Analyzing the Efficiency of Text-to-Image Encryption Algorithm

Ahmad Abusukhon
Computer Network Department
Al-Zaytoonah University of Jordan
Amman, Jordan

Mohammad Talib
Department of Computer Science
University of Botswana
Gaborone, BOTSWANA

Maher A. Nabulsi
Department of Computer Science
Al-Zaytoonah University of Jordan
Amman, Jordan

***Abstract*—Today many of the activities are performed online through the Internet. One of the methods used to protect the data while sending it through the Internet is cryptography. In a previous work we proposed the Text-to-Image Encryption algorithm (TTIE) as a novel algorithm for network security. In this paper we investigate the efficiency of (TTIE) for large scale collection.**

Keywords-Algorithm; Network; Secured Communication; Encryption & Decryption; Private key; Encoding.

I. Introduction

Cryptography or sometimes referred to as encipherment is used to convert the plaintext to encode or make unreadable form of text [1]. The sensitive data are encrypted on the sender side in order to have them hidden and protected from unauthorized access and then sent via the network. When the data are received they are decrypted depending on an algorithm and zero or more encryption keys as described in Fig.1.

Decryption is the process of converting data from encrypted format back to their original format [2]. Data encryption becomes an important issue when sensitive data are to be sent through a network where unauthorized users may attack the network. These attacks include IP spoofing in which intruders create packets with false IP addresses and exploit applications that use authentication based on IP and packet sniffing in which hackers read transmitted information.

One of the techniques that are used to verify the user identity (i.e. to verify that a user sending a message is the one who claims to be) is the digital signature [3]. Digital signature is not the focus of this research.

There are some standard methods which are used with cryptography such as private-key (also known as symmetric, conventional, or secret key), public-key (also known as asymmetric), digital signature, and hash functions [4]. In private-key cryptography, a single key is used for both encryption and decryption. This requires that each individual must possess a copy of the key and the key must be passed over a secure channel to the other individual [5]. Private-key algorithms are very fast and easily implemented in hardware;therefore, they are commonly used for bulk data encryption.

The main components of the symmetric encryption include - plaintext, encryption algorithm, secret key, ciphertext and decryption algorithm. The plaintext is the text before applying the encryption algorithm. It is one of the inputs to the encryption algorithm. The encryption algorithm is the algorithm used to transfer the data from plaintext to ciphertext. The secret key is a value independent of the encryption

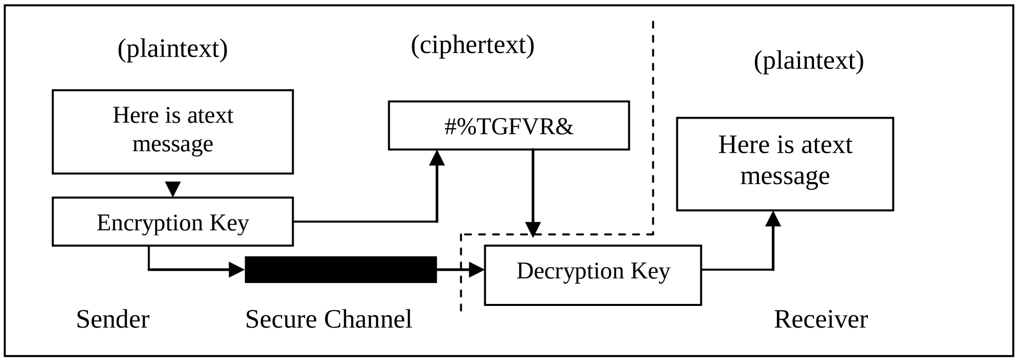


Figure 3. Encryption and Decryption methods with a secure channel for key exchange.

algorithm and of the plaintext and it is one of the inputs of the encryption algorithm.The ciphertext is the scrambled text produced as output. The decryption algorithm is the encryption algorithm run in reverse [6, 2, 7]. There are two main categories of private-key algorithms, namely block and stream encryption [8].

Public-key encryption uses two distinct but mathematically related keys - public and private. The public key is the non-secret key that is available to anyone you choose (it is often made available through a digital certificate). The private key is kept in a secure location used only by the user. When data are sent they are protected with a secret-key encryption that was encrypted with the public key. The encrypted secret key is then transmitted to the recipient along with the encrypted data. The recipient will then use the private key to decrypt the secret key. The secret key will then be used to decrypt the message itself. This way the data can be sent over insecure communication channels [6].

