IMAGE BASED STEGANOGRAPHY AND CRYPTOGRAPHY

Domenico Bloisi and Luca Iocchi
Dipartimento di Informatica e Sistemistica
Sapienza University of Rome, Italy
E-mail: *<lastname>@dis.uniroma1.it*

Keywords: Steganography, cryptography.

Abstract: In this paper we describe a method for integrating together cryptography and steganography through image processing. In particular, we present a system able to perform steganography and cryptography at the same time using images as cover objects for steganography and as keys for cryptography. We will show such system is an effective steganographic one (making a comparison with the well known F5 algorithm) and is also a theoretically unbreakable cryptographic one (demonstrating its equivalence to the Vernam Cipher).

1 INTRODUCTION

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, corporate data, etc.

More specifically, steganography[1] is the art and science of communicating in a way which hides the existence of the communication (Johnson and Jajodia, 1998). A steganographic system thus embeds hidden content in unremarkable cover media so as not to arouse an eavesdropper's suspicion (Provos and Honeyman, 2003). As an example, it is possible to embed a text inside an image or an audio file.

On the other hand, cryptography is the study of mathematical techniques related to aspects of information security such as confidentiality, data integrity, entity authentication, and data origin authentication (Menezes et al., 1996). In this paper we will focus only on confidentiality, i.e., the service used to keep the content of information from all but those authorized to have it.

Cryptography protects information by transforming it into an unreadable format. It is useful to achieve confidential transmission over a public network. The original text, or *plaintext*, is converted into a coded equivalent called *ciphertext* via an encryption algorithm. Only those who possess a secret key can decipher (*decrypt*) the ciphertext into plaintext.

Cryptography systems can be broadly classified into symmetric-key systems (see Fig. 1) that use a single key (i.e., a *password*) that both the sender and the receiver have, and public-key systems that use two keys, a public key known to everyone and a private key that only the recipient of messages uses. In the rest of this paper, we will discuss only symmetric-key systems.

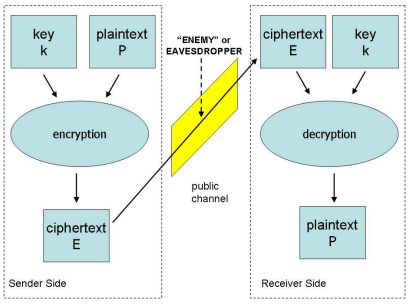

Figure 1: Symmetric-key Cryptographic Model.

Cryptography and steganography are cousins in the spy craft family: the former scrambles a message so it cannot be understood, the latter hides the

[1] from Greek, it literally means "covered writing"

message so it cannot be seen. A cipher message, for instance, might arouse suspicion on the part of the recipient while an invisible message created with steganographic methods will not.

In fact, steganography can be useful when the use of cryptography is forbidden: where cryptography and strong encryption are outlawed, steganography can circumvent such policies to pass message covertly. However, steganography and cryptography differ in the way they are evaluated: steganography fails when the "enemy" is able to access the content of the cipher message, while cryptography fails when the "enemy" detects that there is a secret message present in the steganographic medium (Johnson and Jajodia, 1998).

The disciplines that study techniques for deciphering cipher messages and detecting hide messages are called *cryptanalysis* and *steganalysis*. The former denotes the set of methods for obtaining the meaning of encrypted information, while the latter is the art of discovering covert messages.

The aim of this paper is to describe a method for integrating together cryptography and steganography through image processing. In particular, we present a system able to perform steganography and cryptography at the same time. We will show such system is an effective steganographic one (making a comparison with the well known F5 algorithm (Westfeld, 2001)) and is also a theoretically unbreakable cryptographic one (we will demonstrate our system is equivalent to the Vernam cipher (Menezes et al., 1996)).

2 IMAGE BASED STEGANOGRAPHIC SYSTEMS

The majority of today's steganographic systems uses images as cover media because people often transmit digital pictures over email and other Internet communication (e.g., eBay). Moreover, after digitalization, images contain the so-called quantization noise which provides space to embed data (Westfeld and Pfitzmann, 1999). In this article, we will concentrate only on images as carrier media.

