ISTANBUL UNIVERSITY –
JOURNAL OF ELECTRICAL & ELECTRONICS ENGINEERING

YEAR : 2003 (801-806)
VOLUME : 3
NUMBER : 1

A SECURE E-MAIL APPLICATION USING THE ELGAMAL ALGORITHM: MD MESSAGE CONTROLLER

Mustafa DÜLGERLER[1] **M. Nusret SARISAKAL**[2]

[1] Microsoft Turkey

[2]Istanbul University, Engineering Faculty, Computer Engineering Department
34850, Avcilar,Istanbul,Turkey

[1]E-mail: mdulgerler@hotmail.com [2]E-mail: nsarisakal@istanbul.edu.tr

ABSTRACT

In this study, it is examined how data can be sent to a receiver on more secure ways through sending e-mail. In this project The ElGamal Encryption Algorithm is used while sending an e-mail. Identity Authentication is performed by a unique hash function. This application is originated from an interface which runs on Windows Operating Systems and is called MD Message Controller.

Keywords: *Security, Encryption, The ElGamal Algorithm, Hash Function, E-mail*

1. INTRODUCTION

Current world technologies continuously changes and develops. All technological developments bring some problems besides. Internet Applications are more common today. The biggest problem of these applications is security. New algorithms and methods are being developed to make these applications more secure. At this point, software developers have to try new methods then if they get success, they apply these methods in to their own applications E-mail applications which nowadays are used commonly today's world, started to be used firstly at America Defence Ministry in 1970 that is safe and fast communication which every body wishes. Many processes are done by sending mails now such as corresponding in a company or corresponding internationally. Because there is a copy of messages at each side so conformation is possible in any case. Besides them, its cost, speed and affectivity are other important features of sending mails. So software sending mails in large environments with these features explained above attracts humans who have bad intentions. First method to protect our software from their attacks is to use a strong encryption algorithm. We know a lot of algorithm about encryption but they are also known by someone who has some bad wishes. Therefore we have to derivate new methods from old algorithms. To develop new user friendly software with new methods in well-built programming languages by coding, will solve some security problems.

When designing new security algorithms, benefits from cryptology science. Cryptography involves some encryption, keying and decryption algorithms. In this project, The ElGamal

Received Date :19.07.2002
Accepted Date: 24.12.2002

Encryption Algorithm is used to encrypt data. Identity Authentication is obtained with an unique SHA (Secure Hashing Algorithm) function.

First let us define some definitions used in cryptology: sender, who sends message; receiver, who gets message. Encryption is a transformation process that changes a message into incomprehensible form with nonlinear functions. A form of message which can be understood by anyone by reading is called as "plaintext" and encrypted form of plaintext is called as "cipher text". Changing a cipher text into plaintext with inverse function of function used for encryption to plaintext before, is called as "decryption".

Let us point out a text which will be forwarded from sender to recipient, with "P", a cipher text encrypted with any encryption algorithm with "C", and encryption function with "F". Generally length of cipher text is longer than length of plaintext. In this case, we compress cipher text with compress algorithms. According to these definitions, mathematical encryption model is; (1)

$$\mathbf{F(P)=C} \qquad \mathbf{(1)}$$

Decryption model is; (2)

$$\mathbf{F^{-1}(C)=P} \qquad \mathbf{(2)}$$

F^{-1} is inverse function of F and is called "decryption function".

Generally, There is a decryption function for each encryption function. If we want to build a strong encryption algorithm, linear functions must not be used. If not, decryption time is made smaller for malicious persons.

Cryptography was used only by political or military communications for long time. As usage areas of cryptography expanded it was become standard. The ElGamal Encryption algorithm, used in this project, is a public-key algorithm introduced in 1985 by T. ElGamal . There has been no successful attack on this algorithm ever reported. In 1994, several independent research teams presented their findings that all discrete logarithm based public-key digital signature algorithms are variants of a generalized "meta" algorithm. These various logarithms included by The ElGamal indicates its security [1,2].

