METHODS TOWARD ENHANCING RSA ALGORITHM : A SURVEY

Engr. Shaheen Saad Al-Kaabi and Dr. Samir Brahim Belhaouari

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

ABSTRACT

Cryptography defines different methods and technologies used in ensuring communication between two parties over any communication medium is secure, especially in presence of a third part. 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. Initially, cryptography was applied in the securing of military secrets. For instance, to communicate to his commanders and soldiers in the battle field, Julius Caesar used Caesar ciphertext to conceal information from the enemies or unauthorized parties. Since then, the encoded messages have been used by government, private entities, and militaries around the world to protect sensitive information. From these applications, cryptography can be defined as methods or techniques used in ensuring that exchange of information between two parties remain secure, and the information itself remains confidential, authentic, and maintains high level of integrity [8].

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. On the other hand, authentication helps to ensure that the information or data is true and is from the expected source. Under non-repudiation, cryptography ensures that an author of a given statement or piece of data cannot deny it. Cryptography therefore plays a crucial role in ensuring that the data is confidential, authentic,

and maintains its integrity while on transit or storage. 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 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 cipher text 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 cipher text. The recovery process of plaintext from the cipher text 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. This also implies that in case a third party gains access to the key, he/she can easily decrypt the information. As such, it is crucial to ensure that the key is kept as secret as possible. The figure below illustrates the encryption and decryption process in symmetric key cryptography:

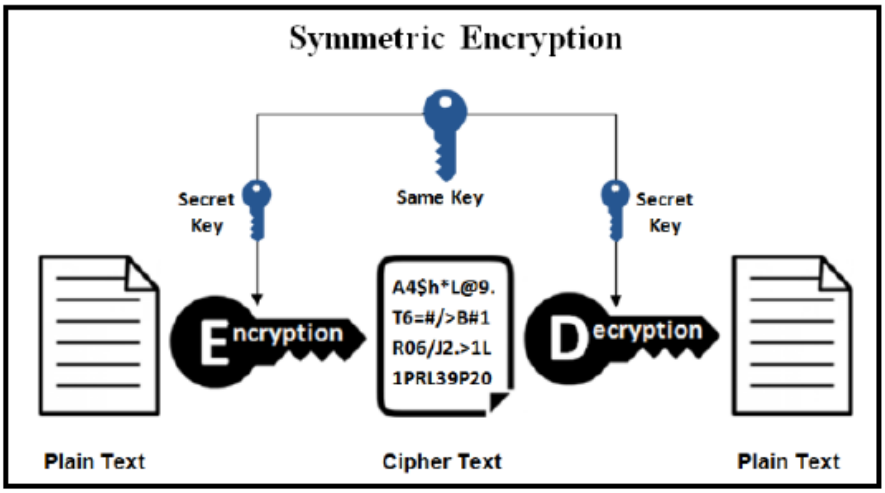

Figure 1. Symmetric key Cryptography

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 Two fish. 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. On the other hand,

AES make it possible to select a key with either128, 192, or 256 bit length. Each of these keys encrypts using different rounds of processing; the 128, 192, and 256 bit key encrypt in 10, 12, and 14 rounds of processing respectively.

1.1.2. ASYMMETRIC KEY CRYPTOGRAPHY

Unlike symmetric key cryptography, asymmetric key cryptography uses two separate keys, where one is a private key for data decryption, and the other one is a public key for data encryption. This implies that the public key is published for anyone to see, but the private key is kept a secret. As such, anybody with access to the public key encrypts the data, but only the intended person (with private key) can decrypt the data. The figure below is an illustration of how asymmetrical cryptography works:

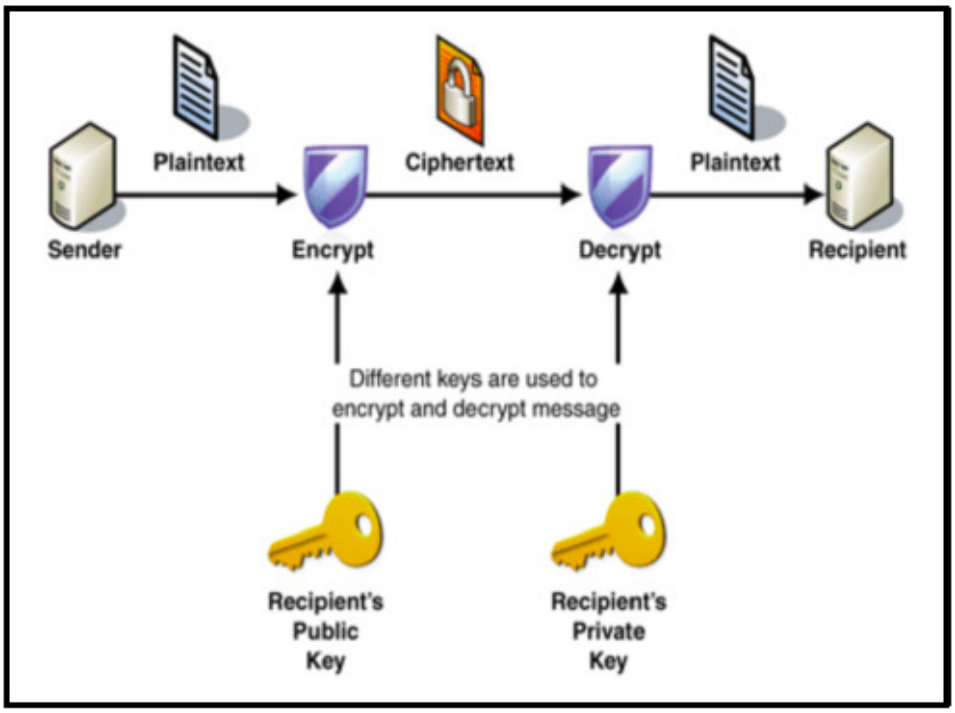

Figure 2. Asymmetric key Cryptography

One of the key advantages 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. This therefore means that only the public key is transmitted during the communication between the two parties. 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 [14], 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, before being used for encryption and decryption of data. 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** . $(\mathbf{p} \neq \mathbf{q})$.
b. Calculate the common module **n** such that $\mathbf{n = p \cdot q}$.
c. Calculate the Euler's $\mathbf{phi : \emptyset(n) = (p - 1).(q - 1)}$
d. Select integer **e** which is the encryption (public) key such that :
$\mathbf{GCD\ (\emptyset(n)\ ,\ e\ ) = 1\ ;\ 1 < e < \emptyset(n)}$
e. Calculate **d** is the decryption (private) key such that $\mathbf{d \models e^{-1}\ (mod\ \emptyset(n)\ )}$.
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. In essence, Randomized Polynomial Time Monte Carlo approach, such as SOLOVAY-STRASSEN, or MILLER-RABIN approach can be used to test whether the large random numbers selected are prime numbers (primality testing algorithms). Randomized Polynomial Time Monte Carlo algorithms are fast primality tester that can help in testing the selected numbers in polynomial in in log2 n [18]. Prime number theorem ((N) ≈ N/ ln$N$, where (N)= primes ≤ N) can be used in this case to determine the number of random integers that need to be tested before finding prime. If p is considered as 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" [18].

