Improved Meet-in-the-Middle Distinguisher on Feistel Schemes

Li Lin[1,2] and Wenling Wu[1]

[1] Trusted Computing and Information Assurance Laboratory, Institute of Software, Chinese Academy of Sciences, Beijing 100190, China
[2] Graduate University of Chinese Academy of Sciences, Beijing 100190, China
{linli, wwl}@tca.iscas.ac.cn

Abstract. Improved meet-in-the-middle cryptanalysis with efficient tabulation technique has been shown to be a very powerful form of cryptanalysis against SPN block ciphers. However, few literatures show the effectiveness of this cryptanalysis against Balanced-Feistel-Networks (BFN) and Generalized-Feistel-Networks (GFN) ciphers due to the stagger of affected trail and special truncated differential trail. In this paper, we describe a versatile and powerful algorithm for searching the best improved meet-in-the-middle distinguisher with efficient tabulation technique on word-oriented BFN and GFN block ciphers, which is based on recursion and greedy algorithm. To demonstrate the usefulness of our approach, we show key recovery attacks on 14/16-round CLEFIA-192/256 which are the best attacks. We also propose key recovery attacks on 13/15-round Camellia-192/256 (without FL/FL^{-1}).

Keywords: Block Ciphers, Improved Meet-in-the-Middle Attack, Efficient Tabulation Technique, Automatic Search Tool, Truncated Differential Trail, CLEFIA, Camellia.

1 Introduction

The meet-in-the-middle attack was first proposed by Diffie and Hellman to attack DES [7]. In recent years, it was widely researched due to its effectiveness against block cipher AES [4]. For AES, Gilbert and Minier showed in [10] a collision attack on 7-round AES. At FSE 2008, Demirci and Selçuk improved the Gilbert and Minier attack using meet-in-the-middle technique instead of collision ideas [5]. At ASIACRYPT 2010, Dunkelman, Keller and Shamir improved Dermirci and Selçuk attacks on 7-round AES-192 and 8-round AES-256 by introducing the idea of *multiset* and differential enumeration technique in [8]. Further more, Derbez, Fouque and Jean presented a significant improvement of Dunkelman et al. attack at EUROCRYPT 2013 [6], called efficient tabulation technique. Using the rebound-like idea, they showed that many values in precomputation table are not reached at all under the constraint of the truncated differential trail. At FSE 2014, Li et al. proposed key-dependent sieve technique to attack 9-round AES-192 [13].

In [14], Lin et al. defined the *T-δ-set* which is a special set of states and the *S-multiset* which is a *multiset* of S cells. Then using these definitions, they got the least spread *T-δ-set* which has the minimum number of active cells and *$affected$-cell-set* which is affected by the minimun number of active cells[3]. After that, they introduced a general algorithm for searching the best affected trail from a *T-δ-set* to an *S-multiset*, and found that building a better affected trail is equivalent to a positive integer optimization problem.

Although pnew results on Substitution-Permutation Networks (SPN) block ciphers using improved meet-in-the-middle attack with efficient tabulation technique were given in many literatures, few results were proposed on Balanced-Feistel-Networks (BPN) [11] and Generalized-Feistel-Networks (GFN) [19]. This is due to the stagger of affected trail and truncated differential trail in the precomputation phase.

Our contribution. In this paper, we describe a versatile and powerful algorithm for searching the best improved meet-in-the-middle distinguisher with efficient tabulation technique on word-oriented BFN and GFN block ciphers. This algorithm is based on recursion and greedy algorithm. Given an affected trail $\mathcal{D}$ from a *T-δ-set* to an *S-multiset* by the algorithm of [14], our algorithm is made up of two phase: table construction phase and searching phase.

The table construction phase is based on the precomputation phase Fouque et al. proposed in [9]. In this phase, we build a graph G that contains all the 2-equipartite directed acyclic graph of all the possible one-round transitions.

The searching phase is based on the algorithm Matsui proposed to find the best differential characteristics for DES [18]. Our algorithm works by recursion and can be seen as a tree traversal in a depth-first manner. One truncated differential trail is a path in this tree, and its weight equals the product of all traversed edges. We are looking for the path with the minimum number of guessed-cells in this tree under certain transition probability. The knowledge of the previous best truncated differential trail allows pruning during the procedure. To speed up this algorithm, We also use greedy algorithm to divide the search into 2 parts.

To demonstrate the usefulness of our approach, we apply our algorithm to CLEFIA-192/256 [20] and Camellia-192/256 (without FL/FL^{-1})[4] [1]. For CLEFIA-256, we propose a 10-round distinguisher, then give a 16-round key recovery attack with data complexity of $2^{121.5}$ chosen-plaintexts, time complexity of $2^{203.5}$ encryptions and memory complexity of $2^{201.5}$ 128-bit blocks. To the best of our knowledge, this is currently the best attack with respect to the number of attacked rounds. For CLEFIA-192, we propose a 9-round distinguisher, then give a 14-round key recovery attack with data complexity of $2^{121.5}$ chosen-plaintexts, time complexity of $2^{139.5}$ encryptions and memory complexity of $2^{137.5}$ 128-bit blocks. To the best of our knowledge, this is currently the best attack with respect to the attack complexity.

[3] We call it the affected trail in this paper

[4] We call it Camellia* in this paper.

We also propose an 8-round distinguisher on Camellia*-192, then give a 13-round key recovery attack with data complexity of 2^{113} chosen-plaintexts, time complexity of 2^{180} encryptions and memory complexity of 2^{130} 128-bit blocks. For Camellia*-256, we propose a 9-round distinguisher, then give a 15-round key recovery attack with data complexity of 2^{113} chosen-plaintexts, time complexity of 2^{244} encryptions and memory complexity of 2^{194} 128-bit blocks. Although Lu et al. proposed a 14-round attack on Camellia*-192 and a 16-round attack on Camellia*-256 [17], they didn't consider the whitening operations. So we think our works on Camellia* are quite meaningful.

We present here a summary of our attack result on CLEFIA and Camellia*, and compare them to the best attacks known for them. This summary is given in Table 1.

Cipher	Attack type	Rounds	Data	Memory (Bytes)	Time (Euc)	Source
CLEFIA-192	Improbable	14	$2^{127.0}$ CPs	$2^{127.0}$	$2^{183.2}$	[21]
	Multidim. ZC	14	$2^{127.5}$ KPs	2^{115}	$2^{180.2}$	[2]
	Improved MITM	14	$2^{121.5}$ CPs	$2^{141.5}$	$2^{139.5}$	Sec. 4.1
CLEFIA-256	Improbable	15	$2^{127.4}$ CPs	$2^{127.4}$	$2^{247.5}$	[21]
	Multidim. ZC	15	$2^{127.5}$ KPs	2^{115}	$2^{244.2}$	[2]
	Improved MITM	16	$2^{121.5}$ CPs	$2^{205.5}$	$2^{203.5}$	Sec. 4.1
Camellia*-192	Imposible Diff	12	2^{119} CPs	2^{124}	$2^{147.3}$	[15]
	Imposible Diff	14^{ww}	2^{117} CPs	$2^{122.1}$	$2^{182.2}$	[16]
	HO MITM	14^{ww}	2^{118} CPs	2^{166}	$2^{164.6}$	[17]
	Improved MITM	13	2^{113} CPs	2^{134}	2^{180}	Sec. 4.2.1
Camellia*-256	Imposible Diff	15^{ww}	$2^{122.5}$ KPs	2^{233}	$2^{236.1}$	[3]
	Imposible Diff	16^{ww}	2^{123} KPs	2^{129}	2^{249}	[16]
	HO MITM	16^{ww}	2^{120} CPs	2^{230}	2^{252}	[17]
	Improved MITM	15	2^{113} CPs	2^{198}	2^{244}	Sec. 4.2.2

Table 1. Summary of the best attacks on CLEFIA-256, Camellia*-192/256. KPs: Known-Plaintexts. CPs: Chosen-Plaintexts. ww: Without Whiten Operation

The rest of this paper is organized as follows. Section 2 provides notations and definitions used throughout this paper, then gives a brief review of the algorithm to build the affected trail. Section 2 also gives the general attack scheme, and discusses the distinguisher on Feistel schemes with efficient tabulation technique. We provide the automatic search tool to search the best improved meet-in-the-middle distinguisher with efficient tabulation technique on Feistel schemes in Section 3. Our attacks on CLEFIA-192/256 and Camellia*-192/256 are described in Section 4. Finally, we conclude in Section 5.

