Generalized Impossible Differential Attacks on Block Ciphers: Application to SKINNY and ForkSKINNY

Ling Song[1], Qinggan Fu[1], Qianqian Yang[2,4*], Yin Lv[3], Lei Hu[2,4]

[1]College of Cyber Security, Jinan University, Guangzhou, China.
[2*]Key Laboratory of Cyberspace Security Defense, Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China.
[3]College of Computer Science, South China Normal University, Guangzhou, China.
[4]School of Cyber Security, University of Chinese Academy of Sciences, Beijing, China.

*Corresponding author(s). E-mail(s): yangqianqian@iie.ac.cn;
Contributing authors: songling.qs@gmail.com;
fuqinggan@stu2018.jnu.edu.cn; lvyin@scnu.edu.cn; hulei@iie.ac.cn;

Abstract

Impossible differential cryptanalysis is a crucial cryptanalytical method for symmetric ciphers. Given an impossible differential, the key recovery attack typically proceeds in two steps: generating pairs of data and then identifying wrong keys using the guess-and-filtering method. At CRYPTO 2023, Boura *et al.* first proposed a new key recovery technique - the differential meet-in-the-middle attack, which recovers the key in a meet-in-the-middle manner. Inspired by this technique, we incorporate the meet-in-the-middle technique into impossible cryptanalysis and propose a generic impossible differential meet-in-the-middle attack (`IDMA`) framework. We apply `IDMA` to block ciphers `SKINNY`, `SKINNYe`-v2, and `ForkSKINNY` and achieve remarkably efficient attacks. We improve the impossible differential attack on `SKINNY-`$\boldsymbol{n}$`-3`$\boldsymbol{n}$ by 2 rounds in the single-tweakey setting and 1 round in the related-tweakey setting. For `SKINNYe`-v2, the impossible differential attacks now can cover 2 more rounds in the related-tweakey setting and the first 23/24/25-round attacks in the single-tweakey model are given. For `ForkSKINNY-`$\boldsymbol{n}$`-3`$\boldsymbol{n}$, we improve the attacks by 2 rounds in the limited setting specified by the designers and 1 round in relaxed settings. These results confirm

that the meet-in-the-middle technique can result in more efficient key recovery, reaching beyond what traditional methods can achieve on certain ciphers.

Keywords: Impossible differential cryptanalysis, Meet-in-the-middle, Key recovery, SKINNY, ForkSKINNY

1 Introduction

Differential cryptanalysis [1] is one of the most powerful techniques in symmetric cryptography, aimed at recovering the secret key by exploiting high-probability differential characteristics that describe the propagation of input differences through a cipher. Over the years, several variants of differential cryptanalysis have been developed, including boomerang attacks [2], higher-order differential cryptanalysis [3], and impossible differential cryptanalysis, among others.

The impossible differential (ID) attack, introduced by Biham *et al.* [4] and Knudsen [5], specifically exploits differentials that occur with a probability of zero. For a long time, it was one of the most effective attacks against `AES`-128 [6, 7]. Differential-like attacks generally involve two main steps. The first step is to identify a long characteristic or distinguisher that spans as many rounds as possible. The search for (impossible) differential characteristics has been thoroughly explored in the literature [8–13]. Once a distinguisher is found, a key recovery attack is typically launched by adding several rounds around the distinguisher.

In differential cryptanalysis, the objective is to identify the most probable key candidates. In contrast, impossible differential cryptanalysis focuses on eliminating key candidates that would result in impossible differentials, thereby facilitating the recovery of the master key. Despite their differences, these two cryptanalytic approaches share many key recovery techniques, including the use of structures of data [14], the early abort technique [15], and state test [16]. In [16], Boura *et al.* formalized impossible differential cryptanalysis, where pairs of data that satisfy the plaintext and ciphertext differences are first generated, and then key candidates that would allow any pair of data to lead to the impossible differential characteristic are identified and discarded. Generic complexity analysis formulas for executing such attacks were also derived in [16]. Since then, this approach has become the standard framework for all impossible differential attacks on block ciphers [7, 11, 12, 17, 18].

Recently, Boura *et al.* introduced a novel cryptanalysis technique known as the differential meet-in-the-middle (MITM) attack [19]. This technique proposes to recover the key in an MITM manner. In this approach, the conditions on plaintexts and ciphertexts are considered separately at first and then combined later. Significantly different from traditional differential key recovery attacks, this approach has yielded favorable results on block ciphers `SKINNY`-128-384 and `AES`-256. Given that impossible differential cryptanalysis is derived from differential cryptanalysis, one might wonder if the MITM technique could be combined with impossible differential cryptanalysis. This combination could potentially lead to more effective key recovery attacks for certain block ciphers.

Our contributions. In this paper, we incorporate the meet-in-the-middle technique into the key recovery process of impossible differential attacks. Since the probability of an impossible differential is zero, the MITM procedure of the differential meet-in-the-middle attack, which is carried out on individual plaintext-ciphertext pairs, does not apply in impossible differential attacks. To overcome this limitation, we proposed implementing the MITM procedure for structures of plaintexts as in [20]. This gives rise to the impossible differential meet-in-the-middle attack (`IDMA`).

In `IDMA`, in order to have efficient key recovery attacks, it is desirable to involve balanced key bits before and after the impossible differential, as in standard MITM attacks, *e.g.*, [21, 22]. When the two sets of key bits around the impossible differential are not balanced, inspired by the parallel partition technique as described in [19], we propose an extended technique called isolate-and-unite. This technique is designed to make the key bits balanced in the MITM procedure. It works by peeling off some outer parts of the encryption that are isolated during the MITM procedure but will be taken back into consideration during the matching process. This technique further enhances the `IDMA`.

We apply the `IDMA` framework to three tweakable block ciphers, namely `SKINNY`, `SKINNYe`-v2, and `ForkSKINNY`, in both the single-tweakey and related-tweakey settings. In order to mount more effective ID attacks, we develop an integrated search model that not only includes the distinguisher but also the extended parts. Our model of `IDMA` is built upon an efficient CP-based tool recently proposed by Hadipour et al. [11, 12]. As a result, we obtain significantly better attacks compared to traditional impossible differential attacks for versions where the key size is more than twice the block size. This confirms that the `IDMA` can outperform the traditional impossible differential attacks on certain block ciphers. Our results are summarized as follows, and the comparison with previous works is demonstrated in Table 1 and Table 2.

1. For `SKINNY`-n-$3n$ in the single-tweakey setting, we not only improve the time complexity of the impossible differential attack on 21-round `SKINNY`-n-$3n$, but also successfully extend the attack by 2 rounds. At the same time, for `SKINNY`-n-$3n$ in the related-tweakey setting, we obtain an enhanced impossible differential attack on 27 rounds and extend the attack by 1 round.
2. For `SKINNYe`-v2, for the first time, we present impossible differential attacks on 23/24/25-round `SKINNYe`-v2 in the single-tweakey setting. Additionally, we achieve an improved impossible differential attack on 31 rounds in the related-tweakey setting and further extend the attack by 2 rounds.
3. For `ForkSKINNY`, on the one hand, we not only provide improved related-tweakey impossible differential attacks on 28-round `ForkSKINNY`-n-$3n$ in the limited setting but also successfully extend the attack by 2 rounds. On the other hand, we propose enhanced related-tweakey impossible differential attacks on 32-round `ForkSKINNY`-n-$3n$ in the arbitrary setting and extend the attack by 1 round.

Organization. The rest of the paper is organized as follows. Some preliminaries of impossible differential attacks and the differential meet-in-the-middle attack are given in Section 2. Section 3 presents our generic impossible differential meet-in-the-middle

Table 1: Summary of the cryptanalytic results of SKINNY-n-3n.

Cipher	#R	Attack	Time	Data	Memory	Setting	Ref.
SKINNY-64-192	21	ID	$2^{180.50}$	2^{62}	2^{170}	STK	[23]
	21	ID	$2^{174.42}$	$2^{62.43}$	2^{168}	STK	[11]
	21	**IDMA**	$\mathbf{2^{171.03}}$	$\mathbf{2^{62.43}}$	$\mathbf{2^{168}}$	STK	App. A.1
	22	**IDMA**	$\mathbf{2^{188.99}}$	$\mathbf{2^{61.03}}$	$\mathbf{2^{188}}$	STK	Sect. 4.5.1
	23	Int	$2^{155.60}$	$2^{73.20}$	2^{138}	180,SK	[24]
	23	MITM	2^{188}	2^{28}	2^{4}	STK	[25]
	23	Tr-Diff-MITM	2^{188}	2^{56}	2^{104}	STK	[26]
	23	**IDMA**	$\mathbf{2^{188.99}}$	$\mathbf{2^{61.03}}$	$\mathbf{2^{188}}$	STK	App. A.2
	26	Int	2^{172}	2^{61}	2^{172}	180,SK	[11]
	26	Int	2^{166}	2^{62}	2^{164}	180,SK	[12]
	27	ID	2^{189}	$2^{63.53}$	2^{184}	RTK	[27]
	27	IB	$2^{168.23}$	$2^{67.10}$	2^{160}	RTK	[28]
	27	ID	$2^{183.26}$	$2^{63.64}$	2^{172}	RTK	[11]
	27	**IDMA**	$\mathbf{2^{183.00}}$	$\mathbf{2^{63.64}}$	$\mathbf{2^{172}}$	RTK	App. A.3
	28	IB	$2^{190.80}$	$2^{66.37}$	2^{184}	RTK	[28]
	28	**IDMA**	$\mathbf{2^{188.99}}$	$\mathbf{2^{65.03}}$	$\mathbf{2^{188}}$	RTK	App. A.4
	31	Rect.	$2^{182.07}$	$2^{62.78}$	$2^{62.79}$	RTK	[29]
SKINNY-128-384	21	ID	$2^{347.35}$	$2^{122.89}$	2^{336}	STK	[11]
	21	**IDMA**	$\mathbf{2^{344.33}}$	$\mathbf{2^{122.89}}$	$\mathbf{2^{336}}$	STK	App. A.1
	22	**IDMA**	$\mathbf{2^{378.22}}$	$\mathbf{2^{121.50}}$	$\mathbf{2^{376}}$	STK	Sect. 4.5.1
	23	MITM	2^{376}	2^{104}	2^{8}	STK	[30]
	23	DS-MITM	2^{372}	2^{96}	$2^{352.46}$	STK	[31]
	23	Diff-MITM	$2^{361.90}$	2^{117}	$2^{118.50}$	STK	[19]
	23	**IDMA**	$\mathbf{2^{378.22}}$	$\mathbf{2^{121.50}}$	$\mathbf{2^{376}}$	STK	App. A.2
	24	Diff-MITM	$2^{361.90}$	2^{117}	2^{183}	STK	[19]
	24	Diff-MITM	$2^{372.50}$	$2^{122.30}$	$2^{123.80}$	STK	[19]
	25	Diff-MITM	$2^{372.50}$	$2^{122.30}$	$2^{188.30}$	STK	[19]
	25	Diff-MITM	$2^{378.90}$	2^{177}	2^{165}	STK	[26]
	25	Diff-MITM	2^{366}	$2^{122.30}$	$2^{188.30}$	STK	[26]
	26	Int	2^{344}	2^{121}	2^{340}	360,SK	[11]
	26	Int	2^{331}	2^{122}	2^{328}	360,SK	[12]
	27	ID	2^{378}	$2^{126.03}$	2^{368}	RTK	[27]
	27	IB	2^{337}	$2^{131.30}$	2^{320}	RTK	[28]
	27	ID	$2^{362.61}$	$2^{124.99}$	2^{344}	RTK	[11]
	27	**IDMA**	$\mathbf{2^{361.06}}$	$\mathbf{2^{124.99}}$	$\mathbf{2^{344}}$	RTK	App. A.3
	28	IB	$2^{382.80}$	$2^{130.26}$	2^{368}	RTK	[28]
	28	**IDMA**	$\mathbf{2^{378.22}}$	$\mathbf{2^{129.50}}$	$\mathbf{2^{376}}$	RTK	App. A.4
	32	Rect.	$2^{344.78}$	$2^{123.54}$	$2^{129.54}$	RTK	[32]

Int: Integral, IB: Impossible Boomerang, DS-MITM: Demirci-Selcuk MITM, Rect.: Rectangle, Diff-MITM: Differential MITM, Tr-Diff-MITM: Truncated Differential MITM, IDMA: Impossible Differential MITM Attack, STK/RTK: single/related-tweakey, SK: single-key with given keysize.

attack (IDMA) framework and a flexible key guessing strategy supported by the isolate-and-unite technique. In Section 4, we apply the IDMA framework to SKINNY, SKINNYe-v2, and ForkSKINNY and present a 22-round attack on SKINNY-n-3n in the single-tweakey setting and a 30-round attack on ForkSKINNY-n-3n in the related-tweakey setting. Other attacks are postponed to appendices. Finnaly, Section 5 concludes the paper.

Table 2: Summary of the cryptanalytic results of SKINNYe-v2 and ForkSKINNY-n-3n.

Cipher	#R	Attack	Time	Data	Memory	Setting	Ref.
SKINNYe-v2	**23**	**IDMA**	$\mathbf{2^{255.29}}$	$\mathbf{2^{63.99}}$	$\mathbf{2^{240}}$	STK	App. B.1
	23	**IDMA**	$\mathbf{2^{243.00}}$	$\mathbf{2^{68.17}}$	$\mathbf{2^{240}}$	STK	App. B.1
	24	**IDMA**	$\mathbf{2^{255.29}}$	$\mathbf{2^{63.99}}$	$\mathbf{2^{252}}$	STK	App. B.2
	25	**IDMA**	$\mathbf{2^{255.29}}$	$\mathbf{2^{63.99}}$	$\mathbf{2^{252}}$	STK	App. B.3
	30	Int	2^{232}	2^{65}	2^{228}	240,SK	[11]
	31	ID	$2^{251.14}$	2^{63}	2^{110}	RTK	[12]
	31	**IDMA**	$\mathbf{2^{250.22}}$	$\mathbf{2^{63}}$	$\mathbf{2^{236}}$	RTK	App. B.4
	32	**IDMA**	$\mathbf{2^{252.99}}$	$\mathbf{2^{65.03}}$	$\mathbf{2^{252}}$	RTK	App. B.5
	33	**IDMA**	$\mathbf{2^{252.99}}$	$\mathbf{2^{65.03}}$	$\mathbf{2^{252}}$	RTK	App. B.6
ForkSKINNY-64-192	28*	ID	$2^{169.60}$	2^{61}	2^{104}	RTK	[12]
	28*	**IDMA**	$\mathbf{2^{168.90}}$	$\mathbf{2^{61}}$	$\mathbf{2^{156}}$	RTK	App. C.1
	30*	**IDMA**	$\mathbf{2^{188.99}}$	$\mathbf{2^{65.03}}$	$\mathbf{2^{188}}$	RTK	Sect. 4.5.2
	32	ID	$2^{186.27}$	2^{63}	2^{114}	RTK	[12]
	32	**IDMA**	$\mathbf{2^{186.22}}$	$\mathbf{2^{63}}$	$\mathbf{2^{176}}$	RTK	App. C.2
	33	**IDMA**	$\mathbf{2^{188.99}}$	$\mathbf{2^{65.03}}$	$\mathbf{2^{188}}$	RTK	App. C.3
ForkSKINNY-128-384	**28***	**IDMA**	$\mathbf{2^{316.82}}$	$\mathbf{2^{118.54}}$	$\mathbf{2^{312}}$	RTK	App. C.1
	30*	**IDMA**	$\mathbf{2^{378.22}}$	$\mathbf{2^{129.50}}$	$\mathbf{2^{376}}$	RTK	Sect. 4.5.2
	32	**IDMA**	$\mathbf{2^{356.14}}$	$\mathbf{2^{125.27}}$	$\mathbf{2^{352}}$	RTK	App. C.2
	33	**IDMA**	$\mathbf{2^{378.22}}$	$\mathbf{2^{129.50}}$	$\mathbf{2^{376}}$	RTK	App. C.3

* indicates the limit setting of ForkSKINNY specified by the designers.