2.2. ENCRYPTION

The process of encoding a message is defined by the encryption scheme used. 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:

$$\mathbf{C = M_e \bmod n}$$

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. Only individuals with access to the private key can decrypt the data in RSA algorithm. 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:

$$\mathbf{M = C^d \bmod n}$$

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

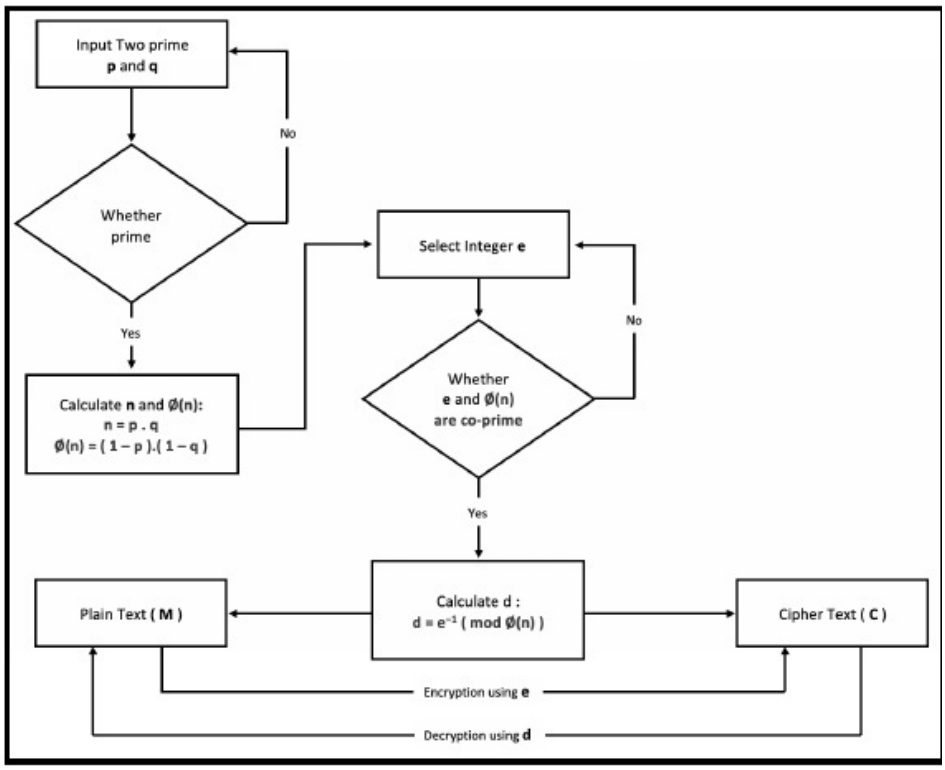

Figure 3. Flow Design of RSA Algorithm

2.4. 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. Compared to other symmetric cryptographic systems, RSA is relatively slow, owing to the fact that it uses public key. Also, the size of the product of the two prime numbers selected (N=P X Q) limits the length of the plain text that can be encrypted, as well as the fact that initialization of each of the RSA process required random selection of two large prime number (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]. In line with this, Jaju and Chowhan also observed that the execution time for RSA algorithm was relatively higher as compared to that of 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]. Other researchers who pinpointed several limitations in regard to RSA algorithm included Mathur et al.[6] who delved in to improving the model by enhancing security and elimination of redundant messages using K-nearest neighbor algorithm.

2.5. FACTORING RSA MODULUS

Factoring the public modulus is the most evident way to attack RSA cryptosystem. Some of the algorithms that are considered to be the most effective on factoring very large numbers include Number Field Sieve (NFS), Quadratic Sieve (QS), and Elliptic Curve Factoring Algorithm (ECFA) [18]. 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 [18].

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, a study by Kleinjung et al.(2009) points out some implication for RSA following the use of NFS to factor out the a number with 768-bit and RSA -768[17]. 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, while the major parts was computed at two different locations on four clusters that were running in parallel to each other. Where the computation involved a consecutive sequence of matrix time's vector multiplication, the block wiedemann algorithm for the matrix was used to facilitate it. As a result of increased independent parallelization that ensued, a number of challenges that needed to be addressed before handing more complex issues were encountered. The first step was to show how flexibility could be attained in the number, as well as different clusters could contribute to achieving a scalable solution. This helped in solving a matrix step that would have otherwise been nine times harder.

