A Step Towards Practical Steganography Systems

Abdelkader H. Ouda and Mahmoud R. El-Sakka, Senior Member, IEEE

Computer Science Department, University of Western Ontario, London, Ontario, Canada
{kader, elsakka}@csd.uwo.ca

Abstract. There has been many hidden communication techniques proposed in the last few years. The focus was given to steganography to build such techniques. Utilizing stego-key(s) to hide secret messages into images strengthen the security of these techniques. However, adopting one of the available key-agreement protocols, to distribute stego-key(s) between the communicating parties, will destroy the infrastructure of the entire communication. The reason is that, these protocols perform their transactions on sight, while the desirable communications need to be completely hidden. In this paper, a *key-generation unit* is proposed to be added to the steganography general model. This unit utilizes a new key-agreement protocol, *stego-KA*, to help support the entire class of hidden communication techniques to exchange the sego-key(s) covertly. The proposed stego-KA protocol is based on *Diffie-Hellman* key establishment protocol and has significant advantages that support hidden communications.

1 Introduction

It has been said throughout time that, “a picture is worth a thousand words.” However, in this digital era, it could be said that, “a picture is worth a thousand *secrets.*” It should come as no surprise that a picture (digital image) might be distributed while it contains a *secret message* that is hidden to the human eye. This message can be extracted only by a sophisticated image utility, using a *secret key*. These types of applications are known as *hidden communication* techniques, which utilizing a technology known by *steganography*.

One of the most realistic schemes for steganography applications goes back to Simmons in 1984 [1]. Simmons introduced his hidden communication model using the prisoner's problem, which became the most widely used scenario characterizing these models.

The *prisoner's problem* states that, there are two criminals confined in separated cells. The warden gives them the opportunity to communicate with each other through a message-exchanging channel, which is monitored by the warden. The only restriction on this channel is that it is open to the warden, and if any message is encrypted, the warden should have access to the decryption key. The main reason for this communication is that the warden will mislead the prisoners by sending them false messages in order to trick the criminals into thinking they were sent by the other party. The prisoners, on the other hand, plan to use this channel in order to arrange an escape. To do this, the prisoners will have to deceive the warden by finding a way of

M. Kamel and A. Campilho (Eds.): ICIAR 2005, LNCS 3656, pp. 1158 – 1166, 2005.

communicating secretly between them in full view of the warden. This means that even if a message contains secret information it would look innocuous to the warden. Since the prisoners anticipate that the warden will try to deceive them by introducing fraudulent messages, they should prepare an authentication model along with their hidden communication.

While Simmons utilized cryptography in his scheme, the vast majority of the information hiding schemes in literature [2-11] utilize steganography to solve the prisoners' problem. The main purpose of these schemes is that a secret message can be transmitted invisibly within another innocent medium, such as images. This transmission should occur so that only the sender and the receiver have the ability to insert, detect and extract the hidden message.

The rest of this paper is organized as follows. Section 2 demonstrates the general framework of the steganography model. The analysis of the related steganographic techniques is given in Section 3. Section 4 states the main problem. The proposed solution is described in detail in Section 5. The conclusion is offered in Section 6.

2 Framework of Steganography Model

In general, the basic framework of the image steganography model is illustrated in Fig. 1. This model consists of two main processes, namely the *embedding process* and the *extracting process*. The main function of the embedding process is to hide the secret message, called *embedded message*, in a given image, called *cover-image*. In hidden communication techniques, the cover-image is no more than an innocent (unrelated to the embedded message) piece of information that is used to hide the secret information. A secret key, called *stego-key*, is used in the embedding process such that it makes the embedded message computationally infeasible to extract without possessing this key. The output of the embedding process is called *stego-image*, which is the original image holding the hidden secret message. This output becomes, at the other end, the input of the extracting process, in which the embedded message is

