# A SURVEY ON ENHANCED RSA ALGORITHMS

Shaheen Saad Al-Kaabi and Samir Brahim Belhaouari

College of Science and Engineering, Hamad Bin Khalifa University (HBKU), Doha, Qatar

**ABSTRACT**

*Cryptography is a field in computer science and mathematics that entails techniques of securing communication between two parties in the presence of a third party. This is achieved through the use of several methods, such as encryption, decryption, signing, generating of pseudo-random numbers, among many others. Cryptography uses a key, or some sort of a password to either encrypt or decrypt a message that needs to be kept secret. This is made possible using two classes of key-based encryption and decryption algorithms, namely symmetric and asymmetric algorithms. The best known and the most widely used public key system is RSA. This algorithm comprises of three phases, which are the key generation phase, encryption phase, and the decryption phase. Owing to the advancement in computing technology, RSA is prone to some security risks, which makes it less secure. The following paper preview different proposals on different methods used to enhance the RSA algorithm and increase its security. Some of these enhancements include combining the RSA algorithm with Diffie-Hellman or ElGamal algorithm, modification of RSA to include three or four prime numbers, offline storage of generated keys, a secured algorithm for RSA where the message can be encrypted using dual encryption keys, etc.*

**KEYWORDS**

*Cryptography, RSA Algorithm, Encryption, Decryption, Cryptosystem, Security, Public Key, Private Key.*

## 1. INTRODUCTION

The use of cryptography to conceal information dates back thousands of years ago. The oldest recorded application of cryptography is the protection of military secrets. A good example here is the Caesar Cipher used by Julius Caesar to send confidential information to his commanders and soldiers on the battlefield. Since then the encoded messages have been used by government, private entities, and militaries around the world to protect sensitive information. Barakat et al. In defined cryptography as a field in computer science and mathematics that entails techniques of securing communication between two parties in the presence of a third party [10]. This is achieved through the use of several methods, such as encryption, decryption, signing, generating of pseudo-random numbers, among many others. Cryptography is anchored in four major principles whose main objectives include ensuring confidentiality, data integrity, authenticity, and non-repudiation. Cryptography ensures confidentiality by defining a set of rules that limit access to certain information. On the other hand, data integrity is upheld by ensuring consistency and accuracy of data during its entire life-cycle. Authentication, on the other hand, helps in confirming the truth of an attribute of a datum that claims to be true by some entity. Under nonrepudiation, cryptography ensures that an author of a given statement or piece of data cannot deny it. Through cryptographic technology, communication or transfer of data electronically can be done without any worry of deceit or deception (confidentiality), while at the same time maintaining the integrity of the message and the authenticity of the sender. This is achieved through the conversion of data unto different forms that are incomprehensible (ciphertext or code). The process through which data is converted into ciphertext is called encryption, while the

Dhinaharan Nagamalai et al. (Eds) : WIMO, ICAIT, ICDIPV, NC, CRYPIS - 2019
pp. 123-142, 2019. © CS & IT-CSCP 2019 DOI: 10.5121/csit.2019.90411

process through which the ciphertext converted to comprehensible information (plaintext) is called decryption. Both encryption and decryption processes are done using secret information such as passwords or keys.

## 1.1. Symmetric and Asymmetric Cryptography

As indicated in the discussion above, cryptography uses a key, or some sort of a password to either encrypt or decrypt a message that needs to be kept secret. This is made possible using two classes of key-based encryption and decryption algorithms, namely symmetric, also known as secret-key, and asymmetric, which is also known as public –key.

### 1.1.1. Symmetric Key Cryptography

Before the 1970s, the symmetric key encryption, also known as the secret-key encryption, or conventional system was the only encryption technology in use, and still, remain by far the most widely used method of encryption. Before the advent of computers, the ciphertext used in symmetric key encryption was called the classical encryption algorithms. Currently, and with the advent of computing technology, symmetric encryption uses bits and bytes as the encryption keys, together with various encryption algorithms to transform plaintext into ciphertext. The recovery process of plaintext from the ciphertext in symmetric key cryptography is done using the same key used in encryption, and a different decryption algorithm. This is made possible by sharing the secret key between the sender and the receiver. Given the fact that anyone who gains access to the key can decrypt the information, the key is usually highly secured and is kept as a secret between the two parties sharing information. The figure below illustrates the encryption and decryption process in symmetric key cryptography:

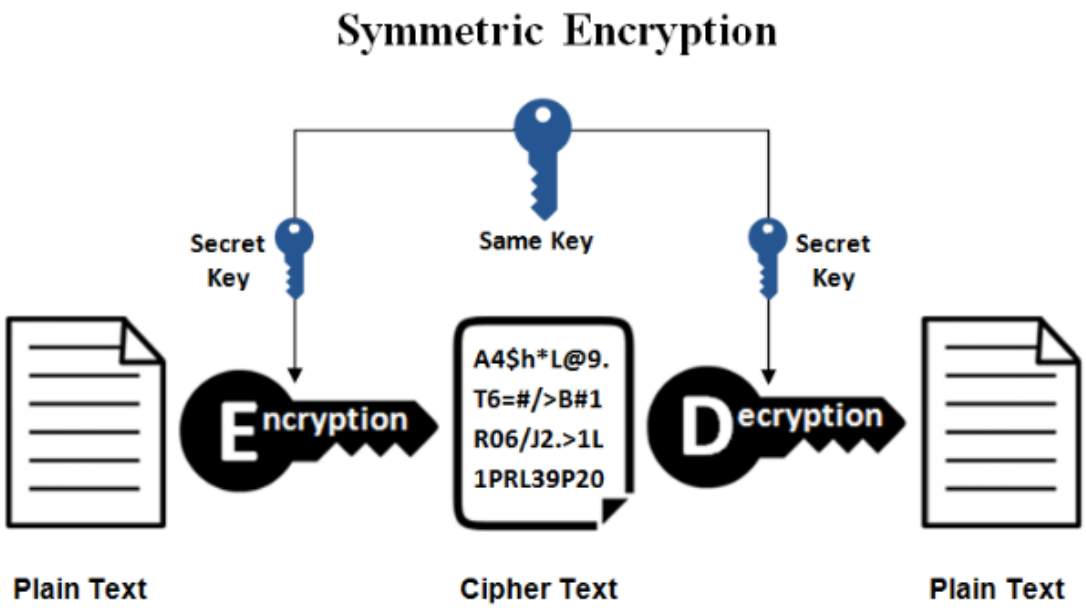


Figure 1. Symmetric key Cryptography

Some of the most common symmetric key algorithms include Data Encryption Standard (DES), Triple Data Encryption Standard (TRIPLEDES), Advanced Encryption Standard (AES), RC2, RC4, RC5, RC6, Blowfish, and Twofish. Triple DES algorithm was developed in response to the increase in the computational power that made brute-force attacks feasible. As such, this algorithm was designed to maximize the security of DES against such attacks. AES, on the other hand, allows the use of three different key lengths, including 128, 192, or 256 bits. The encryption process for AES comprises of 10 rounds of processing for the 128-bit keys, 12 rounds of processing for a 192-bit key, and 14 rounds of processing for the 256-bit keys.