2 Preliminaries

In this section, we give notations used throughout this paper, then briefly review the approach presented in [14], after that give the attack scheme. Finally, we discuss the improved meet-in-the-middle distinguisher on Feistel schemes with efficient tabulation technique.

2.1 Notation

The following notation will be used throughout this paper (the size is counted in number of cells):

- b: block size.
- k: the size of the master key.
- o: the size of one branch.
- r: number of rounds.
- c: number of bits in a cell.
- n: number of branches that will go through F-function in a state.
- $|T|$: number of active cells on a state T.
- x_i: the input state of round-i.
- y_i: the state after key addition layer of round-i.
- z_i: the state after S-box layer of round-i.
- w_i: the state after the linear transformation layer of round-i.
- $s[i]$: the i^{th} branch of a state.
- $s[i][j]$: the j^{th} cell in the i^{th} branch of a state.
- $RK_i[j]$: the j^{th} cell of the i^{th} round key.
- $\mathcal{G}(\mathcal{T})$: number of guessed-cells for a truncated differential trail $\mathcal{T}$.
- $\mathcal{P}(\mathcal{T})$: probability of a truncated differential trail $\mathcal{T}$.
- $s_1 \to s_2$: the probability that $s_1 \xrightarrow{\text{1-round transition}} s_2$ is greater than 0.

2.2 Reviews of Former Worksp

In [14], Lin et al. gave the algorithm to search the affected trail. In this subsection, we briefly review the definitions and algorithms they gave.

The improved meet-in-the-middle distinguisher is based on particular structures of messages captured by Definition 1 and 2.

Definition 1 (*T-δ-set*)**.** *T-δ-set is a set of $2^{T\times c}$ states that are all different in T cells (the active cells) and all equal in the other cells (the inactive cells), where c is the number of bits in a cell and $T \leq b$.*

Definition 2 (*S-Multiset*)**.** *S-multiset is an unordered set in which elements can occur many times, and every element of the S-multiset consists of S cells, $S \leq b$.*

In this paper, T presents T-δ-*set* and S presents S-$Multiset$. With these two definitions in mind, we canp choose proper values of $|T|$ and $|S|$ [14].

The searching algorithm is based on a propagation-then-pruning method as shown in Fig. 1. Suppose we have one (T, S) pair, the building of an affected trail $\mathcal{D}$ is as follows:

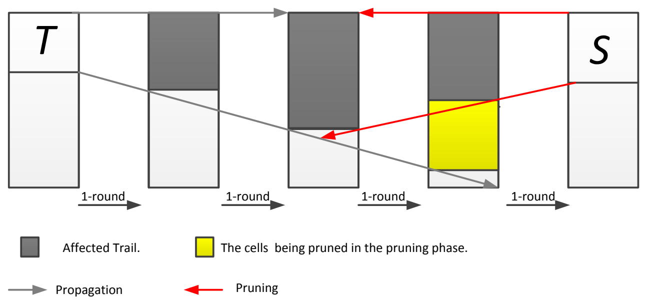


Fig. 1. 4-round example of propagation-then-pruning method.

1. **Propagation.** In the forward direction, differences of T can propagate from one round to the next. We need to guess the active cells going through the S-box layer, then get S by this trail.
2. **Pruning.** In this trail, some guessed-cells have nothing to do with the building of S. These cells are pruned from the trail.

Using this algorithm, we can get an affected trail $\mathcal{D}$ for (T, S).

2.3 Attack Scheme

In this subsection, we present our new unified view of the single-key meet-in-the-middle attack, where R rounds of the block cipher can be split into three consecutive parts of r_1, r, and r_2.

The general attack uses two successive phases as shown in Fig. 2.

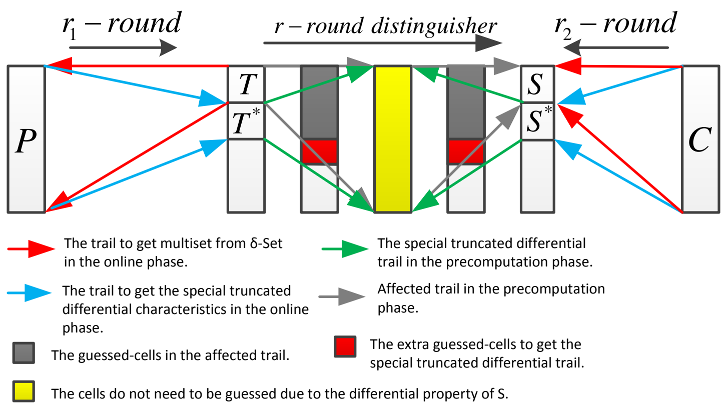

$r_1 - round$
$r - round$ $distinguisher$
$r_2 - round$
P
T
T^*
S
S^*
C
The trail to get multiset from δ-Set in the online phase.
The special truncated differential trail in the precomputation phase.
The trail to get the special truncated differential characteristics in the online phase.
Affected trail in the precomputation phase.
The guessed-cells in the affected trail.
The extra guessed-cells to get the special truncated differential trail.
The cells do not need to be guessed due to the differential property of S.

Fig. 2. General scheme of the improved meet-in-the-middle attack with efficient tabulation technique.

Precomputation phase

1. In the precomputation phase, we build an affected trail from T to S by guessing some cells.
2. Use efficient tabulation technique to build a special truncated differential trail from T^* to one middle round, then build a truncated differential trail from S^* to another middle round in the reverse direction. This step needs to guess some more cells. With the differential property of S [6], we can prune some guessed-cells made in step 1.
3. Build a lookup table L containing all the possible trail constructed from T to S.

Online phase

1. In the online phase, we need to identify a T-δ-set containing a message m verifying the desired property. This is done by using a large number of plaintexts and ciphertexts, and expecting that for each key candidate, there is one pair of plaintexts satisfying the truncated differential trail from $P \to T^*$ and $C \to S^*$. m is one member of this plaintext pair. Then use m to build a T-δ-set.
2. Finally, we partially decrypt the associated T-δ-set through the last r_2 rounds and check whether it belongs to L.

2.4 Feistel Ciphers with Efficient Tabulation Technique

In this paper, we focus on the application of efficient tabulation technique on BFN and GFN. The round functions F of them arep made up of 3 layers: key addition layer, the S-box layer and the linear transformation layer. This is true for most of BFN and GFN ciphers.

