Known and Chosen Key Differential Distinguishers for Block Ciphers

Ivica Nikolić[1]*, Josef Pieprzyk[2], Przemysław Sokołowski[2,3], Ron Steinfeld[2]

[1] University of Luxembourg, Luxembourg
[2] Macquarie University, Australia
[3] Adam Mickiewicz University, Poland

ivica.nikolic@uni.lu, josef.pieprzyk@mq.edu.au, przemyslaw.sokolowski@mq.edu.au, ron.steinfeld@mq.edu.au

Abstract. In this paper we investigate the differential properties of block ciphers in hash function modes of operation. First we show the impact of differential trails for block ciphers on collision attacks for various hash function constructions based on block ciphers. Further, we prove the lower bound for finding a pair that follows some truncated differential in case of a random permutation. Then we present open-key differential distinguishers for some well known round-reduced block ciphers.

Keywords: Block cipher, differential attack, open-key distinguisher, Crypton, Hierocrypt, SAFER++, Square.

1 Introduction

Block ciphers play an important role in symmetric cryptography providing the basic tool for encryption. They are the oldest and most scrutinized cryptographic tool. Consequently, they are the most trusted cryptographic algorithms that are often used as the underlying tool to construct other cryptographic algorithms. One such application of block ciphers is for building compression functions for the hash functions.

There are many constructions (also called hash function *modes*) for turning a block cipher into a compression function. Probably the most popular is the well-known Davies-Meyer mode. Preneel et al. in [27] have considered all possible modes that can be defined for a single application of n-bit block cipher in order to produce an n-bit compression function. They have found that there are 12 modes that are resistant against generic attacks. Later these findings have been formally proven in [7]. To make hash functions resistant against the birthday-paradox attack, it is better to use double-block modes. Basic double-block modes have been proposed in [8, 14, 20]. Note that the Tandem-DM mode has been proven to be collision resistant in [12], while a weakness in MDC-2 was found in [17].

Proofs of security of the above modes are performed under the assumption that the underlying block cipher is ideal. This assumption is not satisfied if the cipher is used to build hash functions. Note that the ideal cipher is related to the concept of pseudo-random permutation, where the adversary does not know the cryptographic key. Clearly, for hash function constructions based on block ciphers, the adversary fully controls the key.

Biham and Shamir introduced differential analysis in [3] and successfully analyzed DES. The idea is to follow the propagation of a difference in the state of the cipher throughout consecutive rounds. When the input-output differences can be predicted with a sufficiently high probability, then the cipher can be distinguished from a pseudo-random permutation. This concept can trivially be

[*] The work was done while this author was visiting Macquarie University.

adjusted for the case, where the adversary knows/controls the key of the cipher (open-key differential distinguishers). The goal of adversary in this case would be to find an input-output pair of differences for the cipher that can be predicted with a probability higher than in a random permutation.

Unlike in the secret-key model, where the complexity of an attack is usually bounded by the size of the key space (i.e. 2^k for a k-bit key), the attacks in the open-key model are bounded by the size of the state space (i.e. 2^n for an n-bit state). Therefore, some of the published attacks in the secret-key model (precisely, the attacks with a complexity higher than 2^n) become worse than simple generic attacks, when applied in the open-key model.

Our Contributions. We investigate the impact of block cipher open-key differential distinguishers on hash function modes of operation. Our main contributions can be summarized as follows:

1. For a variety of hash function modes based on block-ciphers, we determine which collision finding attack variants (collisions, pseudo collisions, semi-free start collisions, or free start collisions) are feasible, assuming that the adversary is given a specific differential trail for the underlying block cipher in the open-key model. We target all Preneel-Govaerts-Vandewalle (PGV) single-block-length compression modes, as well as four double-block-length modes.
2. We examine several well known block ciphers (Crypton, Hierocrypt-3, SAFER++, Square, and generic Feistel ciphers) and for each of them, we present new known-key and chosen-key differential distinguishers - see Table 1. Our distinguishers use the rebound attack [25] as a starting point, but we obtain substantial improvements in the number of attacked rounds by exploiting some cipher-specific properties that allow us to manipulate bits of the subkeys (a similar technique was used in the context of analysing the Whirlpool function [21]). In the chosen-key model, for substitution-permutation (SP) ciphers, we obtain an explicit formula for the number of additional rounds that can be attacked for free, when the cipher has an invertible key schedule.

Table 1. Summary of attacks on the ciphers examined in the paper. The "Encryptions" column gives the expected number of encriptions in the case of a SP cipher, while the "Lower bound" column – the expected number of encryptions required in the case of a random permutation. In case of n-bit Feistel cipher r is a number of covered rounds, and 2^c is the complexity of some differential attack.

Cipher	Distinguisher	Rounds	Encryptions	Lower bound	Reference
Crypton	Known-key	7	2^{48}	2^{61}	Section 5.1
	Chosen-key	9	2^{48}	2^{61}	Section 5.1
Hierocrypt-3	Known-key	3.5	2^{48}	2^{61}	Section 5.1
	Chosen-key	4.5	2^{48}	2^{61}	Section 5.1
SAFER++	Known-key	6.5	2^{120}	2^{128}	Section 5.2
	Chosen-key	6.5	2^{112}	2^{128}	Section 5.2
Square	Known-key	7	2^{48}	2^{61}	Section 5.1
	Chosen-key	8	2^{48}	2^{61}	Section 5.1
n-bit Feistel	Differential attack	r	2^c		
with k-bit key	Known-key	$r+2$	2^c		Section 5.3
	Chosen-key	$r+\lfloor \frac{2k}{n} \rfloor$	2^c		Section 5.3

3. To show the efficiency of our distinguishers, we give a proof of a lower bound on the complexity of differential distinguishers in the case of a black-box random permutation. Although this bound has been used for a while (mainly as an upper bound, e.g. in [13] it is called a limited-birthday distinguisher), as far as we know, it has never been formally proved.

Organization. The paper is organized as follows. In Section 2 we define the open-key distinguishers and review techniques for constructing differential trails. In Section 3, we present our findings about the impact of block cipher differential trails on the security of hash function modes. Section 4 contains our lower bound on the complexity of differential distinguishers for black-box random permutations. In Section 5, we present our cipher specific known-key and chosen-key differential distinguishers for various block ciphers. Section 6 concludes the paper.

2 Preliminaries

2.1 Open-key Distinguishers for Block Ciphers

A distinguisher is one of the weakest cryptographic attacks that can be launched against a secret-key cipher. In this attack, there are two oracles: one that simulates the cipher for which the cryptographic key has been chosen at random and the other simulates a truly random permutation. The adversary can query both oracles and their task is to decide which oracle is the cipher (or random permutation). The attack is considered to be successful if the number of queries required to make a correct decision is below a well defined level.

The idea of open-key distinguishers was introduced by Knudsen and Rijmen in [18] for analysis of AES and a class of Feistel ciphers. They examined the security of these block ciphers in *a model where the adversary knows the key*. Later, the same approach was used in the attack on 8-round reduced AES-128 [13] and for analysis of Rijndael with large blocks [26], where the authors defined a new security notion for a known-key cipher. The idea of chosen-key distinguishers was introduced in the attack on the full-round AES-256 [5]. This time *the adversary* is assumed to have *a full control over the key*. A chosen-key attack was launched on 8-round reduced AES-128 in [6].

Both the known-key and chosen-key distinguishers are collectively known *open-key distinguishers*. The adversary has the knowledge of the key or even can choose a value of the key. To succeed, the adversary has to discover some property of the attacked cipher that holds with a probability higher than for a random permutation.

Differential distinguishers in the open-key model are defined in similar way as in the secret-key model. The adversary builds a differential trail $(\Delta_P, \Delta_K) \rightarrow \Delta_2$ for the block cipher $E_K(P)$. In other words, he finds a pair[4] of plaintexts (P_1,P_2) and a pair of keys (K_1, K_2), together known as a differential pair, such that $P_1 \oplus P_2 = \Delta_P, K_1 \oplus K_2 = \Delta_K$ and $E_{K_1}(P_1) \oplus E_{K_2}(P_2) = \Delta_2$. The pair (Δ_P, Δ_K) is the input difference, while Δ_2 is the output difference. At least one of Δ_P and Δ_K has to be non-zero. For example, the trails given in [6, 13, 26] have differences only in the plaintext, while the trail from [5] has differences in both the key and the plaintext.

