DATA CRYPTOGRAPHY ALGORITHM

by

Rawan Moied Almutairi

A research project submitted for the requirements of the degree of
Executive Master of Cybersecurity

Faculty of Computing and Information Technology
King Abdulaziz University
Jeddah, Saudi Arabia
Dhul Qi'dah 1443 H - Jun 2022 G

DATA CRYPTOGRAPHY ALGORITHM

by

Rawan Moied Almutairi

A research project submitted for the requirements of the degree of Executive Master of Cybersecurity

Advisor

Dr. Shahenda Sarhan

Faculty of Computing and Information Technology
King Abdulaziz University
Jeddah, Saudi Arabia
Dhul Qi'dah 1443 H - Jun 2022 G


بسم الله الرحمن الرحيم

DATA CRYPTOGRAPHY ALGORITHM

by

Rawan Moied Almutairi

A research project submitted for the requirements of the degree of
Executive Master of Cybersecurity

Faculty of Computing and Information Technology
King Abdulaziz University
Jeddah, Saudi Arabia
Dhul Qi'dah 1443 H - Jun 2022 G

Copyright

Dedication

I dedicate this work to my lovely family: my mother, and my husband, who has always been patient and supportive of me. They always remember me in their prayers. Also, I dedicate this work to my brothers and sisters who have supported me throughout my research project. I hope I have made you proud.

Acknowledgments

In the Name of Allah, the Most Gracious, the Most Merciful. First and foremost, all Praises and Thanks be to Allah for providing me with the health, inspiration, wisdom, and ability complete my Master of Cybersecurity degree. Then, I would like to express my gratitude and my special thanks to my supervisor, Dr. Shahenda Sarhan Her valuable advice, endless support, and continuous encouragement were motivation and guidance for me while working on my research project. My thanks and my appreciation also go to Dr. Abdullah Alshalahi and Dr.Abdulmajeed Moied for their precious feedback which gave me much encouragement and support to enable the completion of this study

My gratitude and my appreciation should go also to my brother Khaled; words are not enough to express how beholden I am to my mother who is always with me and always in my heart. Also, I am grateful for my sibling's and friends' advice. With all of my heart, I always appreciate my loyal husband's endless support until I finished my research project.

Abstract

In the encryption and decryption processes, a cryptographic algorithm is a mathematical function. The plaintext is encrypted using the cryptographic algorithm and a key (a word, number, or phrase). Cryptography allows you to store or transfer sensitive information securely over unprotected networks like the Internet, ensuring that only the intended receiver may see it. encryption with symmetric keys (secret key) Encryption with asymmetric keys (public key). It works because of the encryption algorithm, all potential keys, and all protocols. The strength of the cryptography method and the secrecy of the key are both important factors in the security of completely encrypted data. The problem cryptographic algorithm of attacks and malicious hacking on keys. This research work is trying to address this problem will cryptographic algorithm will be developed to add security features with the data exchanged as per security requirements. Also, it is highly efficient and easy to understand and apply for users.

Key Word: *Cryptography, Algorithm, Encryption, Decryption, symmetric, asymmetric.*

Contents

List of Tables

List of Figures

Glossary

RSA	Rivest Shamir Adleman.
IV	Initialization Vectors.
SSL	Secure Socket Layer
TLS	Transport Layer Security.
RC4	Rivest Cipher 4.
RC5	Rivest Cipher 5
DES	Data Encryption Standard.
AES	Advanced Encryption Standard.
IP	Initial Permutation
LPT	Left Plain Text.
RPT	Right Plain Text.
FP	Final Permutation
3DEA	Triple Data Encryption Algorithm.
ECC	Elliptic Curve Cryptography
ECDSA	Elliptic Curve Digital Signature Algorithm.
SLEA	Simple Lightweight Encryption Algorithm entitled.
DLS	Dynamic Light-weight Symmetric.
CRSA	Catalan Rivest Shamir Adleman.
IDE	Integral Developing Environment.
JVM	Java Virtual Machine.

Chapter 1

Introduction

1.1 Introduction

Cryptography algorithms Encryption and decryption algorithms are sets of operations or rules that are used to encrypt and decode communications. In plain words, they're procedures that safeguard data by preventing unauthorized access. The word cryptography is derived from the Greek word cryptography. It is divided into two parts: "cryptography" means "writing" and "crypto" means "hidden, secret." Data integrity, confidentiality, authentication, and non-repudiation are examples of methods for ensuring information security in the presence of third parties. [11]

Encryption is used in the majority of cryptography techniques, allowing two parties to communicate while preventing unwanted third parties from understanding their messages. The process of transforming human-readable plaintext into unreadable ciphertext is known as encryption. After that, the encrypted data is decoded

so that the intended receiver may read it. Encryption and decoding both employ algorithms.

An encryption algorithm is used to encrypt data and describes how the message should be encoded. Any intruder who sees the encrypted text should be unable to figure out what the original communication was. The ciphertext can only be decoded by an authorized party using a secret decryption key.[10]The encoding of information in cryptography follows the mathematical hypotheses and some arithmetic operations described as algorithms. The data is sent encrypted so that the original data is hard to find. These sets of rules are used in digital signature procedures, authentication to secure data, encryption key development, and the protection of all your financial transactions. Mostly, organizations follow cryptography to align with the goals of :

- **The message's confidentiality** as only an authorized recipient must be able to decrypt a message's content it from encrypted state. As a result of attempts to keep uncontrolled access to encrypted data hidden, halted, or postponed.
- **Integrity** That recipient must be able to identify if the communication has been tampered with.
- **Authentication of the sender** To confirm the emitter's claims or the recipient's expectations, a receivers must be able to authenticate the sender's identity, origin, or path travelled (or combination) from the message.
- **The sender's non-repudiation** The remitter may be

unable to object to the message being sent.

Not that all cryptographic systems accomplish all of the aforementioned objectives. Some cryptographic applications have various aims; for example, some cases need repudiation, where a participant may credibly deny being the sender or recipient of a message, or expand those goals to include variants such as: message access control: Which are the message's legitimate recipients?

Availability of messages: By allowing the message, channel, emitter, or recipient's validity to be limited in time or space.

1.2 Cryptography algorithms

Cryptography algorithms exist in many different forms and sizes, but the vast majority of the rest of these fall into one of two categories: symmetric or asymmetric. Some systems, however, mix the two classes. Symmetric encryption methods, also known as shared-key or symmetric-key algorithms, are using a key that is only known by that of the two people who have access to it. A message is encrypted and decrypted that use the same key, whether they are implemented as block ciphers or stream ciphers.

1.2.1 Cryptography algorithms Classifications

Strictly speaking there are two types of cryptographic algorithms: symmetric key and asymmetric key ciphers. Classical and Modern Symmetric Key Cryptography are two types of symmetric key

cryptography. Transposition Cipher and Substitution Cipher are two types of classical cryptography. Modern cryptography, on the other hand, is separated into Block Cipher and Stream Cipher. Figure 1.1 shows the *Cryptography Classification*

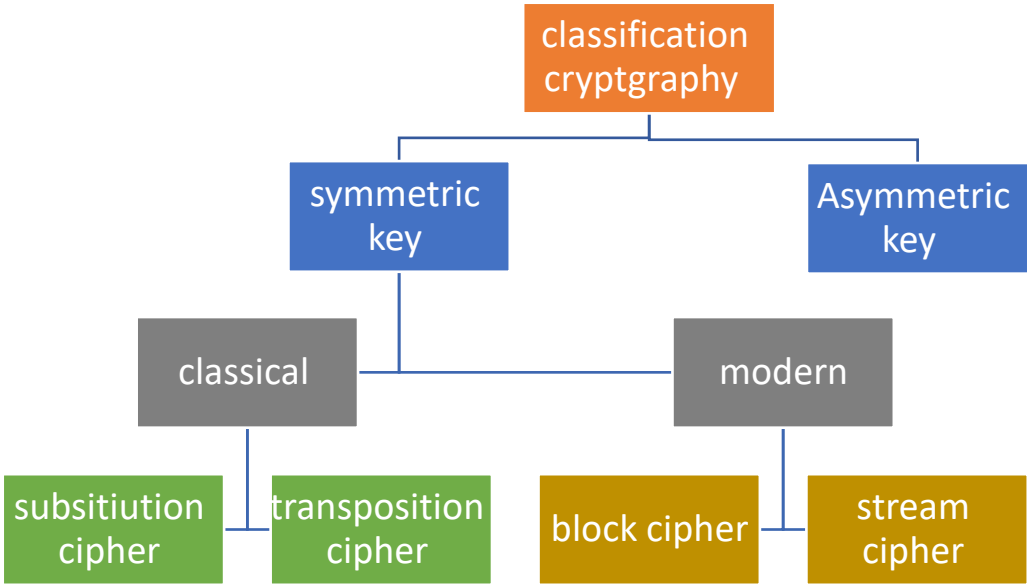

Figure 1.1 *The Cryptography Classification*

1.3 Symmetric key cryptography

Symmetric key cryptography it is a kind of encryption in which the same key is used to encrypt and decode the message. Asymmetric or public-key cryptography, on the other hand, encrypts the communication with one key and then decrypts it with another.

Symmetric encryption is a type of computerized cryptography that conceals the contents of an electronic message by using a unique encryption key. Its data conversion uses a mathematical technique and a private key, which means that someone who doesn't have the

appropriate equipment to decode a message may be unable to understand it. Symmetric encryption is a two-way procedure since the mathematical process then reversed while decrypting the message and the same private key is employed. Symmetric encryption is sometimes known as private-key encryption or safe-key encryption. This is the most basic kind of encryption since it simply uses one private key to cipher and decode data. Symmetrical encryption is an old and well-known method. A private key is used, and it might be a number, a word, or a random string of characters. It's mixed in with plain text in a message to alter it content in some way. Both the sender and the recipient should have access to the private key used it to encrypt and decode all messages.

