PARTIAL ENCRYPTION OF IMAGES FOR REAL-TIME APPLICATIONS

Marc Van Droogenbroeck

M.VanDroogenbroeck@ulg.ac.be
Institut Montefiore B-28, Department of Electricity, Electronics and Computer Science,
Sart Tilman, B-4000 Liège, Belgium

ABSTRACT

Multimedia systems mostly base security on a restricted access to services. In the context of real-time imaging applications, this model suffers several drawbacks. Applications become vulnerable to password attacks and once exposed attackers have access to all the data. An alternative consists of the systematic encryption of all data. When dealing with images, this approach is inconvenient because the data needs to be processed in its entirety before users can gain any insight. As well, the decryption task requires large amounts of processing power. This presentation shows how partial encryption can match applications requirements without the overhead of full encryption.

The paper first analyzes several schemes mixing encryption and image encoding. Then we focus on a technique that implements partial encryption of images based on JPEG. The technique is built to meet two major requirements: (1) preserve the overall bitrate, and (2) remain compliant with the JPEG file format. Finally we introduce and elaborate on a new scheme that combines flexibility, multiple encryption, spatial selectivity, self sufficiency, and format compliance. We show how it could fit the needs of real-time applications.

1. INTRODUCTION

Hiding the content of a message when it enters an insecure channel should be common practice. Unfortunately none of the audio-visual compression standards includes any mechanism to convert part of the bitstream into ciphertext prior to transmission.

The encryption process requires an encryption *algorithm* and a *key*. The process of recovering plaintext from ciphertext is called *decryption*. The accepted view amongst cryptographers is that the encryption algorithm should be published, whereas the key must be kept secret (KERKHOFF's law). In practice, the distribution of keys is difficult since keys should be exchanged only when a trusted channel has been established. Furthermore for real-time video systems there are other issues to be addressed: *speed*, *compression efficiency*, and *flexibility*.

Speed depends on the encryption algorithm, and on the type of information that is processed. In real-time video transmission, there are basically two strategies depending on wether or not compression takes place before encryption, since we may assume that compression and encryption are both unavoidable. Compression applied first reduces the bitstream but it offers less secrecy. On the other hand if encryption is applied first, compression is ineffective.

Fortunately there exists an alternative called *selective encryption* that works as depicted in Figure 1 and is the main topic of this paper. The image is first compressed and only

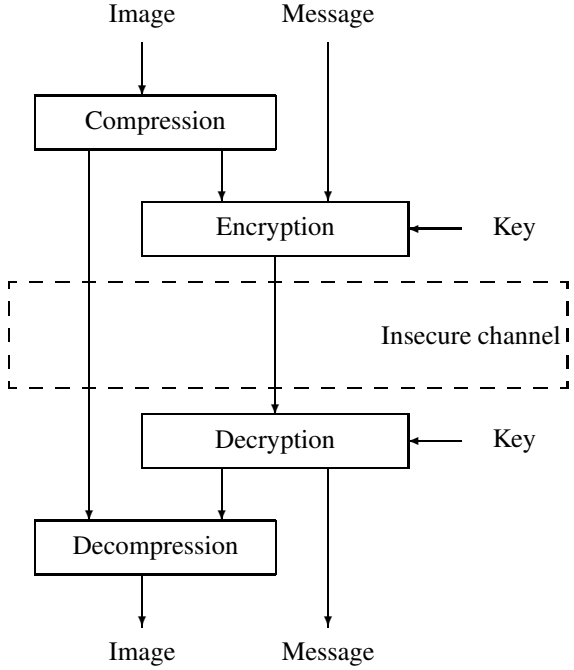


Figure 1: Selective encryption mechanism [1].

parts of the bitstream are encrypted.

An interesting additional feature for real time applications is the ability to use any decoder, even if parts of the bitstream have been encrypted. If bitstream compatibility is targeted, the bitstream should only be altered at places where it does not compromise the compliance to the original format.

Many algorithms for selective encryption have been pro-

posed but they usually require a proprietary decoder which is unsuitable in the field of video transmission where ISO standards dominates the market.

