An Introduction to Block Cipher Cryptanalysis

CHRISTOPHE DE CANNIÈRE, ALEX BIRYUKOV, AND BART PRENEEL

Invited Paper

Since the introduction of the Data Encryption Standard (DES) in the mid-1970s, block ciphers have played an ever-increasing role in cryptology. Because of the growing number of practical applications relying on their security, block ciphers have received, and are still receiving, a substantial amount of attention from academic cryptanalysts. This has led, over the last decades, to the development of several general techniques to analyze the security of block ciphers. This paper reviews the fundamental principles behind today's state of the art in block cipher cryptanalysis.

Keywords*—*Block cipher, cryptanalysis, symmetric encryption.

I. INTRODUCTION

The era of modern cryptology is generally agreed to have started in 1949, when Shannon transformed cryptography from an art to a science with the publication of a paper entitled "Communication theory of secrecy systems" [1]. However, while cryptology took a new fundamental direction from that point on, most of the major innovations in the field date from the last 30 years. This productive period was initiated by two important developments in the mid-1970s.

One of these revolutions was the publication in 1976 of "New directions in cryptography" [2]. In their work Diffie and Hellman suggested ways to insure the privacy of data sent over an insecure channel, without the need for a separate secure channel to exchange secret keys. The introduction of *public key cryptography*, as they called it, opened a whole new research field in cryptology. Still, while Diffie and Hellman's surprising result immediately caught the interest of the academic world, it would take until the early 1990s before public key cryptography found its way to the industry.

Manuscript received November 29, 2004; revised January 4, 2005. This work was supported in part by the European Commission under Contract IST- 2002-507932 (ECRYPT). The work of C. De Cannière, a F.W.O. Research Assistant, is supported by the Fund for Scientific Research—Flanders (Belgium).

The authors are with the Katholieke Universiteit Leuven, Leuven 3000, Belgium (e-mail: christophe.decanniere@esat.kuleuven.be; alex.biryukov@esat.kuleuven.be; bart.preneel@esat.kuleuven.be).

Digital Object Identifier 10.1109/JPROC.2005.862300

The other development, which started a few years earlier, had a more immediate impact on the industry. Realizing that the increasing use of electronic data would entail security risks, and that there was a need for a standardized and publicly available encryption algorithm, the U.S. National Bureau of Standards (NBS) decided in 1973 to issue an open call for encryption primitives. After a second call in 1974, LUCIFER, a block cipher designed by IBM in 1971, emerged as the only serious candidate. After a year of collaboration between IBM and the NSA, LUCIFER was turned into the Data Encryption Standard (DES). In 1977 the complete specifications of the algorithm were finally published as a U.S. Federal Information Processing Standard, FIPS-46 [3].

As soon as the specifications of DES were made public, the cipher became the subject of controversy. Doubts about the security of DES arose from the fact that LUCIFER's original 128-bit secret key had been reduced to 56 bits, and also that the design principles of its substitution and permutation tables were never made public. However, despite these criticisms, the standard would soon be widely used, both in governmental and private organizations. As the number of applications using DES increased, so did the intensity of the search for weaknesses by cryptographers. Still, for about 15 years, exhaustive key search, which was recognized as a serious threat from the start, would remain the most efficient attack. Curiously, the introduction of FEAL [4], a cipher designed by NTT, was the event that triggered a change in the late 1980s. The new cipher was presented as an efficient alternative to DES, but its simple structure was quickly found to be considerably less secure. As the attacks against FEAL improved, it was realized that some of the ideas could be generalized and that they also applied to DES itself. Eventually, in 1991, Biham and Shamir presented the first attack against DES which was faster than exhaustive search [10].

The ideas developed in the early 1990s led to an explosion of new techniques that constitute the base of today's state of the art in block cipher cryptanalysis. The intention of this paper is to provide an overview of the basic principles behind these techniques. The survey focuses especially on attack methodologies which played an important role in

the development of the current standard in block encryption: the Advanced Encryption Standard (AES) [5].

This article is organized as follows: Section II provides a short introduction to symmetric encryption in general, and to block ciphers in particular. Section III discusses a number of general aspects of block cipher cryptanalysis. Sections IV –VI elaborate on three important attacks: differential cryptanalysis, linear cryptanalysis, and multiset attacks. The next section discusses algebraic cryptanalysis, a new approach which is currently being explored by cryptanalysts. Finally, Section VIII briefly mentions techniques which do not attack the block cipher as an algorithm, but instead try to exploit weaknesses in its physical implementation.

II. Symmetric Encryption

The purpose of an encryption algorithm is to protect the secrecy of messages which are sent over an insecure channel. A general encryption algorithm consists of two mathematical transformations: an encryption function E and a decryption function $D = E^{-1}$. In order to communicate in a secure way, the sender (usually called Alice) will apply the encryption function to the original message P (called *plaintext*), and transmit the resulting *ciphertext* $C = E(P)$. Once C is received by Bob (the intended recipient), the plaintext is recovered by computing $D(C) = P$.

In order for this scheme to be of any cryptographic use, two conditions need to be fulfilled. First, the transformation E must be designed in such a way that an eavesdropper (often called Eve) cannot extract any information about the plaintext after intercepting the ciphertext. Second, the decryption function D must be known to Bob, but kept secret from anybody else (with the possible exception of Alice).

A. Symmetric Versus Asymmetric Encryption

Until the 1970s, it was intuitively assumed that the previous conditions immediately implied that the encryption function E had to be secret as well. The reasoning was that if Eve was given E, it would suffice for her to reverse this transformation to recover D. In the mid-1970s, Diffie and Hellman realized that the secrecy of the encryption function was not necessary, at least in theory, provided that one could construct so-called *trapdoor one-way functions*. These are functions which are easy to evaluate, but cannot be efficiently inverted, unless some extra information (the trapdoor) is given. Examples of trapdoor one-way functions were soon found (e.g., [6]) and allowed the development of practical public key encryption algorithms.

While public key cryptography has the huge advantage that Bob does not need to exchange any secret information with Alice before she can start encrypting, schemes which do rely on the secrecy of their encryption function still play a vital role in practical systems. The reason is that implementations of *secret key* or *symmetric* encryption algorithms, as they are called nowadays, are orders of magnitude more efficient than their public key (or *asymmetric*) counterparts. The remainder of this paper is exclusively devoted to symmetric algorithms.

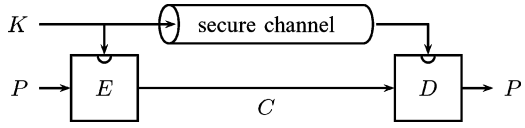


Fig. 1. Model for symmetric encryption.

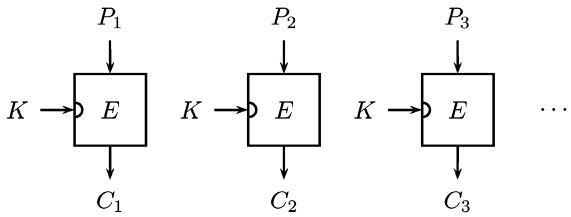


Fig. 2. Block encryption (ECB mode).

