1. Introduction

Enormous quantity of data is being transmitted from different parts of the world. With this, computer information and network technology had been widely used and utilized in the everyday life of people. As the data transference turned to be a major trend, the need for data security becomes vital and significant. Cryptography is one of the methods to protect and secure the data and guarantee its confidentiality.

The term cryptography is originated from the Greek word kryptos which implies secret writing. It is the science and art of altering the messages into another form to make them secure and invulnerable to attacks. Technically, cryptography is the conversion of plaintext (original message) to ciphertext (hidden message). With cryptography, only the owner, together with the authorized persons, is allowed to read and write an information, and to keep it hidden from third-party users (intruders). The two very important process of securing the message are encryption and decryption. Additionally, cryptography is differentiated into two primary categories: Symmetric and Asymmetric Key Cryptography. The Symmetric Key Cryptography also called as Private Key Cryptography uses the same key for the two process. Whereas Asymmetric Key Cryptography also known as Public Key Cryptography uses two different keys for the two process. Many cryptographic algorithms have emerged over time due to different attacks on the existing ones. For example, the brute force attack, differential cryptanalysis and linear cryptanalysis are the attacks made against Data Encryption Standard (DES) algorithm. Triple DES was attacked using chosen...
plaintext attacks or known plaintext attacks. On the other hand, Advanced Encryption Standard (AES) was attacked using brute force attack, biclique attack and related-key attacks while the attack on Blowfish algorithm was second order differential attack.

With these forms of attacks on different cryptographic algorithms, the security needed for every data had also arise and traditional encryption (single data encryption) had eventually not been sufficient for the protection and security of every data. As the requirement for security arises, advancement of new, simple and effective security frameworks has been preferred by many people. One way to develop the security of data that can also answer today’s problems in security is through hybridization of prevailing cryptographic algorithms.

2. Existing work

2.1 Symmetric Key Cryptography
Symmetric Key Cryptography or the Shared Key Cryptography utilizes same key for the process of encryption and decryption.

For instance, the message that will be sent by the sender is encrypted using a secret key and will be received by the receiver which is decrypted using the same secret key. Figure 1 shows the utilization of same key in encryption and decryption of the Symmetric Key Cryptography.

Symmetric Key Cryptography algorithms are simple and subsequently have a quick execution time. It is normally utilized in long messages. Some of the algorithms that uses symmetric key cryptography are: (1) International Data Encryption Algorithm (IDEA); (2) Advanced Encryption Standard (AES) Algorithm; (3) Data Encryption Standard (DES) Algorithm; (4) Blowfish Algorithm; and (5) Triple DES (3DES) Algorithm.

2.2 Asymmetric Key Cryptography
Asymmetric Key Cryptography or the Public Key Cryptography uses a pair of keys for encryption and decryption process. These pair of keys is the public key and the private key. The public key is recognized by everyone while the private key is kept confidentially by the receiver.

For instance, the message is encrypted with the public key (recognized by all other people) and is decrypted by the receiver with his/her private key (recognized only by the receiver). Figure 2 shows the utilization of keys in encryption and decryption of Asymmetric Key Cryptography.

In addition, the number of keys required in Asymmetric Key is small; however, it is not effective in long messages. Some of the algorithms that uses Asymmetric Key Cryptography are: (1) Elgamal Encryption Algorithm; (2) Rivest-Shamir-Adleman (RSA) Algorithm; (3) Elliptic Curve Cryptography (ECC); and (4) Digital Signature Algorithm (DSA).

2.3 Secure and Fast Chaos-based Cryptosystem
Khare, Shukla and Silakari had introduced a new algorithm for cryptosystems. They utilized the properties of chaotic framework like ergodicity, sensitive reliance on starting condition and system parameters in encryption process to improve the security. A detailed discussion with this new chaotic encryption algorithm can be found in.

This proposed algorithm used a Symmetric key encryption technique which utilized at a minimum of one key for encryption and decryption process. Be that
as it may, regardless, such keys are practically identical in the process of encryption and decryption at all moment of time. It implies that distinctive keys are utilized for various messages for upgrading the security. The step by step procedure of the Secure and Fast Algorithm can be described as follows:

(i) Algorithm for key generation

- Set the values of the parameters \((M, A, X_n)\), where \(A = \text{any integer} (1, 2, 3, 4, 5, 6, 7 \ldots)\)
- \(X_n = \text{initial value of chaotic function which is} 2, 3, 4, 5, 6, 7, 8 \ldots\)
- Generate the pseudo random numbers from the equation by means of the chaos logistic function (logistic map) at both ends (sender and receiver):
  \[ X_{n+1} = (A \times X_n(X_n - 1)) \mod 256 \]
- Apply gray code on these random numbers which are produced from \(X_{n+1}\) to develop the keys \(K_1, K_2, \ldots, K_j\).
  - The keys \(K_1, K_2, K_3, \ldots, K_j\) are shown in 8 bit binary form.

