An Implementation of ElGamal Scheme for Laplace Transform Cryptosystem

Department of Mathematics, GIS,
GITAM (Deemed to be University), Visakhapatnam, Andhra Pradesh, INDIA
nagalakshmigangu@gmail.com
sekar.akkapeddi@gmail.com
drravi68@gmail.com

Abstract - ElGamal algorithm is public key cryptosystem and a signature scheme in the speed of the procedures for generating and verifying signatures. An implementation of ElGamal scheme for Laplace Transform Cryptosystem is proposed and it is executed using a .NET program. The time analysis is, compared with existing algorithms. The comparison reveals that the proposed cryptosystem enhances the data security and password security. The statistical tools are use for the planned scheme and accessible algorithms. They are, analyzed graphically.

Keywords: ElGamal algorithm, Encryption, Decryption, Laplace Transform, Inverse Laplace Transform, public key, privet key, statistical analysis.

1. Introduction

The ElGamal introduced public key cryptosystem and a signature scheme [1] using discrete algorithms with the help of finite fields. Schnorr [2] improved the ElGamal signature scheme in the speed of the procedures for the generator, verifying signatures and the bit length of signatures. Security of blind signatures [3] was studied in off line electronic cash system. In this, a notion of security related to the setting of electronic cash was defined. Zero knowledge proofs [4] of bi encryption are constructed for standard homomorphism encryption scheme [1]. An effective public key encryption with conjunctive key word search (PECK) scheme was constructed [5] and it is proved over a Diffie-Hellman assumption in the random oracle model. Chebyshev polynomials were studied [6] in a public – key cryptosystem which provided encryption and digital signature. ElGamal encryption method was modified [7] for two are more senders and one receiver. In the paper [8] the modified ElGamal encryption scheme is secure against an adaptive chosen ciphertext attacks in the standard model.


The present work is proposed on implementation of ElGamal scheme for Laplace Transform cryptosystem to provide high level of data security and password security. This work is completely depended on the polynomial expansion upto infinite series. First step of this method is to apply Laplace Transform on the expansion with message and second step is to implement of ElGamal algorithm. It gives encipher values which differ from the plaintext values.

2. ElGamal Cryptosystem

The ElGamal algorithm [7] has give following steps, Creation of key, Encipher process and Decipher process.

2.1 Reflection of key generation

A large prime order P is selected with generator ‘e1’. Where P is a cyclic group from the set {1,2,…,P-1}. A random number ‘ d’ selected is private key and public key e2 is calculate using the formula $e_2 = e_1^d \mod P$ (1)
The open key parameters are (e1, e2, P).

2.2. Encipher process

A random number ‘r’ is selected from the set {1,2,…,P-1}, the ciphertext $(c_1,c_2)$ to encrypt a message ‘m’ is computed as $c_1 = e_1^r \mod P$ (2)
$c_2 = m \cdot e_1^r \mod P$ (3)
The joint message $(c_1,c_2)$ is sent to the receiver.
2.3. Decipher process
Given ciphertext \((c_1, c_2)\) the plaintext can be obtained by \(m = c_2 \cdot (c_1^{-1})^d \mod P\) (4)

\(P\) is the public key where \(a \cdot d\) is the receiver's private key.

3. Laplace Transform (LT)

3.1. Laplace Transform and Inverse Laplace Transform definitions

If \(\tilde{L}\{g(i)\} = \int_0^\infty e^{-st} g(t) \, dt = \tilde{G}(s)\) then \(\tilde{L}^{-1}\{\tilde{G}(s)\} = g(i)\) provide that the integral exists. Where \(\tilde{g}(i)\) function of \(i \in R^+ or C^+\) and the parameter \(s \in R or C\).

3.2. Properties of transform

If \(\tilde{L}\{g_1(i)\} = \tilde{G}_1(s), \tilde{L}\{g_2(i)\} = \tilde{G}_2(s), \ldots, \tilde{L}\{g_n(i)\} = \tilde{G}_n(s)\), then \(\tilde{L}\{\hat{c}_1 g_1(i) + \hat{c}_2 g_2(i) + \ldots + \hat{c}_n g_n(i)\} = \hat{c}_1 \tilde{G}_1(s) + \hat{c}_2 \tilde{G}_2(s) + \ldots + \hat{c}_n \tilde{G}_n(s)\)

where \(\hat{c}_1, \hat{c}_2, \ldots, \hat{c}_n\) are constants.

