Journal of Applied Sciences 5 (5): 883-886, 2005
ISSN 1812-5654

Security and Performance of ElGamal Encryption Parameters

Allam Mousa
Department of Electrical Engineering, An Najah National University, Nablus, Palestine

Abstract: ElGamal encryption/decryption algorithm is based on the difficulty of discrete logarithm problem where it is strait forward to raise numbers to large powers but it is much harder to do the inverse computation of the discrete logarithm. The ElGamal algorithm depends on certain parameters which are affecting the performance, speed and security of the algorithm. Here, the importance of these parameters and the role it takes in the security and complexity of the system are analyzed, particularly the effect of changing the length of the modulo number and the private key number are investigated.

Key words: Encryption, public key, private key, security, ElGamal parameters

INTRODUCTION

The importance of cryptography systems and techniques is becoming a fundamental issue for large sector of society. The use of encryption is so important for both storing and transmitting the data. In order to secure this digital data, strong cryptography techniques are required.

The importance of encryption is increasing rapidly due to the growing traffic on the public networks like the Internet. Private persons, companies, government agencies and other organizations use encryption techniques in order to safely communicate with partners and costumers. Encryption, here, also increases the security of the internal process to prevent stored information.

For communication over an open and public network, encryption is used to ensure[1]:

- Confidentiality: preventing the unauthorized reception of the message
- Authentication: verifying the message's origin
- Integrity: establishing that a received message has not been altered

The encryption algorithms are mainly divided into two types; 1) block cipher which split the plaintext into fixed length fragments (blocks) and then operate on each fragment separately to produce a corresponding ciphertext fragments. 2) Stream cipher which operates on the plaintext bit by bit (or character by character) rather than on plaintext block.

The cryptosystem can be one of two kinds; the symmetric encryption and asymmetric encryption. In both cases, it requires the use of some keys to perform the encryption and decryption process. Symmetric key encryption uses the same key for both encrypting and decrypting the data, this kind is also known as secrete key encryption. On the other hand, asymmetric encryption uses two different keys; the first one is the public key used to encrypt the data and the other one is the private key used to decrypt the data, asymmetric encryption is referred to public key encryption[2].

It is well known that the security of the RSA cryptosystem depends on the difficulty of factorization large numbers[3]. However ElGamal system, which is a public key cryptosystem, is based on the difficulty of discrete logarithm problem where it is directly forward to raise numbers to large powers but it is much harder to do the inverse computation of the discrete logarithm[2].

The discrete logarithm: When dealing with real numbers, then the solution of the equation y=b^x can be found such that $x= \log_b y$. Given the integers b and n such that b<n, the discrete logarithm of integer y to the base b is an integer x, such that b^x=y mod n. Hence, unlike logarithms, the discrete logarithm problem is defined in a discrete domain where a solution must be exact[4].

ElGamal encryption/decryption algorithm: The ElGamal encryption system parameters consists of a prime number (p) and an integer number (g) which is a root of (p-1) and whose power modulo (p) generate a large number of elements[4]. To encrypt plaintext, a private key (a) is used to generate the public key (y) as given in Eq. 1 such that (a) is an integer between 1 and p-2[5].

$$y=g^a \bmod p \quad (1)$$

The public key for ElGamal encryption algorithm consists of the triple (p,g,y).

To encrypt a plaintext message (m), a random integer (k) is chosen such that it is between 1 and p-2[6]. The message is converted to numbers before producing the ciphertext (c) which consists of the pair (y1,y2) calculated as given by Eq. 2a and b[5] ;

$$y1=g^k \bmod p \quad (2a)$$

$$y2=m\, y^k \bmod p \quad (2b)$$

The decryption process is applied in a reverse order such that the encrypted message (y1,y2) is used with the private key (a) and the prime number (p) to retrieve the original message (m) as given by Eq. 3[5] ;

$$m=y2/\, y1^a \bmod p \quad (3)$$

The division by y1^a in Eq. 3 should be interpreted in the context of modular arithmetic[4], that is y2 is multiplied by the inverse of y1^a.

The main functions of the algorithm can be summarized as illustrated by the following steps;

Step1 initialization:

1) Chose a random prime number (p) such that it is limited by the number of bits (bit) used to generate it.
2) Chose the value of the private number (a)
3) Chose the value for the integer generator number (g)
4) Compute the value of the public key y=g^a mod p
5) Publicize the values (p,g,y) and keep the private key.

