Data Security System based on Cryptography and Steganography

Aye Aye Nwe, Myat Su Win
Computer University, Taung Ngu, Myanmar
ayeayenwe2012@gmail.com, myattsuro@gmail.com

Abstract

Data security system is implemented by using two methods: cryptography and steganography. The effect of these two methods enhances the security of the data and intended to hide and cover the messages within one of the standard media, images. Encryption the data with a cryptography algorithm called Sosemanuk stream cipher. This encryption process is done with help of crypto-key, and the detection or reading of encrypted information is possible only having the key. After encryption, a part of the encrypted message (cipher) is hidden in Least Significant Bit of an image file using steganography algorithm. The technique used in this field replaces the LSB of image pixels with intended secret bits. The system can be used as a part to protect e-mail messages, credit card information, corporate data, etc.

Keywords: Cryptography, Steganography, Data Security, Sosemanuk Stream Cipher, LSB

1. Introduction

The network Security is becoming more important as the amount of data being exchanged on the Internet is increasing. People have desired to keep certain sensitive communications secret for thousands of years. In this age of universal electronic connectivity, of viruses and hackers, of electronic eavesdropping and electronic fraud, there is indeed a need to protect information from passing before curious eyes or, more importantly, from falling into wrong hands. Thus, multimedia security is much to consider in distributing digital information safety.

Cryptography and steganography are well known and widely used techniques that manipulate information (messages) in order to cipher or hide their existence. These techniques have many applications in computer science and other related fields: they are used to protect e-mail messages, credit card information, coporate data, etc. Several steganography systems combine steganography with cryptography seeking more security. Steganography is a technique of hiding information in digital media. A message in ciphertext, for instance, might arouse suspicion on the part of the recipient while an "invisible" message created with steganographic methods will not.

2. Related Work

Steganography and cryptography are two related fields. Several steganography systems combine steganography with cryptograpphy seeking more security. Domenico Bloisi and Luca Iocchi [4] described method for integrating together cryptography and steganography through image processing. This system presents that a system able to perform steganography and cryptography at the same time using images as cover objects for steganography and as keys for cryptography. Mohammed A.F. Al-Husainy [6] discussed the system is to use enough number of bits from each pixel in an image (7-bits in this study) to map them to 26 alphabetic English characters ('a'…'z') with some special characters that are mostly using in writing a secret message.

3. Background Theory

3.1. Cryptography

Cryptography is a science of protecting information by encoding it into an unreadable format [2]. A cryptographic algorithm, or cipher, is a mathematical function used in the encryption and decryption process. The security of encrypted data is entirely dependent on two things: the strength of the cryptographic algorithm and the secrecy of the key.

A key is a value that works with a cryptographic algorithm to produce a specific ciphertext. Key size is measured in bits; the number representing a 1024-bit key is darn huge. Key is a mathematical value, formula, or process that determines how a plaintext message is encrypted or decrypted [1]. As a subset of cryptography, cryptographic algorithms can be divided into two categories:

Block Cipher Algorithms – A block cipher processes the input one block of elements at a time, producing an output block for each input block.

Stream Cipher Algorithms – A stream cipher processes the input elements continuously, producing output one bit or one byte element at a time, as it goes along [4].

3.2. Steganography

Steganography is a way of protecting information, similar to cryptography and watermarking. It includes vast array of secret communication method that conceals message very existence. Digital images, videos, sound files, and other computer files that contain perceptually irrelevant or redundant information can be used as "cover" or carriers to hide secret messages.

The basic model of steganography consists of Carrier, Message, Embedding algorithm and Stego-key. Carrier is also known as a cover-object or stego-object, which embeds the message. Message is the data that the sender wishes to remain it confidential. Password is known as a stego-key, which ensures that only the recipient who knows the message from a cover-object [8]. There have been many techniques for hiding messages in images in such a manner that the alterations made to the image are perceptually indiscernible. Common approaches are including [7]:

(i) Least significant bit insertion (LSB)
(ii) Masking and filtering
(iii) Transform techniques

Least significant bits (LSB) insertion is a simple approach to embedding information in image file. The simplest steganographic techniques embed the bits of the message directly into least significant bit plane of the *cover-image* in a deterministic sequence.