Approximately one terabyte of memory was required during one of the sub-steps. 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. The first step in this case was to factor a composite integer n. This was achieved by determining the solution of the integer (x; y) of the congruence of square x2 ° y2 mod n. At this point, the author only hoped that n had been factored by writing it as a product GCD of (x –y, n).

The chances of finding a non-trivial factor of n using this method for a random pair are at around 0.5[17]. The Morrison-Brillhart approach solves the equation x2 ° y2 mod n by combining the congruencies of smooth squares. The paper also comprises of discussion on the implication for moduli larger than RSA-768. To assist in completion of the project, the author sought the assistance of well-informed contributors who dedicated their time and computational resources to see the project through. This is despite the fact that it was possible to run NFS as a BOINC

project. This in return allowed the authors to target a reasonable completion date that is easily manageable despite the intensive oversieving. The author also insists on communication of a fair amount of data to the central storage area. This was based on the fact that most clients fail to communicate during such project. This can be achieved through organizational efforts that may help in occasional recovery from errors that may results from unplugged network cables, servers that have been switched off, or faulty raids and constantly increasing the backup drives. [17].

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 [8].

3. ENHANCEMENT PROPOSALS FOR RSA ALGORITHM

3.1. 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 astounding and extensive algorithm that has been applied on the internet to secure different connectivity protocols. Examples of such protocols include SSL, IPsec, and SSH [1, 9, 11]. 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 [10]. 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.

The sender will use K for the encryption of the plain text PT. The sender will then send the encrypted PT to the receiver as cipher text CT. Once the receiver received the cipher text CT, he/she uses k to decrypt it for the recovery of the plain text PT.

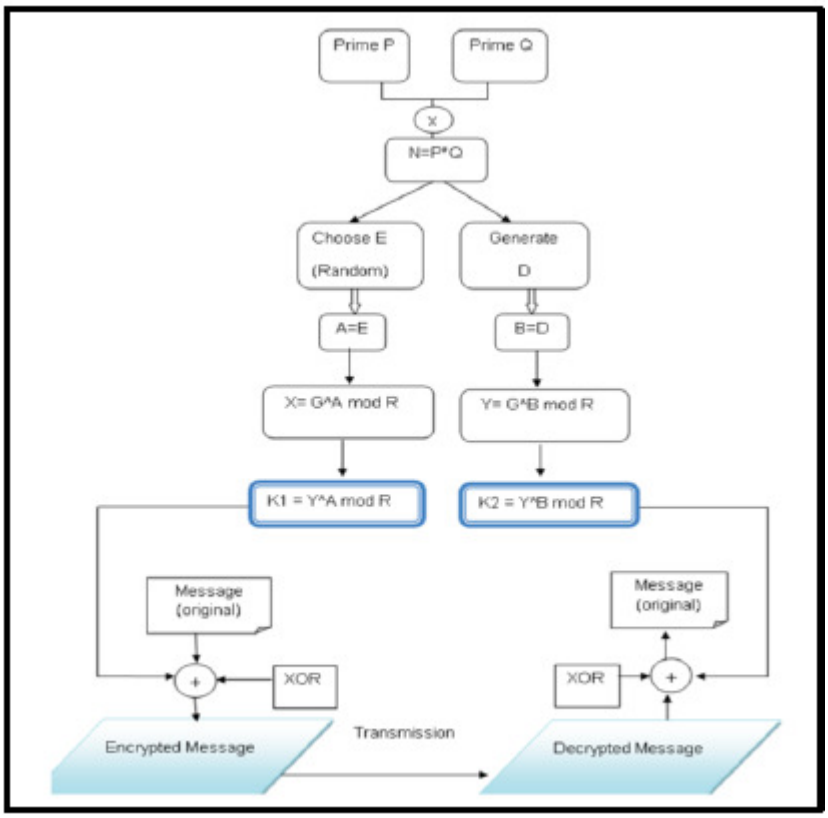

Figure 4. Flow design of a hybrid RSA & Diffie-Hellman algorithm

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].

