Pre-sieve, Partial-guess, and Accurate estimation: Full-round Related-key Impossible Boomerang Attack on ARADI

Xichao Hu[1], Lin Jiao[1]

State Key Laboratory of Cryptology, Beijing, China xchao_h@163.com, jiaolin jl@126.com

Abstract. The impossible boomerang attack is a very powerful attack, and the existing results show that it is more effective than the impossible differential attack in the related-key scenario. However, in the current key recovery process, the details of a block cipher are ignored, only fixed keys are pre-guessed, and the time complexity of the early abort technique is roughly estimated. These limitations are obstacles to the broader application of impossible boomerang attack. In this paper, we propose the pre-sieving technique, partial pre-guess key technique and precise complexity evaluation technique. For the pre-sieving technique, we capitalize on the specific features of both the linear layer and the nonlinear layer to expeditiously filter out the impossible quartets at the earliest possible stage. Regarding the partial pre-guess key technique, we are able to selectively determine the keys that require guessing according to our requirements. Moreover, the precise complexity evaluation technique empowers us to explicitly compute the complexity associated with each step of the attack.

We integrate these techniques and utilize them to launch an attack on ARADI, which is a low-latency block cipher proposed by the NSA (National Security Agency) in 2024 for the purpose of memory encryption. Eventually, we achieve the first full-round attack with a data complexity of 2^{130}, a time complexity of $2^{254.81}$, and a memory complexity of $2^{252.14}$. None of the previous key recovery methods have been able to attain such an outcome, thereby demonstrating the high efficacy of our new technique.

Keywords: ARADI· Impossible boomerang attack · Pre-sieving technique · Partial pre-guess key technique · Precise complexity evaluation technique

1 Introduction

The impossible boomerang attack (IBA) is a universal key recovery cryptanalysis method for block ciphers, which was first introduced and extended to related-key scenarios by Lu in [Lu,Lu11]. It has effectively targeted 6-round AES-128, 7-round AES-192/AES-256 [DR02] in single-key settings, and 8-round AES-192, 9-round AES-256 in related-key settings.

The basic idea of an impossible boomerang distinguisher (IBD), the core of IBAs, can be best elucidated through a boomerang distinguisher with a probability of 0. Specifically, for a block cipher E_d, given two input differences α, α' and two output differences β, β', if no pair of plaintexts (x_1, x_2) can satisfy the following conditions:

$$E_d(x_1) \oplus E_d(x_2) = \beta, E_d(x_1 \oplus \alpha) \oplus E_d(x_2 \oplus \alpha') = \beta',$$

then $(\alpha, \alpha') \nrightarrow (\beta, \beta')$ forms an IBD of E_d. For the construction of IBDs, the initial method was proposed by Lu [Lu], which decomposes a block cipher E_d into two sub-ciphers E_0 and E_1 ($E_d = E_1 \circ E_0$). Specifically, $(\alpha, \alpha') \nrightarrow (\beta, \beta')$ holds if for $\forall \gamma, \gamma', \delta, \delta'$ such that $\alpha \xrightarrow{E_0} \gamma$, $\alpha' \xrightarrow{E_0} \gamma'$, $\beta \xrightarrow{E_1^{-1}} \delta$ and $\beta' \xrightarrow{E_1^{-1}} \delta'$, it follows that $\gamma \oplus \gamma' \oplus \delta \oplus \delta' \neq 0$. However, this method overlooks the dependence between the two sub-ciphers as highlighted by Murphy [Mur11], which could hinder the discovery of longer IBDs. With the advancement of boomerang attacks, Dunkelman et al. [DKS10,DKS14], introduced the sandwich framework, dividing the block cipher E_d into three parts: $E_1 \circ E_m \circ E_0$, as illustrated in Fig. 1. To evaluate the probability of the boomerang distinguisher on E_m, new tables such as the Boomerang Connectivity Table (BCT) [CHP+18], Double Boomerang Connectivity Table (DBCT) [WP19,DDV20] and others [BHL+20] were proposed for S-box based block ciphers. Building on the concepts of BCT and DBCT, two papers [BCL+24,ZWT24]

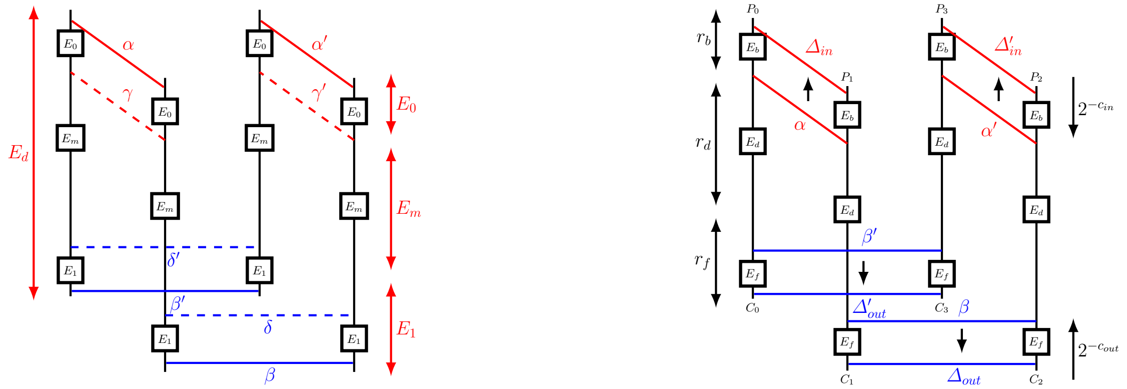

Fig. 1. The IBD and its extended IBA.

proposed new methods for constructing IBDs, including SAT/SMT-based approaches [BL23,WWS23] and CP-based approaches [HSE23] to search for IBDs.

To launch the IBA given an IBD, an attacker can extend r_b rounds before the IBD and r_f rounds after the IBD, as shown in Fig. 1. Two primary key recovery methods have been identified: the impossible differential style (IDS) and the boomerang style (BS) [Lu11,BCL$^+$24,ZWT24]. In IDS, the attacker constructs a set of quartets that satisfy the two input differences and two output differences of the IBD. Subsequently, the early abort technique [LKKD08a] is employed to eliminate incorrect key guesses. In BS, the attacker first guesses all necessary keys in the first r_b rounds (resp. the last r_f rounds) to build the quartets that satisfy the two input differences and two output differences of the IBD. Then, the early abort technique [LKKD08a] is employed to discard incorrect keys in the last r_f rounds (resp. the first r_b rounds). Additionally, the IBAs were initially launched manually for AES [Lu11], while recent studies have utilized automatic methods [BCL$^+$24,ZWT24], leading to new results for block ciphers such as SKINNY [BJK$^+$16] and SKINNYee [NSS22].

Compared with the differential attack of block ciphers, the research and application of IBA remain relatively insufficient, especially in the key recovery process, where several limitations persist.

