Complementing Feistel Ciphers

Alex Biryukov[1] and Ivica Nikolić[2]

[1] University of Luxembourg
[2] Nanyang Technological University, Singapore
`alex.biryukov@uni.lu` `inikolic@ntu.edu.sg`

Abstract. In this paper, we propose related-key differential distinguishers based on the complementation property of Feistel ciphers. We show that with relaxed requirements on the complementation, i.e. the property does not have to hold for all keys and the complementation does not have to be on all bits, one can obtain a variety of distinguishers. We formulate criteria sufficient for attacks based on the complementation property. To stress the importance of our findings we provide analysis of the *full-round* primitives:

– For the hash mode of *Camellia-128* without FL, FL^{-1} layers, differential multicollisions with 2^{112} time
– For GOST, practical recovery of the full key with 31 related keys and 2^{38} time/data

Key words: Complementation, Feistel, Camellia, GOST

1 Introduction

It is a well established fact that the effective key size of DES[9] is 55 instead of 56 bits. The reduction of one bit is due to the complementation property of DES, i.e. by flipping all the bits in the key and in the plaintext, all the bits of the ciphertext will flip as well. Hence in an exhaustive key search, one has to try only half of the possible values for the key – the other complemented half would produce related ciphertexts. This property applies to all Feistel ciphers with round keys obtained as permutations of the master key bits/words, and with a round function that starts with an XOR of a single round key.

The complementation property can be seen as a simple related-key distinguisher applicable to all of the keys and detectable with a single pair of plaintexts and a corresponding pair of ciphertexts. The difference in the round keys, plaintexts and the ciphertexts is always -1, i.e. it is in all of the bits. In this paper we investigate the cases of ciphers with complementation properties applicable not necessarily to all of the keys, but only to a subset i.e. weak-key class, and with round key differences other than -1. We are aware of only one published result that analyzes the complementation property – the work of Bouillaguet et al.[3]. Even there the focus in not on the original property – the authors examine the generalizations of the complementation property, and exploit self-similarity of the rounds in the ciphers. Our work however targets exclusively the cases of complementation and only Feistel ciphers.

The starting point of our analysis is the observation that if instead of the requirement that the complementation property holds for all keys (as in the case of DES), we can examine only a subset of keys for which it applies. This leads to the problem of constructing a high probability differential in the key schedule of the cipher. We give the conditions on the output difference in the differential and obtain quite simple criteria for existence of related-key attacks based on the complementation property. The importance of our findings is shown on the example of two full-round Feistel ciphers: Camellia-128[1] and GOST[5]. We analyze Camellia-128 without the non-linear layers FL, FL^{-1} and show how to find a pair of keys that follow the low probability differential in the key schedule constructed to exploit the complementation – this allows us to attack the hash mode of this version of the cipher. Thus we obtain the first analysis on the full-round Camellia without the FL, FL^{-1} in the hash mode – it requires around 2^{112} encryptions. Complementation property of GOST has been known (see [7, 4]), however all of the proposed key recovery attacks require impractical time complexity. We show that if one uses several similar complementation properties, an efficient key recovery attack on GOST exists. Our attack requires 31 related-key pair, and only 2^{38} time and data complexities to recover the full 256-bit key. Thus we are able to perform the first experimental cryptanalysis of GOST on a computer.

2 Complementation Property of Feistel Constructions

The complementation property was first observed in DES. It is based on the observation that if one flips all of the bits of the master key and the plaintext, then all of the bits of the ciphertext will flip as well. The foundation of these observations for Feistel ciphers is given below. Without loss of generality we assume that the Feistel is balanced as the case for unbalanced Feistels can be examined similarly.

A balanced Feistel with r rounds is defined as:

$$\begin{aligned} L_{n+1} &= F(L_n, K_n) \oplus R_n \\ R_{n+1} &= L_n, \end{aligned}$$

where K_n is the n-th round key, $P = L_0||R_0$ is the plaintext, and $C = L_r||R_r$ is the ciphertext. In the vast majority of Feistel ciphers, the round function $F(L, K)$ can be decomposed as[3]:

$$F(L, K) = G(L \oplus K),$$

i.e. first the round key is bitwise added to the state L, followed by some additional non-linear and linear transformations (G is usually a Substitution-Permutation

[3] The round function of DES does not strictly follow this definition due to the expansion of the initial input from 32 bits to 48 bits, nonetheless our reasoning can still be applied to DES.

network). We use the term *classical Feistels* for the ciphers that have such an F function.

Let $KS(K)$ be the key schedule function of the cipher, i.e. given the master key K, the function produces $K_i, i = 1, \dots, r$ round keys:

$$KS(K) = (K_1, \dots, K_r)$$

Further assume that all of the round keys K_i are obtained by (possibly different) bit permutations of the master key K (as in the case of DES). If one has two related master keys K^1, K^2 such that $K^1 \oplus K^2 = -1$ (with -1 we denote the difference in all of the bits) then the following holds for all i: $K_i^1 \oplus K_i^2 = -1$. Let P^1, P^2 be two related plaintexts such that $P^1 \oplus P^2 = -1$, i.e. $L_0^1 \oplus L_0^2 = -1$ and $R_0^1 \oplus R_0^2 = -1$. Then by induction for each i we get:

$$\begin{aligned} L_{i+1}^1 \oplus L_{i+1}^2 =& F(L_i^1, K_i^1) \oplus R_i^1 \oplus F(L_i^1, K_i^1) \oplus R_i^1 = \\ & G(L_i^1 \oplus K_i^1) \oplus R_i^1 \oplus G(L_i^1 \oplus -1 \oplus K_i^1 \oplus -1) \oplus R_i^1 = R_i^1 \oplus R_i^2 = -1 \\ R_{i+1}^1 \oplus R_{i+1}^2 =& L_i^1 \oplus L_i^2 = -1 \end{aligned}$$

Therefore $L_r^1 \oplus L_r^2 = -1, R_r^1 \oplus R_r^2 = -1$ and hence there is a difference in all of the bits of the ciphertext.