The advantage of improved meet-in-the-middle attack with efficient tabulation technique on Feistel ciphers is that it can use the differential property of S in $\lfloor \frac{b}{n \times o} \rfloor$ rounds. We take CLEFIA [20] as an example. CLEFIA is a 4-branch type-2 GFN cipher with $b = 16$, $n = 2$ and $o = 4$.

As shown in Fig. 3 (A), $\Delta y_i[0] = \Delta x_i[0]$, $\Delta z_i[0] = M^{-1}(\Delta x_i[1] \oplus \Delta x_{i+2}[3])$. If we know Δx_i and Δx_{i+2}, using the differential property of S, we can get the active bytes of $y_i[0]$. Using the same method, if we know Δx_i and Δx_{i+2}, we can get the active bytes of $y_i[2]$, $y_{i+1}[0]$ and $y_{i+1}[2]$ as well. So if we get the special truncated differential trail, we can prune some guessed-cells.

However, things are just not as what we think. For SPN ciphers, affected trail and truncated differential trail are almost the same. But for BFN and GFN ciphers, they differ a lot from each other. We also take CLEFIA as an example. As shown in Fig. 3 (B), the guessed-cells of affected trail and truncated differential trail are totally different. For dependent trail, we only need to guess $y_i[2]$. For truncated differential trail, we need to guess $y_{i+1}[2][0]$ and $y_i[0]$.

To solve this problem, we present an automatic search tool in Section 3 to search the best improved meet-in-the-middle distinguisher with efficient tabulation technique on GFN and BFN.

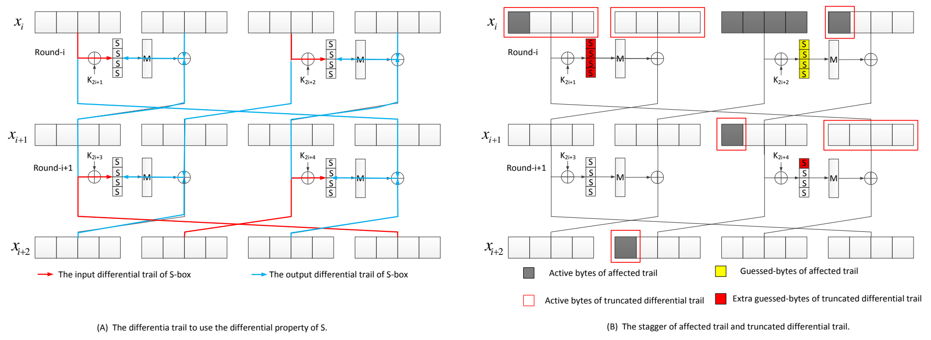

x_i
Round-i
K_{2i+1}
K_{2i+2}
x_{i+1}
Round-i+1
K_{2i+3}
K_{2i+4}
x_{i+2}
S
M
The input differential trail of S-box
The output differential trail of S-box
(A) The differentia trail to use the differential property of S.
Active bytes of affected trail
Guessed-bytes of affected trail
Active bytes of truncated differential trail
Extra guessed-bytes of truncated differential trail
(B) The stagger of affected trail and truncated differential trail.

Fig. 3. (A) The differential trail with the differential property of S; (B) The stagger of an affected trail and a truncated differential trail.

3 The Automatic Search Tool using Efficient Tabulation Technique

In this section, we present a practical algorithm for deriving the best improved meet-in-the-middle distinguisher on Feistel schemes in terms of efficient tabulation technique, which combines the precomputation phase of [9] and the search procedure of [18].

Suppose we have the affected trail Section 2.2 gives, our tool works by recursion and consists on 2 phases: table construction phase and searching phase.

3.1 Table Construction Phase

As shown in [9], the table construction phase builds a graph G that contains all the 2-equipartite directed acyclic graph of all the possible one-round transitions. This graph can be built and stored efficiently by observing its inner structure: the block cipher internal state output depends only on the block cipher internal state input. Unlike [9], since we don't consider the key schedule, the graph is small and we can store it in truncated differential characteristic[5]. A toy example of G is shown in Fig. 4.

We should build a graph G^{-1} that contains all the 2-equipartite directed acyclic graph of all the possible one-round transitions in the backward direction as well.

3.2 Searching Phase

As the algorithm in [18], our searching phase find the best n-round improved meet-in-the-middle distinguisher. The algorithm works by recursion and can be seen as a tree traversal in a depth-first manner, where the tree presents all

[5] The memory cost of CLEFIA stored in truncated differences is less than 20 MB.

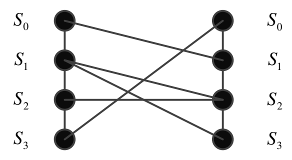


Fig. 4. Example of graph product to build G, with 4 possible internal state s_0, s_1, s_2 and s_3. There is an edge from s_i to s_j if and only if $s_i \to s_j$ after one round encryption.

the possible truncated differential trail in the cipher layered by round. The nodes present the truncated differential characteristics and the edges the possible transitions between them, and are labeled by their numbers of guessed-cells and transition probabilities. One truncated differential trail is a path in this tree, and its weight equals the product of all traversed edges. We are looking for the path with the minimum number of guessed-cells in this tree under certain transition probability. The knowledge of the previous best truncated differential trail allows pruning during the procedure. Since we only consider truncated differential characteristic and the pruning is very efficient, the running time will not be so long.

As analyzed in Section 2.4, the advantage of efficient tabulation technique on Feistel schemes is that it can guess less cells in the middle $\lfloor \frac{b}{n \times o} \rfloor$ rounds using a special truncated differential trail. However, since the stagger of affected trail and truncated differential trail, we can only limit parts of values in the affected trail.

3.2.1 Trail Probability. Almost all the truncated differential trails used in the distinguisher on SPN ciphers are with probability 1. In our algorithm, we consider the truncated differential trail with probability less than 1, which means we can guess less extra cells. Suppose we guess one less cell in the backward direction (with probability 2^{-c}), it may cause less extra cells to be guessed. Getting a trail with probability 2^{-c} means that we should repeat the online phase 2^c times. So our algorithm require an "initial value" for the minimum trail probability, which is presented as $\overline{P}$. This value can be determined by analyzing the online phase.

3.2.2 Comparing Trails. Next we introduce a (quasi-)order relation for 2 truncated differential trail $\mathcal{T}_1$ and $\mathcal{T}_2$ as follows:

Definition 3 ($\succ$). *$\mathcal{T}_1$ is better than $\mathcal{T}_2$ if and only if it has less guessed-cells or its probability is higher with the same number of guessed-cells, i.e.*

$$\mathcal{T}_1 \succ \mathcal{T}_2 \Leftrightarrow \begin{cases} \mathcal{G}(\mathcal{T}_1) < \mathcal{G}(\mathcal{T}_2) \\ or \\ \mathcal{G}(\mathcal{T}_1) = \mathcal{G}(\mathcal{T}_2) \text{ and } \mathcal{P}(\mathcal{T}_1) > \mathcal{P}(\mathcal{T}_2) \end{cases} \quad (1)$$

Also $\mathcal{T}_1 \equiv \mathcal{T}_2 \Leftrightarrow \mathcal{G}(\mathcal{T}_1) = \mathcal{G}(\mathcal{T}_2)$ and $\mathcal{P}(\mathcal{T}_1) = \mathcal{P}(\mathcal{T}_2)$[6].