- **Ignore the details of a block cipher.** In current key recovery methods, if a difference α can affect l bits through the inverse of a function F, then all $2^l - 1$ differences that are active on at least one of those bits are considered to be able to propagate to this difference through F. This undoubtedly increases the number of impossible quartets, leading to an increase in the complexity of the attack.
- **Only pre-guess fixed keys.** In the impossible differential style, the attacker does not pre-guess the keys, and in boomerang style, the attacker pre-guesses the keys in the first r_b rounds or the last r_f rounds. These two styles may have high complexity in different steps, which may lead to the overall attack being unavailable.
- **Roughly estimate the time complexity of the early abort technique.** Currently, all key recovery methods will use an approximate formula to estimate the complexity. There are two problems here. One is that if the time complexity of the early abort technique dominates the overall complexity, the result obtained by using the approximation formula is not necessarily the optimal one. The other is that even if we use the approximation formula to obtain the optimal solution, we still have to manually derive the specific key recovery process so as to give the detailed attack steps, which is undoubtedly complicated and boring.

Our contributions. In this paper, we focus on automated key recovery of IBA. In order to break through the limitations of current automated algorithms, we propose three new techniques as follows.

- **Pre-sieving technique.** In this technique, we utilize the details of the linear layer and the nonlinear layer to obtain the set of possible differences as accurately as possible, and then filter out the impossible quartets as early as possible.

- **Partial pre-guess key technique.** In this technique, we define the forward dependency graph of known variables and backward dependency graph of known variables. Using these two graph, we can divide the key blocks that can be guessed separately, thus achieving partial guessed keys. That is, we can choose the key blocks in the first r_b rounds of the distinguisher and the last r_f rounds of the distinguisher for pre-guessing according to one's own needs.
- **Precise complexity evaluation technique.** In this technique, we define the forward dependency graph of key recovery and backward dependency graph of key recovery. According to these two diagrams, the early abort technique can be clearly described step by step, which allows us to clearly calculate the complexity of each step of the attack. By exhaustively searching for all given distinguishers, we can automatically provide the optimal key recovery strategy as well as the detailed process of the attack.

Finally, we integrate the above techniques together and present a new automated key recovery technique for IBA.

As a result, we implement the IBA on the block cipher ARADI.

- **Distinguishers.** By carefully studying the key schedule of ARADI, we found 3-round related-key differentials with a probability of 1. Utilizing two such differentials and according to the BCT, we found 11-round IBDs.
- **Key recovery.** We add 2 rounds before and 3 rounds after the 11-round distinguisher. Then, we use our new key recovery method to launch the full-round attack. Finally, we get an optimal attack with the data complexity is 2^{128}, the time complexity is $2^{254.81}$ and the memory complexity is $2^{252.14}$, which means we break the block cipher totally. This implies that the block cipher is completely broken. To the best of our knowledge, this constitutes the first full-round attack. Moreover, even by leveraging the impossible differential style and the boomerang style IBA, a full-round break remains unattainable. This indicates that our new method is highly effective.

Outline. We introduce the notations and related work in Section 2. The new key recovery techniques are presented in Section 3. In Section 4, we detail the full-round attack for the block cipher ARADI. In Section 5, we conclude this paper.

2 Preliminaries

Our key recovery method is applicable to S-box based block ciphers. To provide a clearer description, we use the SPN block cipher as an illustrative example and present the following notations accordingly.

2.1 Notations

Let E denote an n-bit SPN block cipher and has a key size of m bits. One encryption round of E is illustrated in Fig. 2, which consists of three fundamental operations:

- `SL`: The S-box layer, wherein t parallel q-bit S-boxes are employed, introducing non-linearity to the cipher
- `LL`: The linear layer, which adopts a global linear transformation, further enhancing the diffusion.
- $\texttt{AKL}_{k^r}$: The key addition layer, where the round key k^r in round r is XORed with the internal state.

The following notations are used hereafter.

- $\mathbb{Z}_n$: The set $\{0, 1, \ldots, n-1\}$.
- $\alpha \xrightarrow{F} \beta$: The input difference α can propagate to the output difference β through the function F.
- $K_i, i = 0, 1, 2, 3$: The keys of E in the related-key setting.
- $T_i, i = 0, 1, 2, 3$: The plaintext-ciphertext sets encrypted by K_i.
- $IX_i^r, i = 0, 1, 2, 3$: The internal state of E under K_i before the key addition layer in round r.
- $IY_i^r, i = 0, 1, 2, 3$: The internal state of E under K_i before the S-box layer in round r.
- $IZ_i^r, i = 0, 1, 2, 3$: The internal state of E under K_i before the linear map in round r.

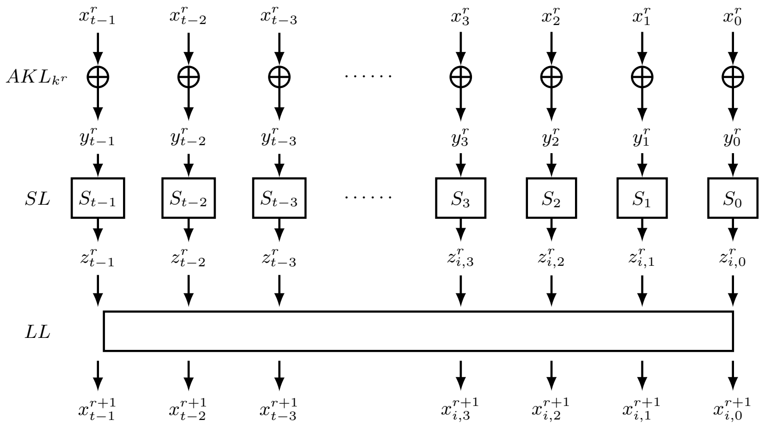

Fig. 2. One round of SPN structure block cipher.

- $IK^r_i, i = 0, 1, 2, 3$: The round key in round r under K_i.
- $\varDelta X^r_{01}, \varDelta X^r_{23}$: Differences in the upper trail of the IBD, i.e. $\varDelta X^r_{01} = IX^r_0 \oplus IX^r_1$ and $\varDelta X^r_{23} = IX^r_2 \oplus IX^r_3$. Analogous notations apply for IY, IZ, and IK.
- $\nabla X^r_{12}, \nabla X^r_{03}$: Differences in the lower trail of the IBD, i.e. $\nabla X^r_{12} = X^r_1 \oplus X^r_2$ and $\nabla X^r_{03} = X^r_0 \oplus X^r_3$. Analogous notations apply for IY, IZ, and IK.
- $\mathcal{N}^r_j(\beta)$: The number of input differences that can propagate to the output difference β for the j-th S-box in round r.
- $\overline{\mathcal{N}}^r_j(\alpha)$: The number of output differences that can propagate to the input difference α for the j-th S-box in round r.

The notations of an IBA according to Fig. 1 are details as follows:

- α, α' (resp. β, β'): The input (resp. output) differences of the IBD.
- $\varOmega_{in}$,d_{in} (resp. $\varOmega_{out}, d_{out}$): $\varOmega_{in}$ (resp. $\varOmega_{out}$) denotes the set of plaintext (resp. ciphertext) differences that may lead to the input (resp. output) difference α (resp. β) of the IBD, where $d_{in} = log_2|\varOmega_{in}|$ (resp. $d_{out} = log_2|\varOmega_{out}|$).
- p_{in} (resp. p_{out}): p_{in} (resp. p_{out}) denotes the probability of reaching the input (resp. output) difference α (resp. β) of the IBD from the plaintext (resp. ciphertext) difference in $\varOmega_{in}$ (resp. $\varOmega_{out}$).
- K_{in} (resp. K_{out}): The key bits involved in the IBA in E_b (resp. E_f).
- N^r_a, J^r: N^r_a denotes the number of active S-boxes in round r, and $J^r = \{j^r_0, \ldots, j^r_{N^r_a-1}\}$ denotes the indices of the active S-boxes in round r.
- $\varOmega^r_{in}$, p^r_{in}: $\varOmega^r_{in}$ denotes the set of input difference that may lead to the input difference α of the IBD in round r when considering the details of S-boxes, and p^r_{in} represents the probability of reaching α from the difference within $\varOmega^r_{in}$, for $0 \le r \le r_b - 1$.

