ISSN 1751-8644
doi: 0000000000
www.ietdl.org

A novel hybrid cryptographic system with ElGamal scheme using OAEP padding

Animesh Hazra[1*], *Tarit Goswami*[1], *Tanurima Halder*[2]

[1,2] *Department of Computer Science and Engineering, Jalpaiguri Government Engineering College, Jalpaiguri, West Bengal, India*
* *E-mail: animesh.hazra@cse.jgec.ac.in • tarit.goswami.2@gmail.com • haldertanurima@gmail.com*

Abstract: Nowadays in data communication the prior requirement is security. Different asymmetric algorithms like RSA provides security but RSA-OAEP scheme(RSA with OAEP padding) gives an extra benefits by adding some randomness to the scheme. Similarly, another asymmetric encryption algorithm ElGamal encryption tool also provides security during transmission but the different cryptographic attacks encroach the security and making it insecure altogether. In this paper, a novel methodology has been proposed, in which the plain-text is being encrypted with ElGamal encryption technique after padding it using Optimal Asymmetric Encryption Padding(OAEP) scheme. Before encryption, padding the plaintext with OAEP padding technique using the hashing function BLAKE2b is being formulated. This method makes the encryption and decryption procedures much faster than the usual encryption/decryption using conventional ElGamal scheme. With the increase of plain-text file size, encryption or decryption time of the proposed methodology increases very slowly. The proposed scheme is secure under the chosen-plaintext attack(IND-CPA) and also solves the problem of 1-bit information leakage present in the ElGamal encryption system.

1 Introduction

Each and every user while communicating needs a secure network channel so that data communication can be done in a secured fashion and no intruder can decipher their data. For providing secure data communication, cryptography is used in wireless and wired medium where this technique converts the plain text into cipher text and cipher text into a plain text. At the sender side, plain text is converted into a cipher text known as encryption process and cipher text is converted into plain text is known as decryption process. Cryptography is classified into symmetric key cryptography and asymmetric key cryptography. In symmetric key cryptography, the same key is used by both parties. The sender uses this key and an encryption scheme to encrypt the plain text messages, similarly the receiver on the other end uses the same key and the corresponding decryption scheme to decrypt the cipher text data. In asymmetric key or public-key cryptography, there are two keys i.e. a private key and a public key.The private key is kept by the receiver and the public key is used by the sender to encrypt the plain text messages, while the receiver decrypts the cipher text by using this private key. Further some types of asymmetric cryptography techniques are given by the different researchers. Some commonly used symmetric algorithms are AES, DES and some asymmetric cryptography techniques are RSA, Diffie-Hellman,ECC(Elliptic curve cryptography)[22] and ECC signature[21], DSA(Digital Signature Algorithm)[20], ElGamal [9-10] etc. In this section the two basic concepts ElGamal encryption and OAEP Padding[25] schemes based on which the proposed algorithm depends on are discussed as follows.

1.1 ElGamal encryption system

In public-key cryptography, the ElGamal encryption system [10] is an asymmetric key-encryption algorithm based on Diffie-Hellman key exchange. The security of this scheme is based on the difficulty of finding discrete logarithms modulo a large prime. The ElGamal encryption can be defined over arbitrary cyclic group G, e.g.: multiplicative group of integers modulo n. It's security depends upon the difficulty of a problem in G related to computing discrete logarithms. The ElGamal algorithm provides an alternative to the RSA algorithm for public key encryption where security of RSA depends on the presumed difficulty of factoring large integers, on the other hand security of ElGamal algorithm [9] depends on the presumed difficulty of computing discrete logarithms in a large prime modulus. ElGamal mechanism is applied in GNU privacy guard software [15], recent versions of PGP [16] and in many other places [5].

1.2 Optimal Asymmetric Encryption Padding Scheme

The Optimal Asymmetric Encryption Padding(OAEP) algorithm [3][7] is defined as a form of feistel network that uses a pair of random functions G and H to compute the plaintext before the initiation of asymmetric encryption. While combining with any one-way trapdoor permutation f, the processing has been proven in the random oracle model, resulting in a combined scheme which does not allow any information about the plaintext to be feasibly extracted from the ciphertext(aka semantically secure) under chosen plaintext attack(IND-CPA) [11][17]. This padding scheme satisfies two goals: firstly, adding randomness to the plain-text and secondly, preventing partial decryption of ciphertext or other information.

2 Working mechanism of the ElGamal scheme

ElGamal encryption [9][10] consists of three steps i.e. the key generation and the encryption followed by decryption. Suppose, there are two parties Alice and Bob. Alice wants to communicate with Bob. Alice, the first party sends message which is to be received by Bob, the second party. Now, the above three steps are discussed below in details.

2.1 Key generation procedure

In the key generation phase Bob generates a pair of key, while generating this key we are using a cyclic group G of order q with any generator g. Let, e represents an unit element of G. Then randomly choose an integer in the range of $\{1, 2, \cdots, q-1\}$ and compute $h = g^x$. Alice publishes the public key which consists of the values (G, q, g, h) and retains x as her private key.