This type of methods are based on finding the private-key from the public-key. This depends on the length of the public/private key pair and the computing difficulty that might be used to "crack" the key pair.

The key length of The ElGamal can range from 256-bit to arbitrarily long. A key length ranging from 1024 to 2048 bits are considered safe for the next 20 years. Of course, this prediction is based on the current computing power and the rough estimate of hardware and cryptanalysis advances in the near future. Private key can range from 160-bit to 240-bit [3,4].

The analysis carried out shows that RSA and The ElGamal have similar security for equivalent key lengths.

2. INFRASTRUCTURE OF THE ELGAMAL ALGORITHM

The ElGamal system is a public-key cryptosystem based on the discrete logarithm problem. It consists of both encryption and signature algorithms. The encryption algorithm is similar in nature to the Diffie-Hellman key agreement protocol.

The system parameters consist of a prime p and an integer g, whose powers modulo p generate a large number of elements. Receiver has a private key a and a public key y, where $y = g^a \pmod p$. Let's Suppose a *sender* wishes to send a message m to a *receiver*. The sender first generates a random number k less than p, then computes (3)

$$\mathbf{y_1 = g^k \pmod p \text{ and } y_2 = m \text{ xor } y^k} \qquad \mathbf{(3)}$$

where xor denotes the bit-wise exclusive-or and gets y_1 and y_2[5,6]. The s*ender* sends (y_1 ,y_2) to the *receiver*. Upon receiving the cipher text, the *receiver* computes (4)

$$\mathbf{m = (y_1^a \pmod p) \text{ xor } y_2} \qquad \mathbf{(4)}$$

then the *receiver* has original message called "plaintext".

2.1. Encryption Process in The Application

This section shows how we used The ElGamal Encryption Algorithm in our project.

First process is to divide plain text in to blocks after it is taken. Encryption is applied on these blocks and block encryption is ensured. In order to define the length of blocks, namely how many members the blocks will have , the following process is executed. (5)

Block=int(log(p)/log(strlen(alph))) **(5)**

Alph symbolizes alphabet. The alphabet used in our project includes basic symbols and contains 96 characters. *Strlen* is a function which calculates the number of characters in our alphabet. The value of P is our key. The rates of logarithm of the value of P to logarithm of number of characters in our alphabet equals to number of characters in each block. While the length of alphabet decreases or the value of P increases , the length of blocks increases. As known, a well designed encryption algorithm produces a cipher text and a character for plain text. For example, if a character is encrypted, at the end of encryption two characters are produced. That means if our blocks have only one character, after encryption a cipher text which has twice as much as the number of the characters in plain text is produced. But if plain text is divided in to blocks, this problem is prevented. For instance, if a plain text including 40 characters after divided into blocks including five characters, is encrypted; a cipher text including 48 characters is obtained. To divide plaintext in to blocks saved memory. The disadvantage increasing of block dimension is making algorithm simplier. Consequently, in this application, value of p is defined as such to make the value of the block 1.

After this step, characters in plaintext are scanned one by one. First character is taken. A random k is defined. The random k is modulated to make it less than the value of p. Then y_1 is computed according to k, p and g parameters (3). Computed the value of y_1 is modulated to the length of alphabet (6).

$y_1 = y_1 \pmod{\text{strlen(alph)}}$ **(6)**

cipher1[i]=alph[y_1] **(7)**

y_1th member in alphabet is assigned to an array named cipher1 (7). (index *i* signs which character is encrypted). This array includes encrypted characters. First taken character is compared with alphabet and its equivalent and its equivalent's index are defined. y_2 is computed with index, k, p, and y (3). Similarly y_2 is reduced according to length alphabet(8).

$y_2 = y_2 \pmod{\text{strlen(alph)}}$ **(8)**

cipher2[i]=alph[y_2] **(9)**

y_2 th member in alphabet is assigned to an array named as *cipher2* (9). This array has secondary encryption keys works as a key for encrypted character. After all process is completed, data in *cipher1* and *cipher2* is combined into an array named as ***cipher*** (10).

cipher[i]=cipher1[i]+cipher2[i] **(10)**

when scanning process of all characters is completed, ***cipher*** array has encrypted text.