2.2 The definitions about IBDs

The original definition of IBD is defined as follows.

Definition 1 (IBD). *Given a block cipher $E : \mathbb{F}^n_2 \times \mathbb{F}^m_2 \to \mathbb{F}^n_2$ under four keys $K_i \in \mathbb{F}^m_2, i = 0, 1, 2, 3$, if for four state differences $\alpha, \alpha', \beta, \beta'$ and three key differences $\kappa_0, \kappa_1, \kappa_2$, any pair of plaintexts (x_1, x_2) cannot satisfy*

$$E_{K_1}(x_1) \oplus E_{K_2}(x_2) = \beta,\ E_{K_0}(x_1 \oplus \alpha) \oplus E_{K_3}(x_2 \oplus \alpha') = \beta' \qquad (1)$$

then $(\alpha, \alpha', \beta, \beta')$ is called an realted-key IBD (RK-IBD) of E under the key differences $\kappa_0, \kappa_1, \kappa_2$, where $(K_0, K_1, K_2, K_3) = (K, K \oplus \kappa_0, K \oplus \kappa_1, K \oplus \kappa_2)$. Particularly, $(\alpha, \alpha', \beta, \beta')$ is called an IBD of E under K if $K = K_0 = K_1 = K_2 = K_3$ in Eq. (1).

Currently, existing techniques [BCL+24,ZWT24] for constructing an IBD involve the utilization of various tables, such as BCT [CHP+18] and DBCT [WP19,DDV20]. They are defined as follows and outlined in Fig. 3.

Definition 2 (BCT). *Let S be a permutation of $\mathbb{F}_2^n$, and $\Delta_i, \nabla_o \in \mathbb{F}_2^n$. The BCT of S is a two-dimensional table defined by:*

$$BCT(\Delta_i, \nabla_o) = \{x \in \mathbb{F}_2^n \mid S^{-1}(S(x) \oplus \nabla_o) \oplus S^{-1}(S(x \oplus \Delta_i) \oplus \nabla_o) = \Delta_i\}.$$

Definition 3 (DBCT). *Let S be a permutation of $\mathbb{F}_2^n$, and $\Delta_i, \Delta_o, \nabla_i, \nabla_o \in \mathbb{F}_2^n$. The DBCT of S is a two-dimensional table defined by:*

$$DBCT(\Delta_i, \nabla_o) = \sum_{\Delta_o, \nabla_i} UBCT(\Delta_i, \Delta_o, \nabla_i) \cdot LBCT(\Delta_o, \nabla_i, \nabla_o),$$

where the UBCT and LBCT of S are three-dimensional tables defined as

$$UBCT(\Delta_i, \Delta_o, \nabla_i) = \# \left\{ x \in \mathbb{F}_2^n \middle| \begin{array}{l} S(x) \oplus S(x \oplus \Delta_i) = \Delta_o \\ S^{-1}(S(x) \oplus \nabla_i) \oplus S^{-1}(S(x \oplus \Delta_i) \oplus \nabla_i) = \Delta_i \end{array} \right\},$$

$$LBCT(\Delta_o, \nabla_i, \nabla_o) = \# \left\{ x \in \mathbb{F}_2^n \middle| \begin{array}{l} S(x) \oplus S(x \oplus \nabla_i) = \nabla_o \\ S^{-1}(S(x) \oplus \nabla_o) \oplus S^{-1}(S(x \oplus \Delta_o) \oplus \nabla_o) = \Delta_o \end{array} \right\}.$$

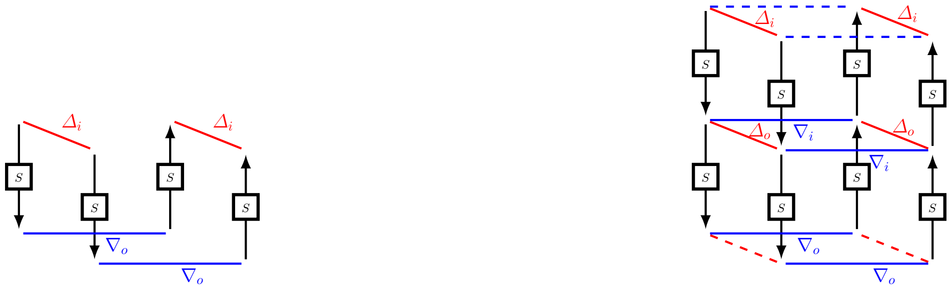

Fig. 3. The illustrations of BCT and DBCT.

As illustrated in Fig. 1, for a block cipher $E_d = E_1 \circ E_m \circ E_0$, if for $\forall \gamma, \gamma', \delta, \delta'$ such that $\alpha \xrightarrow{E_0} \gamma$, $\alpha' \xrightarrow{E_0} \gamma'$, $\beta \xrightarrow{E_1^{-1}} \delta$ and $\beta' \xrightarrow{E_1^{-1}} \delta'$, (γ, γ') cannot propagate to (δ, δ') through E_m according to the BCT or DBCT, then $(\alpha, \alpha') \nrightarrow (\beta, \beta')$.

We now formally describe the automatic search methods of (RK-)IBDs named as **sat model**, introduced in [BCL+24,ZWT24] in a more unified way.

I. Identify the S-boxes with known and non-zero (`KD`) input-output differences.

  i. Set the flags. Categorize the differences of S-boxes into four types: zero difference (`ZD`), known and non-zero difference (`KD`), any non-zero difference (`ND`), and any difference (`AD`). Partition the difference of the internal states into blocks based on the size of the S-box. Set two flags for each block: flag fd to signify the type of the difference, and flag fv to signify the specific difference value if $fd = \texttt{KD}$.

  ii. Build the propagation rule. For the operations in SPN block ciphers, the flags propagate as follows.

  - S-box: Let fd_i and fd_o be the types of input and output differences. Then,

$$fd_o = \begin{cases} \texttt{ZD}, fd_i = \texttt{ZD}, \\ \texttt{ND}, fd_i = \texttt{KD} \text{ or } \texttt{ND}, \\ \texttt{AD}, \text{ otherwise}. \end{cases}$$

- XOR: Let fd_{i_0} and fd_{i_1} be the types of two input differences of XOR, and fd_o be the type of output difference. Then,

$$fd_o = \begin{cases} \texttt{ZD}, (fd_{i_0}, fd_{i_1}) = (\texttt{ZD}, \texttt{ZD}), \\ \texttt{KD}, (fd_{i_0}, fd_{i_1}) = (\texttt{ZD}, \texttt{KD}) \text{ or } (\texttt{KD}, \texttt{ZD}), \\ \texttt{ND}, (fd_{i_0}, fd_{i_1}) = (\texttt{ZD}, \texttt{ND}) \text{ or } (\texttt{ND}, \texttt{ZD}) \\ \texttt{AD}, \text{ otherwise.} \end{cases}$$

For other linear transformations, their propagation rules can be derived based on XOR operations. Additionally, fv adjusts in accordance with the changes of fd.

iii. Detect the positions of S-box. Utilize the MILP method [ZWT24] or CP method [BCL+24] to model the forward propagation of the flag of the input difference over r rounds under the flag of the key difference, and also the backward propagation of the flag of the output difference over r rounds under the flag of the key difference. Find a solution where there exists a target S-box for which both the input and output differences are non-zero and known.