B. Kerckhoffs' Principle

In most situations, it is fairly hard to keep an encryption or decryption algorithm completely secret: either Alice and Bob have to design and implement their own algorithm, or they have to trust a designer not to disclose the algorithm to others. Moreover, for each correspondent Alice wants to communicate with, she will need a different algorithm. The solution to this problem is to introduce a secret parameter K as in Fig. 1 and to construct parameterized encryption and decryption functions, in such a way that $D_{K'}(E_K(P))$ does not reveal anything about P as long as $K' \neq K$. Instead of repeatedly having to design new secret algorithms, it now suffices to agree on a secret value for K, called the *key*. Typically, this key is a short binary string of 80 to a few hundred bits. Since the security of the resulting system only relies on the secrecy of the key, the functions E and D can as well be publicly shared. The principle that the full disclosure of an encryption algorithm should not affect its security as long as the key is secret, is known as *Kerckhoffs' principle*.

C. Stream Ciphers and Block Ciphers

Symmetric encryption algorithms are traditionally divided into two categories: *stream ciphers* and *block ciphers*. A block cipher divides the plaintext into separate blocks of fixed size (e.g., 64 or 128 bits), and encrypts each of them independently using the same key-dependent transformation. A stream cipher, on the other hand, takes as input a continuous stream of plaintext and encrypts it according to an internal state which evolves during the process. The differences between both systems are illustrated in Figs. 2 and 3.

While the definitions above draw a clear theoretical distinction between stream ciphers and block ciphers, the situation is a bit more blurred in practice. Block ciphers, for example, are rarely used in the way shown in Fig. 2 (called the *Electronic Codebook* (ECB) mode). Instead, the output of the key-dependent transformation for a given plaintext block is typically kept in memory and used as a parameter when encrypting the next block. While this approach is still commonly called block encryption, it is strictly speaking a stream

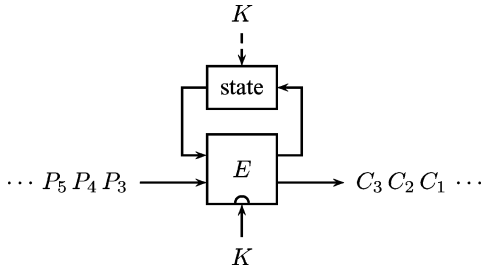


Fig. 3. Stream encryption.

cipher. On the other hand, such constructions do differ considerably from conventional stream ciphers. The latter are expected to process the plaintext in chunks which are small (usually one or a few bits) compared to the size of the internal state. Moreover, as opposed to schemes based on block ciphers, the secret key of a stream cipher is typically only used to initialize the internal state, which from then on is updated in a key- and plaintext-independent way.

Interestingly, the two branches in symmetric cryptology have evolved in rather different circumstances. Block ciphers owe much of their popularity to a few successful designs (such as DES and its successor, AES) which are standardized, freely available, and can be deployed in many different applications. The most widely used stream ciphers, on the contrary, are proprietary designs (e.g., RC4, A5/1), closely tied to a particular application (e.g., GSM). Many of these designs were kept secret until they eventually leaked out or were reverse-engineered. This explains why stream ciphers have tended to receive less attention from the open research community than block ciphers. For the same reason, we have decided to focus on block ciphers in this paper.

D. Anatomy of a Block Cipher

While stream ciphers are based on a variety of principles, most block cipher designs follow the same general approach. They typically consist of a short sequence of simple operations, called the *round function*, which is repeated r times (called *rounds*). The first round takes an n-bit plaintext block as input, and the last round outputs the ciphertext. Additionally, each round depends on a *subkey* (or *round key*) which is derived from a k-bit secret key (this derivation process is called the *key schedule*). Since the receiver must be able to uniquely decrypt the ciphertext, the round function has to be bijective for any value of the secret key. This is usually achieved in one of the following ways.

1) Feistel Ciphers: The round function of a Feistel cipher (named after H. Feistel, one of the IBM researchers who designed LUCIFER and DES) splits the input block into two parts L_{i-1} and R_{i-1}. The right part R_{i-1} is left unchanged and forms the left part of the output L_i. The right part of the output is constructed by adding a modified copy of R_{i-1} to the left part of the input L_{i-1}, i.e.,

$$L_i = R_{i-1},$$
$$R_i = L_{i-1} + f(R_{i-1}, K_i).$$

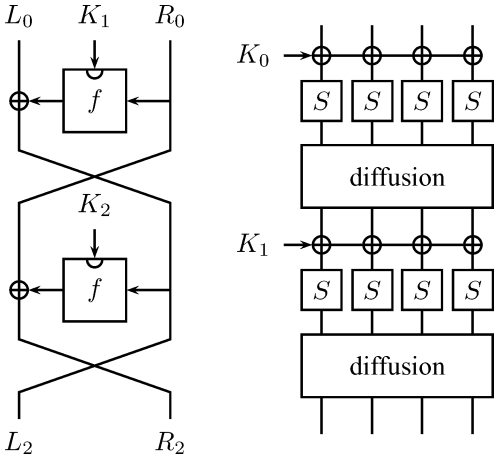


Fig. 4. Feistel cipher versus SP network.

It is not hard to see that this operation can be inverted by subtracting $f(L_i, K_i)$ from R_i, no matter how the function f is constructed. Many block ciphers are based on this structure, including DES.

2) SP Networks: Another approach consists in building a round function by combining layers of simple invertible functions: substitutions (called *S-boxes*) and permutations. The substitution layers act on small units of data (rarely more than eight consecutive bits), and their highly nonlinear properties introduce local *confusion* into the cipher. The permutation layers, on the other hand, are simple linear transformations, but they operate on the complete block, and thus *diffuse* the effect of the substitutions.[1] The most prominent block cipher based on an SP network is the AES. Notice also that the f-functions of many Feistel ciphers consist of a small SP network.

III. BLOCK CIPHER CRYPTANALYSIS

Cryptanalysis is the branch in cryptology which studies how cryptographic algorithms can be broken. While this might not sound very constructive, it is an essential stage in the development of secure algorithms: in order to design a strong cipher, the cryptographer has to understand where the potential weaknesses are.

A. Attack Scenarios

In the case of block ciphers, the task of the adversary Eve consists in recovering unknown parts of the plaintext, or better yet, recovering the secret key. Different attack scenarios can be distinguished depending on what information Eve can obtain, and to what extent she can interfere in the communication between Alice and Bob.

1) Ciphertext-Only Attack: This type of attack only assumes that Eve is capable of capturing encrypted blocks. As this is likely to be the case (otherwise there would be little reason to encrypt the messages in the first place), block ciphers succumbing to cipher-text-only attacks are considered to be very weak.

[1]The terms *confusion* and *diffusion* were introduced by Shannon.

2) Known-Plaintext Attack: A known-plaintext attack requires Eve to have access to (parts of) the plaintext corresponding to the captured ciphertext blocks. This additional requirement is typically rather easy to fulfill. A good example is an online payment on the Internet: while the browser and the server will exchange several kilobytes of encrypted data, it is likely that the only unknown part is a 16-digit credit card number.

