Preimage Attacks on Feistel-SP Functions: Impact of Omitting the Last Network Twist

Yu Sasaki

NTT Secure Platform Laboratories
3-9-11 Midori-cho, Musashino-shi, Tokyo 180-8585 Japan
sasaki.yu@lab.ntt.co.jp

Abstract. In this paper, generic attacks are presented against hash functions that are constructed by a hashing mode instantiating a Feistel or generalized Feistel networks with an SP-round function. It is observed that the omission of the network twist in the last round can be a weakness against preimage attacks. The first target is a standard Feistel network with an SP round function. Up to 11 rounds can be attacked in generic if a condition on a key schedule function is satisfied. The second target is a 4-branch type-2 generalized Feistel network with an SP round function. Up to 15 rounds can be attacked in generic. These generic attacks are then applied to hashing modes of ISO standard ciphers Camellia-128 without FL and whitening layers and CLEFIA-128.

Keywords: Feistel, generalized Feistel, SP round function, hashing modes, meet-in-the-middle attack, preimage attack, Camellia, CLEFIA.

1 Introduction

Designing secure and efficient symmetric-key primitives is a long-term challenge in the cryptographic community. One of the most successful designs is AES [7,28]. Since then, many designs use an AES-based transformation as a core of their algorithms. An unique design philosophy of AES is the omission of the diffusion called MixColumns in the last round. The purpose of this design is making the encryption and decryption algorithms symmetric, while it does not lower the provable security bound against differential and linear cryptanalysis. However, the omission impacts to the security for other cryptanalytic approaches. Dunkelman and Keller discussed its impact in [8]. Sasaki also showed that the omission could be exploited by an attacker in several hashing modes [21].

Another widely used design approach is the Feistel network, which was firstly used in DES [5], and the generalized Feistel network (GFN) [29]. The computation structures of Feistel network and 4-branch type-2 GFN are shown in Fig. 1. In the Feistel network, the data is separated into the left and right halves $L\|R$, and then R is updated by $R \oplus F(k, L)$, where F is called a round function and k represents a subkey. Finally, the left and right halves are exchanged, *i.e.,* $R \oplus F(k, L)\|L$. The ciphertext is computed by iterating this transformation several times. As shown in Fig. 1, several designs omit the network twist in the last

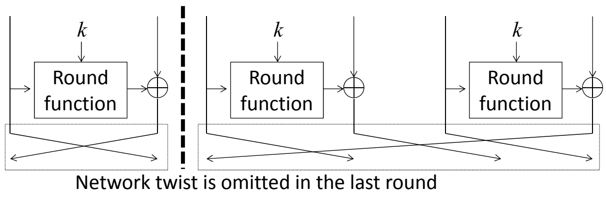

Fig. 1. Left: sketch of Feistel network, Right: sketch of 4-branch type-2 GFN

round, for example, DES [5] and ISO standard ciphers Camellia [2,12], CLEFIA [27,13], and HIGHT [10,12]. The omission of the last network twist makes the encryption and decryption algorithms symmetric.

Here, we raise a simple question; *What is the impact of omitting the last network twist in the Feistel network and GFN with respect to the security?* This paper answers this question by showing an attack against Feistel based hash functions that works more efficiently when the last network twist is omitted.

One may say that analyzing hash functions constructed by a Feistel cipher, especially for dedicated algorithms such as Camellia and CLEFIA, is meaningless unless someone develops a system that actually implements their hashing modes. However, we believe such analysis is important from the following reason.

> Non-cryptographic experts do not always tell cryptographers which hash function algorithm they implemented. Such information is sometimes never opened. Thus, it is important for cryptographers to prepare for the potential use by non-experts. Hashing modes based on an n-bit block-cipher, *e.g.* the Matyas-Meyer-Oseas (MMO) mode [16, Algorithm 9.41], is internationally standardized by ISO [11]. Camellia and CLEFIA are also internationally standardized by ISO [12,13]. MMO-Camellia and MMO-CLEFIA are important candidates for the potential use by non-experts because giving a guideline of good technology to non-experts is one of the purposes of the standardization.

So far, several researchers have studied the security of Feistel functions. Knudsen and Rijmen showed a collision attack for 7 rounds in the MMO mode [15]. Sasaki and Yasuda analyzed the Feistel network with an Substitution-Permutation (SP) round function. They showed a collision attack for a half of the state for 11 rounds in the MMO mode [24]. The attack was later improved and implemented on reduced-round Camellia [23]. Moon *et al.* presented preimage attacks on Feistel network, GFN, and Misty network with an SP round function [17]. They attacked 6 rounds of a Feistel-SP function and 9 rounds of a 4-branch type-2 GFN-SP function.

Our Contributions. In this paper, we present meet-in-the-middle (MitM) preimage attacks against hash functions that are constructed by the MMO or other Preneel-Govaerts-Vandewalle (PGV) modes [20] instantiating a Feistel or 4-branch Type-2 GFN ciphers with an SP-round function. Regarding the Feistel network, 11 rounds can be attacked when an attacker can control several bits of the last round subkey by choosing the first round subkey. Regarding 4-branch type-2 GFN, 15 rounds can be attacked when the relation between the first-round and the last-round subkeys is random. If the last network twist is not omitted. the number of attacked rounds is 6 and 10 for the Feistel and 4-branch type-2 GFN, respectively,

The MitM attack separates the target algorithm into two parts called *forward chunk* and *backward chunk* so that each chunk includes several bits which are independent of the other chunk. Such bits are called *free bits*. So far, there are two types of the MitM attacks. One is setting the free bits in the key and the other is setting the free bits in the internal state. In this paper, we take the second approach.[1] Note that, during the computation of one chunk, all previous work treat the free bits for the other chunk as unknown.

Our attacks are based on the following two ideas. Note that these ideas are not specific for SP-round functions.

1. The omission of the last network twist can be exploited by the splice-and-cut technique [3]. When the last and then first rounds are computed in this order, the input value to the round function do not change. Therefore, if the subkey values are identical, the impact of these two rounds cancel each other. The same situation also occurs between the second last and the second rounds. In the hash function, the key value can be chosen by the attacker and thus the round-shrink can be caused deliberately. Note that the cancellation of the round function was exploited by Gauravaram *et al.* [9]. Our discovery is that the cancellation gives more impacts when the last network twist is omitted.
2. During the computation of one chunk, free bits for the other chunk do not have to be completely independent as long as they linearly relate to the computation. Therefore, for the computation of each chunk, we trace how the free bits for the other chunk relate rather than treat them as unknown immediately. Our attack on 4-branch type-2 GFN traces the linearity over 10 rounds, and thus the idea works efficiently.