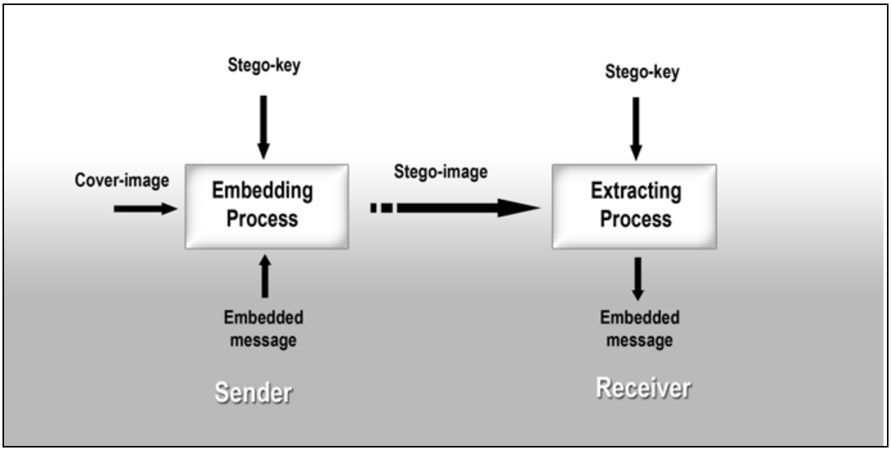

Fig. 1. The framework of the watermark generation and embedding process

extracted from the stego-image to complete the hidden communication process. Since the stego-key is used in the embedding process, it needs to be used in the extracting process. Note that, the notation and naming conventions that are used in **Fig. 1** are adopted after the first Information Hiding Workshop [15].

3 Analysis and Related Works

In this section we will study steganography techniques based on the usage of the stego-key. The reason for this is to show at what extent the stego-key is important in the entire steganography processes, including embedding, extracting, and verifications. The stego-key can be used in one or more of the following functions:

1. Determine the embedding position (the modified pixels) based on:
   - stego-key bit sequence, or
   - generated pseudo-random sequence seeded by a stego-key,
2. Scramble the embedded message (hidden information) to randomize the hidden information,
3. Scramble the cover image (pixel or block permutation) to :
   - protect the embedded message, and/or to
   - increase the embedding capacity.

3.1 Using the Stego-key to Determine the Embedding Positions

Kundar *et al.* [2] propose using the discrete wavelet transform (DWT) [12] to drive a multi-resolution representation of the image data. To hide one bit, the median of three coefficients will be quantized (modified) to match this bit value. These coefficients are selected based on the bit value of a stego-key. To restore the hidden message, the same stego-key is needed to the receiver.

Qi *et al.* [3] use two different stego-keys. In the embedding process, the message will be hidden in specific columns/rows of an image. The selection of these columns/rows is based on the bit sequence of one secret key. A global blur operation is then applied to the entire image in order to make the marked columns/rows unpredictable. The components of the blur kernel are also chosen based upon another secret key. Therefore, this technique has a high security level since it is based on two different secret keys. However, these keys need to be agreed in advance by the both parties.

Other kinds of applications attempt to locate an embedding position in an image using a pseudo-random sequence that is generated either by the stego-key or by any shareable seed between the sender and the receiver. Sharp [4] uses a linear feedback shift register (LFSR) [13] to generate the random sequence. This sequence is used to determine the order in which the pixels from the image are visited to embed the secret information. Therefore, the communicating parties need the same key in order to generate the same sequence.

Licks *at el.* [5] present a technique that utilizes *discrete Fourier transform* (DFT) properties [14] to embed a pseudo-random sequence as a secret message. This sequence is generated in circular form based on a stego-key. The sequence is then embedded into the magnitude part of the DFT coefficients at a specific secret radius. In the other side, this secret information is needed to verify or extract the embedded message.

3.2 Scrambling the Embedded Message to Randomize the Hidden Information

Some techniques attempt to protect the embedded message by scrambling the messages bits before being hidden. Liu *et al.* [6] scramble the secret message by adding to it a pseudo-random sequence generated by a shared secret key. The authors utilize the DWT coefficients to hide the message, and use a technique called error correction code and 2-D interleaving [7] to lower the detection error probability.