3.2. COMBINATION OF RSA AND ELGAMAL ALGORITHM.

Iswari also proposed another enhancement to the RSA algorithm by combining it with the ElGamal algorithm [7]. 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 [7]. The security strength of the ElGamal algorithm lies on the difficulty of computing the discrete logarithm [13]. 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 $y = g^x \bmod p$.
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. Ø (r) = (p -1).(q-1)
4. Generate Random number PK (encryption key), where GCD (PK, Ø (r)) = 1.
5. Compute decryption key SK = PK-1 mod Ø (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, Ø(r)) = 1.
9. Recalculate private key again for RSA algorithm (SK = y-1 mod Ø(r)).

3.3. MODIFIED RSA CRYPTOSYSTEM BASED ON OFFLINE STORAGE AND PRIME NUMBER.

Patidar and Bhartiya also proposed new algorithm concept to improve the performance of the traditional RSA algorithm during exchange of information between two parties across a 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 . q . r).
3. Calculate Ø (n) = (p-1).(q-1).(r-1).
4. Calculate e such that GCD (e, Ø (n)) =1 and 1<e<Ø(n).
5. Encrypt the message M where M<n and encrypt with public key e such that C= Me mod n.
6. Calculate private key $d = e^{-1} \pmod{Ø (n)}$.
7. Decrypt the message M such that $M=C^{d} \bmod n$.

The storage of the keys for the proposed system is done offline before the process is initialized. This leads to enhancement of the speed required for encryption and decryption, as compared to the traditional RSA [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 Ø (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.

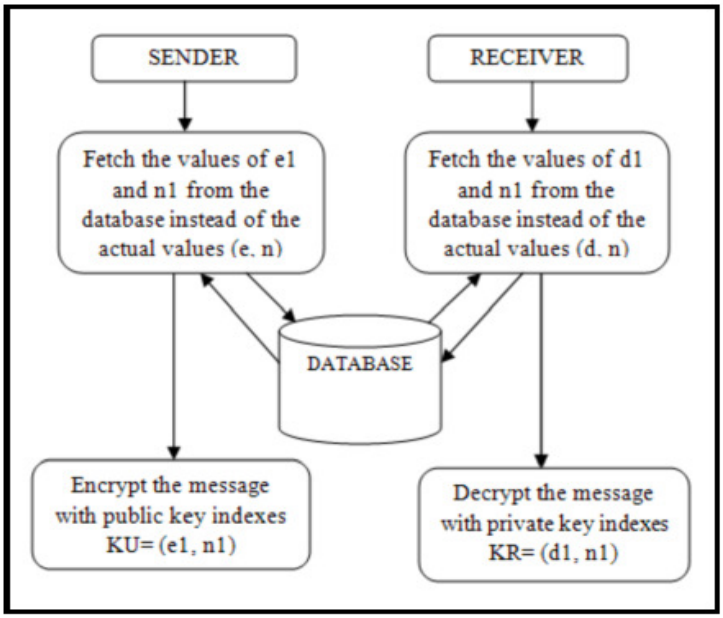

Figure 5. the mechanism of fetching the values from/to the data base.

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 [6].

3.4. ENHANCE THE SECURITY OF RSA BY ELIMINATING THE DISTRIBUTION OF MODULUS N.

Additionally, 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 Ø (N) = (A -1).(B – 1).
4. Calculate k1 based on the following conditions
• Ö N < k1 < Ø (n)
• GCD (k1, Ø (N)) = 1 that is k1 and Ø(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 = PTk1 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 = √ (CTk2 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.