The modern formulation of steganography is often given in terms of the *prisoners' problem* (Simmons, 1984; Kharrazi et al., 2004) where Alice and Bob are two inmates who wish to communicate in order to hatch an escape plan. However, all communication between them is examined by the warden, Wendy, who will put them in solitary confinement at the slightest suspicion of covert communication.

Specifically, in the general model for steganography (see Fig. 2), we have Alice (the *sender*) wishing to send a secret message *M* to Bob (the *receiver*): in order to do this, Alice chooses a cover image *C*.

The steganographic algorithm identifies *C*'s redundant bits (i.e., those that can be modified without arising Wendy's suspicion), then the embedding process creates a *stego image S* by replacing these redundant bits with data from *M*.

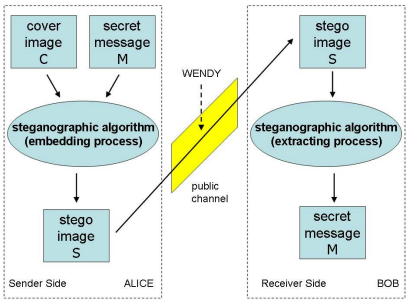

Figure 2: Steganographic Model.

S is transmitted over a public channel (monitored by Wendy) and is received by Bob only if Wendy has no suspicion on it. Once Bob recovers *S*, he can get *M* through the extracting process.

The embedding process represents the critical task for a steganographic system since *S* must be as similar as possible to *C* for avoiding Wendy's intervention (Wendy acts for the *eavesdropper*).

Least significant bit (LSB) insertion is a common and simple approach to embed information in a cover file: it overwrites the LSB of a pixel with an *M*'s bit. If we choose a 24-bit image as cover, we can store 3 bits in each pixel. To the human eye, the resulting stego image will look identical to the cover image (Johnson and Jajodia, 1998).

Unfortunately, modifying the cover image changes its statistical properties, so eavesdroppers can detect the distortions in the resulting stego image's statistical properties. In fact, the embedding of high-entropy data (often due to encryption) changes the histogram of colour frequencies in a predictable way (Provos and Honeyman, 2003; Westfeld and Pfitzmann, 1999).

Westfeld (Westfeld, 2001) proposed F5, an algorithm that does not overwrite LSB and preserves the stego image's statistical properties (see Sect. 5.2).

Since standard steganographic systems do not provide strong message encryption, they recommend to encrypt *M* before embedding. Because of this, we have always to deal with a two-steps protocol: first we must cipher *M* (obtaining *M'*) and then we can embed *M'* in *C*.

In the next sections we will present a new all-in-one method able to perform steganography providing

strong encryption at the same time.

Our method has been planned either to work with bit streams scattered over multiple images (in an online way of functioning) or to work with still images; it yields random outputs, in order to make steganalysis more difficult and it can cipher M in a theoretically secure manner preserving the stego image's statistical properties.

The simplicity of our method gives the possibility of using it in real-time applications such as mobile video communication.

3 A STEGO-CRYPTOGRAPHIC MODEL

Figures 1 and 2 depict the cryptographic and steganographic system components. Here we discuss how we could unify 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 (see Fig. 3).

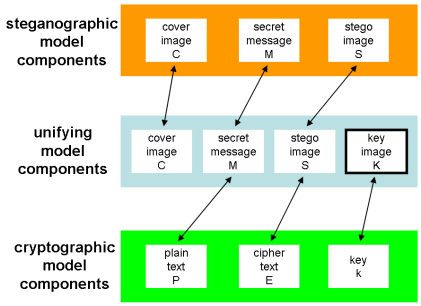

Figure 3: Mapping between model components.

The mapping between P and M, E and S, and k and K is possible because we can consider all the components in Fig. 3 as bit sequences and then realize a relation between the co-respective bit sets.