Marvel *et al.* [8] use a similar idea to protect the embedded message. The spread spectrum communication, error correction coding, and image processing are combined to present their technique. The embedded message is first encrypted using a secret key. Another key is used to generate a pseudo-random sequence. Then both, results are modulated using a third secret key to embed the output into the cover image. These methods are also suffering from the key-distribution problem.

3.3 Scrambling the Cover Image

Another way to protect the embedded message is to randomize, or permute, the cover image using the stego-key before the embedding process. Pan *et al.*, [9] propose to divide the image into subblocks. These blocks are ranked based on a specific pattern matching method so that the higher ranked block is the most suitable for data embedding. The chosen block is then permuted using a secret key before the embedding process.

Tseng *et al.* [10] propose a scheme that is able to conceal critical messages into binary images. The image is divided into small blocks; each block is scrambled by a bitwise exclusive-OR with a binary matrix of the same size. This matrix is played as a secret key. The output is then weighted by another secret integer matrix to determine which pixels need to be swapped to embed the secret message. At the end, the image pixels are modified so that the receiver can extract the message by applying reverse operation using the same secret keys.

Some other applications attempt to increase the number of the transform coefficients that may be used to hide the embedded message bits. This can be done by decorrelating the cover image samples that can result in uncorrelated and identical distributed samples over the entire image. Alturki *et al.*, in [11], use this approach to embed more data into the DCT domain of an image. The stego-key is used to decorrelate the given image. The key is used to shuffle the image pixels so that the resulting image looks like white noise to the viewer.

All these methods require that the sender and the receiver to agree upon the shared stego-key in advance.

4 The Major Problem

We can conclude from the above analysis that, the common requirements to achieve hidden communication are simply: 1) the cover image and the hidden messages should be unrelated, 2) the hidden message should not provide any evidence of its existence, and 3) the hidden message should not be accessible to anyone but the sender and the receiver, who possess the stego-key. We have also shown that, there are many hidden communication techniques that fulfill these requirements. However, the distribution mechanism of the stego-keys has received less attention in most of these techniques. Definitely, stego-key is an essential piece in either the embedding or the extraction process in steganographic systems available today. As a result of this, any steganographic system needs an authenticated protocol that gives the two parties (the sender and the receiver) the ability to communicate and end up with a shared secret key.

At first glance, it appears that utilizing any secure key-agreement protocol might solve this problem (the key distribution problem). As the matter of fact, an authenticated key-agreement protocol is needed, however, one should indeed note that, these protocols always have some public transactions. This of course will flaw the infrastructure of the hidden communication. It is also worse noting that, a secure solution of this problem is, in fact, a solution for the entire class of the hidden communication techniques. In this paper, a new approach that covertly enables two parties to establish a session secret key (stego-key) is proposed. More details are given in the following Section.

5 The Proposed Key Generation Unit

In this paper, we propose to modify the general model of the steganography, see Fig. 2, by adding a new unit called "*key generation unit*", [16]. The main purpose of this unit is to produce a shared secret key to the communicating parties (the sender and the receiver), so that the protocol transactions are performed undercover. This unit utilizes a new hidden key agreement protocol (*stego-KA*). This protocol is based on *Diffie-Hellman* key establishment protocol [17]. It operates on the group of points of an elliptic curve over a finite field [18]. Our protocol closely follows the approach of [19], and has significant advantages that support the hidden communications.