3.3. Laplace Transform formulas

<table>
<thead>
<tr>
<th>Laplace Transform</th>
<th>Inverse Laplace Transform</th>
</tr>
</thead>
<tbody>
<tr>
<td>(\tilde{L}(i^n) = \frac{(n)!}{s^{n+1}})</td>
<td>(\tilde{L}^{-1}\left(\frac{1}{s^{(n+1)}}\right) = \frac{i^n}{(n)!})</td>
</tr>
<tr>
<td>(\tilde{L}(e^{bt}) = \frac{1}{s - b})</td>
<td>(\tilde{L}^{-1}\left(\frac{1}{s - b}\right) = \hat{e}^{(bt)})</td>
</tr>
<tr>
<td>(\tilde{L}(\sinh bt) = \frac{b}{s^2 - b^2})</td>
<td>(\tilde{L}^{-1}\left(\frac{1}{s^2 - b^2}\right) = \frac{1}{b} \sinh bt)</td>
</tr>
<tr>
<td>(\tilde{L}(i^n e^{bt}) = \frac{(n)!}{(s - b)^{(n+1)}})</td>
<td>(\tilde{L}^{-1}\left(\frac{1}{(s - b)^{(n+1)}}\right) = \frac{i^n e^{bt}}{(n)!})</td>
</tr>
</tbody>
</table>

where \(b \in R^+\).

4. The Proposed algorithm

The ElGamal algorithm is individual character algorithm and it provides repeated character in encipher file with the same frequency. The proposed algorithm is string algorithm. The string is depending on the length of the message and it is observed that the repeated character in encipher file has different frequency.

4.1. Encryption development

The proposed algorithm, the procedure starts with the creation of key. The receiver provides a public key \('e_2'\) generated from its own private key \('d'\). With the generator \('e_1'\) from the \(G = 1, 2, 3, \ldots, P - 1\) a large prime number \(P\) is then determined together. The private key of the sender is also selected \('r'\) from the cyclic group \(G\). The value \('e_2'\) is computed using the formula \(e_2 = e_1^d \mod P\).

Now select positive polynomial function \(f(t) = e^t, \sin t, \cosh t\) and so on. So \(\{P, e_1, e_2, f(t)\}\) are shared publicly and \('d'\) is kept private by the receiver. The sender encrypt the plaintext message \(M\) is a string converted by using ASCII table values \(M_0, M_1, M_2, \ldots, M_n\). With the availability of the public keys. After the above process, continue the following steps.

**Step 1:** Select a polynomial function for example \(\hat{f}(i) = i \hat{e}^i = \sum_{n=0}^{\infty} \frac{i^{(n+1)}}{(n)!}\) A large prime number \(P\) then is determined together

**Step 2:** Calculate \(\hat{g}(i) = M \hat{f}(i) = \sum_{n=0}^{\infty} M_n \frac{i^{(n+1)}}{(n)!}\) where \(M = M_0, M_1, M_2, \ldots, M_n\) is a message.
Step 3: Apply Laplace Transform on both sides in step 2

\[ \mathcal{L}\{\tilde{g}(i)\} = \mathcal{L}\left\{ \sum_{n=0}^{\infty} M_n \frac{i^{(n+1)}}{(n+1)!} \right\} = \sum_{n=0}^{\infty} \frac{M_n}{n!} \tilde{L}\{i^{(n+1)}\} \]

Step 4: Evaluate step 3 using standard Laplace Transform formulas

\[ \mathcal{L}\{\tilde{g}(i)\} = \sum_{n=0}^{\infty} \frac{M_n}{(n+1)!} \left[ \frac{(n+1)}{i^{(n+1)}} \right] = \sum_{n=0}^{\infty} \frac{M_n}{n!} \frac{1}{s^{(n+1)}} \]

where: 's' is a parameter

- '+' is a multiplication

\[ \sum_{n=0}^{\infty} N_n = \sum_{n=0}^{\infty} n \cdot M_n \]

Step 5: Using step 4 and ElGamal algorithm calculate \( C(1,1) = e_1^r \mod P \) & \( C(2,n) = (N_n \cdot e_2^r) \mod P \)

where \( r \) : sender private key and \( C(1,1), C(2,n) \) are send privately.