2.2 Design of Differential Distinguishers for Block Ciphers

We will focus our analysis on substitution-permutation (SP) block ciphers. Each round of such ciphers consists of two types of transformations: 1) a non-linear layer of S-boxes, and 2) a linear-diffusion layer (LD). The non-linear layer operates on bytes, i.e. the inputs to the S-boxes are bytes of the state. The linear-diffusion layer may apply different transformations such as multiplications of the columns/rows of the state matrix by a fixed diffusion matrix, transpositions of rows/columns, rotations of elements of the state matrix, subkey additions, and others.

Differential trails for ciphers are given as a sequence of input-output word differences of each transformation of the state. Since SP ciphers are usually byte-oriented, these trails can be given as a sequence of active bytes, i.e. bytes that have differences. Depending on the properties of the S-box layer and the linear-diffusion layer, the adversary can built two types of trails.

The first type is a *standard* differential trail, where the exact values of the input-output differences for each layer and for each round of the trail are fixed. The probability of these trails depends on the differential properties of the S-boxes, i.e. the probability that a given input difference to the S-box

[4] Actually the adversary can build many pairs of plaintexts and keys.

will produce a given output difference. Note that when these differences are fixed, then the trail in the linear-diffusion layer holds with probability 1.

The second type is a *truncated* differential trail [16]. In this trail only the position of the active bytes is important, while the actual difference values are ignored. Since, the S-box operates on a single byte, it means it cannot change an active byte to a non-active and vice-versa. Hence the adversary concentrates only on the linear-diffusion layer and finds the probability of a particular configuration of input-output active bytes.

Although for SP ciphers the truncated differential approach is common, further in our analysis we will use both types of differential trails, together with trails with a difference in the plaintext only.

2.3 Techniques for Differential Trail Constructions

A major improvement in the analysis of SP cryptographic algorithms was the introduction of the rebound attack [25]. The idea is as follows. If we assume that the adversary controls the input to the S-boxes, then any input-output difference [5] to this layer can be obtained for free (simple table lookups). In other words, when $\varDelta_1, \varDelta_2$ are fixed, then it is easy to find x such that $S(x + \varDelta_1) \oplus S(x) = \varDelta_2$. In two consecutive middle rounds the adversary first fixes both the input differences of the LD layer in the first round, and the output differences of the LD layer of the second round. Then he goes forward through the first LD layer and backwards through the second LD layer. He ends up with fully determined differences, since the layers are linear. In between there is only one S-box layer (composed of a number of S-boxes), which can be passed for free when the adversary fixes the values, i.e. when he finds the proper solutions x of the above equation. Therefore, at the beginning of the first, and at the end of the second middle round, not only the differences, but now also the values have been fixed. The rounds that precede and follow the two middle rounds are passed probabilistically.

The technique of the rebound attack was improved with the Super-Sbox cryptanalysis [11, 13, 21]. When the round diffusion is incomplete then two layers of S-boxes can be passed for free using a precomputed lookup tables. The idea is similar to the one of the original rebound attack, but bigger lookup tables are used.

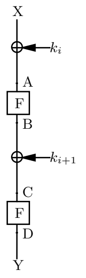


Fig. 1. Chosen-key distinguisher for SP ciphers.

The key can be used to gain an additional degree of freedom, which in return can lead to more S-box layers passed for free. When the adversary controls the key, then the rebound attack can be extended to one or two additional rounds, depending on the size of the key. The subkey (roundkey) is xored in each round of the cipher. The first S-box layer can be passed for free using the previous rebound technique (by fixing not only the difference, but the exact values as well). The second S-box layer can be passed for free as well if the adversary controls the input values to this layer by solving the appropriate equations. These values can be manipulated with the subkey, i.e. the adversary can choose a proper subkey such that the inputs to the S-box layer can be of arbitrary value (yet, their difference is fixed). Hence, the adversary can pass the second S-box layer for free if he controls the subkey of this round. Let us explain the idea on an example (See Fig.1). Let $\varDelta_1 \rightarrow \varDelta_2 \rightarrow \varDelta_3$ be an arbitrary two-round differential trail. First the adversary finds (with the rebound attack) a pair of states that satisfies the differential trail of the first round, i.e. he finds a pair $(A, A \oplus \varDelta_1)$ that produces $(B, B \oplus \varDelta_2)$ on the output. Then independently, he finds a pair of states for the second round, i.e. he finds $(C, C \oplus \varDelta_2)$ that produces the output $(D, D \oplus \varDelta_3)$. In the last step he has to fix a proper subkey k_{i+1} for the second round, which will connect the output of the first round and the input of the second round. To do so, the adversary fixes

[5] Only half of the input-output differences are possible, but for each of them there are two different input values, hence on average it is true.

$k_{i+1} = B \oplus C$, and as the result he obtains a pair of states $(A \oplus k_i, A \oplus k_i \oplus \Delta_1)$ that satisfy the two round differential trail.

Similarly, the adversary can pass more S-box layers when he controls the subkeys of these layers. An obvious requirement for the subkeys of these additional rounds is that they need to be independent. Otherwise, a change in a subkey in one round will change the value of a subkey in another round, which might lead to incorrect input values for the S-box layer of this second round. A second requirement is an invertible key schedule. Since the adversary controls the values of the subkeys of some middle rounds, he has to be able to produce the values of the subkeys of the rounds that precede and follow these rounds, hence he has to find the master key from the fixed subkeys. It is important to notice that this technique requires a negligible memory.

2.4 Building the Differential Trails

For each of the techniques discussed above, the adversary first builds a trail that may have a plenty of active S-boxes in some middle rounds and a few at the ends of the trail. Then, a pair of values that follows the differential trail only in these middle rounds is found with complexity 1. The rest of the rounds, before and after the middle rounds, are covered probabilistically since the adversary has no degree of freedom left.

Finding the optimal differential trails with no difference in the key can be done automatically since the ciphers considered in this paper are byte-oriented with a block size of 16 bytes. This leads to a search space of 2^{16} possible starting values.

Some of the ciphers are based on the so-called wide trail strategy [10], and provide an efficient method for estimating the probability of the best round-reduced standard differential trails. These estimation are based on the differential properties of the S-boxes and the diffusion properties of the LD layers, which are often maximum distance separable mappings.

3 Impact of Block Cipher Known Key Differential Trails on Hash Modes

The most popular design of cryptographic hash is based on iterative use of a compression function. This construction is also known as the Merkle-Damgård (MD) structure. Early compression functions were using block ciphers as the main building block. Assume that we have a single instance of a block cipher $E_K(P)$ and wish to design a compression function that takes a $2n$-bit input (H, M) and outputs a n-bit string $F(H, M)$. This problem has been investigated in [7, 27] and it has been shown that there are 12 structures (modes) that are secure. An example of one such structure is the well-known Davies-Meyer (DM) mode that is defined as $F(H, M) = E_M(H) \oplus H$ (see mode 5 in Table 2), where H and M are the chaining value and the message, respectively.

In this work, we consider four types of collision attacks against the compression functions:

1. Collisions - for a fixed chaining value H_0, the adversary tries to find two distinct messages M_1, M_2 such that $F(H_0, M_1) = F(H_0, M_2)$.
2. Pseudo collisions - for a message M, the adversary wishes to find two distinct chaining values H_1, H_2 such that $F(H_1, M) = F(H_2, M)$.
3. Semi-free start collisions - the adversary attempts to find two distinct messages M_1, M_2 and a chaining value H such that $F(H, M_1) = F(H, M_2)$.
4. Free start collisions - the adversary tries to find two distinct chaining values H_1, H_2, and two distinct messages M_1, M_2 such that $F(H_1, M_1) = F(H_2, M_2)$.