We apply these techniques to block-ciphers Camellia-128 without the FL and whitening layers and CLEFIA-128 in hashing modes. Camellia is a Feistel-SP cipher but its P-layer does not satisfy the maximum branch number. Thus, the attack can be extended compared to the generic case. We show an attack up to 13 rounds of Camellia-128 hashing modes. CLEFIA adopts 4-branch Type-2 GFN but uses two different diffusion matrices for the diffusion switching mechanism [25,26]. This increases the security and thus the attack becomes worse than the generic case. We show an attack up to 12 rounds of CLEFIA-128 hashing modes.

[1] Setting free bits in the key is impossible without defining a key schedule algorithm.

2 Preliminaries

2.1 Specification of Camellia

Camellia was jointly designed by NTT and Mitsubishi Electric Corporation. It is widely standardized or recommended, *e.g.*, ISO [12], NESSIE [19], and CRYPTREC [6]. This paper attacks Camellia-128, where both of the key and block sizes are 128 bits. We attack a weak variant of Camellia-128 where computations called FL and whitening layers are omitted.

Let M and K be a 128-bit plaintext and a secret key, respectively. Eighteen 64-bit round keys $k_0, \ldots, k_{17}$ are generated from K. Let X_r^L and X_r^R $(0 \le r \le 18)$ be left and right 64-bits of the internal state in each round. The plaintext is loaded into $X_0^L \| X_0^R$. Then, $X_r^L = X_{r-1}^R \oplus F(X_{r-1}^L, k_{r-1})$ and $X_r^R = X_{r-1}^L$ for $1 \le r \le 17$ is computed up to the second last round. In the last round, The ciphertext $X_{18}^L \| X_{18}^R$ is computed by $X_{18}^L = X_{17}^L$ and $X_{18}^R = X_{17}^R \oplus F(X_{17}^L, k_{17})$, namely, the last network twist is omitted.

The key schedule takes a 128-bit key K as input and firstly produces another 128-bit value K_A. We later analyze subkey values k_0, k_1, k_{11}, and k_{12}. These subkeys are defined as $k_0 \| k_1 = K_A$, k_{11} is the right half of $(K_A \lll 60)$, and k_{12} is the left half of $(K \lll 94)$.

The round function consists of a 64-bit subkey addition, S-box transformation, and a diffusion called P-layer. The size of each S-box is 8 bits, and thus 8 S-boxes are applied. Let $(z_0 \| z_1 \| \cdots \| z_7)$ be 64-bit values input to the P-layer. The output $(z'_0 \| z'_1 \| \cdots \| z'_7)$ is computed as follows. Here, $z[s, t, u, \cdots]$ means $z_s \oplus z_t \oplus z_u \oplus \cdots$. The branch number of P is only 5. This is different from the case of an MDS matrix multiplication.

$$z'_0 = z[0,2,3,5,6,7],\ z'_2 = z[0,1,2,4,5,7],\ z'_4 = z[0,1,5,6,7],\ z'_6 = z[2,3,4,5,7],$$
$$z'_1 = z[0,1,3,4,6,7],\ z'_3 = z[1,2,3,4,5,6],\ z'_5 = z[1,2,4,6,7],\ z'_7 = z[0,3,4,5,6].$$

2.2 Specification of CLEFIA

CLEFIA is a block-cipher proposed at FSE 2007 by Shirai *et al.* [27]. It is standardized by ISO [13] as a lightweight cipher. In this paper, we attack CLEFIA-128, where both of the block size and the key size are 128 bits. It adopts the type-2 generalized Feistel structure with 4 branches and consists of 18 rounds. Two round functions F^L and F^R consist of a 32-bit subkey addition, an S-box transformation, and a multiplication by an MDS matrix. The size of each S-box is 8 bits, and thus 4 S-boxes are applied in each of the left and right functions. MDS matrices for the left and right functions are different.

Let M and K be a 128-bit plaintext and a secret key, respectively. Thirty-six 32-bit subkeys $k_0, \ldots, k_{35}$ and four 32-bit whitening keys wk_0, wk_1, wk_2, wk_3 are generated from K. Let $X_r^0 \| X_r^1 \| X_r^2 \| X_r^3$ $(0 \le r \le 18)$ be an input internal state in each round. The plaintext is loaded into $X_0^0 \| X_0^1 \| X_0^2 \| X_0^3$. Then, the second and fourth words are updated by the pre-whitening operation, *i.e.*, $X_0^1 \leftarrow$

$X_0^1 \oplus wk_0$ and $X_0^3 \leftarrow X_0^3 \oplus wk_1$. Then, internal state is updated by the following computation up to the second last round (for $1 \leq r \leq 17$);

$$X_r^0 = X_{r-1}^3 \oplus F^R(X_{r-1}^2, k_{2r-1}),\ X_r^1 = X_{r-1}^0,$$
$$X_r^2 = X_{r-1}^1 \oplus F^L(X_{r-1}^0, k_{2r-2}),\ X_r^3 = X_{r-1}^2.$$

In the last round, $X_{18}^0 \| X_{18}^1 \| X_{18}^2 \| X_{18}^3$ is computed by $X_{18}^0 = X_{17}^0, X_{18}^1 = X_{17}^1 \oplus F^L(X_{17}^0, k_{34}), X_{18}^2 = X_{17}^2, X_{18}^3 = X_{17}^3 \oplus F^R(X_{17}^2, k_{35})$, namely, the last network twist is omitted. Finally, the second and fourth words are updated by the post-whitening operation, *i.e.*, $X_{18}^1 \leftarrow X_{18}^1 \oplus wk_2$ and $X_{18}^3 \leftarrow X_{18}^3 \oplus wk_3$, and $X_{18}^0 \| X_{18}^1 \| X_{18}^2 \| X_{18}^3$ is output as the ciphertext.

