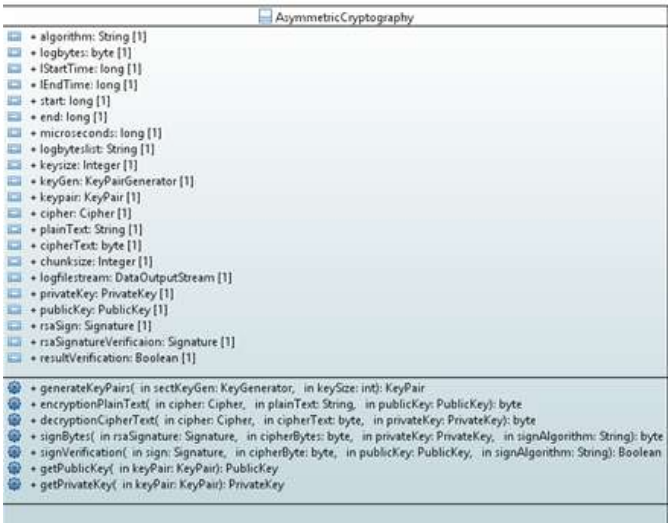

Figure 1. Asymmetric Cryptography Class Diagram

Figure 2 depicts the properties, processes, and parameters of the proposed system ECC algorithm for key pair formation, plaintext encryption, and ciphertext decryption.

The production of ECC keys pairs with curve specification secp112r1. One of the frequencies observed is reproduced below [14]:

Public Key:X.509
Private Key:(PKCS#8)
Public Key:
Sun EC public key, 112 bits
public x coord: 1429944473917238685555471383174188
public y coord: 23598004971726002615866971229249O
parameters: secp112r1 (1.3.132.0.6)
Private Key: 48 44 2 1 0 48 16 6 7 42 -122 72 -50 61 2 1 6 5 43 -127 4 0 6 4 21 48 19 2 1 1 4 14 76 -102 25 116 86 -1 82 21 50 -13 115 111 -121 37

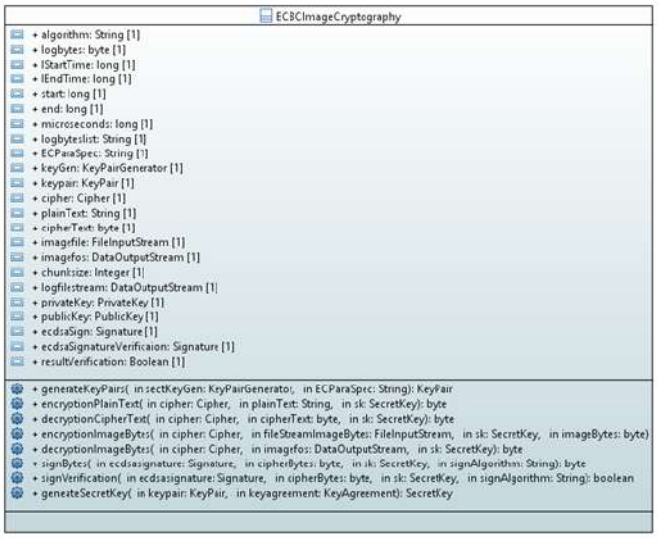

Figure 2: ECC Cryptography Class Diagram

For encryption and decryption, ECC asymmetric encryption with hybrid symmetric key i.e., AES with different ECC key pairs (curve specification) size in bits i.e., secp128r1, secp128r2, secp192r1, and secp256r1 is used. ECC curve specifications using AES The original plaintext is 23 bytes long, while the ciphertext is 32 bits long. The frequencies were recorded, and the data is reproduced below as encrypted:

id : ObjectId("5e0f4c0fc482b3138c98cc5e")
UserId : 3310,
Name : Binary
('LTU5IDI1IC00NCAxMDMgMT
A0IDEwMyAtMiAtMTA3IC01MSAtNz
UgNjYgNzQgL TQgLTExMCA2MiA3OCAt
OTQgLTE2IC05IC0xOSAt...')