### 1.1.1. Symmetric key cryptography

Unlike symmetric key cryptography, asymmetric key cryptography uses two separate keys, where one is a private key and the other one is a public key. The public key is used in encrypting of the data, while the corresponding private key is used to decrypt the data. This implies that the public key is published for anyone to see, but the private key is kept a secret. As such, anyone with a copy of the public key can encrypt a message, but only the person holding the private key can decrypt the message. The figure below is an illustration of how asymmetrical cryptography works:

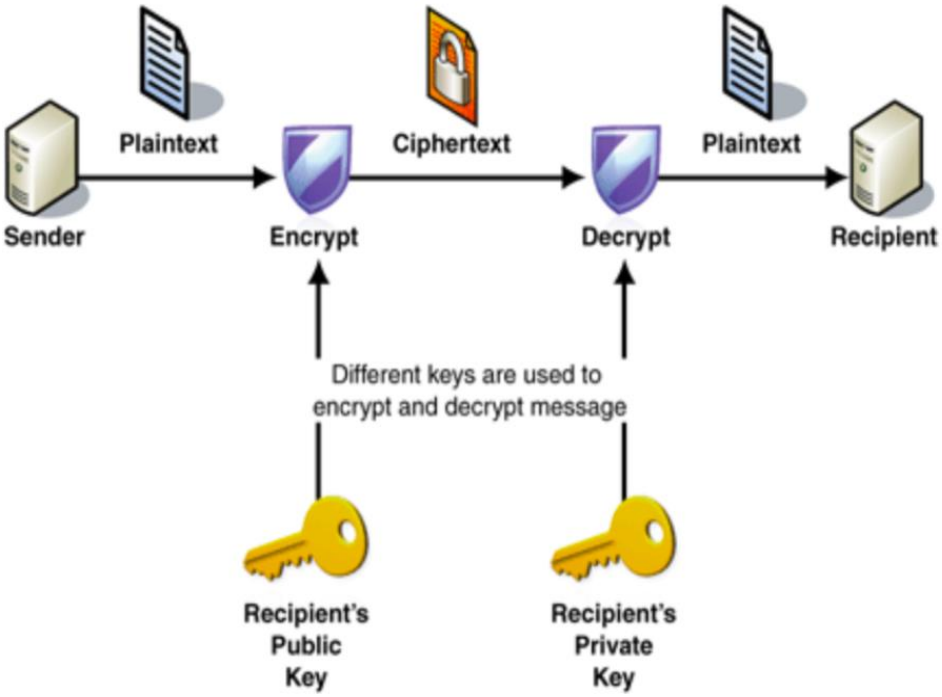


Figure 2. Asymmetric key Cryptography

One of the key advantage of asymmetric cryptography as compared to symmetric cryptography is the fact that asymmetric eliminates the need for the sender and the receiver to share the private key. All communication involves only public keys and no private key is transmitted or shared. Some examples of areas in which asymmetric is applied include Elgamal, RSA, Diffie-Hellman, and DSA.

# 2. RSA ALGORITHM

At the moment, the best known and the most widely used public key system is RSA. It was developed in 1978 by Ronald Rivest, Adi Shamir, and Leonard Adleman. RSA is based on number theory, was the first algorithm known to be suitable for signing as well as encryption, and one of the first great advances in public key encryption. RSA is a public key cryptographic system that uses the concept of number theory. Its security, therefore, depends on the complexity of prime factorization of large numbers [16], which is a well-known mathematical problem with no known effective solution. This makes RSA one of the most widely used technique for asymmetric key cryptography in encryption and digital signature standards. Generally, RSA algorithm comprises of three phases, which are the key generation phase, encryption phase, and the decryption phase

## 2.1. Key Generation

The key generation phase comprises of a process through which cryptographic keys are generated. The generated keys are then used in encrypting and decrypting of the data that need to be transmitted through public and untrusted network. Both private and public key are generated and used in cryptography in RSA algorithm which is available to everyone by means of a digital certificate. Any data to be sent using RSA algorithm is encrypted using the public key. However, it is only a person with a corresponding public key can be able to decrypt the data using the private key. The steps involved in in key generation are as follows:

**a.** Choose two different prime numbers **p** and **q** . (**p** ≠ **q**).
**b.** Calculate the common module **n** such that **n = p . q**.
**c.** Calculate the Euler's **phi : ∅(n) = ( p – 1 ).( q – 1 )**
**d.** Select integer **e** which is the encryption (public) key such that :
**GCD ( ∅(n) , e ) = 1 ; 1 < e < ∅(n)**
**e.** Calculate **d** is the decryption (private) key such that $\mathbf{d = e^{-1} \pmod{\emptyset(n)}}$.
**f.** So, the public key is **PU = [ e , n ]** , and the private key is **PR = [ d , n ]**.

It is necessary to generate large random primes in setting up the RSA algorithm. Practically, testing the primality of large random numbers can be done using randomized polynomial time Monte Carlo algorithm such as SOLOVAY-STRASSEN algorithm or MILLER-RABIN algorithm. These algorithms are known to be fast as primality testing algorithm since an integer $n$ can be tested in time that is polynomial in $log2\ n$ [24].To know how many random integers that need to be tested till prime is found, Prime number theorem can be used which states that $\pi(N) \approx N/\ln N$, where $\pi(N)$ is the number of primes $\leq N$. If we consider $p$ is a randomly selected integer between 1 and $N$, then the probability that $p$ is prime is $1/\ln N$. So, we can adequately generate a large random "probable Prime" [24].

## 2.2. Encryption

In the RSA algorithm, encryption is used to define the process through which a message is encoded in such a way that only the authorized parties can read it. In an encryption scheme, the message or the information that need to be sent is presented in plaintext format. The plaintext message is then encrypted using an encryption algorithm to generate ciphertext that can only be comprehensible if decrypted. The plain text is encrypted in blocks, each block having a value less than common module n. The encryption algorithm is given by:

$$C = M^e \bmod n$$

Where: **M** : Block of plain text
**C** : Block of cipher text
**e** : Public key

## 2.3. Decryption

The process through which ciphertext is decoded to get the plaintext message in a format that can be comprehended is called decryption. In RSA algorithm, decryption can only be done by an authorized person using a private key. As such, anybody else without the private key can intercept the message but cannot comprehend the text without decrypting it using the private key. The decryption algorithm is given by:

$$M = C^d \bmod n$$

Where: **M** : Block of plain text
**C** : Block of cipher text
**d** : Private key