The unifying model results as a steganographic one with the addition of a new element: the *key image* K. It gives the unifying model the cryptographic functionality we are searching for, preserving its steganographic nature.

The unifying model embedding process yields S exploiting not only C's bits but also K's ones (see Sect. 4.1): this way of proceeding gives Alice the chance to embed the secret message M (that is, the plaintext) into the cover image C (as every common steganographic system) encrypting M by the key image K (as a classical cryptographic system) at the same time. At the receiver side, Bob will be able to recover M through S and K (see Sect. 4.2). In addition, Wendy will neither detect that M is embedded in S nor be able to access the content of the secret message (see Fig. 4).

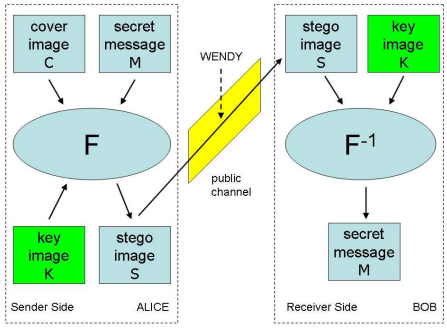

Figure 4: The unifying model.

4 IMAGE BASED STEGANOGRAPHY AND CRYPTOGRAPHY

The function denoted by F in Fig. 4 represents the embedding function we are going to explain in this section. The symbol F^{-1} indicates the extraction function, since it is conceptually the inverse of embedding. We will call ISC (**I**mage-based **S**teganography and **C**ryptography) the algorithm which carries on such functions.

4.1 ISC Embedding Process

Figure 5 shows the embedding process. The choice of the stego image format makes a very big impact on the design of a secure steganographic system.

Raw, uncompressed formats, such as BMP, provide the biggest space for secure steganography, but their obvious redundancy would arise Wendy's suspicion (in fact, why someone would have to transmit big uncompressed files when he can strongly reduce their size through compression? (Fridrich et al., 2002)). Thus, ISC embedding algorithm must yield a compressed stego image, in particular we choose to produce a JPEG file, because it is the most widespread image format.

While the output of the embedding process is a JPEG image (as we noted above), the inputs are: the secret message bit sequence, an image C, and an image K. C and K can be either uncompressed images (e.g., BMP) or compressed ones (e.g., JPEG), in addition they can be either distinct images or the same image.

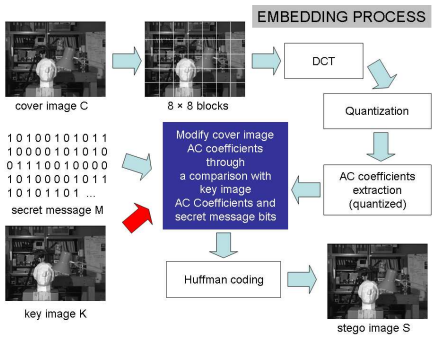

Figure 5: ISC embedding process.

The embedding process will be a modification of the JPEG encoding scheme. First of all, we subdivide *C* in a set of 8 x 8 pixel blocks and compute the Discrete Cosine Transform (DCT) on each block obtaining a set of DCT coefficients; then they are quantized.

After quantization, DC coefficients and AC zero coefficients are discarded. The remaining AC nonzero coefficients are stored in a vector called `coverAC[]`, that is a signed integer array. We have to repeat the previous list of operations for the key image *K* obtaining `keyAC[]`, a signed integer array as `coverAC[]`.

