Data Encryption by Using Vigenere Algorithm with Steganographic Technique

Htike Ayar Hlaing, Soe Soe Aye
University of Computer Studies, Kyainge Tong
htikeayarhlaing@gmail.com , soesoeaye74@gmail.com

Abstract

In the current computing community, secure data transfer is limited due to its attack made on data communication. Solutions which came to the rescue are cryptography and steganography. Cryptography is often used in situations where the existence of the message is clear, but the meaning of the message is obscured. In particular, the sender transforms the message into a form that only the intended recipient of the message can decrypt and read. Steganography is often used in situations where the actual existence of the message needs to be obscured. This paper is intended to ensure secure data transfer between the source and destination by providing double layer of protection using both cryptography and steganography. To implements both cryptography and steganography, the message was encrypted before it is hidden inside a cover image. In this paper, the encryption was done by using vigenere encryption method and the ciphered message was embedded by using LSB algorithm.

1. Introduction

Cryptography is a technique used to hide the meaning of a message and Steganography is a technique used to hide the existence of a message. If a message were to fall into the hands of the wrong person, cryptography should ensure that message could not be read. Typically, the sender and receiver agree upon a message scrambling protocol beforehand and agree upon methods for encrypting and decrypting messages. Cryptography is concerned with the writing (ciphering or encoding) and deciphering (decoding) of messages in secret code. It is divided into two implementation techniques and those include transposition and substitution. *A transposition cipher* is an encoding process that does not change any of the letters of the original message but changes the position of the letters. *A substitution cipher* is an encoding process that maintains the order of the letters in the message but changes their identity [5]. Each letter of the message is replaced by another letter or symbol. A polyalphabetic cipher is any cipher based on substitution, using multiple substitution alphabets. The Vigenère cipher is probably the best-known example of a polyalphabetic cipher, though it is a simplified special case. The cryptographic algorithm used in our implementation is expressed in section 2.

Digital steganography exploits the use of a host data or message (also known as a container) to hide or embed another data or message into it. Unlike encryption, the host data or container used in steganography is not scrambled or hidden during the communication process. Detail expression of steganographic used in our implementation is mention in section 3.

It is possible to combine Cryptography and Steganography together to achieve a higher level of security. Combining these two primitives should both hide the meaning of a message as well as conceal the physical message. If the message is intercepted, it can be blocked but not read.

2. Types of Cryptographic Algorithms

There are several ways of classifying cryptographic algorithms. In this paper, cryptographic algorithms were categorized based on the number of keys that are employed for encryption and decryption. A cryptographic algorithm transforms cryptographic key and readable (plaintext) data into cipher text that can only be understood by applying another (possibly the same) cryptographic key and crypto algorithm to it. If the algorithm involves two different keys, one for enciphering and the other for deciphering, it is called an asymmetric or public-key algorithm. If the keys and algorithms are the same, it is called a symmetric or secret-key crypto system. Since we implemented the secret-key crypto system, we omitted about asymmetric or public-key algorithm in this paper [3].

2.1 Secret Key Cryptography

With secret key cryptography, a single key is used for both encryption and decryption. The sender uses the key to encrypt the plaintext and sends the cipher text to the receiver. The receiver applies the same key to decrypt the message and

recover the plaintext. Because a single key is used for both functions, secret key cryptography is also called symmetric encryption. Among the various secret-key cryptography algorithms, we have applied Vigenère cipher method.

(i) Vigenere Cipher Method.

The **Vigenere cipher** is a method of encrypting alphabetic text by using a series of different Caesar ciphers based on the letters of a keyword. It is a simple form of polyalphabetic substitution. The Vigenère square or Vigenère table, also known as the tabula recta, can be used for encryption and decryption. The Vigenère cipher consists of several Caesar ciphers in sequence with different shift values.

To encipher, a table of alphabets can be used, termed a tabula rectum, Vigenère square, or Vigenère table. We used 256 ASCII character in our implementation. So, our table of alphabets have 256 rows and 256 columns. 0 to 255 ASCII characters code are enough to encrypt and decrypt all of English symbols. Although Vigenère cipher method is ancient and has some weakness, we apply it to acquire simple combination with steganographic method

Following example consists of the alphabet written out 26 times in different rows, each alphabet shifted cyclically to the left compared to the previous alphabet, corresponding to the 26 possible Caesar ciphers. At different points in the encryption process, the cipher uses a different alphabet from one of the rows. The alphabet used at each point depends on a repeating keyword. The Vigenere cipher is a polyalphabetic cipher based on using successively shifted alphabets, a different shifted alphabet for each of the 26 English letters. The procedure is based on the tableau shown in Figure 1 and the use of a keyword. The letters of the keyword determine the shifted alphabets used in the encoding process.