2. SELECTIVE ENCRYPTION OF COMPRESSED IMAGES

2.1. Short review

In the middle of the 90s there have been several papers on the selective encryption of MPEG streams. MAPLES et *al*. [2] proposed an algorithm which encrypts only the Intra (I) frames of an MPEG stream. However AGI et *al.* [3] reported that the selective encryption of the I frames only offers a limited level of security, due mainly to the presence of blocks coded in intra mode in P or B frames, but also to the high correlation of P and B frames when they correspond to the same I frame. This scheme is subject to cryptanalysis, a common problem when compression occurs prior to encryption.
Alternative encryption techniques were developed by other authors. In particular, several techniques have been proposed for the encryption of DCT based coded image. A method called *zig-zag permutation* was originated by TANG [4]. Although this scheme offers more security, it increases the overall bit rate.
Another algorithm, developed by QIAO and NAHRSTEDT, is based on the frequency distribution of pairs of two adjacent bytes in an MPEG bitstream [5]. As proven by the authors, this algorithm provides overall security, and size preservation, but does not meet the requirements of visual acceptance and bitstream compliance.
Other methods have been proposed recently (see [6] for a recent view) but they fail to achieve all of the following *requirements*:

[visual acceptance] part of information may be visible but the encrypted image should look noisy,

[selective encryption] encryption occurs after compression and leaves parts of the bitstream unencrypted,

[constant bit rate] encryption should preserve the size of the bitstream, and

[bitstream compliance] the encryption step should produce a *compliant bitstream* according to the chosen format definition.

Researchers have shown that selective encryption is not restricted to MPEG encoded images. For example POMMER et *al.* [7] and NORCEN et al. [8] have proposed techniques for the selective encryption of wavelet packet subband structures and JPEG 2000 respectively.

2.2. A method for the selective encryption of JPEG images

Because of its widespread use, MPEG was the primary focus for selective encryption. But since MPEG-2 was developed with video broadcasting in mind, selective encryption of MPEG streams will rely on an efficient mechanism for key distribution. An additional difficulty results from the high correlation between frames. MPEG-2 removes many redundancies contained in a video stream but an encoder leaves a residual correlation that affects secrecy and eases cryptanalysis.
Therefore we concentrate on the JPEG standard which is more likely to be used in point to point transmission.
The method described hereafter was first proposed in [1]. We will discuss extensions in Section 3.

2.2.1. Short description of a compliant selective encryption of JPEG images

In JPEG, the HUFFMAN coder aggregates zero coefficients into runs of zeros. In order to approach the entropy, it also uses symbols that combine the run of zeros with magnitude categories for the non-zero coefficients that terminate the runs. 8-bit code words are assigned by the HUFFMAN coder to these symbols. These code words are followed by appended bits that fully specify the sign and magnitude of the non-zero coefficients. We decided to leave the code words but to encrypt the appended bits. The reasons are that code words are essential for synchronization and that it does not make much sense to replace zero coefficients by non-zero coefficients. Therefore it is essential to preserve the run values. Also, it is not effective to encrypt DC coefficients because they carry important visible information and are highly predictable. Our algorithm encrypts appended bits corresponding to a selected number of AC coefficients. This set of coefficients is the same for each DCT block.

3. EXTENSIONS TO SELECTIVE ENCRYPTION

3.1. Multiple selective encryption

If there is a single copyright owner, called *owner* hereafter, he will apply the selective DCT encryption algorithm to a subset C_1 of coefficients of the JPEG image f with a key $k1$. The resulting image is $g = E_{k1}(f)$.
At the receiver side the decryption algorithm D is able to recompute f if and only if $k1$ is known: $f = D_{k1}(E_{k1}(f))$. Alternatively we could have used an encryption technique based on a public key and a private key since our technique can accommodate any encryption process.
If there is a second owner, he should be able to choose a subset C_2 of DCT coefficients and to encrypt them with

his own key $k2$ as well. The image sent over the network is then $h = E_{k2}(E_{k1}(f))$. We named this principle *"multiple selective encryption"* when $C_1$and C_2 are chosen independently.

3.2. Over-encryption

When C_1 intersects with C_2 ($C_1 \cap C_2 \neq \emptyset$), coefficients encrypted twice are more sensitive to attacks and it is recommended to use a technique called *over-encryption*, that corresponds to $E_{k1}(D_{k2}(E_{k1}(f)))$, as proposed by TUCHMAN [9]. According to SCHNEIER [10], over-encryption offers better performances than $E_{k2}(E_{k1}(f))$.