Step 2 prepare the data:

1) Chose the value of the random number (k)
2) Get the data to be encrypted (text, image or sound)
3) Divide the long message into smaller blocks.
4) Convert the characters to numbers

Step 3 encryption: Compute the ciphertext (y1,y2) of each block.

Step 4 decryption: Decipher the ciphertext (y1, y2) of each block.

Example: Let the prime number (p) equals 23 and chose the integer (g) as 11. Chose the private key (a) as 6 which is between 1 and p-2. Computing the public key as given by Eq. 1; y=11^6 mod 23 resulting in y=9, now the public key (p,g,y) is (23,11,9). To encrypt the plaintext message m=10, chose the value of the random number (k) as 3 then calculate the values of y1 and y2 as given by Eq. 2 such that y1=11^3 mod 23 and y2=10 9^3 mod 23 yielding the ciphertext (y1,y2) as (20,22). Hence, the plaintext 10 is transformed into the ciphertext (20,22). The decryption of this pair will produce the original message as given by Eq. 3.

It is clear here that an important step of the algorithm is the initialization of the parameters described above. Choosing the number of bits representing the prime number (p) and then use this number to calculate the consecutive values and so encrypt the input message producing the ciphertext (y1,y2). It is also important to notice here that if the message is huge in length it should be divided into shorter blocks of appropriate and equal length called cut length. Each one of these blocks is encrypted separately and the process is repeated for all of these blocks. This reduction of the message length is so essential before the encryption process due to the huge numbers which may arise where the message is converted to a number. Hence ElGamal algorithm may be applied for text, image or voice data. The main difference is the data size and so the number of blocks that the message will be divided.

Being a public key encryption algorithm, ElGamal algorithm depends on both the private and secret keys it generates to encrypt the data[7]. Thus the main problem becomes securing and managing these keys. This security can be achieved by using certain hardware modules with key management software[8]. Moreover, the length of the keys used to encrypt data is so important in the amount of security that can be achieved; hence, a 64-bits symmetric key will keep the data secure for a long time[2].

Simulation results: The importance of ElGamal parameters is analyzed by changing only one parameter at a time while keeping the other parameters unchanged. The algorithm depends mainly on the prime number whose value depends on the number of bits used to generate this prime. It also depends on the private key, the input message and the (cut length) of the block message.

Initially the parameters are set such that the plain text is (how are you), the cut length is equal to the whole message length and the private key is chosen as five characters. The implementation given in[9] was mainly used here.

Changing the length of the prime number (p) length: The value of the prime number (p), plays an important role in forming the values of the public key (p,g,y) and hence the encrypted message pair (y1,y2). This prime number is a function of the number of bits used to generate it. Changing this bit number is directly reflected on the length of the other parameters as shown in Table 1.

Table 1: Effect of changing the prime number (p) length

bit length	p length	g length	y length	y1 length	y2 length
1	1	1	1	1	4
10	3	1	2	3	7
20	6	1	6	5	10
50	15	1	15	15	20
100	31	1	30	30	35
128	37	2	37	37	42
130	38	1	37	37	42
150	42	1	42	42	46

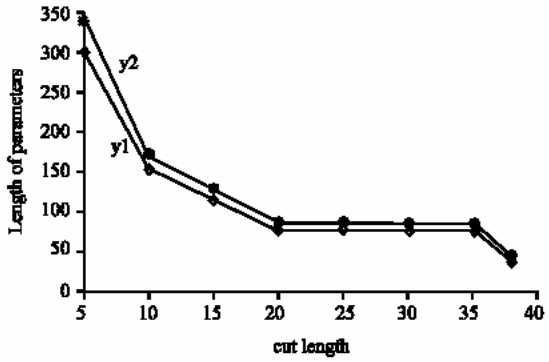

Fig. 1: Effect of the cut length on the chiphertext

The length of the prime number (p) is directly proportional to its generating number of bits since the maximum value of (p) is restricted by 2^bit (Table 1). The length of the encrypted message (y1,y2) also increases as a function of the prime number (p). The security is improved by increasing this ciphertext. The length of the integer (g) is not directly affected by the value of (bit), whereas length of the public key (y) increases in a manner similar to that of the prime number (p) (Table 1).