3) Chosen-Plaintext Attack: Some attacks only succeed when the plaintexts have a specific form. In order to mount such attacks, Eve must find a way to influence the encrypted plaintexts. A practical example is a secure connection between Alice and her mail server. By sending carefully crafted mails to Alice, Eve can get the server to encrypt the plaintexts she needs.

4) Chosen-Ciphertext Attack: This attack requires Eve to have control over the ciphertexts sent to Bob and to be capable of monitoring how they are decrypted. For example, Eve could try to attack a pay TV decoder by feeding it with special ciphertexts and analyzing its output. Notice that such attacks will not work if the receiver has a means to check the integrity of the ciphertexts.

5) Adaptively Chosen-Plaintext/Ciphertext Attack: In order to mount one of the attacks described above, Eve will typically need to obtain the encryptions or decryptions of a whole series of chosen blocks. When the choice of a certain block depends on the results obtained from previous blocks, the attack is called adaptive.

B. Bounds on the Security of a Block Cipher

Because of their limited block and key length, all block ciphers are susceptible to a number of generic attacks. These attacks do not depend on the internal structure of the cipher and can only be avoided (or at least made impractical) by choosing appropriate external parameters.

The most obvious attack on a cipher with a k-bit secret key is *exhaustive key search*. If Eve is given a small number of plaintext/ciphertext pairs, she could encrypt the plaintexts with all possible keys and compare the result with the previously observed ciphertext. On average, the correct key will reveal itself after $2^k/2$ trials. In certain circumstances, it is possible to reduce this workload. For example, if a single plaintext is encrypted under 2^t different keys, Eve can attack all keys simultaneously and is expected to find the first match after only 2^{k-t} trials. A second possibility is the time-memory tradeoff proposed by Hellman [7], which requires a precomputed table of $2^{2\cdot k/3}$ entries. The precomputation itself still takes 2^k steps, but once the table is completed, any subsequent key can be recovered in $2^{2\cdot k/3}$ steps. Today, an 80-bit secret key is considered to be the minimum required to preclude exhaustive key search; most modern block ciphers have at least 128-bit secret keys.

Another generic attack is related to the block length n. If Eve manages to capture the ciphertexts of all 2^n possible plaintext blocks, she can construct a dictionary which allows her to decrypt any future message encrypted with the same secret key. In fact, Eve does not necessarily need a complete dictionary: whenever a block cipher outputs the same ciphertext block twice, it leaks information about the plaintexts. Since repetitions in a random set of n-bit blocks start to occur frequently when the number of blocks exceeds $2^{n/2}$ (a consequence of the *birthday paradox*), it is not advisable for Alice and Bob to encrypt more than $2^{n/2}$ blocks with the same secret key.

C. Shortcut Attacks

Once Eve has convinced herself that the block and key lengths of the block cipher prohibit generic attacks, she will search for special properties in the cipher's internal structure. Attacks which reduce the complexity of exhaustive search by exploiting internal properties are called *shortcut attacks*. In the last 15 years, cryptanalysts have started to develop systematic methods to search for shortcut attacks. Many of the successful techniques boil down to the same two-step strategy.

1) Build a Distinguisher: Given a sequence of plaintext/ciphertexts pairs, Eve will always be able to tell from the encryption of an unknown plaintext block whether or not it is equal to one of the plaintexts in the sequence. In order to make sure that this is also the only information Eve can extract, the block cipher must look like a completely random permutation to any adversary which does not know the key and has a limited amount of computational resources. Conversely, any property which allows Eve to distinguish the block cipher from a random permutation is an interesting weakness, which, as explained below, is typically only one step away from a key recovery attack.

2) Recover Round Keys: Let us consider a reduced encryption function constructed by omitting the last round of the block cipher. Suppose that Eve is able to efficiently distinguish whether or not a given sequence of input/output blocks could have been produced by this reduced function for some secret key. If Eve is now given plaintext/ciphertext pairs from the original cipher, she can guess (parts of) the last round key, (partly) decrypt the last round, and use her distinguisher on the first $r-1$ rounds to check whether the guess could have been correct. Once she has obtained a correct round key, she can proceed with an exhaustive search for the remaining key bits, or peel off one round and start again.

The next three sections provide a more detailed discussion of three important cryptanalytical techniques based on these ideas.

IV. Differential Cryptanalysis

Differential cryptanalysis has been, and still is, one of the most influential techniques in block cipher cryptanalysis. It was developed by Biham and Shamir in the late 1980s and was originally used to demonstrate weaknesses in the block cipher FEAL. The technique was first published in a generalized form in 1990 and illustrated with attacks on reduced-round versions of DES [8], [9]. After a few additional improvements it eventually led, in 1991, to the first attack on the full 16-round DES which was faster than exhaustive search [10].

A. A Differential Distinguisher

Constructing an efficient distinguisher essentially consists in finding a distinctive property in the input and output blocks which reveals the use of the block cipher regardless of the value of the secret key.[2] This suggests that the attacker should somehow eliminate the effect of the unknown key. Differential cryptanalysis attempts to do exactly that, by studying differences of input and output blocks encrypted with the same key.

In many block ciphers (including FEAL, DES, and AES), the secret key bits are injected in the encryption function by XORing them to intermediate data blocks at different stages in the computation. Let X_1 and X_2 be the values of such an intermediate data block for two different plaintexts P_1 and P_2. Assuming that both plaintexts are encrypted with the same key, we can write

$$\begin{cases} Y_1 = X_1 \oplus K_i \\ Y_2 = X_2 \oplus K_i \end{cases} \Rightarrow \begin{aligned} \Delta Y &= Y_1 \oplus Y_2 \\ &= (X_1 \oplus K_i) \oplus (X_2 \oplus K_i) \\ &= X_1 \oplus X_2 = \Delta X. \end{aligned}$$

This simple observation illustrates the purpose of a differential approach: while the adversary cannot compute the values Y_1 and Y_2 without knowing the round key K_i, she can easily determine their difference ΔY, given ΔX. The idea of differential cryptanalysis is to try to extend this property over multiple rounds. If Eve manages to predict the output difference ΔC by tracing how the input difference ΔP evolves through the cipher, then this obviously distinguishes the cipher from a random permutation.

B. Differential Characteristics

In practice, a cipher does not only consist of key additions (which, as shown above, are completely transparent to differences); it also contains diffusion components and nonlinear S-boxes. Linear diffusion layers do not pose a serious problem. Although they do not preserve differences, they do transform them in a predictable way

$$\begin{cases} Y_1 = A \cdot X_1 \\ Y_2 = A \cdot X_2 \end{cases} \Rightarrow \quad \Delta Y = A \cdot \Delta X. \tag{1}$$

Unfortunately, this is not true for S-boxes (or any other nonlinear component the cipher may have). Unless the difference ΔX at the input of the S-box is zero, Eve typically cannot determine the output difference ΔY without knowing the actual value of X_1. However, given ΔX and assuming that X_1 is uniformly chosen, she can compute the statistical distribution of possible output differences (we will see an example later). In order to proceed, Eve will simply pick one of these output differences, compute the probability that her choice was correct, and continue her analysis. Eventually, she will reach the output of the cipher, and will have described one of the possible ways in which the difference ΔP at the input could have propagated through the cipher. This is called a differential *characteristic*. The probability p that a given pair of plaintexts actually follows this characteristic is the product of the probabilities of all choices that Eve had to make (assuming that these probabilities are independent).

