The interpolation attack on block ciphers

Chapter *in* Lecture Notes in Computer Science · June 2006

DOI: 10.1007/BFb0052332

CITATIONS
188

READS
538

2 authors:

Thomas Jakobsen
Grazper Technologies ApS
15 PUBLICATIONS **743** CITATIONS
SEE PROFILE

Lars Ramkilde Knudsen
Technical University of Denmark
184 PUBLICATIONS **7,428** CITATIONS
SEE PROFILE

The Interpolation Attack on Block Ciphers*

Thomas Jakobsen[1], Lars R. Knudsen[2]

[1] Department of Mathematics, Building 303, Technical University of Denmark, DK-2800 Lyngby, Denmark, email:jakobsen@mat.dtu.dk.
[2] Katholieke Universiteit Leuven, Dept. Electrical Engineering-ESAT, Kardinaal Mercierlaan 94, B-3001 Heverlee, Belgium, email:knudsen@esat.kuleuven.ac.be.

Abstract. In this paper we introduce a new method of attacks on block ciphers, the interpolation attack. This new method is useful for attacking ciphers using simple algebraic functions (in particular quadratic functions) as S-boxes. Also, ciphers of low non-linear order are vulnerable to attacks based on higher order differentials. Recently, Knudsen and Nyberg presented a 6-round prototype cipher which is provably secure against ordinary differential cryptanalysis. We show how to attack the cipher by using higher order differentials and a variant of the cipher by the interpolation attack. It is possible to successfully cryptanalyse up to 32 rounds of the variant using about 2^{32} chosen plaintexts with a running time less than 2^{64}. Using higher order differentials, a new design concept for block ciphers by Kiefer is also shown to be insecure. Rijmen et al presented a design strategy for block ciphers and the cipher SHARK. We show that there exist ciphers constructed according to this design strategy which can be broken faster than claimed. In particular, we cryptanalyse 5 rounds of a variant of SHARK, which deviates only slightly from the proposed SHARK.

1 Introduction

In an r-round iterated cipher the ciphertext is computed by iteratively applying in r rounds a *round function* G to the plaintext, s.t.

$$C_i = G(K_i, C_{i-1}),$$

where C_0 is the plaintext, K_i is the ith round key, and C_r is the ciphertext. A special kind of iterated ciphers are the **Feistel** ciphers. A Feistel cipher with block size $2n$ and r rounds is defined as follows. Let C_0^L and C_0^R be the left and right hand halves of the plaintext, respectively, each of n bits. The round function G operates as follows

$$C_i^L = C_{i-1}^R$$
$$C_i^R = F(K_i, C_{i-1}^R) + C_{i-1}^L,$$

* The work in this paper was initiated while the authors were visiting the Isaac Newton Institute, Cambridge, U.K., February 1996.

and the ciphertext is the concatenation of C_r^R and C_r^L. Note that F can be any function taking as arguments an n-bit text and a round key K_i and producing n bits. '+' is a commutative group operation on the set of n bit blocks. For the remainder of this paper we will assume that '+' is the exclusive-or operation ($\oplus$).

Based on the use of a quadratic function over a Galois field, Knudsen and Nyberg demonstrated in [10] how to construct a cipher which is provably secure against differential cryptanalysis [1]. The cipher is a Feistel cipher with the function F given by $F : \mathrm{GF}(2^{32}) \to \mathrm{GF}(2^{32})$ with

$$F(k, x) = d(f(e(x) \oplus k)),$$

where $f : \mathrm{GF}(2^{33}) \to \mathrm{GF}(2^{33})$, $f(x) = x^3$, $k \in \mathrm{GF}(2^{33})$, $e : \mathrm{GF}(2^{32}) \to \mathrm{GF}(2^{33})$ is a function which extends its argument by concatenation with an affine combination of the input bits, and $d : \mathrm{GF}(2^{33}) \to \mathrm{GF}(2^{32})$ discards one bit from its argument. We call this cipher $\mathcal{KN}$.

Also, we will consider the cipher with round function given by $F_k(x) = f(x \oplus k)$ where $f : \mathrm{GF}(2^{32}) \to \mathrm{GF}(2^{32})$, $f(x) = x^3$, i.e., the cubing function's input is not extended and the output not truncated as in the previous case. We call this cipher $\mathcal{PURE}$.

Both ciphers are secure against differential attacks [10]. Also, both ciphers are secure against the linear attack [7], which follows from [9].

In [10] the cipher $\mathcal{KN}$ is defined to be used with 6 rounds and since $f(x)$ is differentially 2-uniform, it is possible to prove that this yields a provably secure cipher (secure against conventional differential cryptanalysis). The same holds for $\mathcal{PURE}$. However, in both cases the non-linear order of the output is low with respect to the input and this can be exploited to mount an attack.