The ciphertext is \( C(2,0), C(2,1), \ldots, C(2,n) \).

4.2. Decryption development

Step 1: Consider the ciphertexts \( C(2,0), C(2,1), \ldots, C(2,n) \) and \( C(1,1) \).

The receiver computes the values of \( N_0, N_1, N_2, \ldots, N_n \) using formula.

\[ N_0 = \{C(2,0) \cdot (C(1,1)^{-1}d)\} \mod P, \quad N_1 = \{C(2,1) \cdot (C(1,1)^{-1}d)\} \mod P, \ldots, N_n = \{C(2,n) \cdot (C(1,1)^{-1}d)\} \mod P. \]

Step 2: Substitute the values \( N_n (n = 0, 1, 2, \ldots, \infty ) \) obtained in step 1 in step 4 of encryption developed

\[ \frac{1}{s^{(n+1)}} \]

Step 3: Apply inverse Laplace Transform on both sides

\[ \tilde{g}(i) = \sum_{n=0}^{\infty} N_n \tilde{L}^{-1}\left\{ \frac{1}{s^{(n+1)}} \right\} \]

Step 4: Evaluate step 3 using inverse Laplace Transform formulas

\[ \hat{g}(i) = \sum_{n=0}^{\infty} N_n \frac{i^{(n+1)}}{(n+1)!} = \sum_{n=0}^{\infty} \frac{N_n}{n!} \frac{i^{(n+1)}}{n!} \]

Step 5: Express \( \hat{g}(i) = \sum_{n=0}^{\infty} M_n \frac{i^{(n+1)}}{(n+1)!} \) where \( M_n = \frac{N_n}{n!} \)

Step 6: Express \( \hat{g}(i) = M \hat{f}(i) \)

where \( M = M_0, M_1, M_2, \ldots, M_n \) is a message.

5. Example of Proposed method

5.1. Encryption development

The proposed algorithm, the process starts from the stage of key generation where in the receiver provides a public key \( 'e_2 = 222' \) generated from its own private key \( 'd = 33' \). A large prime number \( P = 283 \) then is determined together with the generator \( 'e_1 = 47' \) from the cyclic group \( G = 1,2,3,\ldots,283 \). The private key of the sender is also selected \( 'r = 19' \) from the cyclic group \( G \). The value \( 'e_2' \) is calculated using the formulae \( e_2 = e_1^d \mod P \).

The sender encrypt the plaintext message \( M_n = \text{GOOOD} \) is string converted by using ASCII table values \( M_0 = 71, M_1 = 79, M_2 = 79, M_3 = 79, M_4 = 68 \) where \( M_5 \geq 0 \). After the above process, continue the following steps.

Step 1: Selected a polynomial function for example \( f(i) = i \cdot e^i \)

\[ = \sum_{n=0}^{\infty} \frac{i^{(n+1)}}{(n)!} = i + \frac{i^2}{2!} + \frac{i^3}{3!} + \frac{i^4}{4!} \]

Step 2: Calculate \( \hat{g}(i) = M \hat{f}(i) = \sum_{n=0}^{\infty} M_n \frac{i^{(n+1)}}{(n)!} = 71i + 79 \frac{i^2}{1!} + 79 \frac{i^3}{2!} + 79 \frac{i^4}{3!} + 68 \frac{i^5}{4!} \)

where \( M = M_0, M_1, \ldots, M_n \) is a message.
Step 3: Apply Laplace Transform on both sides in step 2

\[ \mathcal{L}\{g(i)\} = \mathcal{L}\left\{ \frac{71}{i^2} + \frac{79}{2i^3} + \frac{79}{3i^4} + \frac{68}{4i^5} \right\} \]

\[ \mathcal{L}\{g(i)\} = 71\mathcal{L}\{i^2\} + \frac{79}{2i^3} \mathcal{L}\{i^3\} + \frac{79}{3i^4} \mathcal{L}\{i^4\} + \frac{68}{4i^5} \mathcal{L}\{i^5\} \]