1.3.1 Symmetric Cryptography Types:

1.3.1.1 Modern Cryptography

1. **Blocks cipher** Encrypting blocks for electronic data will be done via block algorithms. While continuing to utilize the defined private key, specified set lengths for bits are changed. After then, For each block, this key is used. When data from the a network stream is encrypted, the cryptography keeps it in that memory component while waiting for the entire block to arrive. The length of time that the system waits can cause a security vulnerability, putting data integrity and security at risk. Reduce the size of the data block and merge this with the contents of previous encrypted data blocks until the rest of the blocks arrive to fix the problem. This method is known as feedback. It is

encrypted only when the entire block has been received.

Data in plaintext is converted in to the ciphertext in fixed-size blocks using block ciphers. The block size is usually in octaves and is determined by the encryption algorithm (64-bit or 128-bit blocks). The encryption algorithm employs padding to ensure entire blocks if the plaintext length is not a multiple types of 8. To conduct 128-bit encryption on a 150-bit plaintext, for example, the encryption algorithm offers two blocks, one with 128 bits and the other with the remaining 22 bits. To make block equal to the encryption scheme's ciphertext block size, 106 redundant bits are added to last block. While block ciphers encrypt and decode data using symmetric keys and algorithms, it also required an initialization vectors (IV) to work. An initialization vectors is a pseudorandom or random sequence of characters that is used to encrypt the plaintext block's first block of characters. The initialization vector for the succeeding blocks is the resulting ciphertext for the first block of characters. As a result, each iteration of the symmetric cipher generates a unique ciphertext block, but the IV is delivered together with the symmetric key it does not need encryption. Block encryption techniques have strong dispersion, which means is that if a single plaintext block is exposed to several encryption rounds, each repetition produces a distinct ciphertext block. Because it is difficult for hostile actors to inject symbols into the a data block without being detected, the encryption system is largely tamper-proof. Block ciphers, on the other hand, have a high error propagate rate because even minor changes in the plaintext result in completely new ciphertext blocks.

2. **Stream cipher** is the second kind of cipher. Stream algorithms are present in data stream algorithms and are not kept in the encrypted system's memory. Because the data is not kept on a disk or in a system without encryption, this technique is seen to be marginally safer. By applying time-varying changes to plaintext data, a stream cipher encrypts a continuously string of binary numbers. As a result, this method of encryption operates bit by bit, generating ciphertext for plain text communications of any length utilizing keystreams. The keystream – a pseudorandom integer XORed with the plaintext to form ciphertext — is created by combining a key (128/256 bits) and a nonce digits (64-128 bits) in the cipher. While the key and nonce can be reused, each encryption iteration's keystream must be unique to maintain security. To build the keystream, stream encryption ciphers employ feedback shift registers to generate a unique nonce (number used only once). Because a mistake in the translating of one bit does not often influence the whole plaintext block, encryption systems that employ stream ciphers are less prone to spread system-wide problems. Stream encryption is also linear and continuous, making it easier and quicker to deploy. Stream ciphers, on the other hand, lack dispersion since each plaintext digit is remapped to a single ciphertext output. Furthermore, they do not verify validity, leaving them open to insertions. If the encryption algorithm is broken, hackers can inject or change the encrypted message without being detected. Stream ciphers are typically used to encrypt data in situations where its amount of plain text cannot be known and when latency is a concern.

1.3.1.2 Classical Cryptography

1. **Cipher of Transposition** The transposition cipher really is an encryption method in which plaintext units (typically letters, groups (characters)) are shifted by a regular pattern, resulting in a ciphertext that is a permutation of the plaintext. That is, the order of the units is changed (the plaintext is reordered). The bijective function is used to encrypt the position of the letters, and an inverse function is used to decrypt.
2. **Substitution Cipher** Units of plaintext are substituted by ciphertext in this kind of encryption, which might be single letters (its most common), letter pairings, letter triplets, or combinations of the aforementioned.

1.3.2 Symmetric Cryptography Algorithms :

- **Blowfish**

  Blowfish is a public-key cryptosystem with symmetric keys. Bruce Schneier invented the algorithm in 1993. Blowfish is also a block cipher that needs 16 rounds to finish its encryption of the a block with a block size of 64 bits. by that turn of the century, Scheiner had an ideal goal in mind: to present the world with a safe, unpatented, and publicly accessible encryption method [16]. Bruce Schneier created Blowfish, a symmetric block cipher. Blowfish use 64-bit blocks followed by a key length ranging from 32 to 448 bits. Twofish, invented by Bruce Schneier, performs a similar operation on 128-bit blocks. The figure above shows that how Blowfish algorithm works at a high level:

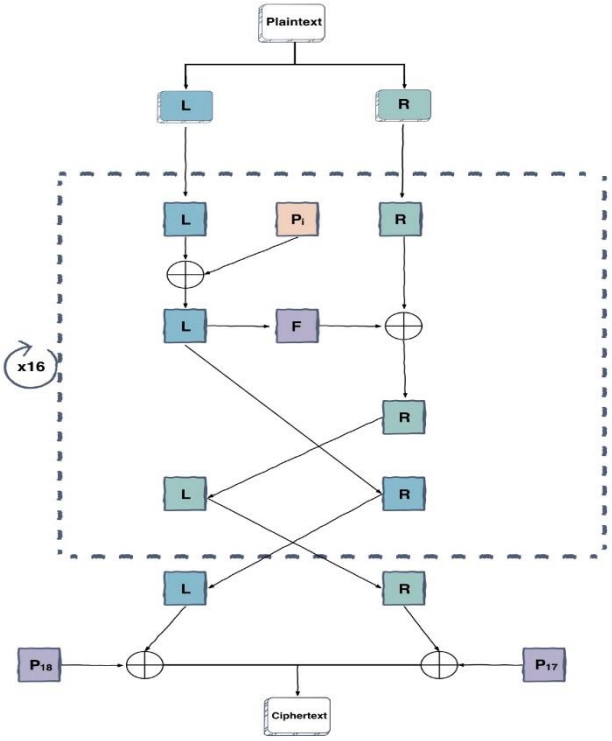

Figure 1.2 The Blowfish Algorithm Works

- **AES (Advanced Encryption Standard)**

  An Identify (pronounced Rhine-doll) algorithm is used in (AES), a newer better encryption standard. Joan Daemen and Vince Rijmen of Belgium created this algorithm. DESX and 3DES will ultimately be replaced by AES. AES may employ keys with lengths of 128 bits, 192 bits, or 256 bits.

- **RC4 is a four-letter code that stands for (Rivest Cipher 4)**

  RC4 is a byte-oriented stream cipher with a changeable key size. The method is based on randomized permutation and is frequently used for SSL-encrypted transmission to and from secure Web sites.Ron Rivest designed Rivest Cipher 4 (RC4) for RSA Security in 1987. It's a Cipher Stream, after all. Stream Ciphers function on

a stream of data byte by byte. RC4 stream encryption is one of the most widely used stream ciphers due to its simplicity and speed of operation. It's a byte-oriented stream cipher with a variable-size key. It supports both 64-bit and 128-bit keys. Secure Socket Layer (SSL) and Transport Layer Security (TLS) applications, as well as IEEE 802.11 wireless LAN standards, use it often. Encryption can prevent unauthorized data access. Third parties will not be able to access data that we give or receive if we encrypt it. The encryption is carried out via a secret key, or more precisely, a public key and a private key. Both the sender and the receiver have their own public and private keys, which are used to encrypt plain text and decrypt ciphertext.

- **DES (Data Encryption Standard)**

  Horst Feistel's LUCIFER block cipher, designed by IBM cryptography researcher Horst Feistel in 1971, provides the foundation for DES. The Feistel structure is used by DES in 16 rounds, each with its key. DES was approved as the official government encryption standard in November 1976, and it was confirmed in 1983, 1988, and 1999. The Advanced Encryption Standard (AES) replaced the DES encryption algorithm as the authorized standard in 2002 after a public competition, thereby ending the DES rule. FIPS 46-3 (the 1999 reaffirmation) was formally canceled by the NIST in May 2005, however, Triple DES (3DES) is still approved for sensitive government data until 2030.

- **RC5 (Rivest Cipher 5)**

  Ronald Rivest developed these algorithms (RSA Labs). Encryption schemes with variable keys and block sizes are forbidden. When attempting to decode seized data, if the attacker does not know the original sizes, it is tough to break.

  Ron Rivest is the creator of RC5. It's a symmetric key-based block encryption technique. The key advantage is that it is fairly quick because it just employs rudimentary computer functions. It adds versatility by allowing for a configurable number of rounds and a variable bit size key. Another benefit of adopting RC5 is that it takes up less RAM to run. This functionality allows RC 5 to be utilized for a variety of tasks, including desktop operations, smart cards, and so on.

  The size of the input plain text block, the number of rounds, and the key's 8-bit bytes can all be configurable in the RC5 method. Once the settings for this are set, they will not change for any further executions of the cryptographic algorithm. Plain text blocks can be 32 bits, 64 bits, or 138 bits in size. The key's length can range from 0 to 2040 bits. The RC5 output is ciphertext, which is the same size as plain text. The plain text message is broken into two 32-bit blocks A and B in RC5. Then S[0] and S[1] are produced as subkeys. These two subkeys are correspondingly added to A and B. The one-time operation comes to a conclusion with this procedure, which creates C and D. After then, the round procedure begins. The following operation is carried out in each round.

- XOR is a bitwise operation.
- Shift to the left circularly.
- For both C and D, the following subkey is added. This is the addition operation, which is followed by the addition mod 2w result.

- **DES (3DES)Triples**

  DES was used to produce Triple-DES, a 64-bit key with 56 effective key bits and 8 parity bits. In 3DES, the plaintext is encrypted three times with DES. The plaintext is encrypted by key A, then decrypted by key B, and encrypted again by key C. The 3DES encryption algorithm is based on block encryption. Although the full term is Triple Data Encryption Algorithm (3DEA), it is most often known as 3DES. This is because the Data Encryption Standard (DES) cipher is used three times in the 3DES approach. DES is a symmetric-key algorithm based on the Feistel network. It's an asymmetric key cipher, which means the encryption and decryption keys are the same. The Feistel network renders each of these processes almost identical, resulting in a more efficient technique to implement.