Figure 3 depicts the properties, actions, and parameters of RSA algorithm for key pair creation, plaintext encryption and decryption.

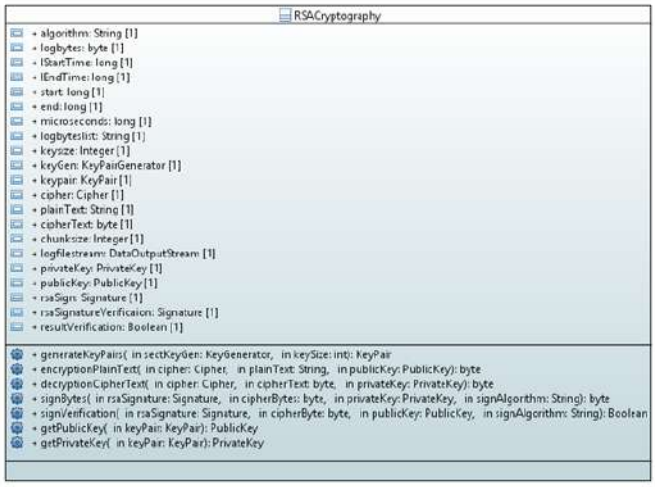

Figure 3: RSA Asymmetric Cryptography Class Diagram

The RSA public key of 1024, 2048, and 4096 bits is used for encryption of plaintext of 23 bytes, which is turned into ciphertext of 128, 256, and 512 bits, respectively and the frequencies are recorded, and the data is looked like using BASE64 encoded when it is encrypted reproduced below [15]: id : ObjectId("5e0f4c0fc482b3138c98cc5e")
UserId: 3310,
Name: Binary ('
dyxWGvzIWkVqHFRReluBJmu0
nnSSIsxA+8VgyTH6AA8QgMbqtss
OnUAV7pGDLncmVFF0r18UAZzePZHr1
31Ik6f1y+fGykEYN7KXkNpOU9mdK
515iaxmW9yobgBLYsho40r7s1/3s
KhvrUR1SU4GLnnciX/
j6l3I+W1nZefTQnU=')

Figure 4 depicts the properties, actions, and parameters of ElGamal's algorithm for key pair creation, plaintext encryption, and ciphertext decryption.

The frequency and ElGamal keys pairs of 128, 192, and 256 are recorded, and the ElGamal public key is used to convert while the method uses the private key to retrieve the original data reproduced below:
Public Key:X.509
Private Key:PKCS#8
Public Key:

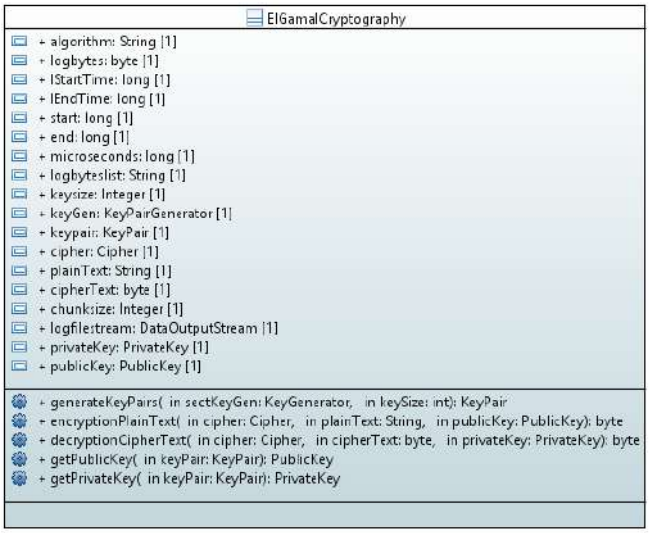

Figure 4: ElGamal Asymmetric Cryptography Diagram

128 Bit 48 70 48 47 6 6 43 14 7 2 1 1 48 37 2 17 0 -127 -77 101-114 -30 69 95 27 66 -92 87 -125 -12 -126 -43 99 2 16 109 92 -116 96 85 -102 48 79 -31 -126 -26 -67 -121 103 40 48 3 19 0 2 16 23 20 -51 -74 -4 16 -110 -4 27 -4 -3 -71 98 114 -58 -65
Private Key:
128 Bit 48 72 2 1 0 48 47 6 6 43 14 7 2 1 1 48 37 2 17 0 -127 -77 101 -114 -30 69 95 27 66 -92 87 -125 -12 -126 -43 99 2 16 109 92 -116 96 85 -102 48 79 -31 -126 -26 -67 -121 103 40 48 4 18 2 16 74 61 62 57 -24 -55 -80 -14 -78 91 95 59 103 -5 -10 123
Public Key:
192 Bit 48 96 48 64 6 6 43 14 7 2 1 1 48 54 2 25 0 -102 -122 -116 3 68 -41 114 -48 73 18 65 -7 121 -75 111 67 17 -29 89 -97 14 60 9 -113 2 25 0 -116 64 -44 1 0 -50 53 98 -44 -91 124 -74 -41 7 127 -60 90 27 -85 123 127 -105 91 4 3 28 0 2 25 0 -118 -41 -104 -981 46 -88 95 53 109 -20 -28 -92 94 -94 101 -5 28 39 -58 45 76 -48 -85
Private Key:
192 Bit 48 97 2 1 0 48 64 6 6 43 14 7 2 1 1 48 54 2 25 0 -102 -122 -116 3 68 -41 114 -48 73 18 65 -7 121 -75 111 67 17 -29 89 -97 14 60 9 -113 2 25 0 -116 64 -44 1 0 -50 53 98 -44 -91 124 -74 -41 7 127 -60 90 27 -85 123 127 -105 91 4 4 26 2 24 16 65 66 6 -121 22 -47 84 -43 77 -9 57 110 65 -25 58 -124 -96 -13 118 65 46 59 -44
Public Key:
256 Bit 48 118 48 79 6 6 43 14 7 2 1 1 48 69 2 33 0 -73 64 -96 -89 101 21 31 -120 -54 9 42 -16 63 0 -73 -93 -57 8 12 45 126 -59 -72 4 88 -127 123 97 115 37 -65 99 2

32 76 -31 -117 63 -27 -24 76 -1 83 20 -2 3 -9 105 110 36 66 124 104 -125 -43 -52 -100 9 7 -125 56 -106 -89 -79 -9 100 3 35 0 2 32 126 -108 -67 29 57 -12 -52 -59 116 109 -98 -36 -122 104 -97 15 69 76 -103 94 36 -85 23 -8 -38 -71 -26 65 1 2 66 -31

Private Key:

256 Bit 48 120 2 1 0 48 79 6 6 43 14 7 2 1 1 48 69 2 33 0 -73 64 -96 -89 101 21 31 -120 -54 9 42 -16 63 0 -73 -93 -57 8 12 45 126 -59 -72 4 88 -127 123 97 115 37 -65 99 2 32 76 -31 -117 63 -27 -24 76 -1 83 20 -2 3 -9 105 110 36 66 124 104 -125 -43 -52-100 9 7 -125 56 -106 -89 -79 -9 100 4 34 2 32 57 -28 65 56 76 81 33 65 -64 -114 -50 67 70 41 -27 90 7 -59 1 55 100 -65 -91 48 117 63 56 18 41 -82 46 2

The rise in the size of the ElGamal keys pair generation i.e., 128, 192, and 256 which gives a level of protection against brute force attacks.