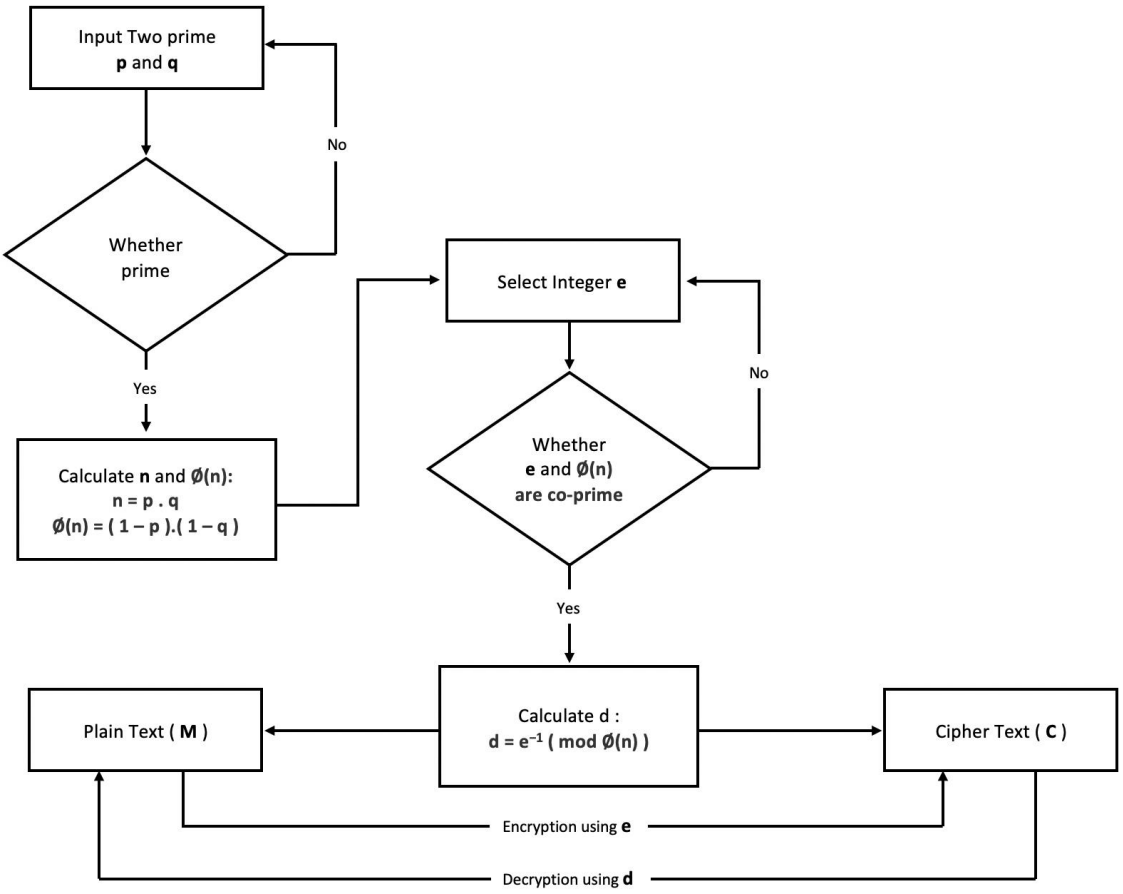


Figure 3. Flow Design of RSA Algorithm

## 3. RSA ALGORITHM LIMITATIONS:

According to Gupta and Sharma, some of the limitations associated with RSA include the fact that if any of p, q, e, and d is known, then the other values can be calculated, and therefore, eliminating the secrecy [1]. Also, RSA requires that the length of the message be less than bit length, otherwise the algorithm is likely to fail. Another limitation of RSA is based on the fact that since it uses the public key, it is much slower as compared to all other symmetric cryptographic systems. In addition, the length of plain text that can be encrypted is limited to the size of N = P x Q. Gupta and Sharma pointed out that each initialization of RSA process requires random selection of two very large prime numbers (P and Q) [1]. Another 2013 study by Patidar and Bhartiya identified several limitations associated with RSA [3]. These included the issue of speed as described by Gupta and Sharma and computational cost due to the need for two different keys. There is also the issue of loss of private key, which may break the security. Here, RSA is criticized based on its application of private key [3]. Patidar and Bhartiya point out that during the decryption process, the private key is used, as such, if any unauthorized person knows the value of the private key, then the entire security of RSA algorithm is compromised [3]. Other researchers such as Minni et al argues that RSA, as it is, is prone to mathematical factorization attacks and therefore needs further enhancements [4]. This was the same argument by Jaju and Chowhan who pointed out that based on the overall execution time, RSA algorithm takes more time as compared to other algorithms [2]. Panda and Chattopadhyay found out that RSA algorithm was easy to factorize since modulo n used is the product of two prime numbers only and therefore making it easy to obtain the decrypted original message [5]. Chaudhury et al. Also identified several limitations to the RSA algorithm [6]. These included the fact that it is prone to various types of attack such as forward search attack common modulus attack, among others. It was also pointed out that RSA is disadvantaged owing to its factorization problem and computational speed. Other researchers who pinpointed several limitations in regard to RSA algorithm included Hussein [8] who delved in to improving the model by enhancing security and elimination of redundant messages using K-nearest neighbor algorithm, this was the same case with others such as Mathur et al. and Iswari whose work led to development of more enhanced RSA algorithm [7][9].

## 4. ATTACKS ON RSA CRYPTOSYSTEM

### 4.1. Factoring RSA Modulus

Factoring the public modulus is the most evident way to attack RSA cryptosystem. The Quadratic Sieve, the Number Field Sieve, and the Elliptic Curve Factoring Algorithm considered being the best and most effective factoring algorithms on very large numbers [24]. Till the mid-1990s, the Quadratic Sieve was the most used algorithm. However, the Number Field Sieve, which is the recently developed algorithm of the three, was proven to be the fastest in term of its asymptotic running time [24].

In the early 1990s, RSA publishes a series of challenges and set valuable prices in a range of 10k$ to 200k $ for factoring algorithms on the Internet. In this regard, Kleinjung et al. provide an indepth report in 2009 on the factorization of the 768-bit number RSA-768 by the number field sieve (NSF) factoring method and provide some implication for the RSA [23]. The first step in this project was the selection of polynomial. This took half a year and 80 processors. The next step was sieving using hundreds of machines, followed by preparation of the sieving data for the matrix step, which took a couple of weeks on a few processors before the final step of debugging. For $2^{1039}$ -1 the matrix steps were performed on different clusters. The main part of the calculation was run in parallel at two locations on four clusters. The use of block Wiedemann

algorithm for the matrix time step made this possible, where the main calculation comprised of two consecutive sequences of matrix times vector multiplication. A greater degree of independent parallelization resulted in several challenges that required solving before handling larger problems. The first step toward a scalable solution was achieved by showing how greater flexibility can be achieved in the number and contribution of different independent clusters. This helped in solving a matrix step that would have otherwise been nine times harder. During one of the sub-steps, about a terabyte of memory was needed. This implied that much larger matrices were within reach in the near future for the matrix required for a 1024-bit NFS factorization.

Factorization of RSA-768 was done using the Morrison-Brillhart approach. The paper also describes the various steps required to solve NFS. Here, a composite integer n is first factored by finding integer solution x; y of the congruence of squares $x^2 \equiv y^2 \bmod n$ and by hoping that n is factored by writing it as a product gcd (x – y, n). gcd (x + y, n). The probability that a non-trivial factor of n is found in this way for a random such pair is at least 0.5 [23]. The Morrison-Brillhart approach solves the equation $x^2 \equiv y^2 \bmod n$ by combining the congruencies of smooth squares. The paper also comprises of discussion on the implication for moduli larger than RSA-768. It is also crucial to note that although NFS can be run as a BOINC project, the author chose not to, and instead gathered a group of knowledgeable contributors who committed themselves to finish the project, while at the same time dedicating a known minimum of computational resources for a substantial period of time. This in return allowed the authors to target a reasonable completion date that is easily manageable despite the intensive oversieving. The authors allude to the fact that although different sieving clients don't communicate, there is a need for each client to communicate a fair amount of data to the central storage location. Achieving this requires a bit more organizational effort that expected occasional recovery from mishaps such as unplugged network cables, switched off servers, or faulty raids and constantly growing farm backup drives [23].

For that, it is assumed nowadays that 1024 bit number will be factored by 2020 and will be not secured enough to stand against the factorization attacks. As a result, it is believed that using 2048 bit key length in RSA should be secured for a longer time [10].

## 4.2. Timing Attacks

Kocher described a timing attack on implementations of Diffie-Hellman, RSA, DSS, and other systems [18]. Diffie-Hellman and RSA private key operations comprise of computing $R = y^x \bmod n$, where n is the public. An attacker can eavesdrop and determine the value of y. The main objective of such an attacker is to find the value of x, which is the secret key. For the attack to be successful, the victim must first compute $y^x \bmod n$ for different value of y. The attacker must also have the knowledge of the value of y, n, and the time taken to compute for the value of y. The attacker gains this knowledge through passively eavesdropping on an interactive protocol. Once the attacker manages to eavesdrop, the message is recorded and it is from such recording that the amount of time taken to respond to each value of y is measured. In addition, the attacker must know the design of the system used by the target for the attack to be successful. The attack can be modified to work with almost any implementation that is not running on fixed time, as long as it is outlined using a simple modular exponentiation algorithm. This form of attack can be thwarted by the use of RSA blinding signature, which makes it impossible for the attacker to gain knowledge of the input of the modular exponentiation function [18].