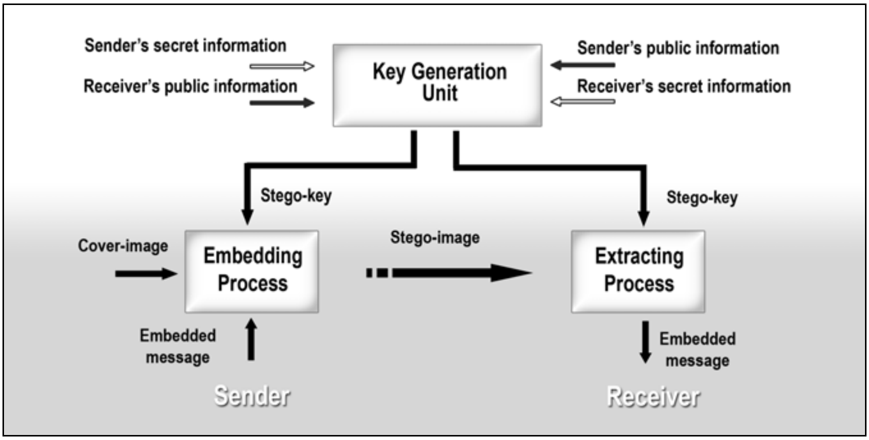

Fig. 2. The modified general steganography model

5.1 Basic Notations

Before discussing the protocol in more details, we first define some notation.

E A non-singular elliptic curve over a finite field $GF(q)$ that defines a set of points (x, y), which satisfy an elliptic curve equation $y^2 = x^3 + ax + b$, where $q = p^m$ and p is a large prime, $a,b \in GF(q)$

P A point (x_p, y_p) of order n that satisfies the elliptic curve E

X (.) The x-coordinate of any point on the elliptic curve E

d_A, d_B Long-term private keys for party A and B respectively, selected randomly from the interval [1,n-1]

Q_A, Q_B Long-term public keys for party A and B respectively, where $Q_A = d_A P$ and $Q_B = d_B P$. Note that, due to the hardness of the elliptic curve discrete logarithm problem [22], it is computationally invisible to get d_A from P and Q_A

r_A, r_B Session private keys for party A and B respectively, selected randomly from the interval [1,n-1] at each protocol run

R_A, R_B Session public keys for party A and B respectively, where $R_A = r_A P$ and $R_B = r_B P$

H(.) The secure hash algorithm SHA-1 [20]. SHA-1 takes a message of an arbitrary length and produces a 160-bit output called a message digest

I Any random stego-image

$Hide_k(m, I)$ Hiding function to conceal the data m into the image I using the key k. Note that, any of the hidden communication techniques described in Section 3 might be used.

$rh(I)$, $lh(I)$ The right half side and left half side of the image I respectively

5.2 Security Attributes Requirements

Any secure key-agreement protocol should fulfill some security attributes [21]. Consequently, the hidden key agreement protocol needs to satisfy these attributes as well to be a reliable protocol. These attributes include:

Known session keys, the security of new session keys will not be affected if some previous session keys were disclosed.

Forward secrecy, the security of previous session keys will not be affected if a long-term secret key of one or more parties is compromised.

Unknown key-share, party A cannot be forced into sharing a key with party B without A's knowledge, i.e., when A believes the key is shared with some party $C \neq B$, and B believes the key is shared with A. This attribute is also called *man-in-the-middle*.

Key-compromise impersonation, if A's secret key is disclosed, any one who knows this key can impersonate A. Hence, this loss of information should not allow the adversary to impersonate other parties to A.

In Section 5.4 we will show how the proposed protocol satisfies these attributes.

5.3 The Hidden Key-Agreement Protocol, *Stego*-KA

The detail transactions of stego-KA protocol are described as follows:

1. *A* selects a session secret key r_A and then computes a session public key R_A .
2. *A* computes the initial session key $K_0 = d_A Q_B = d_A d_B P$, K_0 is a point on the elliptic curve *E*.
3. *A* choses a random digital image I_1 , and then hides the value R_A into the I_1 using initial session key K_0 . I.e., computes the function $Hide_{k_0}(R_A, I_1)$. Note that, this function might be the same as that will be used to hide the secret message in the original technique, see Section 3. Therefore the security strength of this function will be equivalent to the security of the entire technique.
4. Image I_1 is published somewhere in an open network such as Internet.
5. *B* obtains the image I_1 then performs the following:
   - Computes the initial session key $K_0 = d_B Q_A = d_A d_B P$
   - Uses K_0 to extract R_A from the image I_1 . Note that, the embedding and extracting processes are public methods, however K_0 is accessible only to *A* and *B*.
   - Selects a session secret key r_B and then computes a session public key R_B
   - Generates the target session key $K = X(Q_A r_B + R_A d_B)$
   - Computes the value $Z_1 = H(lh(I_2)) \| X(R_A) \| X(R_B)$, where || be a bit-stream concatenation
   - Select a random image I_2 , and apply the function $Hide_K(R_B \| H(Z_1), I_2)$
   - Publishes the image I_2 somewhere in an open network such as Internet
6. *A* obtains the image I_2 then performs the following:
   - Generates the target session key $K = X(Q_B r_A + R_B d_A)$
   - Uses K to extract R_B and $H(Z_1)$, which is *z*, from the image I_2
   - Computes the value $Z^* = H(lh(I_2)) \| X(R_A) \| X(R_B)$
   - Verifies if $H(Z^*) = z$; if the validation failed, the protocol will be ended with a failure
   - Otherwise, computes the value $Z_2 = H(rh(I_2)) \| X(R_B) \| X(R_A)$
   - Apply the function $Hide_K(H(Z_2), I_2)$
7. *B* extracts $H(Z_2)$, which is z_2 , from the image I_2 using the session key K and verify if $H(H(rh(I_2)) \| X(R_B) \| X(R_A)) = z_2$
8. If the validation failed the protocol will be ended with a failure, otherwise *K* will be the secret session key between *A* and *B*

5.4 The Major Features of the Proposed Model

The key-generation unit is able to provide the communicating parties with some assurance that they know each other's true identities. Stego-KA protocol, which utilizes hidden key-confirmation transactions, has helped these parties end up sharing a common stego-key known only to them. This stego-key can then be used thereafter to establish the desirable hidden communications as it is described earlier in Section 3.

In addition to the hidden transactions property, there are other security attributes for the Stego-KA protocol.

Known session keys

Based on the security definition of the elliptic curve addition, losing any information about previous stego-key(s), i.e., $K = X\ (Q_A r_B + R_A d_B)$, does not affect the protocol security. I.e., it doesn't help an adversary to be able to discover a stego-key that might be generated by a fresh protocol run, especially when the session keys, i.e., r_A and r_B are refreshed each time the protocol is carried out.

Forward secrecy

Stego-KA protocol provides perfect forward secrecy. If for example the long-term secret key of the party *A* is disclosed, i.e., d_A , the protocol security might be affected. However, the past produced stego-key(s) will not. The reason for this is that, the agreed stego-key $K = X\ (Q_B r_A + R_B d_A)$ is chosen also based on the values r_A and r_B , which were chosen independently at random by parties *A* and *B* respectively. Therefore, the adversary will face the elliptic curve discrete logarithm problem [22] to learn any extra information about the key.

Unknown key-share

Stego-KA protocol will not be completed until both parties prove knowledge of the shared stego-key by using it in subsequent communications. The hidden message send from *B* to *A* provides key confirmation of *B* to *A*. The hidden message embedded and send from *A* gives an assurance to *B* that *A* actually possesses the key.

Key-compromise impersonation

Generating session keys r_A and r_B at each protocol run kills any hope to an adversary to impersonate any party *C* to *A*, when d_A is disclosed. Note that, if these session keys are not evolved in the stego-key, the adversary can compute the secret $X\ (Q_C + d_A)$ easily to impersonate *C* to *A*.

6 Conclusion

The main goal of this paper is to make stride towards the practical use of steganography in hidden communications. The paper enhances the general steganography model by enabling the use of a hidden key-agreement protocol "stego-KA" through a new steganography unit called "key-generation unit". Stego-KA protocol is based on the idea that the communicating parties need to contribute their information, through a hidden channel, by which the stego-key will be established.

This paper also provides a new approach to classify key-based steganography techniques, which are grouped based on the usage of secret keys. This new classification facilitates the way by which these hiding techniques could be utilized in the proposed protocol.

References

1. G. Simmons, "The Prisoners' Problem and the Subliminal Channel," CRYPTO83 - Advances in Cryptology, August 22-24. 1984. pp. 51–67.
2. D. Kundur, D. Hatzinakos, "Digital watermarking using multiresolution wavelet decomposition," In IEEE ICASSP'98, volume 5, pages 2659– 2662, Seattle, May 1998.

