Published in X. Zhuang and W. Gao, Eds, *2007 International Conference on Multimedia & Expo*, pp. 1683–1686, IEEE Press, 2007.

CRYPTANALYSIS OF A VIDEO SCRAMBLING BASED ON SPACE FILLING CURVES

Ayoub Massoudi, Frédéric Lefèbvre, Marc Joye

Thomson R&D France
Technology Group, Corporate Research, Security Laboratory
1 avenue de Belle Fontaine, 35576 Cesson-Sévigné Cedex, France

ABSTRACT

In this paper, we study the security of an image scrambling algorithm based on Space-Filling Curves (SFC). A random SFC is a pixel permutation that changes the scanning order without changing pixels values. Few attacks were reported in the literature. A ciphertext-only attack when a different scan is generated for each frame in the video was proposed. The success of this attack heavily depends on the statistics of the plain frame. In addition, scanning each frame with a different SFC is not adapted to compression as many compression algorithms exploit temporal redundancy.

We revisit the security properties of this scrambling technique and propose an efficient and low-cost chosen-plaintext attack when the same SFC is used for each frame in the video.

1. INTRODUCTION

Securing video signals is becoming a more and more important topic for video producers and distributors. Additional constraints need to be considered when tackling video encryption. Indeed, many encryption standards dedicated to data (such as DES, AES, ...) are computationally demanding and are not suited to the huge size of videos.

In [1], Matias and Shamir proposed a symmetric video scrambling technique based on Space-Filling Curves (SFC). An SFC is a Hamiltonian path that traverses every pixel of a frame exactly once. From a security point of view, finding all possible SFCs in a graph is an NP-problem. At a given pixel in the frame, only four transitions are allowed (left, right, up and down). Much of the spatial correlation between pixel values is preserved and thus little bandwidth expansion is introduced. It has been shown that the frame autocorrelation increases after scrambling [1]. In addition, for an image of size $M \times N$, this scrambling technique allows about $2^{0.5573MN}$ possible SFCs [2]. As an example, for a given image of size 64×64 pixels, we obtain about 2^{2283} different SFCs. Brute force attacks become infeasible.

Few works have been proposed about designing efficient scrambling algorithms that guarantee enough security, compression friendliness and speed efficiency. One solution is selective encryption where only part of the data is encrypted (see [3] for a good survey on this topic). SFC scrambling is one of the very early examples of selective encryption technique that only modifies pixel positions. Although very fast, we show that it is not secure enough. Nevertheless, Space-Filling Curves still remain an interesting topic as they better exploit the two-dimensional locality of the image signal. Therefore, it is a useful feature that can be used to design compression friendly scrambling algorithms. In the rest of this paper, we show that SFC scrambling cannot be used alone and needs additional treatment to offer a sufficient level of security.

Fig. 1. Example of an SFC.

2. RELATED WORKS

In [4], Bertilsson *et al.* propose an attack for the algorithm described in [1] where each frame in the video is scrambled by a different SFC. Their attack targets still sequences in the video with no or very little change. The same frame is displayed about 25 times per second. Thus there are multiple scans of the same frame.

The basic idea of the attack is based on some assumptions:

- There exist pairs of adjacent pixel values that occur only once in the frame called "unique pairs" [4];
- Each frame in the sequence is scrambled by a different SFC.

The attack comprises three steps:

1. First, the scrambled frames are analyzed along the still sequence to get some unique pairs. Indeed, if a sufficient number of SFCs is available, hopefully a large number of pairs are covered.