1.3.3 Advantages and Disadvantages of Symmetric Key Cryptography:

Advantages Symmetric Key:

- Using a symmetric cryptosystem is more efficient.
- Symmetric Cryptosystems allow encrypted data to be sent via a network even if the data will be intercepted. Because no key is included with the files, data decryption is impossible.
- To verify the presence of the receiver, the symmetric cryptosystem utilizes password authentication.
- Communication can only be decrypted by a device with a hidden key.
- Prevent security flaws in large-scale communications systems. To communicate with each side, a unique secret key is used. Only communications from the a specific pair of sender and recipient are impacted when a key is compromised. It is always safe to interact with others.
- Setting up this form of encryption is simple. Users just need to give and exchange the secret key to begin encrypting and decrypting conversations.
- Encrypt and decrypt your data with encryption and decryption software. You don't need to establish different keys if you use encryption for communications or data that you only wish to access once. For this, single-key encryption is suitable.
- Symmetric key encryption is significantly faster than asymmetric key encryption.
- Improves the efficiency of computational resources. When

compared to public-key encryption, single-key encryption uses fewer processing resources.

Disadvantages Symmetric Key:

- Key transportation is a concern in symmetric cryptosystems. The secret key must be supplied to the receiving device before the final message is sent. Electronic communication is unsafe, and no one can be sure that their networks won't be tapped. As a result, sharing keys is only safe when done face to face.
- It's impossible to imagine digital signatures that can't be reversed.
- The message's origins and reliability cannot be ensured. Because both the sender and the recipient have the same key, communications cannot be authenticated as coming from a specific person. If there is a disagreement, this may be problematic.
- For communications between the parties, a new shared key must be created. This makes it difficult to manage and secure both of these keys.

1.4 Asymmetric key cryptography

Asymmetric cryptography, often known as public-key cryptography, encrypts and decrypts a message using a pair of linked keys — one public key and one private key — and protects it from unwanted access or use. The public key is a cryptography

key that anybody may use to encrypt a message such that only the intended recipient's private key can decode it. Only the key's initiator has access to a private key, also known as a secret key. as well when sending an encrypted message, the intended receiver's public key can be obtained from the public directory and used to encrypt the message before sending it. The recipient can then decode the transmission using their associated private key. If the sender encrypts the communication with their private key, only the sender's public key may decode it, allow the sender to be authenticated. Because the decryption and encryption operations are automated, users do not need to manually unlock and unlock the communication.

Many systems, like the transport layer security (TLS) and secure sockets layer (SSL) protocols that enable HTTPS, employ asymmetric cryptography. Software applications that need to construct a network connection across an unsecured network, such as internet browsers, or that need to authenticate a digital signature, employ the encryption process.

When a public key is used to encrypt, the private key that matches it is used for decryption. When a private key is used to encrypt, the public key that matches it is used for decryption.

1.4.1 Asymmetric cryptography's applications :

Digital signatures use asymmetric cryptography to authenticate data. A digital signature is a mechanism for utilizing mathematics to authenticate the integrity and validity of a communication, software, or digital document. It's the digital version of a handwritten signature or a stamped seal. Digital signatures based

on asymmetric cryptography may prove the origins, identity, and status of an electronic document, message, or transaction, as well as recognizing the signer's informed permission. Asymmetric cryptography is also used in systems that encrypt and decode communications for several users, such as:

- **Email that is encrypted.** A public key can encrypt communication and a private key can decode it.
- **SSL/TLS** Asymmetric encryption is also used to build secure communications between browsers and websites.
- **Cryptocurrency** Asymmetric cryptography is used by Bitcoin and other cryptocurrencies. Public keys, which are accessible to everyone, and private keys, which are kept secret, are available to users. Bitcoin uses a cryptographic approach to ensure that money are only accessible to authorized users.

1.4.2 Asymmetric cryptography algorithms :

- **Diffie-Hellman** allowed these keys to be safely transferred through public communication channels, which are commonly utilized by third parties to steal sensitive data and cryptographic keys. The Diffie-Hellman key agreement is depicted in the diagram below.
- **(RSA) Rivest Shamir Adleman** One of the earliest instances of asymmetric encryption is RSA, which was initially published in 1977. RSA encryption was created by Ron Rivest, Adi Shamir, and Leonard Adleman, and it creates a public key and a private

key by multiplying two huge, random prime integers together. The information is encrypted using the public key and decrypted with the private key, just like in traditional asymmetric encryption. Data may be encrypted and signed using this technique. A sequence of modular multiplications are used in the encryption and signing methods.

The core RSA algorithm for secrecy is as follows:

(plaintext)e mod n = ciphertext

(ciphertext)d mod n Plaintext = (ciphertext)d mod n

d, n = Private Key and e, n = public key

The following is a summary of the RSA authentication algorithm:

plaintext = (ciphertext)e mod n ciphertext = (plaintext)d mod n

d, n = private key and e, n = public key

- **(ECC) Elliptic Curve Cryptography** To achieve asymmetric encryption, an RSA variation that employs reduced key sizes and mathematical elliptic curves is shown below. The popular cryptocurrency Bitcoin employs ECC–specifically, the Elliptic Curve Digital Signature Algorithm (ECDSA)–to digitally sign transactions and ensure that only authorized users spend money. In terms of key and signature generation, ECC is significantly quicker than RSA, and many believe it to be the future of asymmetric encryption, particularly for internet traffic and bitcoin, but also for other applications. For smaller devices, like as mobile phones, elliptic cryptography (ECC) is used. When compared to RSA, it requires less computer resources. The goal behind ECC encryption is to create a public/private key combination by utilizing points on a curve.

- **(DSA)Digital Signature Algorithm**

  The US government created the Digital Signature Algorithm (DSA) for digital signatures. The Digital Signature Algorithm can only sign information, not encrypt it. A set of computations based on a prime number are used in the DSA signature mechanism. Longer key sizes are now supported, despite the fact that the maximum key size was meant to be 1,024 bits. The process of establishing a digital signature is faster than verifying it when DSA is employed. When RSA is utilized, verifying a digital signature takes less time than generating one.

- **El Gamal** is a cryptographic technique that is used to send key exchanges and digital signatures. Calculating logarithms is the basis of the procedure. El Gamal's approach is based on the mathematical properties of logarithm numbers. El Gamal is the basis for the Digital Signature Algorithm (DSA).

1.4.3 Advantages and Disadvantages of Asymmetric Key Cryptography

Advantages Asymmetric Key:

1. In asymmetric or public-key cryptography, there is no need to exchange keys, which avoids the issue of key distribution.
2. The fundamental advantage of public-key cryptography is that private keys are never exchanged or made public, resulting in increased security.
3. Digital signatures with the ability to be revoked may be used.
4. Message validation is provided via public-key cryptography,

which demands the use of digital signatures and allows the recipient to verify that the message originated from the same sender.

5. The use of digital signatures in public-key cryptography allows the recipient to determine whether or not the message has been altered with during transit. It is impossible to modify a digitally signed message without rendering the signature null.
6. Signing a message digitally is analogous to signing a paper physically. This cannot be rejected by the sender because it is an acknowledgement of the message.

Disadvantages Asymmetric Key:

1. One downside of using public-key cryptography for encryption is the slowness. The majority of secret-key encrypting methods are far quicker than any publicly accessible public-key encryption solution.
2. Authentication of public keys is required/recommended. Because no one can be certain that a public key belongs to the person it identifies, everyone must double-check that their public keys are theirs.
3. It demands the utilization of more powerful computing resources. Single-key encryption requires a lot more computing resources.
4. A massive security breach is conceivable if an intruder obtains a person's private key and reads his or her whole correspondence.
5. Losing the private key might have long-term ramifications. If a private key is lost, all inbound messages will be deciphered.

1.5 Difference between asymmetric key cryptography vs symmetric key cryptography:

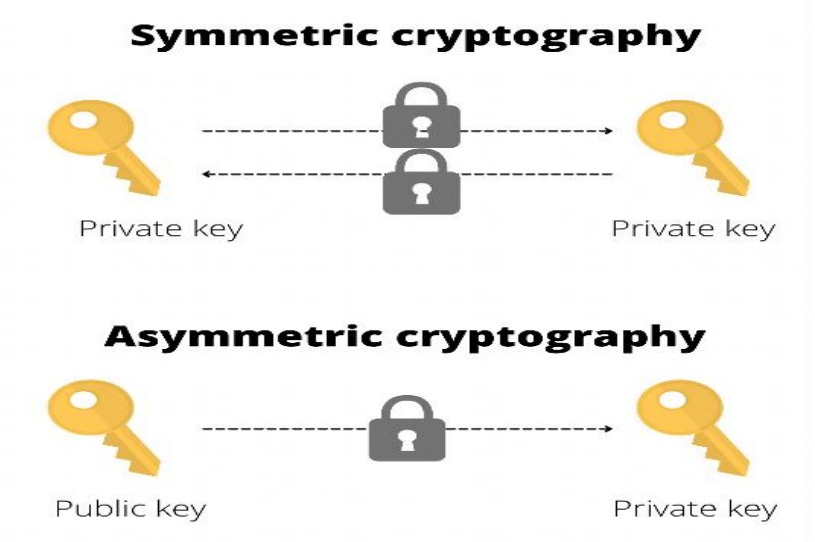

Figure 1.3. asymmetric vs symmetric cryptography

Despite the fact that asymmetric encryption is more advanced than symmetric key cryptography, both are being employed today — and usually in tandem. This is due to the fact that each solution has its own set of advantages and disadvantages. There are two key trade-offs between symmetrical and asymmetrical cryptography: security and speed. Because it does not need the exchange of keys, asymmetric encryption is seen to be more trustworthy.

The user's private key is never revealed or shared. Symmetric key encryption is more time consuming and frequently slower than symmetric encryption. This is not a significant disadvantage, and it is frequently used to encrypt data where confidentiality is critical. A mathematical relationship between a public and private key is required in asymmetric cryptography. Because malicious actors may try to decrypt the data using

this pattern, asymmetrical keys must be longer to give the same level of security. A 2048-bit asymmetrical key and a 128-bit symmetrical key provide nearly equivalent security because to the enormous difference in key lengths. Asymmetric encryption is much slower than symmetric encryption, which is much quicker.