We investigate the resistance of compression functions based on block ciphers against the attacks described above. We assume that the adversary can build a differential trail for the cipher with differences not only in the plaintext, or in the key, but also in both the plaintext and the key. For example, for the DM compression function, this means the adversary can find a pair of chaining values (H_1, H_2) and a pair of messages (M_1, M_2) (possibly in one of the pairs the two values are equal) such that $H_1 \oplus H_2 = \Delta_H, M_1 \oplus M_2 = \Delta_M$ and $F(H_1, M_1) \oplus F(H_2, M_2) = \Delta_H \oplus \Delta_2$. Hence, when the adversary can build *some* trail, i.e. when he cannot control the exact values of the differences Δ_H, Δ_2, then he can find a differential distinguisher for the DM compression function. On the other hand, when the adversary can build a *specific* trail for the cipher with a difference in the plaintext (H is the plaintext input to the cipher), such that $\Delta_H \oplus \Delta_2 = 0$, then he can find: 1) free-start collisions, if $\Delta_M, \Delta_H \neq 0$, 2) pseudo-collisions, if $\Delta_M = 0, \Delta_H \neq 0$, 3) collisions or semi-free start collisions, if $\Delta_M \neq 0, \Delta_H = \Delta_2 = 0$ (note that this implies that there are key collisions in the cipher since in DM, the message is the key).

The same approach can be applied to the other 11 modes. We try to find the all possible collision attacks under the assumption that the adversary can control the relation between the input and the output differences of a trail in the cipher. Our findings are presented in Table 2.

Table 2. The first column consists of numbers from [7]. The entries in the columns plaintext, key, plaintext and key show the best collision attacks for the modes when there is difference only in the plaintext, only in the key or both in the plaintext and key, respectively. The abbreviations C, PC, SFSC, FSC stand for collision, pseudo-collision, semi-free start collision, free start collision, respectively.

mode $(\imath)$	h'	plaintext	key	plaintext and key
1	$E_h(m) \oplus m$	C, SFSC	PC[a]	FSC
2	$E_h(h \oplus m) \oplus h \oplus m$	C, SFSC	PC	PC, FSC
3	$E_h(m) \oplus h \oplus m$	C, SFSC	PC	FSC
4	$E_h(h \oplus m) \oplus m$	C, SFSC	PC	PC, FSC
5	$E_m(h) \oplus h$	PC	C[a], SFSC[a]	FSC
6	$E_m(h \oplus m) \oplus h \oplus m$	PC	FSC	C, SFSC, FSC
7	$E_m(h) \oplus h \oplus m$	PC	C, SFSC	FSC
8	$E_m(h \oplus m) \oplus h$	PC	FSC	C, SFSC, FSC
9	$E_{h \oplus m}(m) \oplus m$	FSC	PC[a]	C, SFSC, FSC
10	$E_{h \oplus m}(h) \oplus h$	FSC	C[a], SFSC[a]	PC, FSC
11	$E_{h \oplus m}(m) \oplus h$	FSC	PC	C, SFSC, FSC
12	$E_{h \oplus m}(h) \oplus m$	FSC	C, SFSC	C, PC, FSC

[a] When key collisions exist in the cipher.

Often the block size of a cipher is too small to be secure in the compression mode. Hence, there is a class of compression functions, also called double-block-length ones, whose output size is two times bigger than the block size of the underlying cipher. We investigate the security of such functions proposed by Lai-Massey in [20], Hirose in [14] and Bracht et al. in [8]. Our results are presented in Table 3.

Although we have analyzed the collision resistance of the above modes, the differential trails for the underlying ciphers in the open-key model can be used as a standalone cryptanalytical result for the compression functions.

Table 3. In the first column A-DM, T-DM, DBL and MDC-2 are abbreviations of Abrest DM, Tandem DM, Double-Block-Length and Modification Detection Code 2 respectively (see [20] for the first two, [14] for the third and [8] for the last). The abbreviations C, PC, SFSC, FSC stand for collision, pseudo-collision, semi-free start collision, free start collision, respectively.

mode	(h', g')	plaintext	key	plaintext and key
A-DM	$h' = E_{g,m}(h) \oplus h$ $g' = E_{m,h}(\bar{g}) \oplus g$	FSC	C, SFSC	PC, FSC
T-DM	$h' = E_{g,m}(h) \oplus h$ $g' = E_{m,E_{g,m}(h)}(g) \oplus g$	FSC	C, SFSC	PC, FSC
DBL	$h' = E_{h \Vert m}(g \oplus c) \oplus g \oplus c$ $g' = E_{h \Vert m}(g) \oplus g$	PC	C, PC, SFSC, FSC	PC, FSC
MDC-2	$h' = (E_h(m) \oplus m)^L \parallel (E_g(m) \oplus m)^R$ $g' = (E_g(m) \oplus m)^L \parallel (E_h(m) \oplus m)^R$	C, SFSC	PC[a]	FSC

[a] When key collisions exist in the cipher.

4 Lower Bound on Complexity of Differential Distinguisher for Random Permutations

In this section we present a lower bound on the complexity of differential distinguishers for a black-box random permutation. This allows us to fairly compare our cipher-specific distinguisher complexities in Section 2.2 to the best possible black-box distinguisher. Although a similar *upper* bound has been used before (see, e.g. [13]), our result proves that it is indeed close to the best possible. To our knowledge, such a lower bound has not been published before, and may be of independent interest.

When the key is fixed, a block cipher becomes a permutation. An open-key differential distinguisher with no difference in the key is valid if the complexity of finding a differential pair is less than the complexity of finding such pair in a random permutation. When the input and output differences are fully fixed, in n-bit random permutation the complexity of finding a differential pair is 2^n, hence any differential distinguisher with a probability higher than 2^{-n} is valid. When the input difference is fixed, and the output difference can take values from a set of the cardinality 2^c, then for a random permutation, a differential pair can be found after performing 2^{n-c} encryptions. The general case when both the input and the output differences are taken from sets of fixed cardinalities, is discussed in the following lemma.

Lemma 1. *Let D_I, D_O denote subsets of $\{0,1\}^n$, which are closed under $\oplus$, i.e. $x \oplus y \in D_I$ (respectively D_O) for $x, y \in D_I$ (resp. D_O). For any attacker making queries to a random n-bit permutation π and its inverse π^{-1}, the complexity (measured in expected number of oracle queries) of finding a pair of inputs (x, y), where $x \oplus y \in D_I, |D_I| = 2^{c_I}$, such that $\pi(x) \oplus \pi(y) \in D_O, |D_O| = 2^{c_O}$, is lower bounded as $Q \geq \min(2^{\frac{n}{2}-2}, 2^{n-(c_I+c_O)-3})$.*

Proof. Since D_I and D_O are closed under $\oplus$, we may partition $\{0,1\}^n$ into input sets $A_1, \dots, A_N$, where each $|A_i| = |D_I| = 2^{c_I}$, $N = \frac{2^n}{|D_I|} = 2^{n-c_I}$, such that $x \oplus y \in D_I$ for $x, y \in A_i$ for $i = 1, \dots, N$. Similarly, we have a partition into output sets $B_1, \dots, B_M$ where $|B_j| = |D_O| = 2^{c_O}$, $M = \frac{2^n}{|D_O|} = 2^{n-c_O}$ for all $j = 1, \dots, M$.

Let us define the following game G_0: attacker $\mathcal{A}$ has an access to a random permutation oracle $\pi \colon \{0,1\}^n \to \{0,1\}^n$ and its inverse π^{-1}, making a total of q queries to these oracles.

In the following games G_k ($k = 0, 1, 2$), let E_k be the following event: $\mathcal{A}$ finds $x \neq y$ with $x, y \in A_i$ and $\pi(x), \pi(y) \in B_j$ for some i, j while interacting with game G_k.

We show below the following upper bound:

$$\Pr(E_0) \leq \frac{q^2}{2^n} + \frac{q}{2^{n-(c_O+c_I)}}. \tag{1}$$

Before we explain the formal proof, we remark that the intuition for this result is as follows. The first term $\frac{q^2}{2^n}$ is the upper bound on the collision probability error due to the fact that we simplify the problem by replacing the random permutation π with a random function. The last term arise because at each query to π (resp. π^{-1}) which is in some input set A_i (resp. output set B_j) there are at most 2^{c_I} points in A_i whose image under π is already defined (resp. at most 2^{c_O} points in B_j whose image under π^{-1} is already defined), thus occupying at most 2^{c_I} out of the 2^{n-c_O} output sets (resp. at most 2^{c_O} out of the 2^{n-c_I} input sets).