In the following, $x = (x_L, x_R)$ denotes the plaintext where x_L and x_R denote the left and right hand side of x, respectively. Similarly, $y = (y_L, y_R)$ denotes the ciphertext. By the *reduced cipher*, we denote the cipher that one gets by removing the final round of the original cipher. The output from this cipher is denoted $\tilde{y} = (\tilde{y}_L, \tilde{y}_R)$.

The attacks presented in this paper are classified according to the taxonomy of [4]. That is, by a *key-recovery attack* we mean that an attacker finds the secret key. By a *global deduction* we mean that an attacker finds an algorithm, which encrypts any plaintext into a valid ciphertext without knowing the secret key. By an *instance deduction* we mean that an attacker finds an algorithm, which encrypts a subset of all plaintexts into valid ciphertexts without knowing the secret key. In the key-recovery attacks we try to guess the last-round key. The guess is then used to decrypt the ciphertext by one round and in this way one (hopefully) obtains the output from the reduced cipher. If there exists a method to distinguish whether this is the actual output from the reduced cipher or not, then we can find the last-round key. Once this key has been found, attacks similar to the ones we present can be mounted on a cipher one round shorter than the original. As the measurement of the time needed by an attack, we use the total number of encryptions of the attacked block cipher.

This paper is organised as follows. In § 2 we give new attacks based on higher order differentials. We apply the attacks to the cipher $\mathcal{KN}$ by Knudsen and

Nyberg [10] and to a cipher by Kiefer [3]. In § 3 we present our new attack on block ciphers, the interpolation attack. We apply the attack to a cipher, provably secure against differential and linear attacks. Also, we apply our methods to a slightly modified version of the cipher SHARK [11]. We conclude in § 4.

2 Attacks Using Higher Order Differentials

In [6] Lai gave a definition of higher order derivatives of discrete functions. Later Knudsen used higher order differentials to cryptanalyse ciphers presumably secure against conventional differential attacks, i.e. attacks based on first order differentials [5]. In this section we give an extension of Knudsen's attacks and apply it in an attack on the cipher $\mathcal{KN}$. We refer to [6, 5] for the definitions of higher order differentials.

Consider a Feistel cipher with block size $2n$. Suppose that x_R is kept constant and consider the right hand side $\tilde{y}_R$ of the output from the reduced cipher. Since x_R is a constant, the bits in $\tilde{y}_R$ are all expressible as polynomials $\mathrm{GF}(2)[x_1, x_2, \ldots, x_n]$ in the bits of $x_L = (x_1, x_2, \ldots, x_n)$. Assume that these polynomials have degree not higher than d. Then according to [6, Proposition 2] (see also [5]), we have

$$\sum_{x_L \in \mathcal{L}_d} p(x_L) = c, \tag{1}$$

where $\mathcal{L}_d$ denotes a d-dimensional subspace of $\mathrm{GF}(2)^n$, c is the same for any space parallel to $\mathcal{L}_d$, and p is a function which computes the output from the reduced cipher. It follows that

$$\sigma(w) = \sum_{x_L \in \mathcal{L}_{d+1}} p(x_L + w) = 0 \text{ for all } w \in GF(2)^n \tag{2}$$

if and only if $p(x)$ is a polynomial of degree d or lower. In the following algorithm, the variables $x = (x_L, x_R)$ and $y = (y_L, y_R)$ hold the plaintext and the ciphertext, respectively. L is a full rank $(d+1) \times n$ matrix over $\mathrm{GF}(2)$ and F the round function.

1. Let x_R and w be n-bit constants.
2. For all $a \in \mathrm{GF}(2)^{d+1}$:
   (a) Let $x_L = aL + w$.
   (b) Obtain the ciphertext $y(a)$ of plaintext (x_L, x_R).
3. For all values, k, of the last-round key:
   (a) Let $\sigma = 0$.
   (b) For all $a \in \mathrm{GF}(2)^{d+1}$:
      i. Let $y = y(a)$.
      ii. Let $\tilde{y}_R = y_L \oplus F(k, y_R)$.
      iii. Let $\sigma = \sigma \oplus \tilde{y}_R$.

The key for which σ ends up being zero is the correct last-round key with a high probability. Consequently, for every possible value k of the last-round key, we check whether the corresponding value of σ is zero, and if it is, then we have found the correct key with high probability. If one wants a higher level of certainty, the algorithm is simply repeated with another choice of w. This method is easily generalised to an iterated cipher, and we get the following result, extending that of [5, Th. 11].