The complementation property of such ciphers allows reduction of the key space by one bit as for the brute force of the whole key space it is sufficient to try only one half of all possible keys – the other half will produce a compliment ciphertext under a compliment plaintext.

The complementation property can be observed for ciphers that not necessarily have a key schedule composed of bit permutations. Notice, the only requirement on the key schedule is to produce complemented round keys.

Lemma 1 (Classical Feistel complementation). *Let for an n-bit classical Feistel cipher $E_K(P)$ with k-bit keys and a key schedule $KS(K)$ exist a differential with probability p for $KS(K)$ with output difference in all of the bits in all of the round keys, i.e.*

$$\exists \Delta : KS(K \oplus \Delta) \oplus KS(K) \xrightarrow{p} (-1, \dots, -1)$$

Then, if $p > 2^{-k}$, distinguisher for a weak-key class of size $p \cdot 2^k$ exists for the cipher $E_K(P)$.

Proof. Once the difference in all of the round keys is -1, the complementation property can be applied, i.e. the differential in the state holds with probability 1. Therefore if the attacker can build a differential with the input difference in the master keys Δ, and output difference -1 in all of the round keys, then the differential $(-1, \Delta) \rightarrow (-1)$ for the cipher $E_K(P)$ holds with probability p. To find the right key pair that follows the differential in the key schedule one has to try around $1/p$ pairs of randomly chosen master keys with input difference Δ, therefore the size of this weak key class is $2^k \cdot p$. For any cipher, to produce a pair of complemented plaintexts that result in complemented ciphertexts, one has to try around 2^n pairs, however even when $p < 2^{-n}$, a false positive (i.e. a complementation pair of plaintexts-ciphertexts that indicate belonging of a key to the weak-key class) can be easily detected by trying a few more pairs of complementing plaintexts. □

Remark 1 *The complementation property holds regardless of the number of rounds in the cipher, by increasing the number of rounds one cannot expect to get a better resistance against this type of attacks.*

Remark 2 *The additional key whitenings at the beginning and at the end of the Feistel do not influence the attack complexities, but merely change the input difference in the plaintext and the output difference in the ciphertext.*

The requirement of having the difference -1 in all of the round keys can be replaced with the requirement of having some difference $\varDelta$ which is not necessarily -1. We call this property *a partial complementation.* Also, instead of a single difference $\varDelta$ one can require two differences $\varDelta_1, \varDelta_2$ that alternate, i.e. the first round key has $\varDelta_1$, the second $\varDelta_2$, the third $\varDelta_1$, etc. – this is *an alternating complementation.*

Lemma 2 (Classical Feistel partial alternating complementation). *Let for an n-bit classical Feistel cipher $E_K(P)$ with k-bit keys and a key schedule $KS(K)$ exist a differential with probability p for $KS(K)$ with alternating differences in the round keys, i.e.*

$$\exists \varDelta : KS(K \oplus \varDelta) \oplus KS(K) \xrightarrow{p} (\varDelta_1, \varDelta_2, \varDelta_1, \varDelta_2, \ldots, \varDelta_1, \varDelta_2)$$

Then, if $p > 2^{-k}$, distinguisher for a weak-key class of size $p \cdot 2^k$ exists for the cipher $E_K(P)$.

Proof. We can follow the same logic as in the proof of Lemma 1 with one exception – the initial difference in the plaintext should be $(\varDelta_1, \varDelta_2)$. Then in each round, in the XOR the difference from the round key (either $\varDelta_1$ or $\varDelta_2$) would cancel the difference in the state. As they alternate with the same period of two rounds, the XOR will always produce zero difference, hence the probability of the differential in the state would be 1. Depending if the number of rounds is even or odd, the difference in the ciphertext would be either $(\varDelta_1, \varDelta_2)$ for even rounds, or $(\varDelta_2, \varDelta_1)$ for odd rounds. □

Remark 3 *Lemma 2 is more general then Lemma 1, as the later is a particular case of the former for $\varDelta_1 = \varDelta_2 = -1$.*

The round function of some Feistel ciphers instead of an XOR applies modular addition of the round key, i.e. $F(L, K) = G(L + K)$. We call this type of ciphers, *modular Feistels.* The (complementary) differential in the state of a modular Feistel not necessarily holds with probability 1 – the precise probability depends on the differences in the round key K_i and the state word L_i as well as on the number of rounds.

An efficient algorithm for computing the differential probability of modular addition was presented by Limpaa and Moriai in [8]. Our further analysis is based on this algorithm, however, due to space constraints we would not provide its description. Let $(X)_m$ be the m rightmost (least significant) bits of an n-bit word X and let $|X|$ be the Hamming weight, i.e. the number of bits with value 1, of the word X.

Lemma 3 (Modular Feistel alternating complementation[4]). *Let for an r-round n-bit modular Feistel cipher $E_K(P)$ with k-bit keys and a key schedule $KS(K)$ exist a differential with probability p for $KS(K)$ with alternating differences in the round keys, i.e.*

$$\exists \Delta : KS(K \oplus \Delta) \oplus KS(K) \xrightarrow{p} (\Delta_1, \Delta_2, \Delta_1, \Delta_2, \ldots, \Delta_1, \Delta_2)$$

Then, if $p \cdot 2^{-\lceil \frac{r}{2} \rceil (|(\Delta_1)_{n-1}| + |(\Delta_2)_{n-1}|)} > 2^{-k}$ and $2^{-\lceil \frac{r}{2} \rceil (|(\Delta_1)_{n-1}| + |(\Delta_2)_{n-1}|)} > 2^{-n}$, distinguisher for a weak-key class of size $p \cdot 2^k$ exists for the cipher $E_K(P)$.