We first show that (1) implies the claimed expected complexity bound. In game G_0, let T denote the random variable defined as the number of oracle queries until the event E_0 occurs. We lower bound the expected value $Q = \mathrm{E}(T)$ as follows. Let $p(q)$ denote the right hand side of (1), and let q^* be such that $p(q^*) = \frac{1}{2}$. Since $\Pr(T \leq q) \leq p(q)$, we have

$$Q \geq \sum_{q>q^*} \Pr(T = q) \cdot q \geq q^* \cdot \Pr(T > q^*) \geq \frac{q^*}{2}. \tag{2}$$

Now, for $i \in \{1, 2\}$, let q_i denote the value of q such that the ith term on the right hand side of (1) is equal to $\frac{1}{4}$. Since there are 2 terms in (1), we may take $\min(q_1, q_2)$ as lower bound for q^*. Since $q_1 = 2^{\frac{n}{2}-1}$ and $q_2 = 2^{n-(c_I+c_O)-2}$, the claimed lower bound on Q follows.

It remains to prove (1). We will do this by building a chain of games, starting with G_0, which are similar until *bad* is set (for further details of this methodology see for example [2]).

First define a game G_1 to be similar to G_0 except that the permutation π is replaced by a relation $P \subset \{0,1\}^n \times \{0,1\}^n$ that is injective and functional, but not necessary defined in whole domain. According to naming convention in [2] relation P is called partial permutation, whereas injectivity and functional conditions together are named "permutation constraint". Initially P is empty and through execution of G_1 its values are being sampled randomly with respect to "permutation constraint". Whenever $P(x)$ (resp. $P^{-1}(y)$) is needed first it is checked if P (resp. P^{-1}) is defined on x (resp. y). If this is the case then appropriate value is returned, otherwise $P(x)$ (resp. $P^{-1}(y)$) is sampled uniformly at random from $\overline{img(P)}$ (resp. $\overline{img(P^{-1})}$), where $\overline{img(P)}$ is complement of image of P. Because the sampling is the same as in Game G_0, we have

$$\Pr(E_0) = \Pr(E_1). \tag{3}$$

Next we define game G_2 which is the same as G_1 except "permutation constraint" for P does not need to be fulfilled. That means the values $P(x)$ (resp. $P^{-1}(y)$) are sampled at random from $\{0,1\}^n$, but the game stops immediately when the "permutation constraint" is not satisfied. Unless the "permutation constraint" is violated by the occurrence of a collision between a new output value returned by P and a previous output value of P or input value queried to P^{-1} (resp. a collision between a new output value returned by P^{-1} and an previous output value of P^{-1} or input value queried to P), the games G_1 and G_2 proceed identically. Since at each query there are at most q previous P (resp. P^{-1}) output values already defined, we have

$$|\Pr(E_2) - \Pr(E_1)| \leq \frac{q^2}{2^n}. \tag{4}$$

At this stage we stop building chain of games and we upper bound the probability $\Pr(E_2)$ directly. We claim that

$$\Pr(E_2) \leq \frac{q}{2^{n-(c_O+c_I)}}. \tag{5}$$

Let x denote the qth query of the attacker, define the following variables for $i = 1, \ldots, N$ and $j = 1, \ldots, M$:

- a_i^F = number of P oracle queries made so far which are in A_i,
- a_i^R = number of P^{-1} oracle answers given so far which fell in A_i,
- b_j^F = number of P^{-1} oracle queries made so far which are in B_j,
- b_j^R = number of P oracle answers given so far which fell in B_j.

Suppose that x is a query to P and that $x \in A_i$ for some i. We have so far $a_i^F + a_i^R$ points in A_i whose B_j sets are already defined. Hence the event E_2 will occur only if the uniformly random (in $\{0,1\}^n$) answer of P falls in one of those output sets, so it will happen in this query with probability $\leq \frac{a_i^F + a_i^R}{M} \leq \frac{|D_I|}{M} = \frac{1}{2^{n-(c_I+c_O)}}$, using $a_i^F + a_i^R \leq |D_I|$ (since the game did not stop so far). Similarly, if x is a query to P^{-1} and $x \in B_j$ for some j, then E_2 will occur in this query with probability $\leq \frac{b_j^F + b_j^R}{N} \leq \frac{|D_O|}{N} = \frac{1}{2^{n-(c_I+c_O)}}$. It follows that E_2 occurs among the first q queries with probability bounded by (5), as claimed. This completes the proof of the Lemma. ■

5 Differential Trails for Specific Block Ciphers

We have searched for differential trails in the following ciphers: Crypton, Hierocrypt-3, SAFER++, and Square. Specifically, we have tried to build standard and/or truncated trails, which can be used in a rebound-type attack. For some of the ciphers, the probabilities for the both standard and truncated differential trails were higher than in a random permutation. In this case, only the trails (which are usually truncated) with higher probability are presented.

The trails for the chosen-key distinguishers were built upon the trails for the known-key distinguishers by increasing the number of the full active middle rounds which can be covered for free when a proper subkey is fixed. When n-bit key is used, with an invertible key schedule that produces s-bit subkeys, then the chosen-key distinguisher has $\lfloor \frac{n}{s} \rfloor$ more rounds then the known-key distinguisher.

Due to space limitation, we will not give a full description of the attacked ciphers, but rather, introduce them briefly using the original notions and definitions from the source papers.

5.1 Crypton, Hierocrypt-3 and Square

Crypton [22], Hierocrypt-3 [28], and Square [9] are 128-bit SP block ciphers and have a various number of internal rounds depending on the length of the key. The best published attacks in the secret-key model are on 8 rounds of Crypton [15], 3-3.5 rounds of Hierocrypt-3 [1], and 8 rounds of Square [19].

The internal state of each cipher can be seen as 4×4 matrix of bytes, while a round consists of three types of transformations of the state: 1) byte-wise application of a non-linear S-box, 2) matrix-wise linear-diffusion (LD) layer that applies different linear transformations of various bytes of the matrix to introduce a sufficient diffusion among the bytes of the state, 3) subkey addition – a simple xor of the round key to the matrix. A round of Crypton consists of an S-box layer γ, LD layer composed of two transforms π and τ, and subkey addition σ. Hierocrypt-3 has six round transforms: two S-box layers $[S]$, two LD layers $[MDS_L]$ and $[MDS_H]$, and two subkey additions $[AK]$. A round of Square consists of four transforms: S-box layer γ, LD layer with two transforms θ and π, and a subkey addition σ. It is important to notice that all three ciphers have a non-linear, but invertible, key schedule. The 256-bit key versions of Crypton and Hierocrypt-3, have a key schedule such that each two consecutive 128-bit subkeys are independent.

For each cipher, we can build 7-round truncated differential trails (7 S-box layer trail in case of Hierocript), that have a full active state in the middle round, but only a few active S-boxes in the rest of the 3+3 rounds (S-box layers of Hierocript). These trails can be used to construct known-key distinguishers on 7 rounds of the ciphers, based on the rebound technique. Since the ciphers have invertible key schedules, we can increase the number of attacked rounds by switching from the known-key to the chosen-key attacks and using the degrees of freedom of the subkeys. Hence, we can construct a chosen-key differential distinguisher on 8 rounds of Crypton with 128-bit keys, and 9 rounds of Crypton with 256-bit keys (the additional round comes from extra 128-bit freedom of the key; the chosen-key has $\lfloor \frac{256}{128} \rfloor = 2$ more rounds than the known-key, see Section 2.3). For Hierocrypt-3, the result is a chosen-key distinguisher on 8 S-box layers = 4 rounds for 128-bit keys, and on 9 S-box layers=4.5 rounds for 256-bit keys. Square only supports 128-bit keys, hence the chosen-key distinguisher works on 8 rounds, which is indeed the total number of rounds of this cipher.

The trails used in the chosen-key distinguishers for 9, 4.5 and 8 rounds of Crypton, Hierocrypt-3, and Square, respectively are given in the Appendix A. Since the middle full-active state round(s) are covered by the rebound attack and by fixing the subkeys used in these rounds, we can assume that the probability of the trails in these rounds is 1. Hence, we count only the probability of the rest of the rounds. In each of the three trails, we have twice 2^{-24} – that is the probability that the linear-diffusion transformation will turn four active bytes into one active byte. The probability of the trail in the rest of the layers is 1. Therefore, to find pairs of plaintexts and ciphertexts that will follow the truncated differential trails, one has to start with 2^{48} pairs of states that pass the middle rounds (each pair can be build with negligible complexity). Out of 2^{48} pairs, 2^{24} will produce four-to-one active byte in the first half of the trail, leading to a plaintext difference as the one in the trail. Out of these 2^{24}, one will produce four-to-one active in the second half of the trail and a ciphertext difference as the one in the trail. Now, let us try to compare our complexity of 2^{48} encryptions to the complexity in a case of a random permutation. By Lemma 1, to find this complexity we have to find the cardinalities of the plaintext and the ciphertext differences in the truncated trails. Although some of the plaintext/ciphertext differences in the trails have full active states, they are obtained by a linear transformation of some state with a four active bytes. Hence the cardinalities in all cases are $2^{4 \cdot 8} = 2^{32}$, and the complexity of producing a pair for a random permutation, that follows the trails, is at least $\min(2^{\frac{128}{2}-2}, 2^{128-(32+32)-3}) = 2^{61}$ encryptions.