2 Preliminaries

2.1 Impossible Differential Attacks

The impossible differential attack is a significant cryptanalysis of block ciphers, which was independently introduced by Biham *et al.*[4] and Knudsen [5]. The core idea of the attack lies in exploiting the impossible differentials, which are the differentials that occur with zero probability. The common approach is to extend the impossible differential by a specific number of rounds in both directions. Subsequently, guess the key bits involved in those rounds and check whether a pair is partially encrypted and decrypted to the impossible differential. If so, the key should be eliminated from the space of candidate keys.

Key Recovery Attacks. Let $E = E_f \circ E_d \circ E_b$ be a block cipher with an impossible differential $\Delta x \nrightarrow \Delta y$ over E_d, where E_b and E_f are the extended parts on both sides of E_d, as shown in Figure 1.

Suppose the block size of the block cipher E is n bits and the key size is k bits. The input difference Δx (resp. Δy) of E_d propagates over E_b^{-1} (reps. E_f) with probability 1 to ΔP (resp. ΔC). Let V_b (resp. V_f) be the space spanned by all possible ΔP (resp. ΔC) where $r_b = \log_2 |V_b|$ (resp. $r_f = \log_2 |V_f|$). Let k_b be the subset of subkey bits which are employed in E_b and k_f be the subset of subkey bits which are employed in E_f. Let $|k_b|$ and $|k_f|$ be the number of bits in k_b and k_f. Let c_b be the number of conditions that should be satisfied for $\Delta P \to \Delta x$, that is, c_b is the number of filtering bits under the guessed key bits for E_b. Let c_f be the number of conditions that should

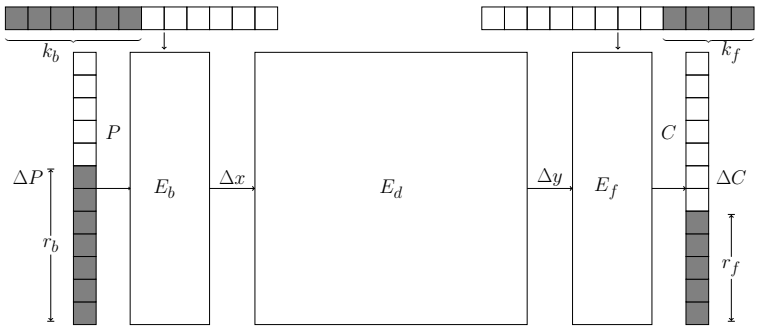

Fig. 1: A high-level description of the impossible differential attack

be satisfied for $\Delta C \to \Delta y$, that is, c_f is the number of filtering bits under the guessed key bits for E_f.

The core idea for the key recovery attack based on an impossible differential is to exploit the impossible differential in E_d to retrieve the key by discarding all key candidates leading to $\Delta x \nrightarrow \Delta y$ over E_d. The number of all possible candidates for the subkey involved in E_b and E_f is $2^{|k_b \cup k_f|}$. Suppose the impossible differential attack needs N pairs messages that satisfy the plaintext difference ΔP and the ciphertext difference ΔC. For a given key, the probability that the key is retained is

$$Pr = (1 - (2^{-(c_b+c_f)}))^N \approx e^{-N2^{-(c_b+c_f)}}.$$

Thus, after testing all the key candidates with N pairs, the number of the remaining key candidates is $2^{|k_b \cup k_f|} \cdot Pr$.

Complexities. Let $Pr = 2^{-x}$, *i.e.*, x information bits of the key can be recovered using N pairs of messages. Thus $e^{-N2^{-(c_b+c_f)}} = 2^{-x}$ and $N = 2^{c_b+c_f} x \ln 2 = 2^{c_b+c_f+z}$ if we let $x \ln 2 = 2^z$. We consider all possible ways of acquiring data, including multiple structures and a partial structure.

- When $y \geq 1$ structures of data is needed, let $D = y \cdot 2^{r_b}$ and $N = y \cdot 2^{2r_b-1} \cdot 2^{-n+r_f} = D \cdot 2^{r_b-1-n+r_f}$. We get that $D = 2^{c_b+c_f+z+n+1-r_b-r_f}$.
- When a partial structure is enough, D plaintexts will construct $N = D^2/2 \cdot 2^{-n+r_f}$ pairs by choosing the plaintexts and $N = D^2/2 \cdot 2^{-n+r_b}$ pairs by choosing the ciphertexts. We get $D = \min_{r \in \{r_b, r_f\}} \sqrt{2^{c_b+c_f+n+1-r+z}}$.

Thus, the data complexity is

$$D = \max\{\min_{r \in \{r_b, r_f\}} \{\sqrt{2^{c_b+c_f+n+1-r+z}}\}, 2^{c_b+c_f+n+1-r_b-r_f+z}\}.$$

The time complexity is composed of three parts:

$$T = D + 2^{|k_b \cup k_f|+z} + 2^{k-x},$$

where the first term is the time complexity of collecting data, the second term is the time complexity of processing the key candidates, and the last one is the time complexity of the exhaustive search. Note that, this formula of the time complexity is a lower bound while the actual time complexity depends on the concrete situation. The memory complexity is $M = \min\{N, 2^{|k_b \cup k_f|}\}$.

2.2 Differential Meet-in-the-Middle Attacks

The differential meet-in-the-middle (MITM) attack was first proposed by Boura *et al.* [19] at CRYPTO 2023, as depicted in Figure 1. Suppose the probability of the differential $\Delta x \to \Delta y$ over E_d is 2^{-p}. The core idea of the attack is to randomly choose a plaintext-ciphertext pair (P, C), prepare two separate sets, each for one side, and find out the possible right key when the two sets are matched. This process repeats about 2^p times until the unique right key is found.

Detailed steps of the differential meet-in-the-middle attack are illustrated in Algorithm 1. For the chosen pair (P, C), it generates a set of $\widetilde{P}$ by traversing all possible subkey k_b involved in E_b so that $(P, \widetilde{P})$ satisfies the input difference Δx, generate a set of $\widehat{C}$ by traversing all possible subkey k_f involved in E_f so that $(C, \widehat{C})$ satisfies the output difference Δy. If P belongs to one right pair, then the right k_b and k_f must lead to a match between $\widetilde{C}$ and $\widehat{C}$. In other words, we will get one right pair satisfying $E_b(P) \oplus E_b(\widetilde{P}) = \Delta x$ and $E_f^{-1}(C) \oplus E_f^{-1}(\widehat{C}) = \Delta y$ and the corresponding keys k_b and k_f.

Algorithm 1 Differential Meet-in-the-Middle Attack [19]

Randomly choose 2^p plaintext-ciphertext pairs (P, C).
for each one **do**
  for each possible i for k_b, **do**
    Compute $\widetilde{P}^i$ that $E_b(P) \oplus E_b(\widetilde{P}^i) = \Delta x$.
    Acquire $\widetilde{C}^i = E(\widetilde{P}^i)$ and store $(\widetilde{C}^i, i)$ in a hash table H.
  end for
  for each possible j for k_f, **do**
    Compute $\widehat{C}^j$ that $E_f^{-1}(C) \oplus E_f^{-1}(\widehat{C}^j) = \Delta y$.
    for each $i \in H(\widehat{C}^j)$ **do**
      Get $(\widetilde{C}^i, \widehat{C}^j)$ and (i, j).
      Test candidates (i, j) against extra data.
    end for
  end for
end for

Complexities. The data complexity of the attack can be roughly estimated as $D = \min\{2^n, 2^{p+\min(|k_b|,|k_f|)}\}$. The time complexity of this attack can be estimated as

$$T = 2^p \times \left(2^{|k_b|} + 2^{|k_f|}\right) + 2^{|k_b \cup k_f| - n + p} + 2^{k - n + p},$$

where the first term corresponds to the computations done in the upper part E_b and the lower part E_f, the second one to the number of expected key candidates for $k_b \cup k_f$ and the last one to the exhaustive search. The memory complexity is given by $M = 2^{\min(|k_b|,|k_f|)}$, but it can be improved to $2^{\min(|k_b|,|k_f|)-|k_b \cap k_f|}$ by guessing the common key material at the beginning.

3 New Generalized Impossible Differential Attack

3.1 Motivations and Obstacles

The differential meet-in-the-middle attack has demonstrated significantly better results than traditional differential attacks in the analysis of certain block ciphers. For instance, the differential meet-in-the-middle attack on SKINNY-128-384 [19] can successfully cover 25 rounds in the single-key setting, a feat beyond the reach of conventional differential key recovery attacks. This leads to an intriguing question: *can the meet-in-the-middle technique be combined with the impossible differential attack to enhance its effectiveness against certain block ciphers?*

In a differential meet-in-the-middle attack, if the underlying differential characteristic has a probability of 2^{-p}, then typically 2^p plaintext-ciphertext pairs (P, C) are chosen for the meet-in-the-middle procedure, as shown in Algorithm 1. However, in an impossible differential attack, the probability of the differential is zero. How can this obstacle be addressed?

Upon examining the differential meet-in-the-middle attack, it becomes clear that it essentially involves storing data that satisfies the conditions on one side of the differential characteristic in a table, generating data that satisfies the conditions on the other side, and then combining data from both sides by looking up the table. Notably, choosing plaintext-ciphertext pairs (P, C) is unnecessary before initiating the MITM procedure. Instead, one potential solution is to store full pairs $(C, \widetilde{C})$ that meet the conditions on one side, rather than storing individual ciphertexts $\widetilde{C}$. With this adjustment, matching should occur based on both messages. Building on this idea, we propose the generalized impossible differential attack framework that specifically utilizes the meet-in-the-middle technique.

3.2 Impossible Differential Meet-in-the-Middle Attack (IDMA)

In the following, we describe our algorithm for the impossible differential attack that utilizes the meet-in-the-middle technique. Suppose an impossible differential $\Delta x \nrightarrow \Delta y$ over E_d is used in the attack. Other parameters for the key recovery are the same as introduced in Section 2.1 and Figure 1. The specific steps of our algorithm are as follows when multiple structures of data are used. When a partial structure is enough, the steps of the attack are similar.

1. Initialize a vector L of flags for all possible $k_b \cup k_f$.
2. Phase of data collection. Collect and store y structures of 2^{r_b} plaintexts and the corresponding ciphertexts, from which there will be $N = y2^{2r_b-1+r_f-n}$ pairs of messages having an input difference in ΔP and an output difference in ΔC.
3. Phase of MITM. For each of the y structure, guess the subkey $k_\cap = k_b \cap k_f$:

(a) Generate the first set. For each data (P, C) in the structure, guess the subkey $k_b \setminus k_\cap$ and partially encrypt P. Store the data in a hash table indexed by the $n - r_f + c_b$ filtering bits and one can get $\frac{N}{y}2^{|k_b|-c_b}$ pairs with the fixed difference on the filtering bits. Further, store the corresponding $(C, \widetilde{C})$ together with the guessed k_b in a hash table H.
The time complexity of this step is $2^{|k_b|}D$ partial encryption and $N2^{|k_b|-c_b}$ memory accesses.

(b) Generate the second set. For each data (P, C) in the structure, guess the subkey $k_f \setminus k_\cap$ and partially decrypt C. Store the data in a hash table indexed by the $n - r_f + c_f$ filtering bits and one can get $\frac{N}{y}2^{|k_f|-c_f}$ pairs $(C, \widetilde{C})$ with the fixed difference on the filtering bits.
The time complexity of this step is $2^{|k_f|}D$ partial decryption and $N2^{|k_f|-c_f}$ memory accesses.

(c) Match two sets. Look up the hash table H by using the pairs from the second set as the index. There will be $N2^{|k_b \cup k_f|-c_b-c_f}$ matched pairs $(C, \widetilde{C})$.

(d) Discard wrong key candidates. For each matched pair $(C, \widetilde{C})$, extract the associated key candidates for $k_b \cup k_f$ and discard these key candidates by updating the key flags.
The time complexity of this step is $N2^{|k_b \cup k_f|-c_b-c_f}$ memory accesses.

4. Phase of exhaustive search. For the left key candidates, guess the remaining key bits and exhaustively search over them to recover the right key. The time complexity of this step is 2^{k-x} where $2^{-x} = (1 - 2^{-(c_b+c_f)})^N$.

Note that, in Step 3(a), pairs of messages are stored. Analyzing how many matched pairs can be obtained in Step 3(c) is crucial. We divide it into two sub-cases.

- When multiple structures are used. Given two random pairs from the same structure, they will match with probability 2^{-2r_b+1} as there are 2^{2r_b-1} pairs in a structure. Therefore, there will be $y \cdot 2^{|k_b \cup k_f|} \cdot 2^{2r_b-1-c_b} \cdot 2^{2r_b-1+r_f-n-c_f} \cdot 2^{-2r_b+1} = N2^{|k_b \cup k_f|-c_b-c_f}$ pairs in Step 3(c).
- When a partial structure is used. Given two random pairs from a partial structure of $D = \sqrt{N2^{n+1-r_f}}$ plaintexts, they will match with probability $2/D^2$, resulting $N2^{|k_b \cup k_f|-c_b-c_f}$ pairs in Step 3(c) as well due to $2^{|k_b \cup k_f|} \cdot D^2 \cdot 2^{-1-c_b} \cdot D^2 \cdot 2^{-1+r_f-n-c'_f} \cdot 2/D^2 = N2^{|k_b \cup k_f|-c_b-c_f}$.

Therefore, there are $N2^{|k_b \cup k_f|-c_b-c_f}$ matched pairs in either cases. The complexities of the differential meet-in-the-middle attack are as follows.

Data complexity. Taken all possible ways of acquiring data, the data complexity is

$$D = \max\{\min_{r \in \{r_b, r_f\}}\{\sqrt{2^{c_b+c_f+n+1-r+z}}\}, 2^{c_b+c_f+n+1-r_b-r_f+z}\}, \tag{1}$$

where $2^z = x \ln 2$.