2.3 Feistel and 4-Branch Type-2 GFN with an SP Round Function

In this paper, we firstly analyze generic Feistel and 4-Branch Type-2 GFN structures with an SP round function. Analyzing such generic structure can be seen many papers [4,14,17,23,24,26]. These structures are generally represented by several parameters *i.e.*, the block size N, the S-box size c, and the number of S-boxes in each round b. The attack strategy and efficiency depends on these parameters. In this paper, to make a comparison of attacks against Camellia and CLEFIA clear, we fix the parameters to $(N, c, b) = (128, 8, 8)$ for the standard Feistel and $(N, c, b) = (128, 8, 4)$ for 4-branch type-2 GFN.

An SP round function consists of three operations: subkey addition, S-layer, and P-layer. In the subkey addition, a subkey is XORed to the state. In the S-layer, b S-boxes with the size of c bits are applied. In the P-layer, a linear computation whose branch number is $b+1$ is performed. An MDS multiplication is an example of the operation. We assume that all round functions are identical. We also assume that whitening operations are not performed.

Hereafter we use the notations S_i and P_i for the standard Feistel to represent the state immediately after the S-layer and P-layer in round i, respectively. For 4-branch type-2 GFN, we use the notations S_i^L, S_i^R, P_i^L, and P_i^R to further distinguish the left and right round functions.

2.4 Domain Extension and Hashing Modes

Main targets of this paper are compression functions which are constructed by PGV modes with a Feistel-SP or GFN-SP cipher. For simplicity, we explain the attack on the compression function constructed by the Davies-Meyer mode [16, Algorithm 9.42] or MMO mode, in which the compression function output is computed by an XOR of plaintext and ciphertext.

Suppose that the compression function is constructed by the Davies-Meyer mode and the hash function is constructed by the narrow-pipe Merkle-Damgård domain extension. It is well known that a pseudo-preimage attack on the compression function with a complexity of 2^x can be converted to a preimage attack on the hash function with a complexity of $2^{((x+N)/2)+1}$ [16, Fact 9.99]. If the MMO mode is adopted, the attack is converted to a second preimage attack on the hash function with the same complexity.

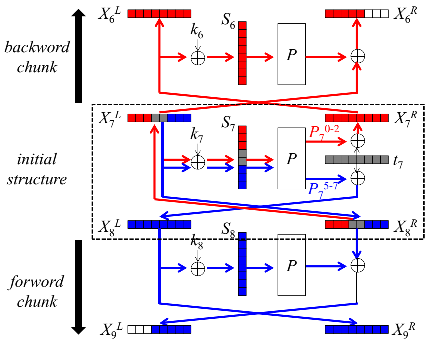

Fig. 2. 3 rounds including the 1-round initial structure

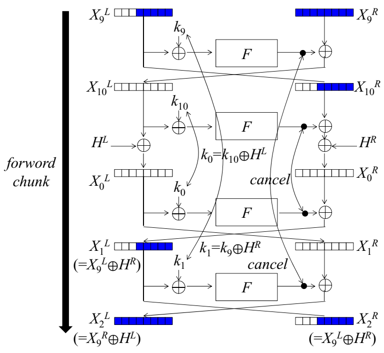

Fig. 3. 4-round shrink for generic Feistel

3 Preimage Attacks on Feistel-SP and GFN-SP Functions

3.1 Attacks on 11-round Feistel-SP Compression Function

The attack is a MitM attack with the splice-and-cut [3], initial structure [22], and indirect-partial matching [1]. This attack only can work if a condition on the key schedule function is satisfied. (Later we show the condition can be satisfied for 13-round Camellia-128.) The attacked rounds are from round 0 to round 10.

The attacker firstly searches for a key value satisfying the condition. Then, for the fixed key value, the MitM attack is performed. During the MitM attack, 4 rounds will be shrunken when we analyze the last and first rounds sequentially with the splice-and-cut technique. The valid pair can be identified by efficiently matching the results from two chunks with skipping several rounds.

1-round Initial Structure Plus 2 Rounds (Rounds 6 to 8). The attack starts from round 7 by constructing the initial structure. The detailed construction is given in Fig. 2. Throughout this paper, free bytes for the forward chunk and values depending of them are shown in blue, while free bytes for the backward chunk and values depending of them are shown in red. Grey bytes are fixed during the MitM attack. In the computation for each chunk, free bytes of the other chunk are regarded as unknown value which are shown with blank squares. Although one of the technical contributions of this paper is tracing linear relations of free bytes for the other chunk, this technique does not lead to any advantage for the case of a generic Feistel-SP. Hence, in this attack, to make the attack simple, we do not trace linear relations.

The computation for each chunk starts from choosing the value of the free bytes. The free bytes for the forward chunk and backward chunk are the last three bytes of X_7^L and the first three bytes of X_7^L, respectively. Because P is a linear operation, the impact from the free bytes for each chunk, denoted by P_7^{0-2} and P_7^{5-7} can be computed independently, *i.e.*, $P_7^{0-2} = P\big(S(X_7^L[0] \oplus$

$k_7[0])\|S(X_7^L[1]\oplus k_7[1])\|S(X_7^L[2]\oplus k_7[2])\|0\|0\|0\|0\|0\big)$ and $P_7^{5-7} = P\big(0\|0\|0\|0\|0\| S(X_7^L[5]\oplus k_7[5])\|S(X_7^L[6]\oplus k_7[6])\|S(X_7^L[7]\oplus k_7[7])\big)$. Therefore, two chunks can be computed independently. In Fig. 2, one round computation is added for both chunks after the 1-round initial structure.

4-round Shrink (Rounds 9, 10, 0, and 1). We continue the forward chunk after Fig. 2. The next 4 rounds are given in Fig. 3. The splice-and-cut technique is used, namely, after we obtain $X_{11}^L\|X_{11}^R$ we obtain the values of $X_0^L\|X_0^R$ by taking an XOR with the hash value denoted by $H^L\|H^R$. The analysis for these 4 rounds does not use the property of an SP-round function. Therefore, we describe the round function in a more generic form.