II. RELATED WORK

Bh, Chandravathi, and Roja [9] proposed encoding and decoding a message in the implementation of Elliptic Curve Cryptography is a public key cryptography using Koblitz's method [10, 11]. In their work, each character in a message is encoded by its ASCII code then the ASCII value is encoded to a point on the curve. Each point is encrypted to two ciphertext points. Our work differs from their work. In their work they used public-key technique whereas in our work we use private key technique. They encoded each character by its ASCII value but we encode each character by one pixel (three integer values - R for Red, G for Green and B for Blue).

Singh and Gilhorta [5] proposed encrypting a word of text to a floating point number that lie in range 0 to 1. The floating point number is then converted into binary number and after that one time key is used to encrypt this binary number. In this paper, we encode each character by one pixel (three integer values R, G and B).

Kiran Kumar, MukthyarAzam, and Rasool [12] proposed a new technique of data encryption. Their technique is based on matrix disordering which was relied on generating random numbers used for rows or columns transformations. In their work, the original plaintext was ordered into a Two-directional circular queue in a matrix A of order m x n. A number of column and row transformations were carried-out on the matrix and to do so a random function was used to generate positive integer say X and then X is converted to a binary number. Rows or columns transformation was made based on the values of the individual bits in the binary number resulted from the X value.

Another random number was generated in order to determine the transformation operation. The random number was divided by three (as we have three types of transformation operations) and the modulus (0, 1, or 2) was used to determine the operation type. The operation type could be 0 (means circular left shift), 1 (means circular right shift) and 2 (means reverse operation on the selected rows). In case rows were selected to perform a transformation operation (the selection was made depending on the bit value of X) two random numbers r1 and r2 were generated where r1 and r2 represent two distinct rows. Another two random numbers were generated c1 and c2 that represent two distinct columns. The two columns c1 and c2 were generated in order to determine the range of rows in which transformation had to be performed. After the completion of each transformation a sub-key was generated and stored in a file key which was sent later to the receiver to be used as decryption key. The sub-key format is T, Op, R1, R2, Min, and Max, where:

T: the transformation applied to either row or column

Op: the operation type coded as 0, 1, or 2, e.g., shift left array contents, shift right array contents, and reverse array contents.

R1 and R2: two random rows or columns

Min, Max: minimum and maximum values of a range for two selected rows namely, R1 and R2.

Abusukhon and Talib [13] proposed a novel data encryption algorithm called Text-to-Image Encryption algorithm (TTIE) in which a given text is encrypted into an image. Each letter from the plaintext is encrypted into one pixel. Each pixel consists of three integers and each integer represents one color (for example, Red, Green, and Blue). Each color has a value in the range from 0 to 255. The private key is generated randomly by creating three random integers (one pixel) for each letter. For example, letter "A" may be represented by the RGB value (0, 7, 0). The whole letters of a given text is transformed into a two dimensional array of pixels say M, then M is shuffled a number of times by performing row and column swapping. In their work, they analyzed the TTIE algorithm by calculating the number of possible permutations to be guessed by hackers. The TTIE algorithm is described in Fig. 2. In their work, they mentioned that the TTIE algorithm is useful for text encryption for individual offline machines, a network system, and for e-mail security. They proposed the TTIE algorithm for e-mail security since the text messages in the mail box appear as images making it difficult for others to guess the plaintext messages (i.e. the original messages sent from the other side).

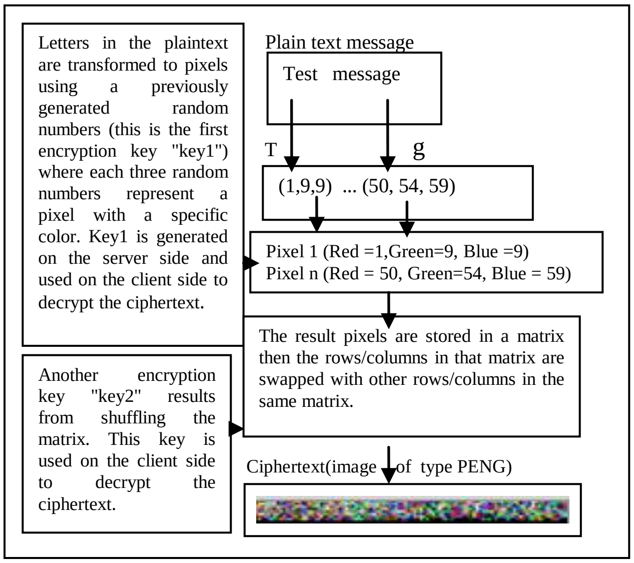

Letters in the plaintext are transformed to pixels using a previously generated random numbers (this is the first encryption key "key1") where each three random numbers represent a pixel with a specific color. Key1 is generated on the server side and used on the client side to decrypt the ciphertext.
Plain text message
Test message
T
g
(1,9,9) ... (50, 54, 59)
Pixel 1 (Red =1,Green=9, Blue =9)
Pixel n (Red = 50, Green=54, Blue = 59)
The result pixels are stored in a matrix then the rows/columns in that matrix are swapped with other rows/columns in the same matrix.
Another encryption key "key2" results from shuffling the matrix. This key is used on the client side to decrypt the ciphertext.
Ciphertext(image of type PENG)

Figure 2. The Text-to-Image Encryption Algorithm

In this paper, we propose to investigate the efficiency of the TTIE algorithm when various memory sizes and various data sizes are used.