Memory complexity. Memory consumption consists of three components: storage for data, storage for pairs, and storage for key flags. Therefore, the memory complexity

is

$$M = \max\{D/\max\{1, y\}, \min\{N2^{|k_b|-c_b}, N2^{|k_f|-c_f}\}/\max\{1, y\}, 2^{|k_b \cup k_f|}\}.$$

Time complexity. The time complexity of collecting data is $T_0 = D$, the time complexity of doing partial encryption and decryption under the guessed key bis is

$$T_1 = (2^{|k_b|} + 2^{|k_f|}) \cdot D,$$

the time complexity of getting pairs of data that satisfy the conditions of the input difference and the ouput difference is

$$T_2 = N2^{|k_b|-c_b} + N2^{|k_f|-c_f},$$

the time complexity of extracting the wrong key candidates is

$$T_3 = 2^{|k_b \cup k_f|+z},$$

and the time complexity for the exhaustive search is $T_4 = 2^{k-x}$. The overall time complexity is equivalent to

$$T = T_0 + T_1 \cdot C_E + (T_2 + T_3) \cdot C_M + T_4$$

times of encryption, where C_E is the ratio of the cost of partial encryption to the full encryption and C_M is the ratio of the cost of memory access to the full encryption.

3.3 A Flexible Key Guessing Strategy

In the impossible differential meet-in-the-middle attack described in Section 3.2, all possible values for k_b and k_f are exhaustively guessed to get the two sets during the meet-in-the-middle procedure (Step 3(a) and 3(b)). However, in scenarios where $2^{|k_b|} \cdot D$ and $2^{|k_f|} \cdot D$ are not close, it may be advantageous to guess fewer bits of k_b or k_f. To address this, we propose a method called the *isolate-and-unite* technique that allows skipping some outer key bits on one side in the meet-in-the-middle procedure.

Isolate-and-unite: outer key bits unguessed. Inspired by the techniques in meet-in-the-middle attacks like the initial structure [33], an improvement called *parallel partition* was proposed for the differential meet-in-the-middle attack [19, 34]. This technique allows to increase the number of rounds attacked in certain situations.

Building on the same idea of parallel partition, we formalize the isolate-and-unite technique for `IDMA` which allows to leave some outer key bits unguessed. These outer key bits are used at the beginning or end of the encryption process. This approach involves peeling off a portion of the outer encryption and then applying the meet-in-the-middle procedure to the remaining part. The isolated portion is accounted for during the matching process.

To illustrate this concept, we showcase the technique on a toy cipher as depicted in Figure 2. The cipher $E = E_f \circ E_d \circ E_b$ has block size $n = 6c$, where E_b begins with a key

addition (AK). Suppose the key recovery attack is mounted based on an impossible differential $\Delta x \nrightarrow \Delta y$ over E_d. The parameters are: $r_b = 5c, r_f = 4c, |k_b| = 8c$ and $|k_f| = 6c$. Suppose the attack requires $y > 1$ structures of plaintexts, so $D = y \cdot 2^{r_b}$. Then computing the two sets in the MITM procedure takes a time complexity $T_1 = 2^{|k_b|} \cdot D + 2^{|k_f|} \cdot D \approx 2^{8c} \cdot D$.

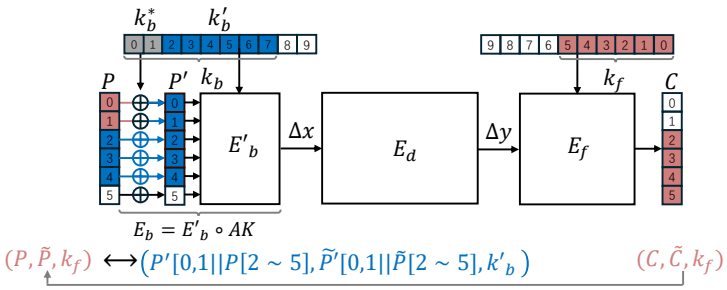

Fig. 2: An illustration of the isolate-and-unite method

As the sizes of k_b and k_f are not balanced, we consider peeling off a portion at the beginning of the encryption to reduce k_b. For example, we peel off $P[0,1] \oplus k_b^* = P'[0,1]$, as illustrated in Figure 2. Note that the map from $P[0,1]$ to $P'[0,1]$ is a keyed permutation within the active part of E_b, so each structure of P is also a structure of $P'[0,1]||P[2 \sim 5]$ after the keyed permutation. We can then carry the MITM procedure for $P'[0,1]||P[2 \sim 5]$ and the ciphertext C with the $6c$-cell k_b' involved and the $2c$-cell k_b^* is isolated. Therefore, computing the two sets now takes a time complexity $T_2 = 2^{|k_b'|} \cdot D + 2^{|k_f|} \cdot D \approx 2^{6c} \cdot D$, which is lower than before.

During the matching process, the relation $P[0,1] \oplus k_b^* = P'[0,1]$ can be considered. For the data in the first set, $P'[0,1]$ is known; for the data in the second set, C is known and thus P is known according to the data set. If k_b^* can be computed with $k_b^* = g_b(k_b') \oplus g_f(k_f)$ for some functions g_b, g_f, the data of the two sets can be matched by $P[0,1] \oplus g_f(k_f) = P'[0,1] \oplus g_b(k_b')$, meaning the matching can be performed effectively with no extra computational cost. Consequently, the time complexities $T_1 = (2^{|\mathbf{k_b'}|} + 2^{|k_f|}) \cdot D$, $T_2 = N2^{|\mathbf{k_b'}|-c_b} + N2^{|k_f|-c_f}$, and $T_3 = 2^{|k_b \cup k_f|+z}$ can be achieved, thereby potentially reducing the overall time complexity of the attack.

In this case, we call k_b^* cost-free isolable. When k_b^* shares no common information with $k_b' \cup k_f$, k_b^* is also cost-free isolable, and each combination of $P[0,1]$ and $P'[0,1]$ suggests a candidate for k_b^*. We formalize the definition of the cost-free isolable key bits as follows.

Definition 1. *(Cost-free isolable key bits) Let k_b^* be the key involved in the isolated function from $P[I]$ to $P'[I]$ where I is an index set for cells and $f_1(P'[I]) = k_b^* \oplus f_2(P[I])$ for some reversible functions f_1, f_2. If matching can be performed and k_b^* determined with the time complexity equivalent to the number of values of $k_b \cup k_f$ to be discarded, then k_b^* is referred to as **cost-free isolable key bits**. Specifically, this is the case if:*

- $k_b^* \subset k_b \setminus (k_b \cap k_f)$ *or*
- $k_b^* \subset (k_b \cap k_f)$ *can be derived from* k_b' *and* k_f *such that* $k_b^* = g_b(k_b') \oplus g_f(k_f)$ *for some functions* g_b, g_f,

then k_b^* *is cost-free isolable. The same definition applies to the key bits in the forward extension* E_f.

Note the isolated function should be located within the active part of E_b or E_f. Since structures of messages can only be constructed on one side at a time, this method applies to either the plaintext or ciphertext side. An example application can be found in the attacks on `ForkSKINNY`-n-$3n$ discussed in Section 4.5.2.

Comparison with the parallel partition. When the parallel partition technique is employed, the differential meet-in-the-middle attack can extend by one or two extra rounds without increasing complexity, provided that the subkey used in the extra round(s) can be deduced from k_b and k_f. In the context of impossible differential attacks, the isolate-and-unite technique serves as a counterpart to the parallel partition method. It is important to note that in an impossible differential attack, the condition $|k_b \cup k_f| < k$ must hold, whereas this restriction does not apply in differential attacks. The parallel partition technique typically allows for covering an entire round at no additional cost in a differential meet-in-the-middle attack. However, adding a full round is often infeasible in an impossible differential attack due to this restriction. In our work, the isolate-and-unite technique extends the parallel partition approach by applying isolation at a more granular level, such as S-boxes, rather than full rounds. Additionally, the conditions for employing this technique are more flexible, as it can be utilized even when the isolated subkey k_b^* cannot be directly derived. As a result, the isolate-and-unite technique not only aligns more effectively with the impossible differential attacks but also generalizes and enhances the parallel partition technique.

3.4 Comparison of the IDA, DMA and IDMA

As variants of differential cryptanalysis, the Impossible Differential Attack (IDA), the Differential MITM Attack (DMA), and the Impossible Differential MIMT Attack (`IDMA`) share similarities in the key recovery process. They all involve a distinguisher and several outer rounds, with the potential keys for the outer rounds determined using the distinguisher as a sieve. However, these approaches differ in how data pairs are generated and how filtering techniques are applied.

The IDA generally uses the data structure to generate pairs and then employs the guess-and-determine technique to verify whether the distinguisher is satisfied. The DMA generates data pairs by traversing the keys and then verifies whether the distinguisher is satisfied by matching a single ciphertext through the meet-in-the-middle technique. The `IDMA` makes use of the data structure to generate pairs, combines the flexible guessing strategy with the meet-in-the-middle technique, and finally verifies whether the distinguisher is satisfied by matching on the plaintext-ciphertext pairs. We present the comparisons in Table 3.

Table 3: Comparisons of three related attacks

Attacks	Structure technique for pair generation	MITM technique for key recovery	Flexible key guessing strategy
IDA	✓	×	×
DMA	×	✓ on one ciphertext	×
IDMA	✓	✓ on a pair of ciphertexts	✓

4 Applications to SKINNY and ForkSKINNY

In this section, we analyze two block ciphers, SKINNY and ForkSKINNY, using the impossible differential meet-in-the-middle attack framework described in Section 3.2.

We begin with a brief overview of SKINNY and ForkSKINNY, followed by the development of a constraint programming (CP) model. This model is designed to search for efficient impossible differential meet-in-the-middle attacks on these ciphers by minimizing the complexities of the attack. Both the single-tweakey and related-tweakey settings are considered. In the related-tweakey setting, multiple impossible differential characteristics may share the same activeness pattern. We formalize the use of these multiple characteristics to effectively reduce the data complexity. The application of the IDMA framework to SKINNY and ForkSKINNY demonstrates that when $k = 3n$ or $k = 4n$, our approach achieves more effective attacks compared to traditional impossible differential attacks.

4.1 The Description of SKINNY and ForkSKINNY

4.1.1 Description of SKINNY

SKINNY is a family of lightweight block ciphers proposed by Beierle *et al.* at CRYPTO 2016 [35], which employs the TWEAKEY framework from [36]. SKINNY has two block sizes, $n \in \{64, 128\}$, and for each block size, it provides three main tweakey size versions, $t \in \{n, 2n, 3n\}$. SKINNY-n-t represents SKINNY with n-bit block size and t-bit tweakey size. The internal state of SKINNY is a 4×4 array of cells arranged in a row-major order, where each cell is a nibble (in the $n = 64$ case) or a byte (in the $n = 128$ case). For simplicity, we use c to denote the cell size. The tweakey state of SKINNY is viewed as a collection of z 4×4 square arrays of cells, where $z = t/n \in \{1, 2, 3\}$ denotes the tweakey size to block size ratio. These arrays are denoted as $TK1$ when $z = 1$, $TK1$ and $Tk2$ when $z = 2$, and finally $TK1$, $TK2$ and $TK3$ when $z = 3$.

As shown in Figure 3, the round function of SKINNY is composed of five transformations as follows.

1. SubCells (SC) - A 4-bit (resp. 8-bit) S-box is applied to every cells of the cipher internal state when n is 64 (resp. n is 128).
2. AddConstants (AC) - This step adds constants to the internal state using the bitwise exclusive-or (XOR).

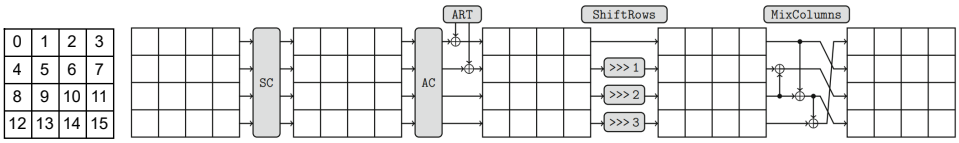

Fig. 3: The ordering of state cells and the SKINNY round function [35].

3. AddRoundTweakey (ART) - The first two rows of the internal state absorb the first two rows of STK, where $STK = \bigoplus_{i=1}^{z} TKi$.
4. ShiftRows (SR) - Each cell in row j is rotated to the right by j cells.
5. MixColumns (MC) - Each column of the internal state is multiplied by the matrix M. The matrix M and its inverse are as follows.

$$M = \begin{pmatrix} 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 \end{pmatrix}, M^{-1} = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 \end{pmatrix}$$

Tweakey schedule of SKINNY. The master tweakey state of SKINNY is split into z 4×4 square arrays of cells, denoted as $(TK1, ..., TKz)$, where $z \in \{1, 2, 3\}$. The tweakey arrays in round $r \geq 0$ are represented as $(TK1_r, ..., TKz_r)$, and the subtweakey $STK_r = \bigoplus_{i=1}^{z} TKi_r$. For $r \geq 1$, TKi_r is generated in two steps:

- First, apply the permutation $P = [9, 15, 8, 13, 10, 14, 12, 11, 0, 1, 2, 3, 4, 5, 6, 7]$ to each tweakey array: $TKj_r[i] \leftarrow TKj_{(r-1)}[P[i]]$, where $1 \leq j \leq z, 0 \leq i \leq 15$;
- Then, apply an $LFSR$ to update each cell of the first and second rows of TKj_r for $j = 2, 3$.

For more detailed information about SKINNY, please refer to [35].

SKINNYe-v2. At EUROCRYPT 2020, Naito *et al.* extended SKINNY-64 to create SKINNYe-64-256 [37]. Due to security concerns raised by Thomas Peyrin, an updated version, SKINNYe-v2, was proposed in 2020 [38], which had the same round functions as SKINNY. SKINNYe-v2 employs the same TWEAKEY framework as SKINNY but has a 256-bit tweakey state. The tweakey schedule of SKINNYe-v2 splits the master tweakey into 4 tweakey arrays, denoted as $TK1, TK2, TK3$, and $TK4$. The tweakey state is updated as follows. First, the same permutation P of SKINNY is applied to each tweakey array. Then, every nibble of the first two rows of $TK2, TK3$ and $TK4$ is individually updated with the corresponding LFSR. For more detailed information about SKINNYe-v2, please refer to [38].

4.1.2 Description of ForkSKINNY

ForkSKINNY is a tweakable block cipher proposed by Andreeva *et al.* [39] at ASIACRYPT 2019, which follows the Forkcipher design strategy and employs SKINNY as its underlying primitive. As shown in Figure 4, ForkSKINNY takes n-bit plaintext as input and generates $2n$-bit ciphertext. We use the same notation as in [18], with r_i

denoting the number of rounds before the fork point, and r_0 and r_1 representing the number of rounds in the C_0 and C_1 branches, respectively. According to the underlying SKINNY variant and the parameters (r_i, r_0, r_1), ForkSKINNY has five variants. We focus on two variants of it, as listed in Table 4. Each ForkSKINNY was designed to have a 128-bit key. However, we allow the key size as large as the tweakey size. Extending the key to more than 128 bits definitely leads to a variant that differs from the original one. However, the security analysis of the variant with a longer key may help define the upper bounds for the original cipher. For ForkSKINNY, it has been analyzed with its key extended to 192 bits or 256 bits in [12, 18].