2. Next pairs that occur in many SFCs or several times in the same SFC are rejected.

3. Finally, unique pairs that are close to each other (a maximum distance is defined) are identified.

To this purpose, Lee distance is used (see Figure 2). The Lee distance between two unique pairs UP_1 and UP_2 (represented by the arrows in Figure 2) is given by:

$$d_{\text{Lee}}(UP_1, UP_2) = n_v + n_h$$

where n_v and n_h are the number of pixels between UP_1 and UP_2 in the vertical and horizontal directions, respectively, in the original image. This distance is not known by the attacker, the aim of the third step in the attack is to determine the Lee distance between unique pairs identified in the two first steps. This is given by:

$$d_{\text{Lee}}(UP_1, UP_2) = \min_{SFCs} \text{NP}(UP_1, UP_2)$$

where $\text{NP}(UP_1, UP_2)$ is the number of pixels visited by the SFC between UP_1 and UP_2. The minimum value of $\text{NP}(UP_1, UP_2)$ over the set of available SFCs containing both unique pairs gives the Lee distance.

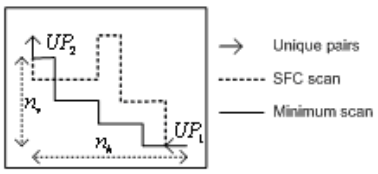


Fig. 2. Computing Lee distance between two unique pairs.

Then, once we get unique pairs and Lee distances, we consider close unique pairs by examining local neighborhoods of each of them. A local reconstruction of the original content is possible around each unique pair. An assembling is then required between close unique pairs to recover pieces of the original content.

This attack showed a leakage in the scrambling algorithm. However, it is based on strong assumptions: the still sequence has to be long enough to contain a relevant number of SFCs; this is required to minimize the false unique pairs. In addition, this attack is image content dependent; unique pairs are concentrated around high frequency features and regions with poor edge distribution are difficult to reconstruct.

3. SECURITY OF THE SFC SCRAMBLING TECHNIQUE

Shannon [5] suggested that perfect secrecy requires that the ciphertext should not give out any information about the plaintext. Given any pair (P, C) of plaintext-ciphertext (C is not necessarily the scrambling of P), we should have:

$$\Pr(P|C) = \Pr(P) \ .$$

This property is not satisfied in [1]. Given an original frame P and a scrambled frame C, if the luminance (i.e., gray level) histogram of P is different from the one of C, then $\Pr(P|C) = 0$. Likewise, if they have the same luminance histogram, then $\Pr(P|C) \gg \Pr(P)$. Note that the histogram can be replaced by any other global statistical characteristic of the frame pixel values (mean, variance, ...). This leakage is due to the fact that the scrambling consists only of a permutation of the pixel positions. The scrambling preserves global statistics of the image. This leads to another leakage: no diffusion effect is obtained, changing a pixel value in the original frame will only affect the same pixel value in the scrambled frame. This makes known-plaintext and chosen-plaintext attacks applicable.

Finally, in [1], the starting point of any SFC is the upper left pixel of the original frame. Our attack is more general and considers random SFCs with randomly chosen starting point (see Figure 3).

In the next section, we present a chosen-plaintext attack requiring few chosen examples compared to brute force attack.

4. CHOSEN-PLAINTEXT ATTACK

The attack[1] we propose relies on the following assumptions:

- The attack does not depend on the algorithm used to compute the SFC;
- The SFC starting point is unknown;
- The video is scrambled by scanning each frame with the same SFC.

Given a video V with a resolution $M \times N$, and G gray levels per frame — the video being scrambled with a unique secret SFC_{sec}, the goal is to recover SFC_{sec}. Our attack consists in using particular chosen frames. We can suppose that the attacker has been able to insert particular chosen frames in the original video. Then, by examining the scrambling result of the chosen frames, he is able to fully reconstruct SFC_{sec}. Chosen-frame number i will be denoted by I_i and its scrambling under SFC_{sec} will be denoted by C_i.

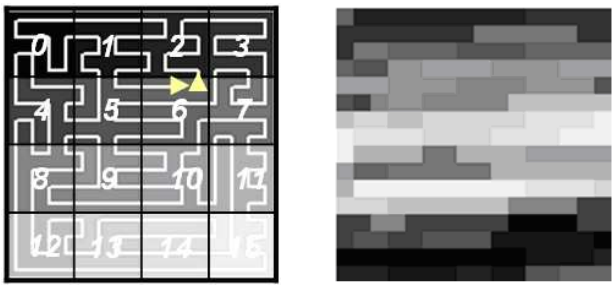
[1]For illustration purposes, we focus on the Y-channel (luminance).

