Visual Cryptanalysis

Adi Shamir
Dept. of Applied Math. and Computer Science
The Weizmann Institute of Science
Rehovot 76100
Israel
shamir@wisdom.weizmann.ac.il

Abstract. In this paper we describe a new paradigm for carrying out massively parallel computations such as brute force cryptanalysis of encryption schemes. It uses photographic films to store the internal state of a bit-sliced computation, and contact printing to carry out the computational steps. While it does not have the exponential speedup of quantum computation or the potential parallelism of DNA computation, it seems to be more practical since it is based on simple commercially available technology. In the last part of the paper we consider hybrid electronic/photographic computations, which combine the advantages of the two models of computation.

Keywords: cryptanalysis, visual computation, bit slice computation, parallel computers, quantum computers, DNA computers, photography.

1 Introduction

Any cryptosystem can be broken by exhaustive search, and in many cases this is the best known cryptanalytic attack. When the number of key bits exceeds 100, the attack is infeasible, but in many cases of practical importance, the number of key bits is 40 (in exportable cryptosystems), 56 (in DES), or 64 (in many proposed cryptosystems). This makes exhaustive search doable, but requires a certain level of parallel processing to complete the attack in a reasonable amount of time.

Exhaustive search is very easy to parallelize, since the various processors do not have to communicate during the computation, and only one of them has to report back the correct key at the end of the computation. In particular, the attacker can use any number of processors up to the number of keys, and obtain a linear speedup. However, size and cost considerations typically limit the number of microprocessors (or special purpose VLSI chips) in such a code breaking machine to tens of thousands, and thus each processor has to carry out a large number of sequential computations which do not fully exploit the available parallelism of the problem.

During the last few years, several researchers proposed novel paradigms of parallel computation, which make it possible to carry out a larger number of

parallel computations. The two best known examples of such novel paradigms are quantum computation and DNA computation.

A quantum computer ([Shor]) uses the surprising properties of quantum mechanics to carry out 2^n parallel computations in a machine of size n. However, only certain types of difficult computations (such as discrete log and factoring) are known to be solvable in polynomial time on a quantum computer, and it is widely believed that general NP-complete problems cannot be solved efficiently on such a machine. In particular, there is no known construction of a quantum computer which can break DES-like cryptosystems in subexponential time.

Another practical drawback of quantum computers is that the whole approach is highly speculative. Physicists have managed to construct a single gate quantum circuit (and are working hard to try to build a two gate device), but there are formidable technical problems in extending this to realistic sizes. Current technology cannot keep hundreds of photons for a long period of time in precisely defined superpositions of quantum states without any destructive interaction with their environment. As a result, quantum cryptanalysis is not likely to become a real threat for the next 20 years, if at all.

DNA computers were first proposed by Adleman ([Adleman]), and attracted considerable attention. The idea was practically demonstrated by solving the toy example of finding a Hamiltonian circuit in a graph with 7 nodes. Humans can find this solution almost instantaneously by looking at a drawing of the graph, but the demonstration proved that the basic biological operations are technically possible. The approach is based on the observation that we can encode the state of a computation by the sequence of bases in a DNA molecule, and manipulate this state by using appropriate restriction enzymes and selective extraction methods. The achievable parallelism is proportional to the number of DNA molecules we can store and process. Unfortunately, current biological techniques are imperfect, and it is not clear whether it is really possible to keep all the DNA molecules stable and error-free during a lengthy computation. In particular, biologists typically apply a small number of modifications to trace amounts of DNA in very low concentrations in a test tube, and it is not clear whether it will scale up to kilograms of DNA undergoing thousands of successive modifications.

In this paper we propose a different paradigm of massively parallel computation, based on photographic techniques. This is a very mature technology, whose capabilities and limitations are very well understood. Photographic devices films and chemicals are cheap and readily available, and anyone can use it in the comfort and privacy of his own home.

We call the new paradigm visual computation, and it should not be confused with the following related but different types of computation:

1. *Optical computing:* Optical computers typically rely on the optical properties of lenses, mirrors, prisms, and lcd arrays in order to modify the path and intensity of multiple rays of light. A simple example of optical computation is the bluring effect of an out-of-focus lense. (A good textbook describing this approach is [Feitelson]). The only operation in our visual computation

is contact printing, in which pixels are never supposed to split, move to a different location or influence their neighbours.