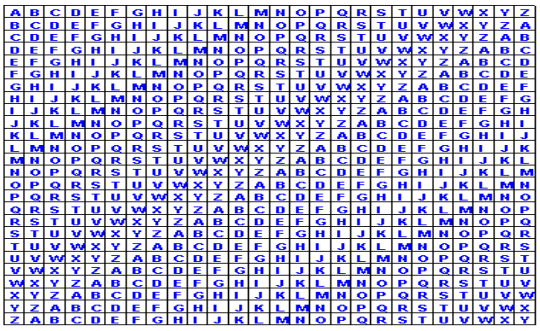

Figure 1. Vigenere tableau example

(ii) Vigenere Cipher Example.

For example, to encrypt the message COMPUTING GIVES INSIGHT with the keyword LUCKY. Vigenère proceed by repeating the keyword as many times as needed above the message as follows.

Figure 2. Message and Key example

The Vigenere tableau in Figure 1 can be used directly. For each letter of the message use the letter of the keyword to determine a row and go across the row to the column headed by the corresponding letter of the message. Figure 3 is illustration of the row-column look-up for the first two letters of the message. The red arrows indicate the encoding of the first letter C and the green arrows indicate the encoding of the second letter O. It follows that the first two letters "CO" in the message are encoded as "NI".

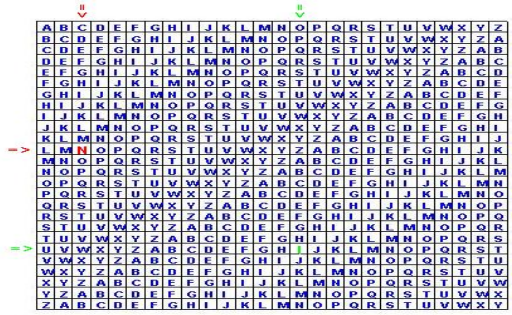

Figure 3. Vigenere row-column look-up

Continuing in this way using, the encoded message was appeared as in Figure 4.

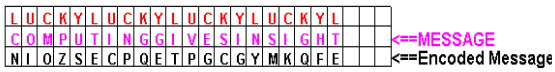

Figure 4. Vigenere encoding example

3. Steganography

Steganography can be used anytime you want to hide data. There are many reasons to hide data, but they all boil down to the desire to prevent unauthorized persons from becoming aware of the existence of a message [3]. In the business world steganography can be used to hide a secret chemical formula or plans for an invention. Steganography can also be used for corporate espionage by sending out trade secrets without anyone at the company being any the wiser. Steganography can also be used in the non-commercial sector to hide information that someone wants to keep private.

3.1 Steganographic Embedding Methods

A few ways exist to hide information in digital images. Common approaches include:

- Least significant bit (LSB) insertion.
- Masking and filtering.
- Algorithms and transformations.

We used LSB embedding method in our experimentation and other embedding methods are omitted in this paper. Least significant bit (LSB) insertion is a common, simple approach to embedding information in a cover image [4].

3.2 LSB Insertion Method

The least significant bit (in other words, the 8th bit) of some or all the bytes inside an image is changed to a bit of the secret message. When using a 24-bit image, a bit of each of the red, green and blue color components can be used, since they are each represented by a byte [2].

In other words, one can store 3 bits in each pixel. An 800 × 600-pixel image, can thus store a total amount of 1,440,000 bits or 180,000 bytes of embedded data [19]. For example, a grid for 3 pixels of a 24-bit image can be as follows:

(00101101 00011100 11011100)
(10100110 11000100 00001100)
(11010010 10101101 01100011)

When the number 200, which binary representation is 11001000, is embedded into the least significant bits of this part of the image, the resulting grid is as follows:

(00101101 00011101 11011100)
(10100110 11000101 00001100)
(11010010 10101100 01100011)

Although the number was embedded into the first 8 bytes of the grid, only the 3 underlined bits needed to be changed according to the embedded message. On average, only half of the bits in an image need to be modified to hide a secret message using the maximum cover size [6]. Since there are 256 possible intensities of each primary color, changing the LSB of a pixel results in small changes in the intensity of the colors. These changes cannot be perceived by the human eye - thus the message is successfully hidden. With a well-chosen image, one can even hide the message in the least as well as second to least significant bit and still not see the difference [1].