3.3. Generalized selective encryption scheme

In a generalized scheme we may want to provide:

1. *Flexibility*.
   A user should be able to tune the level of encryption, i.e. the subset of DCT coefficients.

2. *Multiplicity*.
   Suppose C_1 and C_2 are informations that owners 1 and 2 will encrypt. $(C_1 \cup C_2) \backslash (C_1 \cap C_2)$ are encrypted independently but $C_1 \cap C_2$ is preferrably over-encrypted. Figure 2 shows an image encrypted by owner 1 (b), and the same image further encrypted by owner 2 (c). Note that:

   (a) if $C_1 \cap C_2 = \emptyset$, multiplicity is nothing but a parallelization.

   (b) there is no need for C_1 and C_2 to be fixed through the whole encryption process. These coefficient sets could change randomly over time to enhance secrecy.

3. *Spatial selectivity*.
   It is often not required to encrypt the whole image. For example in "head and shoulder" sequences, it might be sufficient to encrypt the "head". Encrypted image zones are marked in a binary map, called *selection map,* and there is one bit per 8×8 block.
   By default a selection map is uniform, but when several owners implement spatial selectivity there are as many selection maps as owners. Figure 2(d) illustrates spatial selectivity.

4. *Self sufficiency* and *compliance*.
   C_1, C_2 and selection maps are additional informations a decoder needs to decrypt the image. It is possible to embed them into an image as described by FRIDRICH [11] although at the expense of a bitrate increase.

The scheme drawn in Figure 3 implements all these properties.

Data is split into several slices. Some slices are left unmodified (this is referred to as part (1) on Figure 3) while other slices (2) are processed by encryption blocks ordered into a sequence S. Slices are encrypted by known algorithms (RSA, Rijndael, etc) with different keys. Note that all encryption blocks may be different. However if a slice is encrypted twice with a similar algorithm, it should be over-encrypted.

The encryption sequence S and the selection map, which states which slice is encoded, used by each encryptor have to be known to the decoder or otherwise specified into a information stream I_1, itself encrypted into I_2. The type of algorithms has to be known or provided into a stream I_3 as well.

All data, whether encrypted or not, are then reassembled into a format compliant stream. As far as merging is concerned, the amounts of bits prior to and after encryption are the same. Since encrypted slices are put in place of the original data and the number of encrypted bits is low, substitution is fast to accommodate to speed requirements of real-time processing.

After merging, there follows 2 embedding steps.