(ii) Algorithm for encryption

- Each character of the message is shown in ASCII equivalence, \(P_i = \text{ASCII (character} i)\), where \(P_i = \text{Plaintext}\)
- ASCII character \(P_i\) is transformed into 8 bit binary form and then converted to their corresponding decimal numbers.
- Using the equation \(E_{k_m}(P_i) = C_i\) for all \(i > 0\), and \(m = 1\) to \(j\) for encryption, where
  \(E_{k_m}(P_i)\) is bit wise XORing on plaintext \(P_i\) with single key \(k_m\)
  \(C_i = \text{Ciphertext}\)
  \(E(P_i) = \text{Encryption of the plaintext } P_i\)

Encrypt each characters via utilizing the digital logic XOR function, such that
- \(P_1 = \text{ASCII (character 1)}\)
- \(P_j = \text{ASCII (character} j)\)
- \(P_2 = \text{ASCII (character 2)}\)

\(P_2\) is converted into 8 bit binary numbers
\[ E_{k_2}(P_2) = C_2 \]
\(P_i = \text{ASCII (character} i)\)
\(P_i\) is converted into 8 bit binary numbers
\[ E_{k_m}(P_i) = C_i \]
where \(m = 1\) to \(j\).

(iii) Algorithm for decryption

- Find the 1’s complement of receiving ciphertext \(C_i\)
- Using the equation \(P_i = D_{k_m}(C_i)\), where \(m = 1\) to \(j\) for decryption
  \(D_{k_m}(C_i)\) is bit wise XORing on ciphertext \(C_i\) with single key \(k_m\)
  \(D(C_i) = \text{Decryption of the ciphertext } C_i\)
- Plaintext \(P_i\) is converted into ASCII(\(P_i\)) with respect to its decimal value.
  Hence,
  \[ P_1 = D_{k_1}(C_1) \]
  \(P_1\) is converted into ASCII(\(P_1\)) with respect to its decimal value.
  \[ P_2 = D_{k_2}(C_2) \]
  \(P_2\) is converted into ASCII(\(P_2\)) with respect to its decimal value.
  \(P_i = D_{k_m}(C_i)\)
  \(P_i\) is converted into ASCII(\(P_i\)) with respect to its decimal value.
  where \(m = 1\) to \(j\).
- Then character \(i = \text{ASCII}(P_i)\).

2.4 ElGamal Cryptosystem

The algorithm of ElGamal Cryptosystem, as discussed in detail in, is a process the same in nature to the Diffie-Hellman key agreement protocol. The three steps involved in the process can be described as:

(i) Algorithm for key generation

- Generate a large random prime number \(p\).
- Generate a random multiplicative generator element \(g\), such that \(g < p\).
- Generate a random number \(x\), such that \(x < p-2\).
- Calculate the public key \(y\) using the equation \(y = g^x \mod p\)
- Publish \((p, g, y)\) publicly and retain \((x)\) privately.
(ii) Algorithm for encryption
- To encrypt the message \( m \), the sender first chooses a random number \( k \), such that \( \gcd (k, p-1) = 1 \), and \( 1 < k < p-2 \).
- Represent the plaintext as an integer \( m \), such that \( 0 \leq m \leq p-1 \).
- Compute the value of \( r \) and \( s \) using the equation
  \[
  r = g^k \pmod{p} \\
  s = (y^k \pmod{p})(m \pmod{(p-1)})
  \]

(iii) Algorithm for decryption
- To decrypt the ciphertext, the receiver computes the ratio of \( s \) and \( r^x \pmod{p} \), or simply
  \[
  m = \frac{s}{r^x \pmod{p}}
  \]

2.5 RSA Cryptosystem
Rivest-Shamir-Adleman (RSA) Algorithm, created and developed by Ron Rivest, Adi Shamir and Leonard Adleman at MIT in 1978, is the best known public key cryptosystem. The core of RSA has withstood each assault from the best cryptographic minds and personalities. The power of the algorithm, the nonappearance of careful evidence in any case, gives a sagacity of security. Explicitly, the foundation of security of RSA is the difficulty of factoring large prime numbers. The key pair is derived from a vast number \( n \), which is the product of two huge random and distinct prime numbers \( p \) and \( q \), chosen according to special rules; these primes may be at least 100 digits in length each, yielding an \( n \) with roughly twice as many digits as the prime factors. The strength and quality of RSA algorithm is further talked about in.

The process of RSA algorithm can be described as follows:
(i) Algorithm for the generation of public/private key pair
First, key generation is done before encrypting the data.
- Select two large random and distinct prime numbers \( p \) and \( q \).
- Calculate the modulus \( n \), where \( n = pq \) and Euler’s totient function \( \phi(n) \), where \( \phi(n) = (p-1)(q-1) \).
- Generate a prime number \( e \), such that \( 1 < e < \phi \) and relatively prime to \( \phi(n) \).
- Calculate the unique integer \( d \), such that \( 1 < d < \phi \) where \( ed = 1 \pmod{\phi(n)} \).
- Publish public key \((n, e)\) and keep private key \( d \).

(ii) Algorithm for encryption
- Represent the message as an integer \( m \in \{0, \ldots, n-1\} \)
- Compute \( c = m^e \pmod{n} \).

(iii) Algorithm for decryption
- Decryption, which is the method of converting the ciphertext back to the original message includes the use of the private key \( d \). Compute \( m = c^d \pmod{n} \).