3.2.3 Ending Condition. Given key length k, probability lower bound $\overline{P}$ and the best trail so far $\mathcal{T}_{best}$, we define ending condition $\mathcal{E}$ as follows:

$$\mathcal{T} \in \mathcal{E} \Leftrightarrow \begin{cases} \mathcal{P}(\mathcal{T}) \leq \overline{P} \\ \text{or } \mathcal{G}(\mathcal{T}) \geq k \\ \text{or } \mathcal{T}_{best} \succ \mathcal{T} \end{cases} \tag{2}$$

If a trail belongs to $\mathcal{E}$, we should stop the search procedure and try another trail.

3.2.4 Finding the Best Trail. Although we could test all the truncated differential trail under the probability lower bound $\overline{P}$ to find the best one, we have a more efficient way using the greedy algorithm. Since the affected trail is unique for each (S, T) pair, we can find out $\lfloor \frac{b}{n \times o} \rfloor$ successive rounds which need to guess more cells than others. This means that we can use the differential property of S to prune more guessed-cells. If the number of these successive rounds is more than one, we present the beginning of these rounds as a set called SR**-set**.

With SR-set in mind, we can divide the search procedure into 2 parts: one starts at the first round from the forward direction, the other starts at the last round from the backward direction. This algorithm will divide an r-round truncated differential trail search into 2 parts of r_1 and r_2, where $r = r_1 + r_2 + \lfloor \frac{b}{n \times o} \rfloor$.

At the beginning of the algorithm, we should decrease $\mathcal{G}(\mathcal{D})$ by the number of guessed-cells in the middle $\lfloor \frac{b}{n \times o} \rfloor$ rounds. Then at the end of the search for one trail, increase the number of guessed-cells that the truncated differential trail can't prune in these $\lfloor \frac{b}{n \times o} \rfloor$ rounds . After that, we could get the exact number of guessed-cells in this trail.

Although the first and the last truncated differential characteristics in the trail can take any values, we put some limitations on them by some observations. Since there is little difference between an affected trail and a truncated differential trail on the first r_1-round, we fix the first truncated differential characteristic to T. And by the propagation of differences, we constrain values of the last truncated differential characteristics S^* with $|S^*| \leq |S|$.

The framework of our algorithm for the first r_1 and the last r_2 rounds is now established by Algorithm 1 and 2 including essentially recursive calls.

The inputs of Algorithm 1 and 2 are affected trail $\mathcal{D}$, input truncated differential trail $\mathcal{T}$, best truncated differential trail $\mathcal{T}_{best}$, probability lower bound $\overline{P}$ and graph G/G^{-1}.

The sorting algorithm of line 3 in Algorithm 1 and Algorithm 2 is according to $\succ$.

[6] $\mathcal{G}$ means the total number of guessed-cells including the affected trail.

Algorithm 1 Search the first r_1 rounds

```
1: function Procedure $Round_i^{begin}(\mathcal{D}, \mathcal{T}, \mathcal{T}_{best}, \overline{P})$
2:     Find all truncated values this trail can lead to in graph $G$
3:     Sort these truncated differential characteristics
4:     for all truncated differential characteristics do
5:         Add this characteristic to $\mathcal{T}$
6:         if $\mathcal{T} \notin \mathcal{E}$ then
7:             if $i < r_1 - 1$ then
8:                 Call Procedure $Round_{i+1}^{begin}$
9:             else
10:                for all truncated differences $S^*$ satisfying $|S^*| \leq |S|$ do
11:                    Call Procedure $Round_0^{end}$ with $S^*$
12:                end for
13:            end if
14:        end if
15:    end for
16: end function
```

Line 10 of Algorithm 2 means $s_{r_1} \xrightarrow{\lfloor \frac{b}{n \times o} \rfloor rounds} s_{r_2}$, where s_{r_1} and s_{r_2} are truncated differential characteristics of round-r_1 and round-$(r_1 + \lfloor \frac{b}{n \times o} \rfloor)$ in the trail, respectively.

Line 11 of Algorithm 2 means we should increase $\mathcal{G}(\mathcal{T})$ by the guessed-cells it can't prune in the middle $\lfloor \frac{b}{n \times o} \rfloor$ rounds.

Algorithm 3 present the searching algorithm for a (T, S) pair.

We can loop through all possible (T, S) pairs to find the best r-round distinguisher under $\overline{P}$, then find $r_{max} = max\{r | \mathcal{P}(\mathcal{T}_{best}) > \overline{P}$ and $\mathcal{G}(\mathcal{T}_{best}) < k\}$.

4 Applications

In this section, we propose our attacks on CLEFIA-192/256 and Camellia*-192/256.

4.1 Applications to CLEFIA-192/256

CLEFIA is a lightweight 128-bit block cipher designed by Shirai et al. in 2007 [20] and based in a 4-branch type-2 GFN. It is adopted as an international ISO/IEC 29192 standard in lightweight cryptography. We refer to [20] for a detail description.

4.1.1 9/10-Round Distinguisher on CLEFIA. First, we use our search tool to find the best 10-round distinguisher on CLEFIA-256 and 9-round distinguisher on CLEFIA-192, they are shown in Fig. 5 and Fig. 6.

Algorithm 2 Search the last r_2 rounds

```
1: function Procedure $Round_i^{end}(\mathcal{D}, \mathcal{T}, \mathcal{T}_{best}, \overline{P})$
2:     Find all truncated values this trail can lead to in graph $G^{-1}$
3:     Sort these truncated differential characteristics
4:     for all truncated differential characteristics do
5:         Add this characteristic to $\mathcal{T}$
6:         if $\mathcal{T} \notin \mathcal{E}$ then
7:             if $i < r_2 - 1$ then
8:                 Call Procedure $Round_{i+1}^{end}$
9:             else
10:                if Combining 2 parts of $\mathcal{T}$ together leads to a trail then
11:                    Increase $\mathcal{G}(\mathcal{T})$
12:                    if $\mathcal{T} \notin \mathcal{E}$ then
13:                        $\mathcal{T}_{best} \leftarrow \mathcal{T}$
14:                    end if
15:                end if
16:            end if
17:        end if
18:    end for
19: end function
```

In the attack of CLEFIA, we apply an equivalent transformation to the 10-round and 9-round distinguishers, as shown in Fig. 5 and Fig. 6. Namely, the right linear transformations of round $i+8$ and $i+7$ are removed from these rounds, and linear transformations are added to three different positions in order to obtain distinguishers that are computationally equivalent to the original one.

The 10-round distinguisher on CLEFIA-256 is based on the proposition below.

Proposition 1. *Considering to encrypt 2^8 values of the (1-)δ-set through 10-round CLEFIA-256 starting from round-i, where $x_i[1][0]$ is the active byte, in the case of that a message of the δ-set belongs to a pair which conforms to the*

Algorithm 3 Search an optimal trail for (T, S)

```
1: function Procedure $SerachingTrail(\mathcal{D}, \overline{P}, T, S, \lfloor \frac{b}{n \times o} \rfloor)$
2:     Initial $\mathcal{T}_{best}$ with $k$ and $\overline{P}$
3:     Get the $SR$-set from $\mathcal{D}$
4:     for all $r_1$ in $SR$-set do
5:         Decrease $\mathcal{G}(\mathcal{D})$ by the number of guessed-cells in these $\lfloor \frac{b}{n \times o} \rfloor$ rounds
6:         Call $Round_0^{begin}$ with $r_1$ and $T$
7:     end for
8: return $\mathcal{T}_{best}$
9: end function
```

truncated differential trail outlined in Fig. 5, then the corresponding (1-)multisets of $x_{i+10}[1][0]$ only take about 2^{208} values.