Despite the fact that asymmetric encryption is more advanced than symmetric key cryptography, both are being employed today — and usually in tandem. This is due to the fact that each solution has its own set of advantages and disadvantages. There are two key trade-offs between symmetrical and asymmetrical cryptography: security and speed. Because it does not need the exchange of keys, asymmetric encryption is seen to be more trustworthy. The user's private key is never revealed or shared. Symmetric key encryption is more time consuming and frequently slower than symmetric encryption. This is not a significant disadvantage, and it is frequently used to encrypt data where confidentiality is critical.

Table 1.1 Difference between Asymmetric key and Symmetric key

Comparison	Pros	Cons
Symmetric Encryption	• Performance is relatively high. • It has two aspects which are the encryption algorithm and the encryption key. • It can be implemented directly on devices easily. • More effective software. • Integrity and confidentiality check.	• Weakness of symmetric algorithms in symmetric key sharing between sender and receiver. • Because the secret key is shared with the sender and receiver, it is risky.

	• It helps us verify the efficiency of the proposed method in the safety of the medical image.	
Asymmetric encryption	• It uses two different keys are used for encryption and decryption. • Multiple image encryption successfully. • The ability to initial photos with high quality. • Avoid loss of high-frequency information. • Achieve multi-image encryption with a small amount of data. • Prevent information disclosure. • Enjoy high security. • Resist image retrieval attack. • Effectively prevent information leakage. • It is the best choice in transferring information. • It is used in client-server model communications.	• Cannot use any other key than the private key for decryption. • The public key can be known to anyone because it can only be used for message encryption, therefore anyone can encrypt the message but only the authorized person can decrypt the message using their private key. • The performance is relatively low compared to a symmetric encryption key. • It operates slower than symmetric encryption. • Most algorithms rely on the properties of difficult problems in mathematics and these problems are usually in one direction and impossible in the other.

2. Project Problem

Some symmetric key algorithms require more computing power, which makes the original cipher vulnerable [5]. They'll have a 64-bit data block and a key size ranging from 40 to 128 bits (with an 8-bit increment step) [7]. Asymmetric key algorithms, on the other hand, can have two keys: one for encryption and the other for decryption operations. It consists of three steps: key generation, encryption, and decryption. The key size will range from 1024 to 4096 bits[2].

3. Project Objectives (Motivation)

1) Proposing a novel cryptography algorithm that achieves dispersion and confusion principles while also making the encryption and decryption operations easier for the user.

2) Algorithm time and space complexity must be balanced.

3) Authentication: Before sending or receiving data, the sender and data receiver must be authenticated.

4) Confidentiality: Only authenticated users have access to the communications.

5) Integrity: Data is unaltered between sender and receiver.

6) Non-Repudiation: Neither the sender nor the receiver can dispute that a communication was sent.

7) Service Reliability: Attackers can compromise secure systems, compromising the user's service.

4. Scope of Deliverables

1. A comprehensive study about different cryptography algorithms
2. An implementation of the proposed algorithm
3. Thesis documentation

5. Summary

In this project, we aim to introduce a new cryptography algorithm that tries to achieve diffusion and confusion concepts while preserving the easiness of the user to encrypt and decrypt data. Through this, we will try to balance the algorithm strength and time and space complexity by using one of the optimization algorithms.

The remainder of this thesis is organized as follows.: In Chapter 2. Literature Survey, Chapter 3. Model design of Cryptography Algorithms, Chapter 4. Practical application of Cryptography Techniques, Chapter 5. Conclusion.

Chapter2

Literature Review

Based on what we mentioned in chapter 1 there are two types of cryptography algorithm symmetric and introduce some of the related work in these two types.

2.1 The symmetric encryption

2.1 The asymmetric encryption

2.1 The symmetric encryption

Mushtaq et al introduce a survey on cryptographic encryption algorithms, aimed to address the security aspects and processes involved in the design and implementation of the most widely used symmetric encryption algorithms, such as Data Encryption Standard (DES), Triple Data Encryption Standard (3DES), Blowfish, Advanced Encryption Standard (AES), and Hybrid Cubes Encryption Algorithm (HiSea). Furthermore, the performance of various encryption methods was assessed and compared in terms of encryption and decryption time, throughput, key size, avalanche impact, memory, correlation assessment, and entropy. As a result, by employing an appropriate encryption method based on several criteria that are best suited to the user's requirements. Finally, the results demonstrate that Blowfish is the

best choice in situations where memory and encryption/decryption time are critical, as well as in software. However, AES may be assessed using the avalanche effect, which demonstrates outstanding performance, while HiSea can be assessed using entropy and correlation. According to the findings, the Advanced Encryption Standard, as well as the Hybrid Cube Encryption algorithm, can be used in situations where integrity and secrecy are critical.

(Vengadapurvaja, Nisha, Aarthy& Sasikaladevi, 2017) the study recommended an effective symmetric encryption algorithm to encrypt medical images and perform fruitful operations on them without violating confidentiality. An entire symmetric encryption scheme has been recommended to support the aforementioned symmetric encryption algorithm in both addition and multiplication. The input images, the corresponding encrypted images, capturing the image of digital photography, and communication in the medical field were viewed as input images. After that, the study made analyses, such as key-space analysis, key sensitivity analysis, chart analysis, correlation analysis, peak signal-to-noise ratio analysis, mean squared error analysis, and noise analysis; the results revealed that the analyses conducted assisted us in checking the efficiency of the recommended method in the light and security of medical images.

In the study (Ahmad& Ismail, 2018) entitled User Selective Encryption Method for Securing MANETs. The performance of

symmetric encryption algorithms (DES, 3DES, and AES) have been evaluated in MANET. The study applied a safe option for key management using the Diffie-Hellman Key Exchange (DHKE) protocol, furthermore, it provides to the MANET user the ability to Dynamic selection of the preferred encryption based on the required security level, and several simulation experiments have been carried out. The results show the superiority of AES over data Encryption Standard (DES), Triple Data Encryption Standard (3DES) for all performance measures.

In the study of (Aboshosha, Dessouky& Elsayed, 2019) entitled Energy Efficient Encryption Algorithm for Low Resources Devices, a Simple Lightweight Encryption Algorithm entitled (SLEA) based on addition and subtraction operations and compact Substitution-boxes (S-boxes) is proposed for wireless networks due to its low energy consumption, simple hardware requirements, in addition to a suitable level of security. Its objective is to provide practical and secure encryption for low-resource Applications, and the Algorithm attempts to overcome the limitations of both public key and symmetric protocols, besides, it relies on a smart version of (Feistel) structure, an Algorithm is designed to facilitate the implementation of both software and hardware; furthermore, the study concluded the simplicity of the (SLEA) Algorithm in terms of both the functions and the compact boxes employed. (SLEA) also may be useful for wireless networks, especially in limited resources networks such as (WSN).

In the study of (Banani, Thiemjarus, Wongthavarawat& Ounanong, 2022) This paper offers a Dynamic Light-Weight Symmetric (DLS) encryption method to handle the issues in data security and real-time secure data transfer via message advertising. The algorithm encrypts each transmitting packet with a unique temporal encryption key, as well as a modest additional burden on computing resources, using a basic function like XOR Gate. Over existing baseline encryption techniques, the Digital Subscriber Line (DSL) may greatly improve security. The technique was tested on beacon data encryption across advertising channels to see how well it worked. Furthermore,The tests also demonstrated the usage of DSL encryption on top of several lightweight symmetric encryption algorithms (such as TEA, XTEA, and PRESENT) and an MD5 hash function. The experimental findings suggest that with a small increase in resource utilization, the (DLS) may obtain acceptable results for avalanche effect, key sensitivity, and unpredictability in ciphertexts. The proposed DLS encryption algorithm is suitable for application-layer implementation, is light and energy-efficient, reduces or eliminates the need for secret key exchange between sensor nodes and the server, applies to dynamic message size, and protects against known-plaintext, brute-force, replaying, and differential attacks.

Commenting on symmetric encryption studies:

Through the aforementioned data on symmetric encryption studies, the following is concluded:

- (Mushtaq, Jamel, Disina, Pindar, Shakir& Deris, 2017) study stated that the blowfish algorithm is the best algorithm in terms of memory

and encryption time.

- (Vengadapurvaja, Nisha, Aarthy& Sasikaladevi, 2017) study results viewed that symmetric encryption assists us in checking the security of medical images.
- (Ahmad& Ismail, 2018) study results showed the superiority of the advanced encryption standard over other algorithms.
- (Aboshosha, Dessouky& Elsayed, 2019) study concluded that the algorithm used in the study is simple in terms of the functions it's used for, and it's also useful in wireless networks.
- (Banani, Thiemjarus, Wongthavarawat& Ounanong, 2022) study viewed that digital subscriber line encryption algorithm is more suitable than other algorithms.
    - The aforementioned studies' results agree that symmetric encryption is quicker in data transmission.

2.2 The asymmetric encryption

The study of Chen, Liu, Wang& Wang, 2018) has proposed an asymmetric encryption system that relies on Compressed Sensing and combined features, the spectra of four gray images are respectively obtained through the optical wavelet transform, By the various properties of the factor of the wavelet transform. the high-frequency and low-frequency components of four images are combined in the form of both high-frequency and low-frequency fusion images respectively. the high-frequency fusion image is then divided into two matrices after being measured by the compressed sensing, then both the matrices and the low-frequency

fusion image are cyphered into a single ciphertext through phase-truncation within the Fresnel domain. The findings have concluded that the proposed system can't only encrypt the multiple images successfully; but also can recover the primary images with high quality to avoid losing the high-frequency information, furthermore, it has concluded the safety and effectiveness of the proposed method.

While in the study of (Wu, Wang, Chen, Wang& Wang, 2019) that is entitled the Asymmetric encryption of multi-image based on compressed sensing and phase-truncation in the cylindrical diffraction domain, an asymmetric encryption algorithm has been proposed for the encryption of multi-image; based on the compressed sensing and the cylindrical diffraction domain. These multi and grayscale images have been compressed through the discrete compressed sensing, then the single capacity ciphertext is accessed through the asymmetric processing of phase-truncation after the cylindrical diffraction domain. The findings have refereed that this method can not only achieve multi-image encryption with a small amount of data, but it can also prevent the disclosure of information, moreover, this proposed method has high security because it can resist Phase retrieval attacks. The numerical findings have shown the safety and effectiveness of the proposed method.