The attack comprises two steps:

1. Locate the SFC starting point;
2. Reconstruct the complete SFC.

4.1. Location of the starting point

To locate the starting point, we subdivide the $M \times N$ frame space into G squared sections in a grid-like way. Each section is then filled with a uniform and different gray level. We consider an incremental filling. It sets the top left section to 0 and then gradually increments the gray level by 1 for each next section. Figure 3(a) shows a subdivision example with a frame size of 16×16, with 16 gray levels. So we obtain 16 sections, each section has a uniform gray level. In the same figure we represent SFC_{sec}.



(a) I_0 (b) C_0

(c) I_1 (d) C_1

Fig. 3. I_0, C_0, I_1 and C_1 with $M = N = 16$, $G = 16$. C_0 is used to locate the starting point with accuracy of $1/16$ and C_1 is used to locate the starting point with accuracy of 1.

Note that in the scrambling of the frame I_0 in Figure 3(b), the upper left pixel (at position $(0, 0)$) of C_0 will have the gray level of the section containing the starting point. This will locate the starting point with accuracy $G/(MN)$ and allows selecting an initial section that we will call S_0 (in the example, it is section 6). We want to determine the exact location of the starting point. We build new chosen frames that will allow tracking the starting point. This is an iterative process (see Figure 4). At iteration $i+1$, each section obtained at iteration i is subdivided into G squared sections that are filled as described above with gradual gray levels. We obtain I_1, its scrambling is C_1. Figure 3(c) shows I_1. By computing C_1, we get the starting point location with more accuracy, namely $G^2/(MN)$. We need as much iterations as necessary to locate the exact location of the starting point. This is achieved when the unit section S_i has a size smaller than or equal to G. If we let n denote the minimum number of needed iterations (i.e., chosen frames), we must have:

$$G^n \geq M \times N \implies n = \left\lceil \frac{\log MN}{\log G} \right\rceil .$$

In the example shown in Figure 3, we need only $n = 2$ chosen frames (I_0 and I_1) to locate unambiguously the starting point.

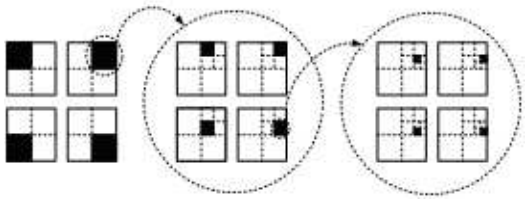


Fig. 4. Tracking of the starting point.

In [1], the starting point is always the upper left pixel of the original frame and so can easily be identified. Moreover, only one chosen frame is needed (I_n), regardless of the video resolution and the available gray levels.

The next section explains how to reconstruct the secret SFC from the knowledge of the starting point.

4.2. Experimental results: SFC reconstruction

In the previous section, we located the starting point of SFC_{sec}. All we need now is to analyze the scrambling of the last chosen example C_n to reconstruct the secret SFC. The starting point is the upper left pixel of C_n. Note that possible transitions in the same section in I_n are restricted and well known. This is due to the *a priori* knowledge of the relative pixels positions and values in I_n. Transitions between sections are well detected. In the following, we present a simple example to illustrate the attack.

Assumptions

- What the attacker knows: the scrambling of the chosen frames $\{C_i\}_{i=1,\ldots,n}$.
- What the attacker wants to know: the secret SFC used to scramble the chosen frames, SFC_{sec}.

In this example we consider a video of resolution 16×16 (for simplicity), with $G = 16$ gray levels. The aim of the attack is to reconstruct the secret SFC represented in Figure 3. In this section, we suppose the starting point already detected. According to Figure 3, the starting point is located at position $(11, 4)$. The last chosen example used here to reconstruct is

illustrated in Figure 3(c). The 16 sections are numbered from left to right and from top to bottom. The first section (upper left) is numbered 0, the last one (right bottom) is numbered 15. The unit section size here is $G = 16$.