With a straight-forward method, the forward chunk cannot continue even two rounds because the unknown three bytes at X_9^L makes all bytes of X_{10}^L unknown and all bytes of X_{11}^R unknown. However, we observe that, with the help of the omission of the network twist after round 10, we can cancel the round function output in round 10, $F(X_{10}^L\oplus k_{10})$, by the one in round 0, $F(X_0^L\oplus k_0)$, with setting $k_0 = k_{10}\oplus H^L$. This is because $F(X_0^L\oplus k_0) = F((X_{10}^L\oplus H^L)\oplus(k_{10}\oplus H^L)) = F(X_{10}^L\oplus k_{10})$. Then, we can preserve the known bytes of X_9^L in X_1^L. Moreover, because X_1^L and X_9^L have the relation H^R, we can cancel the impact of round 9 with the one in round 1 by setting $k_1 = k_9\oplus H^R$.

In the end, after 4 rounds, X_2^L and X_2^R become $X_9^R\oplus H^L$ and $X_9^L\oplus H^R$, respectively. This is stronger than just skipping 4 rounds because the unknown bytes (X_9^L) are moved to the right half of the state (X_2^R) which is not used to update the next round.

We set two $N/2$-bit conditions on the key. If the output of the key schedule function is uniformly distributed, satisfying this condition will take the same cost as the brute-force preimage attack. However, satisfying this condition is often possible because the key schedule function is usually light. For example, if some bits of the secret key are used as subkeys *e.g.* DES [5] and XTEA [18], satisfying the condition is trivial. Moreover, in some ciphers, the last-round subkey can be directly generated from the secret key for achieving on-the-fly key generation for decryption, *e.g.* HIGHT [10]. In such a case, the condition is easily satisfied.

4-round Match (Rounds 2 to 5). The remaining 4 rounds are shown in Fig. 4. If we compute 2 rounds in backwards, all bytes become unknown due to the three unknown bytes of X_6^R. Hence, the direct match cannot be applied.

We observe that the computation over three rounds denoted by bold lines in Fig. 4 is linear. The equation is $S_5 \leftarrow P^{-1}(P(S_3)\oplus X_3^R\oplus X_6^L)$. By applying a linear transformation, this part can be converted into a partial match. A simplified description of these computations is given in Fig. 5. The transformed equation is $S_5 \leftarrow S_3\oplus P^{-1}(X_3^R)\oplus P^{-1}(X_6^L)$. Note that the attacker knows all values of X_3^R and X_6^L. Hence, $P^{-1}(X_3^R)$ and $P^{-1}(X_6^L)$ can be computed in each chunk independently of the other chunk. In more details, in the forward chunk, we compute $S_3\oplus P^{-1}(X_3^R)$ and store them in a table. In the backward chunk,

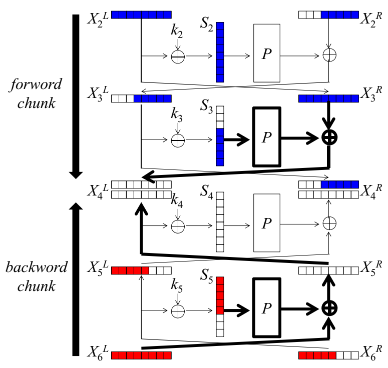

Fig. 4. The match over 4 rounds

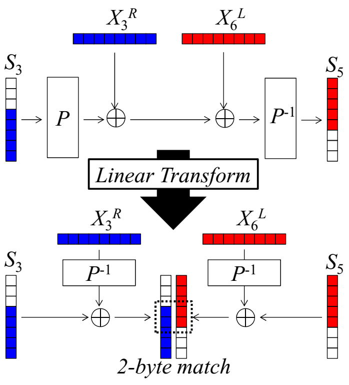

Fig. 5. Detailed matching procedure

we compute $S_5 \oplus P^{-1}(X_6^L)$ and check the match. Because 2 bytes are overlapped between these values, 2-byte match can be performed.

Attack Procedure. The attack procedure for a target $H^L \| H^R$ is as follows.

1. Find a key value K such that $k_0 = k_{10} \oplus H^L$ and $k_1 = k_9 \oplus H^R$ are satisfied.
2. For all choices of the ten fixed-byte values, $X_7^L[3,4]$ and t_7, do as follows.
3. Choose three free bytes for the forward chunk, $X_7^L[5,6,7]$, and compute the value of $S_3 \oplus P^{-1}(X_3^R)$. Store the results in a table.
4. Choose three free bytes for the backward chunk, $X_7^L[0,1,2]$, compute the value of $S_5 \oplus P^{-1}(X_6^L)$, and check if the same value exists in the table with respect to the 3rd and 4th bytes.
5. If the match is found, check the match of all bits with the corresponding free bytes for both chunks. If all bits match, output it as a pseudo-preimage.

The complexity for Step 1 depends on the key schedule function. Let T_{key} be the complexity of Step 1. Step 2 iterates the following steps 2^{80} times. For each value of Step 2, Step 3 is iterated 2^{24} times, and requires 2^{24} amount of memory. Step 4 is also iterated 2^{24} times. The sum of the complexities for Steps 3 and 4 is about 2^{24} 11-round compression function computations. Strictly speaking, the attacker does not have to compute the shrunken 4 rounds. Here, we ignore its impact. After Step 4, $2^{24+24-16} = 2^{32}$ values will remain. These values are examined in Step 5 that requires 2^{32} 11-round compression function computations. Finally, this 2^{32} computations are iterated by 2^{80} times due to Step 2, which results in 2^{112} 11-round compression function computations. Note that, for a fixed key, all output values of the compression function cannot be produced. Hence, the success probability of the attack is $1 - 1/e \approx 0.63$. Note that the attack can be iterated as long as several key values satisfying the conditions are available.

In summary, the total computational complexity is $T_{key} + 2^{112}$ computations and the memory requirement is 2^{24} internal-state values. Suppose that T_{key} is

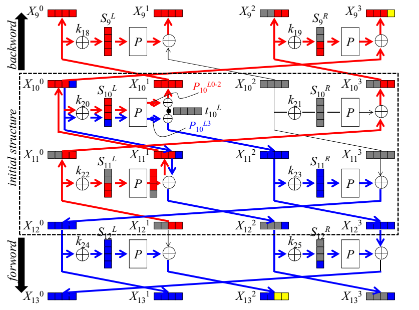
Fig. 6. 4 rounds with 2-round initial structure

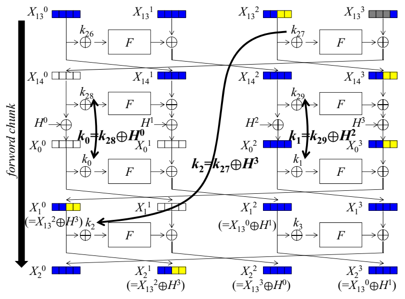
Fig. 7. 4-round shrink for generic GFN-SP

much smaller than 2^{112} 11-round computations. Then, the attack is converted to the preimage attack or the second preimage attack on the hash function with a complexity of $2^{((112+128)/2)+1} = 2^{121}$ computations.