II. Check for contradictions according to the BCT. If a solution is identified, assign specific values to the input difference and output difference (and key difference in the related-key setting). Subsequently, derive the specific values of the input and output differences for the target S-box. If these specific values do not align with the possible input and output values in the BCT, an RK-IBD is confirmed.

Within this model, there is no need to pre-specify the differences in input, output and key. Each solution is associated with a set of flags that may generate an (RK-)IBD. Contradictions can be verified post-model solving. Consequently, this approach remains effective in searching for (RK-)IBDs even when the weights of the input, output, and key differences are high.

2.3 Key recovery process of IBAs

As depicted in Fig. 1, given an r_d-round IBD of E_d, attackers add r_b rounds before and r_f rounds after the IBD to launch an $(r_b + r_d + r_f)$ rounds IBA. Similar as that in [BCL+24,ZWT24], the two input differences and the two output differences of the IBD are set equal hereafter, i.e. $\alpha = \alpha'$ and $\beta = \beta'$. Besides, we focus on the related-key setting with

$$(K_0, K_1, K_2, K_3) = (K_0, K_0 \oplus \mathit{\Delta} K, K_0 \oplus \nabla K \oplus \mathit{\Delta} K, K_0 \oplus \nabla K). \tag{2}$$

The single-key setting can be derived analogously by setting $\mathit{\Delta} K = \nabla K = 0$. Consequently, the sets of plaintext and ciphertext differences leading to α and β are identical, i.e., $\mathit{\Omega}_{in} = \mathit{\Omega}'_{in}$ and $\mathit{\Omega}_{out} = \mathit{\Omega}'_{out}$. Without loss of generality, we assume that the queries are directed to the encryption oracle. Similarly, these queries can also be submitted to the decryption oracle.

Subsequently, we provide an overview of the state-of-the-art automatic key recovery techniques for IBA proposed in [BCL+24,ZWT24], named **Impossible Differential Style** (IDS) and **Boomerang Style** (BS). Before introducing the two key-recovery attack styles, we recall the early abort technique used in both.

Early abort technique [LKKD08b]. Depending on the round function, instead of guessing all of the required round key bits $K_{in} \cup K_{out}$ at once, attackers can partially check if a plaintext or ciphertext pair produces the expected difference of the distinguisher by guessing fractions of them step by step, discarding invalid pairs after each guess. This reduces the attack's computational workload.

Impossible Differential Style

-IDS.1: Get plaintext-ciphertext pairs. Construct 2^s plaintext structures, each containing $2^{d_{in}}$ plaintexts activated at d_{in} fixed bits. Query the ciphertexts corresponding to the $2^{s+d_{in}}$ plaintexts under four related keys as specified in Eq. (2). In total, $\mathcal{D} = 2^{2+s+d_{in}}$ plaintext-ciphertext pairs are required.

-IDS.2: Produce quartets.

- **-IDS.2a:** Construct plaintext pairs within each plaintext structure, and derive $\mathcal{P}$ pairs of $((P_0, C_0), (P_1, C_1))$ under (K_0, K_1) and $\mathcal{P}$ pairs of $((P_3, C_3), (P_2, C_2))$ under (K_2, K_3), where $\mathcal{P} = 2^{s+2d_{in}}$.
- **-IDS.2b:** Construct a hash table H_0 that lists the pairs of $((P_0, C_0), (P_1, C_1))$, indexed by the two $(n - d_{out})$ bits of the ciphertexts not belonging to Ω_{out}. For each $((P_3, C_3), (P_2, C_2))$, lookup the hash table H_0 using the two $(n - d_{out})$ bits of C_3 and C_2 to find the corresponding $((P_0, C_0), (P_1, C_1))$. On average, $\mathcal{Q} = 2^{2(s+2d_{in})-2(n-d_{out})}$ quartets of $((P_0, C_0), (P_1, C_1), (P_2, C_2), (P_3, C_3))$ are derived, where (P_0, P_1) and (P_2, P_3) have differences in Ω_{in}, and (C_0, C_3) and (C_1, C_2) have differences in Ω_{out}.

-IDS.3: Assume that the keys in K_{in} are recovered first, , followed by the keys in K_{out}.

- **-IDS.3a:** Adopt the early abort technique to recovery the K_{in} by the $\mathcal{Q}$ quartets.
- **-IDS.3b:** Adopt the early abort technique to recovery the K_{out} by the remaining quartets.

-IDS.4: Perform an exhaustive search on the remaining keys.

Complexity. The date complexity is $\mathcal{DC}_{IDS} = 2^{2+s+d_{in}}$. For the time complexity $\mathcal{TC}_{IDS}$, it consists of the following five parts, i.e., $\mathcal{TC}_{IDS} = \mathcal{D} + 2\mathcal{P} + \mathcal{Q} + \mathcal{A} + \mathcal{E}$:

- Cost of data generation: $\mathcal{D} = 2^{2+s+d_{in}}$.
- Cost of building pairs: $2\mathcal{P}$, where $\mathcal{P} = 2^{s+2d_{in}}$.
- Cost of producing quartets: $\mathcal{Q} = 2^{2(s+2d_{in})-2(n-d_{out})}$.
- Cost of adopting the early abort technique to recovery keys: The time complexity of this step is estimated as $\mathcal{A} = (\mathcal{Q} \times p_{in}^2 p_{out}^2 \times 2^{|K_{in} \cup K_{out}|})C'_E$, where C'_E represents the ratio of the cost for one partial encryption to the full encryption.
- Cost of final exhaustive search: If such a quartet indeed leads to the input and output differences of the IBD, which occurs with a probability of $p_{in}^2 p_{out}^2$, it is able to discard a key. Thus, the probability of a key being retained is $p = (1 - p_{in}^2 p_{out}^2)^{\mathcal{Q}}$. The time complexity of this step is $\mathcal{E} = p \cdot 2^{|K|} = 2^{|K|}(1 - p_{in}^2 p_{out}^2)^{\mathcal{Q}}$.

The memory complexity is determined by the cost of storing the data, pairs, quartets and remaining keys: $\mathcal{MC}_{IDS} = \mathcal{D} + 2\mathcal{P} + \mathcal{Q} + \mathcal{K}$, where $\mathcal{K} = 2^{|K_{in} \cup K_{out}|}$.

Boomerang Style

-BS.1: This step is identical to Step IDS.1 of the impossible differential style.