C. Minimizing the Data Requirements

In order to use the probability p to distinguish the block cipher from a random permutation, Eve will need the encryptions of a sufficient amount of plaintext pairs with a fixed difference ΔP. Notice that this assumes that she can *choose* the plaintexts. Eve will then count the number of pairs which produce the output difference predicted by her characteristic. For N pairs of plaintexts encrypted with the block cipher, this number is expected to be at least[3] $p \cdot N$. In the random case, Eve expects the predicted output difference to appear only $p_R \cdot N$ times, with p_R in the order of $1/2^n$. In order to clearly distinguish both cases, the numbers must differ by at least a few standard deviations

$$|p \cdot N - p_R \cdot N| \propto \sqrt{N \cdot p(1-p) + N \cdot p_R(1-p_R)}. \tag{2}$$

Hence, assuming that $p_R \ll p \ll 1$, we obtain the condition

$$N \propto \frac{1}{p}.$$

This clearly shows that the larger the probability of Eve's characteristic, the more efficient the distinguisher will be. Searching for the most probable characteristic typically involves a tradeoff between two objectives. Eve's first goal is to select the differences at the inputs and outputs of the diffusion layers in such a way that they affect as few S-boxes in the neighboring layers as possible. Whenever an S-box is kept *inactive* this way, its output difference does not need to be guessed (it can only be zero). Second, in all places where the differences do affect an S-box (called an *active* S-box), Eve will try to choose the pair of input and output differences that has the largest possible probability. To facilitate this task, she will construct a *difference distribution table*, which lists the probabilities of all possible pairs of differences at the input and the output of the given S-box. Table 1 gives an example for a 3-bit S-box. For each input difference ΔX, the table contains a row showing the distribution of possible output differences ΔY. Notice that in this specific example, $\Delta X = 2$ results in $\Delta Y = 6$ with probability 1. Such a weakness is not likely to exist in a larger S-box.

D. Applications and Extensions

In their original attack, Biham and Shamir used a 13-round distinguisher to recover key bits from the last two rounds of

[2]Efficient distinguishers which only work for specific values of the key (called *weak* keys), are also useful, provided that the fraction of these keys is sufficiently large. However, such attacks are out of the scope of the present survey.

[3]As an input difference might propagate to the same output difference in multiple ways, this number is sometimes significantly higher. The set of all characteristics with the same input and output differences is called a *differential*.

Table 1
Difference Distribution and Linear Approximation Table for a 3-bit S-box $Y = S(X)$

X	Y
0	7
1	5
2	1
3	3
4	2
5	6
6	4
7	0

Δ	0	1	2	3	4	5	6	7
0	1	0	0	0	0	0	0	0
1	0	0	1/2	0	1/2	0	0	0
2	0	0	0	0	0	0	1	0
3	0	0	1/2	0	1/2	0	0	0
4	0	0	0	1/2	0	1/2	0	0
5	0	1/2	0	0	0	0	0	1/2
6	0	0	0	1/2	0	1/2	0	0
7	0	1/2	0	0	0	0	0	1/2

Γ	0	1	2	3	4	5	6	7
0	1/2	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	−1/2
2	0	0	−1/4	−1/4	−1/4	1/4	0	0
3	0	0	−1/4	1/4	1/4	1/4	0	0
4	0	−1/2	0	0	0	0	0	0
5	0	0	0	0	0	0	1/2	0
6	0	0	1/4	1/4	−1/4	1/4	0	0
7	0	0	−1/4	1/4	−1/4	−1/4	0	0

a DES variant reduced to 15 rounds. The distinguisher was based on a 13-round differential characteristic with probability 2^{-47}. In 1991, the two researchers realized that they could allow an extra round at the input of the distinguisher by imposing additional restrictions on the plaintexts. This observation, together with an improved procedure to eliminate wrong key candidates, eventually resulted in the first theoretical break of the full 16-round DES cipher. The attack required an impractical amount of data (2^{47} chosen plaintexts), but was significantly more efficient than exhaustive search (i.e., trying out about half of all 2^{56} possible keys).

After the publication of these first differential attacks, various improvements and extensions have been proposed. Techniques have been developed to exploit *truncated differences* [11] (differences which leave a number of bits undetermined), *impossible differentials* [12] (combinations of input and output differences that can never occur), and *higher order differences* [13] (differences of differences). Another interesting development is the *boomerang attack* [14], which builds an adaptive attack using two separate differential characteristics, each covering half of the cipher.

V. Linear Cryptanalysis

The second powerful technique developed in the early 1990s is linear cryptanalysis. The attack in its current form was introduced by Matsui in 1993 [15] and was first applied to DES. However, as was the case with differential cryptanalysis, early variants of the attack were already used in 1992 to break FEAL [16].

A. Linear Approximations

Whereas differential cryptanalysis focuses on differences in data blocks, linear cryptanalysis studies the relation between linear combinations of plaintext and ciphertext bits. The attack relies on the existence of a *linear approximation* of the cipher. This is a linear expression of the form

$$\Gamma_P^{\mathsf{T}} \cdot P \oplus \Gamma_C^{\mathsf{T}} \cdot C = \Gamma_K^{\mathsf{T}} \cdot K \tag{3}$$

which holds with probability $p \neq 1/2$, where C is the encryption of P under the key K. The column vectors Γ_P, Γ_C, and Γ_K are called *linear masks* and represent a particular linear combination of bits.

The motivation to study differential properties in the previous section was that it allowed to eliminate the secret key. In linear cryptanalysis this goal is only partly achieved: the secret key is reduced to a single unknown bit, $\Gamma_K^{\mathsf{T}} \cdot K$. As a result, three cases can be distinguished. Assuming that Eve is given the encryptions of N arbitrary plaintexts, let T be the number of texts such that the left-hand side of (3) is zero. If Eve finds that T is close to $p \cdot N$, she will conclude that the block cipher was used with a secret key K satisfying $\Gamma_K^{\mathsf{T}} \cdot K = 0$. On the other hand, if T converges to $(1-p) \cdot N$, she will assume that $\Gamma_K^{\mathsf{T}} \cdot K = 1$. Finally, a value of T close to $(1/2) \cdot N$ (which is what Eve would expect in the random case) indicates that the plaintext/ciphertext pairs were probably not generated by the block cipher. The number of texts required to accurately distinguish these three cases can be computed using (2). This time, we have $p \approx p_R = 1/2$, resulting in the condition

$$N \propto \frac{1}{\left(p - \frac{1}{2}\right)^2}.$$

As can be noticed in the previous paragraph, an important advantage of linear cryptanalysis over differential cryptanalysis, is that it does not impose restrictions on the plaintexts: it is a *known* instead of a *chosen* plaintext attack. Moreover, the procedure described above does not only allow Eve to distinguish the block cipher from a random permutation, it also immediately provides her with (the equivalent of) one secret key bit. Nevertheless, in order to recover the complete key efficiently, the distinguisher will still be used as explained in Section III-C.