Now, in order to yield the stego image *S*, we are able to modify `coverAC[]` according to the following **Em1** embedding algorithm. We will call `stegoAC[]` the modified `coverAC[]` array.

```
Embedding Algorithm Em1.
Input: coverAC[], keyAC[], message bit array M
Output: stegoAC[]

for every bit M[i] of the message array M
  if  (M[i] == 1) // we want to codify a 1
    if (coverAC[i] and keyAC[i] are both even or
        both odd numbers)
      if(coverAC[i] == 1) stegoAC[i] = 2
      else if(coverAC[i] == -1) stegoAC[i] = -2
      else
        if(random() < 0.5)
            stegoaAC[i] = coverAC[i] - 1;
        else
            stegoaAC[i] = coverAC[i] + 1;
    end if
  else  // M[i] = 0, we want to codify a 0
    if (coverAC[i] and keyAC[i] are one equal
        and one uneven)
      if(coverAC[i] == 1) stegoAC[i] = 2
      else if(coverAC[i] == -1) stegoAC[i] = -2
      else
        if(random() < 0.5)
            stegoaAC[i] = coverAC[i] - 1;
        else
            stegoaAC[i] = coverAC[i] + 1;
    end if
  end if
end for
```

In the algorithm, `random()` returns a real value in [0, 1) that is chosen pseudorandomly with (approximately) uniform distribution from that range.

Notice that we must avoid to produce zero coefficients otherwise we would be unable to extract them at the receiver side (see Sect. 4.2).

Once the embedding algorithm terminates, we can proceed with `stegoAC[]` Huffman coding and eventually we obtain a JPEG image *S* as similar as possible to *C*. We can embed into *S* a number of bits equal to $min(length(\texttt{coverAC[]}), length(\texttt{keyAC[]}))$.

We have experimentally determined that we can hide in a JPEG image a message of size about 14% of the JPEG file dimension. Clearly the more amount of information we embed into *S* the more *S* will result different from *C*.

4.2 ISC Extracting Process

The ISC extracting process is very simple and consists in a comparison between *S* nonzero AC coefficients (`stegoAC[]`) and *K* nonzero AC coefficients (`keyAC[]`). In order to obtain these two sets of coefficients we perform a Huffman decoding step followed by the quantized AC coefficients extraction (see Fig. 6).

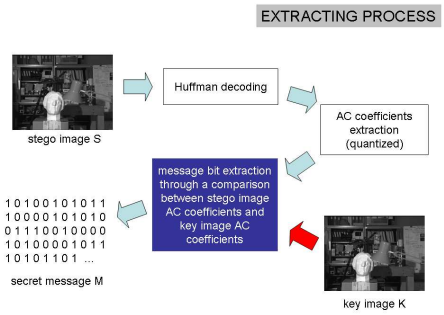

Figure 6: ISC extracting process.

Once the extraction is finished we compute the following **Ex1** extracting algorithm:

```
Extracting Algorithm Ex1.
Input: stegoAC[], keyAC[]
Output: message bit array M

for every coefficient stegoAC[i]
  if (stegoAC[i] and keyAC[i] are both even or both
      odd)
    M[i] = 0;
  else
    M[i] = 1;
  end if
end for
```

Images *C* and *K* depicted in Fig. 5 are two well known stereo images (the University of Tsukuba's

Stereo Image Pair). In fact, the key image idea derives from stereo vision: if you imagine the extracting process is a correlation algorithm, the secret message M could be seen as a disparity map between S and K, the embedding process as a sort of inverse correlation.

5 ISC PERFORMANCE

In this section we will present ISC performance with respect to both steganography and cryptography. We first demonstrate that ISC has optimum cryptographic performance, by proving that it is equivalent to Vernam cipher (Menezes et al., 1996), and then compare ISC steganographic performance with respect to the well known F5 algorithm (Westfeld, 2001).

5.1 ISC Cryptographic Performance

The Vernam Cipher. The Vernam cipher is a symmetric-key cipher defined on the alphabet $A = \{0,1\}$. A binary message $m_1, m_2, ..., m_t$ is operated on by a binary key string $k_1, k_2, ..., k_t$ of the same length to produce a ciphertext string $c_1, c_2, ..., c_t$ where $c_i = m_i \oplus k_i$, for $1 \leq i \leq t$ and $\oplus$ is the XOR operator. The ciphertext is turned back into plaintext simply inverting the previous procedure, i.e., $m_i = c_i \oplus k_i$, for $1 \leq i \leq t$.