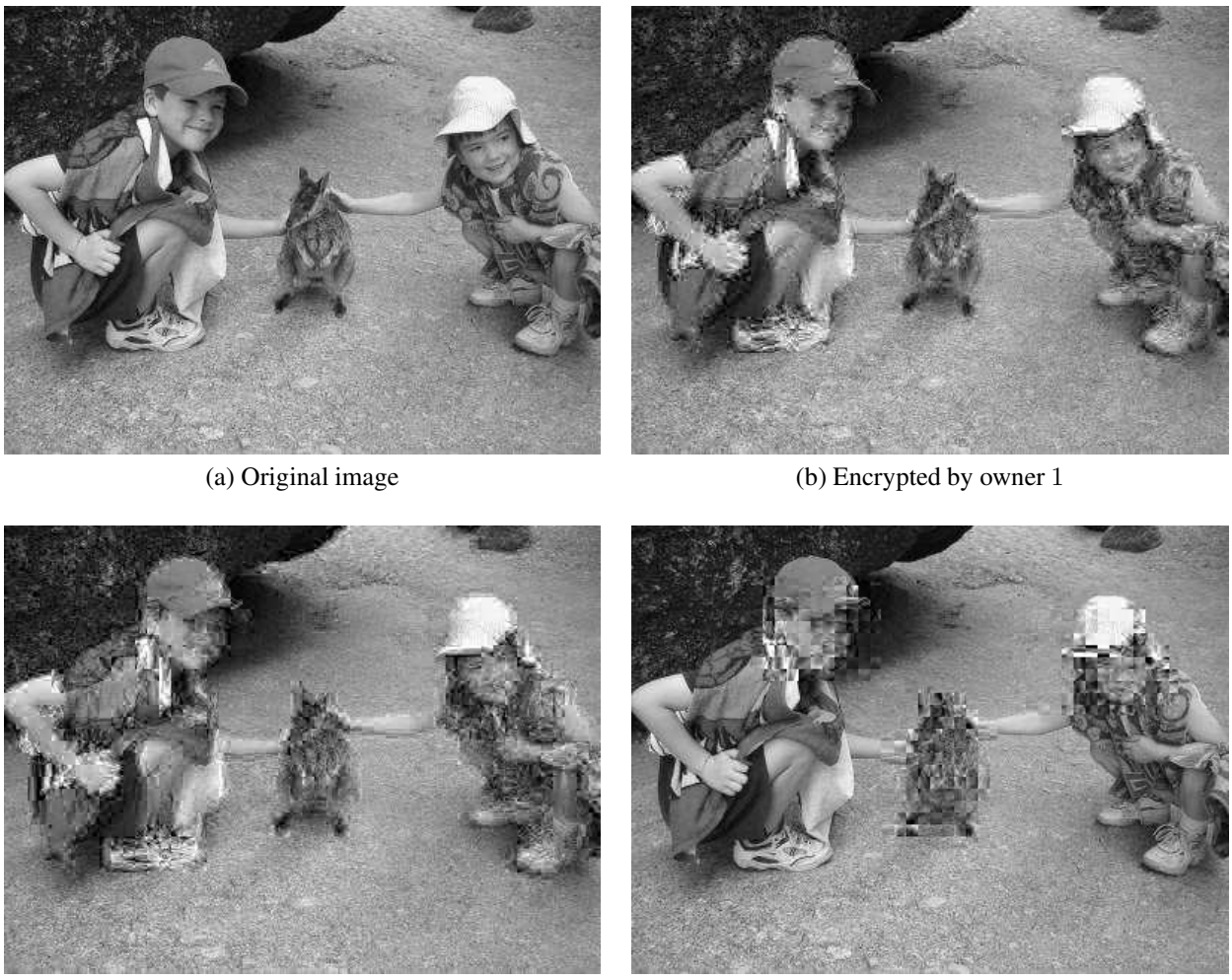
1. The first step embeds all the information related to the encryptors (algorithms, keys in the case of public key cryptography, etc). Lossless data embedding techniques are used; there are several techniques for embedding regardless of the data formats but as a general rule the bitstream size is increased. Subsequently the header might have to be adapted to take changements into account.

2. The second step embeds the selection map and associated data like used parameters. The technique used for embedding is similar to the previous one.

Both embedding steps are complex and time consuming because the format is altered at several places (it is not just a substitution). However, if enough knowledge is available to the receiver, embedding can be skipped without any further security weakness.

4. CONCLUSIONS

In this paper, we propose a generalized scheme to *selectively encrypt* an image. The scheme offers several advantages: flexibility, multiplicity, spatial selectively and format compliance. Secrecy results from a tradeoff between processing power and speed, but real-time processing is achievable.



(a) Original image

(b) Encrypted by owner 1

(c) Encrypted by owner 1 and owner 2

(d) Locally encrypted image

Figure 2: Flexible multiple encryption and spatial selectivity.

Acknowledgments

The author is grateful to Hugues TALBOT for his comments.

5. REFERENCES

[1] M. Van Droogenbroeck and R. Benedett, "Techniques for a selective encryption of uncompressed and compressed images," in *ACIVS Advanced Concepts for Intelligent Vision Systems*, Ghent, Belgium, September 2002, pp. 90–97.

[2] T. Maples and G. Spanos, "Performance study of a selective encryption scheme for the security of networked, real-time video," in *Proceedings of the 4th International Conference on Computer Communications and Networks*, Las Vegas, Nevada, September 1995.

[3] I. Agi and L. Gong, "An empirical study of secure MPEG video transmission," in *Symposium on Network and Distributed Systems Security*, 1996.

[4] Lei Tang, "Methods for encrypting and decrypting MPEG video data efficiently," in *ACM Multimedia*, 1996, pp. 219–229.

[5] Lintian Qiao and Klara Nahrstedt, "Comparison of MPEG encryption algorithms," *Computers and Graphics*, vol. 22, no. 4, pp. 437–448, 1998.

[6] A. Eskicioglu, "Multimedia content protection in digital distribution networks," Document available on the Internet, 2003.