3. H. Qi, W. Snyder, W. Sander, “Blind Consistency Based Steganography For Information Hiding In Digital Media,” IEEE International Conference on Multimedia and Expo, 2002. ICME '02. Proceedings. 2002, Volume: 1 , Page(s): 585 –588, August. 2002.
4. T. Sharp, “An Implementation of Key-Based Digital Signal Steganography,” Information Hiding, LNCS 2137, pp. 13-26
5. V. Licks, R. Hordan, “On digital image watermarking robust to geometric transformations,” International Conference on Image Processing, Volume: 3 , Page(s): 690–693, September. 2000.
6. H. Liu, J. Liu, J. Huang, D. Huang, Y. Shi, “A robust DWT-based blind data hiding algorithm,” IEEE International Symposium on Circuits and Systems, ISCAS 2002, Volume: 2 , Page(s): II-672 -II-675, May 2002.
7. G. Elmasry, Y. Shi, “2-D Interleaving for Enhancing the Robustness of Watermark Signals Embedded in Still Images,” IEEE International Conference on Multimedia and Expo (II) Page(s): 731–734, 2000.
8. M. Marvel, C. Retter, C. Boncelet, “A methodology for data hiding using images, IEEE Proceedings on Military Communications Conference, MILCOM 98, Volume: 3 , Page(s): 1044–1047, October 1998.
9. G. Pan, Z. Wu, Y. Pan, “A data hiding method for few-color images,” Proceedings IEEE International Conference on Acoustics, Speech, and Signal Processing, (ICASSP '02), Volume: 4, PP 3469–3472, May 2002.
10. Y. Tseng, Y. Chen, H. Pan, “A secure data hiding scheme for binary images,” IEEE Transactions on Communications, Volume: 50 Issue: 8, Page(s): 1227 -1231Aug. 2002
11. F. Alturki, R. Mersereau, “A novel approach for increasing security and data embedding capacity in images for data hiding applications,” International Conference on Information Technology: Coding and Computing, Page(s): 228 –233, April 2001.
12. R. Rao, A. Bopardikar, “Wavelet Transforms”, Addison Wesley Longman Inc., Reading, Massachusetts, 1998.
13. N. Zierler, “Linear Recurring Sequences”, Journal of the Society for Industrial and Applied Mathematics, Vol 7, No. 1, pp. 31-48, March 1959.
14. G. Gonzalez, R. Woods, “Digital Image Processing”, Addison-Wesly Publication Company. 1992.
15. B. Pfitzmann, “Information Hiding Terminology,” Information Hiding: first international workshop, Proceedings LNCS 1147, Berlin: Springer, 1996.
16. A. Ouda, “Digital Watermarking Techniques for Image Security and Hidden Communications”, Ph.D. Dissertation, Computer Science Department, University of Western Ontario, Canada, 2004.
17. W. Diffie, M. Hellman. New directions in cryptography. IEEE Transactions on Information Theory, IT-22: 644-654, 1976.
18. A. Menezes, P. Oorchot, and S. Vanstone, Handbook of Applied Cryptography, CRC Press, Florida, USA, 1997.
19. L. Law, A. Menezes, M. Qu and S. Vanstone. “An Efficient Protocol for Authenticated Key Agreement,” Technical Report CORR 98-05, Department of C&O, University of Waterloo, 1998.
20. National Institute of Standards and Technology NIST, “Secure hash standard”, Federal Information Processing Standards Publication FIPS, 180-1.
21. S. Wilson, D. Johnson and A. Menezes, “Key agreement protocols and their security analysis,” proceedings of the sixth IMA international Conference on Cryptography and Coding, LNCS 1355, Springer-Verlag, pp: 30-45, 1997.
22. A. Menezes, “Evaluation of security level of cryptography: The elliptic curve discrete logarithm problem,” CRYPTREC Report, December 14 2001.