Another form of timing attack that occurs as a result of SSL handshake failure was described by Brumley and Boneh in 2005 [19]. In this study, the properly formatted CLIENT- KEY-EXCHANGE message between the server and the client was replaced with guess g, The server

decrypts g as a normal CLIENT-KEY-EXCHANGE message, and then checks the resulting plaintext for proper PKCS 1 padding. Given the fact that the decryption of g will not be

formatted properly, the server and the client will not compute the same master secret, and the client will ultimately receive an ALERT message from the server. The attacking client computes the time difference from sending g as the CLIENT-KEY-EXCHANGE message to receiving the response message from the server as the time to decrypt g. This form of attack can be prevented by performing RSA blinding and making all RSA decryption not dependent upon the input ciphertext [19].

## 4.3. Adaptive Chosen Ciphertext Attacks

For the sake of improving the speed of encryption as well as its efficiency, a small encryption exponent is usually selected such as e = 3 . However, sending same message m to three different entities using the same small exponent e whose public moduli are $n_1$,$n_2$, and $n_3$, then the sender entity would send $c_i = m^3 \bmod n_i$ for i = 1,2,3 . An eavesdropper observing $c_1$, $c_2$, $c_3$ can use the Gauss's Algorithm to find a solution of m since these moduli are most likely relatively prime. Therefore, it is not recommended to use a small encryption exponent such as e = 3 if the same message even with known variations is sent to many entities. Alternatively, appending a pseudorandomly generated bit string independently to each message before the encryption process should be done to prevent against such an attack [25].

In a 1998 study, Bleichenbacher describes new adaptive chosen ciphertext attacks against protocols based on the RSA encryption standard (PKCS) [20]. This form of attack can be launched in four main steps. The first step involves blinding, where given an integer c, the attacker choose different random integer $S_0$ and check whether c $(S_0)^e$ mod n is PKCS conforming by accessing the Oracle. In the second step, the attacker search for PKCS conforming messages. "This is done by assuming that if i = 1, then search for the smallest positive integer $S_1 \leq$ n/ (3B), such that the ciphertext $C_0$ $(S_1)^e$ mod n is PKCS conforming" [20]. The attacker can also search the message by different intervals. Bleichenbacher points out that " if i >1, and the number of intervals vin M i-1 is at least 2, then the attack search for the smallest integer $S_{i-1}$, such that ciphertext $C_0$ $(S_i)^e$ mod n is PKCS conforming" [20]. The next step is narrowing the set of solution. After $S_i$ has been found, the set $M_i$ is computed as shown in the equation below:

$$M_i \leftarrow \bigcup_{(a,b,r)} \left\{ \left[ \max\left(a, \left\lceil \frac{2B + rn}{s_i} \right\rceil \right), \min\left(b, \left\lfloor \frac{3B - 1 + rn}{s_i} \right\rfloor \right) \right] \right\}$$

$$\text{for all } [a,b] \in M_{i-1} \text{ and } \frac{as_i - 3B + 1}{n} \leq r \leq \frac{bs_i - 2B}{n}.$$

The fourth step is the computation of the solution. "If Mi contains only one interval of length 1 (i.e. Mi ={ [a, a]}, then the set m ß a $(S_0)^{-1}$ mod n, and return m as a solution of $m \equiv C^d \pmod n$. Otherwise, set i ß i +1 and go to second step" [20]. To prevent such kind of an attack, Beechen bacher points out the need to have a receiver check the integrity of a message before or immediately after decrypting it [20].

### 4.4. Side-Channel Analysis Attacks

Aciicmez et al. described a new form of side channel analysis attack using a spy-process running simultaneously with an RSA-process [21]. The spy process is based on analysis of the CPU's Branch predictor states and is able to collect almost all the secret on a single quasi-parallel computation process. This discovery proves that blinding technique is totally useless in the case of this form of attack. This form of attack uses much execution-time measurement under the same key order to statistically amplify some small but key dependent timing differences. Aciicmez et al calls this form of attack as Simple Branch Prediction Analysis (SBPA) attack. Another form of side-channel analysis attack was described by Pellegrini et al under fault-based attack of RSA authentication [22]. This form of attack exploits the flaw on the implementation of RSA signature algorithm on OpenSSL used for SSL encryption and authentication. The attack is carried out through extraction of a server's private key by injecting faults in the server's hardware, leading to the production of intermittent computational errors during the authentication of a message. The attacker can then use an extraction algorithm to compute the private key d from several unique message m and their corresponding erroneous signature [22].

## 5. ENHANCEMENT PROPOSALS FOR RSA ALGORITHM

### 5.1. A Hybrid Encryption Algorithm Based on RSA and Diffie-Hellman.

In their work, Gupta and Sharma suggested a new hybrid encryption algorithm based on RSA and Diffie-Hellman algorithm to address some of the major security issues identified in the RSA algorithm [1]. The Diffie-Helaman algorithm (DH) was founded by Whitfield Diffie and Martin Hellman in 1976. It is astonishing and widespread algorithm used on the internet in numerous secured connectivity protocols such as Secure Shell (SSH), Internet Protocol Security (IPSec), and Secure Sockets Layer (SSL) [1,11,13]. The method of DH lies on securely interchanging a shared secret between two parties on a public network and each party has public and private key in order to correspond on a shared secret value [12]. The main aim of this proposal in combining these two algorithms is to achieve better and more secure cryptosystem, taking advantage of the security of public key system and the speed of the secret key system. The steps involved in the algorithm included:

1. Choose two large prime numbers P and Q.
   a. Calculate N = P x Q.
   b. Select public key ( Encryption key ) E such that it is not a factor of ( P - 1 ) and ( Q -1 ).
   c. Select the private key ( Decryption key ) D such that the following equation is true ( D x E ) mod ( P – 1 ) x ( Q – 1 ) = 1.
   d. Suppose R, S and G is automatic generated prime constants.
   e. And put the value of E and D from above as secret number such that A=E and B=D.
2. Now calculate following as public number
   X = GA mod R
   Y = GB mod R
3. Calculate session key with formula
   KA = YA mod R
   KA = (GB mod R) A mod R
   KA = (GB)A mod R
   **KA = GBA mod R.**

KB = XB mod R
KB = (GA mod R)B mod R
KB = (GA)B mod R
**KB = GAB mod R**

Such that KA = KB = K.

**K** will be used by the sender to encrypt the plain text PT and will be sent as cipher text CT to the receiver, then also **K** will be used by the receiver to decrypt CT and recover the plain text PT.