To test the correctness of our results, we have constructed a chosen-key distinguisher on mCrypton [23], which has the same design as Crypton, but instead of bytes (8-bit words), it works with nibbles (4-bit words), and uses a non-invertible key schedule. The above distinguishers for Crypton can easily be applied to a modified version of mCrypton with a (invertible) key schedule identical to the one of Crypton. The chosen-key distinguisher for 9 rounds of this modified mCrypton was implemented on a PC, and a differential pair was found. The results are given in Appendix B.

5.2 SAFER++

SAFER++ [24] is a 128-bit SP block cipher. The version with 128-bit key has 7 rounds and the best published attack works for 5.5 rounds [4]. A round of SAFER++ consists of: 1) a byte-wise subkey addition, 2) a byte-wise S-box layer, 3) a byte-wise subkey addition, and 4) a state-wise linear-diffusion layer in the form of four 4-PHT. The subkey additions are modular and xor, and two different S-boxes are used. After the last round, there is an extra subkey addition. The key schedule is linear.

When the subkeys are fixed, then the S-box layer can be merged with the subkey additions to form another S-box layer, with the same input and output size. In other words, the subkey addition together with S-box and the subkey addition can be seen simply as some S-box (since the bytes of the subkeys are different, the S-boxes are also different). Hence, we can assume that a round of the cipher is composed of an S-box layer and a linear-diffusion layer, and all the additions in the cipher are modular.

Our automatic search for the best round-reduced standard differentials has found that there exist only two three-round trails with 10 active S-boxes (the rest of the trails have more than 10 active

S-boxes). The first trail has 4,2,4 while the second has 2,3, and 5 active S-boxes in the first, the second, and the third round, respectively. We have used two 4-2-4 trails in our standard differential

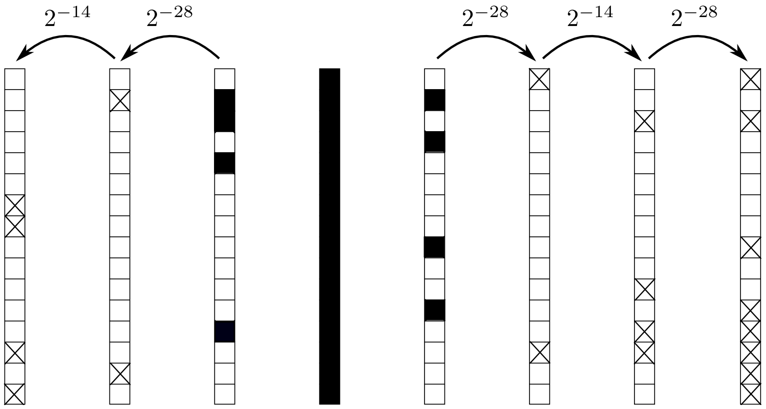


Fig. 2. Standard differential trail for 6.5 rounds of SAFER++ for the chosen-key distinguisher and 128-bit key. The first round is without the S-box layers, crossed square represents fixed 8-bit difference. The detailed trail is given in Fig.7 of the Appendix A.

attack (see Fig.2). We attack 6.5 rounds of SAFER++, which is the full cipher, except for the first round, where the three transforms: subkey addition, S-box and subkey addition, are missing. As far as we know this is the first rebound attack with standard differentials. Therefore, we will describe it in more details.

First, to cancel the effects of the last extra subkey addition, we fix the MSB of the bytes 1, 3, 9, 12, 13, 14, 15, 16 of the last subkey to zero, while the values for the other bits of the subkey are randomly chosen. Then, from the mentioned subkey, we find the value of the master key, and the values for all remaining subkeys. Now we are ready to start the rebound attack.

We assign differences to the bytes 2, 3, 5, 13 (and no difference to other bytes) of the state before the linear layer in round 3. The differences should be such that after the linear layer all bytes are active (this holds for almost any assigned values). Similarly, we assign differences to the bytes 2, 4, 9, 12 of the state after the linear layer in round 4, go backwards through the linear layer and obtain a full active state. In between the top and the bottom active states, there is only the S-box layer, hence we match the differences through this layer, i.e. we fix the values of the bytes such that all the input differences produce all the output differences. Since the values of the full state have been fix, the rest of the rounds are passed probabilistically. There are 2, 4, 4, 2, 4 active S-boxes (16 in total) in the rounds 2, 3, 5, 6, 7, respectively.

If we assume that the differential propagation through all of the S-boxes occurs with the probability 2^{-7} then the complexity of the whole attack is $2^{7 \cdot 16} = 2^{112}$ encryptions. Note that for a fixed key, we have 2^{64} starting values for the rebound attack. We can choose different keys (such that the last subkey has the MSB of the mentioned above bytes fixed to zero) to get the necessary number of starting pairs for the differential attack. Since the input and output differences of the differential pair are fully fixed, such a pair in a random permutation can be found with 2^{128} encryptions.

5.3 Feistel Ciphers

Feistel ciphers with a SP round function can have a number of rounds covered for free in the known and chosen-key differential attacks. When the key is known, the S-box layers of two consecutive rounds can be attacked independently since the round function uses only half of the input. For a given two-round differential, first a pair of input states that satisfy the differential of the first round function is fixed, and then a pair of states of the second round function. Therefore, in an known-key attack, any differential trail can be extended by two additional rounds (this should not be confused with the distinguishers on 7-round Feistel ciphers proposed in [18]).

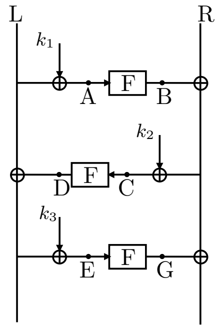


Fig. 3. Chosen-key distinguisher for 3-round Feistel ciphers.

Assume that the adversary can control the key in a Feistel cipher. As the size of the input to the round function and the size of the round key are (usually) half as big as in the SP ciphers, the number of rounds that can be attacked for free is twice as big as for the SP ciphers. Let us examine the possibility of obtaining a pair of states for a three-round differential. Let n-bit Feistel cipher has an invertible key schedule that generates $\frac{n}{2}$-bit subkeys. To find a pair of states that follows some three-round differential:

$$(\Delta_1^L, \Delta_1^R) \to (\Delta_2^L, \Delta_2^R) \to (\Delta_3^L, \Delta_3^R) \to (\Delta_4^L, \Delta_4^R)$$

(the pair of states is $(L, R), (L \oplus \Delta_1^L, R \oplus \Delta_1^R)$), the adversary builds, as in the rebound attack, three pairs of states, separately for each round, that satisfy the one-round differentials, i.e. he finds the values A, C, E, such that

$$\begin{aligned} F(A) \oplus F(A \oplus \Delta_1^L) &= \Delta_1^R \oplus \Delta_2^L, \\ F(C) \oplus F(C \oplus \Delta_2^L) &= \Delta_2^R \oplus \Delta_3^L, \\ F(E) \oplus F(E \oplus \Delta_3^L) &= \Delta_3^R \oplus \Delta_4^L. \end{aligned}$$

Let $F(A) = B, F(C) = D, F(E) = G$. Then, in order to connect these three one-round differentials, the following conditions for the subkeys k_1, k_2, k_3 apply:

$$\begin{aligned} L \oplus k_1 &= A, \\ R \oplus B \oplus k_2 &= C, \\ L \oplus D \oplus k_3 &= E. \end{aligned}$$

From the first and the third equation, we get the relation $k_1 \oplus k_3 = A \oplus D \oplus E$ (note that the adversary does not control the values of A, D, E because they are fixed by the rebound attack). To satisfy this relation, the keys k_1, k_3 have to be independent (or be linearly dependent – but this is not common for ciphers). Once this is satisfied, the solution (L, R, k_1, k_2, k_3) for the system can be found in linear time. Hence in general, the master key has to be at least $\frac{3n}{2}$-bit long.

