**CRYPTANALYSIS OF VIDEO ENCRYPTION
BASED ON SPACE-FILLING CURVES**

Michael Bertilsson¹⁾ Ernest F. Brickell²⁾
Ingemar Ingemarsson¹⁾

¹⁾Dept of Electrical Engineering
Linköping University
581 83 LINKÖPING, Sweden

²⁾Sandia National Laboratories
Albuquerque
NM 87185, USA

ABSTRACT

In this paper we cryptanalyze a method for enciphering images constructed by Adi Shamir and Yossi Matias. The enciphering is done using space-filling curves. We present an algorithm doing the cryptanalysis. We also present results achieved using the algorithm on a part of an image.

I. INTRODUCTION

In CRYPTO'87 Adi Shamir and Yossi Matias presented a video scrambling technique [1]. This technique uses space-filling curves (SFC) to encipher images and sequences of images. Shamir and Matias state that standard cryptographic techniques are inadequate for enciphering of images. There are three basic reasons for this claim.

- 1 The transmitted signal is analog
- 2 The transmission rate is very high
- 3 The allowable bandwidth is limited

The main idea presented in [1] is to store an image in a frame buffer and then scan the image along a pseudo-random SFC. Several algorithms generating SFC:s are presented in [1]. Independently of the algorithm used they get SFC:s looking something like the one in Figure 1.

Figure 1

We show in this paper that it is not a good idea to use SFC:s for enciphering sequences of images. This is true independently of the algorithm used to generate the SFC. Our algorithm only use the fact that the enciphering is done using SFC:s and is not dependent on the algorithm used to generate the SFC. We assume that the encipherer chooses a new SFC for every frame.

This is not an attack on the use of SFC:s for enciphering still images. Our method is not possible to use on still images.

II. PROPERTIES OF IMAGES

In this section we give some properties of images and sequences of images that is used in the cryptanalysis.

Transmission of sequences of images is usually done at a rate of approximately 25 frames per second. This rate is needed when there is a lot of motion in the sequence. But sometimes there is no motion in the sequence. When there is no motion the same image is transmitted in a number of consecutive frames. We assume that the encipherer chooses a new SFC for every new frame. This means that the same image will be scanned by different SFC:s if there is no motion in the sequence.

Pixel values are non-uniformly distributed in a quantized image. This is also valid for adjacent pixels. In particular there are some pairs of adjacent pixel that will occur only once in the image. The pairs occurring only once will be called unique pairs.

We define the distance between two points p_1 and p_2 in a square lattice as follows

$$d(p_1, p_2) = p_v + p_h$$

where p_v is the number of points between p_1 and p_2 vertically and p_h the number of points between them horizontally. This distance is similar to the Lee distance used in coding theory.

Using this distance we can calculate a maximum possible distance between unique pairs even if we only have a SFC scan of the image. Suppose that we have two unique pairs, up_1 and up_2 in a SFC scan. We count the number of pixels, $n(up_1, up_2)$, between the two unique pairs. This is the same as the maximum possible distance between the two unique pairs in the original image. If we have several SFC scans of the same image we can get a better result. We then calculate the maximum possible distance, d_{max} , between the two unique pairs as

$$d_{max}(up_1, up_2) = \min n(up_1, up_2)$$

where the min is over all the SFC scans containing both up_1 and up_2 . We define $n(up_1, up_2) = 0$ if the two unique pairs have one pixel in common.

III. CRYPTANALYSIS

The cryptanalysis is divided into several steps which can be performed separately.

First we have to find the unique pairs in the different scans of the image. We are faced with two problems when we do this. Both problems come from the fact that we do not get all the pairs of pixels in the image when we are scanning it using a SFC. We only get approximately half of the pairs in the original image in our SFC scan.

There is a risk that we miss a unique pair in a SFC scan. We can solve this problem if we have many scans of the same image. If the same image is transmitted several times and the encipherer selects a new SFC for every frame we have many scans of the same image. Since it is probable that a unique pair occurs in at least one of the scans we will find most of the unique pairs.

Some pairs of pixels will occur only a few times in an image. If we miss all of these pairs but one we get a pair occurring only once in this specific SFC scan. This means that in this scan the pair will look like a unique pair. This problem is also solved using the different scans we have of the same image. We can detect these false unique pairs in two different ways. First it is likely that this false unique pair will occur two times or more in at least one of the other scans and if so it is removed from the unique pairs. Secondly a pixel belonging to a unique pair can have only four values on the adjacent pixels, one of which is the value of the other pixel in the unique pair. So if a pixel belonging to a possible unique pair has five or more different values on its adjacent pixels the pair can not be unique.

The second step in the cryptanalysis is to synchronize the different scans locally around the unique pairs. Different SFC:s scan the surrounding of the unique pairs in different ways. Eight of these are interesting in this step in the cryptanalysis. These are shown in Figure 2a and 2b.