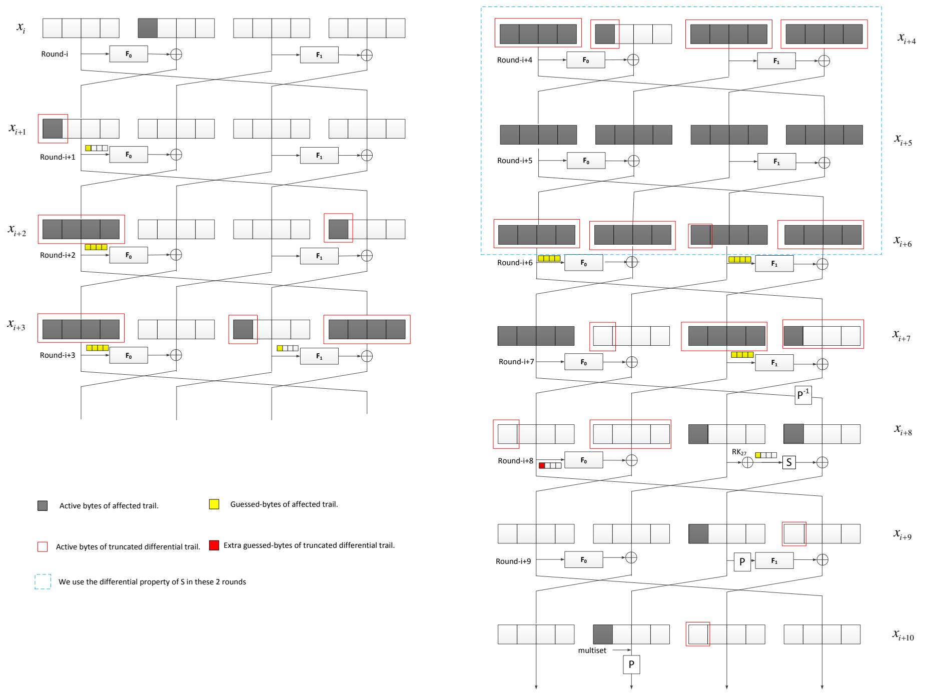


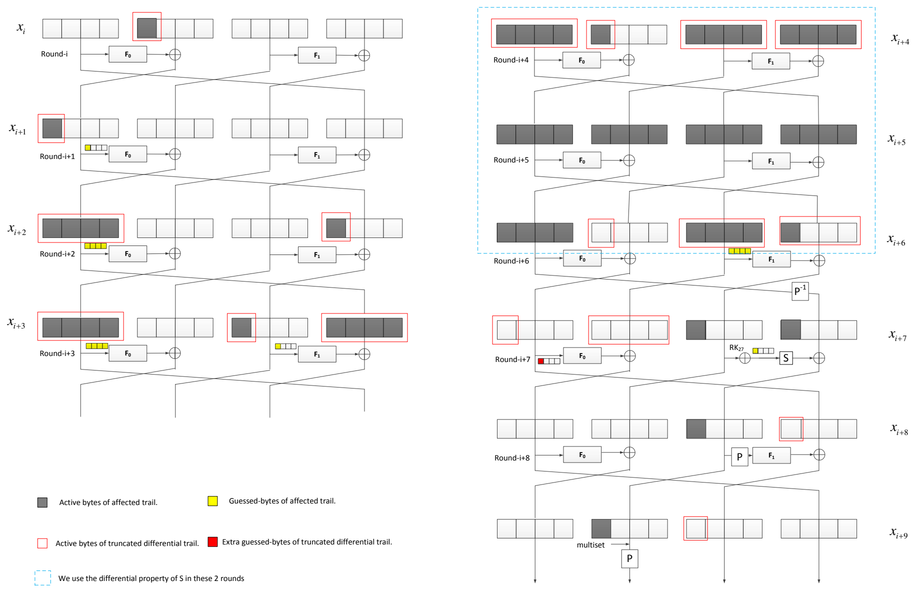
Fig. 5. 10-Round Distinguisher on CLEFIA-256

Proof. As shown in Fig. 5, we first consider the affected trail from $x_i[1][0]$ to $x_{i+10}[1][0]$. This affected trail is determined by 39-byte intermediate variable: $y_{i+1}[0][0]$, $y_{i+2}[0]$, $y_{i+3}[0]$, $y_{i+3}[2][0]$, $y_{i+4}[0]$, $y_{i+4}[2]$, $y_{i+5}[0]$, $y_{i+5}[2]$, $y_{i+6}[0]$, $y_{i+6}[2]$, $y_{i+7}[2]$, $y_{i+8}[2][0]$.

This can be easily seen from the figure.

Furthermore, if there exists a message of the (1-)δ-set belongs to a pair which conforms the truncated differential trail as in Fig. 5, the 35-byte variable $y_{i+1}[0][0]$, $y_{i+2}[0]$, $y_{i+3}[0]$, $y_{i+3}[2][0]$, $y_{i+4}[0]$, $y_{i+4}[2]$, $y_{i+5}[0]$, $y_{i+5}[2]$, $y_{i+6}[0]$, $y_{i+6}[2][0]$, $y_{i+7}[2]$ is determined by 22-byte variable: $\Delta x_i[1][0]$, $y_{i+1}[0][0]$, $y_{i+2}[0]$, $y_{i+3}[0]$, $y_{i+3}[2][0]$, $y_{i+6}[0]$, $y_{i+6}[2][0]$, $y_{i+7}[2]$, $y_{i+8}[0][0]$, $\Delta x_{i+10}[2][0]$.

Using 11-byte variable $\Delta x_i[1][0]$, $y_{i+1}[0][0]$, $y_{i+2}[0]$, $y_{i+3}[0]$, $y_{i+3}[2][0]$, we can deduce Δx_{i+4}. In the backward direction, using $\Delta x_{i+10}[2][0]$, $y_{i+8}[0][0]$,, $y_{i+7}[2]$, $y_{i+6}[0]$, $y_{i+6}[2][0]$,we can deduce Δx_{i+6}. By the differential property of S, this can deduce $y_{i+4}[0]$, $y_{i+4}[2]$, $y_{i+5}[0]$, $y_{i+5}[2]$.



x_i
Round-i
x_{i+1}
Round-i+1
x_{i+2}
Round-i+2
x_{i+3}
Round-i+3
x_{i+4}
Round-i+4
x_{i+5}
Round-i+5
x_{i+6}
Round-i+6
x_{i+7}
Round-i+7
x_{i+8}
Round-i+8
x_{i+9}
F_0
F_1
P^{-1}
RK_{27}
S
P
multiset
Active bytes of affected trail.
Guessed-bytes of affected trail.
Active bytes of truncated differential trail.
Extra guessed-bytes of truncated differential trail.
We use the differential property of S in these 2 rounds

Fig. 6. 9-Round Distinguisher on CLEFIA-192

In conclusion, the corresponding (1-)multisets of byte $x_{i+10}[1][0]$ only take about 2^{208} values with the truncated differential trail.

□

The 9-round distinguisher on CLEFIA-192 is based on the proposition below.