A similar analysis applies to cases when a higher number of rounds has to be covered for free. The only difference is that the resulting system has more equations. When r rounds are fixed, the system has r equation and $r + 2$ unknowns: $L, R, k_1, \ldots, k_r$. In order to find the solution in linear time, for any invertible key schedule, the subkeys have to be independent. Hence, to attack an additional r rounds of a n-bit Feistel cipher the key has to be at least $\frac{rn}{2}$-bit long.

6 Conclusions

We have examined the application of the differential trails in analysis of ciphers that are used for compression function constructions. We have considered both the known-key and chosen-key models. Specifically, we have analyzed the collision resistance of all compression functions based on single block ciphers as well as the four known double-block compression functions, when specific differential trails for the underlying ciphers can be built. Furthermore, we have presented differential distinguishers for Crypton, Hierocrypt-3, SAFER++, and Square. For these ciphers, we have shown that when the attack model is switched from secret-key to open-key, the number of rounds that can be attacked increases. We have given as well a formal proof of lower bound of constructing pair that follow a truncated trail in the case of a random permutation. Our results are summarized in Table 1.

The area of open-key distinguishers is largely unexplored. Finding similar distinguishers based on related-key differentials remains an open problem.

Acknowledgement.

The authors would like to thank anonymous reviewers for their helpful comments.
Ivica Nikolić is supported by the Fonds National de la Recherche Luxembourg grant TR-PHD-BFR07-031. Josef Pieprzyk and Ron Steinfield are supported by Australian Research Council grant DP0987734. Przemysław Sokołowski is supported by cotutelle Macquarie University Research Excellence Scholarship (cotutelle MQRES) and partially supported by Ministry of Science and Higher Education grant N N206 2701 33, 2007-2010.

References

1. P. S. L. M. Barreto, V. Rijmen, J. N. Jr., B. Preneel, J. Vandewalle, and H. Y. Kim. Improved SQUARE Attacks against Reduced-Round HIEROCRYPT. In M. Matsui, editor, *FSE*, volume 2355 of *Lecture Notes in Computer Science*, pages 165–173. Springer, 2001.
2. M. Bellare and P. Rogaway. The Security of Triple Encryption and a Framework for Code-Based Game-Playing Proofs. In S. Vaudenay, editor, *EUROCRYPT*, volume 4004 of *Lecture Notes in Computer Science*, pages 409–426. Springer, 2006.
3. E. Biham and A. Shamir. Differential Cryptanalysis of DES-like Cryptosystems. In A. Menezes and S. A. Vanstone, editors, *CRYPTO*, volume 537 of *Lecture Notes in Computer Science*, pages 2–21. Springer, 1990.
4. A. Biryukov, C. D. Cannière, and G. Dellkrantz. Cryptanalysis of SAFER++. In D. Boneh, editor, *CRYPTO*, volume 2729 of *Lecture Notes in Computer Science*, pages 195–211. Springer, 2003.
5. A. Biryukov, D. Khovratovich, and I. Nikolić. Distinguisher and Related-Key Attack on the Full AES-256. In S. Halevi, editor, *CRYPTO*, volume 5677 of *Lecture Notes in Computer Science*, pages 231–249. Springer, 2009.
6. A. Biryukov and I. Nikolić. A New Security Analysis of AES-128. *CRYPTO 2009 rump session*, 2009.
7. J. Black, P. Rogaway, and T. Shrimpton. Black-Box Analysis of the Block-Cipher-Based Hash-Function Constructions from PGV. In M. Yung, editor, *CRYPTO*, volume 2442 of *Lecture Notes in Computer Science*, pages 320–335. Springer, 2002.
8. B. O. Brachtl, D. Coppersmith, M. M. Hyden, S. M. M. Jr, C. H. W. Meyer, J. Oseas, S. Pilpel, and M. Schilling. Data authentication using modification detection codes based on a public one way encryption function. US Patent no. 4,908,861. Assigned to IBM. Filed August 28, 1987, March 13, 1990.
9. J. Daemen, L. R. Knudsen, and V. Rijmen. The Block Cipher Square. In E. Biham, editor, *FSE*, volume 1267 of *Lecture Notes in Computer Science*, pages 149–165. Springer, 1997.
10. J. Daemen and V. Rijmen. The Wide Trail Design Strategy. In B. Honary, editor, *IMA Int. Conf.*, volume 2260 of *Lecture Notes in Computer Science*, pages 222–238. Springer, 2001.
11. J. Daemen and V. Rijmen. Understanding Two-Round Differentials in AES. In R. D. Prisco and M. Yung, editors, *SCN*, volume 4116 of *Lecture Notes in Computer Science*, pages 78–94. Springer, 2006.
12. E. Fleischmann, M. Gorski, and S. Lucks. On the Security of Tandem-DM. In O. Dunkelman, editor, *FSE*, volume 5665 of *Lecture Notes in Computer Science*, pages 84–103. Springer, 2009.
13. H. Gilbert and T. Peyrin. Super-Sbox Cryptanalysis: Improved Attacks for AES-like permutations. *Fast Software Encryption, 2010, to appear.*, 2009.
14. S. Hirose. Some Plausible Constructions of Double-Block-Length Hash Functions. In M. J. B. Robshaw, editor, *FSE*, volume 4047 of *Lecture Notes in Computer Science*, pages 210–225. Springer, 2006.
15. J. Kim, S. Hong, S. Lee, J. H. Song, and H. Yang. Truncated Differential Attacks on 8-Round CRYPTON. In J. I. Lim and D. H. Lee, editors, *ICISC*, volume 2971 of *Lecture Notes in Computer Science*, pages 446–456. Springer, 2003.

16. L. R. Knudsen. Truncated and Higher Order Differentials. In B. Preneel, editor, *FSE*, volume 1008 of *Lecture Notes in Computer Science*, pages 196–211. Springer, 1994.
17. L. R. Knudsen, F. Mendel, C. Rechberger, and S. S. Thomsen. Cryptanalysis of MDC-2. In A. Joux, editor, *EUROCRYPT*, volume 5479 of *Lecture Notes in Computer Science*, pages 106–120. Springer, 2009.
18. L. R. Knudsen and V. Rijmen. Known-Key Distinguishers for Some Block Ciphers. In K. Kurosawa, editor, *ASIACRYPT*, volume 4833 of *Lecture Notes in Computer Science*, pages 315–324. Springer, 2007.
19. B. Koo, Y. Yeom, and J. Song. Related-Key Boomerang Attack on Block Cipher SQUARE. Cryptology ePrint Archive, Report 2010/073, 2010. http://eprint.iacr.org/2010/073.pdf.
20. X. Lai and J. L. Massey. Hash Function Based on Block Ciphers. In R. A. Rueppel, editor, *EUROCRYPT*, volume 658 of *Lecture Notes in Computer Science*, pages 55–70. Springer, 1992.
21. M. Lamberger, F. Mendel, C. Rechberger, V. Rijmen, and M. Schläffer. Rebound Distinguishers: Results on the Full Whirlpool Compression Function. In M. Matsui, editor, *ASIACRYPT*, volume 5912 of *Lecture Notes in Computer Science*, pages 126–143. Springer, 2009.
22. C. H. Lim. A Revised Version of Crypton - Crypton V1.0. In L. R. Knudsen, editor, *FSE*, volume 1636 of *Lecture Notes in Computer Science*, pages 31–45. Springer, 1999.
23. C. H. Lim and T. Korkishko. mCrypton - A Lightweight Block Cipher for Security of Low-Cost RFID Tags and Sensors. In J. Song, T. Kwon, and M. Yung, editors, *WISA*, volume 3786 of *Lecture Notes in Computer Science*, pages 243–258. Springer, 2005.
24. J. Massey, G. Khachatrian, and M. Kuregian. Nomination of SAFER++ as Candidate Algorithm for the New European Schemes for Signatures, Integrity, and Encryption (NESSIE). *First Open NESSIE Workshop*, November,2000.
25. F. Mendel, C. Rechberger, M. Schläffer, and S. S. Thomsen. The Rebound Attack: Cryptanalysis of Reduced Whirlpool and Grøstl. In O. Dunkelman, editor, *FSE*, volume 5665 of *Lecture Notes in Computer Science*, pages 260–276. Springer, 2009.
26. M. Minier, R. C.-W. Phan, and B. Pousse. Distinguishers for Ciphers and Known Key Attack against Rijndael with Large Blocks. In B. Preneel, editor, *AFRICACRYPT*, volume 5580 of *Lecture Notes in Computer Science*, pages 60–76. Springer, 2009.
27. B. Preneel, R. Govaerts, and J. Vandewalle. Hash Functions Based on Block Ciphers: A Synthetic Approach. In D. R. Stinson, editor, *CRYPTO*, volume 773 of *Lecture Notes in Computer Science*, pages 368–378. Springer, 1993.
28. Toshiba Corporation. Specification of Hierocrypt-3. *submitted to the First Open NESSIE Workshop*, 13-14 November,Leuven, Belgium 2000.