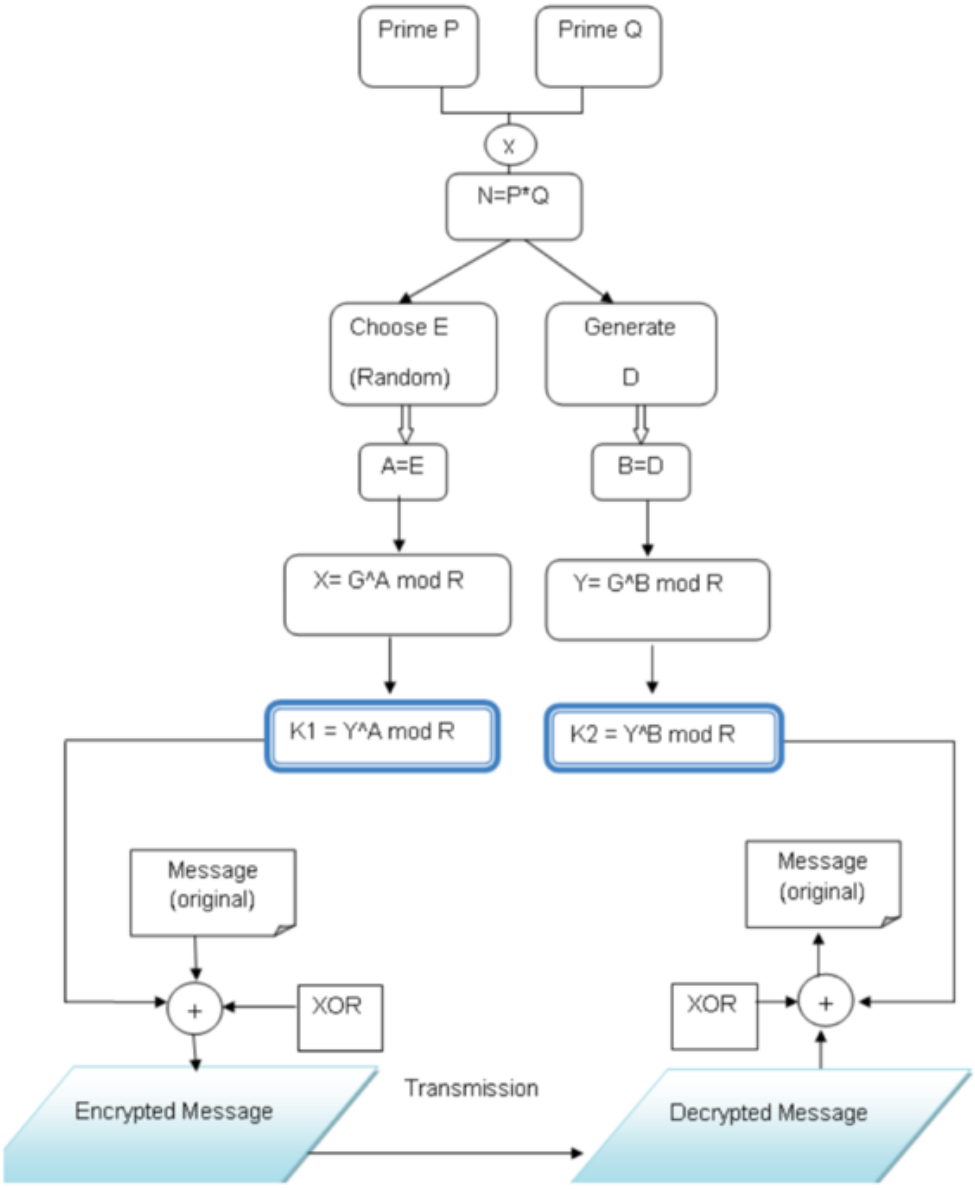


Figure 4. Flow design of a hybrid RSA & Diffie-Hellman algorithm

This hybrid algorithm tends to be easy for users to communicate securely over the public network, especially for messages or files that need to be exchanged confidentially. Authors mentioned that the usability of their proposed algorithm is applied with few concepts and ideas that can be extended in the future. The efficiency can be revised in term of time complexity for better functionality of the algorithm, also the key size used for encryption and decryption can be further reduced for better performance [1].

## 5.2. Combination of RSA and ElGamal Algorithm.

**I**swari also proposed another enhancement to the RSA algorithm by combining it with the ElGamal algorithm [9]. ElGamal is a well-known public key cryptosystem algorithm. Initially, it was used only for digital signature. Later on, it has been developed and modified in order to be used for encryption and decryption [9]. The security strength of the ElGamal algorithm lies on the difficulty of computing the discrete logarithm [15]. Its pair of key generation can be summarized as follows:

1. Choose a random prime number **p**
2. Choose two random number, **g** and **x** , where ( g < p ) and ( x < p ) .
3. Calculate $\mathbf{y = g^x \bmod p}$.
4. **y** is the public key and **x** is the private key

In the Author proposal, 256-bit prime numbers were used for the sake of decreasing the computational time required for the key generation instead of 1024-bit prime numbers that are used in the original RSA algorithm. By combining RSA and ElGamal, the security factors and complexity is maintained even if small bit prime numbers are used due to both factorization and discrete logarithm calculation difficulties. The following is the summary of the algorithm:

**1.** Select two prime numbers p and q
**2.** r = p . q
**3.** $\emptyset$ (r) = (p -1).(q-1)
**4.** Generate Random number PK ( encryption key ), where GCD (PK, $\emptyset$ (r)) = 1.
**5.** Compute decryption key $SK = PK^{-1} \bmod \emptyset (r)$.

Now, ElGamal algorithm will be used in the process of the key generation :

**6.** PK = g , and SK = x
**7.** Generate a random number *pEl* , where ( PK < *pEl* ) and ( SK < *pEl* ).
**8.** Calculate public key for ElGamal algorithm ( $y = PK^{SK} \bmod pEl$ ) . where GCD (y, $\emptyset$ (r)) = 1.
**9.** Recalculate private key again for RSA algorithm ( $SK = y^{-1} \bmod \emptyset (r)$ ).

## 5.3. Asymmetric Key Based Cryptographic Algorithm Using Four Prime Numbers.

**C**haudhury et al. uses a modified RSA cryptosystem algorithm to handle four smallest prime numbers and enhance the security of the RSA algorithm [6]. The security of the modified RSA depends on the complexity of disintegrating the four prime numbers used. Compared to the original RSA algorithm, it requires less memory and power while maintaining the same complexity of RSA [6].

**1.** Select four prime numbers w, x, y and z
**2.** n = w.x.y.z.
**3.** $f(n) = (w - 1).(x - 1).(y - 1).(z - 1)$;
**4.** Select an integer p, where 1 < p < f (n), such that GCD ( p, (f(n)) =1 and p and f (n) are co –prime

5. Compute the secret exponent d, where $1 < d < f(n)$ such that $(p \times d) \bmod f(n) = 1$
6. d should be kept private
7. (p, n) is the public key, and (d, n) is the private key
8. Encryption: $CT = (m^{p}) \bmod n$.
   Decryption: $m = (CT^{d}) \bmod n$.

## 5.4. Modified RSA Cryptosystem Based on Offline Storage and Prime Number.

**P**atidar and Bhartiya also proposed new algorithm concept to presents the modified form of RSA algorithm to speed up the implementation of RSA algorithm during data exchange across the network [3]. The proposed modification comprised of the architectural design and an enhanced form of RSA algorithm through the use of a third prime number to make a modulus n which is not easily decomposed by intruders [3]. The following is the summary of the proposed algorithm:

1. Select the random values p, q and r
2. Calculate ( $n = p \cdot q \cdot r$ ).
3. Calculate $\emptyset(n) = (p-1).(q-1).(r-1)$.
4. Calculate e such that $GCD(e, \emptyset(n)) = 1$ and $1 < e < \emptyset(n)$.
5. Encrypt the message M where $M < n$ and encrypt with public key e such that $C = M^{e} \bmod n$.
6. Calculate private key $d = e^{-1} \pmod{\emptyset(n)}$.
7. Decrypt the message M such that $M = C^{d} \bmod n$.

In the proposed method, keys are stored offline before the process is started. As such, the speed of the process of encryption and decryption is increased as compared to the original RSA algorithm [3]. Two tables were created in a database engine for that reason to save the keys. The first table contains the value of p, q, n1, and $\emptyset(n)$, while the second table contains the value of e, d, r, e1, and d1. The following figure shows the mechanism of fetching the values from/to the data base.