The results obtained in Table 1 show that (p), (y) and (y1) have almost the same values, where as (y2) behaves in a similar way but with larger values. On the other hand, the results of (g) is almost constant for all values of the modulo (p).

Changing the private key length: The private key plays an important role in generating the public key and hence the ciphertext values as given by Eq. 1 and 2. To study the performance of this private key, the previous parameters are kept as stated before while the private key length is changed. The value of (bit), which is used to generate the prime number (p) is fixed to 128 then the secrete key is chosen with a variable length. This change has shown no clear effect on the other parameters

Changing the cut length: For a short message as the one given here, changing the cut length has no clear effect on the other parameters. However it may be only the execution time which may be varying as the cut length changes. To see the effect of this cut length parameter,

Fig. 2: Typical grayscale image to be encrypted

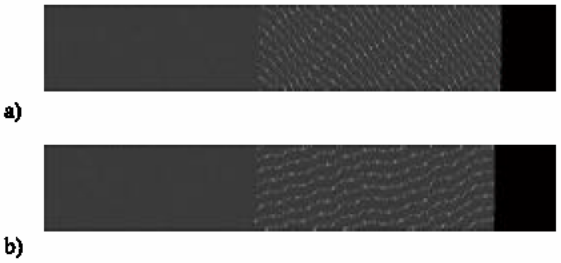

Fig. 3: Encrypted data of an image, a) grayscale image, b) color image

the message is changed into a longer one like (how are you? The weather is nice today). This 38 characters message is treated as blocks of certain length each. This length is changed as given in Fig. 2. Obviously, as the cut length increases then the length of the ciphertext (y1, y2) decreases but this has no effect on the other parameters.

The previous analysis where performed for a text data, however, when image is applied to ElGamal algorithm, a huge data size will be treated compared to the text input, this results in a large ciphertext numbers (y1, y2) which makes it even harder for attacking. Moreover, the numbers (y1, y2) do not tell if they are representing a text or image plaintext data. The image data can be of any format such as the 'bmp' format, uncolored or colored 'tif' format images. One major difference from the text data is the execution time it takes to perform encryption/deception due to the relatively huge data size of the image compared to the text data. As an example of the grayscale image, the image is used as shown in Fig. 2 and another one is used as an example of the color tif format image. The length of the values y1 and y2 were so huge and the ciphertext (y1, y2) were reshaped to form the encrypted image which is shown in Fig. 3 for both the grayscale and the color images.

Sound data can also be used as an input to ElGamal encryption algorithm and there will be no difference in the analysis for different types of input data.

CONCLUSIONS

The parameters which are controlling the performance of ElGamal algorithm were analyzed. It turns out that the length of the modulo parameter is affecting directly the length and hence the security of the ciphertext pair (y1,y2). Also, the data length being processed is important. On the other hand, the choice of the private key has no direct effect on the length of the ciphertext. For image data, the size of the ciphertext is very huge and reshaping the encrypted data was not understood.

REFERENCES

1. Ian, A.G. and P.M. Grant, 2004. Digital Communications 2nd Edn., Prentice Hall
2. Wenbo Mano, 2004. Modern Cryptography Theory and Practice, Prentice Hall
3. Alkar, A.Z. and R. Sonmer, 2004. A hardware version on the RSA using the montgomery's algorithm with systolic arrays. Intl. VLSI J., 38: 299-307.
4. http://www.ics.uci.edu/~goodrich/teach/ics247/W03/notes/elgamal.pdf (dated 1/10/2004)
5. http://www.x5.net/faqs/crypto/q29.html (dated 1/10/2004)
6. http://www.cs.adfa.edu.au/courses/ACSC2010/coursework/lectures/less19.html (dated 1/10/2004)
7. http://diamond.boisestate.edu/~marion/teaching/Skopje/el-gamal.htm (dated 1/10/2004)
8. Shparlinski, I.E., 2004. On the uniformity of distribution of the decryption exponent in fixed encryption exponent RSA. Information Processing Lett., 92: 143-147.
9. http://islab.oregonstate.edu/koc/ece575/02Project/Sin+Cha (dated 1/1/2003)