2.6 Examples for Some Existing Cryptosystems
The word ENCRYPTED is encrypted using the existing symmetric and asymmetric cryptosystems, specifically, secure and fast chaos-based, elgamal and RSA cryptosystems. Similar parameters are also used to test and check the strength of security of each system. Such parameters are the following:
- \( A \) = any integer = 2
- \( X_0 \) = initial condition of chaotic function = 7
- \( j \) = number of keys = 4
- keys, \( k_1 = 126, k_2 = 68, k_3 = 216, k_4 = 144 \)
- two huge prime numbers, \( p = 251 \) and \( q = 229 \)
- \( e = 41 \)
- \( d = 5561 \)
- private key \( x = 74 \)
- \( k = 19 \)

Table 1, clearly seen that in utilizing different numbers of keys \( j \), various ciphertext for the same plaintext is produced, known as the avalanche effect. Consequently, it was manifested that by using various keys, the security and invulnerability was enhanced, and was not required to utilize similar key to encrypt repeated letters of the plaintext.

In order to make confusions for intruders trying to access the encrypted messages intentionally, the proponents adapted the avalanche effect.

The utilization of the remainder when a vast number is divided by a prime number is the extraordinary feature of the ElGamal algorithm, as shown in Table 2, due to the boundless number of combinations of factors, thus, making it hard to identify which of them yields that remainder. Another upside of this cryptosystem is that the foundation of its security is dependent on discovering \( x \) which is unknowable because it is an unsolvable conjuncture in mathematics, known as the discrete
logarithm problem, because it would take a long period of time to discover all the possible solutions. However, the downside of using ElGamal is that the generated ciphertext in encryption is a pair, which means, the yielded plaintext is doubled. With these, the proponents adapted only the key generation of ElGamal.

With the obtained results of encrypting the word ENCRYPTED, as shown in Table 3, it can be observed that the ciphertext generated in RSA algorithm is much larger than ElGamal. This is the advantage of using RSA; the security depends on the problem of factoring huge numbers. This characteristic of RSA was adapted with the proposed method of the authors.

Table 1. Sensitivity of Number of Keys $j$ of Secure and Fast Chaos based Cryptosystem

<table>
<thead>
<tr>
<th>Message</th>
<th>No. of Keys $j$</th>
<th>$X_{n+1}$ (decimal form)</th>
<th>Keys $k_j$ (decimal form)</th>
<th>Ciphertext (decimal form)</th>
<th>Ciphertext (ASCII equivalent)</th>
</tr>
</thead>
<tbody>
<tr>
<td>ENCRYPTED</td>
<td>1</td>
<td>84</td>
<td>126</td>
<td>196</td>
<td>Å</td>
</tr>
<tr>
<td></td>
<td></td>
<td>84</td>
<td>126</td>
<td>207</td>
<td>Í</td>
</tr>
<tr>
<td></td>
<td></td>
<td>84</td>
<td>126</td>
<td>194</td>
<td>Å</td>
</tr>
<tr>
<td></td>
<td></td>
<td>84</td>
<td>126</td>
<td>211</td>
<td>Õ</td>
</tr>
<tr>
<td></td>
<td></td>
<td>84</td>
<td>126</td>
<td>216</td>
<td>Ø</td>
</tr>
<tr>
<td></td>
<td></td>
<td>84</td>
<td>126</td>
<td>209</td>
<td>N</td>
</tr>
<tr>
<td></td>
<td></td>
<td>84</td>
<td>126</td>
<td>213</td>
<td>Ò</td>
</tr>
<tr>
<td></td>
<td></td>
<td>84</td>
<td>126</td>
<td>196</td>
<td>Ä</td>
</tr>
<tr>
<td></td>
<td></td>
<td>84</td>
<td>126</td>
<td>197</td>
<td>Å</td>
</tr>
<tr>
<td>ENCRYPTED</td>
<td>2</td>
<td>84</td>
<td>126</td>
<td>196</td>
<td>Å</td>
</tr>
<tr>
<td></td>
<td></td>
<td>120</td>
<td>68</td>
<td>245</td>
<td>Ò</td>
</tr>
<tr>
<td></td>
<td></td>
<td>84</td>
<td>126</td>
<td>194</td>
<td>Å</td>
</tr>
<tr>
<td></td>
<td></td>
<td>120</td>
<td>68</td>
<td>233</td>
<td>É</td>
</tr>
<tr>
<td></td>
<td></td>
<td>84</td>
<td>126</td>
<td>216</td>
<td>Ø</td>
</tr>
<tr>
<td></td>
<td></td>
<td>120</td>
<td>68</td>
<td>235</td>
<td>É</td>
</tr>
<tr>
<td></td>
<td></td>
<td>84</td>
<td>126</td>
<td>213</td>
<td>Ò</td>
</tr>
<tr>
<td></td>
<td></td>
<td>120</td>
<td>68</td>
<td>254</td>
<td>Þ</td>
</tr>
<tr>
<td></td>
<td></td>
<td>84</td>
<td>126</td>
<td>197</td>
<td>Å</td>
</tr>
<tr>
<td>ENCRYPTED</td>
<td>3</td>
<td>84</td>
<td>126</td>
<td>196</td>
<td>Å</td>
</tr>
<tr>
<td></td>
<td></td>
<td>120</td>
<td>68</td>
<td>245</td>
<td>Ò</td>
</tr>
<tr>
<td></td>
<td></td>
<td>144</td>
<td>216</td>
<td>100</td>
<td>d</td>
</tr>
<tr>
<td></td>
<td></td>
<td>84</td>
<td>126</td>
<td>211</td>
<td>Ò</td>
</tr>
<tr>
<td></td>
<td></td>
<td>120</td>
<td>68</td>
<td>226</td>
<td>Å</td>
</tr>
<tr>
<td></td>
<td></td>
<td>144</td>
<td>216</td>
<td>119</td>
<td>W</td>
</tr>
<tr>
<td></td>
<td></td>
<td>84</td>
<td>126</td>
<td>213</td>
<td>Ò</td>
</tr>
<tr>
<td></td>
<td></td>
<td>120</td>
<td>68</td>
<td>254</td>
<td>Þ</td>
</tr>
<tr>
<td></td>
<td></td>
<td>144</td>
<td>216</td>
<td>99</td>
<td>c</td>
</tr>
<tr>
<td>ENCRYPTED</td>
<td>4</td>
<td>84</td>
<td>126</td>
<td>196</td>
<td>Å</td>
</tr>
<tr>
<td></td>
<td></td>
<td>120</td>
<td>68</td>
<td>245</td>
<td>Ò</td>
</tr>
<tr>
<td></td>
<td></td>
<td>144</td>
<td>216</td>
<td>100</td>
<td>D</td>
</tr>
<tr>
<td></td>
<td></td>
<td>224</td>
<td>144</td>
<td>61</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>84</td>
<td>128</td>
<td>216</td>
<td>Ø</td>
</tr>
<tr>
<td></td>
<td></td>
<td>120</td>
<td>68</td>
<td>23</td>
<td>Ê</td>
</tr>
<tr>
<td></td>
<td></td>
<td>144</td>
<td>216</td>
<td>115</td>
<td>S</td>
</tr>
<tr>
<td></td>
<td></td>
<td>224</td>
<td>144</td>
<td>42</td>
<td>*</td>
</tr>
<tr>
<td></td>
<td></td>
<td>84</td>
<td>128</td>
<td>58</td>
<td>;</td>
</tr>
</tbody>
</table>
### Table 2. Encryption Using ElGamal Cryptosystem