Theorem 1. *Given an iterated block cipher, let d denote the polynomial degree of the ciphertext bits of the round next to the last as a function of the plaintext bits. Furthermore, let b denote the number of last-round key bits. Assume that the polynomial degree of the ciphertext bits increases with the number of rounds. Then there exists a d-th order differential attack of average time complexity 2^{b+d} requiring 2^{d+1} chosen plaintexts which will successfully recover the last-round key.*

Proof. We give the proof in the case of a Feistel cipher, from which the general case follows. Consider the iteration (3b). Let k denote the correct value of the last-round key, and let k' denote any wrong value. Then

$$\begin{aligned} \tilde{y}_R &= y_L \oplus F(k, y_R) \\ \tilde{y'}_R &= y_L \oplus F(k', y_R) \\ &= \tilde{y}_R \oplus F(k, y_R) \oplus F(k', y_R). \end{aligned}$$

The difference between $\tilde{y}_R$, obtained using the correct key, and $\tilde{y'}_R$, obtained with a wrong key, is two applications of the function F. Since by assumption the polynomial degree increases with the number of rounds, one can expect that σ will be zero only for the correct value of the last-round key with a high probability. Running an algorithm similar to the one above takes 2^{d+1} steps for each value of the last-round key. On the average, we have to test half of the keys before finding the correct one, from which the time complexity follows.

The attack can be improved by a factor of two, if the constant of Equation (1) can be predicted. In that case the iterations (2) and (3b) of the above algorithm are performed only for all $a \in \mathrm{GF}(2^d)$. The key for which $\sigma = c$ will be the correct key with a high probability. For most ciphers, depending on the F-function, there are possible extensions to the above attack. It may be possible to perform the attack for only a subset of the last-round key, and also it may be possible to search for (a subset of) the first-round key.

In the following we apply the attack to the cipher $\mathcal{KN}$. We choose plaintexts where the right halves are fixed. Since the output bits from the round function are only quadratic in the input bits, the polynomials in the attack described above on the 6 round version have degree not higher than 8. Therefore the attack requires only $2^{8+1} = 512$ chosen plaintexts and an average running time of order 2^{41}. A variant of the attack guessing for the keys in the last two rounds requires about 32 chosen plaintexts and an average running time of order 2^{70}. Similarly, there are attacks on the 7 and 8 rounds versions of $\mathcal{KN}$, the complexities are

# Rounds	# Chosen plaintexts	Running time
6	2^9	2^{41}
6	2^5	2^{70}
7	2^{17}	2^{49}
7	2^9	2^{74}
8	2^{17}	2^{82}

Table 1. Higher order differential attacks on the Knudsen-Nyberg cipher.

given in Table 1. The attack on $\mathcal{KN}$ using higher order differentials has been implemented, and it recovers the last round key as predicted. Note that these attacks are applicable to ciphers with any block size $2n$, as long as the number of chosen plaintexts is less than 2^n. The bigger the block size the more rounds can be attacked.

We now attack the scheme by Kiefer [3] by the use of higher order differentials[3]. The cipher is probabilistic and uses the following encryption rule:

$$m_i \mapsto (F(k) \oplus r_i, f_k(r_i) \oplus m_i), \tag{3}$$

where $F : \mathrm{GF}(2^n) \to \mathrm{GF}(2^n)$ is a one-way function, $f_k : \mathrm{GF}(2^n) \to \mathrm{GF}(2^n)$ is a function depending on the key $k \in \mathrm{GF}(2^n)$ in some complex way, $r_i \in \mathrm{GF}(2^n)$ is a random value, and $m_i \in \mathrm{GF}(2^n)$ is a message block. The function f_k has the form $f_k = \pi_k \circ g$ where $\pi_k : \mathrm{GF}(2^n) \to \mathrm{GF}(2^n)$ is a bitwise linear transform depending on k and $g : \mathrm{GF}(2^n) \to \mathrm{GF}(2^n)$ is a public, almost perfectly non-linear function of the form $g(x) = x^{2^s+1}$ for some s.

Assume that we know enough plaintext to have four pairs on the form

$$(a_i, b_i) = (F(k) \oplus r_i, f_k(r_i)), \ i = 1, \ldots, 4 \tag{4}$$

such that $a_1 \oplus a_2 = a_3 \oplus a_4$. Define $\beta = \bigoplus_{i=1}^4 b_i$ and $\gamma = \bigoplus_{i=1}^4 g(r_i)$. Then

$$\beta = \bigoplus_{i=1}^{4} b_i = \pi_k \left(\bigoplus_{i=1}^{4} g(r_i) \right) = \pi_k(\gamma). \tag{5}$$

Since $\{a_1, \ldots, a_4\}$ is a two-dimensional subspace of $\mathrm{GF}(2^n)$, the elements in $\{r_1, \ldots, r_4\}$ also constitute a two-dimensional subspace. Note also that the Hamming weight of the exponent in the definition of g expressed as a binary number is only two, implying that the output bits are only quadratic in the input bits. By Equation (1), this implies that we can compute the value of γ.

If repeated n times, we will have n corresponding pairs of β and γ. This makes it possible to solve Equation (5) with respect to the unknown function π_k (it is a linear transform). After having found π_k, we can invert f_k and thus obtain a value of r_i. Using this, we compute $F(k)$ and the system is broken.