Proof. In modular ciphers, we have to compute the probability of the differential in the state as well. As in r rounds, there are[5] $\lceil \frac{r}{2} \rceil$ round keys with Δ_1 difference, and the same number of keys with difference Δ_2, it is sufficient to find only the probability of one round (with both Δ_1 and Δ_2). The differences from the incoming round key and the state word should cancel, thus avoid any incoming difference in the SP network of the round function. Hence, by Algorithm 2 of [8], γ should be equal to zero, and the maximal probability of one round is reached when the incoming differences in the round key K_i and the state word L_i (or in the notation from [8], $\alpha = \beta$) are the same – in this case the probability of modular addition is $2^{-|(\Delta_1)_{n-1}|}$ or $2^{-|(\Delta_2)_{n-1}|}$. Taking into account the number of rounds, one obtains the claimed probability. The second requirement in the Lemma is to ensure that the probability of the differential in the state is not bellow 2^{-n}. $\square$

The variations of the complementation property presented above are indeed related-key differential distinguishers for ciphers. In both classical and modular Feistels, the size of the weak-key class depends only on the probability of the differential in the key schedule. However, to find and detect if a specific key belongs to the weak-key class differs between these two families, as for classical Feistels, the probability of the differential in the state is 1, whereas for modular Feistels, this probability might be lower. Hence, in the case of former one has to try around 2^P different pairs of keys and encrypt one pair of plaintexts, while in the case of modular Feistels, for each of the 2^P pairs of related-key has to encrypt 2^Q pairs of plaintexts ($2^{-P}, 2^{-Q}$ are the probabilities of the differential in the key schedule and in the state).

[4] One of our anonymous reviewers has informed us that a similar idea was used against DESX in Kelsey et al. [6].

[5] When r is odd, there are $\lceil \frac{r}{2} \rceil$ round keys with difference Δ_1, and $\lceil \frac{r}{2} \rceil - 1$ round keys with Δ_2.

3 The Case of Camellia-128

In this section we show how to apply the complementation property (Lemma 1) to Camellia-128[1] in the hash mode. We analyze the full-round *Camellia-128* without the non-linear layers, i.e. we assume FL, FL^{-1} to be identity functions.

3.1 Description

Camellia is a classical Feistel cipher with a non-linear key schedule defined as follows. The 128-bit master key K_L is split into two keys L, R, i.e. $K_L = L||R$ – both L and R are seen as 8-byte vectors. Further, these keys are fed to a 4-round Feistel-like transformation with an additional keys feedback after the second round (see Fig.1). Formally, the key schedule can be described as:

$$L_1||R_1 = K_L \tag{1}$$
$$L_2 = F(L_1 \oplus \Sigma_1) \oplus R_1; \qquad R_2 = L_1 \tag{2}$$
$$L_3 = F(L_2 \oplus \Sigma_2) \oplus R_2; \qquad R_3 = L_2 \tag{3}$$
$$\overline{L_3} = L_3 \oplus L_1; \qquad \overline{R_3} = R_3 \oplus R_1 \tag{4}$$
$$L_4 = F(\overline{L_3} \oplus \Sigma_3) \oplus \overline{R_3}; \qquad R_4 = \overline{L_3} \tag{5}$$
$$L_5 = F(L_4 \oplus \Sigma_4) \oplus R_4; \qquad R_5 = L_4 \tag{6}$$
$$K_A = L_5||R_5 \tag{7}$$

where Σ_i are word constants. In the sequel, we omit the addition of the constants as they play no role in our analysis. The function F is an SP network, with the S-layer defined as application of eight 8x8 S-boxes, and P-layer is a multiplication of the eight-byte input with 8x8 byte matrix P. All the round keys K_i used in the state are obtained from the two keys K_L and K_A with rotations on various amounts, e.g. $K_4 = K_L \lll_{15}, K_{15} = K_A \lll_{95}$, etc.

3.2 Complementing *Camellia-128*

From the description of Camellia-128 it follows that two different keys K_L, K_A are used, the first key being also the only input to the key schedule. Since the round keys are produced from these two keys with various rotations it follows that the differences in K_L, K_A have to be invariant of rotations and thus -1. Therefore, we need the differential $\Delta K_L \to (\Delta K_L, \Delta K_A)$ to be $(-1) \to (-1, -1)$.

The easiest way to build such differential is by providing a differential trail, i.e. besides specifying the input and output differences, fixing as well the intermediate differences after each transformation in the key schedule. Note that from the condition on the differential it follows that $\Delta L_1 = \Delta R_1 = \Delta L_5 = \Delta R_5 = -1$, i.e. each byte of these words has the fixed difference -1 (or ff in the hexadecimal representation). Therefore, in the first and the fourth round of the key schedule, the number of active bytes has to be maximal, i.e. eight active bytes will enter the S-layer. It is tempting to go with a trail that has no active bytes (or one