ElGamal public keys of 128, 192, and 256 bits are used for encryption plaintext of 16 bytes and ciphertext length created with varied widths depending on the key, namely 32, 48, and 64 bytes. ElGamal created 32, 48, and 64 bytes ciphertext, which looked like when encrypted from the plaintext of 16 bytes, which is a Base64 encoded encrypted field in a document database, utilizing 128, 192, and 256-bit long keys:

id : ObjectId(“5e0f4c0fc482b3138c98cc5e”)
UserId : 3310,
Name : Binary
(’OTYgMTEzIDQ3IC02MCA0MCA2IDUzIC0xMTUgLTcx IDQ5IDM4IC04IDc0IC00MCA xMTAgLTEwOSA4O SAxMjQgNDMgNTEgLTYwIDYg...’)

4 Hardware and Software Requirements

The system is implemented and tests are conducted to evaluate the research work’s goals and objectives. For the cryptographic implementation of asymmetric algorithms, the Java Development Kit with cryptographic libraries (JCA and JCC) and third-party cryptography providers (SunEC and Bouncy Castle API) are utilized to execute the complicated mathematical processes involved in asymmetric cryptography. The implementation IDE and system architecture designed using ADL (Papyrus), AcmeStudio and Eclipse Modeling Neon.

5 Results and Discussion

The system is implemented and tests are conducted to evaluate the research work’s goals and objectives. For the cryptographic implementation of asymmetric algorithms, the Java Development Kit with cryptographic libraries (JCA and JCC) and third-party cryptography providers (SunEC and Bouncy Castle API) are utilized to execute the complicated mathematical processes involved in asymmetric cryptography. The implementation IDE and system architecture designed using ADL (Papyrus), AcmeStudio and Eclipse Modeling Neon. The cryptographic algorithm’s security is evaluated by its resistance to brute force assaults, factorization attacks and other attacks. The metrics and variables utilized in a comparative study of asymmetric algorithms is based on the analysis’s aims and objectives. However, variables are used to assess the performance of asymmetric algorithms. The time takes the algorithm to execute a operation, such as encrypting or decrypting data. The amount of memory consumed by the algorithm while it is running. The size of the algorithm’s cryptographic keys. Larger key sizes result in better encryption, but they increase the algorithm’s execution time and memory use.

Cryptography Algorithm Metric Descriptions

Asymmetric key cryptography known as public key cryptography, employs a pair of mathematically linked keys known as public and private keys. Key length is a metric to use as a number of bits. The initial parameter for describing cryptographic methods is key length. The public key is used to encrypt data, while the private key is used to decrypt data. There are several asymmetric key algorithms, each with advantages and disadvantages that may be employed as a user or organization’s security need. The size of the keys pair defines the security level; as the size of the keys increases it provides level of security. The key length in this study was chosen to offer a comparative value of the outcomes synthesis. The security of asymmetric algorithms (ElGamal, RSA, DSA, and ECC)

is enhanced by the length of the key. The longer the key, the more resistant the algorithm is to a successful brute force attack.

Empirical comparison of a maximum key size asymmetric algorithms

Asymmetric Algorithms that is ElGamal, DSA, RSA and ECC (with various curve specifications prime and binary fields) is used to collect/gather primary data which is then arranged and evaluated. Algorithm performance is measured in milliseconds. Comparative examination of maximal size asymmetric key pair generation of ElGamal (256), DSA (1024), RSA (4096), ECC (secp521k1 sect571k1) is in milliseconds, which is 754.2, 64.4, 4419.9, and 49 54 (ECC) respectively, as shown in Table 3. The data indicates that ECC required less time to generate key pairs than other asymmetric algorithms (ElGamal, DSA, and RSA).