2.2 Encryption procedure

With the help of the public key generated by Bob, Alice encrypts the message M. Next, map the message M to an element m of group G and then choose an integer y randomly from the set $\{1, 2, \cdots, q-1\}$. Compute the shared secret values $s = h^y$, $c_1 = g^y$ and $c_2 = m \cdot s$. Alice sends the ciphertext (c_1, c_2) to Bob.

2.3 Decryption procedure

With the help of private key x, Bob decrypts the ciphertext (c_1, c_2). Compute $s = c_1^x$, and compute s^{-1}, the inverse of s in group G. Then, compute $m = c_2 \cdot s^{-1}$, which gives back the original message m. Finally, map m to the plain-text M.

3 How OAEP algorithm works

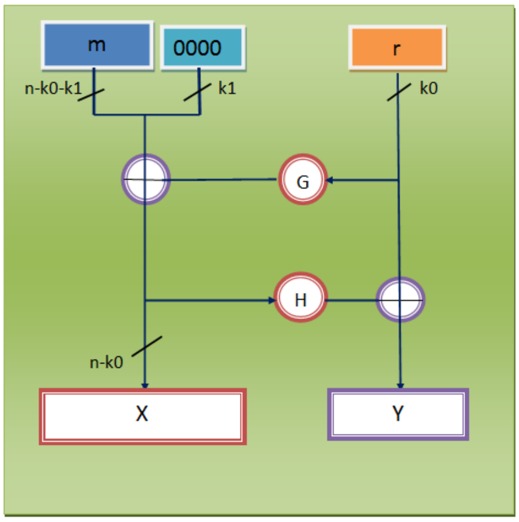

Fig. 1: Illustration of the OAEP padding scheme

In Fig.1, n is the number of plain-text message bits, k_0 and k_1 are integers fixed by the protocols, M is the plain-text message which is a $(n - k_0 - k_1)$ bit string, whereas G and H are random cryptographic hash functions.

In the process of encoding, the message is padded with k_1 zeros to be $(n - k_0)$ bits in length. Here, r is a randomly generated k_0 bit string. G expands the k_0 bits of r to $(n - k_0)$ bits. Then, compute $X = m00\cdots0 \oplus G(r)$. H reduces the $(n - k_0)$ bits of X to k_0 bits and $Y = r \oplus H(X)$. Now the output is $X||Y$, where X and Y are shown in the above diagram as the leftmost and rightmost blocks respectively.

On the other hand, while decoding this encoded message, recover the random string as $r = Y \oplus H(X)$ and followed by recovering the message as $m00\cdots0 = X \oplus G(r)$.

4 Motivation behind the proposed scheme

The ElGamal encryption scheme [9][10] uses public parameter a large prime p and a primitive element α of $\mathbb{Z}_p^*$ (the multiplicative group modulo p). Thus, $x \mapsto \alpha^x \pmod p$ is a bijective function over $[1, p)$. Ensuring that $(p-1)$ has a large prime factor which makes reversing this function (the discrete logarithm problem) hard [4][8].

Recipient Bob chooses a random secret private key $x_B \in [1, p)$, computes and publishes his public key $y_B = \alpha^{x_B} \pmod p$.

Sender Alice, wanting to encipher a secret message $m \in [1, p)$ to Bob, picks a random secret $k \in [1, p)$, computes the secret key $K = {y_B}^k \pmod p$ and computes $c_1 = \alpha^k \pmod p$ then finally computes $c_2 = Ks \cdot m \pmod p$, and sends the ciphertext (c_1, c_2) to Bob.

Recipient Bob receives the ciphertext (c_1, c_2), and deciphers it as $m = {c_1}^{p-1-x_B}\, c_2 \pmod p$. This works because $K = {c_1}^{x_B} \pmod p$.

Observe that given $y = \alpha^x \pmod p$ with $y \in [1, p)$, we can determine with certainty if x is odd or even. To know the parity of x, we compute $y^{(p-1)/2} \pmod p$ and that's 1 or $(p-1)$ when x is even or odd respectively. Expressed using the Legendre symbol for y modulo p, that's $\left(\frac{y}{p}\right) = +1$ when $y^{(p-1)/2} \pmod p = 1$ (even x), or $\left(\frac{y}{p}\right) = -1$ when $y^{(p-1)/2} \pmod p = p-1$ (odd x). This allows an adversary to win the IND-CPA game with certainty by choosing two messages m_0 and m_1 with $\left(\frac{m_0}{p}\right) = +1$ and $\left(\frac{m_1}{p}\right) = -1$. The choice of $m_1 = 1$ and $m_2 = \alpha$ will do, or it can be found by trial and error method into meaningful messages until two have different Legendre symbols [12].

Submitting m_0 and m_1 to the challenger, which picks $b \in \{0, 1\}$ at random and sets $m = m_b$, computes and reveals (c_1, c_2) as above.

The value of the parameter b can be found out as per the following table described below.