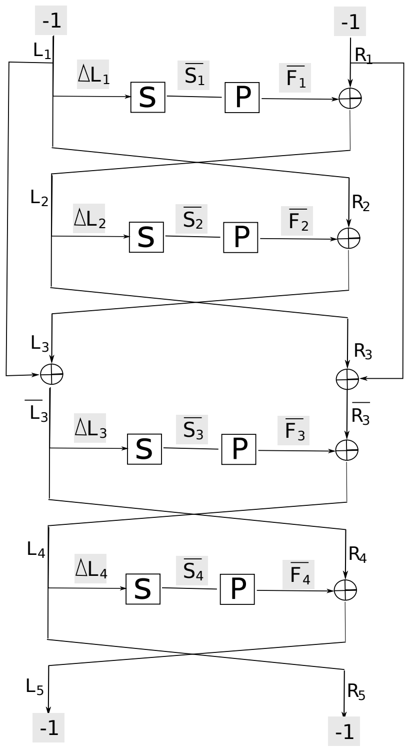

Fig. 1. The key schedule of *Camellia-128* with the $(-1,-1) \rightarrow (-1,-1)$ differential. The gray values are the differences.

active byte) in both the second and third round, hence obtain a trail of the form (we write only the round-by-round active bytes entering the F function):

$$8 \rightarrow 0 \rightarrow 0 \rightarrow 8 \text{ or } 8 \rightarrow 1 \rightarrow 1 \rightarrow 8$$

However, these types of trails are not possible due to the matrix multiplication P, i.e. P-layer. For example, if we require no active bytes in the second round, then this means the output of the F function in the first round has canceled with the -1 difference in R_1, i.e. if we denote with $\tilde{a} = (a_1, \dots, a_8)$ the output difference of the S-boxes in the function F of the first round, then the above condition can be expressed as:

$$P \cdot \tilde{a} \oplus (-1) = 0 \Rightarrow \tilde{a} = (0,0,0,0,-1,-1,-1,-1)$$

The solution vector $\tilde{a}$ has difference only in 4 bytes out of 8, while all the bijective S-boxes are active, i.e. we get a contradiction. Therefore, the second round of the key schedule cannot have zero active bytes. A similar situation can be observed when the second (or the third) round has only 1 active byte.

The above result suggests that the minimal number of active bytes in the key schedule is 8+2+2+8 = 20. Theoretically, this can lead to a trail with probability $2^{-6\cdot 20} = 2^{-120} > 2^{-128}$ when all the active S-boxes hold with probability 2^{-6}. Due to the specific input and output differences in the active S-boxes in the first and the fourth rounds, this is not achievable – the differential probability of these S-boxes is 2^{-7}. Therefore if we assume the differential is composed of a single trail only, its probability would always be lower than 2^{-128}.

Further we try to find the actual probability of the differential taking into account all possible differential trails that compose it. All the trails can be divided into two groups: trails that have the same path (i.e. they have the same position of the active bytes, but different values for the differences), and trails that have different path.

Let $\tilde{S_i}$ be a possible output difference of the S-layer at round i, and $\tilde{F_i}$ be an output difference of the F function at round i. Note, both $\tilde{S_i}, \tilde{F_i}$ are 8 byte vectors – $\tilde{S_i} = (s_i^1, \ldots, s_i^8), \tilde{F_i} = (f_i^1, \ldots, f_i^8)$. Also, let F_i be the actual output of the F function at round i. We will use $S(x)$ to denote the S-layer, and ΔL_i to denote the difference of the left state at round i, hence $S(\Delta L_i) = \tilde{S_i}$. From the definition of the round function it holds $F(\Delta L_i) = P \cdot S(\Delta L_i) = P \cdot \tilde{S_i} = \tilde{F_i}$.

For $\tilde{S_1}, \tilde{S_2}, \tilde{S_3}, \tilde{S_3}$ the following conditions apply (see Fig.1):

– $\tilde{S_1}$ is produced when -1 difference in L_1 goes through the S-layer:

$$\tilde{S_1} = S(-1) \tag{8}$$

– $\tilde{S_2}$ is produced with an XOR of $\tilde{F_1}$ and the difference -1 in R_1, followed by the S-layer:

$$\tilde{S_2} = S(\tilde{F_1} \oplus (-1)) = S(P \cdot \tilde{S_1} \oplus (-1)) \tag{9}$$

– $\tilde{S_3}$ is produced with application of the S-layer to $\Delta\overline{L_3}$:

$$\tilde{S_3} = S(\Delta\overline{L_3})) = S(P \cdot \tilde{S_2}) \tag{10}$$

Additionally, when $\tilde{F_3}$ is XOR-ed to $\Delta\overline{R_3}$, the output difference -1 is obtained in R_5:

$$\tilde{F_3} \oplus \Delta\overline{R_3} = P \cdot \tilde{S_3} \oplus P \cdot \tilde{S_1} = -1 \tag{11}$$

– $\tilde{S_4}$ is produced when -1 difference in R_5 goes through the S-layer:

$$\tilde{S_4} = S(-1) \tag{12}$$

Additionally, when $\tilde{F_4}$ is XOR-ed to $\Delta\overline{L_3}$, the output difference -1 is obtained in L_5:

$$\tilde{F_4} \oplus \Delta\overline{L_3} = P \cdot \tilde{S_4} \oplus P \cdot \tilde{S_2} = -1 \tag{13}$$

The probability of the differential can be computed as the sum of probabilities of all differential trails defined with 4 intermediate differences:

$$\sum_{(\tilde{S_1},\tilde{S_2},\tilde{S_3},\tilde{S_4})|\ (8),(9),(10),(11),(12),(13)\text{ are satisfied}} 2^{-7(|\tilde{S_1}|+|\tilde{S_2}|+|\tilde{S_3}|+|\tilde{S_4}|)} \tag{14}$$

where $|\tilde{S_i}|$ denotes the number of active bytes in $\tilde{S_i}$. In the following, we try to simplify the conditions and to achieve formula for computing the above probability.