2. *Visual cryptosystems:* This approach (first proposed in Naor and Shamir [NS], and extended in several other papers) uses one transparency to store a key, another transparency to store a ciphertext, and stacks the two transparencies to recover the cleartext. This is an extremely simple example of visual computation, which requires a single alignment step and no photolab operations. Our visual computations require hundreds or thousands of consecutive photographic steps, and achieves the completely different goal of brute force cryptanalysis.
3. *Perforated sheets:* To break the German Enigma cryptosystem during the second world war, British cyptanalysts used large cardboard sheets, with holes punched at various precomputed locations. These sheets were then stacked with a certain relative shift, and the locations of common holes were carefully recorded. This is again an example of a very simple visual computation, using very course pixels and a single visual step. Our visual computations use much smaller pixels, do not employ any shift operations, and carry out a large number of sequential photographic steps which simulate the whole cryptanalytic process (starting with the ciphertext and ending with the key). In other words, the perforated sheets were used just as a convenient way to organize and compare the results of several hundred hand simulations, whereas in visual computation we carry out the actual computation on photographic films.

2 Visual Computation

At any given moment, the state of a visual computer is described by a collection of photographic films or plates. Each film contains a very large number of pixels, which are either black (representing a 1 bit) or transparent (representing a 0 bit). In typical cryptographic applications, these bits are random and thus each film looks uniformly grey (the individual values of the pixels are visible only under strong magnification).

For the sake of concreteness, we'll assume that each film is kept unrolled on a horizontal plane, and thus we can stack them on top of each other in the vertical direction (in reality, we can keep them rolled up except for the region being processed, just as in a real camera).

The visual computer carries out parallel and unrelated computations in a pure SIMD (single instruction multiple data) architecture. The state of each computation is described by the bits along some vertical line through the stacked collection of films, but we do not keep track of its coordinates and cannot address or manipulate it individually.

Simple photographic techniques can support the following operations:

1. *Generation of a 0 film:*
   To generate a completely transparent film (representing a collection of zero bits), develop an unexposed film.

2. *Generation of a 1 film:*
   To generate a completely black film (representing a collection of one bits) develop a film which was taken out of its box under strong light.
3. *Generation of a random film:*
   A more interesting problem is the generation of a film containing a random pattern of zeroes and ones. This can be done by photographing computer-generated random patterns or by using photoprocessing chemicals which create a grainy pattern on high contrast films. A simpler technique is to spray a transparent developed film with a fine mist of an opaque material from a spray can until about 50% of its surface is covered by tiny droplets. If the size of droplets is not small enough, we can use standard photographic reduction techniques to bring it down to the desired pixel size.
4. *Computing NOT:*
   The simplest photographic operation we can apply to a given film is to contact print it to another film. After developing the new film, we get the photographic negative of the original film, which exchanges its black and transparent regions. The net effect of this visual operation is to simultaneously compute the NOT of all the bits stored on the original film.
5. *Computing NOR:*
   A more complicated photographic operation is to contact print two or more stacked films to a new film. The result will be black only when all the corresponding areas in the stacked films are transparent. This is the visual analog of a NOR operation.
6. *Computing NAND:*
   Another photographic operation we can use is double exposure, in which we contact print $film_1$ on a new $film_0$. Before developing $film_0$, we contact print $film_2$ on the same $film_0$. We can extend this process to multiple exposure with any number of input films. After developing $film_0$, we get a black area if any one of the corresponding areas in the input films is transparent. This is the visual analog of a NAND operation.
7. *Computing XOR:*
   Assume that we are given two input films $film_1$ and $film_2$, and would like to compute the output $film_0$ which contains their bitwise XOR. We start by creating $film_3$ which is the photographic negative of $film_1$, and $film_4$ which is the photographic negative of $film_2$. We now expose $film_0$ through the stacked combination of $film_1$ and $film_4$, and then double expose it through the stacked combination of $film_2$ and $film_3$. An interesting open problem is whether it is possible to compute such a XOR in a single step by using more sophisticated photographic processing techniques.

Since the NAND (or the NOR) function is a universal boolean function, we can visually evaluate any boolean function in time complexity which is comparable to its circuit complexity. The big advantage of the visual computation paradigm is that it operates in parallel on a huge number of unrelated inputs, and computes their unrelated outputs.

3 Visual Cryptanalysis of DES

In this section we describe in greater detail the process of breaking the Data Encryption Standard by visual computation. This process is very similar to (and was motivated by) the idea of bit slice computation of DES described in [Biham], which is currently the fastest known way of evaluating DES in software on workstations.