-BS.2: Guess all the key candidates K_{in}:

- **-BS.2a:** For each plaintext structure, partially encrypt P_0 to the beginning of the IBD under k_0, XOR the resulting state with α, and then decrypt it to produce the plaintext P_1 under k_1. Get their corresponding ciphertexts (C_0, C_1) by consulting table T_i, $i \in \{0, 1\}$. Consequently, $2^{s+d_{in}}$ pairs $((P_0, C_0), (P_1, C_1))$ are derived. Consequently, $2^{s+d_{in}}$ pairs $((P_3, C_3), (P_2, C_2))$ are constructed.
- **-BS.2b:** This step is identical to Step IDS.2b of the impossible differential style. On average, $\mathcal{Q} = 2^{2(s+d_{in})-2(n-d_{out})}$ quartets of $((P_0, C_0), (P_1, C_1), (P_2, C_2), (P_3, C_3))$ are derived, where (P_0, P_1) and (P_2, P_3) have differences in Ω_{in}, and (C_0, C_3) and (C_1, C_2) have differences in Ω_{out}.
- **-BS.2c:** Adopt the early abort technique to recovery the K_f for the $\mathcal{Q}$ quartets.

-BS.3: Perform an exhaustive search on the remaining keys.

Complexity. The date complexity is $\mathcal{DC}_{BS} = 2^{2+s+d_{in}}$. For the time complexity $\mathcal{TC}_{IDS}$, it consists of the following five parts, i.e., $\mathcal{TC}_{IDS} = \mathcal{D} + \mathcal{P}' + \mathcal{Q}' + \mathcal{A} + \mathcal{E}$:

- Cost of data generation: $\mathcal{D} = 2^{2+s+d_{in}}$.
- Cost of building pairs: $\mathcal{P}' = 2^{|K_{in}|} \times 2\mathcal{P} \times 2|E_b|/|E|$, where $\mathcal{P} = 2^{s+d_{in}}$.
- Cost of producing quartets: $\mathcal{Q}' = 2^{|K_{in}|} \times \mathcal{Q} = 2^{|K_{in}|+2(s+d_{in})-2(n-d_{out})}$.
- Cost of adopting the early abort technique to recovery keys : $\mathcal{A} = 2^{|K_{in}|} \times (\mathcal{Q} \times 2^{|K_{out}/K_{in}|-2c_{out}})C'_E$, where C'_E is the ratio of the cost for one partial encryption to the full encryption.
- Cost of final exhaustive search: If such a quartet indeed leads to the input and output differences of the IBD, which occurs with a probability of p_{out}^2, it is able to discard a key. Thus, the probability of a key being retained is $p = (1 - p_{out}^2)^{\mathcal{Q}}$. The time complexity of this step is $\mathcal{E} = p \cdot 2^{|K|} = 2^{|K|}(1 - p_{out}^2)^{\mathcal{Q}}$.

The memory complexity is determined by the cost of storing the data, pairs, quartets and remaining keys: $\mathcal{MC}_{BS} = \mathcal{D} + 2\mathcal{P} + \mathcal{Q} + \mathcal{K}$, where $\mathcal{K} = 2^{|K_{in} \cup K_{out}|1}$.

[1] We summarize some general details for deriving the common parameters used in both IDS and BS key recovery:

3 New Technologies Facilitating IBAs

In this section, we introduce the pre-sieving technique, the partial pre-guess key technique, and the precise complexity evaluation technique to optimize the key recovery in IBA[2].

3.1 Pre-sieving technique

The core of our pre-sieving technique lies in determining the possible propagation difference set as precisely as possible based on the details of nonlinear layers, thereby enabling the early elimination of impossible quartets. For an SPN block cipher E, given an r_d-round IBD $(\alpha, \alpha, \beta, \beta)$, r_b rounds before and r_f rounds after the IBD are added to launch an $(r_b + r_d + r_f)$ rounds IBA. The pre-sieving technique can be applied to the first r_b rounds when queries are directed towards the encryption oracle as detailed subsequently. It is also applicable to the last r_f rounds for decryption queries.

Let $\varphi^r = (\varphi_0^r, \ldots, \varphi_{t-1}^r)$ and $\eta^r = (\eta_0^r, \ldots, \eta_{t-1}^r)$ denote the input and output differences of the S-box layer in round r, respectively. With truncated differential propagation rules, the indexes J^r and the number N_a^r of active S-boxes in round r within E_b are derived based on the given α and round key differences. For determining Ω_{in}, current key recovery methods consider a nonzero output difference of a q-bit S-box as potentially propagated from any one of the $2^q - 1$ nonzero input differences. Thereby $|\Omega_{in}| = 2^{N_a^0 \cdot q}$. However, these methods fail to account for the details of the S-box, which can further reduce $|\Omega_{in}|$.

Consequently, we propose the pre-sieving technique. For instance, when $r_b = 1$, the optimized plaintext difference set Ω_{in}^0 is derived according to the DDT of S-boxes with $|\Omega_{in}^0| = \prod_{j \in J^0} \mathcal{N}_j^0(\eta_j^0)$, where η^0 is determined by α. Given that $\mathcal{N}_j^0(\eta_j^0) \leq 2^q$, it follows that $|\Omega_{in}^0| \leq |\Omega_{in}|$. This implies that the differences in $\Omega_{in}/\Omega_{in}^0$ cannot propagate to α, and thus can be initially disregarded to prevent the unnecessary addition of impossible quartets. Furthermore, when $r_b > 1$, the set of differences Ω_{in}^r that might lead to α at round r is derived by back-propagating each difference in Ω_{in}^{r+1} through one round of E under the corresponding round key difference, for $0 \leq r \leq r_b - 2$. Similarly, $\Omega_{in}^{r_b - 1}$ is derived as in the case when $r_b = 1$.

To evaluate the feasibility of the attack, it is necessary to estimate the attack complexity, which hinges on determining the value of $|\Omega_{in}^0|$. As described above, we iteratively compute Ω_{in}^r to a computable intermediate set $\Omega_{in}^{r_{bm}}$, where $0 \leq r_{bm} \leq r_b - 1$. When $bm = 0$, $|\Omega_{in}^0|$ is derived; otherwise, we need further estimate Ω_{in}^0, which exceeds the current computing capacity. For S-boxes over $\mathbb{F}_2$, at most 2^{q-1} differences can propagate to a given output difference. Therefore, we amplify $\mathcal{N}_j^r(\eta_j^r)$ to 2^{q-1} for all $j \in J^r$ and $0 \leq r \leq r_{bm} - 1$. Consequently, $|\Omega_{in}^0|$ is estimated as $|\Omega_{in}^{r_{bm}}| \prod_{r=0}^{r_{bm}-1} \left(\prod_{j \in J^r} 2^{q-1} \right)$, which is upper bounded by $2^{N_a^0 \cdot q}$.

Next, we prove that p_{in}^0, the probability of reaching the differences α from the difference within Ω_{in}^0, is $1/|\Omega_{in}^0|$.

Theorem 1. *Let $E_k^{r_b}$ denote the r_b-round encryption function under the round keys $k = (k^0, \ldots, k^{r_b - 1})$, and Ω_{in}^0 be the set of plaintext differences that back-propagate the difference α through r_b rounds under the key difference $\Delta k = (\Delta k^0, \ldots, \Delta k^{r_b - 1})$ using the pre-sieving technique. For a plaintext pair $(x_0, x_1) \in \{(x, x \oplus \mu) | \mu \in \Omega_{in}^0\}$, the probability that $E_k^{r_b}(x_0) \oplus E_{k \oplus \Delta k}^{r_b}(x_1) = \alpha$ is $1/|\Omega_{in}^0|$.*

Proof. We prove this theorem using the recursive method.