If the key string is randomly chosen and never used again, the Vernam cipher is called a *one-time pad*.

One-time pad is theoretically unbreakable: if a cryptanalyst has a ciphertext string $c_1, c_2, ..., c_t$ encrypted using a random key string which as been used only once, the cryptanalyst can do no better then guess at the plaintext being any binary string of length t. To realize an unbreakable system requires a random key of the same length as the message (Shannon, 1949).

Equivalence between Vernam Cipher and ISC. Let `keyAC[]` and `coverAC[]` be two arrays containing the AC nonzero coefficients extracted from the key image K and the cover image C respectively.

Let `stegoAC[]` be an array initialized identical to `coverAC[]` (`stegoAC[]` will be modified during the embedding process because it will store the change needed by `coverAC[]`).

Let `M[]` be a binary array containing all the bits from the secret message M and let us suppose, for the sake of simplicity, that $length(\texttt{keyAC}[]) = length(\texttt{coverAC}[]) = length(\texttt{M}[])$. We want to find the following one-way relations *RK*, *RS*, and *RM*:

$$\texttt{keyAC}[] \xrightarrow{RK} k_1, k_2, ..., k_t$$
$$\texttt{stegoAC}[] \xrightarrow{RS} c_1, c_2, ..., c_t$$
$$\texttt{M}[] \xrightarrow{RM} m_1, m_2, ..., m_t$$

The last relation *RM* is simply the relation of equivalence since both `M[]` and $m_1, m_2, ..., m_t$ are bit sequences.

For finding RK we have to transform `keyAC[]` in a bit sequence through two further relations *RK1* and *RK2*:

$$keyAC[] \xrightarrow{RK1} keyEO[] \xrightarrow{RK2} k_1, k_2, ..., k_t$$

RK1 maps each AC coefficient `keyAC[i]` over a binary alphabet and store the corresponding bit value in `keyEO[i]` trough the rule:

```
if  keyAC[i] is even
  keyEO[i] = 0
else
  keyEO[i] = 1.
end if
```

RK2 is the relation of equivalence between `KeyEO[]` and $k_1, k_2, ..., k_t$. *RK* results as the combination of *RK1* and *RK2*.

We can repeat the above procedure for finding *RS* as a combination of *RS1* and *RS2*, i.e.,

$$stegoAC[] \xrightarrow{RS1} stegoEO[] \xrightarrow{RS2} c_1, c_2, ..., c_t$$

Let us use *RS1* on `coverAC[]` in order to obtain `coverEO[]` identical to `stegoEO[]` (note that initially `stegoAC[]` is equal to `coverAC[]`).

$$coverAC[] \xrightarrow{RS1} coverEO[]$$

Now we transform **Em1** in order to work with bit sequences, obtaining the algorithm **Em2**:

```
Embedding Algorithm Em2.
Input: coverEO[], keyEO[], M[]
Output: stegoEO[]

for every bit M[i] of the binary array M[]
  if (M[i] == 1)
    if (coverEO[i] ⊕ keyEO[i] == 0)      (1)
      stegoEO[i] = coverEO[i] ⊕ 1        (2)
    end if
  end if
  else //M[i] = 0
    if (coverEO[i] ⊕ keyEO[i] == 1)      (3)
      stegoEO[i] = coverEO[i] ⊕ 1        (4)
    end if
  end else
end for
```

Lines 1,2,3, and 4 perform (in the binary domain) the same operations made by algorithm **Em1**. Table 1 shows the truth table for every input feasible by algorithm **Em2**.

Table 1: Truth table for algorithm Em2.