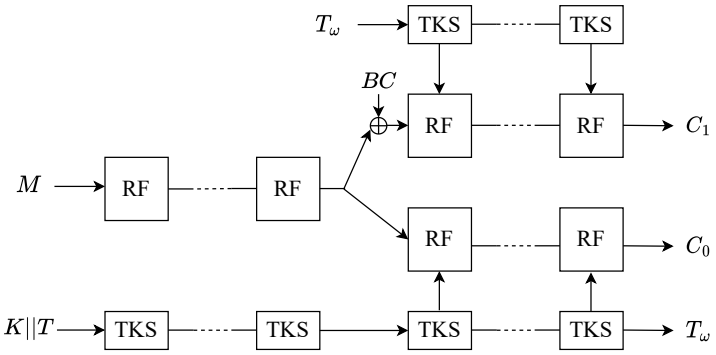

Fig. 4: The Forkcipher framework

Table 4: The two ForkSKINNY variants investigated in this work.

Variant	Block size	Tweakey size	r_i	r_0	r_1
ForkSKINNY-64-192	64	192	17	23	23
ForkSKINNY-128-384	128	384	25	31	31

According to the designers, a reduced-round instance of ForkSKINNY should decrease the number of rounds equally from all three branches. This configuration is referred to as the limited setting, whereas the arbitrary setting does not impose this requirement. For more detailed information about ForkSKINNY, please refer to [39].

4.2 Searching for Attacks on SKINNY and ForkSKINNY

Key recovery attacks based on the meet-in-the-middle technique are typically advantageous when the key size is at least twice the block size. In this section, we aim to apply the impossible differential meet-in-the-middle attack framework (IDMA) to variants of SKINNY and ForkSKINNY with large tweakey sizes and try to find more efficient attacks than those achieved through traditional key recovery methods. To accomplish this, we build Constraint Programming (CP) models specifically designed for searching

impossible differential attacks. These models not only specify the underlying impossible differential distinguisher but also optimize the time complexity of the key recovery attack.

4.2.1 Model for Searching Impossible Differential Meet-in-the-Middle Attacks

Goal. According to our `IDMA` framework and the formula for the time complexity, the core parameters that directly influence the complexities of impossible differential attacks are c_b, c_f, $|k_b|$, $|k_f|$, and $|k_b \cup k_f|$. First, it is crucial that $|k_b \cup k_f|$ remains smaller than k, as T_3 is bounded by $|k_b \cup k_f|$. Additionally, we have observed that balancing $|k_b|$ and $|k_f|$ as evenly as possible leads to more efficient impossible differential attacks using the meet-in-the-middle technique. This also implies that the number of rounds extended on both sides of the impossible differential distinguisher needs to be balanced. Specifically, the goal of the model is to identify the impossible differential attacks that either cover more rounds or have lower time complexity by maintaining a balance between $|k_b|$ and $|k_f|$ while ensuring $|k_b \cup k_f| < k$.

Model. The impossible differential attack typically involves two main steps: first, identifying an impossible distinguisher over a set of rounds, and second, extending additional rounds around the distinguisher to perform a key recovery attack. However, the best impossible differential distinguisher does not always result in a key recovery attack with optimal complexities, as demonstrated in many previous studies [27, 40]. To address this issue, it is more effective to integrate the impossible differential distinguisher and the extended parts into a unified approach, enabling the search for the most efficient key recovery attacks.

Next, we outline how to build our models for searching impossible differential attacks dedicated to `IDMA`. The model not only covers the distinguisher but also the extended parts, which contribute to the core parameters c_b, c_f, $|k_b|$, $|k_f|$ and $|k_b \cup k_f|$. While this might initially appear complex, it is made more manageable by leveraging an efficient CP-based tool recently proposed by Hadipour *et al.* [11, 12]. Their tool exploits Minizinc [41] and is designed for traditional key recovery, focusing primarily on the size of $|k_b \cup k_f|$ and other related parameters. However, in our attacks, the individual sizes of k_b and k_f are also of significant importance.

To address these concerns, we change Hadipour *et al.*'s model and replace the key recovery part with a new one that captures k_b and k_f and reflects the time complexities unique to our new attack strategy, as shown below. To ensure compatibility with MiniZinc, the complexities are evaluated by their exponents. This results in a model that is fully aligned with our impossible differential attack framework. The source codes of our model can be accessed via this link. By running this updated model, we can achieve key recovery attacks that are optimized for the new impossible differential attack methodology.

$$
\begin{aligned}
&D_0 := c_b + c_f + z + n + 1 - r_b - r_f, 2^z = x \ln 2,\\
&D_1 := \min_{r \in \{r_b, r_f\}} \{(c_b + c_f + z + n - r + 1)/2\},\\
&T_0 := \max\{D_0, D_1\},\\
&T_1 := \max\{|k_b|, |k_f|\} + T_0,\\
&T_2 := \max\{c_f + z + |k_b|, c_b + z + |k_f|\},\\
&T_3 := |k_b \cup k_f| + z, |k_b \cup k_f| < k,\\
&T_4 := k - x,\\
&T := \max\{T_0, T_1, T_2, T_3, T_4\}.
\end{aligned}
$$

4.3 Multiple Impossible Differential Characteristics in the Related-Tweakey Setting

In the related-tweakey setting, an impossible differential characteristic of `SKINNY` and `ForkSKINNY` is identified by its input/output activeness pattern and the tweakey activeness pattern, where the tweakey difference typically determines the specific input difference and output difference of the impossible differential characteristic. When the tweakey difference varies, this results in a set of impossible differential characteristics. Note in a key recovery attack, the same k_b and k_f are involved for any impossible differential characteristic in this set due to the same input/output activeness pattern. Such multiple impossible differential characteristics are different from those in the single-key setting as discussed in [16]. Suppose that the number of possible tweakey differences is 2^{r_k}. What advantages can we gain from utilizing such a set of impossible differential characteristics?

4.3.1 How to Use Multiple Impossible Differential Characteristics

Plaintext structures are a common technique in differential-like cryptanalysis, enabling an efficient generation of plaintext pairs by enjoying the birthday effect. In the key recovery attack illustrated in Figure 1, r_b bits of the plaintext are activated, which directly determines the size of the plaintext structures. With a set of 2^{r_k} impossible differential characteristics, the allowable size for a structure can be expanded from 2^{r_b} to $2^{r_b+r_k}$ if the tweakey difference is considered as part of the input. This expansion can result in a reduction in data complexity.

4.3.2 Improvement on the Data Complexity

Recall that $N = 2^{c_b+c_f+z}$. In the related-tweakey setting where impossible differential characteristics are employed, we have $c_b = r_b$ and $c_f = r_f$, leading to $N = 2^{r_b+r_f+z}$. Let $N_0 = 2^{2r_b+r_f-n}$ be the number of pairs from two original structures (under two related tweakeys) that satisfy both the plaintext and ciphertext differences. Similarly, let $N_1 = 2^{2r'_b-1+r_f-n}$ be the number of pairs from an expanded structure that meet both the plaintext and ciphertext differences. It is evident that $N_0 < N$.

- When $N > N_1$, multiple expanded structures are required and $D = N \cdot 2^{n+1-r'_b-r_f}$;

- When $N \leq N_1$, a partial structure of size D between 2^{r_b} and $2^{r'_b}$ is needed with $D = \sqrt{N \cdot 2^{n+1-r_f}}$.
  Without the expansion, it would require N/N_0 pairs of original structures, *i.e.*, $D = \frac{N}{N_0} \cdot 2^{r_b+1} = N \cdot 2^{n+1-r_b-r_f}$, which is greater than $\sqrt{N \cdot 2^{n+1-r_f}}$[1].

Therefore, the data complexity of the attack, when multiple impossible difference characteristics with the same activeness pattern are used, is given by

$$D = \max\{N \cdot 2^{n+1-r'_b-r_f}, \sqrt{N \cdot 2^{n+1-r_f}}\}.$$

Example 1 illustrates the use of multiple impossible differential characteristics. Concrete applications can be found in attacks on `SKINNY`, `SKINNYe`-v2, and `ForkSKINNY` in the the related-tweakey setting.

Example 1. *Suppose* $c_b = 12c$, $c_f = 15c$, $r_k = 2c$, $r_b = 12c$, $r_f = 15c$, *where* c *denotes the cell size, and* n *denotes the block size. Then, the data complexity of the key recovery attack is*

$$r'_b = r_b + r_k, \quad N = 2^{c_b+c_f} \cdot x \cdot \ln 2,$$
$$D = \max\{N \cdot 2^{n+1-r'_b-r_f}, \sqrt{N \cdot 2^{n+1-r_f}}\} = 2^{n+1-r_k} \cdot x \ln 2,$$

instead of $D = \max\{N \cdot 2^{n+1-r_b-r_f}, \sqrt{N \cdot 2^{n+1-r_f}}\} = 2^{n+1} \cdot x \ln 2$.

4.4 Requirements for Successful Attacks

Beyond codebook attacks. In impossible differential attacks where a single characteristic is used, $r_b = c_b, r_f = c_f$ and thus the data complexity is beyond 2^n according to Equation (1). Even if multiple characteristics are used to reduce data, the required data may still exceed 2^n. While one could argue about the applicability of attacks with $D > 2^n$, such attacks may still play an important role in the security analysis, especially for tweakable block ciphers. For example, such attacks have shown to be powerful and realistic against real-world tweakable block cipher FPE [42].

The time complexity for the pure exhaustive search. In the single-key setting, if the required data exceeds 2^n, we need to turn some key bits, say t bits, into tweak bits so the time complexity of the exhaustive search becomes 2^{k-t}.

In the related-tweakey attacks on `SKINNY`, `SKINNYe`-v2, and `ForkSKINNY`, there is a freedom to choose a key difference of r_k bits. If $(P, K \oplus \Delta K)$ is different, we treat it as new data. Where multiple related keys are used, the time complexity should be compared to the exhaustive search of the secret key in the same scenario, as pointed out in [43]. If $2^{r'_k}$ ($r'_k \leq r_k$) related keys are used, the time complexity of the exhaustive search is $2^{k-r'_k}$ as we can test $2^{r'_k}$ keys at once. This is true if the cost of checking the equality of two messages $2^{r'_k}$ times is negligible when compared to one encryption.

- When the data complexity is $2^{n+r'_k}$ ($0 < r'_k \leq r_k$), the complexity of the exhaustive attack is $2^{k-\lceil r'_k \rceil}$.

[1] $N \cdot 2^{n+1-r_b-r_f} / \sqrt{N \cdot 2^{n+1-r_f}} = 2^{(n+1+z-r_b)/2} > 1$.

- When the required data exceeds 2^{n+r_k}, we have to set some key bits as tweak bits, say t bits, to get sufficient data. The complexity of an exhaustive search in this case is 2^{k-t-r_k}.

A successful attack should have a time complexity lower than that of the pure exhaustive search.

4.5 Improved Impossible Differential Attacks

We apply the IDMA to SKINNY, SKINNYe-v2 and ForkSKINNY. Furthermore, we employ the isolate-and-unite technique to skip some outer key bits in the meet-in-the-middle procedure to make some attacks that are invalid by just using the IDMA still work. To be specific, it is applied to the attacks on 23-round SKINNY-n-3n, 28-round SKINNY-n-3n, 33-round SKINNYe-v2, 30-round ForkSKINNY-n-3n and 33-round ForkSKINNY-n-3n. The main results are summarized as follows.

- For SKINNY-n-3n, we obtain enhanced impossible differential attacks on 21 rounds in the single-tweakey setting and successfully extend the attack to 22/23 rounds. Meanwhile, we improve impossible differential attacks on 27-round SKINNY-n-3n in the related-tweakey setting and extend the attack to 28 rounds.
- For SKINNYe-v2, we achieve, for the first time, impossible differential attacks on 23, 24 and 25 rounds in the single-tweakey setting, while in the related-tweakey setting, we obtain an improved impossible differential attack on 31 rounds and extend the attack to 33 rounds.
- Since ForkSKINNY shares the same result as SKINNY in the single-tweakey setting, our focus shifts to impossible differential attacks on ForkSKINNY-n-3n in the related-tweakey setting. We provide improved related-tweakey impossible differential attacks on 28-round ForkSKINNY-n-3n and successfully extend the attack to 30 rounds in the limited setting. We also present improved related-tweakey impossible differential attacks on 32-round ForkSKINNY-n-3n in the arbitrary setting and extend the impossible differential attack by 1 round.

Next, we describe in detail the impossible differential attacks on 22-round SKINNY-n-3n in the single-tweakey setting and 30-round ForkSKINNY-64-192 with a 192-bit key ($r_i = 12, r_0 = 18, r_1 = 18$) in the related-tweakey setting. Also, we discuss the application of IDMA to SKINNY and ForkSKINNY when $k = 2n$. Other impossible differential attacks on SKINNY and ForkSKINNY are postponed to Appendices A, B and C.

4.5.1 A 22-round ID Attack on SKINNY-n-3n in the Single-Tweakey Setting

In this subsection, we present the impossible differential attacks on 22-round SKINNY-n-3n that take an 11-round truncated impossible differential characteristic and extend it by five rounds in the backward direction and six rounds in the forward direction, as shown in Figure 5. For the impossible differential, the white cell and pink cell denote zero difference and nonzero difference, respectively. The state cell $X_{11}[13]$ from the forward direction and the state cell $X_{11}[13]$ from the backward direction will produce a contradiction, as illustrated in Figure 5.

Fig. 5: ID attacks on 22-round SKINNY-n-3n. $c_b = 15c, c_f = 15c, r_b = 16c, r_f = 16c, |k_b \cup k_f| = 47c, |k_b| = 26c, |k_f| = 30c$.

Based on the IDMA framework, we need to produce enough message pairs that satisfy the input difference and output difference to mount impossible differential attacks. According to the parameters illustrated in Figure 5 and the complexity analysis described in Section 3.2, the number of required message pairs is $N = 2^{c_b+c_f+z} =$

2^{30c+z}. Thus, the data complexity is as follows.

$$D = \max\{\sqrt{N \cdot 2^{n+1-r_f}}, N \cdot 2^{n+1-r_b-r_f}\} = \sqrt{N \cdot 2^{n+1-r_f}} = \sqrt{2^{30c+1+z}}$$

With the number of message pairs N, the data complexity D and other parameters shown in Figure 5, the complexities of the impossible differential attacks on 22-round SKINNY-n-3n are as follows.

- The data complexity is $D = \sqrt{2^{30c+1+z}}$.
- The time complexity:

$$\begin{aligned}
T_0 &= D = \sqrt{2^{30c+1+z}},\\
T_1 &= D \cdot (2^{|k_b|} \cdot 5/22 + 2^{|k_f|} \cdot 6/22) = \sqrt{2^{30c+1+z}} \cdot (2^{26c} \cdot 5/22 + 2^{30c} \cdot 6/22),\\
T_2 &= N \cdot \{2^{|k_b|-c_b} + 2^{|k_f|-c_f}\} = 2^{41c+z} + 2^{45c+z},\\
T_3 &= 2^{|k_b \cup k_f|+z} = 2^{47c+z},\\
T_4 &= 2^{k-x} = 2^{48c-x}.
\end{aligned}$$

- The memory complexity is $M = \max\{D, N2^{|k_b|-c_b}, 2^{|k_b \cup k_f|}\}$.