It remains to compute the minimum number t of known plaintexts needed to obtain n times four pairs (a_i, b_i) with the required property; recall that the

[3] This attack was presented at the rump session of Pragocrypt'96.

cipher is probabilistic and thus we have no control over the values of r_i. By using a birthday paradox type argument it can be shown that $t \approx (n \cdot 2^{n+2})^{\frac{1}{4}}$. For a typical block size of $n = 64$ this gives $t \approx 2^{18}$.

3 The Interpolation Attack

In this section, we introduce a new attack on block ciphers. The attack is based on the following well-known formula.

Let R be a field. Given $2n$ elements $x_1, \ldots, x_n, y_1, \ldots, y_n \in R$, where the x_is are distinct. Define

$$f(x) = \sum_{i=1}^{n} y_i \prod_{1 \le j \le n, j \ne i} \frac{x - x_j}{x_i - x_j}. \tag{6}$$

Then $f(x)$ is the only polynomial over R of degree at most $n - 1$ such that $f(x_i) = y_i$ for $i = 1, \ldots, n$. Equation (6) is known as the *Lagrange interpolation formula* (see e.g. [2, page 185]).

In the *interpolation attacks* presented in this paper we construct polynomials using pairs of plaintexts and ciphertexts. We will assume that the time needed to construct these polynomials is small compared to the time needed to do the encryptions of the plaintexts needed in the attack.

3.1 Global and instance deduction

Consider the cipher $\mathcal{PURE}$ with r rounds. We exploit the fact that the exclusive-or operation used in the cipher corresponds to addition over a finite field with characteristic 2. Consequently, the cipher consists of simple algebraic operations only, and hence each of the two halves of the ciphertext y, e.g., the left hand part, can be described as a polynomial $p(x_L, x_R) \in \mathrm{GF}(2^{32})[x_L, x_R]$ of the plaintext with at most $3^{2r-1}+3^r+3^{r-1}+1$ coefficients. Note, that degrees of x_R and x_L are at most 3^r and 3^{r-1}, respectively. Thus, we can reconstruct this polynomial by considering at most $3^{2r-1} + 3^r + 3^{r-1} + 1$ plaintext/ciphertext pairs (p/c-pairs) using, e.g., Lagrange interpolation. With $r = 6$ the attack needs at most 2^{18} known p/c-pairs, which yields an algorithm for a global deduction. Note that the number of coefficients will be lower than specified, since not all elements $x_L^i x_R^j$ for $0 \le i \le 3^r$ and $0 \le j \le 3^{r-1}$ will appear in the polynomial.

We have the following more general theorem.

Theorem 2. *Consider an iterated block cipher with block size m. Express the ciphertext as a polynomial of the plaintext and let n denote the number of coefficients in the polynomial. If $n \le 2^m$, then there exists an interpolation attack of time complexity n requiring n known plaintexts encrypted with a secret key K, which finds an algorithm equivalent to encryption (or decryption) with K.*

In a chosen plaintext variant of this attack it is possible for an attacker to establish polynomials with a reduced number of coefficients by fixing some of the bits in the chosen plaintexts. In that case, the result is an instance deduction,

since the obtained algorithm can only encrypt plaintexts for which a number of bits are fixed to a certain value. As as example, $\mathcal{PURE}$ can be attacked in such a way using only 730 chosen p/c-pairs. Subsequently, the attacker has an algorithm, which encrypts 2^{32} plaintexts without knowing the secret key.

3.2 Key-recovery

In this section we extend the method of the previous section to a key-recovery attack.

Consider first a known plaintext attack. Instead of specifying the ciphertext as a function of the plaintext, we express the output from the reduced cipher $\tilde{y}$ as a polynomial $p(x) \in \mathrm{GF}(2^m)[x]$ of the plaintext. Assume that this polynomial has degree d and that $(d+1)$ known p/c-pairs are available. Then for all values of the last-round key one decrypts the ciphertexts one round and tries to construct the polynomial. With one extra p/c-pair one checks whether the polynomial is correct. If this is the case, then the correct value of the last-round key has been found with a high probability, by reasoning similarly as in the proof of Theorem 1.

The chosen plaintext variant of this attack is quite similar. Let us illustrate the method with an example. Once again, consider the cipher $\mathcal{PURE}$ with 6 rounds. Assume that the right hand half x_R of the plaintext is fixed (that is, we consider a chosen plaintext attack), and consider the right hand side of the output $\tilde{y}_R = p(x_L)$ from the reduced cipher expressed as a polynomial $p(x_L) \in \mathrm{GF}(2^{32})[x_L]$. This polynomial has degree at most $3^3 = 27$ since the degree does not increase in the first round and since ty_R equals the left half of the output of the fourth round. Consequently, 28 pairs of corresponding values of x_L and $\tilde{y}$ are enough to determine it uniquely (using Lagrange interpolation).

We then test whether $\tilde{y}$ is actually output from the reduced cipher or not. This is done by verifying whether a 29-th p/c-pair agrees with the obtained polynomial. If it does, then we assume that we have found the correct key. The average time complexity is $29 \times 2^{32-1} \approx 2^{36}$.

More generally, we have the following theorem.