Table 1 Finding out the value of parameter b with respect to different conditions

$\left(\frac{y_B}{p}\right)$	$\left(\frac{c_1}{p}\right)$	$\left(\frac{c_2}{p}\right)$	b
-1	-1	-1	0
-1	-1	+1	1
any	+1	-1	1
any	+1	+1	0
+1	any	-1	1
+1	any	+1	0

This works because $\left(\frac{y_B}{p}\right) = -1 \iff x_B$ is odd and $\left(\frac{c_1}{p}\right) = -1 \iff k$ is odd. Since $K = \alpha^{x_B k}$ that allows to determine $\left(\frac{K}{p}\right)$, which is -1 if and only if both $\left(\frac{c_1}{p}\right) = -1$ and $\left(\frac{c_2}{p}\right) = -1$. And then $\left(\frac{c_2}{p}\right) = \left(\frac{K}{p}\right)\left(\frac{m_b}{p}\right)$ allows to conclude on b.

The motivation of using OAEP padding scheme [4][7] in order to prepare the message to form m in ElGamal encryption [9] are as follows:

(i) it is non-iterative and faster (ii) it should make the ElGamal encryption IND-CPA secure, because the partial information that may leak won't be enough to allow the adversary to undo the padding.

5 Related work

In general, there are two different types of cryptography namely symmetric key cryptography and asymmetric key cryptography. An asymmetric algorithm is defined as an algorithm where the encryption key used in encryption algorithm is not identical to the decryption key in decryption algorithm [29–30]. It is poles apart from symmetric algorithm which uses the same two keys during encryption and decryption process [31][32][33]. The asymmetric key uses two keys, public and private key during encryption and decryption respectively.

Mainly focussing on asymmetric key cryptography, there are different algorithms formulated like RSA, RSA with OAEP [3], ElGamal [9][10] etc. While studying through the encryption scheme RSA, it is observed that this scheme is vulnerable to the different attacks like IND-CPA [7] and IND-CCA-2 [2], which makes the scheme insecure one. These different attacks (from which RSA scheme was suffering) could be partially overcome by a new scheme RSA-OAEP padding. In RSA-OAEP padding [3], the security of RSA scheme is strengthened by OAEP scheme. RSA-OAEP is a public key encryption scheme which combines the encoding method Optimal Asymmetric Encryption Padding (OAEP) with RSA encryption primitive RSAEP. RSAES-OAEP takes a plaintext as input, transforms it into an encoded message via OAEP and applies RSAEP to the result which is interpreted as an integer using an RSA public key.

On the other hand, while studying the properties of another well known scheme ElGamal encryption, this scheme also suffers from the above mentioned cryptographic attacks. The different attacks like 1-bit information leakage, IND-CPA [7] are being suffered by this well known scheme ElGamal. This problem had been kept on table for discussion and tried to devise solutions so that this public key encryption scheme remains unharmed by these attacks.

Data security is very significant to mark down particularly if it is in the public domain [23–27]. The data passed on the power system [28] effortlessly so that the techniques used in the field of cryptography should protect the information. As discussed in the previous section, RSA and ElGamal both use asymmetric key techniques. The basic difference lies in the number of variables used in both the algorithms. RSA[19] has used two variables during encryption, on the other hand ElGamal has used three variables. The strength of RSA algorithm works at the highest level of difficulty in factoring the numbers into a prime factor. The variables p and q should be co-prime to each other(i.e. the greatest common factor of these two variables is 1). The public key n is numerically equal to the multiplication of two numbers kept in the variables p and q. The process of factorization in determining the value of p and q depends on the value of n. If the value of n is being factorized, then determining the value m is easy where m is the value of the message to be sent. Although the value e is known, the key calculation d is not easy because the m value is unknown. The strong points of RSA algorithm is the mechanism to defend the various attacks, specifically brute force attacks.The main reason behind this is the complexity during decryption can be evaluated by calculating the large values p and q at the time of the key pair generation[34].The result n is quite large a number that gives room to RSA algorithm to be resistant to attack. However, the magnitude of the private key that is huge will upshot in a fairly slow decryption procedure, especially for large message sizes. In concluding words about RSA algorithm, it is mainly used to encrypt small messages such as password encryption and pin number.

Unlike ElGamal algorithm, this algorithm executes the encryption process on the plaintext blocks which then gives output as the ciphertext blocks. These encrypted message blocks will be deciphered again, and the result is then integrated into the original plaintext. The certainity of the ElGamal algorithm lies in the difficulty of computing discrete logarithms on large prime modulo [35]. Decoding this logarithm problem is an arduous thing to resolve. The benefit of ElGamal algorithm is the production of keys using discrete logarithms. The main cons of this algorithm is that it needs a massive resource since the arising ciphertext is twice the length of the plaintext message and requires a processor competent of executing extensive computations for these lengthy logarithmic calculations.