Table 4. Compares key pair creation

ElGamal	DSA	RSA	ECC	ECC
754.2	64.4	4419.9	49	54
407.593	1.075	1118.271	1.023	2.015

Source: primary data

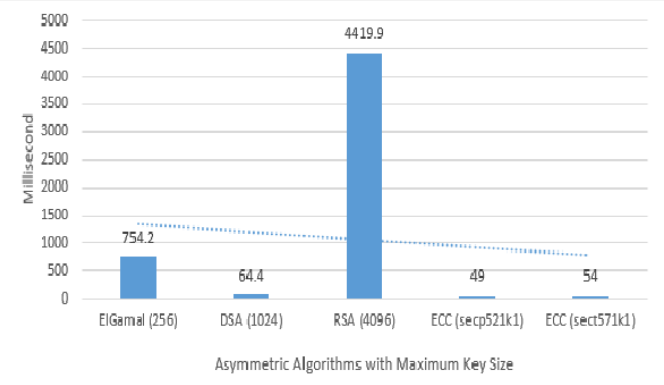

Figure 5: Comparison key pair creation

Figure 5 depicts a cooperative and comparative analysis of ElGamal, DSA, RSA, and ECC at the application level for symmetric secret key creation. DSA was discovered to be 485.704% quicker than ElGamal and 2846.415 percent faster than RSA. The results reveal that ECC with curve specification prime fields (secp521k1) is 31.556 percent quicker than DSA, while ECC with curve specification binary fields (sect571k1) is 34.776 percent faster.

A comparison of maximum size asymmetric cryptography encrypting and decryption process. The security analysis of each algorithm's execution time is reproduced in Table 6 which is thereafter used for cooperative and comparative analysis of asymmetric algorithms maximum key size in bits long encryption and decryption of ElGamal (256), RSA (4096), and ECC (secp256r1) at the application level. In asymmetric cryptography, encryption is the process of converting plaintext to ciphertext using the public key. The time is measured in milliseconds i.e., 375.3, 5.5, and 2 respectively. The private key used for decryption is 2.9, 58.6, and 198.2 respectively. It is found from the results that ECC takes less time to encrypt than other asymmetric techniques. Meanwhile, it has been discovered that ElGamal takes less time to decrypt than other asymmetric algorithms.

Table 5. Asymmetric algorithms plaintext comparative analysis

ElGamal(256)	RSA(4096)	ECC (secp256r1)
1108/6	5.5/58.6	2/198.2
78/2	0.527/9.913	0.471/32.424
-	13.3/160.2	4/1
-	1.032/39.32	1/1
-	-	5/2
-	-	1/1

Source: primary data

Figure 6 depicts the cooperative research of asymmetric algorithms with maximum key size in bits encryption and decryption using ElGamal, RSA and ECC at the application level for small plaintext datasets. ECC (secp256r1) is determined to be 175 percent quicker than RSA for encryption operations. Meanwhile, in decryption, ECC is 238.22 percent slower than RSA. ElGamal algorithm with 256 bit key size is slower than ECC and RSA for encryption, but it is quicker than ECC and RSA for decryption. It can be deduced from the results that ECC (secp256r1) consumed less time for encryption operation than other asymmetric algorithms with small standard deviation recorded that indicate the behavior of data is consistent because data point tends

to be very close, ElGamal decryption operation time is smaller than ECC and RSA with small consistent behavior.