Let us discuss the comparison with the case where the last network twist is not omitted. In this case, the cancellation property cannot be exploited in the MitM attack. We focus on the identical pattern of the known byte positions between (X_9^L, X_9^R) and (X_3^L, X_3^R). This indicates that if we remove rounds 9, 10, 0, 1, and 2, in total 5 rounds, the MitM attack can work. Hence, the number of attacked rounds is 6, which is significantly smaller than the case without the last network twist.

3.2 Attacks on 15-round Type-2 GFN-SP Compression Function

2-round Initial Structure Plus 2 Rounds (Rounds 9 to 12). The attack starts from round 10 by constructing a 2-round initial structure. The detailed construction is given in Fig. 6. Hereafter, yellow bytes represent the ones that are linearly dependent of free bytes for the other chunk.

Inside the 2-round initial structure, we need to ensure that the impact from two chunks do not mix. Here, we explain the computation of each chunk.

Forward Chunk: The free byte for the forward chunk (blue) is $X_{10}^0[3]$. In round 10, Because P is linear, the impact from $X_{10}^0[3]$ denoted by P_{10}^{L3} is independently computed from the backward chunk *i.e.*, $P_{10}^{L3} = P\big(0\|0\|0\| S(X_{10}^0[3] \oplus k_{20}[3])\big)$. In round 11, suppose that the value of P_{11}^{L3} is independent of the backward chunk. Then, the output of round 11 is computed by simply computing the round function.

Backward Chunk: The free bytes for the forward chunk (red) are $X_{12}^1[2]$ and $X_{12}^1[3]$. We only choose 1-byte (256) possibilities for these two bytes so that P_{11}^{L3} in round 11 can be a fixed value. In details, we first choose the value of $S_{11}^L[2]$ and then choose the corresponding $S_{11}^L[3]$ that makes P_{11}^{L3} be a predetermined fixed value. The value of $S_{11}^L[3]$ depends on the specification of P. In general, we can have unique candidate of $S_{11}^L[3]$ due to the linearity of P. After we choose $S^L_{11}[2], S^L_{11}[3]$, we compute $X^1_{12}[2], X^1_{12}[3]$ by XORing $k_{22}[2], k_{22}[3]$. In round 10, P^{L0-2}_{10} can be computed independently of the forward chunk as explained before. In the end, X^1_{10} can be computed independently of the forward chunk.

In Fig. 6, one round computation is added for both chunks after the 2-round initial structure. Note that $X^2_{13}[2,3]$ is linearly affected by the free bytes for the backward chunk, $X^1_{12}[2,3]$. Similarly, $X^3_9[3]$ is linearly affected by the free byte for the forward chunk, $X^0_{10}[3]$.

4-round Shrink (Rounds 13, 14, 0 to 2). 4 rounds after Fig. 6 is shown in Fig. 7. After we obtain $X^0_{15}\|\cdots\|X^3_{15}$, we obtain $X^0_0\|\cdots\|X^3_0$ by taking an XOR with the hash value $H^0\|\cdots\|H^3$. The analysis does not use the property of an SP-round function, thus the round function is described in a more generic form.

We observe that, with the help of the omission of the network twist after round 14, we can cancel the round function output in round 14, $F(X^0_{14} \oplus k_{28})$ and $F(X^2_{14} \oplus k_{29})$, by the ones in round 0, $F(X^0_0 \oplus k_0)$ and $F(X^2_0 \oplus k_1)$, with setting $k_0 = k_{28} \oplus H^0$ and $k_1 = k_{29} \oplus H^2$. Then, we can preserve the known bytes. Moreover, because X^0_1 and X^2_{13} have the relation H^3, we can cancel the impact of $F(X^2_{13} \oplus k_{27})$ in round 13 by setting $k_2 = k_{27} \oplus H^3$.

Similar to the attack on the Feistel network in Sect. 3.1, byte positions affected by the other chunk moved from the input side to the output side of the round function. This helps the attacker in subsequent rounds. Note that $X^1_2[2,3]$ is still only linearly affected by the free bytes for the backward chunk.

Different from the attack on a Feistel network, we only set three $N/4$-bit conditions on the subkeys. These $3N/4$-bit relations can be satisfied by the brute force search with a complexity of $2^{3N/4}$ key schedule function.

7-round Match (Rounds 3 to 9). The remaining 7 rounds are shown in Fig. 8. If we compute 3 rounds in backwards and 4 rounds in forwards, the direct match cannot be applied. We then use the linear computation over three rounds denoted by bold lines in Fig. 8. The equation is $S^R_8 \leftarrow p^{-1}(P(S^L_6) \oplus X^1_6 \oplus X^0_9)$. A simplified description is given in the top of Fig. 9. By applying a linear transformation, this part can be converted into a partial match. The transformed equation is $S^R_8 \leftarrow S^L_6 \oplus P^{-1}(X^1_6) \oplus P^{-1}(X^0_9)$. X^1_6 consists of the values dependent of the forward chunk and the free bytes for the backward chunk, X^1_{12}. This is shown in the middle of Fig. 9. Then, we further apply a transformation as the bottom of Fig. 9, and perform the 3-byte match. In more details, in the forward chunk, we compute $P^{-1}(X^1_6)$ and store them in a table. In the backward chunk, we compute $S^R_8 \oplus P^{-1}(S^0_9) \oplus S^L_6 \oplus P^{-1}(X^1_{12})$ and check the match. Because 3 bytes are overlapped, 3-byte match can be performed.