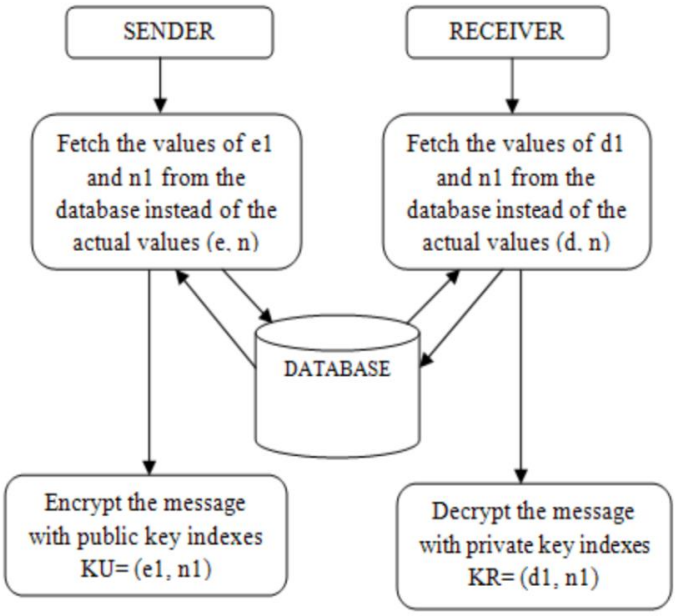


Figure 5. the mechanism of fetching the values from/to the data base.

Although this proposal increases the speed of encryption and decryption of the message, the concept of using a database for storing keys is still critical, since keys can be retrieved easily if the system is hacked [7].

### 5.5. Enhance the Security of RSA by Eliminating the Distribution of Modulus N.

**A**dditionally, Minni et al. proposed another secure algorithm by eliminating the distribution of **n** which is the large number whose factor if found compromises the RSA algorithm [4]. The following are the major steps in the implementation of the proposed algorithm:

1. Under key generation, two different prime numbers are selected randomly A and B
2. Calculate N= A . B.
3. Calculate $\emptyset$ (N) = (A -1).(B – 1).
4. Calculate k1 based on the following conditions
   - $\sqrt{N} < k1 < \emptyset$ ( n)
   - GCD (k1, $\emptyset$ (N)) = 1 that is k1 and $\emptyset$(N) are co-prime
   - k1 is short bit length and small hamming weight
5. Compute X to replace N.
   If A > B then consider X such that
   - N – A < X < N
   - GCD(X, N) = 1.

   If A < B then, consider X such that
   - N – B < X < N
   - GCD (X, N) = 1
6. Find k2 such that k1 x k2 Mod (X) = 1

The public key therefore consists of (k1, X) while the private key is (k2, X). To encipher the plain text PT, the sender uses public key (K1, X) by $CT = PT^{k1}$ Mod(X) where CT is the cipher text that is generated after encryption. The receiver decrypts the cipher text CT using the private key (K2, X) by $PT = \sqrt{}$ ($CT^{k2}$ Mod(X)) [4]. One disadvantage of the modified algorithm is that it takes more time in term of key generation process as compared to the RSA method.

### 5.6. A Modified RSA Algorithm to Enhance Security for Digital Signature.

**I**n the bid to increase speed and enhance the security of the RSA algorithm, Jaju and Chowhan modified the RSA algorithm to include three prime numbers instead of two prime random numbers for calculating **n** and passing value of **X** instead of **n** in public key and private key [2]. The following is a summary of the modified RSA algorithm:

1. Selecting any three prime numbers: p, q and r, such that $p \neq q \neq r$ .
2. Calculating the product of the three prime number (n = p . q . r). The length of **n** is the key length expressed in bits
3. Calculate $\emptyset$ (n) = ( p – 1 ).( q – 1 ).( r – 1 ).
4. Calculating integer e which is based on
   - $\sqrt{}$ n < e < $\emptyset$ (n)
   - GCD ($\emptyset$ (n), e) = 1 that is e and $\emptyset$(n) are co-prime
   - e is short bit length and small hamming weight
5. Compute X to replace n
   - If p > q, then consider X such that n-p <X<n and GCD (X, n) = 1
   - If p<q, then consider X such that n-q <X< n and GCD (X, n) = 1
6. Calculate d such that $d = e^{-1}$ (mod $\emptyset$ (n))
7. Now the public key PU = [e, X]

**8.** Now the private key PR = [d, X]
**9.** Consider plain text M, M < n
**10.** Find cipher of plain text by $C = M^e \bmod X$
**11.** Transmit the coded message to receiver by sender
**12.** Find plain text from cipher by receiver using $M = C^d \bmod X$

From security level respective, the proposed algorithm considered to be more secure compared to original RSA due to the following reasons : (1) The common modulus n can only be known by factoring three prime numbers p, q, and r which is a time-consuming and more difficult challenge for intruders to apply. (2) Since the value of **X** is transmitted instead of **n** in the public key, it will be difficult to know the hidden value of n when factorization attack is attempted. Although the modified algorithm addresses the issue of security and key generation speed, it takes a longer time to encrypt and decrypt text as compared to the RSA algorithm [2].

## 5.7. An Enhanced and Secured RSA Key Generation Scheme (ESRKGS).

**T**hangavel et al. also proposed a modified and enhanced scheme based on RSA public-key cryptosystem using four prime numbers [14]. The value of N, which is a product of the four prime numbers determine the value of E, D. Additionally, the computation of E is not direct, where in order to determine the value of E1, the values of e1 and e2 must be obtained. This helps in increasing the time taken to attack the system. "Only the value of n is kept as public and private component, this means that an attacker with the knowledge of n cannot determine all the primes which are the basis for finding the value of N, and subsequently D" [14]. The parameter E1 also helps in increasing the complexity of the system [14]. Below is a summary of the algorithm

***Key Generation***

1- Select four large prime numbers p, q, r, and s
2- Compute n = p . q and m = r . s
3- Compute N = m . n
4- Compute the Euler phi value of n and m
   $\Phi(n) = ( p - 1 ) . ( q - 1 )$
   $\Phi(m) = ( r - 1 ) . ( s - 1 )$
5- Compute $\Phi(N) = \Phi(n) . \Phi(m)$
6- Find a random number e1 that satisfy
   $1 < e1 < \Phi(n)$ and $\gcd(e1, \Phi(n)) = 1$
7- Find a random number e2 that satisfy
   $1 < e2 < \Phi(m)$ and $\gcd(e2, \Phi(m)) = 1$
8- Compute $E1 = e1^{e2} \bmod N$.
9- Find a random number E that satisfy
   $1 < E < (\Phi(N) . E1)$ and $\gcd(E, (\Phi(N) . E1)) = 1$
10- Compute a random number D, such that
   $D = E^{-1} \bmod (\Phi(N) . E1)$

***Encryption***

Plain text message M (<n) and Public Key Components {E, n}
Encryption : $C = M^E \bmod n$

***Decryption***

Cipher text message, C and Private Key component {D, n}
Decryption: $P = C^D \bmod n$

The enhanced RSA was implemented using the Java BigInteger Library function. Using this algorithm, the user is able to specify the prime numbers to use or determine the length of the bit to used using random functions. "The operations for modular arithmetic, GCD calculation, primary testing, prime generation, bit manipulation, and a few other miscellaneous operations are provided by the BigInteger Library" [14].

In comparison to other RSA-based algorithm reviewed in this study, the time taken in the generation of the key for the enhanced scheme is slightly greater [14]. This can be attributed to the fact that the time required to break the system is high owing to the complexity introduced. This is the same case for the encryption and decryption process due to the use of four prime numbers. Increase in the time taken increases the level of security in the enhanced system. Security analysis of the system indicated that the time taken for a brute-force attack on the enhanced system is far higher than other RSA scheme. This is owing to the fact that only 'n' is known to the attack but finding E and D the value of 'N' is needed, therefore making the system difficult to break. From the study, Thangavel et al prove that the proposed enhanced system is highly secure and not easily breakable as compared to RSA and the modified RSA algorithms used for comparison purposes [14].

## 5.8. A Hybrid Security Algorithm for RSA Cryptosystem.

**I**n another work, Panda and Chattopadhyay propose a new hybrid security algorithm for RSA where the computation of public key P and private key Q depends on the value of N, where N is the product of four reduced-size prime numbers [5]. This increases the complexity of factorizing the variable N, and therefore enhancing the security [14]. Similar to what is proposed in [14], the computation of P is not direct, where in order to determine the value of P1, the values of p1 and p2 must be obtained. Additionally, the value of **w** is distributed instead of **M** in the public key to hidden the value of **M** when the factorization attack is attempted. The following is a summary of the new hybrid security algorithm for RSA.