Specifically, for the impossible differential attack on 22-round SKINNY-64-192, we select $(x, z) = (3.01, 1.06)$ to optimize the complexity. Thus, the data and memory complexities are $2^{61.03}$ and 2^{188}; the time complexity includes $2^{189.06}$ memory accesses and $2^{188.99}$ encryptions.

Additionally, for the impossible differential attack on 22-round SKINNY-128-384, choosing $(x, z) = (5.78, 2)$ leads to the data and memory complexities of $2^{121.5}$ and 2^{376} respectively, and the time complexity of 2^{378} memory accesses and $2^{378.22}$ encryptions.

4.5.2 A 30-Round ID Attack on ForkSKINNY-n-3n in the Related-Tweakey Setting

We present an impossible differential attack on 30-round ForkSKINNY-n-3n in the related-key setting, as shown in Figure 6, where a cluster of 19-round impossible differential characteristics is used and six and five rounds are extended in backward and forward directions, respectively.

As illustrated in Figure 6, ◪ denotes the forward direction of the difference propagation while ◪ represents the backward direction of the difference propagation. The pink cell and white cell denote the nonzero difference and zero difference, respectively. Indeed, a valid impossible differential distinguisher needs to ensure at least one inconsistency occurs between the two propagations. Both pink and white occur in a state cell means an inconsistency happens in the impossible differential distinguisher, such as the state cell $Z_{16}[15]$.

This attack follows the limited setting with $r_i = 12$ and $r_0 = r_1 = 18$, reducing the number of rounds from three branches by the same amount. Since the subtweakey difference $\Delta STK_6[1]$ can be any nonzero value, there are $2^{r_k} = 2^c - 1$ impossible differential characteristics of the same active pattern. Other parameters of our attack

Fig. 6: ID attack on 30-round `ForkSKINNY`-n-$3n$. $c_b = 16c, c_f = 16c, r'_b = 17c (r_b = 16c), r_f = 16c, |k_b \cup k_f| = 47c, |k_b| = 34c, |k_f| = 29c$

are as follows:

$$c_b = 16c, c_f = 16c, r_b = 16c, r'_b = 17c, r_f = 16c, |k_b \cup k_f| = 47c, |k_b| = 34c, |k_f| = 29c.$$

According to the complexity analysis described in Section 3.2, the number of required message pairs that satisfy both the plaintext and ciphertext difference is $N = 2^{c_b+c_f+z} = 2^{32c+z}$; the data complexity is

$$D = \max\{N \cdot 2^{n+1-r'_b-r_f}, \sqrt{N \cdot 2^{n+1-r_f}}\} = \sqrt{N \cdot 2^{n+1-r_f}} = \sqrt{2^{32c+1+z}}.$$

The time complexity of partial encryption and decryption is

$$T_1 = D \cdot 2^{|k_b|} \cdot \frac{6}{30} + D \cdot 2^{|k_f|} \cdot \frac{5}{30} \approx D \cdot 2^{34c} \cdot \frac{6}{30} > 2^{49c} > 2^k.$$

Unfortunately, the time complexity is larger than that of an exhaustive search, making the attack invalid. However, employing a flexible key guessing strategy, the impossible differential attacks on 30-round `ForkSKINNY`-n-$3n$ can still work under the `IDMA` framework.

Utilizing the isolate-and-unite technique. We peel off $\texttt{ART} \circ \texttt{AC} \circ \texttt{SC}$ of the first round on the cells at the positions $\{0, 1, 2, 3, 6\}$, which involves $k_b^* = STK_0[0, 1, 2, 3, 6]$. The meet-in-the-middle procedure is then performed on the remaining part of the encryption. In Figure 6, each subtweakey cell is labeled with a hex number from $0x0$ to $0xf$, and all subtweakey cells with the same label are computed from three individual cells of the master tweakey with a linear function, according to the tweakey schedule of `SKINNY`. Thus, if three cells with the same label in the subtweakeys are guessed, then the other cells with the same label can be derived. Therefore, it can be seen from Table 5 that k_b^* can be fully derived from k'_b and k_f, *i.e.*, $k_b^* = g_1(k'_b) \oplus g_2(k_f)$ are cost-free isolable key bits with certain linear functions g_1, g_2 that can be derived from the tweakey schedule.

Table 5: The isolable key cells of the attack on 30-round `ForkSKINNY`-n-$3n$

k_b^*	k'_b	k_f
$STK_0[0]$	$STK_2[2]$	$STK_{28}[1]$, $STK_{26}[7]$
$STK_0[1]$	$STK_2[0]$, $STK_4[2]$	$STK_{28}[7]$
$STK_0[2]$	$STK_2[4]$, $STK_4[6]$	$STK_{28}[0]$, $STK_{26}[1]$
$STK_0[3]$	$STK_2[1]$, $STK_4[1]$	$STK_{28}[5]$, $STK_{26}[6]$
$STK_0[6]$	$STK_2[5]$	$STK_{28}[4]$, $STK_{26}[2]$
	...	...

As a result, only $k'_b = k_b \setminus k_b^*$ and k_f are involved in the MITM procedure and k_b^* is taken into account in the matching process. Then the parameters of our impossible differential attack on 30-round `ForkSKINNY`-n-$3n$ are as follows:

$$c_b = 16c, c_f = 16c, r_b = 16c, r'_b = 17c, r_f = 16c, |k'_b \cup k_f| = 47c, |k'_b| = 29c, |k_f| = 29c.$$

Thus, the time complexity is as follows.

$$
\begin{aligned}
T_0 &= D = 2^{16c+\frac{1+z}{2}},\\
T_1 &= (2^{|k_b'|} \cdot 6/30 + 2^{|k_f|} \cdot 5/30) \cdot D = 2^{45c+\frac{1+z}{2}} \cdot (2/5),\\
T_2 &= N(2^{|k_b'|-c_b} + 2^{|k_f|-c_f}) = 2^{45c+z+1},\\
T_3 &= 2^{|k_b \cup k_f|+z} = 2^{47c+z},\\
T_4 &= 2^{k-x} = 2^{48c-x},\\
M &= \max\{D, N2^{|k_f|-c_f}, 2^{|k_b \cup k_f|}\} = 2^{|k_b \cup k_f|}.
\end{aligned}
$$

For ForkSKINNY-64-192 with a 192-bit key, $c = 4$ and we choose $(x, z) = (3.01, 1.06)$. The data and memory complexities are $2^{65.03}$ and 2^{188}; the time complexity consists of $2^{188.99}$ encryption and $2^{189.06}$ memory accesses.

As described in Section 4.4, if $2^{r_k'}$ ($r_k' \leq r_k$) related keys are used, the time complexity of the exhaustive search is $2^{k-r_k'}$ as we can test $2^{r_k'}$ keys at once. We need to use 2^2 related keys to generate more data as the data complexity $2^{65.03}$ exceeds 2^{64}. Thus, in this case, the complexity of the exhaustive attack is reduced from 2^{192} to 2^{190}. Our attack is valid since its time complexity $2^{188.99}$ is lower than 2^{190}.

For ForkSKINNY-128-384 with a 384-bit key, $c = 8$ and we choose $(x, z) = (5.78, 2)$. The data and memory complexities are $2^{129.5}$ and 2^{376}; the time complexity consists of $2^{378.22}$ encryption and 2^{378} memory accesses.

Similarly, we need to use 2^2 related keys to generate more data since the data complexity $2^{129.5}$ is larger than 2^{128}. Thus, the complexity of the exhaustive attack changes from 2^{384} to 2^{382}. Our attack is valid, as its time complexity $2^{378.22}$ is lower than 2^{382}.

4.5.3 Impossible Differential Attacks on SKINNY-n-$2n$ and ForkSKINNY-n-$2n$

We also searched for impossible differential attacks on SKINNY-n-$2n$ and ForkSKINNY-n-$2n$, but failed to find more efficient attacks than using the traditional key recovery method. For example, for SKINNY-n-$2n$, we can find an impossible differential attack on 19 rounds in the single-tweakey setting, where both four rounds are added before and after an 11-round truncated impossible differential. The parameters are as follows

$$c_b = 11c, c_f = 11c, r_b = 12c, r_f = 12c, |k_b \cup k_f| = 28c, |k_b| = 18c, |k_f| = 14c.$$

For SKINNY-64-128, a key recovery attack on 19 rounds can be obtained with data and time complexities $2^{59.93}$ and $2^{118.38}$. However, the best previous impossible differential attack on 19-round SKINNY-64-128 [11] has data and time complexities $2^{60.86}$ and $2^{110.34}$. That is, the IDMA framework is unable to reduce the time complexity.

In impossible differential attacks, the time complexity is bounded by $2^{|k_b \cup k_f|+z}$ regardless of the methods for key recovery. In the IDMA framework, balanced k_b and k_f are desirable. However, balancing k_b and k_f typically leads to a large $|k_b \cup k_f|$ when the ratio of the key size to the block size is 2. On the contrary, if no restriction is imposed

on the size of k_b and k_f, $|k_b \cup k_f|$ is usually smaller. Particularly, for SKINNY-n-2n, traditional impossible differential attacks can be found with $|k_b \cup k_f| = 26c < 28c$. That is why the IDMA framework does not outperform the traditional impossible differential attack on SKINNY-n-2n and ForkSKINNY-n-2n.

5 Conclusion

In this paper, we propose a new impossible differential meet-in-the-middle attack framework, denoted as IDMA. We apply our new key recovery framework to the block ciphers SKINNY, SKINNYe, and ForkSKINNY, obtaining better results. For SKINNY-n-3n, we present 23-round attacks in the single-tweakey setting and 28-round attacks in the related-tweakey setting, which are respectively 2 and 1 rounds more than those of previous works. For SKINNYe-v2, we obtain a 33-round impossible differential attack with related tweakeys and a 25-round impossible differential attack with a single tweakey. For ForkSKINNY-64-192, we improve by 2 rounds in the limited setting and give 30-round impossible differential attacks. In another situation, we improve by 1 round and give 33-round impossible differential attacks. For ForkSKINNY-128-384, we achieve the first 30-round related-tweakey impossible differential attack with a limit setting and the first 33-round related-tweakey impossible differential attack without the limit setting.

Acknowledgements. This work was partially supported by the National Natural Science Foundation of China under Grant Nos. 62132008, 62372213, 62202460.

Declarations

The authors declare that they have no competing interests.

References

[1] Biham, E., Shamir, A.: Differential cryptanalysis of DES-like cryptosystems. Journal of CRYPTOLOGY **4**(1), 3–72 (1991)

[2] Wagner, D.: The boomerang attack. In: International Workshop on Fast Software Encryption, pp. 156–170 (1999). https://doi.org/10.1007/3-540-48519-8_12 . Springer. https://doi.org/10.1007/3-540-48519-8_12

[3] Lai, X.: In: Blahut, R.E., Costello, D.J., Maurer, U., Mittelholzer, T. (eds.) Higher Order Derivatives and Differential Cryptanalysis, pp. 227–233. Springer, Boston, MA (1994). https://doi.org/10.1007/978-1-4615-2694-0_23 . https://doi.org/10.1007/978-1-4615-2694-0_23

[4] Biham, E., Biryukov, A., Shamir, A.: Cryptanalysis of skipjack reduced to 31 rounds using impossible differentials. In: Advances in Cryptology—EUROCRYPT'99: International Conference on the Theory and Application of Cryptographic Techniques Prague, Czech Republic, May 2–6, 1999 Proceedings 18, pp. 12–23 (1999). Springer

[5] Knudsen, L.: Deal-a 128-bit block cipher. complexity **258**(2), 216 (1998)

[6] Mala, H., Dakhilalian, M., Rijmen, V., Modarres-Hashemi, M.: Improved impossible differential cryptanalysis of 7-round AES-128. In: Gong, G., Gupta, K.C. (eds.) Progress in Cryptology - INDOCRYPT 2010 - 11th International Conference on Cryptology in India, Hyderabad, India, December 12-15, 2010. Proceedings. Lecture Notes in Computer Science, vol. 6498, pp. 282–291 (2010). Springer

[7] Boura, C., Lallemand, V., Naya-Plasencia, M., Suder, V.: Making the impossible possible. J. Cryptol. **31**(1), 101–133 (2018) https://doi.org/10.1007/S00145-016-9251-7

[8] Sun, S., Hu, L., Wang, P., Qiao, K., Ma, X., Song, L.: Automatic security evaluation and (related-key) differential characteristic search: Application to Simon, PRESENT, LBlock, DES(L) and other bit-oriented block ciphers. In: Sarkar, P., Iwata, T. (eds.) Advances in Cryptology - ASIACRYPT 2014 - 20th International Conference on the Theory and Application of Cryptology and Information Security, Kaoshiung, Taiwan, December 7-11, 2014. Proceedings, Part I. Lecture Notes in Computer Science, vol. 8873, pp. 158–178 (2014). Springer

[9] Sasaki, Y., Todo, Y.: New impossible differential search tool from design and cryptanalysis aspects - revealing structural properties of several ciphers. In: Coron, J., Nielsen, J.B. (eds.) Advances in Cryptology - EUROCRYPT 2017 - 36th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Paris, France, April 30 - May 4, 2017, Proceedings, Part III. Lecture Notes in Computer Science, vol. 10212, pp. 185–215 (2017). https://doi.org/10.1007/978-3-319-56617-7_7 . https://doi.org/10.1007/978-3-319-56617-7_7

[10] Sun, L., Wang, W., Wang, M.: Accelerating the search of differential and linear characteristics with the SAT method. IACR Trans. Symmetric Cryptol. **2021**(1), 269–315 (2021) https://doi.org/10.46586/tosc.v2021.i1.269-315

[11] Hadipour, H., Sadeghi, S., Eichlseder, M.: Finding the impossible: Automated search for full impossible-differential, zero-correlation, and integral attacks. In: Hazay, C., Stam, M. (eds.) Advances in Cryptology - EUROCRYPT 2023 - 42nd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Lyon, France, April 23-27, 2023, Proceedings, Part IV. Lecture Notes in Computer Science, vol. 14007, pp. 128–157 (2023). Springer

[12] Hadipour, H., Gerhalter, S., Sadeghi, S., Eichlseder, M.: Improved search for integral, impossible differential and zero-correlation attacks application to Ascon, ForkSKINNY, SKINNY, MANTIS, PRESENT and QARMAv2. IACR Trans. Symmetric Cryptol. **2024**(1), 234–325 (2024) https://doi.org/10.46586/TOSC.V2024.I1.234-325

[13] Zhang, K., Wang, S., Lai, X., Wang, L., Guan, J., Hu, B., Shi, T.: Impossible differential cryptanalysis and a security evaluation framework for AND-RX ciphers. IEEE Trans. Inf. Theory **70**(8), 6025–6040 (2024) https://doi.org/10.1109/TIT.2023.3292241

[14] Biham, E., Shamir, A.: Differential cryptanalysis of the full 16-round DES. In: Brickell, E.F. (ed.) Advances in Cryptology - CRYPTO '92, 12th Annual International Cryptology Conference, Santa Barbara, California, USA, August 16-20, 1992, Proceedings. Lecture Notes in Computer Science, vol. 740, pp. 487–496 (1992). https://doi.org/10.1007/3-540-48071-4_34 . Springer. https://doi.org/10.1007/3-540-48071-4_34