Theorem 3. *Consider an iterated block cipher of size m. Express the output from the round next to the last as a polynomial of the plaintext and let n denote the number of coefficients in the polynomial. Furthermore, let b denote the number of last-round key bits. Then there exists an interpolation attack of average time complexity $2^{b-1}(n+1)$ requiring $n+1$ known (or chosen) plaintexts which will successfully recover the last-round key.*

Similar to the attack of Theorem 1 it may be possible to perform the attack for only a subset of the last-round key, and also it may be possible to search for (a subset of) the first-round key, depending on the structure of the round function.

3.3 Meet-in-the-middle approach

The attacks described in this section are extensions of the attacks in the previous sections using a meet-in-the-middle technique. We describe only the extension

of the key-recovery attack; the extension of the global and instance deductions follow easily.

Once more, we try guessing the correct last-round key and use this to (hopefully) obtain $\tilde{y}$, the output from the reduced cipher. In the following, only the verification of $\tilde{y}$ is described. Given an iterated cipher of r rounds, let z denote the output of round s, where $s \leq (r-1)$. The value of z is expressible via the plaintext x as a polynomial $g(x) \in \mathrm{GF}(2^m)[x]$ where m is the block size. Similarly, z can be expressed as a polynomial $h(\tilde{y}) \in \mathrm{GF}(2^m)[\tilde{y}]$ of the output $\tilde{y}$ of the reduced cipher. Let the degree of $g(x)$ be d_g, the degree of $h(\tilde{y})$ be d_h and let $d_{gh} = d_g + d_h$. Thus, the following equation

$$g(x) = h(\tilde{y}) \tag{7}$$

has at most $d_{gh} + 2$ unknowns. The equation is solvable up to a multiplication and an addition of both g and h with a constant. Therefore, to ensure that we obtain a non-trivial and unique solution, we set the coefficient corresponding to the highest exponent equal to 1 and the constant term equal to 0. After this, we solve the equation by using d_{gh} known or chosen plaintexts. We then check whether yet another p/c-pair $(x, \tilde{y})$ obeys $g(x) = h(\tilde{y})$. If it does, then we assume that we have guessed the correct value of the last-round key.

Again, let us illustrate the attack on the cipher $\mathcal{PURE}$ with 6 rounds. Assume that the right hand half x_R of the plaintext is fixed (that is we consider a chosen plaintext attack.) Let z_L denote the left half of the output from round four. The value of z_L is expressible via the plaintext as a polynomial $g(x_L) \in \mathrm{GF}(2^{32})[x_L]$. This polynomial has degree at most 3^2, i.e. there are at most 10 non-zero coefficients in $g(x_L)$. Similarly, z_L can be expressed as a polynomial $h(\tilde{y}_L, \tilde{y}_R) \in \mathrm{GF}(2^{32})[\tilde{y}_L, \tilde{y}_R]$ of the output from the reduced cipher. It follows that $h(\tilde{y}_L, \tilde{y}_R) = \tilde{y}_L^3 \oplus a\tilde{y}_L^2 \oplus b\tilde{y}_L \oplus c \oplus \tilde{y}_R$, where a, b, and c are some key-dependent constants. Thus, there are at most $10 + 3 = 13$ unknown coefficients of the equation

$$g(x_L) = h(\tilde{y}_L, \tilde{y}_R) \tag{8}$$

Setting the constant term of g to equal 0 (the coefficient corresponding to the highest exponent in h has already been found to equal 1), we proceed to solve the resulting system of equations by using 12 p/c-pairs from the reduced cipher. This gives us the polynomials g and h. We then check whether yet another p/c-pair $(x, \tilde{y})$ obeys $g(x_L) = h(\tilde{y}_L, \tilde{y}_R)$. If it does, then we assume that we have guessed the correct key.

Similar attacks can be applied to versions of $\mathcal{PURE}$ with up to 32 rounds. Consider the version with 32 rounds. Let $g(x_L) \in \mathrm{GF}(2^{32})[x_L]$ be an expression of the left half z_L of the output from round 22. The degree of this polynomial is at most 3^{20}. Let $h(\tilde{y}_L, \tilde{y}_R) \in \mathrm{GF}(2^{32})[\tilde{y}_L, \tilde{y}_R]$ be an expression of z_L from the output of the reduced cipher. In the algebraic normal form of $h(\tilde{y}_L, \tilde{y}_R)$, the number of exponents in $\tilde{y}_L$ and $\tilde{y}_R$ is at most $(3^9 + 1)$ and $(3^{10} + 1)$, respectively. Thus the number of coefficients in $h(\tilde{y}_L, \tilde{y}_R)$ is at most $(3^9+1)(3^{10}+1) \approx 3^{19}$. This means that the number of coefficients in Equation (8) is at most $3^{20} + 3^{19} \approx 2^{32}$. I.e.,

the average time complexity for this attack is about 2^{63} and it requires about 2^{32} chosen plaintexts.