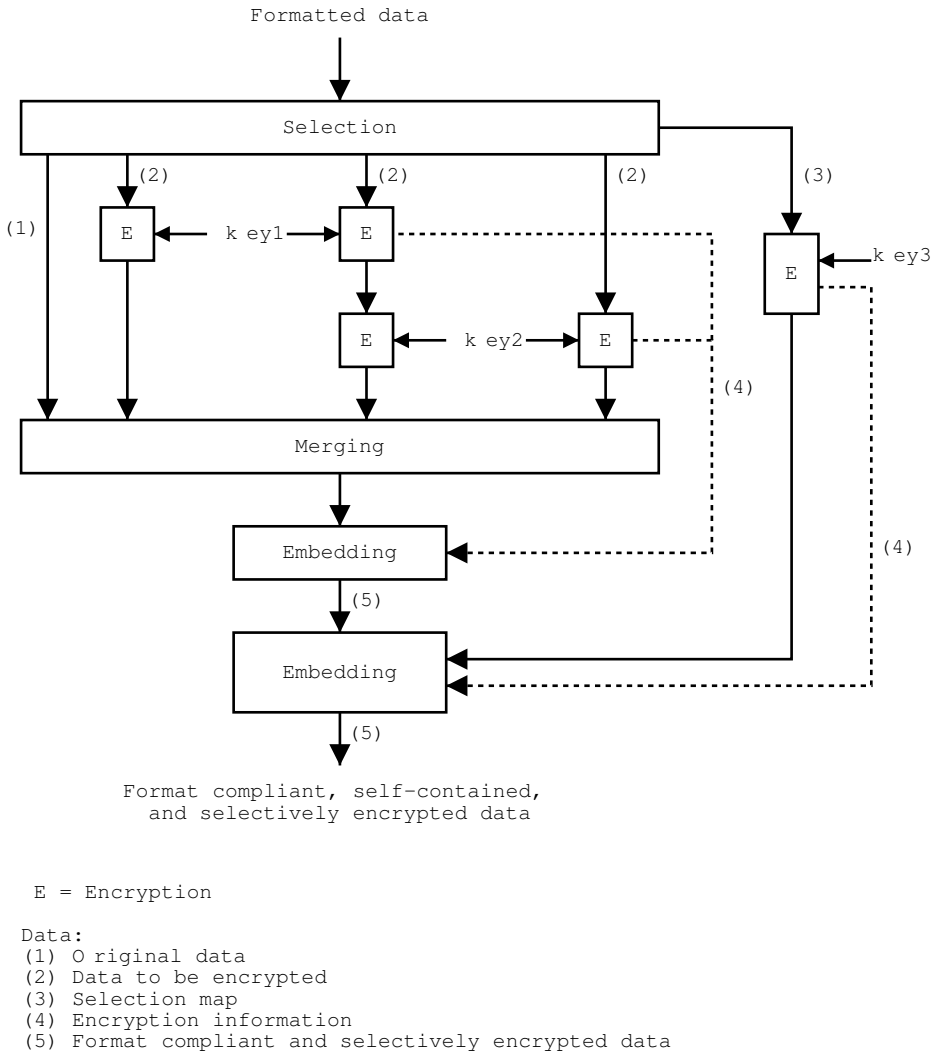


Figure 3: Self-sufficient selective encryption unit.

[7] A. Pommer and A. Uhl, "Selective encryption of wavelet packet subband structures for obscured transmission of visual data," in *Proceedings of the 3rd IEEE Benelux Signal Processing Symposium (SPS 2002)*, Leuven, Belgium, 2002, pp. 25–28.

[8] R. Norcen and A. Uhl, "Selective encryption of the JPEG2000 bitstream," in *Proc. IFIP TC6/TC11 7th Joint Working Conference on Communications and Multimedia Security (CMS 2003), Lecture Notes in Computer Science, volume 2828*, 2003, pp. 194–204.

[9] W. Tuchman, "Hellman presents no shortcut solutions to DES," *IEEE Spectrum*, vol. 16, no. 7, pp. 40–41, July 1979.

[10] B. Schneier, *Applied cryptography*, John Wiley & Sons, second edition, 1996.

[11] J. Fridrich, M. Goljan, and R. Du, "Lossless data embedding for all image formats," in *Proc. SPIE Photonic West, Vol. 4675, Electronic Imaging 2002, Security and Watermarking of Multimedia Contents*, San Jose, California, January 2002, pp. 572–583.