B. Linear Characteristics

When mounting a linear attack, Eve's first task consists in finding a useful linear approximation. As deduced above, the more the probability of the approximation differs from 1/2, the lower the number of plaintexts required by the attack. Finding the best linear approximation for an arbitrary cipher is in general not a trivial task. However, if the cipher is composed of simple components (as is mostly the case), one could try to approximate the complete cipher by combining linear approximations for individual components.

Finding linear relations between the input and the output bits of components which are linear already, is obviously very easy. For example, in order to write a linear expression

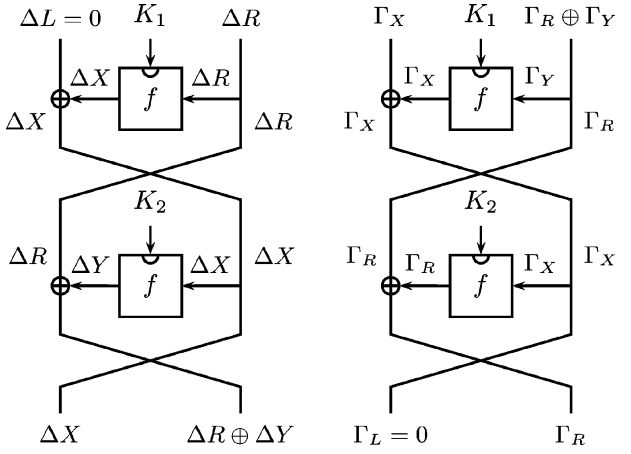


Fig. 5. Propagation of differential and linear characteristics in a Feistel cipher. The difference ΔR propagates from the input to the output; the linear mask Γ_R takes the opposite direction.

which holds with probability 1 for a key addition, it suffices to choose masks Γ_X, Γ_Y, and Γ_K in the following way:

$$Y = X \oplus K_i \quad \Rightarrow \quad \begin{array}{c} \Gamma_Y^\mathsf{T} \cdot Y = \Gamma_Y^\mathsf{T} \cdot X \oplus \Gamma_Y^\mathsf{T} \cdot K_i \\ \Downarrow \\ \Gamma_X = \Gamma_K = \Gamma_Y. \end{array}$$

Similarly, if Eve wants to construct a linear relation for a linear diffusion layer, she can choose the masks as follows:

$$Y = A \cdot X \quad \Rightarrow \quad \begin{array}{l} \Gamma_Y^\mathsf{T} \cdot Y = \Gamma_Y^\mathsf{T} \cdot A \cdot X \\ \qquad\quad = (A^\mathsf{T} \cdot \Gamma_Y)^\mathsf{T} \cdot X \\ \qquad\quad \Downarrow \\ \Gamma_X = A^\mathsf{T} \cdot \Gamma_Y. \end{array} \tag{4}$$

S-boxes, which are designed to be highly nonlinear, can typically not be approximated very accurately with a linear expression. The linear approximation with the best correlation can be found by constructing a *linear approximation table*, which is the equivalent of the difference distribution table used in differential cryptanalysis. An example is given in Table 1. For all pairs (Γ_X, Γ_Y), the table lists the value of $(p - 1/2)$, with p the probability that $\Gamma_X^\mathsf{T} \cdot X = \Gamma_Y^\mathsf{T} \cdot Y$. The more this value differs from zero, the better the approximation.

When comparing (1) and (4), we notice that there is a certain duality between differential and linear cryptanalysis: the first equation describes how differences propagate from the input to the output of a diffusion layer; the second equation shows a similar property for linear masks, but this time the masks propagate from the output to the input, and they are multiplied with A^T instead of A. A result of this duality is that chains of approximations, called *linear characteristics* can be constructed in exactly the same way as differential characteristics. This is illustrated in Fig. 5.

C. Piling-Up Lemma

The only missing link in the construction of linear characteristics is a rule for computing the total probability of a chain of approximations. This is where the so-called *Piling-up Lemma* comes into play.

Lemma 1: Given n *independent* linear approximations of the form $\Gamma_i^\mathsf{T} \cdot X_i = \Gamma_{i-1}^\mathsf{T} \cdot X_{i-1}$, each with a probability $p_i = 1/2 + \epsilon_i$, then the combined probability of the approximation $\Gamma_n^\mathsf{T} \cdot X_n = \Gamma_0^\mathsf{T} \cdot X_0$ is given by $p = 1/2 + \epsilon$ with

$$\epsilon = 2^{n-1} \prod_{i=1}^{n} \epsilon_i. \tag{5}$$

The values ϵ_i used above are called the *biases* of the linear approximations. The lemma can be further simplified by defining $c_i = 2 \cdot \epsilon_i$, known as the *correlation* or the *imbalance*. With this notation, (5) reduces to $c = \prod c_i$. The square of the correlation, appropriately called the *linear probability*, makes the similarity between linear and differential cryptanalysis even more apparent: linear probabilities can be multiplied as before, and just as in differential cryptanalysis, the inverse of their product is proportional to the number of plaintexts required by the distinguisher.

D. Applications

In his original paper, Matsui presented two different attack algorithms for DES. The first, called Algorithm 1, used one large characteristic covering all 16 rounds, and allowed to recover the value of $\Gamma_K^\mathsf{T} \cdot K$. The second algorithm, Algorithm 2, was a key recovery attack based on a 15-round linear distinguisher. In 1994, Matsui proposed an improved variant of Algorithm 2, using a 14-round linear characteristic. The attack required 2^{43} known plaintexts and was the first attack on DES that was verified experimentally.

Today, Matsui's attack is still considered to be amongst the most efficient attacks on DES. A number of interesting variants of linear cryptanalysis have been proposed in the last decade, including attacks using chosen plaintexts [17], nonlinear approximations [18], [19], or multiple linear approximations [20], [21], but when applied to DES, none of these approaches could improve Matsui's attack with more than a factor of four.

VI. Multiset Attacks

After the discovery of linear and differential cryptanalysis, cryptographers started to design ciphers which minimized both the maximum probability of differential characteristics and the maximum correlation of linear characteristics. One of these ciphers was SQUARE, designed by Daemen and Rijmen in 1997. However, during the analysis of a preliminary version of this block cipher, Knudsen discovered that it was vulnerable to a new type of attack. This forced the designers to increase the number of rounds, and the resulting cipher was published in [22], together with the new attack, which was from then on referred to as the "SQUARE attack."

Differential and linear attacks are in general very sensitive to the exact specification of each component in the cipher.

This is much less the case for the type of cryptanalysis described in this section: the SQUARE attack is not affected by specific design choices for individual components, but relies only on how these components, which are considered as black boxes, are interconnected. Another interesting feature of the attack is that it is not probabilistic: if Eve does not detect the special property which the distinguisher relies on, then she knows for sure that the plaintext/ciphertext pairs were not generated by the block cipher.