3.5. A MODIFIED RSA ALGORITHM TO ENHANCE SECURITY FOR DIGITAL SIGNATURE.

In 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 ¹ q ¹ 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 Ø (n) = (p – 1).(q – 1).(r – 1).
4. Calculating integer e which is based on
• Ö n < e < Ø (n)
• GCD (Ø (n), e) = 1 that is e and Ø(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 $d = e^{-1} \pmod{Ø (n)}$
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]

3.6. AN ENHANCED AND SECURED RSA KEY GENERATION SCHEME (ESRKGS).

Thangavel et al. also proposed a modified and enhanced scheme based on RSA public-key cryptosystem using four prime numbers [12]. 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” [12]. The parameter E1 also helps in increasing the complexity of the system [12]. 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" [12].

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 and therefore contributing to the increased complexity required to intrude a communication channel [12]. 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 has proved that the security of the proposed system is top notch and has increased complexity for the attacker. This makes it more secure than the traditional RSA. [12].

3.7. A HYBRID SECURITY ALGORITHM FOR RSA CRYPTOSYSTEM.

In 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 [12]. Similar to what is proposed in [12], 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 Ø () of x, y and M
   a. $Ø(x) \leftarrow (a-1) * (b-1)$
   b. $Ø(y) \leftarrow (c-1) * (d-1)$
   c. $Ø(M) \leftarrow Ø(x) * Ø(y)$
5. Generate a random number p1, such that,
   $\gcd(p1, Ø(x)) = 1,\ 1 < p1 < Ø(x)$
6. Generate a random number p2, such that,
   $\gcd(p2, Ø(y)) = 1,\ 1 < p2 < Ø(y)$
7. Calculate $P1 \leftarrow p1^{p2} \bmod M$
8. Generate a public key P, such that,
   $\gcd(P, Ø(M) * P1) = 1,\ 1 < P < Ø(M) * P1$
9. Calculate the private key Q, such that,
   $Q \leftarrow P^{-1} \bmod (Ø(M) * P1)$
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)
$C \leftarrow TP \bmod w$
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)
$T \leftarrow CQ \bmod w$
End Procedure
End

3.8. RSA ENHANCEMENT USING EXPONENTIAL POWERS, N PRIME NUMBERS, MULTIPLE PUBLIC KEYS, AND K-NN ALGORITHM.

Mathur et al present a modified approach to RSA, which is an enhancement to the traditional RSA method [6]. This enhancement comprised of several algorithms, including K-NN algorithm, use of prime numbers and exponential powers, as well as the use of multiple public keys. 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 [7]. 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 Ø(L)= (A- 1).(B -1).(C -1).(D- 1)
4. Calculate J (public key), such that GCD (J, Ø (L)) = 1
5. Calculate K (private key), such that K . J mod Ø (L) = 1
6. Choose random number N and O. O should not be relative prime to Ø (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
   $R1 = (\text{message})^{K} \bmod L$
4. If the ASCII values and values of R1 comes same, then apply K- Nearest Neighbour algorithm
5. After that calculate $R2 = (\text{message} * N^{R1}) \bmod L$
6. Verification $H(m)^{Y} = (R2^{O} * E^{R1}) \bmod L$

Decryption

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

3.9. A MODIFIED AND SECURED RSA PUBLIC KEY CRYPTOSYSTEM BASED ON "N" PRIME NUMBERS.

In the bid to address some weakness RSA algorithm computation, Islam et al. proposed a modified RSA (MRSA) scheme [15]. 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. On its own, the public key comprises of three components (e, f, and N). Among the three, component e and f are selected randomly.

This, together with factoring of 'N' adds complexity to the key generation function of the cheme. Out of all these values, it's only the value of N that is in both private and public key. This is to imply that it is not possible for the attacker to break the system with the value of N only, as

he/she needs to calculate the value of all the other four large prime numbers. This means that it is impossible for the attacker to also compute 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 $N = w \cdot x \cdot y \cdot z$
2- Compute the Euler phi value of N
$\Phi(N) = (w - 1) \cdot (x - 1) \cdot (y - 1) \cdot (z - 1)$
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
$d \cdot e \equiv 1 \bmod \Phi(N)$
6- Compute another random number g, such that
$f \cdot g \equiv 1 \bmod \Phi(N)$

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 [15]. 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" [15]. 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 increased amount of time required for key generation means an increase in the time required for the attacker 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.