[15] Lu, J., Kim, J., Keller, N., Dunkelman, O.: Improving the efficiency of impossible differential cryptanalysis of reduced Camellia and MISTY1. In: Malkin, T. (ed.) Topics in Cryptology - CT-RSA 2008, The Cryptographers' Track at the RSA Conference 2008, San Francisco, CA, USA, April 8-11, 2008. Proceedings. Lecture Notes in Computer Science, vol. 4964, pp. 370–386 (2008). https://doi.org/10.1007/978-3-540-79263-5_24 . Springer. https://doi.org/10.1007/978-3-540-79263-5_24

[16] Boura, C., Naya-Plasencia, M., Suder, V.: Scrutinizing and improving impossible differential attacks: Applications to CLEFIA, Camellia, LBlock and Simon. In: Sarkar, P., Iwata, T. (eds.) Advances in Cryptology - ASIACRYPT 2014 - 20th International Conference on the Theory and Application of Cryptology and Information Security, Kaoshiung, Taiwan, December 7-11, 2014. Proceedings, Part I. Lecture Notes in Computer Science, vol. 8873, pp. 179–199 (2014). https://doi.org/10.1007/978-3-662-45611-8_10 . Springer. https://doi.org/10.1007/978-3-662-45611-8_10

[17] Sadeghi, S., Mohammadi, T., Bagheri, N.: Cryptanalysis of reduced round SKINNY block cipher. IACR Trans. Symmetric Cryptol. **2018**(3), 124–162 (2018) https://doi.org/10.13154/TOSC.V2018.I3.124-162

[18] Bariant, A., David, N., Leurent, G.: Cryptanalysis of forkciphers. IACR Trans. Symmetric Cryptol. **2020**(1), 233–265 (2020) https://doi.org/10.13154/TOSC.V2020.I1.233-265

[19] Boura, C., David, N., Derbez, P., Leander, G., Naya-Plasencia, M.: Differential meet-in-the-middle cryptanalysis. In: Handschuh, H., Lysyanskaya, A. (eds.) Advances in Cryptology - CRYPTO 2023 - 43rd Annual International Cryptology Conference, CRYPTO 2023, Santa Barbara, CA, USA, August 20-24, 2023, Proceedings, Part III. Lecture Notes in Computer Science, vol. 14083, pp. 240–272 (2023). https://doi.org/10.1007/978-3-031-38548-3_9 . Springer. https://doi.org/10.1007/978-3-031-38548-3_9

[20] Song, L., Liu, H., Yang, Q., Chen, Y., Hu, L., Weng, J.: Generic differential key recovery attacks and beyond. Cryptology ePrint Archive (2024)

[21] Isobe, T., Shibutani, K.: All subkeys recovery attack on block ciphers: Extending meet-in-the-middle approach. In: Knudsen, L.R., Wu, H. (eds.) Selected Areas in Cryptography, 19th International Conference, SAC 2012, Windsor, ON, Canada, August 15-16, 2012, Revised Selected Papers. Lecture Notes in Computer Science, vol. 7707, pp. 202–221 (2012). https://doi.org/10.1007/978-3-642-35999-6_14 . Springer. https://doi.org/10.1007/978-3-642-35999-6_14

[22] Canteaut, A., Naya-Plasencia, M., Vayssière, B.: Sieve-in-the-middle: Improved MITM attacks. In: Canetti, R., Garay, J.A. (eds.) Advances in Cryptology - CRYPTO 2013 - 33rd Annual Cryptology Conference, Santa Barbara, CA, USA, August 18-22, 2013. Proceedings, Part I. Lecture Notes in Computer Science, vol. 8042, pp. 222–240 (2013). https://doi.org/10.1007/978-3-642-40041-4_13 . Springer. https://doi.org/10.1007/978-3-642-40041-4_13

[23] Yang, D., Qi, W., Chen, H.: Impossible differential attacks on the SKINNY family of block ciphers. IET Inf. Secur. **11**(6), 377–385 (2017) https://doi.org/10.1049/IET-IFS.2016.0488

[24] Ankele, R., Dobraunig, C., Guo, J., Lambooij, E., Leander, G., Todo, Y.: Zero-correlation attacks on tweakable block ciphers with linear tweakey expansion. IACR Trans. Symmetric Cryptol. **2019**(1), 192–235 (2019) https://doi.org/10.13154/TOSC.V2019.I1.192-235

[25] Bao, Z., Guo, J., Shi, D., Tu, Y.: Superposition meet-in-the-middle attacks: Updates on fundamental security of aes-like hashing. In: Dodis, Y., Shrimpton, T. (eds.) Advances in Cryptology - CRYPTO 2022 - 42nd Annual International Cryptology Conference, CRYPTO 2022, Santa Barbara, CA, USA, August 15-18, 2022, Proceedings, Part I. Lecture Notes in Computer Science, vol. 13507, pp. 64–93 (2022). https://doi.org/10.1007/978-3-031-15802-5_3 . Springer. https://doi.org/10.1007/978-3-031-15802-5_3

[26] Ahmadian, Z., Khalesi, A., M'foukh, D., Moghimi, H., Naya-Plasencia, M.: Improved differential meet-in-the-middle cryptanalysis. In: Joye, M., Leander, G. (eds.) Advances in Cryptology - EUROCRYPT 2024 - 43rd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Zurich, Switzerland, May 26-30, 2024, Proceedings, Part I. Lecture Notes in Computer Science, vol. 14651, pp. 280–309 (2024). https://doi.org/10.1007/978-3-031-58716-0_10 . Springer. https://doi.org/10.1007/978-3-031-58716-0_10

[27] Liu, G., Ghosh, M., Song, L.: Security analysis of SKINNY under related-tweakey settings (long paper). IACR Trans. Symmetric Cryptol. **2017**(3), 37–72 (2017) https://doi.org/10.13154/TOSC.V2017.I3.37-72

[28] Zhang, J., Wang, H., Tang, D.: Impossible boomerang attacks revisited applications to deoxys-bc, joltik-bc and SKINNY. IACR Trans. Symmetric Cryptol. **2024**(2), 254–295 (2024) https://doi.org/10.46586/TOSC.V2024.I2.254-295

[29] Dong, X., Qin, L., Sun, S., Wang, X.: Key guessing strategies for linear key-schedule algorithms in rectangle attacks. In: Dunkelman, O., Dziembowski, S. (eds.) Advances in Cryptology - EUROCRYPT 2022 - 41st Annual International Conference on the Theory and Applications of Cryptographic Techniques, Trondheim, Norway, May 30 - June 3, 2022, Proceedings, Part III. Lecture Notes in Computer Science, vol. 13277, pp. 3–33 (2022). https://doi.org/10.1007/978-3-031-07082-2_1 . Springer. https://doi.org/10.1007/978-3-031-07082-2_1

[30] Dong, X., Hua, J., Sun, S., Li, Z., Wang, X., Hu, L.: Meet-in-the-middle attacks revisited: Key-recovery, collision, and preimage attacks. In: Malkin, T., Peikert, C. (eds.) Advances in Cryptology - CRYPTO 2021 - 41st Annual International Cryptology Conference, CRYPTO 2021, Virtual Event, August 16-20, 2021, Proceedings, Part III. Lecture Notes in Computer Science, vol. 12827, pp. 278–308 (2021). https://doi.org/10.1007/978-3-030-84252-9_10 . Springer. https://doi.org/10.1007/978-3-030-84252-9_10

[31] Shi, D., Sun, S., Song, L., Hu, L., Yang, Q.: Exploiting non-full key additions: Full-fledged automatic demirci-selçuk meet-in-the-middle cryptanalysis of SKINNY. In: Hazay, C., Stam, M. (eds.) Advances in Cryptology - EUROCRYPT 2023 - 42nd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Lyon, France, April 23-27, 2023, Proceedings, Part IV. Lecture Notes in Computer Science, vol. 14007, pp. 67–97 (2023). https://doi.org/10.1007/978-3-031-30634-1_3 . Springer. https://doi.org/10.1007/978-3-031-30634-1_3

[32] Song, L., Zhang, N., Yang, Q., Shi, D., Zhao, J., Hu, L., Weng, J.: Optimizing rectangle attacks: A unified and generic framework for key recovery. In: Agrawal, S., Lin, D. (eds.) Advances in Cryptology - ASIACRYPT 2022 - 28th International Conference on the Theory and Application of Cryptology and Information Security, Taipei, Taiwan, December 5-9, 2022, Proceedings, Part I. Lecture Notes in Computer Science, vol. 13791, pp. 410–440 (2022). https://doi.org/10.1007/978-3-031-22963-3_14 . Springer. https://doi.org/10.1007/978-3-031-22963-3_14

[33] Aoki, K., Sasaki, Y.: Meet-in-the-middle preimage attacks against reduced SHA-0 and SHA-1. In: Halevi, S. (ed.) Advances in Cryptology - CRYPTO 2009, 29th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 16-20, 2009. Proceedings. Lecture Notes in Computer Science, vol. 5677, pp. 70–89 (2009). https://doi.org/10.1007/978-3-642-03356-8_5 . Springer. https://doi.org/10.1007/978-3-642-03356-8_5

[34] Ahmadian, Z., Khalesi, A., M'foukh, D., Moghimi, H., Naya-Plasencia, M.: Improved differential meet-in-the-middle cryptanalysis. In: Joye, M., Leander, G. (eds.) Advances in Cryptology - EUROCRYPT 2024 - 43rd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Zurich, Switzerland, May 26-30, 2024, Proceedings, Part I. Lecture Notes in Computer Science, vol. 14651, pp. 280–309 (2024). https://doi.org/10.1007/

978-3-031-58716-0_10 . Springer. https://doi.org/10.1007/978-3-031-58716-0_10

[35] Beierle, C., Jean, J., Kölbl, S., Leander, G., Moradi, A., Peyrin, T., Sasaki, Y., Sasdrich, P., Sim, S.M.: The SKINNY family of block ciphers and its low-latency variant MANTIS. In: Annual International Cryptology Conference, pp. 123–153 (2016). Springer

[36] Jean, J., Nikolic, I., Peyrin, T.: Tweaks and keys for block ciphers: The TWEAKEY framework. In: Sarkar, P., Iwata, T. (eds.) Advances in Cryptology - ASIACRYPT 2014 - 20th International Conference on the Theory and Application of Cryptology and Information Security, Kaoshiung, Taiwan, December 7-11, 2014, Proceedings, Part II. Lecture Notes in Computer Science, vol. 8874, pp. 274–288 (2014). https://doi.org/10.1007/978-3-662-45608-8_15 . Springer. https://doi.org/10.1007/978-3-662-45608-8_15

[37] Naito, Y., Sasaki, Y., Sugawara, T.: Lightweight authenticated encryption mode suitable for threshold implementation. In: Annual International Conference on the Theory and Applications of Cryptographic Techniques, pp. 705–735 (2020). Springer

[38] Naito, Y., Sasaki, Y., Sugawara, T.: Lightweight Authenticated Encryption Mode Suitable for Threshold Implementation. Cryptology ePrint Archive, Paper 2020/542. https://eprint.iacr.org/2020/542 (2020). https://eprint.iacr.org/2020/542

[39] Andreeva, E., Lallemand, V., Purnal, A., Reyhanitabar, R., Roy, A., Vizár, D.: Forkcipher: A new primitive for authenticated encryption of very short messages. In: Galbraith, S.D., Moriai, S. (eds.) Advances in Cryptology - ASIACRYPT 2019 - 25th International Conference on the Theory and Application of Cryptology and Information Security, Kobe, Japan, December 8-12, 2019, Proceedings, Part II. Lecture Notes in Computer Science, vol. 11922, pp. 153–182 (2019). https://doi.org/10.1007/978-3-030-34621-8_6 . Springer. https://doi.org/10.1007/978-3-030-34621-8_6

[40] Qin, L., Dong, X., Wang, X., Jia, K., Liu, Y.: Automated search oriented to key recovery on ciphers with linear key schedule applications to boomerangs in SKINNY and ForkSKINNY. IACR Trans. Symmetric Cryptol. **2021**(2), 249–291 (2021) https://doi.org/10.46586/tosc.v2021.i2.249-291

[41] Nethercote, N., Stuckey, P.J., Becket, R., Brand, S., Duck, G.J., Tack, G.: Minizinc: Towards a standard CP modelling language. In: Bessiere, C. (ed.) Principles and Practice of Constraint Programming - CP 2007, 13th International Conference, CP 2007, Providence, RI, USA, September 23-27, 2007, Proceedings. Lecture Notes in Computer Science, vol. 4741, pp. 529–543. Springer, ??? (2007). https://doi.org/10.1007/978-3-540-74970-7_38 . https://doi.org/10.1007/978-3-540-74970-7_38

[42] Bellare, M., Hoang, V.T., Tessaro, S.: Message-recovery attacks on feistel-based format preserving encryption. In: Weippl, E.R., Katzenbeisser, S., Kruegel, C., Myers, A.C., Halevi, S. (eds.) Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security, Vienna, Austria, October 24-28, 2016, pp. 444–455 (2016). https://doi.org/10.1145/2976749.2978390 . ACM. https://doi.org/10.1145/2976749.2978390

[43] Bonnetain, X., Cordero, M., Lallemand, V., Minier, M., Naya-Plasencia, M.: On impossible boomerang attacks application to simon and skinnyee. IACR Trans. Symmetric Cryptol. **2024**(2), 222–253 (2024) https://doi.org/10.46586/TOSC.V2024.I2.222-253

A Other ID Attacks on SKINNY-n-3n

A.1 ID Attacks on 21-round SKINNY-n-3n in the Single-Tweakey Setting

For the impossible differential attack on 21-round SKINNY-64-192, we select $(x, z) = (20.97, 3.86)$ for $c = 4$. The data and memory complexities of the impossible differential attack on 21-round SKINNY-64-192 are $2^{62.43}$ and 2^{168}; the time complexity includes $2^{171.86}$ memory accesses and $2^{171.03}$ encryptions.

Similarly, for the impossible differential attack on 21-round SKINNY-128-384, we select $(x, z) = (39.67, 4.78)$ for the case of $c = 8$ to optimize the complexities. As a result, the data and memory complexities are $2^{122.89}$ and 2^{336}; the time complexity consists of $2^{340.78}$ memory accesses and $2^{344.33}$ encryptions.

A.2 ID Attacks on 23-round SKINNY-n-3n in the Single-Tweakey Setting

According to the parameters shown in Figure 8, the time complexity is larger than that of an exhaustive search. However, the attack can still work by employing the method described in Section 4.5.2. Specifically, all the involved subtweakey cells are guessed except $STK_0[0, 1, 2, 3]$. Then, $STK_0[0, 1, 2, 3]$ can be deduced since the subtweakey cells labeled as $\{0, 1, 2, 3\}$ appear at least three times in the guessed subtweakey cells. As a result, the actual parameters of our impossible differential attacks on 23-round SKINNY-n-3n are as follows:

$$c_b = 15c, c_f = 15c, r_b = 16c, r_f = 16c, |k'_b \cup k_f| = 47c, |k'_b| = 30c, |k_f| = 30c$$

For the impossible differential attack on 23-round SKINNY-64-192, we select $(x, z) = (3.01, 1.06)$ for the case of $c = 4$, resulting in the data and memory complexities of $2^{61.03}$ and 2^{188} respectively, and time complexity of $2^{189.06}$ memory accesses and $2^{188.99}$ encryptions.