Previously discussing about the RSA scheme and its improved version RSA-OAEP [3], a new combinatory algorithm has been proposed in which there is an amalgamation of straight ElGamal encryption scheme [9][10] with OAEP padding [7]. This different scheme ensures the security of ElGamal encryption from 1-bit information leakage and IND-CPA attack.

6 Proposed methodology

This section gives an insight into a novel methodology of sending the message securely over an insecure channel. Also, how a well-known encryption algorithm is modified with a known padding scheme has improved the faults of this known encryption algorithm and a detailed comparative study of this known scheme with the proposed one has been elaborated with standing arguments and graphical representations. Now, let's discuss the encryption and decryption procedure of the new methodology. In the first case, let's highlight the algorithm of encryption which is shown in Fig. 2 that has overcome the flaws discussed in the previous section.

6.1 Proposed encryption algorithm

Alice and Bob are the two parties sending a message over an insecure channel. Alice uses OAEP padding scheme to pad the plain-text message, followed by encrypting the padded message with ElGamal encryption system. The proposed procedure consists of the padding procedure followed by encrypting the output text generated by the former procedure. The Fig. **??** shows the flow of the entire operation.

Initially, Alice applies the OAEP padding scheme as explained in the Algorithm 1 and then encrypts the padded message got from Algorithm 1 with the help of Algorithm 2. Finally,Alice gets the encrypted padded message which she can send to Bob.

Algorithm 1 : Proposed Padding Scheme

Input: The original message M to be sent to Bob.
Output: Padded message.
Purpose: Padding the plaintext M before applying the Algorithm 2.

```
1.  $M \leftarrow M + t$ ;// $t$ is a string of 0s.
2.  $H \leftarrow$ blake_function(ran) ;// $ran$ is a random number.
3.  $X \leftarrow M \oplus H$;// performing XOR between $M$ and $H$.
4.  $G \leftarrow$ blake_function(X);
5.  $Y \leftarrow ran \oplus G$;
6.  **return** $X, Y$;
```

In Algorithm 2 the padded message obtained in Algorithm 1 is encrypted using the ElGamal encryption algorithm. And we finally gets the padded encrypted message. The `gen_key()` function generates the key pair after providing a g value. Then, as we do in ElGamal encryption, we compute s and q.

Algorithm 2 : Proposed Encryption Procedure

Input: $X,\ q,\ h,\ g$.
Output: $X_{Enc},\ p$.
Purpose: Encrypting the padded plaintext obtained in Algorithm 1.

```
1.  $X_{Enc} \leftarrow \phi$ ;// it will store the encrypted text of $X$.
2.  $k \leftarrow$ gen_key(g) ;// Generates the key $k$.
3.  $s \leftarrow h^k \pmod{q}$ ;
4.  $p \leftarrow g^k \pmod{q}$ ;
5.  **for** each character in padded message X **do**
6.      add $X[i]$ to $X_{Enc}$;
7.  **end for**
8.  **for** each character in $X_{Enc}$ **do**
9.      X$_{Enc} = s *$ Ascii(X_Enc); $//converttoASCIIvalue.$
10.   **end for**
11.   **return**X$_{Enc}, p$;
```

Act in accordance with the same encryption procedure for Y in lieu X.

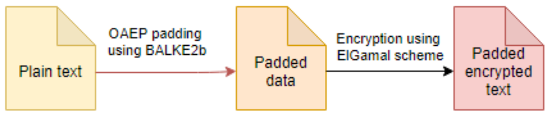

Fig. 2: Encrypting the message using the proposed scheme

6.2 *Proposed decryption algorithm*

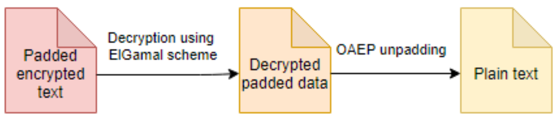

Fig. 3: Decrypting the encrypted message using the proposed scheme

When Bob gets the padded encrypted message from Alice, he needs to decrypt the message to get the padded message using Algorithm 3. Then, he needs to use the Algorithm 4 to unpad the decrypted message to get the plaintext sent by Alice. In Fig. **??**, the flow of proposed decryption procedure is illustrated.

Algorithm 3 : Proposed Decryption Procedure

Input: X_{Enc}, key.
Output: Decrypted padded message.
Purpose: Decrypting the ciphertext obtained in Algorithm 2.

```
1.  $X_{Dec} \leftarrow \phi$ ;// to store the decrypted padded text.
2.  $h \leftarrow p^{key} \pmod q$ ;// $key$ is the private key of Bob.
3.  for each character in $X_{Enc}$ do
4.      add $X_{Enc}[i]/h$ to $X_{Dec}$ ;
5.  end for
6.  return $X_{Dec}$ ;
```

After decrypting the ciphertext obtained in the Algorithm 2 into padded message, we need to unpad the padded text to get the plaintext M. Algorithm 4 helps to unpad the decrypted message and it is shown below. Again we need to use the BLAKE2b hashing function on the decrypted message.