A Differential Trails for Crypton, Hierocrypt-3, SAFER++ and Square

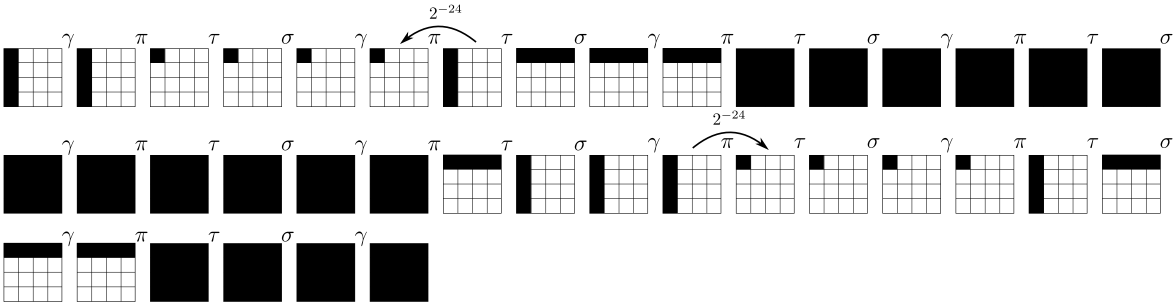

γ π τ σ γ π τ σ γ π τ σ γ π τ σ
2^{-24}
γ π τ σ γ π τ σ γ π τ σ γ π τ σ
2^{-24}
γ π τ σ γ

Fig. 4. Truncated differential trail for 9 rounds of Crypton for chosen-key distinguisher and 256-bit key.

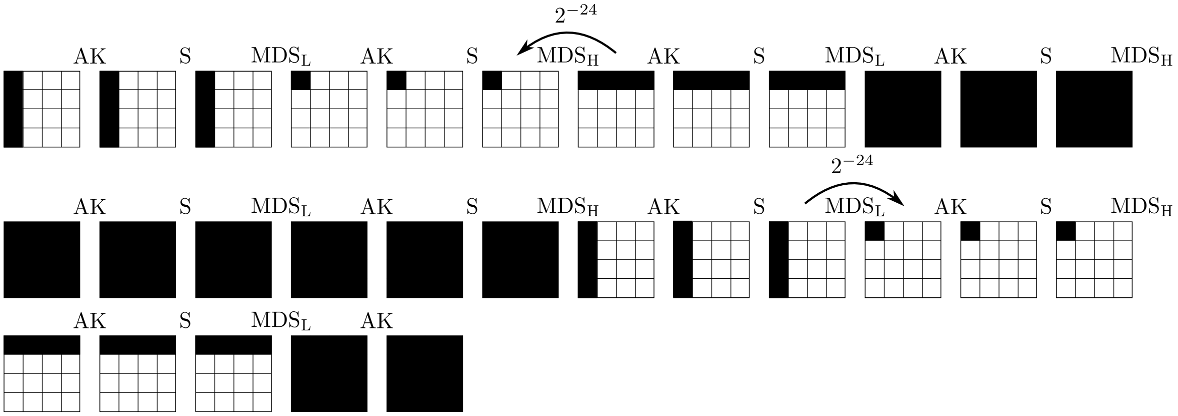


Fig. 5. Truncated differential trail for 4.5 rounds of Hierocrypt for chosen-key distinguisher and 256-bit key.

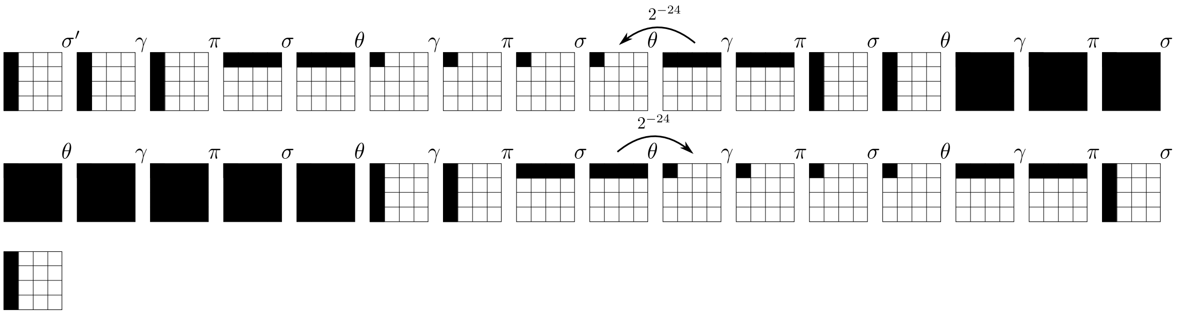


Fig. 6. Truncated differential trail for 8 rounds of Square for chosen-key distinguisher ($\sigma' = \sigma(\theta(k_0))$).

B Truncated Differential Trail for Modified mCrypton

The key scheduling in the test implementation of mCrypton has been adopted from Crypton and has following way:
Let K be a 128-bit encryption key and $K = k_0 \dots k_{31}$ where each k_i is four-bit nibble for $i = 0, \dots, 31$. At first two temporal values U and V are derived from K so that $U[i] = k_{8i}k_{8i+2}k_{8i+4}k_{8i+6}$ and $V[i] = k_{8i+1}k_{8i+3}k_{8i+5}k_{8i+7}$ for $i = 0, 1, 2, 3$. Next for $U' = \gamma(U)$ and $V' = \gamma(V)$ the eight expanded keys are evaluated as:

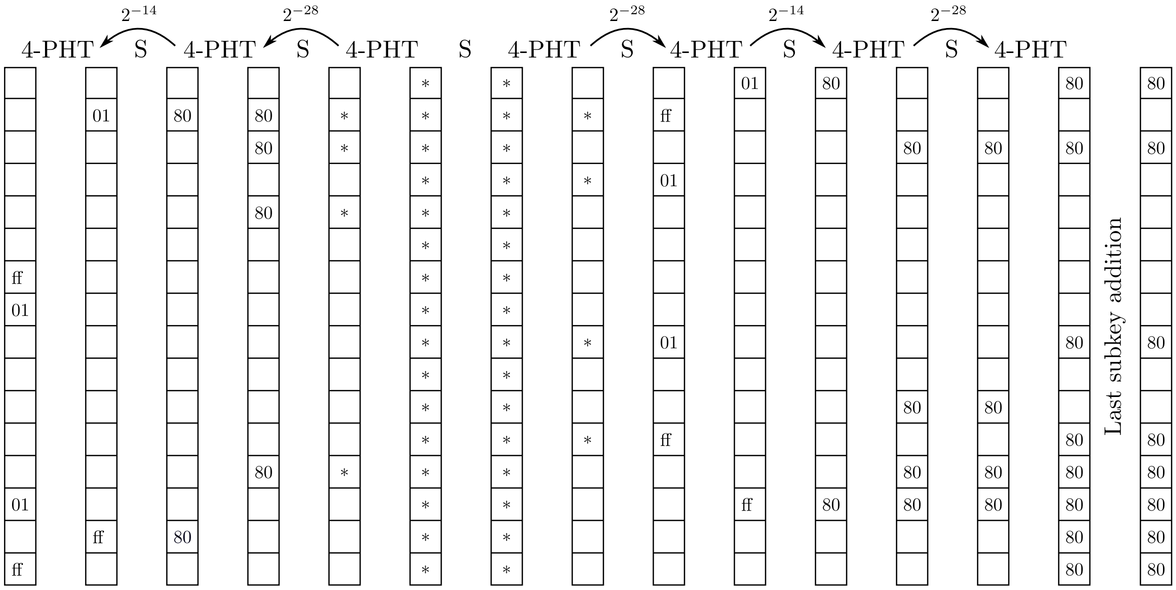
$$E[i] = \bigoplus_{j \neq i} U'[j] \qquad\qquad E[i+4] = \bigoplus_{j \neq i} V'[j]$$

for $i = 0, 1, 2, 3$ with use of which the 13 subkeys for each encryption round are generated according to the following procedure:

1. for the first and the second round:

$$K_1[i] = E[i] \oplus C[0] \oplus MC_i \qquad\qquad K_2 = E[i+4] \oplus C[1] \oplus MC_i$$

   for $i = 0, 1, 2, 3$,
2. for the remaining eleven rounds ($r = 2, \dots, 12$) two steps are executed alternatively:



					*	*			01	80			80	80
	01	80	80	*	*	*	*	ff						
			80	*	*	*					80	80	80	80
					*	*	*	01						
			80	*	*	*								
					*	*								
ff					*	*								
01					*	*								
					*	*	*	01					80	80
					*	*								
					*	*					80	80		
					*	*	*	ff					80	80
			80	*	*	*					80	80	80	80
01					*	*			ff	80	80	80	80	80
	ff	80			*	*							80	80
ff					*	*							80	80

Fig. 7. Standard differential trail for 6.5 rounds of SAFER++ for chosen-key distinguisher and 128-bit key.

(a) for r even:

$$\{E[0], E[1], E[2], E[3]\} \leftarrow \{E[1]^{\lll 12}, E[2]^{\lll 8}, E[3]^{\lll_b 3}, E[0]^{\lll_b 3}\},$$
$$K_r[i] = E[i] \oplus C[r] \oplus MC_i,$$

(b) for r odd:

$$\{E[4], E[5], E[6], E[7]\} \leftarrow \{E[7]^{\lll_b 1}, E[4]^{\lll_b 1}, E[3]^{\lll 4}, E[0]^{\lll 8}\},$$
$$K_r[i] = E[i+4] \oplus C[r] \oplus MC_i,$$

for $i = 0, 1, 2, 3,$

where $C[0] = \texttt{f53a}, C[k] = C[k-1] + \texttt{f372} \bmod 2^{16}$ for $k = 1, \ldots, 12$, $MC_0 = \texttt{acac}, MC_k = MC_{k-1}^{\lll_b 1}$ for $i = 0, 1, 2, 3$ and $\lll_b a$ represents bit-left-rotation by a bits within each four-bit nibble.

Example of a differential trail for mCrypton:

i	$A[i]$	$D[i]$	$D_{\gamma}[i]$	$D_{\pi\circ\gamma}[i]$	$D_{\tau\circ\pi\circ\gamma}[i]$	$K[i]$	i	$A[i]$	$D[i]$	$D_{\gamma}[i]$	$D_{\pi\circ\gamma}[i]$	$D_{\tau\circ\pi\circ\gamma}[i]$	$K[i]$
1	0 1 d 3	6 1 b 9	c 0 0 0	c 0 0 0	c 0 0 0	9 8 2 2	6	0 0 0 5	1 e c 5	e 8 2 7	e a 6 f	e 0 0 0	9 e 0 c
	7 7 0 1	e 7 1 f	c 0 0 0	0 0 0 0	0 0 0 0	e 6 1 5		0 0 6 6	f e 1 1	c a 6 e	0 0 0 0	a 0 0 0	8 a e a
	1 b 6 5	4 b b 3	8 0 0 0	0 0 0 0	0 0 0 0	a 7 a 2		0 0 0 0	d 6 4 3	a 2 6 d	0 0 0 0	6 0 0 0	b 0 4 6
	e a 3 b	c f 4 5	4 0 0 0	0 0 0 0	0 0 0 0	d d 3 4		7 2 3 0	a b d 2	6 a 4 b	0 0 0 0	f 0 0 0	b f b 1
2	1 1 a c	c 0 0 0	9 0 0 0	8 0 0 0	8 9 9 1	e 0 8 d	7	d e 7 f	e 0 0 0	e 0 0 0	e 0 0 0	e 0 0 0	3 e 4 0
	9 1 3 7	0 0 0 0	0 0 0 0	9 0 0 0	0 0 0 0	c a 4 2		b 7 3 7	a 0 0 0	c 0 0 0	0 0 0 0	0 0 0 0	1 7 8 d
	5 9 d 2	0 0 0 0	0 0 0 0	9 0 0 0	0 0 0 0	a 5 4 1		0 d 4 d	6 0 0 0	a 0 0 0	0 0 0 0	0 0 0 0	1 f 6 c
	e 7 1 4	0 0 0 0	0 0 0 0	1 0 0 0	0 0 0 0	2 b c 7		3 3 0 2	f 0 0 0	6 0 0 0	0 0 0 0	0 0 0 0	5 e 4 7
3	7 9 5 2	8 9 9 1	9 a f d	8 8 b 5	8 9 9 1	3 5 3 4	8	1 b 5 0	e 0 0 0	6 0 0 0	6 0 0 0	6 4 2 6	9 8 c 2
	c a 2 8	0 0 0 0	0 0 0 0	9 a 7 c	8 a 2 a	b 8 b 0		f 0 e c	0 0 0 0	0 0 0 0	4 0 0 0	0 0 0 0	d a b 9
	6 a 3 d	0 0 0 0	0 0 0 0	9 2 e d	b 7 e d	c 8 1 9		1 7 1 2	0 0 0 0	0 0 0 0	2 0 0 0	0 0 0 0	4 0 a 9
	6 8 a 0	0 0 0 0	0 0 0 0	1 a d 9	5 c d 9	e 2 9 a		9 2 4 7	0 0 0 0	0 0 0 0	6 0 0 0	0 0 0 0	5 0 a 7
4	6 c b 1	8 9 9 1	d 5 4 b	d 4 5 1	d 1 1 9	5 9 5 7	9	7 7 3 d	6 4 2 6	f 9 c c	e 9 8 4	e d b 7	3 3 3 4
	a d 6 f	8 a 2 a	5 e 9 3	1 e a d	4 e 1 a	c e d 2		e e e d	0 0 0 0	0 0 0 0	d 9 4 c	9 9 1 8	e 6 8 0
	f 9 b 8	b 7 e d	1 1 4 2	1 1 f c	5 a f b	f 3 6 0		0 0 2 a	0 0 0 0	0 0 0 0	b 1 c c	8 4 c c	6 b 9 2
	1 6 8 2	5 c d 9	d b 2 4	9 a b e	1 d c e	d b 2 4		b b c 0	0 0 0 0	0 0 0 0	7 8 c 8	4 c c 8	5 b 9 9
5	8 6 f 0	d 1 1 9	1 6 a 7	1 f d a	1 e c 5	3 d 5 0	10	8 5 d 0	e d b 7	7 2 9 9	0 0 0 0	0 0 0 0	1 3 c 3
	e 0 d 0	4 e 1 a	b c 4 8	e e 6 b	f e 1 1	3 8 e a		9 0 2 6	9 9 1 8	a 8 d d	0 0 0 0	0 0 0 0	8 2 0 9
	0 0 7 0	5 a f b	e 9 b 4	c 1 4 d	d 6 4 3	5 4 4 0		9 e 5 2	8 4 c c	1 c f e	0 0 0 0	0 0 0 0	8 8 3 a
	0 d 0 7	1 d c e	2 2 9 5	5 1 3 2	a b d 2	b f 1 7		5 e a 9	4 c c 8	6 2 7 1	0 0 0 0	0 0 0 0	8 c 1 e

Fig. 8. The columns in the table represent: i - round number, $A[i]$ - value of the state in round i, $D[i]$ - difference between two states in round i, $D_{\gamma}[i]$ - difference between two states after γ in round i, $D_{\pi\circ\gamma}[i]$ - difference between two states after $\pi \circ \gamma$ in round i, $D_{\tau\circ\pi\circ\gamma}[i]$ - difference between two states after $\tau \circ \pi \circ \gamma$ in round i, $K[i]$ - subkey in round i. The trail was obtained for $K =$ 679ff202d5834e529d9cf7013a4d8218.