III. Our Experiments

Abusukhon and Talib [13] tested the TTIE algorithm for a few documents of size 512 KB. In this paper, we propose to investigate the efficiency of the TTIE algorithm when different data sizes are used while fixing the memory size to 250MB. In addition, we propose to investigate the effect of using various memory sizes on the performance of the TTIE algorithm while fixing the data collection size to 1.96 Gigabytes. We use Java NetBeans as a vehicle to carry out our experiments.

A. *Machine Specifications*

Our experiments are carried out on a single machine with the following specifications; processor Intel (R) core (TM)2, Duo CPU T5870 @ 2.00GHz, installed memory (RAM) 2.00GB operating system Windows 7 professional and hard disk 25.2 GB (free space).

B. *Data Collection*

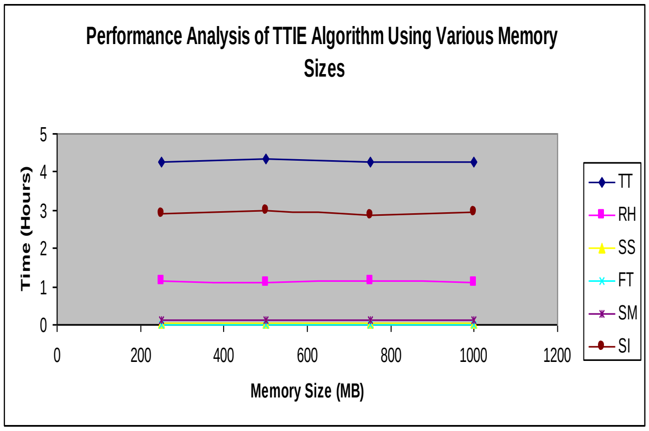
We build the data collection for our experiments by writing a simple Java code. This code is used to generate documents of different sizes. This is done by setting the length of the word and the number of words in each document. The letters in each document are chosen randomly from an array that contains all letters from "A" to "Z". For our experiments, we set the word length to 7 and the number of words in each document to 30.

C. *Investigate the Effect of Memory Size on Performance*

In this section, we investigate how the TTIE algorithm is affected by the memory size. To do so, we divide the time of the TTIE algorithm into five basic times namely, the time required for reading the data collection from the hard disk (RH), the time of the switch statement -switch statement is used to transfer the letters in a given text into pixels and store them into a single dimension array- (SS), the time required to fill the array of pixels into a two dimension array – filling pixels in a two dimensional array is necessary to perform matrix scrambling- (FT), the time required for scrambling the matrix (SM), the time required to store the result image (the encrypted text) on the hard disk (SI). We set the collection size in all experiments carried out in this section to 1.96 GB. We calculate the total time (TT) of each experiment as well as the five basic times mentioned above (i.e. RH, SS, FT, SM, and SI). We investigate how the TTIE is affected by various memory sizes 250MB, 500MB, 750MB, and 1000MB as described in Fig. 3.

Fig. 3, shows that the most dominant time of the TTIE algorithm (with respect to the TT time) is the SI time. The average of the total time (TT) for all experiments carried out in this section is 4.289. However, the average of the SI time for all experiments carried out in this section is 2.922. Thus the SI time is 0.681 of the TT time. In addition, Fig. 3 shows that the TT time and the SI time are slightly affected by changing the memory size.

Figure 3. The performance of the TTIE algorithm using various memory



sizes

D. *Investigate the Effect of Data Size on Performance*

In this section, we investigate how the performance of the TTIE algorithm is affected by various data sizes while fixing the memory size to 250MB. We use different data collection sizes; 118MB, 236MB, 354MB, 473MB, 591MB, 709MB, 827MB, 946MB, 1.03GB, and 1.15GB. Fig. 4 describes the results of these experiments.