Masking and filtering techniques, usually restricted to 24 bits and gray scale images, hide information by marking an image, in a manner similar to paper watermarks. The techniques performs analysis of the image, thus embed the information in significant areas so that the hidden message is more integral to the cover image than just hiding it in the noise level.

Transform techniques embed the message by modulating coefficients in a transform domain, such as the Discrete Cosine Transform (DCT) used in JPEG compression, Discrete Fourier Transform, or Wavelet Transform. These methods hide messages in significant areas of the cover-image, which make them more robust to attack [7].

3.3. Cryptography Vs Steganography

While cryptography and steganography are related, there is a difference between the two. Cryptography hides the contents of a secret message from a malicious people, whereas steganography even conceals the existence of the message. Cryptography is used to scramble messages so that they cannot be understood. It does not hide the fact that the message exists. Steganography, on the other hand, conceals the fact that the message exists by hiding the actual message in another [10].

4. The Proposed System

The proposed system presents data hiding process with respect to both steganography and cryptography. Firstly, a system describes that data hiding has cryptographic technique for security, by providing that it is equivalent to Sosemanuk stream cipher, and then to have security more, least significant bit algorithm has been used in the steganography section. The following figure shows that overview of the system with cryptography and steganography techniques, which refers to information hiding and providing secrecy.

This cryptography algorithm takes the secret key as well as a public initialization vector (IV) as input, and outputs a stream of random-looking symbols, known as the keystream. To encrypt a data stream, one simply has to Exclusive-OR (XOR) the data symbols with the keystream.

In Steganography, the encrypted data produced by cryptography is embedded in an image file. The function denoted by F in the Figure 1 represents the embedding and extracting function. While the output of the embedding process is a stego-image S, the inputs are the secret message C and cover image I. Once the embedding algorithm terminates, a Stego-image S obtained as similar as possible to cover image I. By extracting from the stego-image S, a file containing cipher text is obtained.

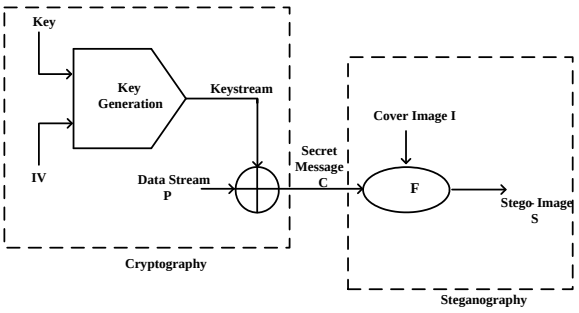

Figure 1 Overview of System

4.1. System Algorithm

In this proposed system, Sosemanuk algorithm uses in the part of cryptography. It is a type of stream cipher with a 128 to 256-bit key and a 128-bit Initial Vector (IV). This algorithm uses both some basic design principles from the stream cipher SNOW 2.0

and some transformations derived from the block cipher Serpent. It has a linear part that consists of a 320-bit Linear Feedback Shift Register (LFSR) and a Finite State Machine (FSM) including two 32-bit registers R1 and R2 [3].

The LFSR operates over elements of F_2^{32}. The initial state, at $t = 0$, entails the ten 32-bit values s_1 to s_{10}. At each step, a new value is computed, with the following recurrence:

$$s_{t+10} = s_{t+9} \oplus \alpha^{-1} s_{t+3} \oplus \alpha s_t, \ \forall t \geq 1$$

And the register is shifted [1].

The FSM takes as inputs some words from the LFSR state. It operates on the LFSR state at time $t \geq 1$ as follows:

$$FSM_t : (R1_{t-1}, R2_{t-1}, s_{t+1}, s_{t+8}, s_{t+9}) \longrightarrow (R1_t, R2_t, f_t)$$

Where

$$R1_t = (R2_{t-1} + \text{mux}(\text{lsb}(R1_{t-1}), s_{t+1}, s_{t+1} \oplus s_{t+8})) \bmod 2^{32}$$

$$R2_t = Trans(R1_{t-1})$$

$$f_t = (s_{t+9} + R1_t \bmod 2^{32}) \oplus R2_t$$

The internal transition function *Trans* on F_2^{32} is defined by

$$Trans(z) = (M \times z \bmod 2^{32}) <<< 7$$

where M is the constant value 0x54655307 and <<<denotes bitwise rotation of a 32-bit value (by 7 bits here) [1].