Proposition 2. *Considering to encrypt 2^8 values of the (1-)δ-set through 9-round CLEFIA-192 starting from round-i, where $x_i[1][0]$ is the active byte, in the case of that a message of the δ-set belongs to a pair which conforms to the truncated differential trail outlined in Fig. 6, then the corresponding (1-)multisets of $x_{i+9}[1][0]$ only take about 2^{144} values.*

The proof of this proposition is the same as before and shown in Fig. 6.

4.1.2 16/14-Round Attack on CLEIFA-256/192. Based on the 10-round distinguisher, we extend 3 rounds on the top and 3 rounds on the bottom to present the 16-round improved meet-in-the-middle attack on CLEFIA-256, and based on the 9-round distinguisher, we extend 3 rounds on the top and 2 rounds on the bottom to present the 14-round improved meet-in-the-middle attack on CLEFIA-192. The procedure of this attack is shown in Fig. 7 and Fig. 8.

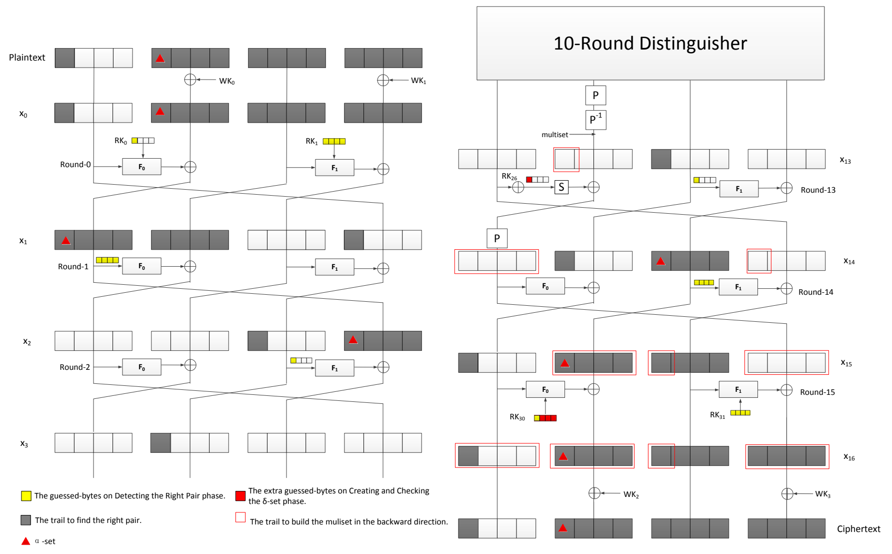
The linear transformation in round 13/12 is moved to two different positions as shown in Fig. 7 and 9. The newly-introduced P^{-1} and the P before $x_{13}[2]/x_{12}[2]$ generated by the distinguisher cancel each other, we therefore ignore them.

The following proposition is important in finding the special truncated differential trail.

Proposition 3. *If $MC(\Delta s_0) = (a_0, 0, 0, 0)$ and $MC(\Delta s_1) = (a_1, 0, 0, 0)$, then $MC(\Delta s_0 \oplus \Delta s_1) = (a_2, 0, 0, 0)$, where a_0, a_1, a_2 are any bytes*[7].

This is because the linearity of MC. We call the set of 2^8 differences that results in $(a, 0, 0, 0)$ after linear transformation layer the α-set which is marked by red triangle in Fig. 9.

The detail attack is shown below:



10-Round Distinguisher
Plaintext
WK₀
WK₁
x₀
RK₀
RK₁
Round-0
F₀
F₁
x₁
Round-1
x₂
Round-2
x₃
P
P⁻¹
multiset
x₁₃
RK₂₆
S
Round-13
x₁₄
Round-14
x₁₅
Round-15
RK₃₀
RK₃₁
x₁₆
WK₂
WK₃
Ciphertext
The guessed-bytes on Detecting the Right Pair phase.
The extra guessed-bytes on Creating and Checking the δ-set phase.
The trail to find the right pair.
The trail to build the mulisết in the backward direction.
α -set

Fig. 7. 16-Round Attack on CLEFIA-256

1. **Precomputation phase.** In the precomputation phase of the attack, we build the lookup table L that contains the 2^{208} multisets for difference $\Delta x_{13}[1][0]$ by following the method of Proposition 2.
2. **Online Phase.**
   (a) **Detecting the Right Pair:**
      i. p We prepare a structure of 2^{80} plaintexts where $\Delta P[0][0]$, $\Delta P[2]$ and $\Delta P[3]$ take all 2^{72} values and $\Delta P[1]$ takes all the value of the α-set. Hence, we can generate $2^{80} \times (2^{80} - 1)/2 \approx 2^{159}$ pairs satisfying the plaintext difference. Choose 2^{33} structures and get the corresponding ciphertexts. Among the $2^{159+33} = 2^{192}$ corresponding ciphertext pairs, we expect $2^{192} \times 2^{-48} = 2^{144}$ pairs to verify the

[7] MC denotes the linear transformation layer.

truncated difference pattern where $\Delta C[0][0]$, $\Delta C[2]$, $\Delta C[3]$ have differences, and $\Delta C[1]$ has difference in α-set. Store the 2^{144} remaining pairs in a hash table. This step require 2^{113} plaintext and ciphertext pairs.

ii. Guess the values of $RK_0[0]$, RK_1, $y_1[0]$, $y_2[2][0]$, using the guessed values to encrypt the remaining pairs to x_3. We choose the pairs that have difference only in byte $x_3[1][0]$, there are $2^{144-72} = 2^{72}$ pairs left.

iii. Guess the values of $RK_{30}[0]$, RK_{31}, $y_{14}[2]$ and $y_{13}[2][0]$, using the guessed values to decrypt the remaining pairs to x_{13}. We choose the pairs that have difference only in byte $x_{13}[2][0]$, there are $2^{72-72} = 1$ pair left.

(b) **Creating and Checking the** ***Multiset*****:**

i. For each guess of the 10 bytes made in Phase (a), decrypt the 2^8 possible differences in Δx_3 to Δx_0. Then XOR it with one plaintext P_0 of the pair.

ii. Using P_0 as the standard plaintext, denote the other 255 plaintexts as P_1 to P_{255}, and the corresponding ciphertexts as C_0 to C_{255}. Partially decrypt the ciphertexts to x_{13}, then get the multiset for difference $\Delta x_{13}[1][0]$ by guessing $RK_{30}[1, 2, 3]$, $y_{13}[0][0]$, and using the knowledge $RK_{30}[0]$, RK_{31}, $y_{14}[2]$.

iii. Checking whether the multiset exists in L. If not, discard the key guess. The probability for a wrong guess to pass this test is smaller than $2^{192}2^{-506.17} = 2^{-314.17}$.

(c) **Searching the Rest of Key:** For each remaining key guess, find the remaining key bytes by exhaustive search.

Complexity. The look up table of the 2^{208} possible multisets requires about 2^{210} 128-bit blocks to be stored [6]. To construct the table, we have to perform 2^{208} partial encryptions on 256 messages, which we estimate to be equivalent to 2^{212} encryptions.

In the online phase, first, we ask the encryption of 2^{113} chosen-plaintexts, so the time/memory complexity of this step is 2^{113}. Then, for each of the 2^{144} found pairs, we perform 2^{48} partial encryptions/decrytions of a δ-set. We evaluate the time complexity of this part to $2^{144+48+8-5}= 2^{195}$ encryptions.

In conclusion, the data complexity is $2^{121.5}$ chosen-plaintexts, the time complexity is $2^{203.5}$ encryptions and the memory complexity is $2^{201.5}$ 128-bit blocks by a trade-off [13].