<table>
<thead>
<tr>
<th>Message</th>
<th>ASCII Decimal Equivalent</th>
<th>r</th>
<th>$s_i$</th>
<th>Ciphertext $\delta = {r, s}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>E</td>
<td>69</td>
<td>187</td>
<td>3519</td>
<td>${187, 3519}$</td>
</tr>
<tr>
<td>N</td>
<td>78</td>
<td>187</td>
<td>3978</td>
<td>${187, 3978}$</td>
</tr>
<tr>
<td>C</td>
<td>67</td>
<td>187</td>
<td>3417</td>
<td>${187, 3417}$</td>
</tr>
<tr>
<td>R</td>
<td>82</td>
<td>187</td>
<td>4182</td>
<td>${187, 4182}$</td>
</tr>
<tr>
<td>Y</td>
<td>89</td>
<td>187</td>
<td>4539</td>
<td>${187, 4539}$</td>
</tr>
<tr>
<td>P</td>
<td>80</td>
<td>187</td>
<td>4080</td>
<td>${187, 4080}$</td>
</tr>
<tr>
<td>T</td>
<td>84</td>
<td>187</td>
<td>4284</td>
<td>${187, 4284}$</td>
</tr>
<tr>
<td>E</td>
<td>69</td>
<td>187</td>
<td>3519</td>
<td>${187, 3519}$</td>
</tr>
<tr>
<td>D</td>
<td>68</td>
<td>187</td>
<td>3468</td>
<td>${187, 3468}$</td>
</tr>
</tbody>
</table>

### Table 3. Encryption Using RSA Cryptosystem

<table>
<thead>
<tr>
<th>Message</th>
<th>ASCII Decimal Equivalent</th>
<th>$c_i$</th>
</tr>
</thead>
<tbody>
<tr>
<td>E</td>
<td>69</td>
<td>1380</td>
</tr>
<tr>
<td>N</td>
<td>78</td>
<td>55938</td>
</tr>
<tr>
<td>C</td>
<td>67</td>
<td>25121</td>
</tr>
<tr>
<td>R</td>
<td>82</td>
<td>10202</td>
</tr>
<tr>
<td>Y</td>
<td>89</td>
<td>26811</td>
</tr>
<tr>
<td>P</td>
<td>80</td>
<td>27309</td>
</tr>
<tr>
<td>T</td>
<td>84</td>
<td>37733</td>
</tr>
<tr>
<td>E</td>
<td>69</td>
<td>1380</td>
</tr>
<tr>
<td>D</td>
<td>68</td>
<td>37342</td>
</tr>
</tbody>
</table>

### 3. Proposed Work

The proposed methodology is a hybrid combination of four existing cryptosystems, specifically, Chaos-Based, AES, RSA and ElGamal. The key generation of ElGamal and Chaos based avalanche effect will be utilized to have strong keys at the same time, have a repetitive multiple numbers of keys. The key generation of RSA will also be used. The block diagram of the proposed methodology is presented in Figure 3. The algorithm of secure and Fast Chaos-Based and RSA, together with AES’s Sub Byte Transformation, solely comprises the encryption and decryption process of the proposed methodology. The keys used in the proposed method are tabulated in Table 4.