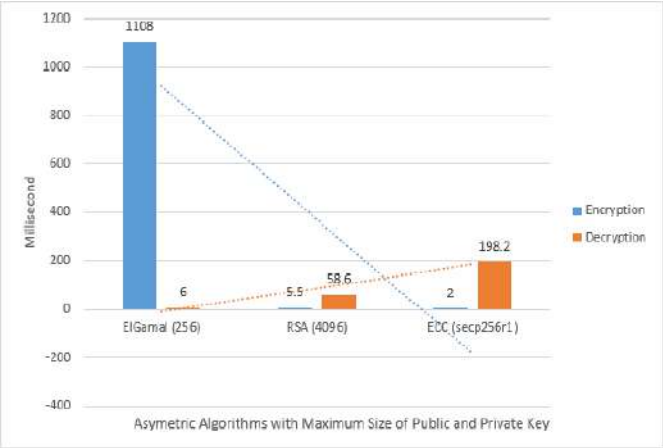

Figure 6: Asymmetric algorithms small plaintext comparative analysis

In Figure 7, RSA (4096) and ECC (secp256r1) algorithms performed the encryption operation on medium dataset. The performance of above mentioned algorithms recorded in a millisecond and dispersion value is 13.3±1.032 and 4±1 respectively. In decryption operation it was found that RSA with 4096 bits key took 160.2±39.32 and ECC with Secp256r1 took 1±1. It is limitation of ElGamal (256) algorithm that it cannot encrypt 276 bytes of plaintext. The both algorithms RSA (4096) and ElGamal (256) cannot perform encryption operation on plaintext of 529 bytes. ECC (secp256r1) took 5±1 for encryption and 2±1 for decryption. ElGamal and RSA are unable to perform the encryption and decryption operation on medium and large datasets due to their limitations as asymmetric algorithms.

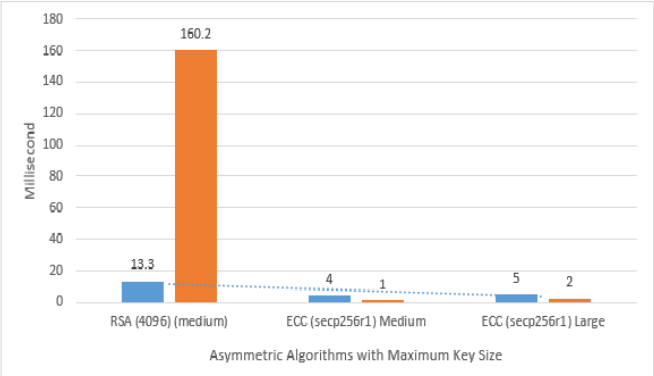

Figure 7: Asymmetric algorithms medium and large plaintext comparative analysis

In Table 6, the performance of various asymmetric algorithms is compared using the percentile comparison statistical analysis method. In a percentile comparison, the algorithms' execution times and memory usage are gathered and listed in order of highest to lowest performance metrics. Then, by figuring out where on the ranked list each algorithm ranks, the effectiveness of each algorithm is compared to the others. The percentile comparison takes into account both the relative ranking of the performance metrics and the distribution of the data, it enables us to meaningfully compare the effectiveness of asymmetric algorithms. We can compare the percentile rankings of various algorithms to see which algorithms perform better than others and to decide which algorithm is best suited for a given application.

Table 6. Asymmetric algorithms plaintext comparative analysis

ElGamal(256)	DSA(1024)	RSA(4096)	ECC(Prime)
482.5/367/3	62.1/-	3624.5/5/52.5	49/2/182.75
590.5/375.5/3	64.4/-	4063/5.5/55	49/2/194
803.75/387/3	67.2/-	5161.5/6/64.5	49/2/227.75

Source: primary data

Findings are presented as an average/median that is 50 percent or second quartile (Q2). Quartile (Q1, Q2, and Q3) are calculated using actual values or recorded results; as a result, results reported in the literature do not represent actual results but rather an analysis of results, i.e. Average or median. The drawback of ElGamal (256 bits key) cryptography is that medium datasets of plaintext (i.e., 276 bytes) and large dataset of plaintext (i.e., 529 bytes) cannot be encrypted. In practical ElGamal algorithm has a restriction that it can only support up to 16 bytes of data with a 128 bit public and private key. With a key pair of 192 and 256 bits, it can encrypt 23 bytes of plaintext. For encryption, it is revealed that there is a huge difference between encryption and decryption of medium datasets of RSA and ECC. It can be inferred from the result less time consumed for encryption and decryption operation of

ECC as compare to other asymmetric algorithms with small standard deviation recorded behavior of data is consistent and data point tends is very close. Our findings demonstrate that while all asymmetric algorithms offer secure encryption and decryption, there are noticeable performance differences between them. In particular, we discovered that ECC had the smallest key size and the least amount of memory usage, whereas RSA had the quickest execution time. While DSA had the largest key size but was relatively slow in terms of execution time, ElGamal had a moderate execution time and memory utilization.