Algorithm 4 : Proposed Unpadding Scheme

Input: X_{Dec}, Y_{Dec}
Output: Original message M.
Purpose: Unpadding the decrypted padded text obtained in Algorithm 3.

```
1.  $x \leftarrow$ blake_function($X_{Dec}$) ;// unpadding $X_{Dec}$.
2.  $r_1 \leftarrow x \oplus Y_{Dec}$;
3.  $r_1' \leftarrow$ blake_function($r_1$);
4.  $M \leftarrow X_{Dec} \oplus r_1'$;
5.  $M \leftarrow M - t$;
6.  return $M$ ;// getting back the message.
```

6.3 `BLAKE2` *cryptographic hashing function*

Introduction of an improved version of SHA-3, the cryptographic hash function BLAKE2 [6][13], helps in optimizing the speed of software. BLAKE2 comes in two main types i.e. BLAKE2b and BLAKE2s where BLAKE2b [13] is optimized for 64-bit platforms and BLAKE2s [13] for smaller architectures. On 64-bit platforms, BLAKE2 is often faster than MD5, yet it provides security similar to that of SHA-3. BLAKE2, an improved version of BLAKE have the following features: (i) faster than MD5 on 64-bit Intel platform.
(ii) direct support with no overhead of parallelism for many-times faster hashing on multicore or SIMD CPUs, tree hashing for incremental update or verification of large files.
(iii) minimal padding, which is faster and simpler to implement.

BLAKE2 [13] has more benefits than just speed, BLAKE2 uses up to 32% less RAM than BLAKE and comes with a comprehensive tree-hashing mode as well as an efficient MAC mode.

7 Results and discussion

7.1 *Results*

Implementation of the proposed scheme has been done in Python 3.6 programming environment where Intel(R) Xeon(R) 2.20 GHz CPU is used. The hashing function BLAKE2 is implemented in python 3.6 where `hashlib` library [14] is used. In Table 2, the comparison of encryption times of conventional Elgamal scheme and the proposed scheme is illustrated. Similarly, in the Table 3, the comparison of decryption times of the plain Elgamal technique with respect to the proposed technique is cited. To measure the average time for encryption and decryption procedures of a given file size, we have ran both ElGamal algorithm and our proposed algorithm 1000 times and calculated the time it takes on average to encrypt and decrypt a file of that given size.

Table 2 Comparison of encryption time between the conventional Elgamal scheme and the proposed scheme

File size	Encryption time of the conventional ElGamal scheme (in miliseconds)	**Encryption time of the proposed scheme (in miliseconds)**
5 MB	249.84636	**6.75327**
10 MB	497.54668	**7.87015**
20 MB	993.68955	**9.94992**
25 MB	1104.56678	**11.35024**
50 MB	13064.58612	**20.61920**
75 MB	20123.01765	**28.41485**
100 MB	27335.98147	**36.44008**
125 MB	33506.21784	**49.29046**
150 MB	39042.54580	**56.30092**

And here in Table 3, the result for decryption time taken by the conventional ElGamal algorithm and our proposed algorithm respectively in the same platform is shown as follows.

Table 3 Comparison of decryption time between the conventional Elgamal scheme and the proposed scheme

File size	Decryption time of the conventional ElGamal scheme (in miliseconds)	**Decryption time of the proposed scheme (in miliseconds)**
5 MB	485.01149	**1.23740**
10 MB	970.75815	**1.04869**
20 MB	1944.81566	**0.57723**
25 MB	21436.01735	**0.57858**
50 MB	25251.33328	**0.60542**
75 MB	39676.73359	**0.57995**
100 MB	52943.51559	**0.58146**
125 MB	61997.88542	**0.58898**
150 MB	77350.02754	**0.64107**

7.2 *Comparative analysis*

Here, we have analyzed the run times for a file with sizes ranging from 5 MB to 150 MB of two schemes,firstly the well known scheme ElGamal and then the proposed scheme ElGamal-OAEP.

Let's discuss about the results of runtime values of the above mentioned schemes(ElGamal and ElGamal-OAEP). In the ElGamal encryption scheme, the recorded runtime values show an abrupt increase in runtime with positive slope along with increasing input file size ranging from 5 MB to 150 MB which is shown in Fig 4.

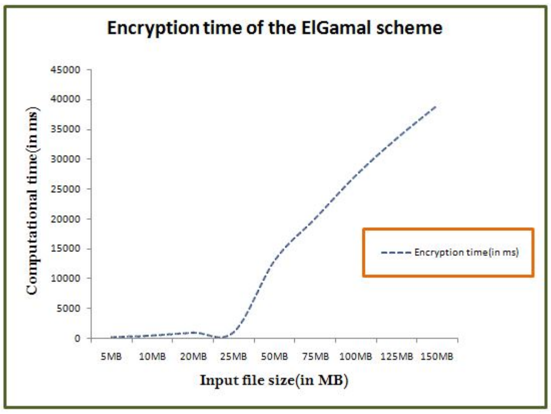

Fig. 4: Plots for encryption time curve of the conventional ElGamal scheme with respect to the increasing file size