The study of (Wu, Hu, Wang, Wang& Wang, 2019) has suggested a method to compress and encrypt the scalable and asymmetric

images; based upon the discrete wavelet analysis and nonlinear processes in the cylindrical diffraction domain; the grayscale image is processed by the discrete wavelet analysis, and then four generated images can be obtained, then the single capacity ciphertext is accessed through the asymmetric processing of phase-truncation after the cylindrical diffraction domain. The findings have concluded that the proposed method is meaningful and effective within encryption, and it effectively prevents the leakage of information, even in case of another leakage of a private key, the most significant information can't lie within that key. The three encryption models have a high degree of security, and decryption can be selected without any loss of data.

(Alarifi, Amoon, Aly& Shafai, 2020) the study aimed to deploy and use asymmetric optical encryption (phase-truncated Fourier transform) to achieve citizens' broadband radio service. The new citizens' broadband radio service was provided on an asymmetric encryption basis (phase-truncated Fourier transform), and the algorithm recommended provides both impacts of confusion and diffusion decryption on biometric templates. Experiments were conducted to validate the promising achievement of the recommended asymmetric encryption algorithm in mixing biometric templates. The results viewed that the algorithm is suitable and recommended to obtain trustworthy biometric images in comparison to optical encryption algorithms. Citizens broadband radio service provides results on receiver operating characteristic curve, PDF, energy efficiency ratio, FAR, free-range

routing, structural similarity index measure, PTD, visibility, histogram, runtime, and correlation. The recommended algorithm has proven its ability to mix different biometric model datasets properly while having various features.

The study of (Cheremkhin, Evtikhiev, Krasnov, Rodin, Shifrina& Starikov, 2020) has developed a new optical asymmetric encryption method depending on double encryption using spatially incoherent illumination to obtain a ciphertext. furthermore, the plain text was encrypted twice, first by the sender and again then by the recipient; So that encryption keys are not required to be exchanged. Two different types of this method were analyzed, one in which both encryption procedures are performed optically and the other in which the first process is performed visually and the second is digital. In addition, numerical and optical experiments were conducted. The results in the numerical and optical experiments showed a high quality in images that have been decrypting, and this confirms the applicability of the proposed method.

Commenting on asymmetric encryption studies:

Through the aforementioned data on asymmetric encryption studies, the following is concluded:

- the study of (Chen, Liu, Wang& Wang, 2018) has concluded that the asymmetric encryption system can encrypt the multi-image and recover the primary one with high quality, moreover, it is safer. The findings of the (Wu, Wang, Chen, Wang& Wang, 2019) study have concluded that an asymmetric encryption algorithm can

encrypt a multi-image with a small amount of data, it can also prevent the disclosure of data. Asymmetric encryption has high security and can resist Phase retrieval attacks.

- the findings of (Wu, Hu, Wang, Wang& Wang, 2019) study have concluded that asymmetric image compression and encryption are meaningful and effective within encryption, and effectively prevent the leakage of information.
- the findings of (Alarifi, Amoon, Aly& Shafai, 2020) study have referred that the citizens' broadband radio algorithm is appropriate to get reliable biometric images, compared to optical encryption algorithms.
- the study of (Cheremkhin, Evtikhiev, Krasnov, Rodin, Shifrina& Starikov, 2020) has concluded that the numerical and visual experiments are of high quality for the images that are decrypted.
    - The findings of the literature review have agreed that asymmetric encryption is safer within the transfer of data, keeping the privacy of information, and preventing its leakage.

Chapter 3

The Proposed Cryptography Algorithm

3.1 Introduction

Asymmetric key cryptography, often known as public-key cryptography, uses two separate keys. For encryption, only one key is used, whereas decryption requires the use of the other key. RSA is a well-known public-key cryptography algorithm. It was the first approach that could be used for both signing and encryption, and it was one of the first important innovations in public-key cryptography. The security of the RSA cryptosystem is based on two mathematical problems: factoring large numbers (a mathematical attack) and trying all possible private keys (known as a brute force assault). Then, to overcome the key generation problem, we updated the RSA approach and employed Catalan Numbers..

3.2 The RSA Algorithm

In 1978, Ron Rivest, Adi Shamir, and Leonard Adleman founded RSA. RSA. It is also one of the most well-known public-key cryptosystems for key exchange, digital signatures, and data encryption. The RSA algorithm is a sort of asymmetric encryption. It is asymmetric because it employs two independent keys: a Public Key and a Private Key. In asymmetric approaches, the key used for encryption is distinct from the key used for decryption, as the name indicates. A decryption key cannot be generated from an encryption key in an acceptable period, and vice versa. RSA is a block cipher system with an asymmetric (public key) cryptosystem based on number theory that uses a variable-size encryption block and a variable-size key. Encryption and decryption are accomplished using these two distinct keys. When the communication is transmitted to the recipient, the sender encrypts it with the receiver's public key, and the receiver decrypts it with his private key. The RSA procedure has three main steps: key generation, encryption, and decryption. The RSA principle is based on the difficulty of factoring in a big number. Two integers make up the public key, one of which is the result of multiplying two huge prime numbers. The private key is also made up of the same two prime numbers. If someone can factorize a huge integer, the private key is compromised. As a result, encryption strength is determined by key size, and increasing key size enhances encryption strength exponentially. The length of an RSA key can range from 512 to 4,096 bits.

RSA is not recommended for commercial usage because to several problems in its design. When microscopic values of p and q are used in the key design, the encryption mechanism becomes too weak, and the data can be deciphered using random probability theories and side-channel assaults.

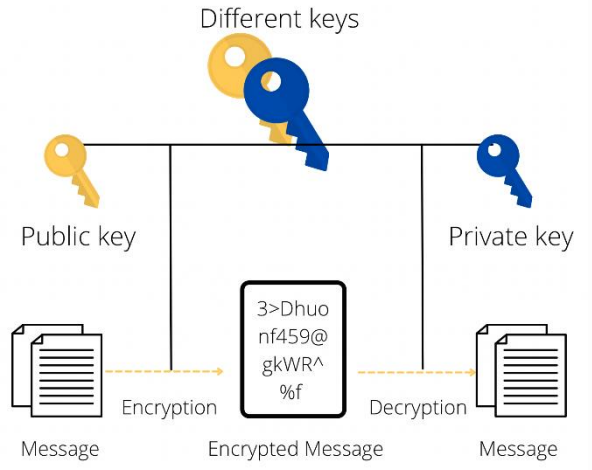

***Figure 3.1**. Encryption key*

3.2.1 How RSA Works

The RSA contains three main procedures Key generation, encryption and decryption as follows:

1- Procedure for Creating a Key [14]

- Choose two large random prime integers, p, and q, such that p and q are equal.
- Calculate n=p × q.
- Make the following calculations: phi (n) = (p-1) (q-1).
- Select the integers e that 1< e< phi (n)
- Calculate d to satisfy the congruence relation d e = 1 mod phi

(n); d is kept as a private key exponential function.

- The public key is (n, e), and the private key is (n, e) (n, d).
- Hide the d, p, q, and phi variables entirely.

2- Encryptions

- Given plain-text P and cipher-text C
- $C = P^e \bmod n$

3- Decryption

- $P = C^d \bmod n$

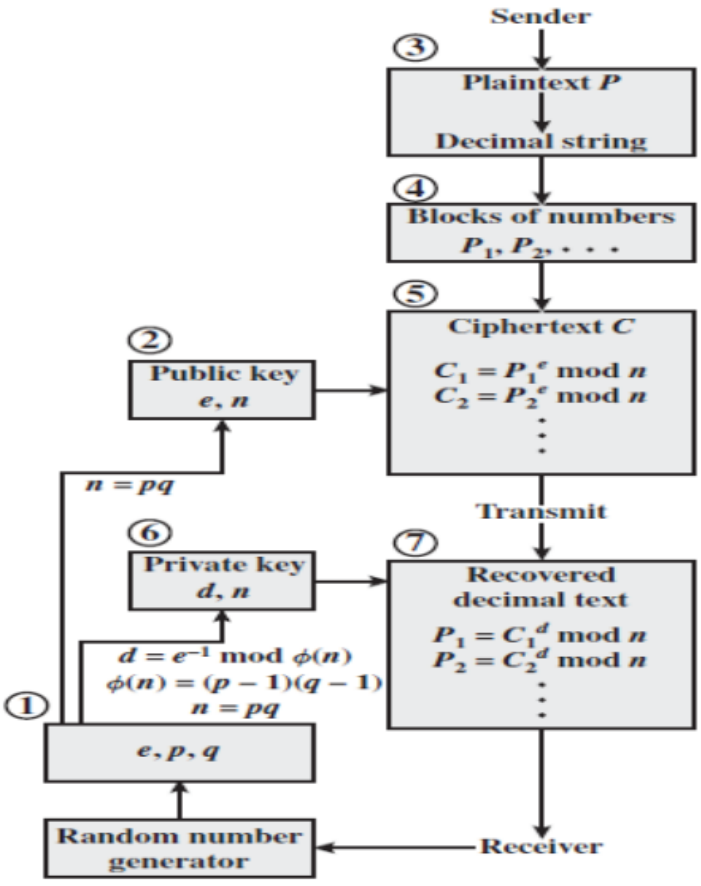

Figure 3.2. RSA algorithm processing of Many Blocks.