Figure 2a

Figure 2b

If we have the leftmost scan in Figure 2a in one SFC and the leftmost scan in Figure 2b in another we get the values of the two pixels above the unique pair. If we have the rightmost scan in 2a and 2b in two different SFC:s we get the values of the pixels below the unique pair. Similarly for the two scans in the middle. If we can get the value of both the pair above and the pair below the unique pair we are usually able to get the values of all the six pixels surrounding the unique pair, see Figure 3. This is possible if we have four different pixel values on pixels adjacent to each of the two pixels in the unique pair.

Figure 3

When we have constructed the small surroundings around the unique pairs we try to put these surroundings together. We search the marked pairs in Figure 4 looking for unique pairs. This is done for all the surroundings.

If we among these pairs find a unique pair we join these two surroundings. This is repeated until the surroundings around all the unique pairs have been searched. This gives us a set of pieces of the image.

Figure 4

We would like to know how far apart these different pieces are. The exact distance is not possible to calculate. But we can calculate the maximum possible distance between unique pairs, see section 2. Using this and the fact that we know to which piece the unique pairs belong we calculate the maximum possible distance, D_{max} , between two pieces $p1$ and $p2$. This is done using

$$D_{max}(p1, p2) = \min D_{max}(upi, upj)$$

where min is over all distances between a unique pair belonging to $p1$, upi , and a unique pair belonging to $p2$, upj . We also know between which unique pairs in the two pieces the distance holds.

Before we use the distances we try to make each piece created above larger. We use the different scans of the same image. The method is explained using Figure 5.

We have the scan showed in Figure 5a and the piece shown in Figure 5b. Pixels 1 and 2 form a unique pair. The pixels both above and below pixel 1 in Figure 5b are different from pixel 3. So the only place where we can put pixel 3 is in the empty place to the right of pixel 2. This means that we can expand the piece a little bit. This is repeated for all the scans and all the pieces. After this step the pieces will be a little larger which makes it easier to fit them together.

Figure 5a

Figure 5b

Now we use the distances between pieces to join the different pieces. A typical situation when the distance between pieces is zero is shown in Figure 6.

Figure 6

In Figure 6 we have two different pieces. The pieces have $D_{\max}(p_1, p_2) = 0$. The pixel marked with a * in the two pieces is common to the two unique pairs which are at distance zero. Knowing this we have eight different ways to fit the pieces together, rotation 90 degrees four times and mirroring. Hopefully only one of these eight is possible. If we have more than one possible way to join two pieces we do not join them. In this case only one of them is possible so we can join the two pieces into one. This way we try to join all the different pieces with mutual distance zero. We also try to do this for larger distances. Since we only know the maximum possible distance between the pieces we have to try to fit the pieces together for all distances shorter than the one calculated. Because of this we do not try to fit pieces together with mutual distance larger than five.

The last two steps can be repeated a desired number of times alternating between the two. This can be done both for the same distance and for increasing distances in the last step.

IV. LOCATION OF UNIQUE PAIRS

One important question is where the unique pairs occur in an image. Suppose we are able to create one large piece consisting of all the unique pairs. Are we then able to see what the image looks like? If not it is not interesting to do the cryptanalysis described in section 3. In this section we claim that you actually get a good idea of what the original image looked like if you can create a piece consisting of all unique pairs.

In a part of the image the pixels may have the same or almost the same values, i.e. low spatial frequency. In this part of the image there will be few or no unique pairs. On the other hand in a part of the image with edges (high spatial frequencies) we have many unique pairs. This is illustrated in Figure 7. In this Figure we show the original image "Lenna" and an image showing the unique pairs in "Lenna". We see that we find the unique pairs where there is a lot of "information" in the image.

Figure 7

V.RESULTS

Using the algorithm described in section 3 we tried to cryptanalyze a part of the image "Lenna". The size was 64x64 pixels. We used 25 scans of the same image. The result from the cryptanalysis was the four large pieces shown in Figure 8 and some smaller pieces. From the calculation of distances we knew that piece 1 was close to piece 2 and piece 2 was close to piece 3. We also knew that piece 4 was distant from the other three pieces. The small pieces was not used since most of them consisted of only one unique pair.

Figure 8

We showed a person the four pieces in Figure 8. We also told her which pieces should be close to each other. Piece number four was not used. Knowing only this she was able to put piece number 1, 2 and 3 together into the piece shown in Figure 9. The size of this piece is 64 pixels from top to bottom and approximately 45 pixels from left to write.

Figure 9

VI. CONCLUSIONS

We have shown that space-filling curves are unfit for secure enciphering of sequences of images. In the experiment on a part of the image "Lenna" the result was so good that you could guess what that part of the original image looked like. This was done using only 25 scans of the same image. We have also indicated what the result would be if we were to perform the analysis on the whole image.

REFERENCE

- [1] Yossi Matias, Adi Shamir, *A Video Scrambling Technique Based On Space Filling Curves*, Proc. of CRYPTO'87 pp 398-417