For the impossible differential attack on 23-round SKINNY-128-384, selecting $(x, z) = (5.78, 2)$ for the case of $c = 8$ leads to the data and memory complexities of $2^{121.50}$ and 2^{376} respectively, and time complexity of $2^{378.22}$ memory accesses and 2^{376} encryptions.

A.3 ID Attacks on 27-round SKINNY-n-3n in the Related-Tweakey Setting

For the impossible differential attack on 27-round SKINNY-64-192, selecting $(x, z) = (9, 3.17)$ for the case of $c = 4$ leads to the data and memory complexities of $2^{63.64}$ and 2^{172} respectively, and time complexity of $2^{174.64}$ memory accesses and $2^{183.00}$ encryptions.

The data complexity $2^{63.64}$ does not exceed 2^{64}, but at least 2 related keys are required for the related key attack. Thus, we use 2 related keys. As a result, the complexity of the exhaustive attack decreases from 2^{192} to 2^{191} in this case. Our attack is valid since its time complexity $2^{183.00}$ is less than 2^{191}.

Additionally, for the impossible differential attack on 27-round SKINNY-128-384, we select $(x, z) = (22.94, 3.99)$ for $c = 8$. The data and memory complexities are $2^{124.99}$ and 2^{344}; the time complexity includes $2^{347.99}$ memory accesses and $2^{361.06}$ encryptions.

Similarly, the complexity of the exhaustive attack is reduced from 2^{384} to 2^{383} in this case. Since the time complexity of our attack $2^{361.06}$ is lower than 2^{383}, our attack is valid.

A.4 ID Attacks on 28-round SKINNY-n-3n in the Related-Tweakey Setting

We guess all the involved subtweakey cells except $STK_{27}[1, 2, 7]$. Similar to the method described in Section 4.5.2, in the guessed subtweakey cells, the subtweakey cells labeled as $\{8, a, e\}$ appear more than or equal to 3 times, thus $STK_{27}[1, 2, 7]$ can be deduced. As a result, the actual parameters of our impossible differential attacks on 28-round SKINNY-n-3n are as follows:

$$c_b = 16c, c_f = 16c, r'_b = 17c (r_b = 16c), r_f = 16c, |k_b \cup k'_f| = 47c, |k_b| = 29c, |k'_f| = 29c$$

For the impossible differential attack on 28-round SKINNY-64-192, we choose $(x, z) = (3.01, 1.06)$ for $c = 4$. Thus, the data and memory complexities are $2^{65.03}$ and 2^{188}, respectively; the time complexity is $2^{189.06}$ memory accesses and $2^{188.99}$ encryptions.

2^2 related keys are required to generate more data as the data complexity $2^{65.03}$ exceeds 2^{64}. Thus, the complexity of the exhaustive attack is reduced from 2^{192} to 2^{190} in this case. Our attack is valid since the time complexity of our attack $2^{188.99}$ is lower than 2^{190}.

Similarly, for the ID attack on 28-round SKINNY-128-384, we select $(x, z) = (5.78, 2)$ for the case of $c = 8$. The data and memory complexities are $2^{129.50}$ and 2^{376} respectively; time complexity includes 2^{378} memory accesses and $2^{378.22}$ encryptions.

Similar to the ID attack on 28-round SKINNY-64-192, the complexity of the exhaustive attack is reduced from 2^{384} to 2^{382} in this case. Our attack is valid, as its time complexity $2^{378.22}$ is lower than 2^{382}.

Fig. 7: ID attack on 21-round SKINNY-n-$3n$. $c_b = 15c, c_f = 15c, r_b = 16c, r_f = 16c, |k_b \cup k_f| = 42c, |k_b| = 26c, |k_f| = 22c$

Fig. 8: ID attack on 23-round SKINNY-n-3n. $c_b = 15c, c_f = 15c, r_b = 16c, r_f = 16c, |k_b \cup k_f| = 47c, |k_b| = 34c, |k_f| = 30c$

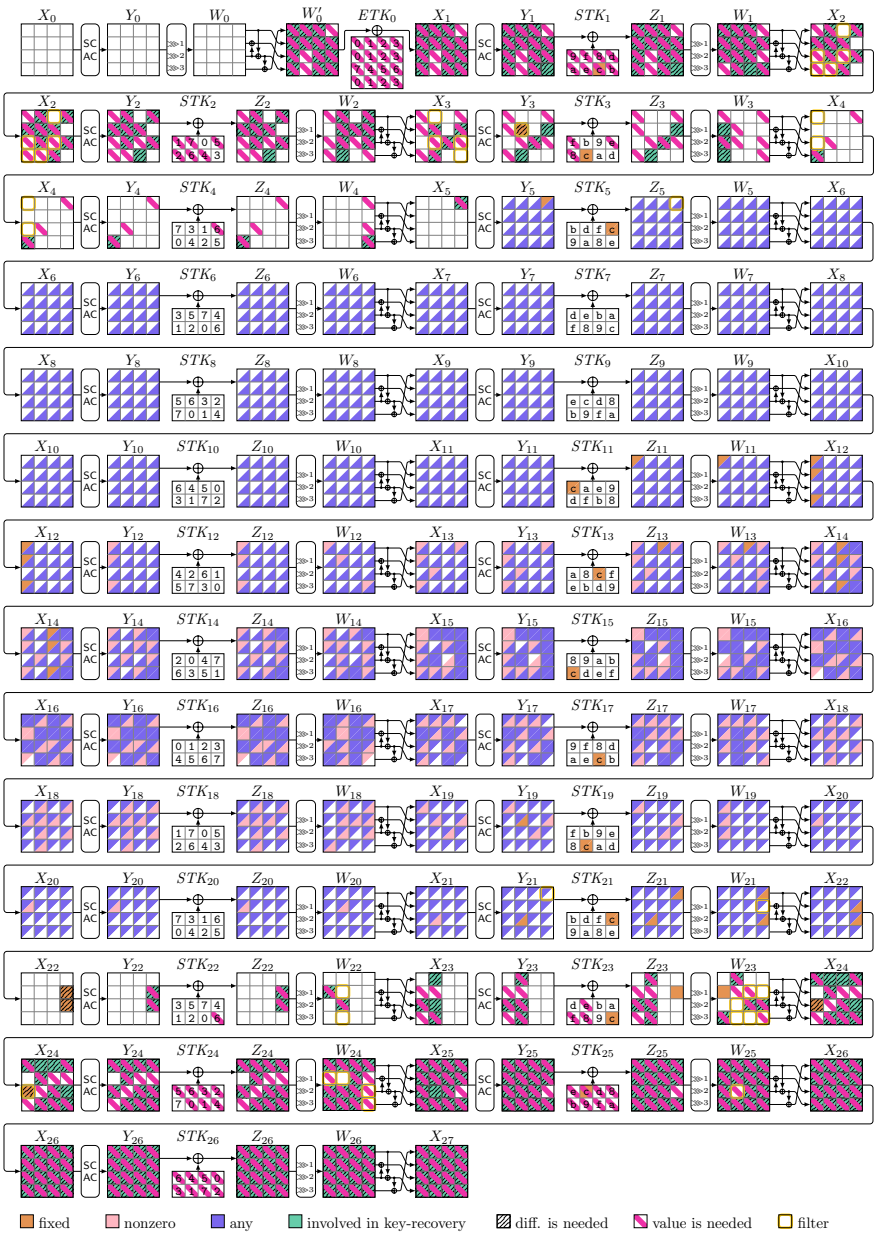

Fig. 9: ID attack on 27-round SKINNY-n-3n. $c_b = 13c, c_f = 16c, r'_b = 14c (r_b = 13c), r_f = 16c, |k_b \cup k_f| = 43c, |k_b| = 26c, |k_f| = 27c$

Fig. 10: ID attack on 28-round SKINNY-n-3n. $c_b = 16c, c_f = 16c, r'_b = 17c (r_b = 16c), r_f = 16c, |k_b \cup k_f| = 47c, |k_b| = 29c, |k_f| = 32c$

B ID Attacks on SKINNYe-v2

B.1 ID Attack on 23-round SKINNYe-v2 in the Single-Tweakey Setting

For the impossible differential attack on 23-round SKINNYe-v2, we select $(x, z) = (13, 3.17)$ to optimize the complexities. Thus, the data and memory complexities of the impossible differential attack on 23-round SKINNYe-v2 are $2^{68.17}$ and 2^{240} respectively; the time complexity consists of $2^{243.17}$ memory accesses and 2^{243} encryptions. Notable, the data complexity $2^{68.17}$ exceeds the codebook 2^{64}, so at least 5 bits of key are required for data expansion. Since the time complexity of the attack is 2^{243}, which is smaller than the remaining key space 2^{251}, the impossible differential attack on 23-round SKINNYe-v2 is valid.

Meanwhile, if we choose $(x, z) = (0.71, -1.03)$ for the attack on 23-round SKINNYe-v2, the data and memory complexities are $2^{63.99}$ and 2^{240}, respectively, with a time complexity of $2^{238.97}$ memory accesses and $2^{255.29}$ encryptions.

B.2 ID Attack on 24-round SKINNYe-v2 in the Single-Tweakey Setting

For the impossible differential attack on 24-round SKINNYe-v2, choosing $(x, z) = (0.71, -1.03)$ leads to the data and memory complexities of $2^{63.99}$ and 2^{252}, respectively, with a time complexity of $2^{250.97}$ memory accesses and $2^{255.29}$ encryptions.

B.3 ID Attack on 25-round SKINNYe-v2 in the Single-Tweakey Setting

For the impossible differential attack on 25-round SKINNYe-v2, selecting $(x, z) = (0.71, -1.03)$ results in the data and memory complexities of $2^{63.99}$ and 2^{252}, respectively, and the time complexity of $2^{251.06}$ memory accesses and $2^{255.29}$ encryptions.

According to Equation 1, when (x, z) is set to $(0.71, -1.03)$, the data complexities of the ID attacks on the 23-round, 24-round, and 25-round SKINNYe-v2 are $D = 2^{63.99}$, which is lower than 2^{64}. The dominant term in the time complexities of all three attacks is $2^{|k|-x}$, and the effect of different k_b and k_f on the time complexity is negligible. Thus, the time complexities of all three attacks are identical. Selecting different x will result in different time complexities.

B.4 ID Attack on 31-round SKINNYe-v2 in the Related-Tweakey Setting

For the impossible differential attack on 31-round SKINNYe-v2, choosing $(x, z) = (5.78, 2)$ leads to the data and memory complexities of 2^{63} and 2^{236} respectively, and the time complexity of 2^{238} memory accesses and $2^{250.22}$ encryptions.

At least 2 related keys are required for the related key attack although the data complexity does not exceed 2^{64}, thus we use 2 related keys. As a result, the complexity of the exhaustive attack decreases from 2^{256} to 2^{255}. Our attack is valid since the time complexity of our attack $2^{250.22}$ is lower than 2^{255}.

B.5 ID Attack on 32-round SKINNYe-v2 in the Related-Tweakey Setting

For the impossible differential attack on 32-round SKINNYe-v2, the data and memory complexities are $2^{65.03}$ and 2^{252} respectively, and the time complexity is $2^{253.06}$ memory accesses and $2^{252.99}$ encryptions when (x, z) is set as $(3.01, 1.06)$.

We need to use 2^2 related keys to generate more data as the data complexity $2^{65.03}$ exceeds 2^{64}. Thus, in this case, the complexity of the exhaustive attack is reduced from 2^{256} to 2^{254}. Our attack is valid since the time complexity of our attack $2^{252.99}$ is lower than 2^{254}.

B.6 ID Attack on 33-round SKINNYe-v2 in the Related-Tweakey Setting

For the impossible differential attack on 33-round SKINNYe-v2, the time complexity is larger than that of the exhaustive search according to the parameters. However, our impossible differential attack on 33-round SKINNYe-v2 can still work by employing the method described in Section 4.5.2. Specifically, all the involved subtweakey cells are guessed except $STK_{32}[0, 1, 2, 3, 4, 5, 6, 7]$. $STK_{32}[0, 1, 2, 3, 4, 5, 6, 7]$ can be deduced by any four groups of the guessed subtweakey cells labeled as $\{0, 1, 2, 3, 4, 5, 6, 7\}$. Thus, the actual parameters of our impossible differential attack on 33-round SKINNYe-v2 are as follows:

$$c_b = 16c, c_f = 16c, r_b = 16c, r_f = 16c, |k_b \cup k'_f| = 63c, |k_b| = 34c, |k'_f| = 40c.$$

For the impossible differential attack on 33-round SKINNYe-v2, selecting $(x, z) = (3.01, 1.06)$ leads to the data and memory complexities of $2^{65.03}$ and 2^{252} respectively, and the time complexity of $2^{253.06}$ memory accesses and $2^{252.99}$ encryptions.

Similarly, 2^2 related keys are required to generate more data as the data complexity $2^{65.03}$ exceeds 2^{64}. Thus, the complexity of the exhaustive attack decreases from 2^{256} to 2^{254} in this case. Our attack is valid, as its time complexity $2^{252.99}$ is lower than 2^{254}.