(i) Scheme of the Key Generation:

The outline for the key generation of the proposed methodology is a combination of ElGamal’s, RSA’s and Khare, Shukla and Silakari’s Secure and Fast Chaos-Based. These are combined to have complex and unpredictable key or keys. Through this system, it is essential for the receiver Bob to publish at least one public key and to keep the private key. He also needs to share at least one secret keys with the sender Alice.

### Table 4. The Proposed Algorithm Methodology

- **Public keys:**
  - $n$ (product of two primes $p$ and $q$ (secret integers))
  - $e$ (encryption key), such that $\text{gcd}(e, \phi(n)) = 1$, and $e < p$

- **Private key:**
  - $d$ (decryption key), such that $d = e^{\phi(n)}$ (mod $\phi(n)$)

- **Shared secret key:**
  - $k_i$, for $i$ is the mutually agreed number of keys

**Symbols:**
- $\oplus$ XOR operation
- ![Symbol](Arithmetic Addition)
- ![Symbol](Arithmetic Multiplication)

Bob will be required to follow the steps to generate the keys:

- **Generation of $p$, $q$ and $g$**
  - It must be taken into account that the exchange of messages amongst Alice and Bob can be any combination of characters, which incorporates English alphabet letters, numbers, symbols and other special and unique characters that can be found in the ASCII table. Since the extended ASCII table has 255 characters, the value of each character of the message of Alice, when converted to its equivalent integer, must be less than 256, hence message $m_j < 256$, for $j$ is the number of characters of the message.
  - Bob will first have to generate two random large and different prime numbers $p$ and $q$, such that $m_j < q < p$.
  - Second, Bob needs to generate a random multiplicative element $g_i$, such that $g_i < q$. for $i$ is the number of keys mutually agreed with Alice.

- **Computation of $y_i$**
  - Next, Bob will calculate the value of $y_i$ using the equation $y_i = g_i^{\chi_i} \pmod{p}$. The first two steps are adapted in ElGamal’s key generation. Through this, the strength of the secret keys generated by Bob will be unpredictable and complex. In addition, publishing the public keys using such algorithm will make it infeasible to intrude
Figure 3. The Proposed Hybrid Encryption Block Diagram.
because of the difficulty of solving discrete logarithm with large prime numbers. The keys are further strengthened with the avalanche effect of Secure and Fast Chaos-Based.

- Generation of the shared secret keys $k_i$

  Afterwards, Bob will convert $y_i$ to its binary form and take the first 8 bits, from right to left. Then get its corresponding gray code, denoted as $k_i$. Applying gray code makes the shared secret keys $k_i$ independent with one another. The values of $k_i$ will be shared by Bob secretly with Alice.

- Generation of $n$ and $\phi(n)$

  Bob will next calculate the value of modulus $n$, where $n$ is the product of the prime numbers $p$ and $q$. Bob will also need to calculate the value of the Euler's totient function $\phi(n)$, where $\phi(n)$ is the product of $p$ and $q$, each subtracted by one before multiplying, hence $n = (p-1)(q-1)$.

- Generation of $e$ and $d$

  Bob will then select a value for $e$. However, it must be considered that $e$ and $\phi(n)$ are coprime, that is gcd $(e, \phi(n)) = 1$, and $e < p$. Lastly, Bob will calculate the value of his private key $d$ by getting the multiplicative inverse of $e$ using Euclidean algorithm $d = e^{-1} \mod \phi(n)$ or $ed = 1 \mod \phi(n)$. The complete instruction on how to get the value of $d$ can be found in [16].

- Publication of $(n, e)$

  The public keys $(n, e)$ generated by Bob should be published so that the sender Alice will have the capability to send an encrypted message, known as ciphertext.

(ii) Scheme of Encryption:

In order to encrypt the message $m_j$ to Bob, Alice must first obtain his public keys $(n, e)$, together with the shared secret keys generated by him.

- Representation of the message $m_j$

  Alice must first represent her message $m_j$ in its equivalent ASCII binary form. Each of the characters of Alice's message will be encrypted one by one.

- Performing the XOR operation

  Afterwards, Alice will perform the XOR operation. She will XOR each bit of the characters of her message $m_j$ with each bit of the corresponding secret key $k_i$.

- Getting the 1's complement

  Alice will next find the 1's complement of $m_j$ and will be denoted as $w_j$.

- Mapping with Rijndael S-Box

  Alice will then map the equivalence of each $w_j$ with Rijndael S-Box. To do this, she will convert the 8-bit binary to hexadecimal, locate the equivalence using the S-Box, and convert it to decimal. This will be denoted as $s_j$.

- Encrypting the plaintext

  Thereafter, Alice will encrypt $s_j$ to $c_j$. To do this, she will calculate for each of the $s_j$ using:

  $$c_j = s_j^d \mod n.$$

(iii) Scheme of Decryption:

After Bob received the encrypted message from Alice, he will need to do the following steps:

- Decrypting the ciphertext

  Bob will first decrypt $s_j$ from ciphertext $c_j$ by using the formula:

  $$s_j = c_j^e \mod n.$$

- Mapping with Rijndael Inverse S-Box

  Bob will next map the equivalence of each $s_j$ with Rijndael Inverse S-Box. To do this, he will convert the decimal to hexadecimal, locate the equivalence using the inverse S-Box, and convert it to binary. The result be denoted as $w_j$.