From figure 3.2 we can explain The stages of the RSA algorithm are as follows:

- Choose p and q, two enormous prime numbers.
- Multiply these numbers to yield n = p x q, where n is the number of encryption / decryption modules.
- Choose a number e smaller than n such as n is close to (p - 1) x. (q -1). It indicates that except for 1, e and (p - 1) x (q - 1) have no commonality. Select "e" so that 1e (n), e are prime to (n), and gcd (e,d(n)) =1.
- If n = p x q, the public key is. A plaintext message m is encrypted using the public key. To produce ciphertext C from plain text, apply the following equations: C = me mod n
- In this scenario, m must be less than n. A message of a length more than n is treated as a series of messages, each of which is encrypted separately.
- The following formula is used to calculate the d to identify the private key:

  (p - 1) x (q - 1) = 1 De mod Alternatively, De mod (n) = 1
- The secret key is The private key is used to decrypt a ciphertext message c. The following equations are used to compute plain text m from ciphertext c. cd mod n = m

3.2.2 RSA Example

The public and private keys of an RSA cryptosystem are generated by using two prime numbers, 13 and 17, respectively. If A's public is 35 people. Then A's private key is? Explanation:

Step 1: Choose two huge primes, p, and q, as the first step.

q = 17 p = 13

Step 2: Multiply those integers to get n = p x q, where n is the encryption/decryption modules.

n = p x q

n = 13 x 17

n = 221

Step 3: Select an integer e smaller than n such as n is close to (p - 1) x (q -1). It indicates that e and (p - 1) x (q - 1) have no common factor except 1. Select "e" in such a way that $1<e<\varphi(n)$, e is prime to $\varphi(n)$, gcd (e, d (n)) =1.

Second, we do the calculations.

$\varphi(n) = (p - 1) \times (q-1)$

$\varphi(n) = (13 - 1) \times (17 - 1)$

$\varphi(n) = 12 \times 16$

$\varphi(n) = 192$

g.c.d (35, 192) = 1

Step 4: To find a private key, we apply the algorithm to compute the d as follows:

Calculate $d = d^e \bmod \varphi(n) = 1$

$d = d \times 35 \bmod 192 = 1$

$d = (1 + k.\varphi(n))/e$ [let k =0, 1, 2, 3……]

Put k = 0

$d = (1 + 0 \times 192)/35$

d = 1/35

Put k = 1

$d = (1 + 1 \times 192)/35$

d = 193/35

Put k = 2

d = (1 + 2 x 192)/35

d = 385/35

d = 11

The private key is <d, n> = (11, 221)

Hence, private key i.e. d = 11

3.2.3 RSA Disadvantages

Unfortunately, RSA has had to keep up with crypto inflation and the advancement of cracking gear. So, with 512-bit RSA keys entirely untrustworthy and 1,024-bit keys following suit, we now have a gigantic 2,048-bit key. Because the CPU must cope with large numbers, this has a big impact. As a result, RSA is still used in digital certificates and identity verification. This is because the cost in the signature process is minimal, but the mechanism for key exchange is outdated in our modern world. So we can summarize the disadvantages of RSA in

- It may occasionally fail since complete encryption requires both symmetric and asymmetric encryption, whereas RSA simply uses symmetric encryption.
- Due to the vast number of people participating, the data transfer rate is sluggish.
- A third party is often required to validate the trustworthiness of public keys.
- Decryption necessitates a significant amount of effort on the receiver's part.

3.3 The proposed Catalan Rivest Shamir Adleman algorithm (CRSA)

From the RSA disadvantages we found that the key generation procedure is one of the main problems that RSA face so in this section we propose a new cryptography algorithm based on modifying the key generation procedure of the RSA using Catalan Numbers.

3.3.1 The Catalan Numbers

Catalan numbers are a collection of positive integers that appear in a variety of combinatorial counting problems. They are named after French-Belgian mathematician Eugène Charles Catalan (1814–1894). They store lattice paths, permutations, binary trees, and a variety of other combinatorial objects. They follow a simple recurrence connection and have a closed-form formula in terms of binomial coefficients.

By explicitly expressing the nth Catalan number in terms of binomial coefficients

$$C_n = \frac{(2n)!}{(n+1)!n!}$$

The Catalan number C_n is the solution for

- The number of proper bracket sequences (n opening and n closing brackets).
- The number of entire binary trees with n+1 leaves that are rooted (vertices are not numbered). If every vertex has either two or no offspring, a rooted binary tree is full.
- The number of methods to parenthesize n+1 components fully.

- A convex polygon with n+2 sides has how many triangulations? (i.e. the number of partitions of the polygon into disjoint triangles by using the diagonals).
- The number of ways to build n disjoint chords by connecting the 2n points on a circle.
- The number of entire binary trees with n internal nodes that are non-isomorphic (i.e. nodes having at least one son).
- The number of monotonic lattice paths from point (0,0) to point (n,n) in a square lattice of size n×n, which do not pass above the main diagonal (i.e. connecting (0,0) to (n,n)).
- Number of permutations of length n that can be stack sorted (i.e. the rearrangement may be demonstrated to be stack sorted if and only if there are no such index $i<j<k$, such that $a_k<a_i<a_j$).
- A set of n items has how many non-crossing partitions

n	C_n	n	C_n	n	C_n
1	1	11	58,786	21	24,466,267,020
2	2	12	208,012	22	91,482,563,640
3	5	13	742,900	23	343,059,613,650
4	14	14	2,674,440	24	1,289,904,147,324
5	42	15	9,694,845	25	4,861,946,401,452
6	132	16	35,357,670	26	18,367,353,072,152
7	429	17	129,644,790	27	69,533,550,916,004
8	1,430	18	477,638,700	28	263,747,951,750,360
9	4,862	19	1,767,263,190	29	1,002,242,216,651,368
10	16,796	20	6,564,120,420	30	3,814,986,502,092,304

***Figure 3.3**. Catalan Numbers calculations example.*

3.3.2 The proposed algorithm steps

Through the proposed CRSA we tried to solve the RSA key generation problem by using the Catalan numbers equation for generating the public and private key as follows

1- Procedure for Creating a Key

a. Pick two big random prime integers, p, and q, such that p ≠ q.

b. Determine n= p × q.

c. Determine e using the Catalan Number equation as follows

$$e = \frac{(2p)!}{(p+1)!p!}$$

Where e is a part of the public key, which will be (n,e).

- Determine d using the Catalan Number equation

$$d = \frac{(2q)!}{(q+1)!q!}$$

Where d is a part of the private key which will be (n,d).

2- Encryptions

- Given plain-text P and cipher-text C
- $C = P^e \bmod n$

3- Decryption

- $P = C^d \bmod n$

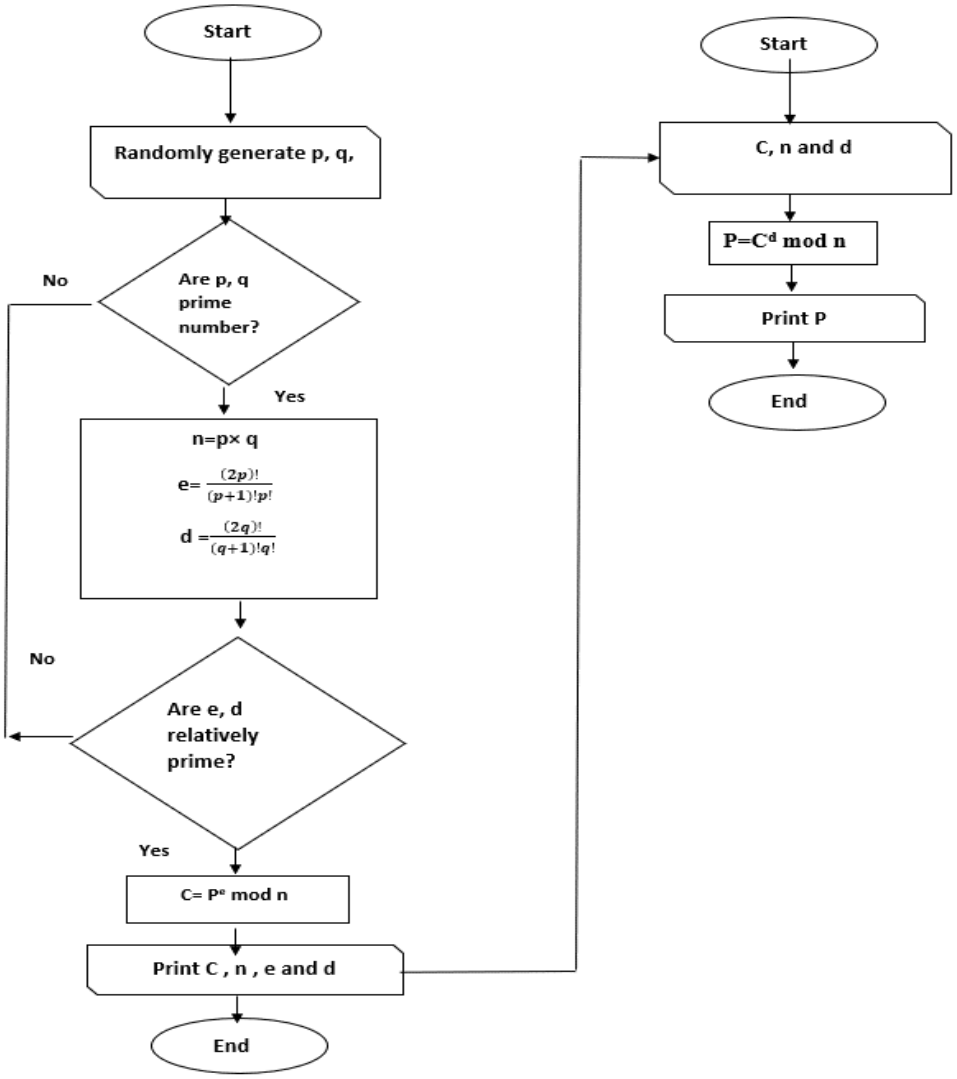

***Figure 3.4**. work calculations Catalan Numbers.*

3.4 The Environment

3.4.1 NetBeans

NetBeans is a Java desktop application platform as well as an integral developing environment (IDE) for creating in a variety of languages, including Java. The NetBeans IDE is designed in Java and may operate on any platform that supports the Java Virtual Machine (JVM), such as Windows, Linux, and Mac.

NetBeans is a free application that allows anybody who installs a computer to experiment with writing code. To test the algorithms and analyze the outcomes. We will show how to apply NetBeans to encrypt and decrypt texts using different Catalan numbers during this project. We will use the RSA algorithm to modify it and solve the key generation problem.

3.4.2 Description of The Environment

The following is a more detailed overview of the basic steps: Choose an algorithm that specifies the type of work that will be done (encryption-decryption), then insert the input data (text - file, picture, etc.) We write code in the application NetBeans and output connector where the result will appear and if everything is as it should be - if not, the program will not allow the completion of connection and will not run - and then shown after the execution result. The results are then analyzed, compared, and utilized in new encryption and decryption techniques.