The general technique used in the SQUARE attack has been given different names in the last few years. Lucks proposed the name *saturation attack* [23], Biryukov and Shamir treated the technique as a special case of *structural cryptanalysis* [24], and Knudsen and Wagner referred to it as *integral cryptanalysis* [25].

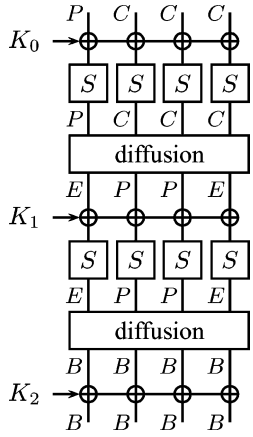


Fig. 6. An example of how multisets might propagate through an SP network. The labels C, P, E, and B respectively stand for constant, permutation, even, and balanced.

A. Multisets

The key idea behind the SQUARE attack is somewhat similar to the differential approach in Section IV. However, instead of analyzing pairs of related plaintexts, the attacker will now study the behavior of complete sets of carefully chosen plaintexts. In order to analyze these sets, the text blocks are first split into m-bit words whose size matches the internal structure of the cipher. The different values taken by each individual word are then treated as multisets. A multiset is a list of values, each of which can appear multiple times, but the order of which is irrelevant. During the attack, a number of special multisets are considered.

1) Constant Multiset: A multiset consisting of a single value repeated an arbitrary number of times.

2) Permutation or Saturated Multiset: A multiset which contains all 2^m possible values for the word exactly once.

3) Even Multiset: A multiset in which each value, if present, occurs an even number of times.

4) Balanced Multiset: A multiset such that the XOR of all values (taking into account their multiplicity) is zero.

Notice that some of these properties are implied by others. For example, a constant multiset with an even number of elements is also an even multisets, and any saturated or even multiset is automatically balanced.

B. How Multisets Propagate

In order to distinguish a block cipher from a random permutation, the adversary will first construct a set of plaintexts such that the values at different positions form special multisets. For example, when analyzing an SP network consisting of 8×8-bit S-boxes, Eve might choose 256 plaintexts which take on all possible values in the first byte (a saturated multiset), but are constant in the others. Her task is then to trace how these multisets are transformed as the plaintexts are encrypted. Interestingly, most of the transformations commonly found in a block cipher preserve or translate at least some of the multiset properties described above. A constant or an even multiset, for example, retains its special properties after having been transformed by an arbitrary function over m-bit values (e.g., an S-box). Similarly, a saturated multiset is preserved by any bijective function, and a balanced multiset by any linear transformation. Finally, if a saturated and a constant multiset are combined in a linear way, the result will be either saturated or even.

Using the propagation rules described above, Eve can typically keep track of the multiset properties over two to four rounds (see, for example, Fig. 6). If the multisets at the output of the last round do not exhibit the predicted properties, then this indicates that the output texts were generated in a different way.

C. Applications

Multiset attacks are of particular significance today because of their applicability to RIJNDAEL. The RIJNDAEL cipher, designed by Daemen and Rijmen in 1998 [26], is a successor of SQUARE. It was submitted to the U.S. National Institute of Standards and Technology[4] (NIST) in response to an open call for 128-bit block ciphers. It was, together with 14 other candidates, extensively evaluated during two years, before NIST announced in 2000 that RIJNDAEL would replace DES and become the new AES.

Just as its predecessor SQUARE, RIJNDAEL was specifically designed to resist differential and linear cryptanalysis. As of today, multiset attacks have shown to be the most effective in breaking reduced versions of RIJNDAEL. The SQUARE attack, which was also applicable to the RIJNDAEL structure, allowed to break six rounds out of ten. It recovered the 128-bit key using a set of 2^{32} special plaintexts, and it required a computational effort of 2^{72} steps. The work factor was later reduced to 2^{44} by performing the calculations in a more efficient way [27]. Ferguson *et al.* [27], as well as Gilbert and Minier [28], have developed more sophisticated multiset attacks that could be applied to seven rounds. However, when RIJNDAEL is used with a 128-bit secret key, both attacks are only marginally faster than generic attacks. If the reduced cipher is used with a larger key, it takes one or two more rounds before the complexities of the currently best attacks exceed the complexity of exhaustive search.

[4]Previously called the National Bureau of Standards (NBS).

VII. Algebraic Cryptanalysis

What distinguishes RIJNDAEL from many other ciphers is that all of its components were intentionally derived from very simple algebraic functions with well-known properties. This strategy has the advantage that some important security aspects of the cipher (e.g., the maximum probability of differential characteristics) can easily be analyzed and proved. The risk, however, is that this approach may expose the cipher to new types of attacks, which precisely exploit these algebraic structures. The most straightforward algebraic attack would consist of two steps. The first is the construction of a simple set of algebraic equations which completely describes how the plaintext, the ciphertext and the key of a specific block cipher are related. The second step consists in filling in the data obtained from a few known plaintext/ciphertext pairs, and solving the equations in order to extract the key.

The first step has received quite some attention in a number of recent papers [29]–[31]. Each of them describes different ways of expressing RIJNDAEL in a simple algebraic form. Ferguson *et al.* [30] observe that the ciphertext bytes can be expressed as a function of the plaintext and the round keys by means of a single, very structured, nonlinear equation in the finite field $GF(2^8)$. Courtois and Pieprzyk [29] construct a set of about 2^{12} quadratic equations describing the complete cipher in $GF(2)$ and a similar system in $GF(2^8)$ is proposed by Murphy and Robshaw [31].

The second step, which consists in solving the equations, is still an ongoing topic of research. The problem is believed to be very hard, but at this stage, it is not clear how the exact workload should be estimated. It seems unlikely however, that a realistic attack (complexity less than 2^{80}) could be developed without a major breakthrough. Interestingly enough, the ideas that were originally developed for exploiting algebraic properties of block ciphers, have had a much larger impact on stream cipher cryptanalysis. In 2003, Courtois and Meier [32] demonstrated that many stream cipher designs could be described as a very overdefined system of low degree algebraic equations and could relatively easily be solved.

VIII. Side-Channel Attacks

A completely different class of attacks exploits the physical characteristics of the actual implementation of a block cipher. These attacks assume that the adversary has physical access to the encrypting device and typically consist in measuring the exact execution time, the instantaneous power consumption, or the electromagnetic emanations during the encryption. If one of these characteristics depends somehow on the value of the secret key, the attacker might be able to break the cipher much more easily than expected. While many of the algorithmic short-cut attacks described in the previous sections still require impractical amounts of data or computational resources, side-channel attacks can be a serious threat in practice.