- The positions of the d_{in} (resp. d_{out}) activated bits in the plaintext (resp. ciphertext) are determined by the truncated differential back-propagation (resp. propagation) from the input difference α (resp. output difference β) of the IBD based on the round key differences.
- The probabilities p_{in} and p_{out} are usually analyzed specifically based on the filtering conditions. For the fixed input and output difference of IBD, p_{in} and p_{out} are usually equal to $1/|\Omega_{in}|$ and $1/|\Omega_{out}|$, respectively.
- C'_E is usually estimated as the number of nonlinear operations in the partial encryption, divided by the number of nonlinear operations in the full-round encryption, such as the ratio of the numbers of S-boxes for an SPN block cipher.

[2] For the sake of simplicity, we consider the scenario where $\alpha = \alpha'$ and $\beta = \beta'$. However, these technology are equally applicable the scenario where they are not equal.

Case $r_b = 1$: For a q-bit bijective S-box S and a given output difference ν, let $\mathcal{U}$ denote the set of input differences that can propagate to the output difference ν and $\mathcal{N} = |\mathcal{U}|$. Since $|\{(x, x \oplus \mu)|S(x \oplus k) \oplus S(x \oplus \mu \oplus k) = \nu, \mu \in \mathcal{U}\}| = 2^q$ and $|\{(x, x \oplus \mu)|x \in \mathbb{F}_2^q, \mu \in \mathcal{U}\}| = \mathcal{N}2^q$, it follows that for $\forall(x_0, x_1) \in \{(x, x \oplus \mu)|x \in \mathbb{F}_2^q, \mu \in \mathcal{U}\}$ and a given $k \in \mathbb{F}_2^q$, the probability that $S(x_0 \oplus k) \oplus S(x_1 \oplus k) = \nu$ is $2^q/(\mathcal{N}2^q) = 1/\mathcal{N}$. When $r_b = 1$, similarly for the S-box layer, $p_{in}^0 = 1/(\prod_{j \in J^0} \mathcal{N}_j^0(\eta_j^0)) = 1/|\Omega_{in}^0|$.

Case $r_b \geq 2$: Assume that for a state pair $(x_0, x_1) \in \{(x, x \oplus \mu)|\mu \in \Omega_{in}^1\}$ in round 1, the probability that $E_k^{r_b-1}(x_0) \oplus E_{k \oplus \Delta k}^{r_b-1}(x_1) = \alpha$ is $p_{in}^1 = 1/|\Omega_{in}^1|$. For each $\eta^0 \in \Omega_{in}^1$, there are $\prod_{j \in J^0} \mathcal{N}_j^0(\eta_j^0)$ possible plaintext pairs in Ω_{in}^0 that may propagate to η^0, but only one pair can reach the η^0 according to the analysis in the case $r_b = 1$. Since the pairs cannot satisfy two different values of η^0 simultaneously, a total of $|\Omega_{in}^1|$ plaintext pairs in Ω_{in}^0 can reach the state difference in Ω_{in}^1. Based on the conditional probability formula, $p_{in}^0 = |\Omega_{in}^1|/|\Omega_{in}^0|1/| \times \Omega_{in}^1| = 1/|\Omega_{in}^0|$.

We illustrate the application of the pre-sieving technique using the example of IDS key recovery. The subsequent part details only the steps that differ from those outlined in Section 2.3.

IDS.2': Produce quartets.

-IDS.2a: Construct plaintext pairs. Compute Ω_{in}^r $(0 \leq r \leq r_b - 1)$ by back-propagating α based on the key differences. For $2^{s+d_{in}}$ plaintexts, construct $\mathcal{P} = |\Omega_{in}^0|2^{s+d_{in}}$ pairs (P_0, C_0, P_1, C_1) and $\mathcal{P}$ pairs (P_2, C_2, P_3, C_3). Insert (P_0, C_0, P_1, C_1) into a hash table H_0 which is indexed by $n - d_{out}$ bits of C_0 and $n - d_{out}$ bits of C_1. For each (P_2, C_2, P_3, C_3), search the hash table H_0 to find the corresponding (P_0, C_0, P_1, C_1). On average, we can find $|\Omega_{in}^0|2^{(s+d_{in}-2(n-d_{out}))} = |\Omega_{in}^0|2^{s+d_{in}+2(d_{out}-n)}$ (P_0, C_0, P_1, C_1) corresponding to each (P_2, C_2, P_3, C_3). Eventually, we obtain $\mathcal{Q} = |\Omega_{in}^0|^2 2^{s+d_{in}+s+d_{in}+2(d_{out}-n)} = |\Omega_{in}^0|^2 2^{2s+2d_{in}+2d_{out}-2n}$ quartets that possess differences in Ω_{in}^0 and Ω_{out}.

-IDS.3a': Rather than applying the traditional early abort technique to recover the K_{in} for the $\mathcal{Q}$ quartets, we adopt the following steps to recover the keys.

-IDS.3a'.1: At round 0, let η^0 and η'^0 be the output differences of the S-box layer in round 0, which are derived from α and the corresponding key differences. Then, there are $|\Omega_{in}^1|^2$ possible values for (η^0, η'^0). For each value of (η^0, η'^0), there are $(\prod_{j \in J^0} \mathcal{N}_j^0(\eta_j^0)\mathcal{N}_j^0(\eta_j'^0))2^{2s+2d_{in}+2d_{out}-2n}$ quartets that might propagate to it. For these quartets and for each $j \in J^0$, we guess the 2^q possible keys $IK_{0,j}^0$ and then filter the quartets successively according to the output differences $(\eta_j^0, \eta_j'^0)$. Eventually, for each guessed key, $|\Omega_{in}^1|^2 2^{2s+2d_{in}+2d_{out}-2n}$ quartets remain.

-IDS.3a'.2 - IDS.3a'.$(r_b - 1)$: At round r $(r \geq 1)$, let η^r and η'^r be the output differences of the S-box layer in round r, which are derived from α and the corresponding key differences. Then, under each guessed key, there are $|\Omega_{in}^{r+1}|^2$ possible values for (η^r, η'^r). Under each guessed key and for each value of (η^r, η'^r), there are $(\prod_{j \in J^r} \mathcal{N}_j^r(\eta_j^r)\mathcal{N}_j^r(\eta_j'^r))2^{2s+2d_{in}+2d_{out}-2n}$ quartets that might propagate to it. For these quartets and for each $j \in J^r$, we guess the 2^q possible keys $IK_{0,j}^r$ as well as the necessary unguessed keys in rounds $0 - (r-1)$, and then filter the quartets successively according to the output differences $(\eta_j^r, \eta_j'^r)$. Finally, for each guessed key, $|\Omega_{in}^{r+1}|^2 2^{2s+2d_{in}+2d_{out}-2n}$ quartets remain.