2.2. Decryption Process in The Application

First, numbers of blocks in ciphertext is defined, using formula 5. A loop repeating *Strlen(cipher)/2*block+2* times is formed. Numerical equivalent of character is assigned to array named as "*plain1*". This value is index of character in alphabet. Through this value, y_1 is computed. Numerical equivalent of following character is assigned to an array named as "*plain2*". These secondary associated characters are produced to help encryption while encryption. Through this value assigned to plain2 array, y_2 is computed. Then using p and a values, $(y_1^a \bmod p) \text{ xor } y_2$ is computed. This function returns an index. Character at that index in alphabet is our decrypted character. Result is assigned to an array named as "*plain*". When all characters are scanned, *plain* has decrypted version of cipher text.

3. HASH ALGORITHM

This algorithm is used while authorized users are registered. Any user cannot use program without registering before. Users enter a password during register. This password is not directly saved into database. It is transformed to another form by a

hashing function, then saved into database. So it is made harder that unauthorized persons get users' passwords. Now let us describe our hash function used in this application . First, entered password is obtained as the index of the first character is 0 then so on, by giving an index to each character. Character's index number is raised to the power of two and the result is multiplied by character's ASCII value. All results are added. If the result is greater than an random epsilon, the result is divided into two pieces, then right piece and left piece are added. This process continues until the result is equal or less than the value of random epsilon.

For example, our password is "mustafa". Realise that all letters are lower cases. As known, ASCII codes of lower cases and upper cases are different. So we have to keep this form of password in other logins. Although the meaning of "Mustafa" equals to "mustafa", ascii code of "M" differs from ascii code of "m", so if you enter "Mustafa" as password, that wont be accepted. Character in entered password are divided in to index. Character m has index 0, u 1, s 2, t 3, a 4, f 5, and a 6 . ascii code of these characters :m:109, u:117, s:115, t:116, a:97 and f=102. then the following function is calculated.
$109*2^0+117*2^1+115*2^2+116*2^3+97*2^4+102*2^5+97*2^6$

Result is 12755. Assume that epsilon is 1000. that value is divided into two pieces because it is greater than epsilon. It is divided into two pieces as right-most. That is, if there is an odd number of digits in the number, the separation operation is done as such to leave one more digit on the right side than the left side. In our example, number 12755 has 5 numbers. Consequently first two numbers represent a number, last three numbers represent another number such as 12 and 755. Then these two new number are added. 12+755=767. due to result is lower than epsilon, process is ended. This value is saved in database at password field instead of password user entered first. When user login to program, user will write his/her password in usual form, then click "OK" button. Entered password is subjected to procedures explained above, then found value is compared with read value from database; if they are equals, user login to program.

4. E-MAIL APPLICATION : MD MESSAGE CONTROLLER

In this project, we used ElGamal which is one of the encryption algorithms designed for distributed systems and used to sign e-mails, to encrypt all of the message. Further when the program is used on any local operating system to provide high-level security in login processes and to prevent from accessing of unauthorized users to the program, we used a hash algorithm during saving password to the database.

4.1. Enter of Application

When program is run, a *user login window* is shown. User enters his/her user name and password in this form(Figure1).

Figure 1. User Login Window

Entered password is encrypted with hash algorithm explained above. Encrypted password is compared with password encrypted on the same way then saved in database. If the values are equals, user is authorized to use program. When user starts using program, program reads SMTP an POP3 servers which were registered for that user during register, from database and connects

4.2. Received Messages

E-mails sent are read from POP3 server then are listed (Figure 2).

If user wishes to read one of listed e-mails, user double clicks over subject of e-mail with mouse or clicks "READ" button on the form (Figure 3). If the e-mail was sent by any user, who uses the same program, in encrypted form; e-mail can be decrypted by clicking "DECRYPT" button on the *read messages* form.