Sosemanuk Workflow - The Sosemanuk cipher combines the FSM and the LFSR to produce the output values z_t. Time $t = 0$ designates the internal state after initialization; the first output value is z_1. Once every four steps, four output values z_t, z_{t+1}, z_{t+2} and z_{t+3} are produced from the accumulated values f_t, f_{t+1}, f_{t+2}, f_{t+3} and s_t, s_{t+1}, s_{t+2}, s_{t+3}. Thus, Sosemanuk produces 32-bit values.

The first four iterations of Sosemanuk are as follows.

- The LFSR initial state contains values s_1 to s_{10}; no value s_0 is defined. The FSM initial state contains $R1_0$ and $R2_0$.
- During the first step, $R1_1$, $R2_1$ and f_1 are computed from $R1_0$, $R2_0$, s_2, s_9 and s_{10}.
- The first step produces the buffered intermediate values s_1 and f_1.
- During the first step, the feedback word s_{11} is computed from s_{10}, s_4 and s_1, and the internal state of the LFSR is updated, leading to a new state composed of s_2 to s_{11}.
- The first four output values are z_1, z_2, z_3 and z_4, and are computed using one application of Serpent1 over (f_4, f_3, f_2, f_1), whose output is combined by XORs with (s_4, s_3, s_2, s_1). The outputs of the FSM are grouped by four, and Serpent1 is applied to each group; the result is then combined by XOR with the corresponding dropped values from the LFSR, to produce the output values z_t:

$$(z_{t+3}, z_{t+2}, z_{t+1}, z_t) = \text{Serpent1}(f_{t+3}, f_{t+2}, f_{t+1}, f_t) \oplus (s_{t+3}, s_{t+2}, s_{t+1}, s_t)$$

To encrypt a data stream, one simply has to exclusive-or (XOR) the data symbols with the keystream. Decryption is of course the exact opposite, since the XOR operation is symmetric.

The structure of Sosemanuk is shown in Figure 2.

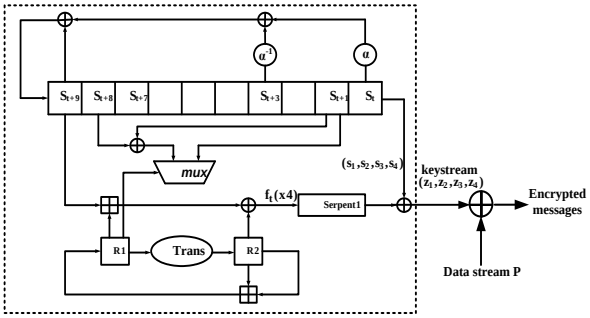

Figure 2. Block Diagram of Sosemanuk

4.1.1. Serpent and Derivatives

Serpent is a block cipher proposed as an AES candidate. From Serpent, it can define two primitives called Serpent 1 and Serpent 24.

Serpent 1 - Serpent 1 is one round of SERPENT, without the key addition and the linear transformation. SERPENT uses eight distinct S-boxes, numbered from S_0 to S_7 on 4-bit words.

Serpent 24 - Serpent 24 is SERPENT reduced to 24 rounds, instead of the 32 rounds of the full version of SERPENT, where the last round (the 24th) is a complete one and includes a complete round with the linear transformation (L) and an XOR with the 25th subkey. Thus, the last round equation is

$$R_{23}(X) = L(S_{23}(X \oplus K_{23})) \oplus K_{24}.$$

Serpent 24 uses only 25 128-bit subkeys, which are the first 25 subkeys produced by the SERPENT key schedule. The following steps are the workflow for the key schedule of Serpent 24 [3].

- IV Injection
- Key Mixing
- Linear Transformation
- S-boxes

Sosemanuk takes secret key as an input to key schedule of Serpent, to generate 25 128-bit subkeys. After the subkey generation, initial vector IV is taken as an input to Serpent 24. Then, intermediate data of rounds 12 and 18 of Serpent 24, and output data of round 24 of Serpent 24 are used as initial values of internal state [3].