We assume that we are given one cleartext-ciphertext pair, and our goal is to find all the keys which give rise to this pair. We do it by encrypting the given cleartext in parallel by all the possible keys to get all the possible ciphertexts, and then locating all the occurrences of the desired output among them and recovering their associated keys.

The state of each one of the parallel computations at any given moment is a vector of 64 bits, and we keep all these states at that moment as a single stack of 64 films, in which the i-th film contains the i-th most significant bit in all the computations.

We initialize the process by preparing transparencies which encode the same given cleartext in all the parallel computations. For example, if the given cleartext is 0101... we stack a completely transparent film, a completely black film, a completely transparent film, a completely black film, and so on.

In addition, we prepare one random transparency for each key bit. When we stack these transparencies, we hope that all (or most of) the possible keys will occur at some vertical line through the stacked transparencies. We have no control over the actual locations of the various keys, and we cannot guarantee that the correct key will be among them, but we can get good statistical estimates based on the total area of the films and the sizes of their pixels. It is interesting to note that the number of actual key combinations can greatly exceed the number of pixels: If we place one circular black region in each film in such a way that they partially overlap (as in a typical Venn diagram), we get multiple combinations of key bits along different vertical lines through the stacked films.

The actual DES computation proceeds in 16 rounds. In each round we have to apply bit permutations, bit expansions, bitwise XOR's, and S-box transformations.

The bit permutation and expansion operations are the most time consuming operations in standard software implementations of DES. In the visual paradigm they require zero time, since they can be carried out by simply renaming or reusing the given films.

XOR'ing two bits (either two data bits or one data bit and one key bit) can be carried out in three photographic steps by the technique described in the previous section.

The most complicated operation we have to implement is the S-box transformation, in which 6 input bits are mapped to 4 output bits. Standard software implementations of DES use table lookup, which cannot be directly translated into the visual computation framework. The same problem was encountered by Biham in his bit slice architecture, and his solution was to represent each S-box as a boolean circuit with 6 inputs, 4 outputs, and 2-input boolean gates. The

combinatorial complexity of this representation was studied and optimized by several researchers over the last year, and is currently known to be less than 66 gates (on average for the 8 actual s-boxes in DES). This number is likely to be somewhat different in our model, since our XOR's are not single step operations, and on the other hand we can compute multi-input NAND's and NOR's at almost no extra cost.

After completing the visual simulation of the 16 rounds of DES, we obtain a stack of 64 films which contain all the ciphertexts which can be generated by keys represented by the stack of random key films. We can destroy all the films containing bits from intermediate stages, and retain only the initial key and final ciphertext films.

We now describe the final stage of visual processing, in which we extract the keys associated with a particular ciphertext. Assume that the given ciphertext is 0110... . We consider the negative of the first ciphertext film, the second ciphertext film, the third ciphertext film, the negative of the fourth ciphertext film, and so on. If a vertical line through the stack of original ciphertext films was 0110..., then the corresponding vertical line through the new stack is 1111..., i.e, a completely black path. We use muliple exposure to contact print each one of these films (separately) on the same film, which we call the locator film. When this film is developed, it is almost completely black, with a few specks of transparent regions which correspond to the locations of the desired ciphertext. These locations can be made clearly visible by placing the locator film next to a strong source of light and looking for bright spots.

All that is left is to compare the locator film computed by the above procedure to each one of the key films. This is done by stacking the locator film and the i-th key film on top of each other, and determining whether the result is completely opaque. If it is, we know that the i-th key bit is 1, otherwise it is 0 (this proceduree has to be modified slightly if there is more than one key giving rise to the given cleartext-ciphertext pair, which is not usually the case in DES).

An alternative way to complete the visual cryptanalysis is to stack together the versions of the 64 output films which make the correct computation appear as a completely transparent vertical path (i.e., 0000...). The combination can be directly used as the locator film, and can be compared with the i-th key film in the usual way. This approach saves 64 photographic steps, but requires extremely careful allignment of a very thick sandwich of 64 films, which makes it less appealing.

4 Cryptanalysis of Other Cryptosystems

The visual cryptanalytic technique can be applied to many other cryptosystems, but its efficiency varies widely and somtimes in unexpected ways.

Consider for example the operation of data-dependent rotation, in which we rotate one part of the state vector by the number of bits determined by another part. This operation is used in several cryptosystems (including RC5) since its software implementation on modern microprocessors is very fast. However, its

visual implementation is very slow, since we have to rotate the bits along each vertical line through the stacked films by a different amount, and we can't easily do it in a SIMD architecture.