- Getting the 1's complement

  Subsequently, Bob will find the 1's complement of $s_j$ and will be denoted as $w_j'$.

- Performing the XOR operation

  Bob will then perform the XOR operation. He will XOR each bit of the characters of $w_j$ with each bit of the corresponding secret key $k_i$.

- Getting the ASCII equivalence

  Bob will finally get the ASCII equivalence of each 8 bit binary. After combining all of the $w_j$, Bob can now read the message sent by Alice.

(iv) Algorithm for key generation

- Generate two random large and dissimilar prime numbers $p$ and $q$, such that $255 < q < p$.

- Generate a random multiplicative element $g_i$, such that $g_i < q$, for $i$ is the mutually agreed number of keys.

- Compute the value of $y_i$ using:

  $$y_i = g_i^{y_i} \mod p.$$  

- Convert $y_i$ to its binary form and take the first 8 bits, from left to right.

- Apply gray code in each $y_i$ and denote this secret key as $k_i$.

- Compute for the modulus $n$, where $n = pq$, and Euler’s totient function $\phi(n)$, where $\phi(n) = (p-1)(q-1)$.

- Select the value of the public key $e$, such that the greatest common factor of $e$ and $\phi(n)$ is 1, and $e < p$.

- Calculate the private key $d$ by getting the multiplicative inverse of $e$ and $\phi(n)$.
inverse of $e$ using Euclidean algorithm $d = e^{-1}(\mod \phi(n))$ or $ed = 1 \mod \phi(n)$. Hence, the public keys are $n$ and $e$, the private key is $d$ and the shared secret key is at least one $k_i$.

(v) Algorithm for encryption
- Each of the characters of the message $m_j$ must be represented first to its ASCII equivalence expressed in binary form.
- XOR the 8 bit binary with the corresponding secret key $k_i$.
- Get the 1’s complement of the result and denote it as $w_j$.
- Convert each $w_j$ to hexadecimal and map the equivalence of each $w_j$ with Rijndael S-Box (as presented in Figure 4).
- Convert equivalence to decimal and denote it as $s_j$.
- Each $s_j$ will be encrypted finally to $c_j$ using the equation $c_j = s_j^d(\mod n)$.

(vi) Algorithm for decryption
- Decrypt each $s_j$ from $c_j$ using the formula $s_j = c_j^d(\mod n)$.
- Convert each $s_j$ to hexadecimal and map the equivalence of each $s_j$ with Rijndael Inverse S-Box (as presented in Figure 5). Then convert the results to 8 bit binary and denote it as $w_j$.
- Get the 1’s complement of each $w_j$ and denote it as $w_j'$.
- XOR the 8 bit binary with the corresponding secret key $k_i$. Denote the result as $m_i$.
- Get the ASCII equivalence of each 8 bit binary.

The proposed methodology is summarized in Figure 6. The message is encrypted by the sender with the receiver’s published public key, together with the at least one shared secret key, and is decrypted by the receiver with his/her private and shared secret keys.
Example Results of the Proposed Methodology

(i) Key Generation

In generating $p$ and $q$, the condition is $255 < q < p$, therefore, $p=251$ and $q=229$.

Let $i = 4$, which means that 4 keys will be used: random multiplicative element $g_i$ such that $g_i < q$

$g_1=126$, $g_2=68$, $g_3=216$, $g_4=144$

To get the value of $y_i$, use the formula $y_i = g_i^q \mod p$

$y_1 = g_1^q \mod p = 126^{229} \mod 251 = 47$
$y_2 = g_2^q \mod p = 68^{229} \mod 251 = 110$
$y_3 = g_3^q \mod p = 216^{229} \mod 251 = 186$
$y_4 = g_4^q \mod p = 144^{229} \mod 251 = 106$

Since there are four $g$'s there will also be 4 $y$'s

The keys ($k_i$) are generated by getting the binary form of $y_i$ and then applying a gray code.

$y_1 = 00101111$
$y_2 = 01101110$
$y_3 = 10111010$
$y_4 = 01101010$

Gray codes of $y_i$:

$k_1 = 00111000$
$k_2 = 01011101$

(ii) Encryption Scheme

Message = ENCRYPTED

First, convert $m_j$ to its equivalent ASCII binary form as shown in Table 5.

<table>
<thead>
<tr>
<th>Letter</th>
<th>ASCII</th>
<th>Binary Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>E</td>
<td>69</td>
<td>01000101</td>
</tr>
<tr>
<td>N</td>
<td>78</td>
<td>01001110</td>
</tr>
<tr>
<td>C</td>
<td>67</td>
<td>01000011</td>
</tr>
<tr>
<td>R</td>
<td>82</td>
<td>01010010</td>
</tr>
<tr>
<td>Y</td>
<td>89</td>
<td>01011001</td>
</tr>
<tr>
<td>P</td>
<td>80</td>
<td>01010000</td>
</tr>
<tr>
<td>T</td>
<td>84</td>
<td>01010100</td>
</tr>
<tr>
<td>E</td>
<td>69</td>
<td>01000101</td>
</tr>
<tr>
<td>D</td>
<td>68</td>
<td>01000100</td>
</tr>
</tbody>
</table>

Then, these binary forms of each character of the message $m_j$ are XOR with the corresponding secret key $k_i$.