From any point in the frame, restricted transitions are allowed in the SFC (left, right, up, down). Given the value of a pair of successive pixels (P_1, P_2) in C_n (following the raster scan), we can easily deduce the transition in the secret SFC that went from P_1 to P_2. If we note $L(P_1)$ and $L(P_2)$ the gray levels of respectively P_1 and P_2, and if we note $M' \times N'$ one section size in C_n, the transition is computed as follows:

- If $L(P_2) = L(P_1) + 1$ then transition = right;
- If $L(P_2) = L(P_1) - 1$ then transition = left;
- If $L(P_2) = L(P_1) + M'$ then transition = down;
- If $L(P_2) = L(P_1) - M'$ then transition = up;
- If $L(P_2) = L(P_1) - M' + 1$ then transition = right (to the right section);
- If $L(P_2) = L(P_1) + M' - 1$ then transition = left (to the left section);
- If $L(P_2) = L(P_1) - (N' - 1)M'$ then transition = down (to the section below);
- If $L(P_2) = L(P_1) + (N' - 1)M'$ then transition = up (to the section above).

Note that the first four transitions are intra-section transitions while the last four ones are inter-section ones. Based on this, the different steps of the SFC reconstruction are illustrated in Table 1 below (the secret SFC is illustrated in Figure 3).

In Table 2, we give the processing times of our chosen-plaintext attack (column 4) and authorized decryption (column 5), the processing time indicates the time required to attack/decrypt one video frame. The experiments were conducted on a Pentium 4 running at 2.80 GHz.

We observe that the attack is very fast and the overhead compared to authorized decryption is very low. Note also that in our example only 2 chosen frames were used instead of about 5×10^{10} SFCs computations required in a brute force attack.

Table 1. Example of an attack (transitions).

Pixel #	Section	Point_in	Point_out	SFC_transition
1	6		$(11, 4)$	
2	2	$(11, 3)$		Up
3	2	$(10, 3)$		Left
...	...	...	...	
11	3		$(11, 0)$	Right
...	...	...	...	
256	6	$(10, 4)$		Right

Table 2. Number of chosen frames required and processing times.

Image size ($M \times N$)	Gray levels (G)	# chosen frames	Attacking time (ms)	Decryption time (ms)
128×128	256	2	165	70
256×256	256	2	245	90
512×512	256	3	350	150

5. CONCLUSION

In this paper, we presented an efficient chosen-plaintext attack on an SFC based scrambling technique. The attack allows important complexity reduction compared to brute force attacks. For most video resolutions, with 256 gray levels, only 2 or 3 chosen examples are needed. Nevertheless, SFC based scrambling presents interesting properties from the compression point of view and our future works will focus on how to modify it to improve its security.

Acknowledgments We are grateful to Olivier Courtay for some help with the figures.

6. REFERENCES

[1] Y. Matias and A. Shamir, "A video scrambling technique based on space filling curves," in *Advances in Cryptology – CRYPTO '87*. 1987, vol. 293 of *LNCS*, pp. 398–417, Springer-Verlag.

[2] H. Orland, C. Itzykson, and C. de Dominicis, "An evaluation of the number of Hamiltonian paths," *J. Physique Lett.*, vol. 4, pp. 353–357, Apr. 1985.

[3] X. Liu and A. Eskicioglu, "Selective encryption of multimedia content in distributed networks: Challenges and new directions," in *IASTED Communications, Internet & Information Technology (CIIT)*, Nov. 2003.

[4] M. Bertilsson, E.F. Brickell, and I. Ingermarsson, "Cryptanalysis of video encryption based on space-filling curves," in *Advances in Cryptology – EUROCRYPT '89*. 1990, vol. 434 of *LNCS*, pp. 403–411, Springer-Verlag.

[5] C.E. Shannon, "Communication theory of secrecy systems," Declassified Report, Sept. 1, 1946.