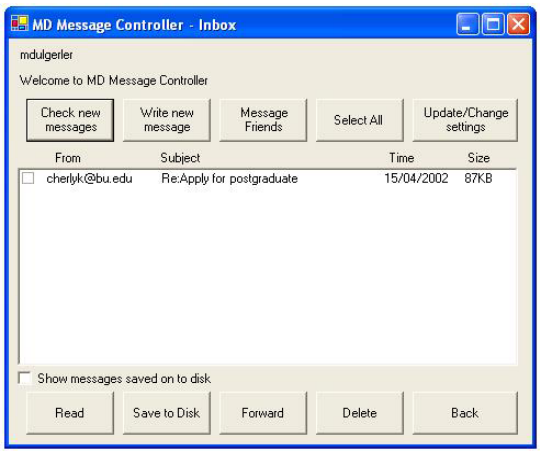

Figure 2. Received Messages Window

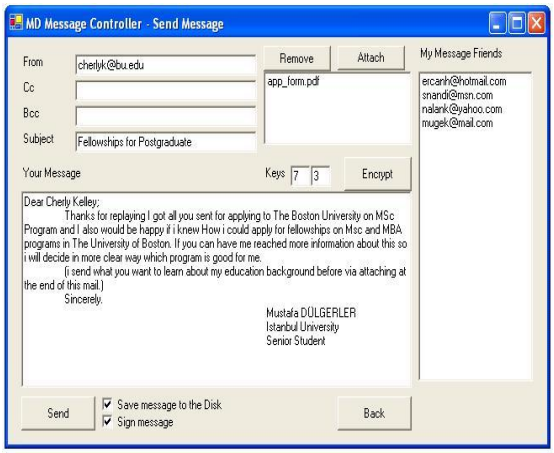

Figure 4. Send Message Window

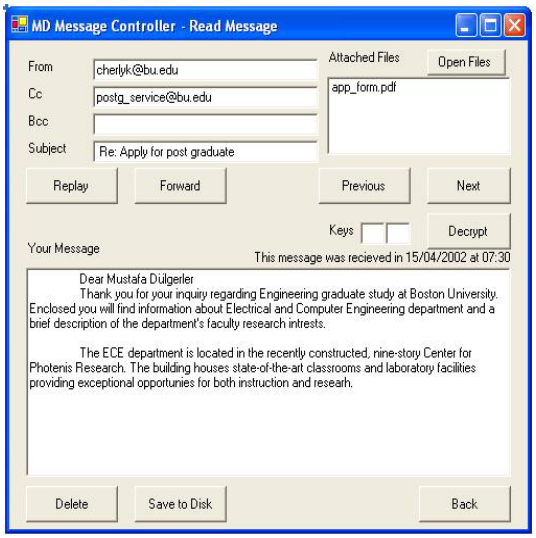

Figure 3. Read Messages Window

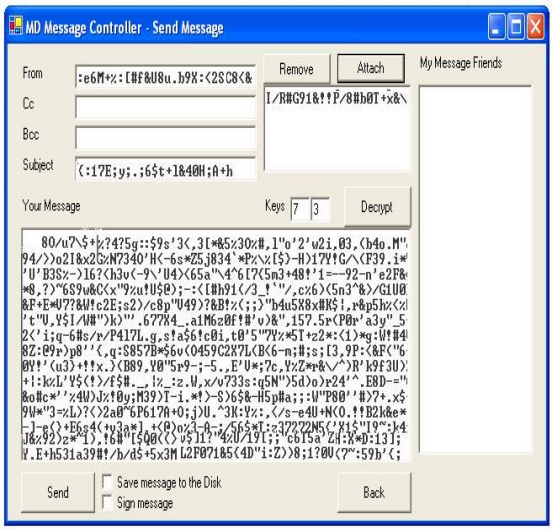

Figure 5. Encrypted Message

4.3. Send Message Window

When e-mail is sent, after the message typing is completed, if "ENCRYPT" button is clicked, message is encrypted with ElGamal Algorithm. If "SEND" button is clicked, encrypted message is sent to receiver (Figure 4). But first two key values have to be defined. Encrypted message is shown in Figure 5.