We obtain the following general result.

Theorem 4. *Consider an iterated block cipher of block size m with r rounds. Express the output from round s, $s \leq r-1$, as a polynomial of the plaintext and let n_1 denote the number of coefficients in the polynomial. Also, express the output from round s as a polynomial of the output from round $(r-1)$, and let n_2 denote the number of coefficients in the polynomial. Furthermore, set $n = n_1 + n_2$ and let b denote the number of last-round key bits. Then there exists an interpolation attack of average time complexity $2^{b-1}(n-1)$ requiring $(n-1)$ known (chosen) plaintexts which will successfully recover the last-round key.*

In the following section we describe a variant of the interpolation attack.

3.4 Attacks on modified SHARK

The iterated cipher SHARK was described by Rijmen, Daemen, et al in [11]. The cipher has block size nm bits and each round has a non-linear layer and a diffusion layer. The non-linear layer consists of n parallel m-bit S-boxes. The diffusion layer consists of an nm-bit linear mapping constructed from the Reed-Solomon code. There are two suggested ways to introduce the keys into the cipher. The first is by a simple exclusive-or with the inputs to the S-boxes, the other uses a key-dependent affine mapping. Also, an output transformation is applied after the last round of SHARK. The transformation consists of a key addition and an inverse diffusion layer.

The design strategy of SHARK is to consider each component of the cipher separately. It is argued "The non-linear layer has uniform non-linear properties, such that when measuring the resistance of the cipher against cryptanalysis we don't have to take the details of the interaction between the non-linear and the diffusion layer into account."[11]. Furthermore, "If, for example, the S-boxes are replaced by other S-boxes, with equivalent non-linearity properties, the resistance of the cipher remains constant"[11].

We will denote by SHARK(n, m, r) the version with block size nm bits using n parallel m-bit S-boxes in r rounds. In [11] an implementation SHARK$(8, 8, r)$ (64 bit blocks) is given. The 8 S-boxes are identical and constructed from the permutation $f : \mathrm{GF}(2^m) \rightarrow \mathrm{GF}(2^m)$ given by $f(x) = x^{-1}$. The cipher is analysed with respect to linear and differential attacks, and it is argued that 8 rounds of SHARK$(8, 8, r)$ give a security level comparable to that of triple-DES, and from [11, Table 1] it follows that 4 rounds of this version give a security level comparable to that of DES.

In the following we will show that there are many instances of SHARK that can be broken significantly faster than expected.

First of all, the number of rounds of SHARK must be determined with respect to the non-linear order of the S-boxes. Assume that the outputs of the S-box have non-linear order d in the input bits. Since the S-boxes represent the only

non-linear component in SHARK, the non-linear order of the ciphertexts after r rounds of encryption will be at most d^r. To avoid attacks based on higher order differentials it must be ensured that d^r is high, preferably that $d^r \geq nm$. Thus, for a 64 bit block cipher, if $d = 2$, e.g. using the cubing function in a Galois field, the number of rounds must be at least 6.

We consider in the following versions of SHARK where the keys are mixed with the texts by the exclusive-or operation. Once again, we make use of the fact, that exclusive-or is equivalent to addition over a finite field of characteristic 2. We will show that there are instances of SHARK(n, m, r), for which the interpolation attacks are applicable. We consider 64-bit versions using as S-box $f(x) = x^{-1}$ in GF(2^m). This is the S-box suggested in [11], but, as it is also said "To remove the fixed points $0 \rightarrow 0$ and $1 \rightarrow 1$ an invertible transformation is applied to the output bits of the S-box." In what we are about to show, these fixed points play no role, so according to the design strategy of SHARK, variants with $f(x)$ as S-box without the invertible transformation should give equivalent security. We stress that the attacks we are about to present are not applicable to the specific instance of SHARK presented in [11].

The interpolation attack described so far in this paper work well for ciphers of low algebraic degree. The inverse permutation in a Galois field has a high algebraic degree, note that $f(x) = x^{-1} = x^{2^m-2}$ in GF(2^m). However, as we will show, there are variants of the interpolation attack, which work for these functions. These attacks depend only on the number of S-boxes and of the number of rounds in the cipher.

Consider first a version with $n = 1$. It follows by easy calculations that the ciphertext y *after any number of rounds* can be expressed as a fraction of polynomials of the plaintext x (or similarly, x can be expressed as a polynomial of y) as follows

$$y = \frac{x \oplus a}{bx \oplus c} \tag{9}$$

where a, b, c are key-dependent constants. These three constants can be found using the interpolation attack with only 4 known p/c-pairs[4] by considering and solving $y \cdot (bx \oplus c) = (x \oplus a)$. The result is a global deduction, i.e. an algorithm that encrypts (decrypts) any plaintext (ciphertext).