-IDS.3a'.r_b: At round $r_b - 1$, let η^{r_b-1} and η'^{r_b-1} be the output differences of the S-box layer in round $r_b - 1$, which are derived from α and the corresponding key differences. Then, the values of η^r and η'^r are uniquely determined by the input difference of the distinguisher α. Under each guessed key, there are $(\prod_{j \in J^{r_b-1}} \mathcal{N}_j^{r_b-1}(\eta_j^{r_b-1})\mathcal{N}_j^{r_b-1}(\eta_j'^{r_b-1}))2^{2s+2d_{in}+2d_{out}-2n}$ quartets that might propagate to it. For these quartets and for each $j \in J^{r_b-1}$, we guess the 2^q possible keys $IK_{0,j}^{r_b-1}$ and the necessary unguessed keys in rounds $0-(r_b-2)$, and then filter the quartets successively according to the output differences $(\eta_j^{r_b-1}, \eta_j'^{r_b-1})$. Eventually, for each guessed key, $2^{2s+2d_{in}+2d_{out}-2n}$ quartets remain.

Complexity. The data complexity is $\mathcal{DC}_{IDS'} = 2^{2+s+d_{in}}$. Regarding the time complexity, it comprises five components:

- Cost of data generation: $\mathcal{D} = 2^{2+s+d_{in}}$.
- Cost of building pairs: $2\mathcal{P}$, where $\mathcal{P} = |\Omega_{in}^0|2^{s+d_{in}}$.

- Cost of producing quartets: $\mathcal{Q} = |\Omega^0_{in}|^2 2^{2s+2d_{in}+2d_{out}-2n}$.
- Cost of recovering keys: The time complexity of this step is rather intricate, and thus we will conduct an in-depth discussion hereinafter.
- Cost of final exhaustive search: After the step IDS.3a', there remain $\mathcal{Q}' = 2^{2s+2d_{in}+2d_{out}-2n}$ quartets. Since the last r_f-round employs the early abort technique to recover keys, the probability for a given quartet to discard a key is $2^{-2c_{out}}$. Hence, the probability of not discarding a key is:

  $$p = (1 - 2^{-2c_{out}})^{\mathcal{Q}'}.$$

  Therefore, the time complexity of this step is $p \cdot 2^k = 2^k(1 - 2^{-2c_{out}})^{\mathcal{Q}'}$.

The memory complexity will be determined by the cost of storing the data, pairs, quartets and remaining keys: $\mathcal{MC}_{IDS'} = \mathcal{D} + 2\mathcal{P} + \mathcal{Q} + \mathcal{K}$, where $\mathcal{K} = 2^{|K_{in} \cup K_{out}|}$.

Now, let's discuss the cost of recovering keys. To begin with, for general S-boxes, we present the following observation.

Observation 1 *For an S-box with a size of s bits, given two output differences ν_0 and ν_1, let $\mathcal{U}_0$ and $\mathcal{U}_1$ be the sets of input differences that are capable of propagating to the output differences ν_0 and ν_1, respectively. Also, let $\mathcal{N}_0 = |\mathcal{U}_0|$ and $\mathcal{N}_1 = |\mathcal{U}_1|$. Then, it holds that $\mathcal{N}_0\mathcal{N}_1 > 2^q$.*

Lemma 1. *For the step IDS.3a'.1, its time complexity is mainly determined by $T^1_1 = |\Omega^1_{in}|^2 2^q(\prod_{j \in J^0} \mathcal{N}^0_j(\eta^0_j)\mathcal{N}^0_j(\eta'^0_j))2^{2s+2d_{in}-}$ where $C'_{E,1}$ represents the ratio of the cost for two S-box operations to the full encryption.*

Proof. Let $J^0 = \{j^0_0, \ldots, j^0_{N^0_a-1}\}$. For each of the $|\Omega^1_{in}|^2$ values of (η^0, η'^0), we have

$$T^1_{1,i} = (2^q)^{i+1}(\prod_{j \in J^0 \setminus \{j^0_0, \ldots, j^0_{i-1}\}} \mathcal{N}^0_j(\eta^0_j)\mathcal{N}^0_j(\eta'^0_j))2^{2s+2d_{in}+2d_{out}-2n}C'_{E,1} \quad (0 \leq i \leq N^0_a - 1).$$

Based on the observation, it follows that $T^1_{1,i} > T^1_{1,i+1}$ for $0 \leq i \leq N^0_a - 2$ with respect to any fixed values of (η^0, η'^0). In other words, for each (η^0, η'^0), the time complexity is mainly determined by $T^1_{1,0}$. Hence, the time complexity of step IDS.3a'.1 is predominantly determined by T^1_1.

Lemma 2. *For the step IDS.3a'.$(r+1)$ within the range from IDS.3a'.2 to IDS.3a'.$(r_b - 1)$, its time complexity is upper-bounded by*

$$T^1_r = |\Omega^{r+1}_{in}|^2 2^q 2^{NK^1_{r,0}+NK^1_{r,1}}(\prod_{j \in J^r} \mathcal{N}^r_j(\eta^r_j)\mathcal{N}^r_j(\eta'^r_j))2^{2s+2d_{in}+2d_{out}-2n}C'_{E,1},$$

where $C'_{E,1}$ denotes the ratio of the cost for two S-box operations to the full encryption, $NK^1_{r,0}$ represents the number of bits of the keys that have already been guessed in the previous $(r-1)$ rounds, and $NK^1_{r,1}$ stands for the number of bits of the keys that need to be guessed for round r in the previous $(r-1)$ rounds.

Proof. Let $J^r = \{j^r_0, \ldots, j^r_{N^r_a-1}\}$. Consider an extreme case in which, for each key guess at round r, we are required to guess the $NK^1_{r,1}$ bits of keys in the previous $(r-1)$ rounds as well. Then, for $0 \leq i \leq N^r_a - 1$, we have

$$T^1_{r,i} = (2^q)^{i+1} 2^{NK^1_{r,0}+NK^1_{r,1}}(\prod_{j \in J^r \setminus \{j^r_0, \ldots, j^r_{i-1}\}} \mathcal{N}^r_j(\eta^r_j)\mathcal{N}^r_j(\eta'^r_j))2^{2s+2d_{in}+2d_{out}-2n}C'_{E,1}.$$

Based on the observation, it follows that $T^1_{r,i} > T^1_{r,i+1}$ for $0 \leq i \leq N^r_a - 2$ with respect to any fixed values of (η^r, η'^r). In other words, for each (η^r, η'^r), the time complexity is mainly determined by $T^1_{r,0}$. Hence, the time complexity of step IDS.3a'.$(r+1)$ is upper-bounded by T^1_r.

Lemma 3. *For the step IDS.3a'.r_b, let $NK^1_{r_b-1,0}$ denote the number of bits of the keys that have already been guessed in the previous (r_b-2) rounds, and let $K^1_{1,j_i^{r_b-1}}$ represent the keys that need to be additionally guessed in the previous (r_b-2) rounds for the $j_i^{r_b-1}$-th S-box at round r_b-1 $(0 \le i \le N_a^{r_b-1})$. Then, the time complexity of this step is bounded by*

$$T^1_{r_b-1} = \sum_{i=0}^{N_a^{r_b-1}-1} T^1_{r_b-1,j_i^{r_b-1}},$$

where

$$T^1_{r_b-1,i} = (2^q)^{i+1} 2^{NK} \Big(\prod_{j \in J^{r_b-1} \backslash \{j_0^{r_b-1},\ldots,j_{i-1}^{r_b-1}\}} \mathcal{N}_j^{r_b-1}(\eta_j^{r_b-1}) \mathcal{N}_j^{r_b-1}(\eta_j'^{r_b-1}) \Big) 2^{2s+2d_{in}+2d_{out}-2n} C'_{E,1}$$

and $NK = NK^1_{r_b-1,0} + \left| \bigcup_{u=0}^{i} K^1_{1,j_u^{r_b-1}} \right|$.