***Key Generation:***

**A-** Select four random distinct prime numbers a,b, c and d.
**B-** Find Public Key (P), Private Key (Q), and random number (w).
**C-** Procedure (a, b, c, d, P, Q and w)

1. $x \leftarrow a * b$
2. $y \leftarrow c * d$
3. $M \leftarrow x * y$
4. Calculate Euler $\emptyset$ () of x, y and M
   a. $\emptyset(x) \leftarrow (a-1) * (b-1)$
   b. $\emptyset(y) \leftarrow (c-1) * (d-1)$
   c. $\emptyset(M) \leftarrow \emptyset(x) * \emptyset(y)$
5. Generate a random number p1, such that,
   $\gcd(p1, \emptyset(x)) = 1,\ 1 < p1 < \emptyset(x)$
6. Generate a random number p2, such that,
   $\gcd(p2, \emptyset(y)) = 1,\ 1 < p2 < \emptyset(y)$
7. Calculate $P1 \leftarrow p1^{p2} \bmod M$
8. Generate a public key P, such that,
   $\gcd(P, \emptyset(M) * P1) = 1,\ 1 < P < \emptyset(M) * P1$
9. Calculate the private key Q, such that,
   $Q \leftarrow P^{-1} \bmod (\emptyset(M) * P1)$
10. Compute a random number w, such that,
    If a>b
    Satisfy $x - a < w < x$ and $\gcd(w, x) = 1$

Else if a<b
Satisfy x− b < w < x and gcd (w, x) = 1

***Encryption :***

**INPUT**: Select Plain text (T), Public key (P)
and Random number (w).
**OUTPUT**: Find Cipher text (C).
Begin
Procedure (T, P, w and C)
$C \leftarrow T^P \bmod w$
End Procedure
End

***Decryption :***

**INPUT**: Select Cipher text (C), Private key (Q) and Random number (w).
**OUTPUT**: Find Plain text (T).
Begin
Procedure (C, Q, w and T)
$T \leftarrow C^Q \bmod w$
End Procedure
End

## 5.9. A Modified RSA Algorithm for Security Enhancement and Redundant Messages Elimination Using K-Nearest Neighbor Algorithm.

**A**bdulameer proposed another effective solution to enhance the security of RSA method to solve the issue of having cipher text being the same as the plain text in some values of n, which is the product of p and q. This was done by eliminating the redundant messages which occurred in some values of n and the product of two prime numbers [8]. The proposed solution depends on the replacement of the value of n using a secure agreement distance in a set of all available prime numbers. The next step is the selection of either one of the primes responsible for generating an alternative n or both primes from the set. The method also helps to eliminate one parameter of the public key which is the value of n and therefore making the method more secure [8]. Owing to its complexity, and compared to other proposed algorithms, the proposed algorithm takes a long time to be executed [8]. The following is a summary of the algorithm:

1. Select PR (a set of prime number p and q)
2. PR = p1, q1, p2, q2…pn ,qn}
3. Divide PR into subsets S= {s1,s2,….Sn} such that each Si contains a limited numbers of primes .
4. Each Si={psi1,psi2, …..psin}
5. Choose two prime numbers p & q from PR
6. Calculate ( n = p . q )
7. Calculate ∅ (n) = (p-1).(q-1)
8. Let e be the public key
9. Let d be the private key
10. $c = m^e \bmod n$.
11. If c=m then :

**Sender operation:**

1: Choose d1 of the one of subsets si in S for the secure class
2: Choose d2 inside si to pick one alternative prime p'
3: Compute n'=p'*q

4: Compute ∅ (n')=(p'-1)*(q-1)
5: Choose alternative public key , lets e'
6: Generate the corresponding private key d'
7: Compute the ciphertext C'=$m^{e'}$ mod n'
8: Combine the agreement factor f with the new ciphertext and send C"" as :C"=[C',f ]

## 5.10. RSA Enhancement Using Exponential Powers, N Prime Numbers, Multiple Public Keys, and K-NN Algorithm.

**M**athur et al present a modified approach to RSA, which is an enhancement to the traditional RSA method [7]. The enhancement includes exponential powers, n prime numbers, multiple public keys, and K-NN algorithm. The modified approach also provides a feature of verification at both sides of the sender and the receiver. The limitation of the proposed approach is that the time required for encryption and decryption is higher than in the original RSA [9]. The following steps summarize the proposed approach:

***Key Generation***

**1.** Select four prime numbers A, B, C, and D
**2.** Calculate L = A . B . C . D
**3.** Calculate ∅(L)= (A- 1).(B -1).(C -1).(D- 1)
**4.** Calculate J (public key), such that GCD (J, ∅ (L)) = 1
**5.** Calculate K (private key), such that K . J mod ∅ (L) = 1
**6.** Choose random number N and O. O should not be relative prime to ∅ (L)
**7.** Choose two numbers P and Q, such that Q = PJ

***Encryption***

**1.** Convert the message that has to be encrypting into their respective ASCII values
**2.** Calculate 'E' for each ASCII value, such that E = (ASCII VALUE $^{Q/P)\,K}$ mod L)
**3.** Calculate R1, as it encrypts the message and gives back cipher text of given plain text R1= (message) $^{K}$ mod L
**4.** If the ASCII values and values of R1 comes same, then apply K- Nearest Neighbour algorithm
**5.** After that calculate R2 = (message * $N^{R1}$) mod L
**6.** Verification $H(m)^{Y}$ = ($R2^{O}$ * $E^{R1}$) mod L

***Decryption***

**1.** Calculate plain text back again from cipher text (m) = $R1^{J}$ mod L
**2.** Verification H $(m)^{Y}$ mod L

## 5.11. A Modified and Secured RSA Public Key Cryptosystem Based on "n" Prime Numbers.

**I**n the bid to address some weakness RSA algorithm computation, Islam et al. proposed a modified RSA (MRSA) scheme [17]. Under key generation, the modified scheme involved the use of 'n' distinct prime numbers. The private and the public described in the MRSA comprises of three different components. One of these components is N, which is the product of four randomly selected large prime numbers w, x, y, and z. Alone, the public key is made up of e, f, and N components, with e and f being randomly selected numbers. This, together with factoring of 'N' adds complexity to the key generation function of the scheme. Out of all these values, it's only the value of N that is in both private and public key. This therefore implies that, with the value of N only, it is impossible for the attacker to determine the value of all the other four large

prime numbers, which also makes it impossible to determine the value of e and f. On the other hand, the private key comprises of three different components d, g, and N. Below is the summary of the key generation for the MRSA

***Key Generation***

- The first step is random selection of four large prime number w, x, y, and z
- Next is random selection of public key exponent e, f, and N
- Followed by computing the private key exponent d, g, and N

Procedure:

1- Compute the value of N = w . x . y . z
2- Compute the Euler phi value of N
$\Phi(N) = ( w - 1 ) . ( x - 1 ) . ( y - 1 ) . ( z - 1 )$
3- Find a random variable e, satisfying
$1 < e < \Phi(N)$ and gcd $(e, \Phi(N)) = 1$
4- Find another random variable f, satisfying
$1 < f < \Phi(N)$ and gcd$( f , \Phi(N )) = 1$
5- Compute a random number d, such that
$d . e \equiv 1 \bmod \Phi(N)$
6- Compute another random number g, such that
$f . g \equiv 1 \bmod \Phi(N)$

While the public key exponent is used for encryption, the private key exponent is used for decryption in the MRSA. Also, besides being related to 'N' both encryption and decryption consist of four random components 'e', 'f', 'd', and 'g', which adds to the complexity of the algorithm [17]. Below is the summary for encryption and decryption.