M[i]	keyEO[i]	coverEO[i]	stegoEO[i]
0	**0**	0	**0**
0	0	1	0
0	**1**	0	**1**
0	1	1	1
1	**0**	0	**1**
1	0	1	1
1	**1**	0	**0**
1	1	1	0

You can notice that bold values correspond to the truth table for $c_i = m_i \oplus k_i$. Since `M[]` corresponds to the Vernam plaintext $m_1, m_2, ..., m_t$ (by virtue of *RM*), `keyAC[]` corresponds to the Vernam key $k_1, k_2, ..., k_t$ (by virtue of *RK1* and *RK2*), and `stegoAC[]` corresponds to the Vernam ciphertext $c_1, c_2, ..., c_t$ (by virtue of *RS1* and *RS2*) we can conclude asserting:

ISC embedding process and Vernam cipher encrypting step are equal.

The proof of equivalence between ISC extracting process and Vernam cipher decrypting step is trivial.

Let us transform algorithm **Ex1** in order to work with `M[]`, `keyEO[]`, and `stegoEO[]`.

```
Algorithm Ex2.
Input: stegoEO[], keyEO[]
Output: keyEO[]

for every bit stegoEO[i] of stegoEO[]
  M[i] = stegoEO[i] ⊕ keyEO[i]
end for
```

Since **Ex2** is identical to the Vernam cipher decrypting step ($m_i = c_i \oplus k_i$, for $1 \leq i \leq t$), we have that *ISC extracting process and Vernam cipher decrypting step are equal.*

Eventually, ISC and Vernam cipher are equivalent.

5.2 ISC Steganographic Performance

The ISC steganographic performance will be measured by comparing it with the well known F5 algorithm (Westfeld, 2001). In order to do this, we will compare the statistical behaviour of these two algorithms on the same input set. This will demonstrate that ISC withstands both visual and statistical attacks (Westfeld and Pfitzmann, 1999): visual attacks mean that one can see steganographic messages on the low bit planes of an image because they overwrite visual structures; statistical attacks consist in measure distortions in the DCT coefficients' frequency histogram produced by embedding.

F5 Algorithm. The F5 steganographic algorithm was introduced by Andreas Westfeld in 2001 (Westfeld, 2001). The goal of his research was to develop concepts and a practical embedding method for JPEG images that would provide high steganographic capacity without sacrificing security (Fridrich et al., 2002).

Instead of replacing the least-significant bit of a DCT coefficient with message data, F5 decrements its absolute value in a process called matrix encoding. As a result, there is no coupling of any fixed pair of DCT coefficients, meaning the χ^2-test (Provos and Honeyman, 2003; Westfeld and Pfitzmann, 1999) cannot detect F5 (χ^2-test measure the probability a DCT coefficients' frequency histogram is the product of a steganographic process).

F5 uses a permutative straddling mechanism to scatter the message over the whole cover medium. The permutation depends on a key derived from a password.

Moreover, F5 (as ISC) embeds data in JPEG images thus resulting immune against visual attacks because it operates in a transform space (i.e., the frequency domain) and not in a spatial domain.

Comparison between F5 and ISC. In order to realize a meaningful comparison between ISC and F5[2], we must embed the same message m into the same cover image c using both ISC and F5. After embedding, we have two stego images: S_{F5} produced by F5 and S_{ISC} generated by ISC. Both S_{F5} and S_{ISC} present a DCT coefficients histogram different from the c's original one. What we are interested in is to compare the amount of modifications introduced by F5 and ISC.

Figure 7 shows the result of such comparison obtained using a JPEG cover set C_{set} of 20 images (1024 x 768, average size 330 KB). In every image of C_{set} we have embedded a canto from Dante's Divina Commedia (about 5 KB for each canto) with a JPEG quality factor set to 80. Only for ISC, we also used the images of C_{set} as key images.

The mean difference (in percentage) for every AC coefficient in the interval $[-8, 8]$ is shown on the y-axis in Fig. 7, in particular the black columns represent the differences introduced by F5 embedding step while the white ones correspond to the number

[2]release 11+

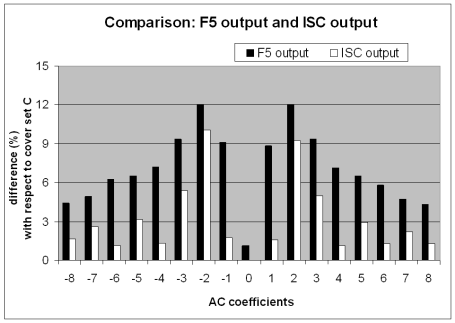

Figure 7: F5 and ISC comparison.

of modifications yielded by ISC embedding process. As one can notice, the respective difference values are comparable.