Fig. 11: ID attack on 23-round SKINNYe-v2. $c_b = 16c, c_f = 16c, r_b = 16c, r_f = 16c, |k_b \cup k_f| = 60c, |k_b| = 34c, |k_f| = 30c$

Fig. 12: ID attack on 24-round SKINNYe-v2. $c_b = 16c, c_f = 16c, r_b = 16c, r_f = 16c, |k_b \cup k_f| = 63c, |k_b| = 34c, |k_f| = 38c$

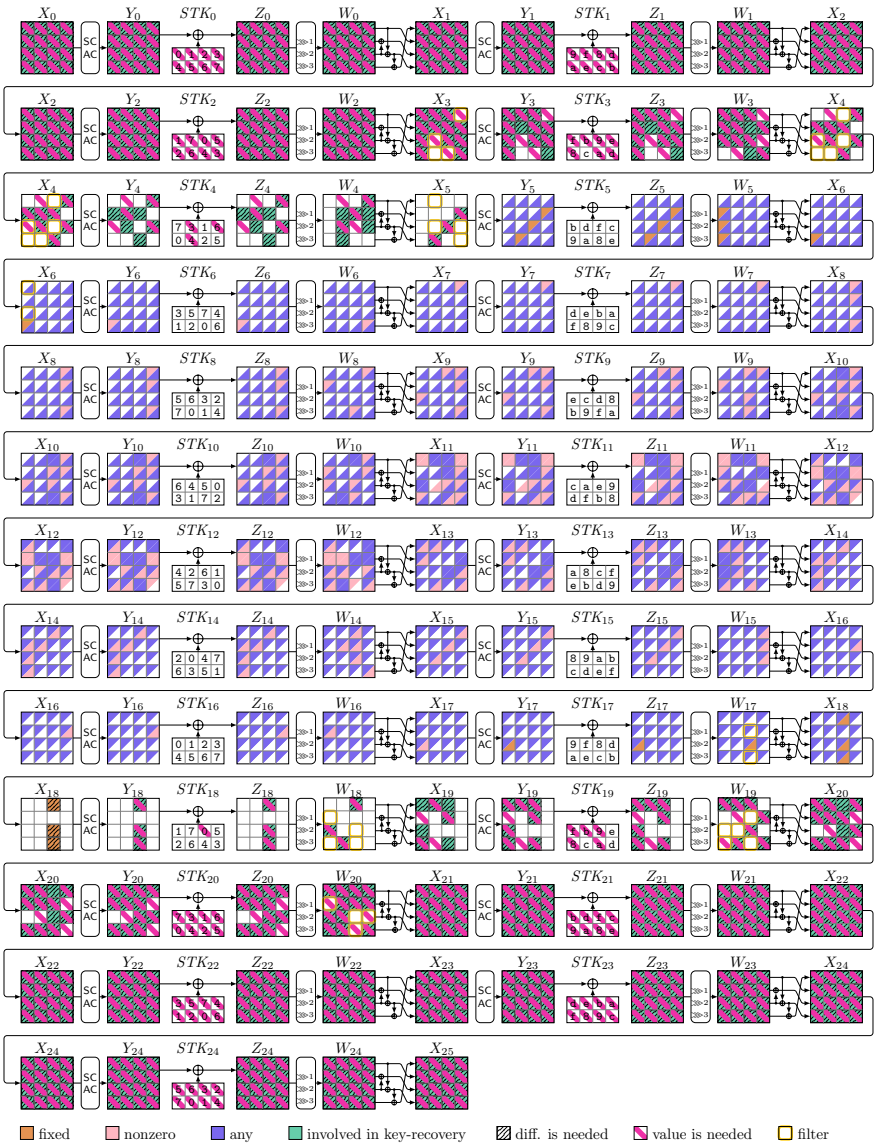

Fig. 13: ID attack on 25-round SKINNYe-v2. $c_b = 16c, c_f = 16c, r_b = 16c, r_f = 16c, |k_b \cup k_f| = 63c, |k_b| = 34c, |k_f| = 46c$

Fig. 14: ID attack on 31-round SKINNYe-v2. $c_b = 13c, c_f = 16c, r_b = 13c, r_f = 16c, |k_b \cup k_f| = 59c, |k_b| = 26c, |k_f| = 41c$

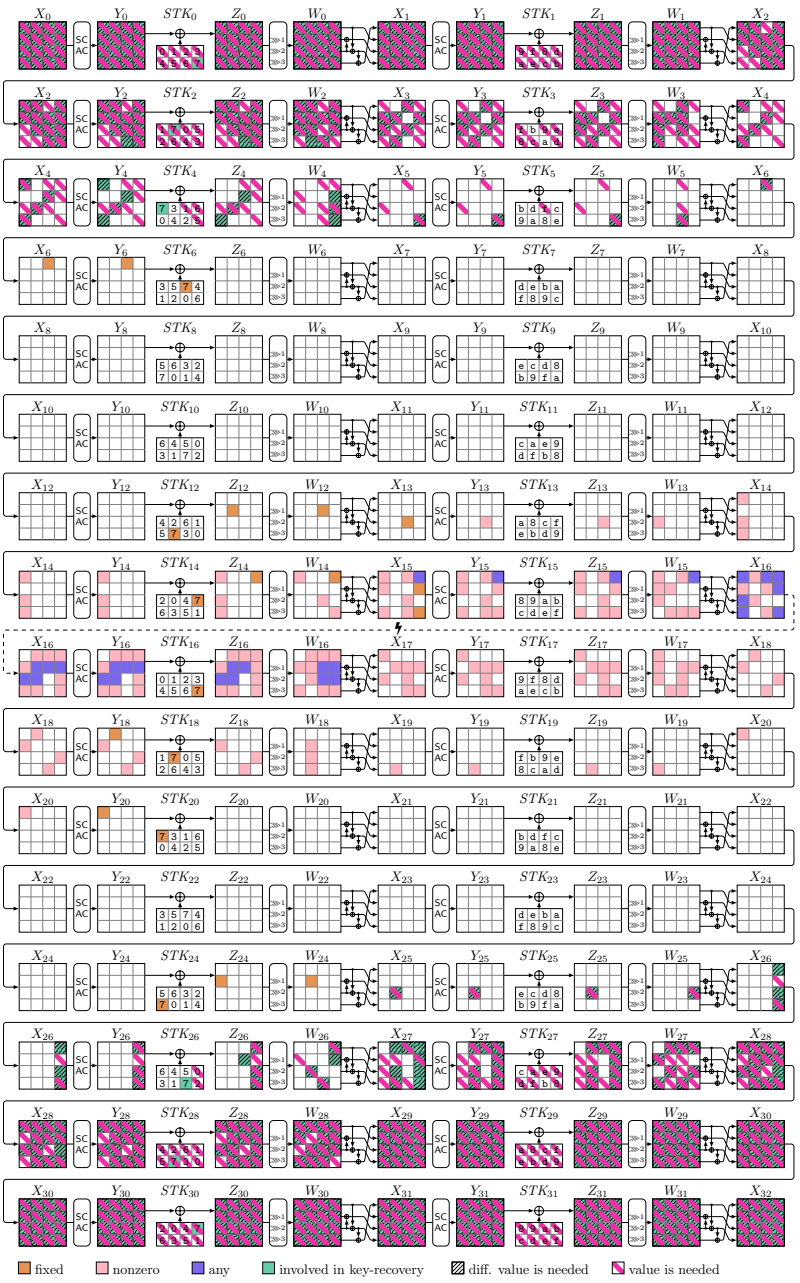

Fig. 15: ID attack on 32-round SKINNYe-v2. $c_b = 16c, c_f = 16c, r_b = 16c, r_f = 16c, |k_b \cup k_f| = 63c, |k_b| = 34c, |k_f| = 40c$

Fig. 16: ID attack on 33-round SKINNYe-v2. $c_b = 16c, c_f = 16c, r_b = 16c, r_f = 16c, |k_b \cup k_f| = 63c, |k_b| = 34c, |k_f| = 48c$

C Other ID Attacks on ForkSKINNY-n-3n

For the impossible differential attack on ForkSKINNY-n-3n, we allow the key size as large as the tweakey size.

C.1 ID Attacks on 28-round ForkSKINNY-n-3n in the Related-Tweakey Setting

We select $(x, z) = (23.10, 4)$ to optimize the complexities for the impossible differential attack on 28-round ForkSKINNY-64-192. Thus, the data and memory complexities are 2^{61} and 2^{156} respectively; the time complexity is 2^{160} memory accesses and $2^{168.90}$ encryptions.

The data complexity 2^{61} does not exceed 2^{64}, but at least 2 related keys are required for the related key attack. The complexity of the exhaustive attack is reduced from 2^{192} to 2^{191} since we use 2 related keys. Our attack is valid, as its time complexity $2^{168.90}$ is lower than 2^{191}.

Additionally, for the impossible differential attack on 28-round ForkSKINNY-128-384, selecting $(x, z) = (67.18, 5.54)$ leads to the data and memory complexities of $2^{118.54}$ and 2^{312} respectively, and the time complexity of $2^{317.54}$ memory accesses and $2^{316.82}$ encryptions.

Similarly, we use 2 related keys for the ID attack on 28-round ForkSKINNY-128-384. Thus, the complexity of the exhaustive attack decreases from 2^{384} to 2^{383} in this case. Our attack is valid since its time complexity $2^{316.82}$ is lower than 2^{383}.

C.2 ID Attacks on 32-round ForkSKINNY-n-3n in the Related-Tweakey setting

For the impossible differential attack on 32-round ForkSKINNY-64-192, choosing $(x, z) = (5.78, 2)$ leads to the data and memory complexities of 2^{63} and 2^{176} respectively, and the time complexity of 2^{178} memory accesses and $2^{186.22}$ encryptions.

At least 2 related keys are required for the related key attack although the data complexity 2^{63} does not exceed 2^{64}. Thus, the complexity of the exhaustive attack decreases from 2^{192} to 2^{191} since we use 2 related keys. Our attack is valid, as its time complexity $2^{186.22}$ is lower than 2^{191}.

Additionally, for the impossible differential attack on 32-round ForkSKINNY-128-384, the data and memory complexities are $2^{125.27}$ and 2^{352} respectively, and the time complexity is $2^{356.27}$ memory accesses and $2^{356.14}$ encryptions when (x, z) is set as $(27.86, 4.27)$.

Similarly, the complexity of the exhaustive attack decreases from 2^{384} to 2^{383} as we use 2 related keys for the ID attack on 32-round ForkSKINNY-128-384. Our attack is valid, as its time complexity $2^{356.14}$ is lower than 2^{383}.

C.3 ID Attacks on 33-round ForkSKINNY-n-3n in the Related-Tweakey Setting

Specifically, all the involved subtweakey cells are guessed except $STK_0[0, 1, 2, 3, 6]$. Similar to the method described in Section 4.5.2, $STK_0[0, 1, 2, 3, 6]$ can be deduced by

any three groups of the guessed subtweakey cells labeled as $\{0, 1, 2, 3, 6\}$. As a result, the actual parameters of our impossible differential attack on 33-round `ForkSKINNY`-n-$3n$ are as follows:

$$c_b = 16c, c_f = 16c, r'_b = 17c(r_b = 16c), r_f = 16c, |k'_b \cup k_f| = 47c, |k'_b| = 29c, |k_f| = 29c$$

For the impossible differential attack on 33-round `ForkSKINNY`-64-192, we select $(x, z) = (3.01, 1.06)$ to optimize the complexities. Thus, the data and memory complexities are $2^{65.03}$ and 2^{188} respectively, and the time complexity is $2^{189.06}$ memory accesses and $2^{188.99}$ encryptions.

We need to use 2^2 related keys to generate more data as the data complexity $2^{65.03}$ exceeds 2^{64}. Thus, the complexity of the exhaustive attack is reduced from 2^{192} to 2^{190} in this case. Our attack is valid since its time complexity $2^{188.99}$ is lower than 2^{190}.

Similarly, for the impossible differential attack on 33-round `ForkSKINNY`-128-384, the data and memory complexities are $2^{129.50}$ and 2^{376} respectively, and the time complexity is 2^{378} memory accesses and $2^{378.22}$ encryptions when (x, z) is set as $(5.78, 2)$.

Similarly, the complexity of the exhaustive attack is reduced from 2^{384} to 2^{382} in this case. Our attack is valid, as its time complexity $2^{378.22}$ is lower than 2^{382}.

Fig. 17: ID attacks on 28-round `ForkSKINNY`-n-$3n$ ($r_i = 11, r_0 = 17, r_1 = 17$). $c_b = 12c, c_f = 15c, r'_b = 14c (r_b = 12c), r_f = 15c, |k_b \cup k_f| = 39c, |k_b| = 22c, |k_f| = 20c$

Fig. 18: ID attacks on 32-round ForkSKINNY-n-3n ($r_i = 11, r_0 = 15, r_1 = 21$). $c_b = 14c, c_f = 16c, r'_b = 15c (r_b = 14c), r_f = 16c, |k_b \cup k_f| = 44c, |k_b| = 27c, |k_f| = 25c$

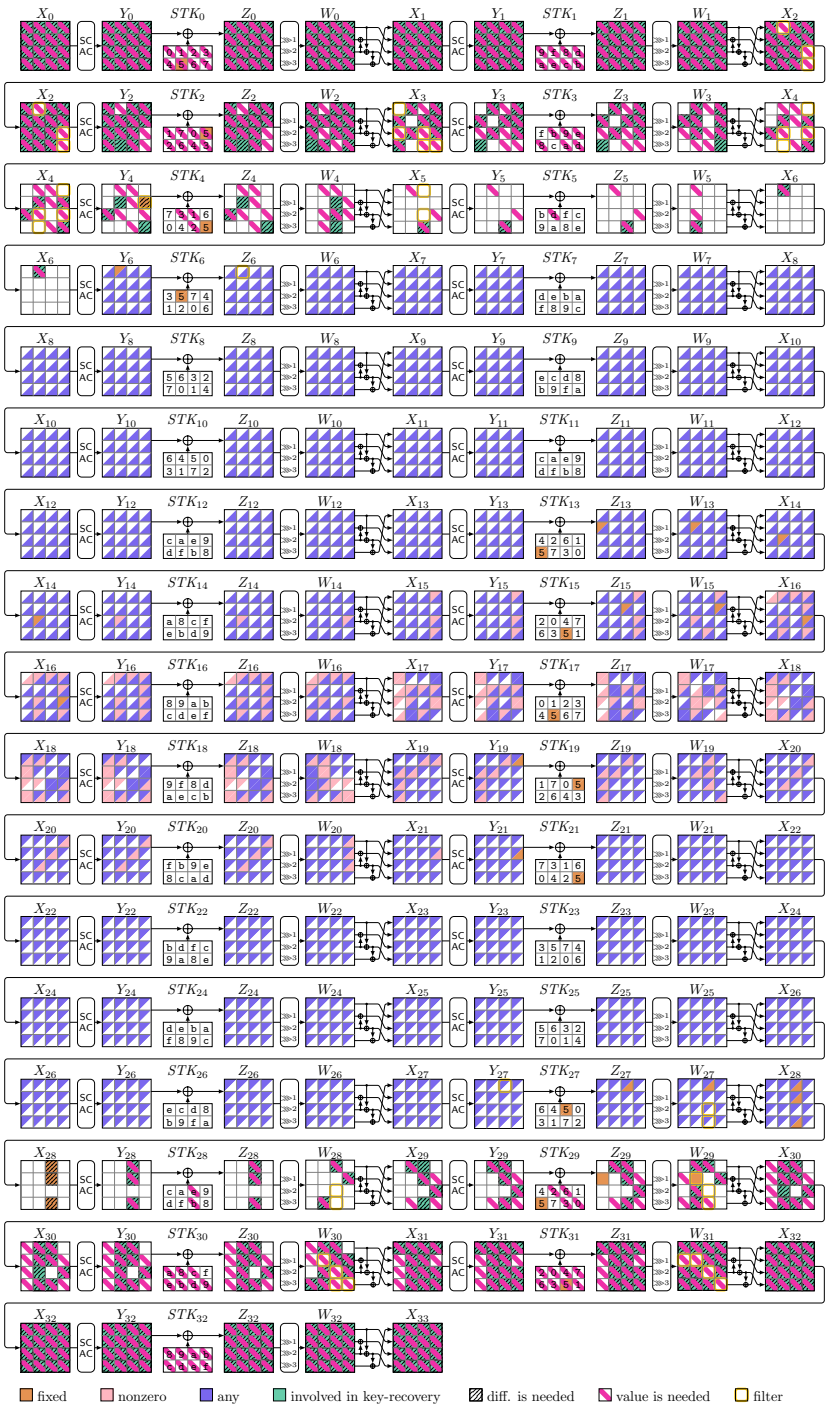

Fig. 19: ID attacks on 33-round ForkSKINNY-n-3n ($r_i = 12, r_0 = 15, r_1 = 21$). $c_b = 16c, c_f = 16c, r'_b = 17c (r_b = 16c), r_f = 16c, |k_b \cup k_f| = 47c, |k_b| = 34c, |k_f| = 29c$