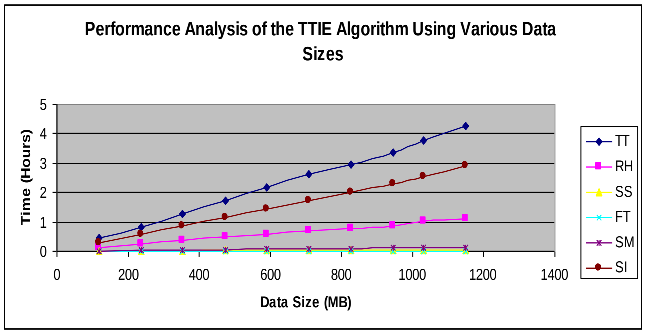


Figure 4. The performance of the TTIE algorithm using various data sizes

Our results show that the most dominant time of the TTIE algorithm is the SI time.The average of the total time (TT) for all experiments carried out in this section is 2.337. However, the average of the SI time for all experiments carried out in this section is 1.577. This means that most of the TTIE time is spent on saving the images on the hard disk.

IV. Conclusion And Future Work

In this paper, we investigated the efficiency of the TTIE algorithm. The TTIE algorithm is good for text encryption for a network system (TTIE is good for a Virtual Private Network, VPN, where encrypted data are sent and received across shared or public networks), individual offline machines, and e-mail security. We investigated the efficiency of the TTIE algorithm when various memory sizes and various data sizes are used. The results from our experiments showed that the most dominant time is the time required to save the encrypted data (images) on the hard disk (this time is called SI). In addition, the results from our experiments showed that the SI time, SM time, FT time, SS time, RH time, and the TT time are very slightly changed when the memory size is increased. In addition, the results from this work showed that the SI time, TT time, and the RH time are greatly increased when the size of the data collection is increased. The SS time, FT time, and the SM time are slightly changed when the size of the data collection is increased. In future, we propose to reduce the time required for saving the encrypted data (images) on the hard disk using distributed computing.

We propose to use a large data collection size (multiple Gigabytes) and investigate distributing the data collection among a number of nodes working together with a server. All nodes work in parallel to encrypt and store the large data collection on the hard disk.

Acknowledgment

Heartfelt gratitude to Al-Zaytoonah Private University of Jordan for their help and for their financial support to carry out this work successfully.

References

[1] K. Lakhtaria, "Protecting computer network with encryption technique: A Study". International Journal of u- and e-service, Science and Technology, vol. 4, 2011.

[2] A. Chan, "A Security framework for privacy-preserving data aggregation in wireless sensor networks". ACM transactions on sensor networks, vol.7, 2011.

[3] M. Savari, and M. Montazerolzohour, "All About encryption in smart card". Proceeding of International conference on Cyber Security, Cyber Warfare and Digital Forensic (Cyber Sec), 2012.

[4] B. Zaidan, A. Zaidan, A. Al-Frajat, and H. Jalab, "On the differences between hiding information and cryptography techniques: An Overview". Journal of Applied Sciences vol. 10, 2010.

[5] A. Singh, and R. Gilhorta, "Data security using private key encryption system based on arithmetic coding". International Journal of Network Security and its Applications (IJNSA), vol. 3, 2011.

[6] W. Stalling, Cryptography and network security principles and practices ,4th ed. Prentice Hall. [online] at: http://www.filecrop.com/cryptography-and-network-security-4th-edition.html, Accessed on 1-Oct-2011.

[7] V. Ggupta, G. Singh, and R. Gupta, "Advance cryptography algorithm for improving data security". vol. 2, 2012. [online] at: http://www.ijarcsse.com/docs/papers/january2012/V2I1055.pdf, Accessed on 19-Nov-2012.

[8] S. Suliman, Z. Muda, and J. Juremi, "The New approach of Rijndael Key schedule". Proceeding of International conference on Cyber Security, Cyber Warfare and Digital Forensic (Cyber Sec), 2012.

[9] P. Bh, D. Chandravathi, and P. Roja, "Encoding and decoding of a message in the implementation of Elliptic Curve cryptography using Koblitz's method". International Journal of Computer Science and Engineering,vol.2,2010.

[10] N. Koblitz, "Elliptic Curve cryptosystems", Mathematics of Computation",vol.48,1987,pp.203-209.

[11] N. Koblitz, A Course in Number Theory and cryptography. 2'nd ed. Springer-Verlag, 1994.

[12] M. Kiran Kumar, S. Mukthyar Azam, and S. Rasool, "Efficient digital encryption algorithm based on matrix scrambling technique". International Journal of Network Security and its Applications (IJNSA), vol.2,2010.

[13] A.Abusukhon, and M. Talib, "A Novel network security algorithm based on Private Key Encryption". In Proceeding of The International Conference on Cyber Security, Cyber Warfare and Digital Forensic (CyberSec12), 2012.