The 14-round attack on CLEFIA-192 is shown in Fig. 8. The procedure is almost the same as the former attack. The data complexity is $2^{121.5}$ chosen-plaintexts, the time complexity is $2^{139.5}$ encryptions and the memory complexity is $2^{137.5}$ 128-bit blocks.

4.2 Applications to Camellia*-192/256

Camellia is a 128-bit block cipher designed by Aoki et al. in 2000 [1]. It is a Feistel-like construction where two key-dependent layer FL and FL^{-1} are ap-

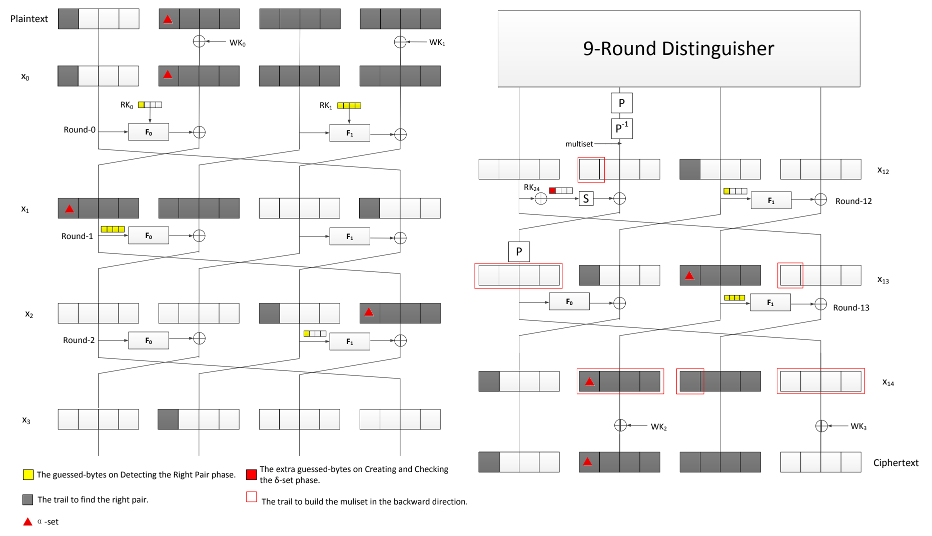

Plaintext
WK₀
WK₁
x₀
RK₀
RK₁
Round-0
F₀
F₁
x₁
Round-1
x₂
Round-2
x₃
9-Round Distinguisher
P
P⁻¹
multiset
x₁₂
RK₂₄
S
Round-12
x₁₃
Round-13
x₁₄
WK₂
WK₃
Ciphertext
The guessed-bytes on Detecting the Right Pair phase.
The extra guessed-bytes on Creating and Checking the δ-set phase.
The trail to find the right pair.
The trail to build the muliset in the backward direction.
α -set

Fig. 8. 15-Round Attack on CLEFIA-192

plied every 6 rounds to each branch. In this paper, we analyze Camellia without FL and FL^{-1}, and call it Camellia* here. We refer to [1] for the detail description of Camellia.

4.2.1 Attack on Camellia*-192. The 8-round distinguisher of Camellia*-192 with 16 guessed-bytes is shown in Fig. 9. In this distinguisher, we modify the method [14] gives, and let the multiset can take place after the permutation. Then use this modified tool to search the affected trail. This idea is inspired by [12].

By guessing $y_{i+1}[0][0]$, $y_{i+2}[0][0,1,2,5,7]$, $y_{i+3}[0]$, $y_{i+4}[0]$, $y_{i+5}[0][1,2,4,6,7]$ and $y_{i+6}[5]$, we can get multiset of $P^{-1}(x_{i+7})[5]$[8]. If there is a truncated differential trail from $x_i[1][0]$ to $x_{i+8}[0][0]$ as the figure shows, then $y_{i+1}[0][0]$, $y_{i+2}[0][0,1,2,5,7]$, $y_{i+3}[0]$, $y_{i+4}[0]$ and $y_{i+5}[0][1,2,4,7]$ can be determined by $\Delta x_i[1][0], y_{i+1}[0][0], y_{i+2}[0][0,1,2,5,7], \Delta x_{i+8}[0][0], y_{i+6}[0][0], y_{i+5}[0][0,1,2,4,7]$.

In a word, the multiset of byte $P^{-1}(x_{i+7}[0])[5]$ can be determined by 16-byte variable.

The online phase of this attack is the same as the 12-round attack on Camellia-192 in [12]. So we can extend 2 rounds on the top and 3 rounds on the bottom to build a 13-round attack on Camellia*-192 with the time complexity of 2^{180} encryption, the data complexity of 2^{113} chosen plaintexts and the memory complexity of 2^{130} 128-bit block.

[8] Since $P^{-1}(x_{i+7}[0])[5] = \Delta z_{i+6}[0][5] \oplus P^{-1}(\Delta x_{i+5}[0])[5]$

4.2.2 Attack on Camellia*-256.
For the distinguisher of Camellia-256, we simply extend one round after round-$(i+3)$ by simply guessing the whole 8-byte state after key addition layer. Then we can get a 10-round distinguisher on Camellia-256.

In the online phase, we simply extend one round after the distinguisher by guessing all the 8-byte state after key addition layer. Then we can build a 15-round attack on Camellia*-256 with the time complexity of 2^{244} encryption, the data complexity of 2^{113} chosen plaintexts and the memory complexity of 2^{194} 128-bit block.

5 Conclusion and Future Work

This paper has shown the improved meet-in-the-middle distinguisher with efficient enumeration technique on BFN and GFN. We discussed the problem why this technique was rarely used on the attacks of BFN and GFN, then described a versatile and powerful algorithm for searching the best improved meet-in-the-middle distinguisher with efficient tabulation technique on them, which is based on recursion and greedy algorithm.