Em1 is a simplified version of ISC, because actually ISC spreads M over the entire stego image, yielding the same embedding density everywhere. In doing this, ISC neither uses permutative straddling nor matrix encoding, but simply divides the nonzero coefficients array in blocks of the same length. If necessary, only one of the coefficients in each block is modified.

Furthermore, ISC presents an on-line mechanism for correcting the statistical deviations created by the embedding step. If the message length is sufficiently short (i.e., it is less than the number of AC nonzero coefficients), ISC transforms useless coefficients in order to restore the original statistical properties characterizing the cover medium.

As an example, if ISC transforms an AC coefficient from -1 into -2, when it encounters the first unused -2, it transforms this value in -1 in order to re-equilibrate the histogram. Naturally, the more information we embed in the cover image, the less ISC can correct the introduced modifications.

Breaking F5 Fridrich and her group presented a steganalytic method that does detect images with F5 content (Fridrich et al., 2002). They estimated the cover image histogram from the stego image and compared statistics from the estimated histogram against the actual histogram.

As a result, they found it possible to get a modification rate that indicates if F5 modified an image. F5 can be defeated because it can only decrement DCT absolute value, giving the chance of predicting the histogram value for the stego image. On the contrary, ISC can increase or decrease DCT absolute values indifferently (see algorithm **Em1**). The decision between these two possibilities are random for default but can also be taken depending on image properties and statistics.

Thus, if the statistical tests used to examine an image for steganographic content are known, ISC is robust to them because ISC uses the remaining redundant bits to correct statistical deviations created by the embedding step, as suggested in (Provos and Honeyman, 2003).

6 ISC FOR IMAGE SEQUENCES

The image based steganographic system illustrated in Fig. 4 requires the receiver (Bob) must posses K (i.e., the key image) in order to get M (i.e., the secret message). If Alice sends Bob the key image K together with the stego image S, Wendy could uncover the steganographic communication simply applying ISC extracting process.

A naïve solution consists in creating a reserved image database shared by Alice and Bob. If Alice and Bob use a new key image for every new message they send each other, ISC is a theoretically secure cryptographic algorithm (a sort of "photographic" one-time pad). Unluckily, sooner or later, Alice and Bob will be forced to reuse a key image already sent (the image database is not infinite).

A reasonable (and more practical) solution is shown in Fig. 8 and 9. Instead of sending a single image, Alice can send Bob a sequence of JPEG images (called *stego sequence* in Fig. 8). In this way, Alice and Bob can communicate each other sharing only a secret password p (that is, a sort of cryptographic symmetric key). In a similar way it is possible to implement the extracting process, as represented in Fig. 9.

The above introduced password p must be shorter than the length of M because p must be as simple as possible (as required by the Kerckhoffs' desiderata (Kerckhoffs, 1883)). Thus the password p will be used as input for a pseudorandom number generator (PRNG) function, in order to produce a message $M(p)$ as long as the message we want to embed (as required by the Vernam cipher).

$M(p)$ together with the images of the stego sequence will be used for generating every key image K_i (see Fig. 8). ISC yields the stego sequence (i.e., a set of stego images S_i), through an iterative process shown in Fig. 8.

Only the first image I_1 of the sequence is sent without any steganographic content.

Bob is able to recover the set of messages sent by Alice without sharing with her any image but only knowing the secret password p (see Fig. 9).