When classifying implementation attacks, a distinction can be made between Timing Analysis [33], Simple Power Analysis (SPA), Differential Power Analysis (DPA) [34], Simple Electromagnetic Analysis (EMA), and Differential Electromagnetic Analysis (DEMA) [35]. In Timing Analysis, the attacker usually exploits conditional branches which cause variations in the execution time. In a similar way, Simple Power/Electromagnetic Analysis targets implementations where different sequences of instructions, corresponding to different power consumption patterns, may be executed depending on the key. It should be pointed out that information provided by these two side channels may be complementary. When a cipher can be implemented with a fixed sequence of instructions (as is the case for RIJNDAEL), these two types of attack are in principle easily prevented. In some implementations, however, the power consumption of certain instructions might be strongly correlated with the Hamming weights of the operands. If this is the case, special types of SPA attacks can be mounted [36].

Differential Power/Electromagnetic Analysis (DPA/DEMA) is the most sophisticated type of implementation attack and consists in measuring small variations in the power consumption when instructions are executed with different operands. Completely protecting an implementation against these attacks, while keeping the implementation cost reasonable, is in general very hard. One of the proposed countermeasures involves *masking* of operands. An efficient example of a simple masking scheme for RIJNDAEL is presented in [37].

IX. Conclusion

In the last 15 years, the field of block cipher cryptanalysis has seen many interesting developments. A number of them have been discussed in this paper; many others have not. Nevertheless, we hope that the selection of techniques covered in the preceding sections provides the reader with a sense of the general approaches taken by today's cryptanalysts.

References

[1] C. E. Shannon, "Communication theory of secrecy systems," *Bell Syst. Tech. J.,* vol. 28, pp. 656–715, 1949.

[2] W. Diffie and M. E. Hellman, "New directions in cryptography," *IEEE Trans. Inf. Theory,* vol. IT-22, no. 6, pp. 644–654, Nov. 1976.

[3] *Data Encryption Standard (DES)*, FIPS-46, National Institute of Standards and Technology, 1979 [Online]. Available: http://csrc.nist.gov/publications/fips/fips46-3/fips46-3.pdf, revised as FIPS 46-1:1988, FIPS 46-2:1993, FIPS 46-3:1999

[4] A. Shimizu and S. Miyaguchi, "Fast data encipherment algorithm FEAL," in *Advances in Cryptology—EUROCRYPT'87*, D. Chaum and W. L. Price, Eds. Heidelberg, Germany: Springer-Verlag, 1988, vol. 304, Lecture Notes in Computer Science, pp. 267–278.

[5] *Advanced Encryption Standard*, ser. FIPS-197, National Institute of Standards and Technology, Nov. 2001 [Online]. Available: http://csrc.nist.gov/encryption/

[6] R. L. Rivest, A. Shamir, and L. M. Adleman, "A method for obtaining digital signatures and public-key cryptosystems," *Commun. ACM,* vol. 21, no. 2, pp. 120–126, Feb. 1978.

[7] M. E. Hellman, "A cryptanalytic time-memory tradeoff," *IEEE Trans. Inf. Theory,* vol. 26, no. 4, pp. 401–406, Jul. 1980.

[8] E. Biham and A. Shamir, "Differential cryptanalysis of DES-like cryptosystems," in *Advances in Cryptology-CRYPTO'90*, A. Menezes and S. A. Vanstone, Eds. Heidelberg, Germany: Springer-Verlag, 1990, vol. 537, Lecture Notes in Computer Science, pp. 2–21.

[9] ——, *Differential Cryptanalysis of the Data Encryption Standard*. New York: Springer-Verlag, 1993.