Note that although both $\tilde{S_1} = S(-1)$ and $\tilde{S_4} = S(-1)$ are produced when (-1) goes through the S-layer, a randomly chosen difference $\tilde{S_1}$ and a difference $\tilde{S_4}$ are not necessarily the same (in fact they are different with a very high probability). To distinguish them we will use $S(-1)_{L_1}$ for the former and $S(-1)_{R_5}$ for the later.

Further we reduce the conditions on all $\tilde{S_i}$ to conditions only on $\tilde{S_2}, \tilde{S_3}$. From (11) and the linearity of the matrix multiplication P it follows that

$$P \cdot \tilde{S_3} \oplus P \cdot \tilde{S_1} = P \cdot (\tilde{S_3} \oplus \tilde{S_1}) = -1$$

This leads to:

$$\tilde{S_3} = P^{-1}(-1) \oplus \tilde{S_1} \tag{15}$$

Similarly, from (12) and (13) we get:

$$\tilde{S_2} = P^{-1}(-1) \oplus S(-1)_{R_5} \tag{16}$$

Taking into account (15), the condition (9) can be expressed as:

$$\tilde{S_2} = S(P \cdot \tilde{S_1} \oplus (-1)) = S(P \cdot (\tilde{S_3} \oplus P^{-1}(-1)) \oplus (-1)) = \tag{17}$$

$$= S(P \cdot \tilde{S_3} \oplus (-1) \oplus (-1)) = S(P \cdot \tilde{S_3}) \tag{18}$$

Let us summarize our findings. We get that for $\tilde{S_2}, \tilde{S_3}$ defined as:

$$\tilde{S_2} = P^{-1}(-1) \oplus S(-1)_{R_5} \tag{19}$$

$$\tilde{S_3} = P^{-1}(-1) \oplus S(-1)_{L_1} \tag{20}$$

two additional conditions have to hold:

$$\tilde{S_2} = S(P \cdot \tilde{S_3}) \tag{21}$$

$$\tilde{S_3} = S(P \cdot \tilde{S_2}) \tag{22}$$

In $\tilde{S_1}, \tilde{S_4}$ there are always 8 active S-boxes. The number of active S-boxes in $\tilde{S_2}, \tilde{S_3}$ is defined by the above conditions. As P is linear, we can compute the value of the vector $P^{-1}(-1)$, i.e.

$$P^{-1}(-1) = (0, 0, 0, 0, ff, ff, ff, ff) \tag{23}$$

Since the S-boxes in Camellia are bijective, the vector $S(-1)$ always has 8 active S-boxes. Therefore from (19),(20) we can conclude that the first 4 elements of $\tilde{S_2}, \tilde{S_3}$ have to be non-zero, thus the number of active S-boxes in round 2 and 3 is at least 4 (the first 4 bytes must be active). Additionally, regarding the number and position of the active S-boxes, since there are always at least 4 active S-boxes in $\tilde{S_2}$ and $\tilde{S_3}$, the conditions (21),(22) can always be satisfied (the branch number of P is 4).

Finally, we can give the probability of the differential $(-1,-1) \rightarrow (-1,-1)$:

$$\sum_{(\tilde{S_2},\tilde{S_3}) \text{ satisfy } (19),\,(20),\,(21),\,(22)} 2^{-7(8+|\tilde{S_2}|+|\tilde{S_3}|+8)} \tag{24}$$

Recall that a differential is a collection of trails that take the same path (have the same position of the active bytes) and trails that take different path. We group all trails that take the same path into one single *truncated trail.* Then a differential is a collection of truncated trails and hence its probability is the sum of probabilities of the truncated trails. To define a truncated trail we just have to fix the position of the actives S-boxes in the four rounds of the key schedule. With T_i we denote the truncated difference entering the round function of round i. Then a truncated trail can be defined as T_1, T_2, T_3, T_4. An actual trail with $\tilde{S_1}, \ldots, \tilde{S_4}$ belongs to a truncated trail if the position of the active S-boxes in S_i coincide with the position of the active S-boxes in T_i. Obviously $T_1 = T_4 = (1, 1, \ldots, 1)$ as all the S-boxes in the first and the fourth round are active. For the probability of the differential we get:

$$\sum_{(T_2,T_3)} 2^{-7(8+|T_2|+|T_3|+8)} \#\{(\tilde{S_2},\tilde{S_3}) | \tilde{S_2} \in T_2, \tilde{S_3} \in T_3, \tilde{S_1}, \tilde{S_2} \text{ satisfy } (19),\,(20),\,(21),\,(22)\ \} \tag{25}$$

Hence, to find the probability of the differential, we only have to count the number of possible differential trails (that satisfy a set of conditions) in all possible truncated trails T_2, T_3 of the form $(1,1,1,1,x_5,x_6,x_7,x_8), x_i \in \{0,1\}$. To proceed further we define the notion of compliance.

Definition 1. *Two differences Δ_1, Δ_2 comply through the function $f(x)$ if there exist x such that $f(x \oplus \Delta_1) \oplus f(x) = \Delta_2$.*

This notion is introduced to check if some input difference Δ_1 at function $f(x)$ can produce output difference Δ_2.

Observation 1 *Two randomly chosen differences Δ_1, Δ_2 comply through the S-boxes of Camellia with probability $\frac{127}{255} \approx 2^{-1}$.*

Every input difference to the S-box can go to 127 output differences or approximately to 2^7 out of $2^8 - 1$ possible, which is around 2^{-1}.