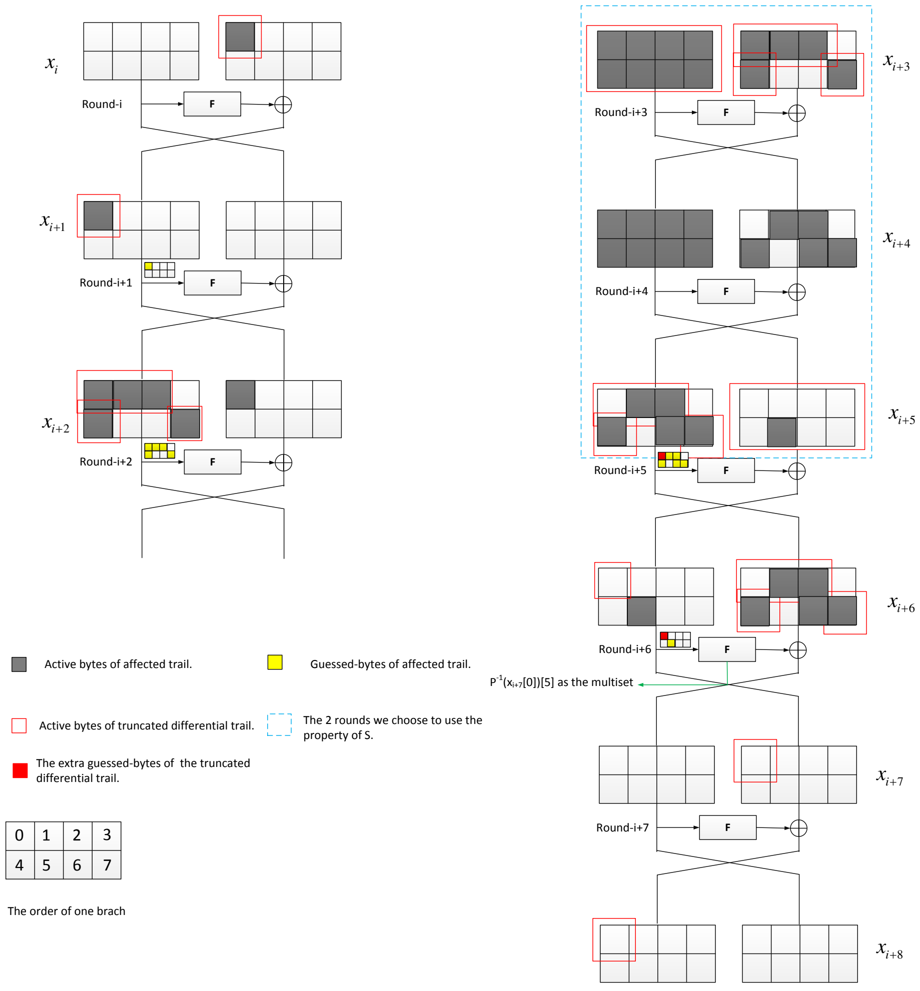
To demonstrate the usefulness and versatility of our approach, we showed several attacks on block ciphers including CLEFIA and Camellia*. Among them, we would like to stress that the presented attack on 14/16-round reduced CLEFIA-192/256 are the best attacks. Since our approach is generic, it is expected to be applied to other BFN and GFN ciphers. We believe that our results are useful not only for a deeper understanding the security of the Feistel schemes, but also for designing a secure block cipher.

The research community has still a lot to learn on the way to build better attacks and there are many future works possible: the algorithm combining the precomputation phase and online phase together, and to find out the link between this kind of attack with other kinds of attacks, such as truncated differential attack and impossible differential attack.

References

1. Kazumaro Aoki, Tetsuya Ichikawa, Masayuki Kanda, Mitsuru Matsui, Shiho Moriai, Junko Nakajima, and Toshio Tokita. Camellia: A 128-bit Block Cipher Suitable for Multiple PlatformsDesign and Analysis. In *Selected Areas in Cryptography*, pages 39–56. Springer, 2001.
2. Andrey Bogdanov, Huizheng Geng, Meiqin Wang, Long Wen, and Baudoin Collard. Zero-Correlation Linear Cryptanalysis with FFT and Iproved Attacks on ISO standards Camellia and CLEFIA. In *Selected Areas in Cryptography–SAC 2013*, pages 306–323. Springer, 2014.
3. Jiazhe Chen, Keting Jia, Hongbo Yu, and Xiaoyun Wang. New Impossible Differential Attacks of Reduced-Round Camellia-192 and Camellia-256. In *Information Security and Privacy*, pages 16–33. Springer, 2011.
4. Joan Daemen and Vincent Rijmen. *The Design of Rijndael: AES-the Advanced Encryption Standard*. Springer, 2002.

5. Hüseyin Demirci and Ali Aydın Selçuk. A Meet-In-the-Middle Attack on 8-Round AES. In *Fast Software Encryption*, pages 116–126. Springer, 2008.
6. Patrick Derbez, Pierre-Alain Fouque, and Jérémy Jean. Improved Key Recovery Attacks on Reduced-Round AES in the Single-Key Setting. In *Advances in Cryptology–EUROCRYPT 2013*, pages 371–387. Springer, 2013.
7. Whitfield Diffie and Martin E Hellman. Special Feature Exhaustive Cryptanalysis of the NBS Data Encryption Standard. *Computer*, 10(6):74–84, 1977.
8. Orr Dunkelman, Nathan Keller, and Adi Shamir. Improved Single-Key Attacks on 8-Round AES-192 and AES-256. In *Advances in Cryptology-ASIACRYPT 2010*, pages 158–176. Springer, 2010.
9. Pierre-Alain Fouque, Jérémy Jean, and Thomas Peyrin. Structural Evaluation of AES and Chosen-Key Distinguisher of 9-round AES-128. In *Advances in Cryptology–CRYPTO 2013*, pages 183–203. Springer, 2013.
10. Henri Gilbert and Marine Minier. A Collisions Sttack on the 7-Rounds Rijndael. 2000.
11. Takanori Isobe and Kyoji Shibutani. Generic Key Recovery Attack on Feistel Scheme. In *Advances in Cryptology-ASIACRYPT 2013*, pages 464–485. Springer, 2013.
12. Leibo Li and Keting Jia. Improved Meet-in-the-Middle Attacks on Reduced-Round Camellia-192/256.
13. Leibo Li, Keting Jia, Xiaoyun Wang, et al. Improved Single-Key Attacks on 9-Round AES-192/256. In *FSE 2014 (21st International Workshop on Fast Software Encryption)*, 2014.
14. Li Lin, Wenling Wu, Yanfeng Wang, and Lei Zhang. General Model of the Single-Key Meet-in-the-Middle Distinguisher on the Word-oriented Block Cipher. In *Information Security and Cryptology–ICISC 2013*, pages 203–223. Springer, 2014.
15. Jiqiang Lu. Cryptanalysis of Block Ciphers. In *PhD thesis.* University of London, UK, 2008.
16. Jiqiang Lu, Yongzhuang Wei, Pierre-Alain Fouque, and Jongsung Kim. Cryptanalysis of Reduced Versions of the Camellia Block Cipher. *IET Information Security*, 6(3):228–238, 2012.
17. Jiqiang Lu, Yongzhuang Wei, Jongsung Kim, and Enes Pasalic. The Higher-Order Meet-in-the-Middle Attack and its Application to the Camellia Block Cipher. *Theoretical Computer Science*, 527:102–122, 2014.
18. Mitsuru Matsui. On Correlation Between the Order of S-boxes and the Strength of DES. In *Advances in Cryptology EUROCRYPT'94*, pages 366–375. Springer, 1995.
19. Kaisa Nyberg. Generalized Feistel Networks. In *Advances in CryptologyASIACRYPT'96*, pages 91–104. Springer, 1996.
20. Taizo Shirai, Kyoji Shibutani, Toru Akishita, Shiho Moriai, and Tetsu Iwata. The 128-bit Blockcipher CLEFIA. In *Fast software encryption*, pages 181–195. Springer, 2007.
21. Cihangir Tezcan. The Improbable Differential Attack: Cryptanalysis of Reduced Round CLEFIA. In *Progress in Cryptology-INDOCRYPT 2010*, pages 197–209. Springer, 2010.



x_i
Round-i
F
x_{i+1}
Round-i+1
x_{i+2}
Round-i+2
x_{i+3}
Round-i+3
x_{i+4}
Round-i+4
x_{i+5}
Round-i+5
x_{i+6}
Round-i+6
P^{-1}(x_{i+7}[0])[5] as the multiset
x_{i+7}
Round-i+7
x_{i+8}
Active bytes of affected trail.
Guessed-bytes of affected trail.
Active bytes of truncated differential trail.
The 2 rounds we choose to use the property of S.
The extra guessed-bytes of the truncated differential trail.
0 1 2 3
4 5 6 7
The order of one brach

Fig. 9. 8-Round Distinguisher on Camellia-192