Attack Summary. Due to the limited space, if omit the detailed attack procedure. In summary, the total computational complexity is 2^{120} 15-round computations and 2^{96} key-schedule computations. The memory requirement is

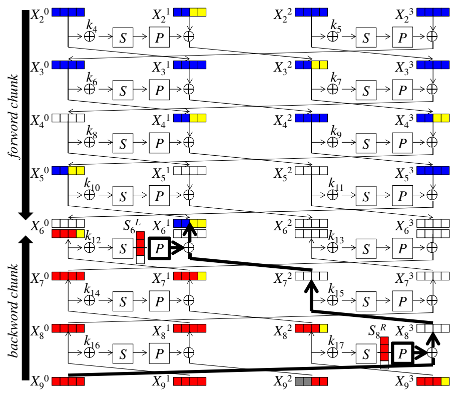

Fig. 8. 7 rounds including the matching procedure

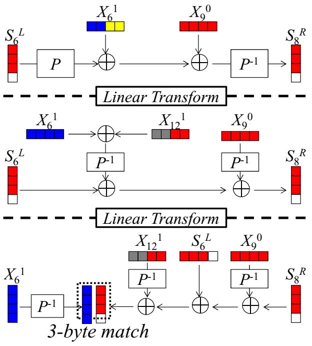

Fig. 9. Details of the matching procedure

2^8 internal state. The attack is converted to the preimage or the second preimage attack on the hash function with a complexity of $2^{((120+128)/2)+1} = 2^{125}$ computations. Similarly to Sect. 3.1, the success probability is 0.63. The attack can be iterated as long as several key values are available.

Let us discuss the comparison with the case where the last network twist is not omitted. Known byte positions between $(X^0_{14}, \ldots, X^3_{14})$ and $(X^0_4, \ldots, X^3_4)$ are identical. Therefore if we remove rounds 14, 0, 1, 2, and 3, in total 5 rounds, the MitM attack can work. Hence, the number of attacked rounds is 10, which is significantly smaller than the case without the last network twist.

4 Application to 13-round Weakened Camellia-128

We analyze hashing modes of Camellia-128 without the FL and whitening layers. Because the P-layer of Camellia does not satisfy the maximum branch number, the attack is extended by 2 rounds compared to the generic case.

2-round Initial Structure. 2-round initial structure can be constructed by exploiting a small branch number of the Camellia's P-layer. 4 rounds, from round 7 to 10, are shown in Fig. 10. The initial structure is located in round 8 and 9.

The forward chunk starts from a single free byte of $X^L_8[7]$. During round 8, it affects the single byte of $S_8[7]$. For the output value of P in rounds 8 and 9, the impact from the first 7 bytes denoted by P^{0-6}_8, P^{0-6}_9 can be independently computed of the impact from the 7th bytes denoted by P^7_8, P^7_9. $S_8[7]$ gives influence to 6 bytes of $X^L_8[0, 1, 2, 4, 5, 6]$. The important point here is that $X^L_9[7]$, which is later used as a free variable for the backward chunk, is not affected by $X^L_8[7]$. $X^L_8[7]$ also affects to a single byte of $X^R_9[7]$. We later show that this byte is not

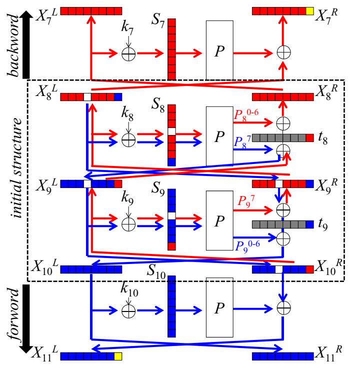

Fig. 10. Initial structure for Camellia

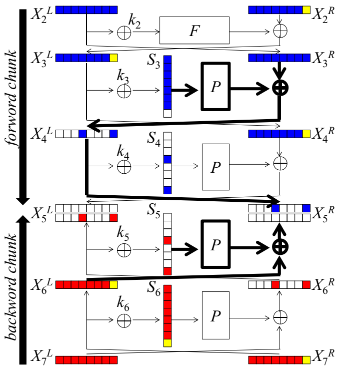

Fig. 11. Matching procedure for Camellia

affected by the free byte of the backward chunk. Computations during round 9 and round 10 are straight-forward.

The backward chunk starts from a single free byte of $X_{10}^R[7]$. During round 9, it affects the single byte of $S_9[7]$. Then, $S_9[7]$ gives influence to 6 bytes of $X_9^R[0,1,2,4,5,6]$. It surely does not affect to $X_9^R[7]$, which is the free byte for the forward chunk. $X_{10}^R[7]$ also affects to $X_9^L[7]$. As mentioned in the previous paragraph, $X_9^L[7]$ is not affected by the forward chunk. Thus, no contradiction occurs. Computations during round 8 and round 7 are straight-forward.

Matching Procedure. The 4-round shrink (round 11 to round 12 and round 0 to round 1) exploiting the omission of the last network twist is exactly the same as the attack on a generic case in Sect. 3.1. After 4 rounds, X_2^L and X_2^R become $X_{11}^R \oplus H^L$ and $X_{11}^L \oplus H^R$, respectively.

We match the results of two chunks in the remaining 5 rounds (round 2 to round 6). These rounds are described in Fig. 11. The form of the match is the same as Fig. 4, hence we omit the details. The equation for the match is written as $S_5 = S_3 \oplus P^{-1}(X_3^R) \oplus P^{-1}(X_6^L \oplus X_8^L[7])$, thus $S_5 \oplus P^{-1}(X_6^L) = S_3 \oplus P^{-1}(X_3^R \oplus X_8^L[7])$. 2 bytes of the left-hand-side and 7 bytes of the right-hand-side can be independently computed, and we can match 1 byte of them.

Analysis of the Key Schedule. For the 4-round shrink, we need to satisfy two conditions of subkeys; $k_0 = k_{12} \oplus H^L$ and $k_1 = k_{11} \oplus H^R$. According to the specification, $k_0 \| k_1 = K_A$, k_{11} is the right half of $(K_A \lll 60)$, and k_{12} is the left half of $(K \lll 94)$. Our strategy is choosing K_A so that the condition $k_1 = k_{11} \oplus H^R$ is deterministically satisfied, and satisfy $k_0 = k_{12} \oplus H^L$ with probability 2^{-64}. The details of the analysis is as follows. See its illustration in Fig. 12. The goal is finding 2^{64} 128-bit values K_A that satisfy $k_1 = k_{11} \oplus H^R$, where k_1 is the right half of K_A, and k_{11} is the right half of $(K_A \lll 60)$. For simplicity, we assume $H^R = 0$ in below. This is trivially extended for any H^R.