As an example, let us compute the number of possible trails for the case when T_2, T_3 have all 8 active bytes. From the properties of the S-boxes used in *Camellia* we have that each input byte difference (including the difference ff) can go to 127 or approximately[6] 2^7 distinct output differences. Since we have

[6] We can approximate with 2^7 as one of the output differences happens twice, which means that although we increase the number from 127 to 128, on the other hand we decrease the probability for this difference from 2^{-6} to 2^{-7}, hence the two rounding errors compensate one another. This fact can easily be checked if one takes instead of bytes, 7-bit nibbles. Then the maximal differential probability of 7x7 S-box can be 2^6.

8 active input bytes in $S(-1)_{L_1}$ and in $S(-1)_{R_5}$, there are in total $2^{7\cdot 8} = 2^{56}$ differences for $\tilde{S_2}$ and $\tilde{S_3}$ (see the definitions (19),(20)). As $\tilde{S_2}$ has 8 active bytes, the following condition has to hold:

$$(d_1, \ldots, d_8) = P^{-1}(-1) \oplus (s_{R_5}^1, \ldots, s_{R_5}^8) \tag{26}$$

$$= (0, 0, 0, 0, ff, ff, ff, ff) \oplus (s_{R_5}^1, \ldots, s_{R_5}^8), \tag{27}$$

where all d_i are non-zero. Hence, out of all 2^{56} this condition satisfy $2^{56} \cdot (1 - 127^{-4}) \approx 2^{56}$ differences, or approximately all. A similar conclusion can be obtained regarding (20).

Now let us focus on (21),(22). The probability that $\tilde{S_2}$ comply with $\tilde{S_3}$ from (21) can be computed as:

1. the probability that $P \cdot \tilde{S_3}$ is 8 byte difference – it is approximately 1. In the general case, when $\tilde{S_2}$ has i active bytes, the probability is approximately $2^{-8\cdot(8-i)}$.
2. the probability that each of the differences in 8 bytes of $\tilde{S_2}$ and $P \cdot \tilde{S_3}$ comply. This is 2^{-8}, while in the general case it is 2^{-i} for differences in i bytes.

Therefore, for a randomly chosen differences the probability of (21) is 2^{-8}. A similar reasoning can be applied to (22). Hence, out of all possible $\tilde{S_2}, \tilde{S_3}$ there are $2^{56} \cdot 2^{56} \cdot 2^{-8} \cdot 2^{-8} = 2^{96}$ differences that satisfy all four conditions. Therefore, for $T_2 = T_3 = (1, 1, 1, 1, 1, 1, 1, 1)$, the probability of the differential is at least:

$$2^{96} \cdot 2^{-7(8+8+8+8)} = 2^{96} \cdot 2^{-224} = 2^{-128} \tag{28}$$

If we take into account all possible T_2, T_3 for the probability of the differential we get:

$$\sum_{i,j} 2^{-7(8+i+j+8)} C_4^{i-4} \cdot C_4^{j-4} 2^{112-8\cdot(8-i)-8\cdot(8-j)} 2^{-8(8-i)-i} 2^{-8(8-j)-j} \approx \tag{29}$$

$$\approx 2^{-128} \tag{30}$$

Thus, by Lemma 1, the size of the weak key class is $2^{128} \cdot 2^{-128} = 1$. For this key K, the complementation property holds, i.e. $KS(K \oplus (-1)) \oplus KS(K) = -1$, and taking into account the whitening keys, we get that for any plaintext P, it holds

$$E_{K\oplus(-1)}(P) = E_K(P).$$

Note that the size of the weak key class is too small for any attack on the cipher, however it is sufficient for an attack on the hash function mode of the cipher. As a compression function, we can choose the standard Davies-Meyer compression mode:

$$C(H, M) = E_M(H) \oplus H \tag{31}$$

Let K be the key value for which the $(-1, -1) \to (-1, -1)$ differential in the key schedule holds. For the compression function we get that for any H the following holds:

$$C(H, K \oplus (-1)) \oplus C(H, K) = E_{K\oplus(-1)}(H) \oplus H \oplus E_K(H) \oplus H = 0 \tag{32}$$

Therefore if we can find the correct key K (which is indeed the correct message M, as $M = K$ in the hash mode), we can produce collisions for the compression function of *Camellia*. Note, as H can be arbitrary, this leads to collisions for the whole hash function. To find the exact value of the key K we use the conditions (19)-(22) combined into the algorithm:

1. Create a set $\tilde{S}$ of all possible differences $P^{-1}(-1) \oplus S(-1)$ – the size of the set is 2^{56}
2. Create a set S^R of pairs of differences $(\delta_2, \delta_3), \delta_2, \delta_3 \in \tilde{S}$ such that δ_2 complies with $P \cdot \delta_3$ and δ_3 complies with $P \cdot \delta_2$ - the size of this set is 2^{96}
3. Choose a random pair (δ_2, δ_3) from S^R
4. Produce the value of L_1 (and the corresponding F_1) that converts -1 into the $\delta_3 \oplus P^{-1}(-1)$, i.e. $S(L_1 \oplus (-1)) \oplus S(L_1) = \delta_3 \oplus P^{-1}(-1)$. As δ_3 has 8 active S-boxes, and for each active S-box there are 2 different values (A and $A \oplus (-1)$), for a fixed δ_3 there are 2^8 possible values of (L_1, F_1)
5. Produce similarly the values of (L_4, F_4) from δ_2
6. Produce $F_3 = L_4 \oplus \overline{F_3} = L_4 \oplus F_1$, and $\overline{L_3} = F^{-1}(F_3)$. Check if $F(\overline{L_3} \oplus P \cdot \delta_2) \oplus F(\overline{L_3}) = P \cdot \delta_3$. If not, go to step 3
7. Produce $F_2 = \overline{L_3}$, and $L_2 = F^{-1}(F_2)$. Check if $F(L_2 \oplus P \cdot \delta_3) \oplus F(L_2) = P \cdot \delta_2$. If not, go to step 3
8. Output the key $(L_1, R_1) = (L_1, F(L_1) \oplus L_2)$

The probability of steps 6,7 is 2^{-56} each and there are $2^{2(48+8)}$ possible (L_1, F_1) and (L_4, F_4). Hence, after repeating step three 2^{96} times and steps four-five 2^{112} times, one key candidate will be produced. Thus the complexity of the algorithm is 2^{112}.