4.4. Quit Window

In quit menu (Figure 6), all connection is disconnected. Receiver has to have the same program to decrypt e-mail which encrypted and sent by this program. It is not necessary to know key values before. Because the key values are encrypted then attached encrypted message.

Figure 6. Quit Window

5. RESULT AND FUTURE PLANS

This project was built by using ElGamal Encryption algorithm and a unique hash algorithm. Performance of software depends on hardware used. A simple hash algorithm was used to make it easy to understand. This structure was planned using of 16-bit values. If this value is increased, The performance of software might be decreased, but security level will increase. ElGamal Encryption Algorithm was used 8 bit on this application. Security level also increases by increasing this value. Difference of this application form other e-mail applications, is that it has encryption and decryption processes. User can perform all process in other e-mail applications with this application.

Analysis based on the best available algorithms for both factoring and discrete logarithms shows that RSA and ElGamal have similar security for equivalent key lengths. [7,8,9].

We aim to add a span dictionary to the application for next step. Thus after encryption process is completed, a word in span dictionary will be found for each encrypted character then changed. This will reduce system performance and make length of message longer but security will reach highest level. To reduce length of message, compress algorithms can be applied.

REFERENCES

[1] Stinson D. R., Cryptography Theory and Practice, *CRC Press*, 1995, Florida.

[2] William S., Network And Internetwork Security Principles And Practice, *Prentice-Hall,* Inc. 1995, New Jersey.

[3] http://www.idssoftware.com/jsecure.html

[4]http://www.d.connect.ti.com/dsp/tpcat/tpcodec.nsf/SoftwareForExternal/9BEAA0765A00DA4862569F2005645E1

[5] http://www.soriac.de

[6] http://www.cryptome.org

[7] Sertbaş A.and Sarısakal M. N, "VMAIL / An Application For A Secure E-Mail Transmission Using Encrypting Techniques" *ELECO'2001 International Conference on Electrical and Electronics Engineering*, 7 - 11 November 2001, Bursa, TURKEY, Proceedings, pp. 363-367,2001.

[8]. Sarısakal M. N., Sevgen S. and Acar D., "Developing An Application of RSA Algorithm With JAVA", *ELECO'2001 International Conference on Electrical and Electronics Engineering,* 7 - 11 November, Bursa, TURKEY, Proceedings, pp. 332-335,2001.

[9]. Sarısakal M. N., Savasan V., Sertbas A., "RSA ve IDEA Algoritmalarını Birlikte Kullanan Güvenli Bir E-Posta Uygulaması : VMAIL", *I.U. Journal of Electrical & Electronics*, Vol. 1, No. 2, pp 297-305, 2001.

Mustafa DÜLGERLER, who was born in 1980 in İstanbul, meet with programming first time in a GwBasic programing course when he was 9. with this programing love, he applied high scholl studied on computer technologies and graduated with first deggre.with the same success, he started Trakya University Computer Engineering Class in 1998. he transfered to İstanbul University Computer Engineering Class in 2000 and graduated with first deggre in July 2002. during his training, he worked in several computer related companies and worked as a trainer in a computer programming course, after graduated, he started to work as a trainee in Microsoft Turkey. Dülgerler had MCSE, MCSD.Net, and MCDBA certificates, is working on Microsoft .Net programming besides all, he was awarded from Junier Chamber as TOYP2002 The best technical software developer in September 2002. Recently he writes articals about Microsoft technologies on www.yazgelistir.com

M. Nusret SARISAKAL was born on the 7th of June 1971, in Istanbul, Turkey. He received both his B.Sc. and M.Sc degrees in Computer Science Engineering from the Faculty of Engineering at Istanbul University in 1997 and 2000 respectively. Between May 1995 and December 1998 he worked as an application programmer, technical coordinator and project administrator for an important software company. He worked as a research assistant from December 1998 to October 2001 in the Computer Science Engineering Department of Istanbul University. Since October 2001, he has been teaching computer science courses for the same department. His research interests include; Computer Networks, Communication, Web Programming, XML, EDI, VRML, Database Systems, Security and Cryptography. He is married and has one son.