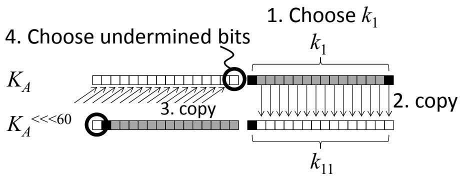

Fig. 12. Analysis of Camellia key schedule function. Each cell represents 4 bits.

1. Choose a 64-bit value of the right half of K_A so that the most significant 4 bits and the the least significant 4 bits are identical.
2. Copy the remaining 60 bits of k_1 to the corresponding bits of k_{11}. These also fix 60 bits of K_A.
3. The remaining 4 bits of K_A can be any value. In other words, we obtain 2^4 key values K_A that satisfy $k_1 = k_{11}$.
4. Finally, we have 60-bit choices for the value of the right half of K_A fixed at Step 1. Thus, we can find $2^4 \cdot 2^{60} = 2^{64}$ values of K_A that satisfy $k_1 = k_{11}$.

From 2^{64} values of K_A with $k_1 = k_{11} \oplus H^R$, we will find one that also satisfies $k_0 = k_{12} \oplus H^L$. Note that the success probability of the key search is 0.63, and we cannot expect more than 1 key.

Summary. We first search for a key value satisfying two conditions for the 4-round shrink. This is done with a complexity of 2^{64} key schedule function. We then start the MitM attack. Both of the forward and backward chunks include 1 free byte, and we match 1 byte of the results from two chunks. Hence, the pseudo-preimage is found faster than the brute force attack by a factor of 2^8, which is 2^{120} computations. The success probability is about $0.63^2 \approx 0.40$ due to the key search phase and the MitM phase. If it succeeds, the pseudo-preimage is converted to the preimage or the second preimage attack on a hash function with a complexity of 2^{125}.

5 Application to 12-round CLEFIA-128

Because F functions are different between the left half and the right half in CLEFIA, the number of attacked rounds is reduced by 3 compared to a generic case. Instead, conditions for subkeys is reduced from $3N/4$ to $N/4$ bits. Interestingly, the whitening operations do not impact to the attack very much.

2-round Initial Structure. The construction of the initial structure is basically the same as the one in Fig. 6. However, because the number of attacked rounds changes, we change the starting position of the backward chunk from the left half to the right half. We also increase the number of free bytes from 1 to 2. The detailed construction for 4 rounds, from round 6 to round 9, is shown in Fig. 13. Due to the similarity to Fig. 6, we omit the detailed explanation.

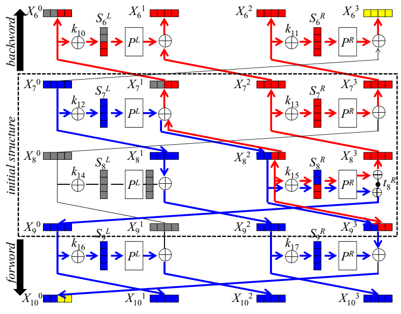

Fig. 13. 2-round initial structure for CLEFIA-128

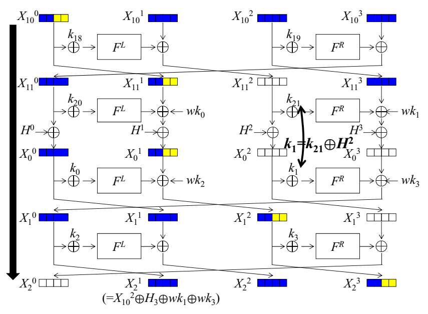

Fig. 14. 4-round shrink that requires an $n/4$-bit condition. The shrink only occurs in the same function.

4-round Shrink. The cancellation only occurs if the F function in consecutive two rounds are identical. Hence, we only make the cancellation between F^R in round 11 and F_R in round 0 by setting an $N/4$-bit condition $k_1 = k_{21} \oplus H^2$. This makes 32 bits of X_2^0 unknown, and the number of attacked rounds is reduced compared to a generic 4-branch type-2 GFN.

Matching Procedure. Again, the different F functions in the left half and the right half prevent the efficient matching over 2 P-layers. Hence, we use the indirect-partial matching technique [1], which enables us to match 4 bytes of the state. Due to the limited space, we omit the figure, but it can be derived in the same way as other attacks.

Summary. We first search for a key value satisfying $N/4$-bit condition for the 4-round shrink. This is done with a complexity of 2^{32} key schedule function. Note that several keys satisfying the condition can be generated by iterating the procedure. We then start the MitM attack. Both of the forward and backward chunks include 2 free bytes, and we match 4 bytes of the results from two chunks. Hence, the pseudo-preimage is found faster than the brute force attack by a factor of 2^{16}, which is 2^{112} computations. Because several keys are available, the success probability can become close to 1. Finally, the pseudo-preimage is converted to the preimage or the second preimage attack on a hash function with a complexity of 2^{121}.

6 Concluding Remarks

In this paper, we analyzed hash functions constructed by a generic Feistel and 4-branch type-2 GFN with an SP function. We showed that the omission of the last network twist can be utilized in the MitM preimage attack. Our attacks can work up to 11 rounds and 15 rounds for a Feistel-SP and 4-branch type-2 GFN-SP functions respectively under several conditions of the subkey relations. We then applied our attacks to hashing modes of Camellia-128 and CLEFIA-128.