Next consider a version with $n = 2$. Let x_L and x_R denote the left and right halves of the plaintext, respectively, and let $y_{i,L}$ and $y_{i,R}$ denote the left and right halves of the ciphertext after i rounds of encryption. In general we get

$$y_{i,L} = \frac{p_{i,1}(x_L, x_R)}{p_{i,2}(x_L, x_R)} \tag{10}$$

and similarly for $y_{i,R}$, where $p_{i,1}, p_{i,2} \in \text{GF}(2^{32})[x_L, x_R]$. It remains to show how many coefficients there are in the two polynomials. First note that the number of coefficients in $p_{i,1}$ is at most the number of coefficients in $p_{i,2}$. Consider the

[4] In [8] a similar cipher was investigated. It was explained that this cipher could be solved with a number of known plaintexts linear in the number of rounds. Our results shows that this number is a constant.

algebraic normal form of $p_{i,2}$ and assume that the largest exponents of x_L and x_R are $e^i_{x_L}$ respectively $e^i_{x_R}$. Then the number of coefficients in $p_{i,2}$ is at most $(e^i_{x_L}+1)\cdot(e^i_{x_R}+1)$. From the description of SHARK [11] it follows that

$$y_{i,L} = \frac{a_1}{y_{(i-1),L} \oplus k_{i,1}} \oplus \frac{a_2}{y_{(i-1),R} \oplus k_{i,2}} \tag{11}$$

$$= \frac{p_{i,1}}{(y_{(i-1),L} \oplus k_{i,1}) \cdot (y_{(i-1),R} \oplus k_{i,2})}, \tag{12}$$

where $k_{i,j}$ are the round keys and a_1, a_2 some constants. Now it is easy to see that for $i > 1$ $e^i_{x_L} \leq 2 \cdot e^{i-1}_{x_L}$ and since $e^1_{x_L} = e^1_{x_R} = 1$, one gets $e^i_{x_L} \leq 2^{i-1}$ and $e^i_{x_R} \leq 2^{i-1}$. Therefore, the number of coefficients in $p_{i,2}$ is at most $(2^{i-1}+1)^2$, which also upper bounds the number of coefficients in $p_{i,1}$. In order to be able to solve Equation (10) one would need at most $2 \cdot (2^{i-1}+1)^2$ plaintexts and their corresponding ciphertexts. Note that the same pairs can be used to solve a similar equation for $y_{i,R}$. Consider versions of the cipher with n S-boxes. One finds by calculations similar as above that the number of known plaintexts needed to solve Equation (10) is $2 \cdot (n^{i-1}+1)^n$. The number of coefficients in the polynomials used in our attacks increases with the number of diffusion layers in the cipher. Note that because of the inverse diffusion layer in the output transformation there are only $r-1$ diffusion layers in an r-round version of SHARK. To sum up, the number of known plaintexts for the interpolation attack on an r-round version yielding a global deduction is

$$2 \cdot (n^{r-2}+1)^n.$$

It follows that the attack is independent of the sizes of the S-boxes, and it depends only the number of S-boxes and the number of rounds.

The interpolation attack with the meet-in-the-middle technique can be applied also for these ciphers. We consider the interpolation attack with known plaintexts. One first establishes

$$\frac{q_{j,1}(y_1,\ldots,y_n)}{q_{j,2}(y_1,\ldots,y_n)} = \frac{p_{i,1}(x_1,\ldots,x_n)}{p_{i,2}(x_1,\ldots,x_n)}, \tag{13}$$

i.e., expressions of the ciphertexts in one middle round, where $i+j = r-1$, using polynomials of both the plaintext and the ciphertext. Subsequently, one can solve the following systems of equations

$$q_{j,1}(y_1,\ldots,y_n) \cdot p_{i,2}(x_1,\ldots(,x_n) = p_{i,1}(x_1,\ldots x_n) \cdot q_{j,2}(y_1,\ldots,y_n). \tag{14}$$

The number of known plaintexts required to solve (14) is

$$2 \cdot (n^{r_1-1}+1)^n \cdot (n^{r_2-1}+1)^n,$$

where $r_1 + r_2 = r-1$ and $r_1, r_2 \geq 1$.

The round keys for SHARK are typically quite big, so the general key-recovery attack described earlier in this paper may be impractical. However, it is possible to perform the attack for only a subset of the first-round and/or last-round keys.

# Rounds	# S-boxes	Known plaintexts	
any	1	3	
6	2	2^9	
6	4	2^{27}	(+)
3	8	2^{17}	(+)
4	8	2^{35}	(+)
5	8	2^{52}	(+)
6	8	2^{75}	(+)
7	8	2^{98}	(+)
8	8	2^{121}	(+)

Table 2. Complexities of the interpolation attack on variants of SHARK using as S-box $f(x) = x^{-1}$. (+) Meet-in-the-middle approach.

As an example, one can repeat the attack for all values of the first s words of the first-round key and express the ciphertext (of a middle round) as a polynomial $p_{i,1}(S(x_1 \oplus k_1), \ldots, S(x_s \oplus k_s), x_{s+1}, \ldots x_n)$, where $S(\cdot)$ are the S-boxes and x_i are the plaintext words. The values of the key words for which the interpolation succeeds are candidates for the secret key, and the attack is repeated sufficiently many times until one value of the secret key is found.