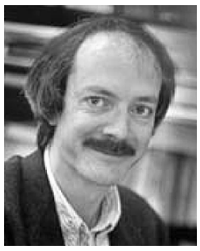
[10] ——, "Differential cryptanalysis of the full 16-round DES," in *Advances in Cryptology—CRYPTO'92,* E. F. Brickell, Ed. Heidelberg, Germany: Springer-Verlag, 1993, vol. 740, Lecture Notes in Computer Science, pp. 487–496.
[11] L. R. Knudsen, "Truncated and higher order differentials," in *Fast Software Encryption, FSE'94*, B. Preneel, Ed. Heidelberg, Germany: Springer-Verlag, 1995, vol. 1008, Lecture Notes in Computer Science, pp. 196–211.
[12] E. Biham, A. Biryukov, and A. Shamir, "Cryptanalysis of Skipjack reduced to 31 rounds using impossible differentials," in *Advances in Cryptology—EUROCRYPT '99* , J. Stern, Ed. Heidelberg, Germany: Springer-Verlag, 1999, vol. 1592, Lecture Notes in Computer Science, pp. 12–23.
[13] X. Lai, "Higher order derivatives and differential cryptanalysis," in *Proc. Symp. Communication, Coding and Cryptography in Honor of James L. Massey on the Occasion of His 60th Birthday* 1994, pp. 227–233.
[14] D. Wagner, "The boomerang attack," in *Fast Software Encryption, FSE'99*, L. R. Knudsen, Ed. Heidelberg, Germany: Springer-Verlag, 1999, vol. 1636, Lecture Notes in Computer Science, pp. 156–170.
[15] M. Matsui, "Linear cryptanalysis method for DES cipher," in *Advances in Cryptology—EUROCRYPT'93*, T. Helleseth, Ed. Heidelberg, Germany: Springer-Verlag, 1993, vol. 765, Lecture Notes in Computer Science, pp. 386–397.
[16] ——, "A new method for known plaintext attack of FEAL cipher," in *Advances in Cryptology—EUROCRYPT'92,* R. A. Rueppel, Ed. Heidelberg, Germany: Springer-Verlag, 1993, vol. 658, Lecture Notes in Computer Science, pp. 81–91.
[17] L. R. Knudsen and J. E. Mathiassen, "A chosen-plaintext linear attack on DES," in *Fast Software Encryption, FSE 2000*, B. Schneier, Ed. Heidelberg, Germany: Springer-Verlag, 2001, vol. 1978, Lecture Notes in Computer Science, pp. 262–272.
[18] L. R. Knudsen and M. J. B. Robshaw, "Non-linear approximations in linear cryptanalysis," in *Advances in Cryptology—EUROCRYPT'96*, U. Maurer, Ed. Heidelberg, Germany: Springer-Verlag, 1996, vol. 1070, Lecture Notes in Computer Science, pp. 224–236.
[19] T. Shimoyama and T. Kaneko, "Quadratic relation of s-box and its application to the linear attack of full round DES," in *Advances in Cryptology—CRYPTO'98*, H. Krawczyk, Ed. Heidelberg, Germany: Springer-Verlag, 1998, vol. 1462, Lecture Notes in Computer Science, pp. 200–211.
[20] B. S. Kaliski and M. J. Robshaw, "Linear cryptanalysis using multiple approximations," in *Advances in Cryptology—CRYPTO'94*, Y. Desmedt, Ed. Heidelberg, Germany: Springer-Verlag, 1994, vol. 839, Lecture Notes in Computer Science, pp. 26–39.
[21] A. Biryukov, C. De Cannière, and M. Quisquater, "On multiple linear approximations," in *Advances in Cryptology—CRYPTO 2004*, M. Franklin, Ed. Heidelberg, Germany: Springer-Verlag, 2004, vol. 3152, Lecture Notes in Computer Science, pp. 1–22.
[22] J. Daemen, L. R. Knudsen, and V. Rijmen, "The block cipher square," in *Fast Software Encryption—FSE'97*, E. Biham, Ed. Heidelberg, Germany: Springer-Verlag, 1997, vol. 1267, Lecture Notes in Computer Science, pp. 149–165.
[23] S. Lucks, "Attacking seven rounds of Rijndael under 192-bit and 256-bit keys," in *Proc. 3rd AES Candidate Conf.* 2000, pp. 215–229.
[24] A. Biryukov and A. Shamir, "Structural cryptanalysis of SASAS," in *Advances in Cryptology—EUROCRYPT 2001*, B. Pfitzmann, Ed. Heidelberg, Germany: Springer-Verlag, 2001, vol. 2045, Lecture Notes in Computer Science, pp. 394–405.
[25] L. R. Knudsen and D. Wagner, "Integral cryptanalysis (extended abstract)," in *Fast Software Encryption, FSE 2002*, J. Daemen and V. Rijmen, Eds. Heidelberg, Germany: Springer-Verlag, 2002, vol. 2365, Lecture Notes in Computer Science, pp. 112–127.
[26] J. Daemen and V. Rijmen, *The Design of Rijndael: AES—The Advanced Encryption Standard.* New York: Springer-Verlag, 2002.
[27] N. Ferguson, J. Kelsey, S. Lucks, B. Schneier, M. Stay, D. Wagner, and D. Whiting, "Improved cryptanalysis of Rijndael," in *Fast Software Encryption, FSE 2000*, B. Schneier, Ed. Heidelberg, Germany: Springer-Verlag, 2001, vol. 1978, Lecture Notes in Computer Science, pp. 213–230.
[28] H. Gilbert and M. Minier, "A collision attack on seven rounds of Rijndael," in *Proc. 3rd AES Candidate Conf.* 2000, pp. 230–241.
[29] N. T. Courtois and J. Pieprzyk, "Cryptanalysis of block ciphers with overdefined systems of equations," in *Advances in Cryptology—ASIACRYPT 2002*, Y. Zheng, Ed. Heidelberg, Germany: Springer-Verlag, 2002, vol. 2501, Lecture Notes in Computer Science, pp. 267–287 [Online]. Available: http://www.iacr.org [earlier version]
[30] N. Ferguson, R. Schroeppel, and D. Whiting, "A simple algebraic representation of Rijndael," in *Selected Areas in Cryptography, SAC 2001*, S. Vaudenay and A. M. Youssef, Eds. Heidelberg, Germany: Springer-Verlag, 2001, vol. 2259, Lecture Notes in Computer Science, pp. 103–111.
[31] S. Murphy and M. J. B. Robshaw, "Essential algebraic structure within the AES," in *Advances in Cryptology—CRYPTO 2002*, M. Yung, Ed. Heidelberg, Germany: Springer-Verlag, 2002, vol. 2442, Lecture Notes in Computer Science, pp. 17–38.
[32] N. T. Courtois and W. Meier, "Algebraic attacks on stream ciphers with linear feedback," in *Advances in Cryptology—EUROCRYPT 2003*, E. Biham, Ed. Heidelberg, Germany: Springer-Verlag, 2003, vol. , Lecture Notes in Computer Science, pp. 345–359.
[33] P. C. Kocher, "Timing attacks on implementations of Diffie–Hellman, RSA, DSS, and other systems," in *Advances in Cryptology—CRYPTO'96*, N. Koblitz, Ed. Heidelberg, Germany: Springer-Verlag, 1996, vol. 1109, Lecture Notes in Computer Science, pp. 104–113.
[34] P. C. Kocher, J. Jaffe, and B. Jun, "Differential power analysis," in *Advances in Cryptology—CRYPTO'99*, M. Wiener, Ed. Heidelberg, Germany: Springer-Verlag, 1999, vol. 1666, Lecture Notes in Computer Science, pp. 388–397.
[35] J.-J. Quisquater and D. Samyde, "Electromagnetic analysis (EMA): Measures and counter-measures for smart cards," in *Proc. Int. Conf. Research in Smart Cards* 2001, pp. 200–210.
[36] S. Mangard, "A simple power-analysis (SPA) attack on implementations of the AES key expansion," in *Information Security and Cryptology—ICISC 2002*, P. J. Lee and C. H. Lim, Eds. Heidelberg, Germany: Springer-Verlag, 2002, vol. 2587, Lecture Notes in Computer Science, pp. 343–358.
[37] M.-L. Akkar and C. Giraud, "An implementation of DES and AES, secure against some attacks," in *Cryptographic Hardware and Embedded Systems, CHES 2001*, Ç. K. Koç, D. Naccache, and C. Paar, Eds. Heidelberg, Germany: Springer-Verlag, 2001, vol. 2162, Lecture Notes in Computer Science, pp. 309–318.

Christophe De Cannière received the M.S. degree in electrical engineering from the Katholieke Universiteit Leuven (K.U.Leuven), Belgium, in 2001. He is currently working toward the Ph.D. degree in the COSIC research group, Electrical Engineering Department (K.U.Leuven).

He is supported by the Fonds voor Wetenschappelijk Onderzoek (FWO) Vlaanderen and is working in the field of symmetric encryption under the supervision of Prof. B. Preneel.

Alex Biryukov received the M.Sc. and Ph.D. degrees from the Technion—Institute of Technology, Haifa, Israel, in 1999.

He was a Postdoctoral Researcher in the Computer Science Department, Weizmann Institute of Science, Rehovot, Israel, in 2000–2001, working with Prof. A. Shamir. He is currently a Visiting Assistant Professor and Researcher in the Electrical Engineering Department (ESAT), Katholieke Universiteit Leuven, Leuven, Belgium. He is a designer and codesigner of several generic cryptanalytic algorithms. His main expertise is in cryptanalysis and design of primitives for symmetric cryptography. His other interests are in text-based information retrieval and in computer game playing algorithms.



Bart Preneel received the electrical engineering and doctorate degrees in applied sciences from the Katholieke Universiteit Leuven (K.U.Leuven), Belgium, in 1987 and 1993, respectively.

He is a Professor in the Electrical Engineering Department, Katholieke Universiteit Leuven, and Visiting Professor at TU Graz in Austria. Together with Prof. J. Vandewalle, he heads the research group COSIC at the K.U.Leuven, which currently has 35 members. He is also Vice President of the International Association of Cryptologic Research and Chairman of the Leuven Security Excellence Consortium. Currently he is project manager of ECRYPT, the EU-funded European Network of Excellence on Cryptology and Watermarking. He has held visiting professor positions at the Ruhr-University Bochum, Germany, the University of Bergen, Norway, and the University of Ghent, Belgium. He was also a Research Fellow in the Electrical Engineering and Computer Science Department, University of California, Berkeley. He has authored and coauthored more than 180 articles in international journals and conference proceedings. He is a member of the editorial board of the *Journal of Cryptology* and the *ACM Transactions on Information Security*. His main research interests are cryptology and information security.

Prof. Preneel received the European Information Security Award in the area of academic research in 2003.