A second example is related to the number of input bits in S-box transformations. If we increase this number from 6 to 16 we do not slow down software implementations based on table lookup, but we make the boolean circuit representation of the S-boxes so large that visual computation becomes essentially undoable.

On the other hand, there are some cryptosystems in which visual cryptanalysis is much easier than in the case of DES. Consider for example the case of a moderately long shift register, in which both the feedback function and the output function are nonlinear combinations of a small number of taps. Such schemes are very common in proprietary commercial systems, and there are no general techniques for breaking them which are much faster than exhaustive search. To apply visual cryptanalysis, we prepare a stack of n random films, where n is the length of the shift register. The vertical lines through this stack represent all the possible initial states of the shift register. To compute the next state, we visually evaluate the two simple functions which yield the feedback and output bits. We throw away the oldest film which represents the oldest state bit, add the new feedback film to the stack of $n-1$ remaining films, and store the output film. We repeat this process n times, and obtain a stack of n output films which are compared in the usual way to the given output string. This will usually suffice to uniquely find the initial state of the shift register, and its total complexity (in terms of visual steps) is likely to be very low.

An interesting research problem is to design cryptographic schemes which are highly resistant to SIMD architectures such as bit slice and visual computation. This resistance can be quantified, and can become a new measure of cryptographic strength, in addition to the known measures of linear complexity, resistance to differential and linear cryptanalytic attacks, etc. Further motivation for such research can be found in a CRAY press release (from March 7, 1995), which mentions a SIMD machine with a bit slice architecture developed jointly by CRAY and the NSA, and describes it in the following way: "For suitable applications, a SIMD processor array of 1 million [single bit] processors would provide up to 32 trillion bit operations per second and price/performance unavailable today on any other high performance platform. The CRAY-3 system with the SSS option will be offered as an application specific product".

5 Practical Aspects of Visual Cryptanalysis

In many cryptanalytic scenarios, the prefix of each message is a fixed known value such as an address, routing information, classification label, or a string of blanks. In this case we can carry out the lengthy visual cryptographic computation only once in a precomputation stage. The running time of this precomputation stage is not critical, and the production of the ciphertext films can take much longer than the validity period of any particular key. When an actual ciphertext is

intercepted, the precomputed ciphertext films can be used in a matter of minutes to produce the locator film defined by the new ciphertext, and thus to find the new key.

In such a scenario, we can use a hybrid electronic/photographic computation, in which the ciphertexts corresponding to the chosen cleartext and all the possible keys are computed electronically by a fast digital computer, and then stored on photographic films by a computer controlled laser beam. Such a hybrid implementation combines the advantages of the two models of computation:

- The precomputation of the ciphertexts is done electronically. This eliminates the potential problems posed by optical imperfections, partially misalligned films, and gradual blurring inherent in a long sequence of visual computations. The resultant ciphertext films are razor sharp, with high contrast and very few errors. In addition, this approach greatly reduces the consumption of photographic films for intermediate results, which are used once or twice during the computation and then discarded. The speed of the sequential generation of the ciphertext films is less crucial in a preprocessing stage, which is allowed to take months or even years.
- The actual extraction of a new key is done photographically. This part benefits from the extremely high parallelism of the visual computation, and requires very few sequential photographic steps to produce the locator film and to compare it to the original key films. In addition, there is no accumulation of errors when the same ciphertext films are used to find additional keys.

Hybrid implementations seem to be truly practical, and use the ciphertext films simply as a storage device which is very cheap, very dense, and highly parallelizable. High quality aerial reconnaisance films can contain up to one billion (2^{30}) pixels per square inch, and they are made in rolls which are several inches wide and several miles long. In quantity, they cost several cents per square inch. A continuous photographic process can thus generate tens of billions of pixels per second, and store them very cheaply for many years with almost no degradation.

It is instructive to compare these characteristics with other common memory types:

1. **Semiconductor memories.** These are the fastest memories, but they are very expensive and offer very limited parallelism. Practical sizes are limited to several gigabytes (about 2^{35} bits), and each gigabyte costs tens of thousands of dollars. They are unsuitable to applications in which we have to precompute the ciphertexts corresponding to a large number of potential cleartexts, and use only one of the precomputed databases at any given time.
2. **Hard disks.** They are much slower (especially in random access applications), and have extremely limited parallelism (from multihead mechanisms). With the new magneto-resistive heads, their bit density approaches 2^{30} bits per square inch, but the area of each platter is limited to tens of square

inches by mechanical considerations. Practical storage capacities are limited to hundreds of gigabytes, and each gigabyte costs tens of dollars.