3.5 Application NetBeans

Examine the application's original picture before making a decision. The horizontal portion on the upper side has menu options for the recognized functions of storage, file selection, updates, settings, and program execution. The toolbox of possibilities is located on the vertical left side. The work area and the implementation of connecting the appropriate parties occupy the remaining space. In Figure 3.5. Application NetBeans.

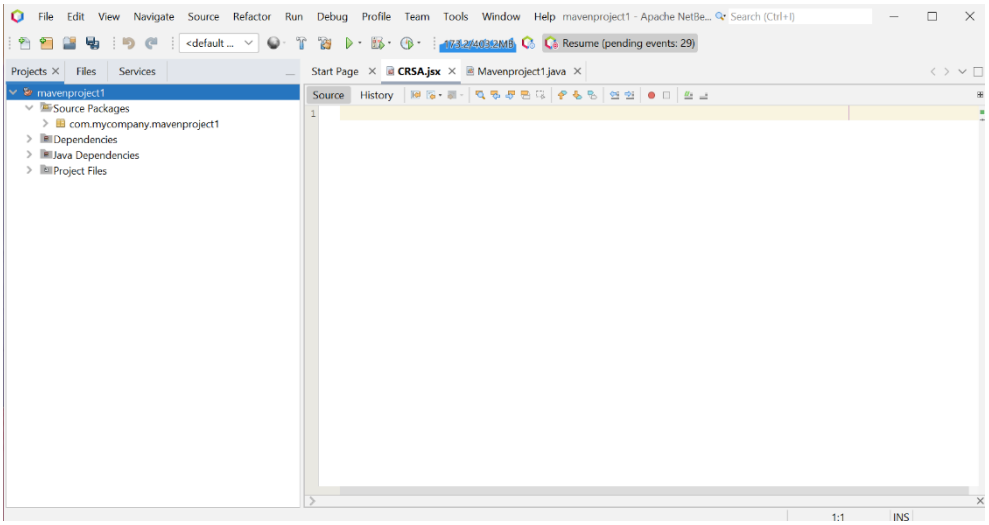

Figure 3.5. Application NetBeans.

Chapter 4

Results of Evaluation

and Experimentation

4.1 Introduction

The suggested method described in Chapter 3 is implemented and examined in this chapter.

4.2 Application of proposed CRSA algorithm

Through the proposed CRSA we tried to solve the RSA key generation problem by using the Catalan numbers equation for generating the public and private keys as follows , In Figure 4.2. Application of proposed algorithm.

1- Procedure for Creating a Key

1. Pick two big random prime integers, p, and q, such that $p \neq q$: We generate two random prime numbers in JAVA, The p and q variables are randomized by the Math.random() function and are in the range of 2 to 128

```java
public class RandomPrimeGenerator {

    public static void main(String[] args) {

        while (true) {
            int pRandom = (int) (Math.random() * (83 - 2) + 2);
            if(isPrime(pRandom)){
                System.out.println("Got Random Prime p :"
                    +pRandom);
                break;
            }
        }
        while(true){
            int qRandom = (int) (Math.random() * (83 - 2) + 2);
            if(isPrime(qRandom)){
                System.out.println("Got Random Prime q:"
                    +qRandom);
                break;
            }
        }

    }

    private static boolean isPrime(int n) {
       int i;
       for(i=2;i<=Math.sqrt(n);i++){
           if(n % i == 0){
              return false;
           }
       }
       return true;
    }

}
```

***Figure 4.1**. Application CRSA algorithm.*

Output

```
ot Random Prime p :73
ot Random Prime q:67
```

***Figure 4.2**. Application CRSA algorithm output.*

- Determine e using the Catalan Number equation as follows

$$e = \frac{(2p)!}{(p+1)!p!}$$

Where e is a part of the public key, which will be (n,e).

- Determine d using the Catalan Number equation

$$d = \frac{(2q)!}{(q+1)!q!}$$

Where d is a part of the private key which will be (n,d).

```
import java.io.*;
import java.util.*;
import java.math.*;

class CRSA
{
    public static BigInteger findCatalan(int n)
    {
        // using BigInteger to calculate large factorials
        BigInteger b = new BigInteger("1");

        // calculating n!
        for (int i = 1; i <= n; i++) {
            b = b.multiply(BigInteger.valueOf(i));
        }

        // calculating n! * n!
        b = b.multiply(b);
        BigInteger d = new BigInteger("1");

        // calculating (2n)!
        for (int i = 1; i <= 2 * n; i++) {
            d = d.multiply(BigInteger.valueOf(i));
        }

        // calculating (2n)! / (n! * n!)
        BigInteger ans = d.divide(b);

        // calculating (2n)! / ((n! * n!) * (n+1))
        ans = ans.divide(BigInteger.valueOf(n + 1));
        return ans;
    }

    // Driver Code
    public static void main(String[] args)
    {
        int n = 5;
        System.out.println(findCatalan(n));
    }
}
```

***Figure 4.3**. Application CRSA algorithm.*

Given plain-text P and cipher-text C

$C = P^e \bmod n$

```
            // e is for public key exponent
            if (gcd(e, z) == 1) {
                break;
            }
        }
        System.out.println("the value of e = " + e);
        for (i = 0; i <= 9; i++) {
            int x = 1 + (i * z);

            // d is for private key exponent
            if (x % e == 0) {
                d = x / e;
                break;
```

***Figure 4.4**. Application CRSA algorithm.*

$P = C^d \bmod n$

```
            }
        }
        System.out.println("the value of d = " + d);
        c = (Math.pow(msg, e)) % n;
        System.out.println("Encrypted message is : " + c);

        // converting int value of n to BigInteger
        BigInteger N = BigInteger.valueOf(n);

        // converting float value of c to BigInteger
        BigInteger C = BigDecimal.valueOf(c).toBigInteger();
        msgback = (C.pow(d)).mod(N);
        System.out.println("Decrypted message is : "
                           + msgback);
    }

    static int gcd(int e, int z)
    {
        if (e == 0)
            return z;
        else
            return gcd(z % e, e);
    }
}
```

***Figure 4.5**. Application CRSA algorithm.*

Output

```
the value of z = 4752
the value of e = 5
the value of d = 1901
Encrypted message is : 4282.0
Decrypted message is : 12
```

Figure 4.6 *Application CRSA algorithm output.*

4.3 Result

We made 10 attempts to result in the mean of the type encryption and decryption to measure the difference in the performance of the proposed algorithm CRSA and RSA.

Table 4.1 : the ten attempts results in terms of time in case of encryption and decryption

Trail No	p	q	n	Time (second)			
				Encryption		Decryption	
				RSA	CRSA	RSA	CRSA
1	97	367	35,599	1.77	1.56	1.97	1.77
2	127	383	48,641	2.65	2.02	2.88	2.45
3	233	409	95,297	3.01	2.67	3.43	3.23
4	353	431	152,143	3. 54	3	3. 76	3.54
5	373	487	181,651	3.89	3.65	4.33	4. 12
6	401	523	209,723	4.23	4.1	4.87	4.79
7	431	587	252,997	5.21	4.87	5.63	5. 41
8	457	619	282,883	6.45	5.99	6.336	6.44
9	547	677	370,319	7.1	6.89	7.24	7.11
10	601	733	440,533	7.9	7.54	7.99	7.87

These The following graph represents the relation between the plain text (message) in kilobytes and the memory usage of the proposed CRSA against RSA.

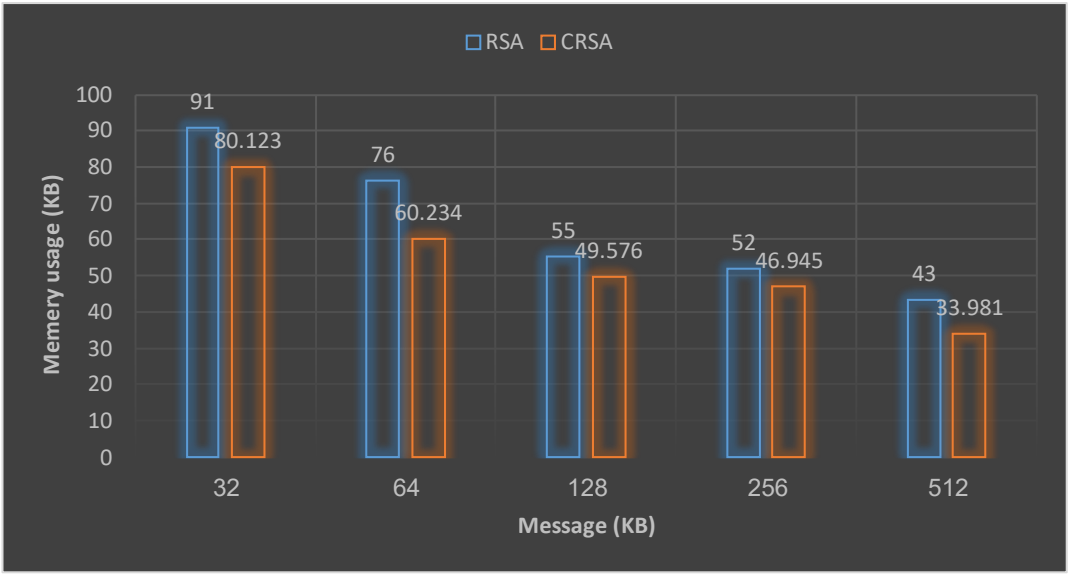

Figure 4.7 Memory usage (KB).

These The following graph represents the relation between the plain text (message) in kilobytes and the time needed for running the proposed CRSA against RSA.