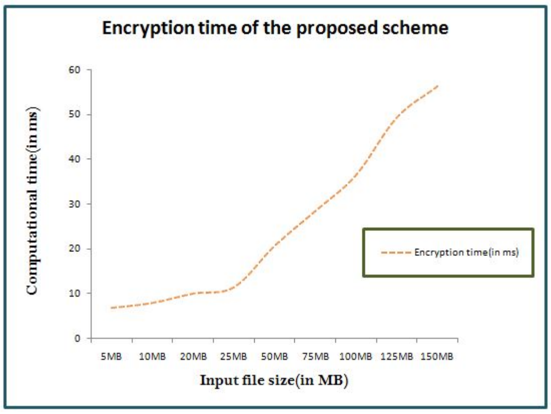

Fig. 5: Plots for encryption time curve of the **proposed scheme** with respect to the increasing file size

On the other hand, in the proposed scheme, there is a noticeable change in the runtime values along with positive slope but much less than ElGamal scheme with the same range of input file sizes from 5 MB to 150 MB which is shown in Fig 5.

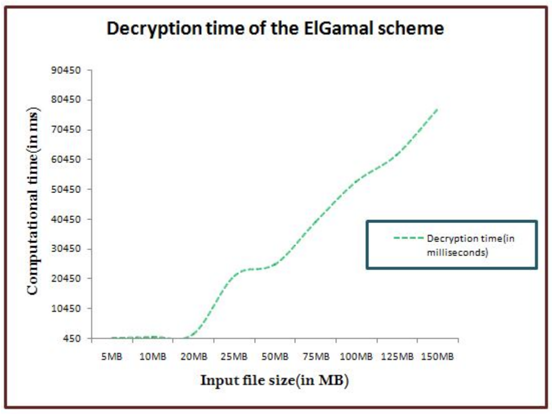

Fig. 6: Plots for decryption time curve of the conventional ElGamal scheme with respect to the increasing file size

In the similar fashion, for ElGamal decryption, the runtime values increases abruptly along with a positive increasing slope with increasing values of file sizes as demonstrated in Fig 6.

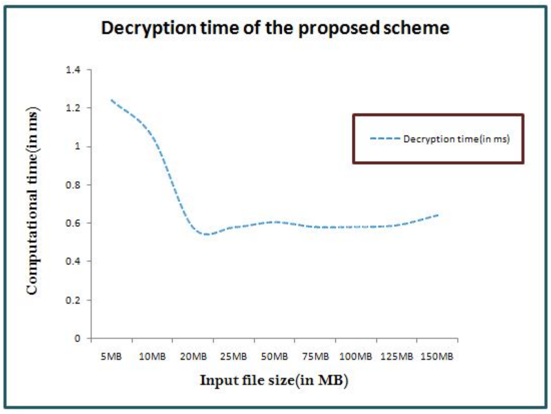

Fig. 7: Plots for decryption time curve of the **proposed scheme** with respect to the increasing file size

On the other hand, in the proposed scheme, the runtime values increases along with a positive slope with increasing values of file size but much less than the ElGamal scheme which is cited in Fig 7.

8 Security of the proposed scheme against different types of attacks

ElGamal is quite malleable and hence it's not secure under different kinds of attacks. The novelty of the scheme described here is securing ElGamal from two major cryptographic attacks i.e. padding oracle attack(1-bit information leak) and IND-CPA attack [11].

8.1 Resolution of 1-bit information leakage in ElGamal encryption using the proposed scheme

The OAEP plaintext to padded message representative transform maps to an uniformly random element of a pseudo-random subset of $[1, p-1]$. Thus,every element of the subset $[1, p-1]$ can be encrypted with ElGamal encryption mechanism.

Straight ElGamal encryption leaks only one bit of information about the plaintext. Since the nature of that bit (Legendre symbol of the plaintext) is unrelated to the pseudo-random permutation, that leakage on the OAEP message representative does not translate to an exploitable leakage for the plaintext of the composite scheme.

8.2 Providing IND-CPA security using the proposed scheme

Neither can any deterministic and conventional algorithm be secure against indistinguishability under chosen plaintext nor can it stop leaking of information.This computational insecurity(IND-CPA insecure) provided by the conventional Elgamal encryption intrigues the proposition of a new algorithm that can easily resolve this IND-CPA attack.

8.2.1 The IND-CPA game: For any probabilistic asymmetric (or public) key encryption algorithm, indistinguishability under chosen plaintext attack(IND-CPA)[18] can be described by the following game between a challenger and an adversary.Generally, for every public key encryption scheme, it will have key generation algorithm K , which will generate a key pair (Ke,Kd) along with an encryption algorithm En and a decryption algorithm De. Encryption is always revertible, but the encryption and decryption keys are different such that: $De(Kd,En(Ke, M))=M$ where M is the plaintext.
Now, let's briefly explain the algorithm of IND-CPA game and give a perception about the working mechanism of this game.