Step 4: Evaluate step 3 using Laplace Transform formulas

\[ \mathcal{L}\{g(i)\} = 71\left( \frac{1}{s^2} \right) + \frac{79}{2i^3} \left( \frac{1}{s^3} \right) + \frac{79}{3i^4} \left( \frac{1}{s^4} \right) + \frac{68}{4i^5} \left( \frac{1}{s^5} \right) \]

\[ \mathcal{L}\{g(i)\} = \frac{71}{s^2} + \frac{79}{2} \left( \frac{1}{s^3} \right) + \frac{79}{3} \left( \frac{1}{s^4} \right) + \frac{68}{4} \left( \frac{1}{s^5} \right) \]

\[ \mathcal{L}\{g(i)\} = \frac{71}{s^2} + \frac{158}{s^3} + \frac{237}{s^4} + \frac{316}{s^5} + \frac{340}{s^6} \]

Where \( N_0 = 71, N_1 = 158, N_2 = 237, N_3 = 316, N_4 = 340 \) and ‘s’ is a parameter.

Step 5: Using step 4 and ElGamal algorithm, calculate

\[ C_{(1,1)} = e_1^r \mod P = (47)^{19} \mod 283 = 133 \]

\[ C_{(2,0)} = (N_0 \times e_2^r) \mod P = (N_0 \times e_1^r) \mod P = (71 \times 133^{19}) \mod 283 = 279 \]

\[ C_{(2,1)} = (N_1 \times e_2^r) \mod P = (N_1 \times e_1^r) \mod P = (158 \times 133^{19}) \mod 283 = 19 \]

\[ C_{(2,2)} = (N_2 \times e_2^r) \mod P = (N_2 \times e_1^r) \mod P = (237 \times 133^{19}) \mod 283 = 170 \]

\[ C_{(2,3)} = (N_3 \times e_2^r) \mod P = (N_3 \times e_1^r) \mod P = (316 \times 133^{19}) \mod 283 = 38 \]

\[ C_{(2,4)} = (N_4 \times e_2^r) \mod P = (N_4 \times e_1^r) \mod P = (340 \times 133^{19}) \mod 283 = 220 \]

\[ C_{(1,1)} = 133 \text{ and } C_{(2,0)} = 279, \quad C_{(2,1)} = 19, \quad C_{(2,2)} = 170, \quad C_{(2,3)} = 38, \quad C_{(2,4)} = 220 \text{ are ciphertext values.} \]

Chiphertext \( C_{(2,0)}, C_{(2,1)}, \ldots, C_{(2,4)} \) converted into ASCII characters and stored ciphertext message.

5.2 Decryption Development

Step 1: The ciphertext message receiver convert by using ASCII table values are \( C_{(2,0)} = 279, C_{(2,1)} = 19, C_{(2,2)} = 170, C_{(2,3)} = 38, C_{(2,4)} = 220 \) are ciphertext values. Chiphertext \( C_{(2,0)}, C_{(2,1)}, \ldots, C_{(2,4)} \) converted into ASCII characters and stored ciphertext message.

Step 2: Substitute the values \( N_0, N_1, N_2, \ldots, N_4 \) obtained in step 1 in step 4 of encryption developed.

\[ \mathcal{L}\{g(i)\} = \sum_{n=0}^{\infty} N_n \left( \frac{1}{s^n} \right) = \frac{71}{s^2} + \frac{158}{s^3} + \frac{237}{s^4} + \frac{316}{s^5} + \frac{340}{s^6} \]