3.3 LSB Embedding and Extracting Example

In this section we illustrate step by step embedding and extracting on 24-bit PNG image which use a byte each for the red (R), green (G), and blue (B) channels.

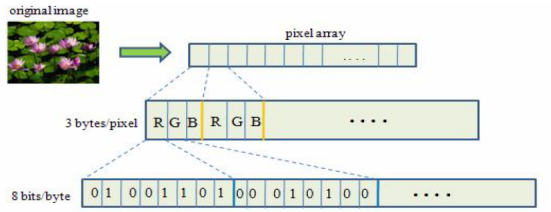

Figure 5. Accessing the Bits of a PNG Image

Figure 5 shows the first stage of the process, when the image data is accessed as a series of bytes. Depending on the image format, a pixel may be represented by one or more bytes.

The next stage is to read in the text file, and access its bits, as shown in Figure 6

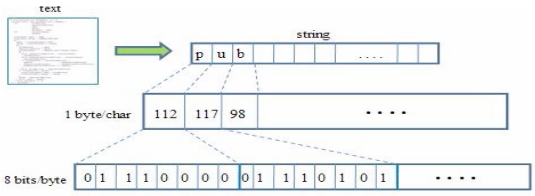

Figure 6. Accessing the Bits of a Text File

Now it is time to insert the bits of the text file into the image. The LSB approach only modifies the least significant bit of each image byte, as illustrated by Figure 7.

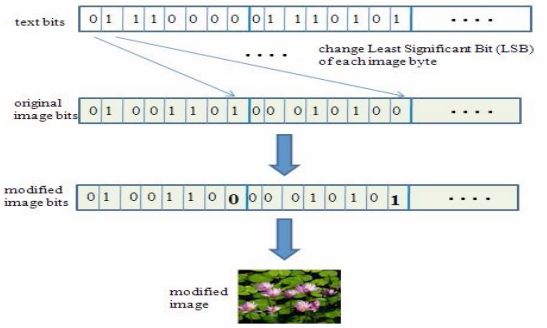

Figure 7. Inserting the Text Bits into the Image

Extracting the text from the image later involves copying the LSBs of the modified image's bytes, and recombining them into bytes in a text file, as in Figure 8.

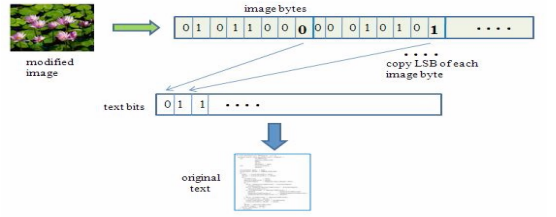

Figure 8. Extracting the Text from the Modified Image

To implement LSB method for hiding text inside a PNG image, the length of the text in binary form is calculated beforehand and hidden in the image before the text.

In other words, steganographic information has two parts: the size of the binary message, followed by the message itself. The size information is utilized when the text is extracted from the image, so the extraction process knows when to stop. Hidden message extraction is done in two stages –first the size of the binary message is read from the image bytes. Second the length is used to constrain the reading of the rest of the image, so that only the message is retrieved.

4. Implementation

In our implementation the message was encrypted before it is hidden inside a *cover* message. This provides a double layer of protection. To begin with, encryption may make the existence of the message even more difficult to detect, due to the fact that some encryption techniques cause the patterns of the characters in the encrypted version to be more random than in the original version. In addition, even if the existence of the encrypted message is detected, it is unlikely that an eavesdropper can read the message. Since this system is layered as cryptographic and steganographic it includes two stages.

Firstly, the original text, or *plaintext*, is converted into a coded equivalent called *cipher text* via an encryption algorithm. Only those who possess a secret key can decipher (*decrypt*) the cipher text into plaintext. In this ciphering stage include three components such as plaintext *P*, key *K* and cipher text *E*.

Secondly, to be embedded data, it is required two files. The first is the innocent-looking image that holds the hidden information, called the cover image. The second file is the message - the information to be hidden. In this paper, the generated cipher text in first stage is embedded in a bit stream. In order to do this process, three components are included as cover image *C*, secret message *M* and stego image *S*.

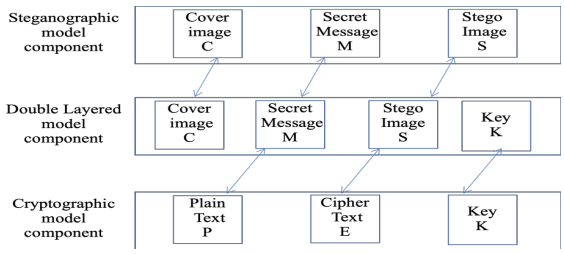

Figure 9. Unified Model Components

To make double layered system, these two stages need to be unified their components as one model. The unified components are illustrated in Figure 9. And double layered model is illustrated in Figure 10.