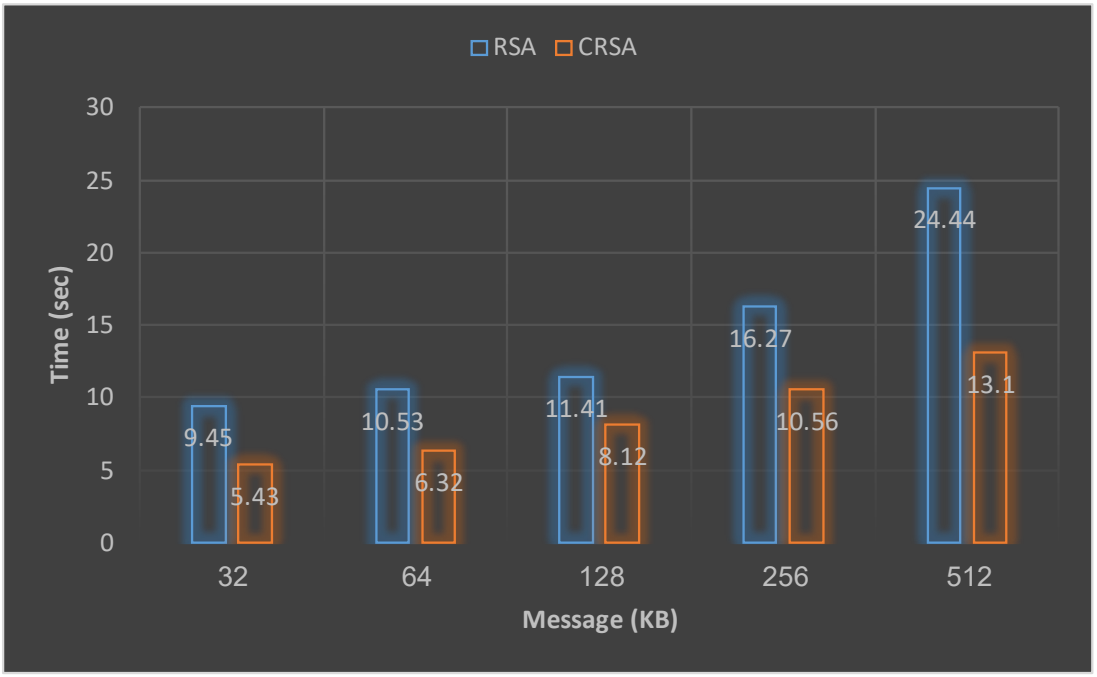

Figure 4.8 Time (Sec).

Chapter 5

Conclusions and Future Work

5.1 Conclusions

The goal of this thesis is to develop a method that uses the RSA algorithm to improve encryption and decryption operations. CRSA is included in the suggested algorithm.

The research accomplishments of this thesis can be described as follows:

1-Using the Catalan numbers equation to generate the public and private keys expands the encryption key space for the proposed approach.

2- Because each communication segment is encrypted with a distinct key created by the CRSA, any hacker or cryptanalyst will find it incredibly difficult to crack the security, because even if one segment is penetrated, each

of the other segments will require equal effort to be compromised.

3- The Java programming language was used in this thesis for coding and testing..

4- RSA is very strong algorithm but the keys its one for encryption and the other to decryption operation. It includes three steps; key generation step, encryption step, and decryption step. They may have a key size from 1024 to 4096 bits. The proposed algorithm provides a flexible and strong key for encryption, as it consists of generating the public and private keys

5.2 Future Work

The material offered in this thesis can be expanded upon to enhance or implement in various practical applications. Here are some ideas for future study and implementation projects.

1- To improve security, encrypted file segments can be sent in Catalan numbers according to a specific CRSA.

2- Combining the suggested technique with additional cryptographic algorithms.

Bibliography

[1] Al-Nbhany W, A N A and Zahary A T 2016 A Comparative Study among Cryptographic Algorithms: Blowfish, AES and RSA (Morocco: International Arab Conference on Information Technology (ACIT2016) Beni Mellal)

[2] Chalurkari S N, khochare N and Mashram B B 2011 Survey on Modular Attack on RSA Algorithm (International Journal of Computational Engineering & Management 14 106

[3] Hercigonja Z 2016 Comparative Analysis of Cryptographic Algorithms (Croatia: International Journal of Digital Technology & Economy 1 127-34

[4] Stallings W 2017 Cryptography and Network Security 7 th Ed, (Prentice Hall) 58-309

[5] Tripathi R and Agrawal S 2014 “Comparative Study of Symmetric and Asymmetric Cryptography Techniques (International Journal of Advance Foundation and Research in Computer (IJAFRC)) 1 68-76

[6] William S 2017 Cryptography and network security principles and practice 7 the Ed (PrenticeHall Inc) 23-50

[7] William J and Buchanan 2011 Rc2 encryption in .net http://buchananweb.co.uk/

[8] Hercigonja Z 2016 Comparative Analysis of Cryptographic Algorithms (Croatia: International Journal of Digital Technology & Economy 1 127-34

[9] Dr. Prerna Mahajan, Abhishek Sachdeva, 2013, A study of Encryption algorithms AES, DES and RSA for security, Global Journal of Computer Science and Technology, Volume 13 Issue 15 Version 1.0

[10] Chia Long Wu, Chen Hao Hu, "Computational Complexity Theoretical Analyses on Cryptographic Algorithms for Computer Security Application", Innovations in Bio-Inspired Computing and Applications(IBICA), 2012, pp. 307 – 311.

[11] Mandal, B.K. , Bhattacharyya, Bandyopadhyay S.K., "Designing and Performance Analysis of a Proposed Symmetric Cryptography Algorithm ", Communication Systems and Network Technologies (CSNT), 2013, pp. 453 – 461.

[12] Aboshosha, B. W., Dessouky, M. M., & Elsayed, A. (2019). Energy-efficient encryption algorithm for low resources devices. The Academic Research Community publication, 3(3), 26-37.

[13] Ahmad, A., & Ismail, S. (2018). User selective encryption method for securing MANETs. International Journal of Electrical and Computer Engineering (IJECE), 8(5), 3103-3111.

[14] Alarifi, A., Amoon, M., Aly, M. H., & El-Shafai, W. (2020). Optical PTFT asymmetric cryptosystem-based secure and efficient cancelable biometric recognition system. IEEE Access, 8, 221246-221268.

[15] Banani, S., Thiemjarus, S., Wongthavarawat, K., & Ounanong, N. (2022). A Dynamic Light-Weight Symmetric Encryption Algorithm for Secure Data Transmission via BLE Beacons. Journal of Sensor and Actuator Networks, 11(1), 2.

[16] Chen, X. D., Liu, Q., Wang, J., & Wang, Q. H. (2018). Asymmetric

encryption of multi-image based on compressed sensing and feature fusion with high-quality image reconstruction. Optics & Laser Technology, 107, 302-312.

[17] Cheremkhin, P. A., Evtikhiev, N. N., Krasnov, V. V., Rodin, V. G., Shifrina, A. V., & Starikov, R. S. (2020). Asymmetric image optical encryption under spatially incoherent illumination. Laser Physics Letters, 17(2), 025204.

[18] Mushtaq, M. F., Jamel, S., Disina, A. H., Pindar, Z. A., Shakir, N. S. A., & Deris, M. M. (2017). A survey on the cryptographic encryption algorithms. International Journal of Advanced Computer Science and Applications, 8(11), 333-344.

[19] Vengadapurvaja, A. M., Nisha, G., Aarthy, R., & Sasikaladevi, N. (2017). An efficient homomorphic medical image encryption algorithm for cloud storage security. Procedia computer science, 115, 643-650.

[20] Wu, C., Hu, K. Y., Wang, Y., Wang, J., & Wang, Q. H. (2019). Scalable asymmetric image encryption based on phase truncation in cylindrical diffraction domain. Optics Communications, 448, 26-32.

[21] Wu, C., Wang, Y., Chen, Y., Wang, J., & Wang, Q. H. (2019). Asymmetric encryption of multiple-image based on compressed sensing and phase-truncation in cylindrical diffraction domain. Optics Communications, 431, 203-209.

Appendix

```
public class RandomPrimeGenerator {

   public static void main(String[] args) {

      while (true) {
         int pRandom = (int) (Math.random() * (83 - 2) + 2);
         if(isPrime(pRandom)){
            System.out.println("Got Random Prime p :"+pRandom);
            break;
         }
      }
      while(true){
         int qRandom = (int) (Math.random() * (83 - 2) + 2);
         if(isPrime(qRandom)){
            System.out.println("Got Random Prime q:"+qRandom);
            break;
         }
      }

   }

   private static boolean isPrime(int n) {
     int i;
     for(i=2;i<=Math.sqrt(n);i++){
        if(n % i == 0){
           return false;
        }
     }
     return true;
   }

}
```

```
import java.io.*;
import java.util.*;
import java.math.*;

class CRSA
{
        public static BigInteger findCatalan(int n)
```

```
	{
		// using BigInteger to calculate large factorials
		BigInteger b = new BigInteger("1");

		// calculating n!
		for (int i = 1; i <= n; i++) {
			b = b.multiply(BigInteger.valueOf(i));
		}

		// calculating n! * n!
		b = b.multiply(b);

		BigInteger d = new BigInteger("1");

		// calculating (2n)!
		for (int i = 1; i <= 2 * n; i++) {
			d = d.multiply(BigInteger.valueOf(i));
		}

		// calculating (2n)! / (n! * n!)
		BigInteger ans = d.divide(b);

		// calculating (2n)! / ((n! * n!) * (n+1))
		ans = ans.divide(BigInteger.valueOf(n + 1));
		return ans;
	}

	// Driver Code
	public static void main(String[] args)
	{
		int n = 5;
		System.out.println(findCatalan(n));
	}
}
```

```
// e is for public key exponent
				if (gcd(e, z) == 1) {
					break;
				}
			}
			System.out.println("the value of e = " + e);
			for (i = 0; i <= 9; i++) {
				int x = 1 + (i * z);
```

```
                    // d is for private key exponent
                    if (x % e == 0) {
                        d = x / e;
                        break;
                    }
                }
                System.out.println("the value of d = " + d);
                c = (Math.pow(msg, e)) % n;
                System.out.println("Encrypted message is : " + c);

                // converting int value of n to BigInteger
                BigInteger N = BigInteger.valueOf(n);

                // converting float value of c to BigInteger
                BigInteger C = BigDecimal.valueOf(c).toBigInteger();
                msgback = (C.pow(d)).mod(N);
                System.out.println("Decrypted message is : "
                                                + msgback);
        }

        static int gcd(int e, int z)
        {
                if (e == 0)
                        return z;
                else
                        return gcd(z % e, e);
        }
}
```