3. **Magnetic tapes.** They have poor areal density and are relatively slow (transferring hundreds of kilobits per second). However, the separation between the cheap storage medium and the transport mechanism makes them much cheaper than hard disks. A typical tape contains several gigabytes and costs several dollars per gigabyte.
4. **CD-ROM's.** They are cheap and have high areal density, but the reading process is quite slow and the writing process is even slower (typically limited to several megabits per second). A write-once half gigabyte CD-ROM costs about a dollar.

It is important to note that a hybrid electronic/photographic computation makes it possible to solve in a direct way some interesting problems in cryptanalysis. Consider, for example, the following scenarios:

1. **Partially specified ciphertext.** In some applications, certain ciphertext bits may be unknown. Consider, for example, a cryptosystem in which the outputs of DES encryption are XOR'ed with an auxiliary fixed key (this process is known in the literature as "postwhiting"). We can guess the first bit of the auxiliary key, and thus find the first bit in several output blocks produced by DES from several known cleartext blocks. We can easily produce a mixed locator film, based on one bit from each one of the blocks, and find the key whenever the accumulated information suffices to make this film almost completely black.
2. **Partially specified cleartext.** In some applications, there are several possible variants of the known cleartext, and the attacker does not know which one of them was actually used. Consider, for example, the case of a known text which may be partitioned into 8 byte blocks in several ways due to its uncertain relative position in the actual cleartext. We can repeat the whole cryptanalytic attack for each one of the possible variants of the cleartext. A better approach is to introduce "don't care's" into the electronically generated ciphertext films, by creating for each ciphertext bit a pair of films which are not exactly the negative of each other: A ciphertext bit which can be either 0 or 1 (for the same key and some of the possible cleartexts) will be represented by black in both the output film and its (partial) negative. The locator film in this case will lead to all the keys which correspond to any combination of known and "don't care" bit values.
3. **Overcoming errors.** The visual computation is extremely sensitive to errors, and even a single misalligned ciphertext film used in the production of the locator film can mask out the speck of light representing the correct key. A more fault tolerant approach is to use ciphertext films in which light grey represents 0, and dark grey represents 1. The locator film will now have multiple grey levels, which indicate for each key how many of the actual ciphertext bits are equal to the computed ciphertext bits. This "Hamming distance computation" is time consuming in electronic computations, but

almost free in visual computations. With a careful choice of the physical parameters, we can make sure that any key which produces more than 10 mismatches will appear as a completely black pixel on the locator film, but a key which produces only one or two mismatches (due to allignment errors) will be represented by an almost white pixel on the locator film.

The number of parallel computations we can carry out in the new paradigm of visual computation is bounded by the number of pixels we can pack into commercially available films. A fully visual computation can probably support up to 2^{40} parallel computations, and a hybrid electronic/photographic computation can easily support more than 2^{50} parallel computations. An interesting conclusion is that exportable cryptosystems (such as modified DES with 40 bit keys) are so weak, that they can be broken in a reasonable amount of time even if we are not allowed to use any computers!

The maximal achievable parallelism of the visual computation paradigm is somewhat lower than that of DNA computing, but it is based on much simpler and more mature technology. The major limitation of visual computation is that it is a two dimensional technology, whereas DNA computation is a three dimensional technology. If we could store and process information photographically in a three dimensional cube (perhaps by using lasers and holographic techniques), we could enormously expand the achivable parallelism of visual computations.

References

[Adleman] L. M. Adleman, *Molecular Computation of Solutions to Combinatorial Problems*
Science, v. 266 n. 11, Nov 1994, p. 1021.

[Biham] E. Biham, *A Fast New DES Implementation in Software*
Proceedings of Fast Software Encryption 1997,
Springer-Verlag, 1997.

[Feitelson] D. G. Feitelson, *Optical Computing*
published by the MIT Press, 1992.

[NS] M. Naor and A. Shamir, *Visual Cryptography*
Proceedings of Eurocrypt 94, Springer-Verlag, pp. 1-12.

[Shor] P. W. Shor, *Algorithms for Quantum Computation: Discrete Log and Factoring*
in Proceedings of the 35-th Symposium on Foundations of Computer Science, 1994, pp. 124-134.