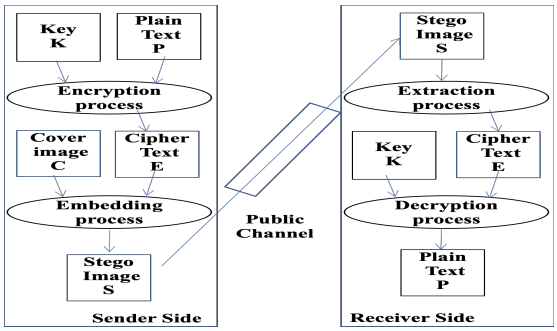

Figure 10. Double Layered Model

In our implementation sender must encrypt the plaintext message before it is hidden inside a cover image. To begin encryption sender

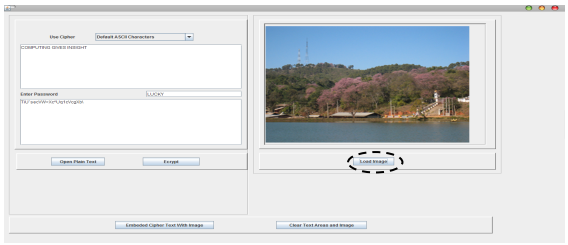

Figure 11. Encryption and Embedding Processes

can choose plaintext file or typing message in plaintext area. Then sender must input the password or the key which already agree with receiver. After filling message and password, he must press encrypt button. Then our system encodes the message into ciphertext. After ciphering process sender must load an image, he desired by pressing load image button. Now, sender can embed the ciphertext to the loaded image by pressing embedded cipher text with image button. These processes are illustrated in Figure 11.

When receiver gets the embedded image file, he/she need to extract the image to ciphertext. For that purpose, user must enter the system from receiver part. Firstly, the receiver must load the

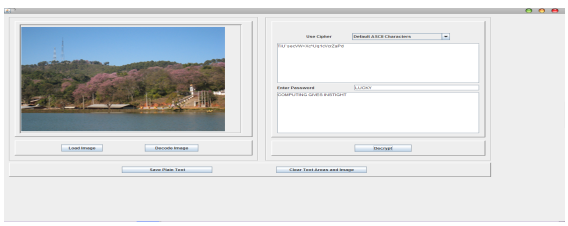

Figure 12. Decryption and Extraction Processes

embedded image file. After loading embedded image, to extract the image user press decode image button. If the loaded image has hiding message the system response and ciphertext appear in encrypted cipher text area. After extraction, user needs to put agreed password in password area and press decrypt button. Then the secret message is appeared. These processes are illustrated in Figure 12.

5. Conclusion

In this paper, we have reviewed the Vigenère cipher method for encrypt a message and applied LSB algorithm for embedding data to an image. By combining cryptography and steganography, this system provides a simple and efficient method for hiding the data from hackers and sent to the destination in a safe manner. Since Encryption and Decryption techniques have been used, it will make the security system robust. Finally, this system implements double layered security to ensure secure data transfer between the source and destination.

6. References

[1] Anderson, R., (ed.): Information hiding first international workshop, Cambridge, UK. Lecture Notes in Computer Science, Vol. 1174. Springer-Verlag, Berlin Heidelberg New York (1996).

[2] Anderson, R., Petitcolas, F.: On the Limits of Steganography, IEEE Journal on Selected Areas in Communications, Vol. 16, No. 4, May (1998) 474–481.

[3] Bender, W., Gruhl, D., Morimoto, N., Lu, A.: Techniques for Data Hiding. IBM Systems Journal Vol. 35, No. 3&4. MIT Media Lab (1996) 313–336.

[4] Fridrich, J., Goljan, M., and Hogea, D. (2002). Steganalysis of jpeg images: Breaking the f5 algorithm. In *Proc of in 5th International Workshop on Information Hiding*.

[5] Johnson, N. F. and Jajodia, S. (1998). Exploring steganography: Seeing the unseen. *Computer*, 31(2):26–34.

[6] Kerckhoffs, A. (1883). La cryptographie militaire. *Journal des Sciences Militaries*, 9th series (IX):5–38.

[7] Kharrazi, M., Sencar, H. T., and Memon, N. (2004). Image steganography: Concepts and practice. In *WSPC Lecture Notes Series*.