These values are used to initialize the Sosemanuk internal state, with the following values:

$$(s_7, s_8, s_9, s_{10}) = (Y_3^{12}, Y_2^{12}, Y_1^{12}, Y_0^{12})$$

$$(s_5, s_6) = (Y_1^{18}, Y_3^{18})$$

$$(s_1, s_2, s_3, s_4) = (Y_3^{24}, Y_2^{24}, Y_1^{24}, Y_0^{24})$$

$$R1_0 = Y_0^{18}$$

$$R2_0 = Y_2^{18}$$

4.1.2. Serpent 24 Key Schedule

The key setup corresponds to the Serpent 24 key schedule, which produces 25 128-bit subkeys. The first expand the user supplied 256-bit key K to 33 128-bit subkeys $K_0,...,K_{32}$, in the following way. The key K is wrote as eight 32-bit words $w_{-8},...,w_{-1}$ and expand these to an intermediate key (prekey) $w_0,...,w_{131}$ by the following affine recurrence [9].

$w_i = (w_{i-8} \oplus w_{i-5} \oplus w_{i-3} \oplus w_{i-1} \oplus const.f \oplus i) <<<11$

where const.f is the fractional part of the golden ratio $(\sqrt{5}+1)/2$ or 0xe3779b9 in hexadecimal.

The round keys are now calculated from the prekeys using the S-boxes, again in bitslice mode and use the S-boxes to transform the prekeys w_i into words k_i of round key in the following way:

$(k_0, k_1, k_2, k_3) = S_3 (w_0, w_1, w_2, w_3)$
$(k_4, k_5, k_6, k_7) = S_2 (w_4, w_5, w_6, w_7)$
$(k_8, k_9, k_{10}, k_{11}) = S_1 (w_8, w_9, w_{10}, w_{11})$
...
...
...
$(k_{128}, k_{129}, k_{130}, k_{131}) = S_3 (w_{128}, w_{129}, w_{130}, w_{131})$

then renumber the 32-bit values k_j as 128-bit subkeys K_i as follow:

$$K_i = k_{4i}, k_{4i+1}, k_{4i+2}, k_{4i+3}$$

4.2. Least Significant Bit

The proposed system uses a LSB technique. This system involved the structure of steganography by using least significant bit (LSB) technique in hiding messages in an image. It works by using the least significant bits of each pixel in one image to hide the most significant bits of another [5]. Digital images are stored in either 24-bit or 8-bit per pixel files.

24-bit images - To hide an image in the LSBs of each byte of a 24-bit image, you 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. 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.

8-bit images - Applying LSB technique to each byte of an 8-bit image, only one bit can be encoded into each pixel, as each pixel is represented by one byte. For example, if we use 8-bit image to hide the letter A (has the binary value 01000001), eight pixels are need. Suppose the original eight pixels are:

(00100111) (11101001) (11001000) (00100111)
(11001000) (11101001) (11001000) (00100111)

Inserting the letter A (as a binary value) into these eight pixels will give the following (starting from left side):

(00100110) (11101001) (11001000) (00100110)
(11001000) (11101000) (11001000) (00100111)

Only the 3 underlined bits needed to be changed according to the embedded message.

5. Experimental Results

Table 1: Measured values of the Cover-image and Stego-image

No.	Message Size	Image Type	Cover	Stego
1	2.27 KB	Image size	28.6 KB	300 KB
		Image width / height	800x600 pixels	800x600 pixels
		Horizontal / Vertical Resolution	72 dpi / 72 dpi	96 dpi / 96 dpi
		Bit Depth	24	24
2	726 B	Image size	1.63 MB	4.07 MB
		Image width / height	1800x1200 pixels	1800x1200 pixels
		Horizontal / Vertical Resolution	300 dpi / 300 dpi	96 dpi / 96 dip
		Bit Depth	24	24
3	1.98 KB	Image size	17 KB	89.4 KB
		Image width / height	320x240 pixels	320x240 pixels
		Horizontal / Vertical Resolution	96 dpi / 96 dpi	96 dpi / 96 dpi
		Bit Depth	8	24

The proposed system found that the size of information to be hidden relatively depends on the size of the cover-image. The message size must be smaller than the image. According to the measurements, image size, resolution, and bit depth have been changed when making experimentation from the cover-image to the stego-image. Whichever sizes of horizontal and vertical resolution of cover-image (tested with 1, 2, 72, 93, 96, 150, 200, 300 dip) have to be changed the same resolution 96 dip/96 dip after embedding process.