Step 3: Apply inverse Laplace Transform on both sides of step 2

\[ g(i) = 71 \mathcal{L}^{-1}\left\{ \frac{1}{s^2} \right\} + 158 \mathcal{L}^{-1}\left\{ \frac{1}{s^3} \right\} + 237 \mathcal{L}^{-1}\left\{ \frac{1}{s^4} \right\} + 316 \mathcal{L}^{-1}\left\{ \frac{1}{s^5} \right\} + 340 \mathcal{L}^{-1}\left\{ \frac{1}{s^6} \right\} \]

Step 4: Evaluate step 3 using Inverse Laplace Transform formulas

\[ g(i) = 71 \frac{i^2}{2!} + 158 \frac{i^3}{3!} + 237 \frac{i^4}{4!} + 316 \frac{i^5}{5!} + 340 \frac{i^6}{6!} \]

\[ g(i) = 71 \frac{i^2}{2!} + \left( \frac{158}{2} \right) \frac{i^3}{3!} + \left( \frac{237}{3} \right) \frac{i^4}{4!} + \left( \frac{316}{4} \right) \frac{i^5}{5!} + \left( \frac{340}{5} \right) \frac{i^6}{6!} \]

Step 5: Express \( g(i) = 71 \frac{i^2}{2!} + 79 \frac{i^3}{3!} + 79 \frac{i^4}{4!} + 68 \frac{i^5}{5!} \)

Step 6: Express \[ g(i) = \sum_{n=0}^{\infty} M_n \left( \begin{array}{c} n+1 \cr n \end{array} \right) = M \tilde{f}(i) \]

where \( M \) is message string is “GOOOD.”
6. Execution of proposal

The proposed algorithm is executed with the help of Visual Studio 2010 on 32-bit operating system of windows 7. The .NET program is used to execute the proposal algorithm with Pentium (R) processor (RAM of 2 GB & speed 2.40 GHz).

6.1. Test results and security Analysis

Using planned algorithm, we attain frequency distribution, encryption and decryption time results with the help of statistical tools. We calculate measures of central tendency such as mean & median for some files and calculate measures of dispersion such as standard deviation and quartile deviation. Correlation coefficient is calculated between plaintext and ciphertext for each algorithm mentioned in the paper.

6.1.1. Frequency test

Figures I & II provide the frequency of same character in a plaintext has same frequency after encryption using ElGamal algorithm and RSA algorithm, where plaintext and frequency level of ciphertext are considered on x – axis and y – axis respectively.

![ElGamal Frequency Distribution](image1)

**Fig I: ElGamal algorithm ciphertext frequency distribution**

![RSA Frequency Distribution](image2)

**Fig II: RSA algorithm ciphertext frequency distribution**

Figure III presents the frequency of each character in a plaintext has different frequency after encryption using proposed algorithm ElGamal using Laplace Transform (ElGamal using LT), where plaintext and frequency level of ciphertext are considered on x – axis and y – axis respectively.
Comparison of frequency distribution explained in figure I, II & III for each algorithm is explained graphical representation in figure IV.

6.1.2. Encryption and Decryption Time results

The proposed algorithm will encrypt the plain text and decrypt from encrypted text and it indicates the time difficulty, which shows how the three algorithms are obtained by using the computer program. Encryption times with file size presented in table I and decryption times with file size presented in table II. The obtained time complexities graphically represented with the help of bar graphs in Figure V & VI.

<table>
<thead>
<tr>
<th>Messages</th>
<th>Message size in Bytes</th>
<th>Encryption results (in millisecond)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>ElGamal</td>
</tr>
<tr>
<td>Message-I</td>
<td>51</td>
<td>20</td>
</tr>
<tr>
<td>Message-II</td>
<td>101</td>
<td>45</td>
</tr>
<tr>
<td>Message-III</td>
<td>151</td>
<td>66</td>
</tr>
<tr>
<td>Message-IV</td>
<td>201</td>
<td>88</td>
</tr>
<tr>
<td>Message-V</td>
<td>251</td>
<td>110</td>
</tr>
</tbody>
</table>
6.2. Central Tendency and Dispersion: ways of measurement