Note that with an effort of 2^{112} we can produce one collision for the compression function of *Camellia-128* (without FL, FL^{-1}). Once we have the correct message M, we can produce collision for any input chaining value. This means that for any messages M_1, M_3 (the M_3 block is used as message padding), we can produce a collision for the hash function of *Camellia-128*. The colliding pairs are $(M_1||M \oplus (-1)||M_3)$ and $(M_1||M||M_3)$. Therefore, to produce q collisions with the same fixed difference between the message words (the difference is $(0||-1||0)$ we need 2^{112} calls to the hash function[7]. On the other hand, for the generic case, producing such collisions (they are indeed called differential q multicollisions, see [2]), one needs around $q2^{\frac{q-2}{q+2}128}$ calls to the hash function. Hence, producing 256 differential multicollisions requires $2^8 \cdot 2^{\frac{254}{258}128} \approx 2^{134}$ encryptions whereas for the hash function of *Camellia-128* without the non-linear layers FL, FL^{-1} in the Davies-Meyer mode, they can be produced with 2^{112} calls to the hash function.

4 The Case of GOST

In this section we show how the partial complementation property (Lemma 3) of GOST can be used to launch a practical related-key recovery attack on the

[7] Actually, the number is smaller, as one hash function call requires much larger number of operations compared to the steps of our algorithm.

full-round cipher. We note that the mentioned below complementation properties have been known and exploited in attacks on GOST[7, 4]. However, to the best of our knowledge, all the attacks on GOST that recover the full key, are impractical.

4.1 Description of GOST

GOST is a modular 32-round Feistel cipher with 256-bit key. The key schedule of GOST is trivial. The master key K is divided into eight 32-bit words $K_i, i = 1, \ldots, 8$ and in each of the four groups of 8 rounds, the round keys $RK_j, j = 1, \ldots, 32$ are permutation of the key words K_i, i.e.

$$(RK_1, \ldots, RK_8) = (K_1, \ldots, K_8) \tag{33}$$
$$(RK_9, \ldots, RK_{16}) = (K_1, \ldots, K_8) \tag{34}$$
$$(RK_{17}, \ldots, RK_{24}) = (K_1, \ldots, K_8) \tag{35}$$
$$(RK_{25}, \ldots, RK_{32}) = (K_8, \ldots, K_1) \tag{36}$$

4.2 Complementing GOST

As GOST is a modular Feistel, Lemma 3 can be applied to this cipher. The round keys do not allow the choice of different alternating Δ_1, Δ_2 as each of the key words K_i is used in both even and odd rounds. For example, K_1 is used in rounds 1, 9, 17, and *32*. Therefore, one has to choose $\Delta_1 = \Delta_2 = \Delta$, i.e. all of the keys K_i have the same difference Δ. The differential in the key schedule holds with probability $p = 1$. To maximize the probability of the differential in the state, one has to choose $\Delta = 2^{31}$ – in this case the size of the class is 2^{256}, i.e. the complementation is applicable to all of the keys.

One can maintain the same size of the weak-key class (all keys), but reduce the probability of the differential in the state. For example, when $\Delta_1 = \Delta_2 = \Delta = 2^i, i = 0, \ldots, 30$, then the partial complementation property is still applicable to all keys, but detecting this property requires more data, i.e. instead of the previous one pair of plaintexts and ciphertexts, by Lemma 3 now one needs $1/2^{-16(1+1)} = 2^{32}$ pairs. This weakens the distinguisher, but allows key recovery attacks. Let $\Delta = 2^m, m < 31$. If in some round i, one knows the value of the state S that is modularly added to the the round key RK_i (in the state/key pair, the m-th bit has the difference 1), then under the assumption that the differences have canceled, one can find the exact value RK_i^m of the m-bit of the round key RK_i, i.e. if S is known, and

$$(S + RK_i) \oplus ((S \oplus 2^m) + (RK_i \oplus 2^m)) = 0,$$

then the value RK_i^m of the m-th bit of RK_i can be computed as:

$$RK_i^m = S^m \oplus 1. \tag{37}$$

It is trivial to check that only under such values of RK_i^m and S^m the differences would cancel. For $m = 31$, i.e. when the difference is in the most significant bit, then the cancellation always occurs, hence one cannot find the exact value of the most significant bit with this approach.

The above single-bit recovery can be applied sequentially to all the bits of the round key RK_i, thus the whole RK_i can be recovered. Once the i-th round key is known, one can compute the state of the cipher in the next round and thus repeat the same process but for the round key RK_{i+1}. Hence this domino effect allows to recover all of the round keys resulting in a full key recovery of the master key. The attack presented below is a related-key attack with 31 related-key pairs. For the secret master key K with the key words K_i, $K = (K_1, \dots, K_8)$, the related keys K^i are defined as $K^i = (K_1 \oplus 2^i, \dots, K_8 \oplus 2^i), i = 0, \dots, 30$. The algorithm can be formulated as:

1. For each of the 31 related-key pairs $(K, K^i), i = 0, \dots, 30$ create 2^{32} pairs of plaintexts $(P^i_j, P^i_j \oplus 2^i)$ and obtain the corresponding ciphertexts $(C^i_j, \tilde{C^i_j}), j = 0, \dots, 2^{32} - 1$.
2. For each $i, i = 0, \dots, 30$, find the pair of ciphertexts that have the required difference 2^i, i.e. find j_i, such that $C^i_{j_i} \oplus \tilde{C^i_{j_i}} = 2^i, i = 0, \dots, 30$. The corresponding plaintext pairs are $(P^i_{j_i}, P^i_{j_i} \oplus 2^i)$. In total there are 31 such pairs.
3. For each round $r = 1, \dots 8$, recover the key word K_r.
   (a) For each $k, k = 0, \dots, 30$, the k-bit of K_r can be recovered from the knowledge of incoming state. In the first round, the value of the state is known, i.e. $P^k_{j_k}$, and therefore $K^k_r = P^k_{j_k} \oplus 1$ (see (37)). In total, 31 out of 32 bits of K_r are recovered.
   (b) Guess the most significant bit of K_r, and compute the values of the 31 states for the next round – this can be performed as one knows both the state and the round key.

The encryption of the initial 2^{32} pairs of plaintexts for each i, guarantees that with a high probability one can find a pair of ciphertexts with the same difference – hence this pair follows the differential in the state. For each round, one has to guess only a single bit (the most significant bit) of the round key, thus step 3 has to be repeated at most 2^8 times. Therefore the time complexity of the full key-recovery attack is $2 \cdot (31 \cdot 2^{32} + 2^8) \approx 2^{38}$ encryptions and a similar data complexity of 2^{38} chosen plaintexts.

The low complexities allow to perform an experimental cryptanalysis of GOST on a computer. We have followed the attack algorithm described above and were able to verify our approach by recovering the full 256-bit key – our unoptimized implementation ran for one day on a single Intel i5 core. As the key recovery can be parallelized, another implementation was able to recover the full key in around 7 hours using four Intel i5 cores.

5 Conclusion

We have shown a potential vulnerability in Feistel ciphers based on the complementation property that results in relatively easily detectable related-key differential attacks. Two such attacks on full-round Feistel primitives, the hash mode of *Camellia-128* without FL, FL^{-1}, and the block cipher GOST, have been presented in this paper.

We have deduced a simple criteria for cryptanalysis of classical Feistel ciphers: *if for the key schedule there exists a high probability differential that produces alternating differences in the round keys then the cipher is vulnerable to related-key attacks, regardless of the number of rounds in the state.* Moreover, from the analysis of *Camellia-128* without FL, FL^{-1}, one can conclude that even if such differential has a low probability, but a pair of keys following the differential could be found, the hash mode of the cipher is still vulnerable.

The Feistel ciphers that use modular addition of the round keys in the state are less susceptible to this type of attacks as the data required to detect the complementation property depends on the number of rounds as well. However, from the analysis of GOST one can see that, *when the alternating differences in the round keys have a low Hamming weight, such ciphers are potential targets of complementation weaknesses as well.* Our related-key attack on GOST was confirmed experimentally.

We believe that our attacks based on the complementation property might be launched on several other existing Feistel primitives, i.e. this paper does not exhaust the possible targets. Thus the approach presented here can be seen as simple tool for cryptanalysis of current Feistel primitives, but also an important security threat that should be taken into account when designing new primitives based on Feistel.

Acknowledgement

The authors would like to thank anonymous reviewers of FSE 2013 for their helpful comments. Ivica Nikolić is supported by the Singapore National Research Foundation under Research Grant NRF-CRP2-2007-03.

References

1. K. Aoki, T. Ichikawa, M. Kanda, M. Matsui, S. Moriai, J. Nakajima, and T. Tokita. Camellia: A 128-bit block cipher suitable for multiple platforms - design and analysis. In D. R. Stinson and S. E. Tavares, editors, *Selected Areas in Cryptography*, volume 2012 of *Lecture Notes in Computer Science*, pages 39–56. Springer, 2000.
2. 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.
3. C. Bouillaguet, O. Dunkelman, G. Leurent, and P.-A. Fouque. Another look at complementation properties. In S. Hong and T. Iwata, editors, *FSE*, volume 6147 of *Lecture Notes in Computer Science*, pages 347–364. Springer, 2010.
4. I. Dinur, O. Dunkelman, and A. Shamir. Improved attacks on full GOST. In A. Canteaut, editor, *FSE*, volume 7549 of *Lecture Notes in Computer Science*, pages 9–28. Springer, 2012.
5. Government Committee of the USSR for Standards. GOST, Gosudarstvennyi Standard 28147-89,"Cryptographic Protection for Data Processing Systems". 1989.

6. J. Kelsey, B. Schneier, and D. Wagner. Related-key cryptanalysis of 3-WAY, Biham-DES, CAST, DES-X, NewDES, RC2, and TEA. In Y. Han, T. Okamoto, and S. Qing, editors, *ICICS*, volume 1334 of *Lecture Notes in Computer Science*, pages 233–246. Springer, 1997.
7. Y. Ko, S. Hong, W. Lee, S. Lee, and J.-S. Kang. Related key differential attacks on 27 rounds of XTEA and full-round GOST. In B. K. Roy and W. Meier, editors, *FSE*, volume 3017 of *Lecture Notes in Computer Science*, pages 299–316. Springer, 2004.
8. H. Lipmaa and S. Moriai. Efficient algorithms for computing differential properties of addition. In M. Matsui, editor, *FSE*, volume 2355 of *Lecture Notes in Computer Science*, pages 336–350. Springer, 2001.
9. National Bureau of Standards. Data Encryption Standard. U.S. Department of Commerce, FIPS pub. 46, January 1977.