Similarly, whichever (8 or 24) size of bit depth in cover-image is changed the size 24 in modified stage. On the other hand, pixel values are the same in the embedding process. Since this system replaces the plain message bit into the appropriate pixel without adding or removing the pixels from the cover-image, the pixels after stego-stage is the same as before.

This process simply embedded the message into the cover-image without supplied any stego-key. If an encrypted message is intercepted, the interceptor knows the text is an encrypted message. But with steganography, the interceptor may not know that a hidden message even exists. Experimental results describe that the stego-image did not release any identifiable visual difference when to compare the cover-image. Therefore, in human vision, the resulting stego-image (Figure 3) will look identical to the cover-image (Figure 4) so that this system has good security.

To measure the experimental result of the system, 20 images are tested with several of message sizes. The Table 1 shows that the measured values of cover-image and stego-image. Thus, the system can be concluded that is quite robust and gives relative faster speed of encryption and embedding tasks. This is reasonable that this may be because of the fact that the image can take much time to embed the message bit into the LSB of image blocks. Too much message has to be encrypted, the larger image file size is needed and consequently the slower the execution speed of the whole process.

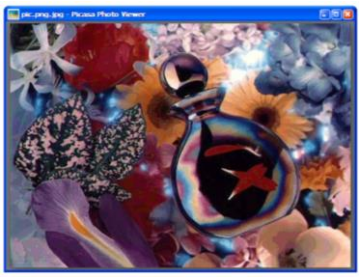

Figure 3 Cover-image

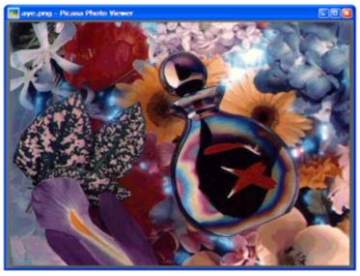

Figure 4 Stego-image

6. Conclusion

Both steganography and cryptography are two integral parts of information security. Their purpose is to provide secrecy to communications and can be used in conjunction. Even if the presence of hidden information is detected, one can protect the contents of the message using encryption. The proposed system have been demonstrated that has optimum cryptographic performance and steganographic performance by unifying those two models, in order to devise a new model holding the features that are peculiar both to the steganographic and to the cryptographic model.

7. References

[1] http://www.io.com/~hcexres/power_tools/hyper web/ website1.pdf

[2] http://www.cccure.org/Documents/Cryptography/ sisspallinone.pdf

[3] C. Berbain, O. Billet, A. Canteaut, N. Courtois, H. Gilbert, L. Goubin, A. Gouget, L. Granboulan, C. Lauradoux, M. Minier, T. Pornin and H. Sibert, "Sosemanuk, a fast software-oriented stream cipher*", http://www.bolet.org/~pornin/2005-skew-rbain+ALL1 .pdf

[4] Domenico Bloisi, Luca Iocchi, "Image based Steganography and Cryptography", Dipartimento di Informatica e Sistemistica Sapienza University of Rome, Italy

[5] Gray C. Kessler, "An Overview of Steganography for the Computer Forensics Examiner"

[6] Mohammed A.F. Al-Husainy, " Image Steganography by Mapping Pixels to Letters", Department of Computer Science, Faculty of Sciences and IT, Al-Zaytoonah University of Jordan

[7] Muhalim Mohamed Amin, Subariah Ibrahim, Mazleena Aslleh, Mohd Rozi Katmin, "Information Hiding Using Steganography", http://eprints.utm.my/4339/1/71847.pdf

[8] Neha, Mr.J.S.Bhatia, Dr(Mrs)Neena Gupta, "An Encrypto-Stego Technique Based Secure Data Transmission System"

[9] Ross Anderson, Eli Biham, Lars Knudsen, "Serpent: A proposal for the Advanced Encryption Standard"

[10] Sonali Gupta, "All About Steganography" http://palisade.plynt.com/issues/2005Apr/ steganography/