In Table 2 we give the complexities of the interpolation attack on variants of SHARK using as S-box $f(x) = x^{-1}$ in $\mathrm{GF}(2^m)$. It follows that using 8 S-boxes, the 64-bit variant with up to 5 rounds and the 128-bit variant with up to 8 rounds are (theoretically) vulnerable to our attacks. The number of required plaintexts of the key-recovery attack is a little less than indicated variants and the workload of the attack is a little higher. We will not go into further details here.

In a chosen plaintext attack the number of coefficients in the polynomials used in the attack can be reduced by fixing some plaintext bits. As examples, there exist interpolation attacks on the variant with 8 S-boxes and 4 rounds using about 2^{21} chosen plaintexts and on the variant with 8 S-boxes and 7 rounds using about 2^{61} chosen plaintexts. In this attack we fix four of the eight plaintext words, so for a 64-bit block cipher the interpolation will work only if the needed number of plaintexts is less than 2^{32} and for a 128-bit block cipher less than 2^{64} plaintexts.

We have demonstrated that certain instantiations of SHARK are insecure. Our results also demonstrate a case where the use of bigger and fewer S-boxes does not result in more secure ciphers. Finally, we note that the designers of SHARK expressed their concern with the use of the inverse in a Galois field as S-boxes: "This may create uneasy feelings, but we are not aware of any vulnerability caused by this property. For the time being we challenge cryptanalysts to demonstrate any vulnerability caused by this property." [11]. Challenge taken!

4 Concluding Remarks

We introduced a new attack on block ciphers, the interpolation attack. We demonstrated the attack on slightly modified versions of a cipher proposed by Knudsen

and Nyberg and of a cipher proposed by Rijmen, Daemen et al. These modifications do not violate the design principles of the original ciphers and are as secure with respect to the security measures proposed by the authors. Also, we presented an improved variant of differential attacks based on higher order differentials, which was used to cryptanalyse the (unmodified) cipher by Knudsen and Nyberg and a cipher by Kiefer.

One might try to find a probabilistic version of the interpolation attack that would also work when the output of the cipher is expressible as a polynomial of low degree in only a fraction of the cases. However, it looks like this attack would require an effective maximum likelihood decoding algorithm for higher order Reed-Muller codes and such an algorithm is not known to exist.

Finally, it should be mentioned that with the use of Newton interpolation instead of Lagrange interpolation one can speed up the attacks slightly.

References

1. E. Biham and A. Shamir. *Differential Cryptanalysis of the Data Encryption Standard.* Springer Verlag, 1993.
2. P.M. Cohn. *Algebra, Volume 1.* John Wiley & Sons, 1982.
3. K. Kiefer. A New Design Concept for Building Secure Block Ciphers. In J. Pribyl, editor, *Proceedings of the 1st International Conference on the Theory and Applications of Cryptology, PRAGOCRYPT'96, Prague, Czech Republic*, pages 30–41. CTU Publishing House, 1996.
4. L.R. Knudsen. *Block Ciphers – Analysis, Design and Applications.* PhD thesis, Aarhus University, Denmark, 1994.
5. L.R. Knudsen. Truncated and higher order differentials. In B. Preneel, editor, *Fast Software Encryption - Second International Workshop, Leuven, Belgium, LNCS 1008*, pages 196–211. Springer Verlag, 1995.
6. X. Lai. Higher order derivatives and differential cryptanalysis. In *Proc. "Symposium on Communication, Coding and Cryptography", in honor of James L. Massey on the occasion of his 60'th birthday, Feb. 10-13, 1994, Monte-Verita, Ascona, Switzerland*, 1994.
7. M. Matsui. Linear cryptanalysis method for DES cipher. In T. Helleseth, editor, *Advances in Cryptology - Proc. Eurocrypt'93, LNCS 765*, pages 386–397. Springer Verlag, 1993.
8. K. Nyberg. Differentially uniform mappings for cryptography. In T. Helleseth, editor, *Advances in Cryptology - Proc. Eurocrypt'93, LNCS 765*, pages 55–64. Springer Verlag, 1993.
9. K. Nyberg. Linear approximations of block ciphers. In A. De Santis, editor, *Advances in Cryptology - Proc. Eurocrypt'94, LNCS 950*, pages 439–444. Springer Verlag, 1994.
10. K. Nyberg and L.R. Knudsen. Provable security against a differential attack. *The Journal of Cryptology*, 8(1):27–38, 1995.
11. V. Rijmen, J. Daemen, B. Preneel, A. Bosselaers, and E. De Win. The cipher SHARK. In Gollmann D., editor, *Fast Software Encryption, Third International Workshop, Cambridge, U.K., February 1996, LNCS 1039*, pages 99–112. Springer Verlag, 1996.