Since p is used as a seed for the PRNG and since p is reused for every new message, the ISC algorithm

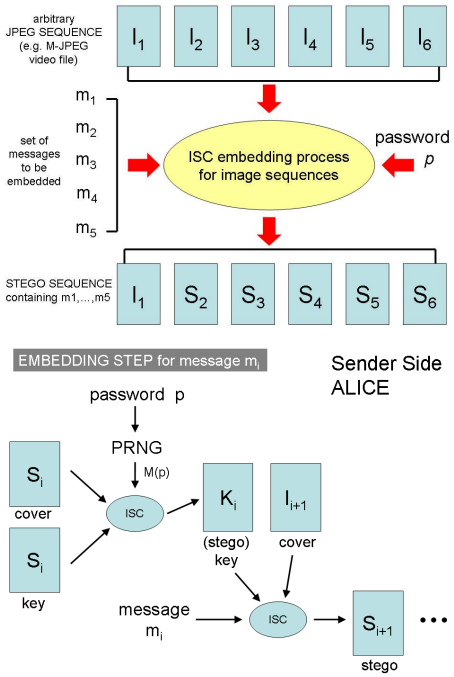

Figure 8: ISC for JPEG sequences (embedding step).

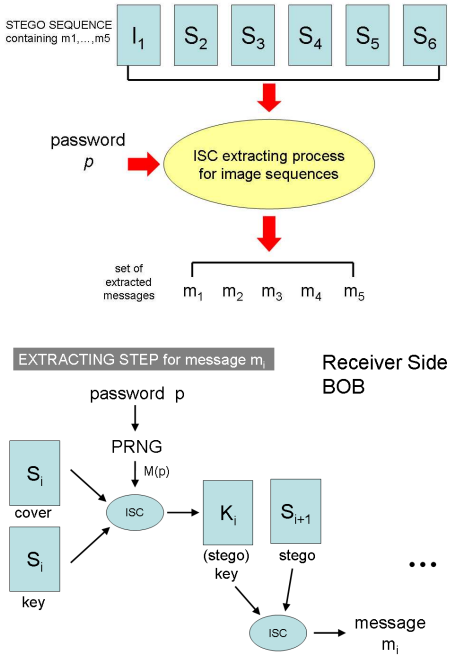

Figure 9: ISC for JPEG sequences (extracting step).

for image sequence is not theoretically secure, but it is equivalent to the Vernam cipher that uses a pseudo-random key.

Likely, since we can assume Wendy has a limited computational power we can assert ISC for image sequence is "unbreakable in practice" (Menezes et al., 1996).

7 Conclusion

In this paper we have presented a novel method for integrating in an uniform model cryptography and steganography. We have proven that the presented ISC algorithm is both an effective steganographic method (we made a comparison with F5) as well as a theoretically unbreakable cryptographic one (ISC is an image based one-time pad).

The strength of our system resides in the new concept of key image. Involving two images (the cover and the key) in place of only one (the cover) we are able to change the cover coefficients randomly. This opportunity does not give a steganalytic tool the chance of searching for a predictable set of modifications.

The proposed approach has many applications in hiding and coding messages within standard medias, such as images or videos. As future work, we intend to study steganalytic techniques for ISC and to extend ISC to mobile video communication.

REFERENCES

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*.

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

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

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

Menezes, A., van Oorschot, P., and Vanstone, S. (1996). *Handbook of Applied Cryptography*. CRC Press.

Provos, N. and Honeyman, P. (2003). Hide and seek: An introduction to steganography. *IEEE SECURITY & PRIVACY*.

Shannon, C. E. (1949). Communication theory of secrecy system. *Bell Syst. Tech. J.*, 28:656–715.

Simmons, G. J. (1984). The prisoners' problem and the subliminal channel. In *Advances in Cryptology: Proceedings of Crypto 83*, pages 51–67. Plenum Press.

Westfeld, A. (2001). F5-a steganographic algorithm: High capacity despite better steganalysis. In *Proc. 4th Int'l Workshop Information Hiding*, pages 289–302.

Westfeld, A. and Pfitzmann, A. (1999). Attacks on steganographic systems. In *Proc. Information Hiding 3rd Int'l Workshop*, pages 61–76.