The measurements of Central Tendency and Dispersion have been used as a measure of homogeneity until now. The comparative analysis as presented in Table III shows that the measures are different in original message and encrypted message. The measures in ciphertext are different and it shows that ciphertext have different mean, median. Similarly the measures of central dispersion in ciphertext are different which shows that ciphertext have different stranded deviation and quartile. As per the table III date, it is not easy to break the original message.
<table>
<thead>
<tr>
<th>Samples files</th>
<th>Plaintext</th>
<th>ciphertext</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>ElGamal</td>
<td>RSA</td>
</tr>
<tr>
<td>Sample I</td>
<td>110</td>
<td>125</td>
</tr>
<tr>
<td>Sample II</td>
<td>101</td>
<td>92</td>
</tr>
<tr>
<td>Sample III</td>
<td>105</td>
<td>146</td>
</tr>
<tr>
<td>Sample IV</td>
<td>105</td>
<td>125</td>
</tr>
<tr>
<td>Sample V</td>
<td>111</td>
<td>205</td>
</tr>
</tbody>
</table>

**STANDARD DEVIATION**

<table>
<thead>
<tr>
<th>Samples files</th>
<th>Plaintext</th>
<th>ciphertext</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>ElGamal</td>
<td>RSA</td>
</tr>
<tr>
<td>Sample I</td>
<td>24.1609</td>
<td>72.1345</td>
</tr>
<tr>
<td>Sample II</td>
<td>25.2199</td>
<td>58.6898</td>
</tr>
<tr>
<td>Sample III</td>
<td>26.0277</td>
<td>65.1991</td>
</tr>
<tr>
<td>Sample IV</td>
<td>31.5687</td>
<td>65.3814</td>
</tr>
<tr>
<td>Sample V</td>
<td>8.2264</td>
<td>43.7609</td>
</tr>
</tbody>
</table>

**QUARTILE DEVIATION**

<table>
<thead>
<tr>
<th>Samples files</th>
<th>Plaintext</th>
<th>ciphertext</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>ElGamal</td>
<td>RSA</td>
</tr>
<tr>
<td>Sample I</td>
<td>99</td>
<td>73.5</td>
</tr>
<tr>
<td>Sample II</td>
<td>98</td>
<td>76</td>
</tr>
<tr>
<td>Sample III</td>
<td>97</td>
<td>71</td>
</tr>
<tr>
<td>Sample IV</td>
<td>97</td>
<td>82</td>
</tr>
<tr>
<td>Sample V</td>
<td>111</td>
<td>205</td>
</tr>
</tbody>
</table>

**6.2.1 Correlation coefficient**

In Table IV correlation coefficient between three algorithms ElGamal, RSA, ElGamal with LT of two sample original message and corresponding encrypted message are calculated and presented from which it can be concluded that the proposed algorithm shows good results.
Table IV: Correlation coefficient

<table>
<thead>
<tr>
<th>Algorithms</th>
<th>Correlation</th>
</tr>
</thead>
<tbody>
<tr>
<td>ElGamal: message1</td>
<td>0.52825</td>
</tr>
<tr>
<td>ElGamal: message2</td>
<td>0.17436</td>
</tr>
<tr>
<td>RSA algorithm: message1</td>
<td>0.39376</td>
</tr>
<tr>
<td>RSA algorithm: message2</td>
<td>0.44188</td>
</tr>
<tr>
<td>ElGamal with LT: message1</td>
<td>0.58958</td>
</tr>
<tr>
<td>ElGamal with LT: message2</td>
<td>0.54783</td>
</tr>
</tbody>
</table>

7. Conclusion

The present work expands a new proposal on implementation of ElGamal scheme for Laplace Transform cryptosystem of function providing a large prime number. In this work, we can execute high-level data security compared with other existing algorithms. Main result of the work is to obtain the frequency of each repeated character in a plaintext has different frequency after encryption. This work can be extended for further Fourier Transform in cryptography, Z – transform in cryptography, Natural Transform in cryptography, Laplace – Mellin Transform in cryptography, Sumudu Transform in cryptography and so on.

References


DOI : 10.21817/indjcse/2020/v11i1/201101032

Vol. 11 No. 1 Jan-Feb 2020