6 Conclusion

Asymmetric cryptography is a necessary component of modern security systems, there are several algorithms have been developed to provide secure and reliable encryption and decryption. Three widely used asymmetric algorithms i.e., RSA, ECC, and ElGamal are empirically contrasted in this study. The metrics of key size, memory usage, and execution time are used to evaluate each algorithm's performance. Our results show that although all three algorithms provide secure encryption and decryption, there are distinct performance differences between them. It is found that the ECC had the smallest key size and the least memory consumption. The outcomes show that ECC's prime and binary fields generated pairs of keys more quickly and securely than other asymmetric algorithms with smaller bit sizes. Plaintext encryption operations on small, medium, and large datasets, our findings have important implications for the selection and use of asymmetric algorithms in different security systems. There are a number of algorithms that can be applied to lessen the security concerns and difficulties brought on by the use of asymmetric algorithms. In order to offer the same level of security as symmetric algorithms, asymmetric algorithms typically need larger key sizes. While maintaining the same level of security, it is possible to increase performance and decrease memory usage by optimizing key sizes. Use of efficient algorithms: Asymmetric algorithms are typically more computationally complex than symmetric algorithms, which can make them more susceptible to attacks like brute-force attacks. However, using more effective algorithms, like elliptic curve cryptography, can significantly boost performance while keeping the same level of security. Post-quantum cryptography is used because many public-key algorithms that are currently in use can be broken by quantum computers, which are capable of attacking and disrupting asymmetric algorithms. Post-quantum cryptography algorithms that can withstand attacks from quantum computers are being created in an effort to lessen this vulnerability. In addition to these methods, it's crucial to choose an algorithm carefully based on the application in question and the security requirements that must be met, as well as to keep up with the most recent advancements in security and industry best practices. An important area of research in the field of cryptography is the comparative analysis of asymmetric algorithms, which identifies the advantages and disadvantages of various algorithms and their trade-offs in terms of speed, security, and memory consumption. Using cutting-edge statistical techniques to analyze the data gathered from the application of each algorithm with various key sizes and computational resources is one of the research's novel features. With the aid of these techniques, it may be possible to pinpoint the variables that each algorithm's performance, security, and memory usage are influenced by as well as to gain a more complex understanding of how these variables are traded off. The use of various metrics to assess each algorithm's security, including key strength and attack vulnerability, is another novel feature of this research. These metrics can offer a more complete picture of the security of each algorithm and can aid in the discovery of any flaws or vulnerabilities that might not be obvious from a purely performance-based analysis. Blockchain technology relies heavily on asymmetric algorithms, which offer security, confidentiality, and authenticity for blockchain transactions. The use of asymmetric algorithms in maintaining the system's security and dependability is likely to grow as the use of blockchain technology does as well. In general, the selection of an asymmetric algorithm is based on the particular needs of the application. ECC might be the

best choice for applications where memory usage is crucial. intended for applications where efficiency and speed are crucial. Overall, comparative analysis of asymmetric algorithms is a novel and significant field of research that can shed light on the benefits and drawbacks of various algorithms as well as their trade-offs in terms of efficiency, security, and memory consumption. This research can aid in the creation of more effective and secure cryptographic systems as well as support the field of cryptography's ongoing development.

7 Future Directions

Asymmetric algorithms are susceptible to attacks from quantum computers, which are able to decrypt many of the public-key algorithms currently in use. Given the increasing power of quantum computers and the possibility that they might exist soon, this is a serious concern. The application of asymmetric algorithms can be very advantageous for 5G technology in a number of ways. By enabling secure communication, data encryption, and digital signature verification, asymmetric algorithms can improve the security and privacy of 5G networks.