***Encryption***

- The input here is the plaintext, M (<N) and Public Key exponent {e, f, N}.
- The output includes ciphertext X.

Procedure:

$X \leftarrow (M^{e} \bmod N)^{f} \bmod N$

***Decryption***

- The Input is the Ciphertext message X and Private key exponent: {d, g , N}.
- The output will be the decrypted plain text, Y

Procedure:

$Y \leftarrow (X^{g} \bmod N)^{d} \bmod N$

"In its implementation in JAVA 8, Islam et al. pointed out that MRSA comes with different crucial parameters that affect the level of security and speed of the algorithm" [17]. The increased length of the modulus invokes complexity of decomposing it into factor, and therefore increasing the length of the private key, which in return makes it difficult to detect the key. The analysis of the MRSA in relation to RSA indicated that the time of key generation of MRSA is higher than that of the traditional RSA algorithm. The increase in the amount of time taken to generate key helps in increasing the time an attacker will take to break the system. This, therefore, is of great importance in enhancing the security through increased complexity as compared to the traditional RSA. As such, as compared to the traditional RSA, MRSA is more secure due to its added level of complexity.

## 6. CONCLUSION

upholding the confidentiality, integrity, availability, and non-repudiation of information and data sent across networks requires more than just cryptography. Over the years, the RSA algorithm has

been applied in different areas to enhance the security of information through encryption and decryptions. However, the advancement in computing technology and hacking methods have rendered the original RSA algorithm ineffective in data protection. It is in light of this that different researchers have focused on the method to enhance the RSA algorithm through the addition of more complexity to the algorithm.

## 7. ACKNOWLEDGMENTS

A biggest thanks to all faculty of the college of science and engineering in Hamad bin Khalifa University (HBKU), especially to Dr. Samir Brahim Belhaouari for his usual motivation and support. And special thanks to my family who are the great supporters during my master's degree journey.

## REFERENCES

[1] S. Gupta and J. Sharma, "A hybrid encryption algorithm based on RSA and Diffie- Hellman," 2012 IEEE International Conference on Computational Intelligence and Computing Research, Coimbatore, 2012, pp. 1-4. doi: 10.1109/ICCIC.2012.6510190

[2] S. A. Jaju and S. S. Chowhan, "A Modified RSA algorithm to enhance security for digital signature," 2015 International Conference and Workshop on Computing and Communication (IEMCON), Vancouver, BC, 2015, pp. 1-5. doi: 10.1109/IEMCON.2015.7344493

[3] R. Patidar and R. Bhartiya, "Modified RSA cryptosystem based on offline storage and prime number," 2013 IEEE International Conference on Computational Intelligence and Computing Research, Enathi, 2013, pp. 1-6. doi: 10.1109/ICCIC.2013.6724176

[4] R. Minni, K. Sultania, S. Mishra and D. R. Vincent, "An algorithm to enhance security in RSA," 2013 Fourth International Conference on Computing, Communications and Networking Technologies (ICCCNT), Tiruchengode, 2013, pp. 1-4. doi: 10.1109/ICCCNT.2013.6726517

[5] P. K. Panda and S. Chattopadhyay, "A hybrid security algorithm for RSA cryptosystem," 2017 4th International Conference on Advanced Computing and Communication Systems (ICACCS), Coimbatore, 2017, pp. 1-6. doi: 10.1109/ICACCS.2017.8014644

[6] P. Chaudhury et al., "ACAFP: Asymmetric key based cryptographic algorithm using four prime numbers to secure message communication. A review on RSA algorithm," 2017 8th Annual Industrial Automation and Electromechanical Engineering Conference (IEMECON), Bangkok, 2017, pp. 332-337. doi: 10.1109/IEMECON.2017.8079618

[7] S. Mathur, D. Gupta, V. Goar and M. Kuri, "Analysis and design of enhanced RSA algorithm to improve the security," 2017 3rd International Conference on Computational Intelligence & Communication Technology (CICT), Ghaziabad, 2017, pp. 1-5. doi: 10.1109/CIACT.2017.7977330

[8] Abdulameer K. Hussain, "A Modified RSA Algorithm for Security Enhancement and Redundant Messages Elimination Using K-Nearest Neighbor Algorithm", IJISET - International Journal of Innovative Science Engineering & Technology, vol. 2, no. 1, January 2015, ISBN 2348-7968.

[9] N. M. S. Iswari, "Key generation algorithm design combination of RSA and ElGamal algorithm," 2016 8th International Conference on Information Technology and Electrical Engineering (ICITEE), Yogyakarta, 2016, pp. 1-5. doi: 10.1109/ICITEED.2016.7863255

[10] Barakat, Mohamed, Christian Eder, and Timo Hanke. "An Introduction to Cryptography." (2018).

[11] Vishal Garg, Rishu, Improved Diffie-Hellman Algorithm for Network Security Enhancement, Int.J.Computer Technology & Applications, Vol 3 (4), 1327-1331.

[12] Emmanual Bresson, Dynamic group Diffi-hellman key exchange under standard assumption, Proceeding of EUROCRYPT, LNSC 2332, page no. 321-336, 2002.

[13] David A. Carts, A Review of the Diffie-Hellman Algorithm and its Use in Secure Internet Protocols SANS Institute Reading Room.

[14] Thangavel M, et al., An Enhanced and Secured RSA Key Generation Scheme (ESRKGS), Journal of Information Security and Applications (2014).

[15] T. Elgamal, "A public key cryptosystem and a signature scheme based on discrete logarithms," in IEEE Transactions on Information Theory, vol. 31, no. 4, pp. 469-472, July 1985.doi: 10.1109/TIT.1985.1057074

[16] Rivest, Ronald L., Adi Shamir, and Leonard Adleman. "A method for obtaining digital signatures and public-key cryptosystems." Communications of the ACM 21.2 (1978): 120-126.

[17] Islam, M.A., Islam, Md.A., Islam, N. and Shabnam, B. (2018) A Modified and Secured RSA Public Key Cryptosystem Based on “n” Prime Numbers. Journal of Computer and Communications, 6, 78-90.

[18] Kocher, Paul C. "Timing attacks on implementations of Diffie-Hellman, RSA, DSS, and other systems." Annual International Cryptology Conference. Springer, Berlin, Heidelberg, 1996.

[19] Brumley, David, and Dan Boneh. "Remote timing attacks are practical." Computer Networks 48.5 (2005): 701-716.

[20] Bleichenbacher, Daniel. "Chosen ciphertext attacks against protocols based on the RSA encryption standard PKCS# 1." Annual International Cryptology Conference. Springer, Berlin, Heidelberg, 1998.

[21] Aciiçmez, Onur, Çetin Kaya Koç, and Jean-Pierre Seifert. "On the power of simple branch prediction analysis." Proceedings of the 2nd ACM symposium on Information, computer and communications security. ACM, 2007.

[22] Pellegrini, Andrea, Valeria Bertacco, and Todd Austin. "Fault-based attack of RSA authentication." Proceedings of the conference on Design, automation and test in Europe. European Design and Automation Association, 2010.

[23] Kleinjung, Thorsten, et al. "Factorization of a 768-bit RSA modulus." Annual Cryptology Conference. Springer, Berlin, Heidelberg, 2010.

[24] Rosen, Kenneth H. Discrete mathematics and its applications. New York: McGraw-Hill, 2011.

[25] Katz, Jonathan, et al. Handbook of applied cryptography. CRC press, 1996.

## AUTHORS

Engr. Shaheen Saad Al-Kaabi, 30 years old. Student in Master of Science in Cypersecurity, Hamad Bin Khalifa Unvirsty ( HBKU )