For $m_1 = E$:

$m_1 \text{ XOR } k_1$:

$m_1 = 01111011$

For $m_2 = N$:
m_j XOR k_j:

For m_1 = C:
  m_1 XOR k_1:
    m_1 = 00010111

For m_2 = R:
  m_2 XOR k_1:
    m_2 = 00001101

For m_3 = Y:
  m_3 XOR k_2:
    m_3 = 10100100

For m_4 = P:
  m_4 XOR k_3:
    m_4 = 00001001

For m_5 = T:
  m_5 XOR k_3:
    m_5 = 10110011

For m_6 = E:
  m_6 XOR k_4:
    m_6 = 00011010

For m_7 = D:
  m_7 XOR k_1:
    m_7 = 01100001

These m_j will be converted into its 1's complement equivalent, w_j:

  m_1 = 01111101   w_1 = 10000010
  m_2 = 00010111   w_2 = 11101000
  m_3 = 10100100   w_3 = 01011011
  m_4 = 00001101   w_4 = 11110010
  m_5 = 01100001   w_5 = 10011110
  m_6 = 00001001   w_6 = 11110110
  m_7 = 10110011   w_7 = 01001100
  m_8 = 00001101   w_8 = 11100101
  m_9 = 01111100   w_9 = 10000011

The 1's complement of m_j, w_j is then converted into its hexadecimal equivalent.

  w_1 = 10000010 = 82
  w_2 = 11101000 = E8
  w_3 = 01011011 = 5B
  w_4 = 11110010 = F2
  w_5 = 10011110 = 9E
  w_6 = 11110110 = F6
  w_7 = 01001100 = 4C
  w_8 = 11100101 = E5
  w_9 = 10000011 = 83

Locate the hexadecimal form into its equivalence in the AES S-Box in Figure 4, and convert to its decimal form denoted as s_j. The results were shown in Table 6.

<table>
<thead>
<tr>
<th>w_j (hexadecimal form)</th>
<th>S-box Equivalent (hexadecimal form)</th>
<th>s_j (decimal form)</th>
</tr>
</thead>
<tbody>
<tr>
<td>82</td>
<td>13</td>
<td>19</td>
</tr>
<tr>
<td>E8</td>
<td>9B</td>
<td>155</td>
</tr>
<tr>
<td>5B</td>
<td>39</td>
<td>57</td>
</tr>
<tr>
<td>F2</td>
<td>89</td>
<td>137</td>
</tr>
<tr>
<td>9E</td>
<td>0B</td>
<td>11</td>
</tr>
<tr>
<td>F6</td>
<td>42</td>
<td>66</td>
</tr>
<tr>
<td>4C</td>
<td>29</td>
<td>41</td>
</tr>
<tr>
<td>E5</td>
<td>D9</td>
<td>217</td>
</tr>
<tr>
<td>83</td>
<td>EC</td>
<td>236</td>
</tr>
</tbody>
</table>

To get the encrypted message c_j, use the formula c_j = s_j^e (mod n).

For s_1:
  c_1 = s_1^e (mod n)
  c_1 = 19^{41} mod 57 479
  c_1 = 24 171

For s_2:
  c_2 = s_2^e (mod n)
  c_2 = 155^{41} mod 57 479
  c_2 = 26 905

For s_3:
  c_3 = s_3^e (mod n)
  c_3 = 50 083

For s_4:
  c_4 = s_4^e (mod n)
  c_4 = 137^{41} mod 57 479
  c_4 = 22 928

For s_5:
  c_5 = s_5^e (mod n)
  c_5 = 11^{41} mod 57 479
  c_5 = 42 093

For s_6:
  c_6 = s_6^e (mod n)
  c_6 = 66^{41} mod 57 479
  c_6 = 37 222

For s_7:
  c_7 = s_7^e (mod n)
  c_7 = 41^{41} mod 57 479
  c_7 = 26 720

For s_8:
  c_8 = s_8^e (mod n)
  c_8 = 217^{41} mod 57 479


$c_8 = 32\ 447$
For $s_8$:
$c_8 = s_8^d (\text{mod } n)$
c_8 = 236^{5561} \text{mod } 57\ 479
$c_9 = 3\ 502$

Encrypted Message (Ciphertext):
$= 24\ 174, 26\ 905, 50\ 083, 22\ 928, 42\ 093, 37\ 222,$
$26\ 720, 32\ 447, 3\ 502.$

(iii) Decryption Scheme
The encrypted message, known as ciphertext, will be decrypted using the formula $s_j = c_j^d (\text{mod } n)$.