For the schemes mainly based on computational security,the adversary is a probabilistic polynomial time Turing machine and not a person.The adversary bring about two messages of equal length. The challenger chooses randomly, to encrypt one of them. Thereby, adversary has to estimate that which of the messages has been encrypted in polynomial times.The steps of this game is shown below as follows.

1. challenger: Ke and Kd are the security parameters.
2. adversary: m_0 and m_1 are the two messages of equal length selected by this Turing machine. Send m_0 and m_1 to the challenger.
3. challenger: b is a variable that randomly chooses any value between 0 and 1.
4. challenger: C is the ciphertext which is to be evaluated by the following function $En(Ke$,$m_b)$. Send C to the adversary.
5. adversary: This function performs additional operations in polynomial time including calls to the encryption oracle thus output *guess* will be produced.
6. If *guess* = b occur, then the adversary wins.

8.2.2 How the proposed scheme provides IND-CPA security: Firstly, in order to recover m_b from c the adversary algorithm has to recover the entirety of X and Y from c, where c = ElGamal-OAEP(m_b) = ElGamal($X||Y$) where Elgamal is the conventional public key encryption scheme. This is because to recover m, the adversary algorithm must be able to compute $r = Y \oplus H(X)$. If this Turing machine(adversary algorithm) can not find even one bit of X,then $H(X)$ function will give a uniform random value (H is a random oracle), and this will make the r value *totally wrong* i.e. we get nothing of the true value of r. In the same way for Y, if this adversary algorithm can not get even one bit of Y, r will be incorrect and as G is also a random oracle mapped = $X \oplus G(r)$ will again give another uniformly random value and as a result nothing gets revealed about the actual value of m.
Thus,the adversary has no better chance than guessing b and has non-negligible advantage hence making the scheme IND-CPA secure.

9 Conclusion and Future scope

The proposed method ElGamal-OAEP, is a combination of public key cryptography scheme ElGamal encryption [9][10] and OAEP padding [3][7], where the data is padded with OAEP scheme and then encrypted with ElGamal scheme. With the help of runtime values plotted into graphs, comparison can be drawn from the straight ElGamal scheme and the proposed scheme ElGamal-OAEP. These runtime graphs and results interprets how the proposed algorithm is much better and faster than the conventional ElGamal scheme. Also, the proposed amalgamated scheme is much more secured and robust compared to the famous ElGamal scheme because the new scheme ensures security from 1-bit information leakage and IND-CPA attack which is the novelty assured by the proposed scheme.

Improvement is gateway to the path of inventions. OAEP can not make ElGamal algorithm IND-CCA2 secure. Hence, people can work further on the proposed scheme ElGamal-OAEP and take it forward to make it IND-CCA2 secure.

10 Acknowledgement

We are very thankful to François Grieu, Innovatron Lab, France for helping us to find out the topic.

11 References

[1] Boneh D., Joux A., Nguyen P.Q. (2000) Why Textbook ElGamal and RSA Encryption Are Insecure. In: Okamoto T. (eds) Advances in Cryptology — ASIACRYPT 2000. ASIACRYPT 2000. Lecture Notes in Computer Science, vol. 1976, Springer, Berlin, Heidelberg.

[2] Ciphertext Indistinguishability, available at `https://wikipedia.org/wiki/Ciphertext_indistinguishability`

[3] Katz J., and Lindell Y.: 'Public-Key Cryptosystems in the Random Oracle Model', in 'Introduction to Modern Cryptography' 2007 edn., pp. 450-452.

[4] Steven D Galbraith. 'Mathematics of public key cryptography'. Cambridge University Press, 2012 edn.

[5] David B., Bogdan W. (2016), 'Cryptographic Voting - A Gentle Introduction', IACR.

[6] Aumasson JP., Neves S., Wilcox-O'Hearn Z., Winnerlein C. (2013) *BLAKE2: Simpler, Smaller, Fast as MD5*. In: Jacobson M., Locasto M., Mohassel P., Safavi-Naini R. (eds) Applied Cryptography and Network Security. ACNS 2013. Lecture Notes in Computer Science, vol 7954. Springer, Berlin, Heidelberg.

[7] Pointcheval D. (2005), 'OAEP: Optimal Asymmetric Encryption Padding'. In: van Tilborg H.C.A. (eds) Encyclopedia of Cryptography and Security. Springer, Boston, MA.

[8] Ian F. Blake and Theo Garefalakis. On the complexity of the discrete logarithm and deffie-hellman problems. J. Complex., 20(2-3):148 - 170, 2004.

[9] ElGamal T. (1985) A Public Key Cryptosystem and a Signature Scheme Based on Discrete Logarithms. In: Blakley G.R., Chaum D. (eds) Advances in Cryptology. CRYPTO 1984. Lecture Notes in Computer Science, vol 196. Springer, Berlin, Heidelberg.

[10] 'ElGamal encryption', In Wikipedia. accessed on 13 March, 2020, from `wikipedia.org/wiki/ElGamal_encryption`.

[11] 'Chosen Plaintext Attack(IND-CPA)', accessed 13 March 2020, `wikipedia.org/wiki/Chosen_plaintext_attack`.