References

1. Aoki, K., Guo, J., Matusiewicz, K., Sasaki, Y., Wang, L.: Preimages for step-reduced SHA-2. In: Matsui, M. (ed.) ASIACRYPT 2009. LNCS, vol. 5912, pp. 578–597. Springer, Heidelberg (2009)
2. Aoki, K., Ichikawa, T., Kanda, M., Matsui, M., Moriai, S., Nakajima, J., Tokita, T.: *Camellia*: A 128-Bit Block Cipher Suitable for Multiple Platforms - Design and Analysis. In: Stinson, D.R., Tavares, S. (eds.) SAC 2000. LNCS, vol. 2012, pp. 39–56. Springer, Heidelberg (2001)
3. Aoki, K., Sasaki, Y.: Preimage Attacks on One-Block MD4, 63-Step MD5 and More. In: Avanzi, R.M., Keliher, L., Sica, F. (eds.) SAC 2008. LNCS, vol. 5381, pp. 103–119. Springer, Heidelberg (2009)
4. Bogdanov, A., Shibutani, K.: Double SP-Functions: Enhanced Generalized Feistel Networks. In: Parampalli, U., Hawkes, P. (eds.) ACISP 2011. LNCS, vol. 6812, pp. 106–119. Springer, Heidelberg (2011)
5. Coppersmith, D.: The data encryption standard (DES) and its strength against attacks. IBM Journal of Research and Development 38(3), 243–250 (1994)
6. Cryptography Research and Evaluation Committees (CRYPTREC). e-Government recommended ciphers list (2003)
7. Daemen, J., Rijmen, V.: The design of Rijndeal: AES – the Advanced Encryption Standard (AES). Springer (2002)
8. Dunkelman, O., Keller, N.: The effects of the omission of last round's MixColumns on AES. Inf. Process. Lett. 110(8-9), 304–308 (2010)
9. Gauravaram, P., Leurent, G., Mendel, F., Naya-Plasencia, M., Peyrin, T., Rechberger, C., Schläffer, M.: Cryptanalysis of the 10-Round Hash and Full Compression Function of SHAvite-3-512. In: Bernstein, D.J., Lange, T. (eds.) AFRICACRYPT 2010. LNCS, vol. 6055, pp. 419–436. Springer, Heidelberg (2010)
10. Hong, D., Sung, J., Hong, S.H., Lim, J.-I., Lee, S.-J., Koo, B.-S., Lee, C.-H., Chang, D., Lee, J., Jeong, K., Kim, H., Kim, J.-S., Chee, S.: HIGHT: A New Block Cipher Suitable for Low-Resource Device. In: Goubin, L., Matsui, M. (eds.) CHES 2006. LNCS, vol. 4249, pp. 46–59. Springer, Heidelberg (2006)
11. International Organization for Standardization. ISO/IEC 10118-2:1994, Information technology – Security techniques – Hash-functions – Part 2: Hash-functions using an n-bit block cipher algorithm (2010)
12. ISO/IEC 18033-3:2010. Information technology–Security techniques–Encryption Algorithms–Part 3: Block ciphers (2010)
13. ISO/IEC 29192-2:2011. Information technology–Security techniques–Lightweight cryptography–Part 2: Block ciphers (2011)
14. Kang, H., Hong, D., Moon, D., Kwon, D., Sung, J., Hong, S.: Known-key attacks on generalized Feistel schemes with SP round function. IEICE Transactions 95-A(9), 1550–1560 (2012)
15. Knudsen, L.R., Rijmen, V.: Known-Key Distinguishers for Some Block Ciphers. In: Kurosawa, K. (ed.) ASIACRYPT 2007. LNCS, vol. 4833, pp. 315–324. Springer, Heidelberg (2007)

16. Menezes, A.J., van Oorschot, P.C., Vanstone, S.A.: Handbook of Applied Cryptography. CRC Press (1997)
17. Moon, D., Hong, D., Kwon, D., Hong, S.: Meet-in-the-Middle preimage attacks on hash modes of generalized Feistel and Misty schemes with SP round function. IEICE Transactions 95-A(8), 1379–1389 (2012)
18. Needham, R.M., Wheeler, D.J.: TEA extensions. Technical report, Computer Laboratory, University of Cambridge (October 1997)
19. New European Schemes for Signatures, Integrity, and Encryption(NESSIE). NESSIE PROJECT ANNOUNCES FINAL SELECTION OF CRYPTO ALGORITHMS (2003)
20. Preneel, B., Govaerts, R., Vandewalle, J.: Hash functions based on block ciphers: A synthetic approach. In: Stinson, D.R. (ed.) CRYPTO 1993. LNCS, vol. 773, pp. 368–378. Springer, Heidelberg (1994)
21. Sasaki, Y.: Meet-in-the-middle preimage attack on AES hashing modes and an application to Whirlpool. In: Joux, A. (ed.) FSE 2011. LNCS, vol. 6733, pp. 378–396. Springer, Heidelberg (2011)
22. Sasaki, Y., Aoki, K.: Finding preimages in full MD5 faster than exhaustive search. In: Joux, A. (ed.) EUROCRYPT 2009. LNCS, vol. 5479, pp. 134–152. Springer, Heidelberg (2009)
23. Sasaki, Y., Emami, S., Hong, D., Kumar, A.: Improved known-key distinguishers on Feistel-SP ciphers and application to Camellia. In: Susilo, W., Mu, Y., Seberry, J. (eds.) ACISP 2012. LNCS, vol. 7372, pp. 87–100. Springer, Heidelberg (2012)
24. Sasaki, Y., Yasuda, K.: Known-key distinguishers on 11-round Feistel and collision attacks on its hashing modes. In: Joux, A. (ed.) FSE 2011. LNCS, vol. 6733, pp. 397–415. Springer, Heidelberg (2011)
25. Shirai, T., Preneel, B.: On Feistel ciphers using optimal diffusion mappings across multiple rounds. In: Lee, P.J. (ed.) ASIACRYPT 2004. LNCS, vol. 3329, pp. 1–15. Springer, Heidelberg (2004)
26. Shirai, T., Shibutani, K.: Improving immunity of Feistel ciphers against differential cryptanalysis by using multiple MDS matrices. In: Roy, B., Meier, W. (eds.) FSE 2004. LNCS, vol. 3017, pp. 260–278. Springer, Heidelberg (2004)
27. Shirai, T., Shibutani, K., Akishita, T., Moriai, S., Iwata, T.: The 128-bit blockcipher CLEFIA (extended abstract). In: Biryukov, A. (ed.) FSE 2007. LNCS, vol. 4593, pp. 181–195. Springer, Heidelberg (2007)
28. U.S. Department of Commerce, National Institute of Standards and Technology. Specification for the ADVANCED ENCRYPTION STANDARD (AES) (Federal Information Processing Standards Publication 197) (2001)
29. Zheng, Y., Matsumoto, T., Imai, H.: On the construction of block ciphers provably secure and not relying on any unproved hypotheses. In: Brassard, G. (ed.) CRYPTO 1989. LNCS, vol. 435, pp. 461–480. Springer, Heidelberg (1990)