Author Contributions

Dr. Mujeeb-ur-Rehman Jamali: Research Experiments and Implementation **Dr. Najma Nawaz Channa**: Conceptualization and Methodology **Aadil Jamali**: Acquisition of data and Evaluation **Asad Jamali**: Writing Original Draft **Mazher Ali**: Editing, Results Comparison **Abdul Khalique Baloch**: Reviewing, Data curation

Compliance with Ethical Standards

There are no conflicts of interest declared by the author.

References

[1] R. Pise and S. Patil, "A deep dive into blockchain-based smart contract-specific security vulnerabilities," in *2022 IEEE International Conference on Blockchain and Distributed Systems Security (ICBDS)*, 2022, pp. 1–6.

[2] A. AlFaw, W. Elmedany, and M. S. Sharif, "Blockchain vulnerabilities and recent security challenges: A review paper," in *2022 International Conference on Data Analytics for Business and Industry (ICDABI)*, 2022, pp. 780–786.

[3] M. Rani, K. Guleria, and S. N. Panda, "Blockchain technology novel prospective for cloud security," in *2022 10th International Conference on Reliability, Infocom Technologies and Optimization (Trends and Future Directions) (ICRITO)*, 2022, pp. 1–6.

[4] I. Muda, S. Madem, S. Hasan, S. Ahmod, R. A. Kayande, and N. Chakraborty, "A survey on applications and security issues of blockchain technology in business sectors," in *2022 2nd International Conference on Technological Advancements in Computational Sciences (ICTACS)*, 2022, pp. 663–668.

[5] M. Agoyi and D. Seral, "Sms security: An asymmetric encryption approach," *IEEE Sixth International Conference on Wireless and Mobile Communications*, 2010.

[6] K. K and D. B. Surendiran, "Elliptic curve cryptography for secured text encryption," *IEEE International Conference on circuits Power and Computing Technologies ICCPCT*, 2017.

[7] S. Y. Bonde and P. D. U. Bhadade, "Analysis of encryption algorithms (rsa, srnn and 2 key pair) for information security," *IEEE*, pp. 1–4, 2017.

[8] E. A. Young Sil Lee and H. J. Lee, "An efficient encryption scheme using elliptic curve cryptography (ecc) with symmetric algorithm for healthcare system," *International Journal of Security and Its Applications*, vol. 8(3), pp. 63–70, 2014.

[9] S. V. V. Z. Parker, S. Poe, "Comparing nosql mongodb to an sql db," *Proceedings of the 51st ACM Southeast Conference (ACMSE '13)*, pp. 1–6, 2013.

[10] D. D. G. Amalarethinam and H. Leena, "Enhanced rsa algorithms with varying key sizes for data security in cloud," *IEEE World Congress on Computing and Communication Technologies (WCCCT))*, pp. 172–175, 2017.

[11] V. Srinivasan Nagaraj, Dr.G.S.V.P.Raju, "Data encryption and authentication using public key approach," *Elsevier Procedia Computer Science*, vol. 48, pp. 122–132, 2015.

[12] M. N. S. S. S. Ashutosh Kumar Dubey, Animesh Kumar Dubey, "Cloud-user security based on rsa and md5 algorithm for resource attestation and sharing in java environment," *IEEE*, pp. 1–8, 2016.

[13] N. M. S. S. S. Kumar Ashutosh, Kumar Animesh, "A survey on querying encrypted data for database as a service," *IEEE International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery*, 2013.

[14] N. A. K. Mujeeb-ur Rehman Jamali, Abdul Ghafoor Memon, "Data integrity issues and challenges in next generation non-relational document-oriented database outsourced in public cloud," *International Journal of Emerging Trends in Engineering Research*, vol. 9(4), pp. 416–420, 2021.

[15] M. R. M. M. R. Jamali, A. G. Memon, "Security issues in data at rest in a non-relational document database," *Sindh Univ. Res. Jour. (Sci. Ser.)*, vol. 52(3), pp. 279–283, 2020.