[12] 'Legendre symbol', accessed 13 March, 2020, from `wikipedia.org/wiki/Legendre_symbol`.

[13] BLAKE2 - fast secure hashing, available at `https://blake2.net/`.

[14] Python 3 library for BLAKE2, available at `https://docs.python.org/3.6/library/hashlib.html#blake2`.

[15] GNU privacy guard software, available at `https://gnupg.org/`.

[16] PGP software, available at `https://pgptool.github.io/`.

[17] Z. Liu and Y. Pan and T. Xie, 'Breaking the hardness assumption and IND-CPA security of HQC submitted to NIST PQC project', under *IET Information Security* 2020.

[18] 'Chosen Plaintext Attack(IND-CPA)', available at `https://open.oregonstate.education/cryptography/chapter/chapter-8- security-against-chosen-plaintext-attacks/`.

[19] Chor, Benny and Goldreich, Oded, 'RSA/Rabin Least Significant Bits Are $1 - 2 \pm 1/Poly(\log N)$ Secure', 1985

[20] 'A Signature Scheme as Secure as the Diffie-Hellman Problem',Eu-Jin Goh and Stanislaw Jarecki, 2003.

[21] Amounas, Fatima Kinani, El & el hassan, el kinani. (2012). Elliptic Curve Digital Signature Algorithm Using Boolean Permutation based ECC. International Journal of Information and Network Security (IJINS). 1. 10.11591/ijins.v1i3.749.

[22] C. Varma, booktitle:*'2018 International Conference on Current Trends towards Converging Technologies (ICCTCT)'*, 'A Study of the ECC, RSA and the Diffie-Hellman Algorithms in Network Security', 2018

[23] R. Rahim et al., "Searching Process with Raita Algorithm and its Application," J. Phys. Conf. Ser., vol. 1007, no. 1, Apr. 2018.

[24] R. Meiyanti, A. Subandi, N. Fuqara, M. A. Budiman, and A. P. U. Siahaan, "The recognition of female voice based on voice registers in singing techniques in real-time using hankel transform method and macdonald function," J. Phys. Conf. Ser., vol. 978, no. 1, Mar. 2018.

[25] R. Rahim, M. Dahria, M. Syahril, and B. Anwar, “Combination of the Blowfish and Lempel-Ziv-Welch algorithms for text compression,” World Trans. Eng. Technol.Educ., vol. 15, no. 3, pp. 292–297, 2017.

[26] R. Rahim, D. Hartama, H. Nurdiyanto, A. S. Ahmar, D. Abdullah, and D. Napitupulu, “Keylogger Application to Monitoring Users Activity with Exact String Matching Algorithm,” J. Phys. Conf. Ser., vol. 954, no. 1, 2018.

[27] H. Nurdiyanto and R. Rahim, “Enhanced pixel value differencing steganography with government standard algorithm,” in 2017 3rd International Conference on Science in Information Technology (ICSITech), 2017, pp. 366–371.

[28] S. Aryza, M. Irwanto, Z. Lubis, A. P. U. Siahaan, R. Rahim, and M. Furqan, “A Novelty Design Of Minimization Of Electrical Losses In A Vector Controlled Induction Machine Drive,” IOP Conf. Ser. Mater. Sci. Eng., vol. 300, 2018.

[29] A. Putera, U. Siahaan, and R. Rahim, “Dynamic Key Matrix of Hill Cipher Using Genetic Algorithm,” Int. J. Secur. Its Appl., vol. 10, no.8, pp. 173–180, Aug. 2016.

[30] E. Kartikadarma, T. Listyorini, and R. Rahim, “An Android mobile RC4 simulation for education,” World Trans. Eng. Technol. Educ., vol. 16, no. 1, pp. 75–79, 2018.

[31] B. Oktaviana and A. P. U. Siahaan, “Three-Pass Protocol Implementation on Caesar Cipher in Classic Cryptography,” IOSR J. Comput. Eng., vol. 18, no. 4, pp. 26–29, 2016.

[32] R. Rahim et al., “Combination Base64 Algorithm and EOF Technique for Steganography,” J. Phys. Conf. Ser., vol. 1007, no. 1, Apr. 2018.

[33] D. Abdullah, R. Rahim, D. Apdilah, S. Efendi, T. Tulus, and S. Suwilo, “Prime Numbers Comparison using Sieve of Eratosthenes and Sieve of Sundaram Algorithm,” in Journal of Physics: Conference Series, vol. 978, no. 1, 2018.

[34] D. Kurnia, H. Dafitri, and A. P. U. Siahaan, “RSA 32-bit Implementation Technique,” Int. J. Recent Trends Eng. Res., vol. 3, no. 7, pp. 279–284, 2017.

[35] T. ElGamal, “A Public Key Cryptosystem and a Signature Scheme Based on Discrete Logarithms,” IEEE Trans. Inf. Theory, vol. 31, no. 4, pp. 469–472, 1985.