3.10. ENCRYPTION ALGORITHM USING DUAL MODULUS

In the bid to address the weakness observed in RSA algorithm, Goel, (2017) proposed a novel RSA-based algorithm capable of resisting attacks that are common in RSA [16]. Some of the key features of this algorithm include a double mod operation-based encryption using two private keys, a double mod operation-based decryption using two public key, and more than two large prime numbers for generating modulus value. The three mentioned features are aimed at enhancing the security of the message by increasing the time required for key generation, encryption, and decryption. The dual modulus scheme is used together with the two large public and private keys, and therefore making it harder to factorize them to gain access to the private key [16].

Key Generation

- Four random prime numbers p1 and p2, q1 and q2 are selected
- Compute n1 =p1 x p2 and n2 = q2 x q2
- Compute Ø1 = LCM ((p 1 -1), ((p2-1)) and φ2 = LCM ((q1-1), (q2-1))
- Two integers e1 and e2 are selected such that 1 < e1< φ1 and 1<e2
- Secret exponents d1 and d2 are computed such that e1 x d1 mod φ1 = 1 and e2 x d2 mod φ2 = 1.
- The public key is (n1, n2, e1, e2) and the private key is (n1, n2, d1, d2).
- Values d1, d2, p1, p2… q1, q2… φ1 and φ2 are kept secret.

Encryption

- Sender Obtain the recipient's public key (n1, n2, e1, e2).
- Plain text message represented as a positive integer m (0<= m < n1 < n2)
- Computes the cipher text c = $((m^{e1} \bmod n1)^{e2} \bmod n2)$.
- Sends the cipher text c to the receiver

Decryption

- The intermediate value C2 can be evaluated as
$C2^{d2} \bmod n2 = (C1^{e2} \bmod n2)\ d2 \bmod n2 = C1$ (by RSA) (Since C2 is reversible cipher text)
- Now, $C1^{d1} \bmod n1 = (m^{e1} \bmod n1)^{d1} \bmod n1 = m$ (by RSA) (Since C1 is reversible)

The two algorithms have similarities and differences. Some of the similarities between the two include the fact that both of them are asymmetric algorithms using two pairs of keys. One of the keys is used to encrypt the data such that it can only be decrypted using the other pair. Another similarity is that both algorithms use a similar process to generate the key. However, they cannot generate from each other. The following is a summary of the proposed algorithm [16].

On the other hand, some of the differences include the fact that RSA uses two different keys to encrypt and decrypt while the proposed algorithm uses four different keys for the same. This

increases the complexity of the algorithm, which in return enhances its security. The use of dual private keys helps to ensure that if an intruder detects a single private key even then it will be impossible to decrypt the ciphertext, as the second key is still secure. In addition, if a brute force attack is launched on the two keys, the time takes to compute key will increase exponentially. This means that the double process used in the proposed framework increases the complexity of factorization. In case an attacker detects a single key, it will still be impossible to decrypt the message [16].

4. 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.

5. 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] 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

[7] 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

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

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

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

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

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

[13] 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

[14] 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.

[15] 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.

[16] Manu and A. Goel, "Encryption algorithm using dual modulus," 2017 3rd International Conference on Computational Intelligence & Communication Technology (CICT), Ghaziabad, 2017, pp. 1-4. doi: 10.1109/CIACT.2017.7977331.

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

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

AUTHOR

Engr. Shaheen Saad Al-Kaabi, 30 years old. B.S. in Electrical Engineering, University of Colorado at Denver (UCD) M.S. in Cypersecurity, Hamad Bin Khalifa Unvirsty (HBKU)