$s_1 = c_1^d (\text{mod } n)$
$s_1 = 24\ 174^{5561} \text{mod } 57\ 479$
$s_1 = 19$

$s_2 = c_2^d (\text{mod } n)$
$s_2 = 26\ 905^{5561} \text{mod } 57\ 479$
$s_2 = 155$

$s_3 = c_3^d (\text{mod } n)$
$s_3 = 50\ 083^{5561} \text{mod } 57\ 479$
$s_3 = 57$

$s_4 = c_4^d (\text{mod } n)$
$s_4 = 22\ 928^{5561} \text{mod } 57\ 479$
$s_4 = 137$

$s_5 = c_5^d (\text{mod } n)$
$s_5 = 42\ 093^{5561} \text{mod } 57\ 479$
$s_5 = 11$

$s_6 = c_6^d (\text{mod } n)$
$s_6 = 37\ 222^{5561} \text{mod } 57\ 479$
$s_6 = 66$

$s_7 = c_7^d (\text{mod } n)$
$s_7 = 26\ 720^{5561} \text{mod } 57\ 479$
$s_7 = 41$

$s_8 = c_8^d (\text{mod } n)$
$s_8 = 32\ 447^{5561} \text{mod } 57\ 479$
$s_8 = 217$

$s_9 = c_9^d (\text{mod } n)$
$s_9 = 3\ 502^{5561} \text{mod } 57\ 479$
$s_9 = 236$

Then, each $s_j$ is converted to its hexadecimal equivalent

and is located in the AES Inverse S-box. The results were shown in Table 7.

<table>
<thead>
<tr>
<th>$s_j$ (decimal form)</th>
<th>hexadecimal form</th>
<th>Inverse S-box equivalent (hexadecimal form)</th>
</tr>
</thead>
<tbody>
<tr>
<td>19</td>
<td>13</td>
<td>82</td>
</tr>
<tr>
<td>155</td>
<td>9B</td>
<td>E8</td>
</tr>
<tr>
<td>57</td>
<td>39</td>
<td>5B</td>
</tr>
<tr>
<td>137</td>
<td>89</td>
<td>F2</td>
</tr>
<tr>
<td>11</td>
<td>0B</td>
<td>9E</td>
</tr>
<tr>
<td>66</td>
<td>42</td>
<td>F6</td>
</tr>
<tr>
<td>41</td>
<td>29</td>
<td>4C</td>
</tr>
<tr>
<td>217</td>
<td>D9</td>
<td>E5</td>
</tr>
<tr>
<td>236</td>
<td>EC</td>
<td>83</td>
</tr>
</tbody>
</table>

Convert AES Inverse S-box equivalence into its 8 bit binary form, $w_j$:
$w_1 = 82 = 10000010$
$w_2 = E8 = 11101000$
$w_3 = 5B = 01011011$
$w_4 = F2 = 11110010$
$w_5 = 9E = 10011110$
$w_6 = F6 = 11110110$
$w_7 = 4C = 01001100$
$w_8 = E5 = 11100101$
$w_9 = 83 = 10000011$

The 8-bit binary form will be converted into its 1’s complement equivalent.
$w_1' = 10000010 = 01111101$
$w_2' = 11101000 = 00010111$
$w_3' = 01011011 = 10100100$
$w_4' = 11110010 = 00001101$
$w_5' = 10011110 = 01100001$
$w_6' = 11110110 = 00001001$
$w_7' = 01001100 = 10110011$
$w_8' = 11100101 = 00011010$
$w_9' = 10000011 = 01111100$

Then, XOR these 8-bit binary forms with the corresponding key $k_j$, denoted as $m_j$:
$w_1' \text{ XOR } k_1 = m_1 = 01000101$
$w_2' \text{ XOR } k_2 = m_2 = 01001110$
$w_3' \text{ XOR } k_3 =$
4. Discussion

With the proposed methodology of encrypting a plaintext, it is proven that the strength of the ciphertext is enhanced and improved. As seen in the above example, it can be seen that the value of the ciphertext of the proposed method is significantly different and, in some cases, larger than that of RSA's. The invulnerability and unpredictability is also enhanced with the avalanche effect and with the use of AES S-Box.

Table 8 shows another example of analysis using the proposed method. The example message is a template wherein the entire English alphabet is present and consists of 36 characters. Two set of keys is used, the former used one key, while the latter used 5 keys. Both scenario used primes p = 157 and q = 149, e = 19, d = 7291. A similar message was encrypted utilizing a similar prime number, public keys, private key and no less than one shared keys.

5. Conclusion

As the worldwide communication of private and confidential data and information progresses and becomes part of day to day life of every individual, the necessity for data security becomes very significant. Encrypting information is one of the methods utilized to keep them invulnerable to potential threat to data authenticity, data accessibility and data privacy. Existing cryptosystems includes Secure and Fast Chaos-Based, AES, RSA and ElGamal. However, there are currently known attacks to break such algorithms. With these, the proponents were motivated to propose an enhanced hybrid algorithm of these four methods. ElGamal's key security based on
difficulty of solving discrete logarithm, Chaos-based's avalanche effect of using multiple keys, RSA's security based on the difficulty of factorization of large numbers and AES's S-Box mapping, which makes the encryption complex and unpredictable were adapted to the proposed methodology.

In this paper, the strength and power of the security and safety of the proposed hybrid was tested by the examples provided. The result shows that by using multiple keys, different ciphertext will be produced from the same plaintext. The difficulty of solving discrete logarithm for key generation will also make hard for intruders to break such encryption. In addition, the ciphertext is made more complicated and unpredictable by the use of S-Box.

6. References