Lemma 3 is rather intuitive, and thus we will not provide a separate proof for it. It should be noted that only a bound for step IDS.3a'.r_b is presented here. Adopting another key recovery order might lead to a tightened value of T_{r_b-1}. The way to conduct the key recovery in step IDS.3a'.r_b and step IDS.3b so as to obtain the globally optimal time complexity will be discussed in the subsequent section. Eventually, we arrive at the following theorem.

Theorem 2. *The time cost of recovering keys by means of the pre-sieve technique is estimated as follows:*

$$T^1 = \sum_{i=0}^{r_b-1} T_i^1 + 2^{|K_{in}|} \times (\mathcal{Q}' \times 2^{|K_{out}/K_{in}|-2c_{out}}) C'_E,$$

where $\mathcal{Q}' = 2^{2s+2d_{in}+2d_{out}-2n}$ and C'_E stands for the ratio of the cost for one partial encryption to the full encryption.

Proof. Once the step IDS.3a' is completed, there are $\mathcal{Q}'$ quartets remaining, and $|K_{in}|$ bits of keys have already been guessed at this point. Subsequently, in accordance with the estimation approach for the time complexity of the early abort technique, the time complexity for recovering K_{out} is given by $2^{|K_{in}|} \times (\mathcal{Q}' \times 2^{|K_{out}/K_{in}|-2c_{out}}) C'_E$. By combining this result with Lemma 1 through Lemma 3, we have thereby proven our theorem.

Undoubtedly, we can also choose to recover the keys of the last r_f rounds first. In such case, we will have guessed $|K_{out}|$ bits of the keys, and the number of remaining quartets is $\mathcal{Q}'' = |\Omega^0_{in}|^2 2^{2s+2d_{in}-2n}$, which is reduced by a factor of $2^{2d_{out}}$ for the first r_b rounds. All things considered, we obtain the following results.

Corollary 1. *If the keys of the last r_f rounds are recovered first, the time cost for recovering keys using the pre-sieve technique is estimated to be*

$$(2^{|K_{out}|-2d_{out}}) \sum_{i=0}^{r_b-1} T_i^1 + (\mathcal{Q} \times 2^{|K_{out}|-2c_{out}}) C'_E,$$

where $\mathcal{Q} = |\Omega^0_{in}|^2 2^{2s+2d_{in}+2d_{out}-2n}$ and C'_E represents the ratio of the cost for one partial encryption to the full encryption.

3.2 Partial pre-guess key technique

In the impossible differential style key recovery, the attacker does not make any pre-guesses regarding the keys. Meanwhile, in the boomerang style key recovery, the attacker pre-guesses all of the K_{in} or K_{out}. In other words, within the existing key recovery methods, the attacker either refrains from making any pre-guesses or is compelled to guess the entire key of either K_{in} or K_{out}. This situation naturally gives rise to two issues. On the one hand, if we choose to guess all of K_{in}, for the majority of items related to the attack complexity in the boomerang-style key recovery, it becomes necessary to multiply them by the value of $2^{|K_{in}|}$. Consequently, if the value of $|K_{in}|$ is excessively large, the overall complexity might surpass the maximum complexity that the attack can tolerate. In such a scenario, we can only pre-guess values that are less than $2^{|K_{in}|}$. On the other hand, aside from pre-guessing K_{in}, if we are also able to guess a portion of K_{out}, then the value of d_{out} will decrease. As a result, the overall complexity may also be lowered. In this situation, pre-guessing values greater than $2^{|K_{in}|}$ proves to be more advantageous for reducing the overall complexity of the attack. Therefore, being able to select the pre-guessed keys in a flexible manner would be highly beneficial for conducting the attack. To achieve this objective, we propose the partial pre-guess key technique.

Regarding the states, differences, and keys, we partition them into blocks based on the size of the S-box. Specifically, assuming that the S-box layer consists of t S-boxes, we define $\mathbb{X}_j^r = (IX_{0,j}^r, IX_{3,j}^r, \Delta X_{01,j}^r, \Delta X_{23,j}^r)$, $\mathbb{Y}_j^r = (IY_{0,j}^r, IY_{3,j}^r, \Delta Y_{01,j}^r, \Delta Y_{23,j}^r)$, $\mathbb{Z}_j^r = (IZ_{0,j}^r, IZ_{3,j}^r, \Delta Z_{01,j}^r, \Delta Z_{23,j}^r)$, and $\mathbb{K}_j^r = (IK_{0,j}^r, IK_{1,j}^r, IK_{2,j}^r, IK_{3,j}^r)$ $(0 \leq j \leq t-1)$.

Given the input difference α of the distinguishers along with the key differences, we are able to back-propagate them for r_b rounds. Consequently, we can define the flag of $\mathbb{X}_j^r$ in the following manner:

$$f\mathbb{X}_j^r = \begin{cases} 0, \text{ if } \Delta X_{01,j}^r \text{ and } \Delta X_{23,j}^r \text{ are inactive,} \\ 1, \text{ if } \Delta X_{01,j}^r \text{ and } \Delta X_{23,j}^r \text{ are active and known,} \\ 2, \text{ if } \Delta X_{01,j}^r \text{ and } \Delta X_{23,j}^r \text{ are unknown.} \end{cases}$$

In the same format, we define $f\mathbb{Y}_j^r$, $f\mathbb{Z}_j^r$, and $f\mathbb{K}_j^r$. Subsequently, we define a type of directed graph for the key recovery during the first r_b rounds.

Definition 4. *The key recovery graph of the first r_b rounds, denoted as $\mathcal{G}_b$, is a directed graph. For $r = r_b - 1, \ldots, 0$, its vertices and edges are defined as follows.*

- *LL: For $0 \leq j \leq t-1$, if $\mathbb{X}_j^{r+1}$ is a vertex or, although $\mathbb{X}_j^{r+1}$ is not a vertex, there exists $\mathbb{Z}_v^r$ with $f\mathbb{Z}_v^r = 2$ that influences $\mathbb{X}_j^{r+1}$, then add $\mathbb{X}_j^{r+1}$ and all such $\mathbb{Z}_v^r$ that influence $\mathbb{X}_j^{r+1}$ as vertices to the graph. Also, add edges directed from each $\mathbb{Z}_v^r$ to $\mathbb{X}_j^{r+1}$.*
- *SL: For $0 \leq j \leq t-1$, if $\mathbb{Z}_j^r$ is a vertex or $f\mathbb{Z}_j^r = 1$, then add $\mathbb{Z}_j^r$ and $\mathbb{Y}_j^r$ as vertices to the graph. Additionally, add an edge directed from $\mathbb{Y}_j^r$ to $\mathbb{Z}_j^r$.*
- *AKL_{k_r}: For $0 \leq j \leq t-1$, if $\mathbb{Y}_j^r$ is a vertex, then add $\mathbb{X}_j^r$ as a vertex to the graph. Also, add an edge directed from $\mathbb{X}_j^r$ to $\mathbb{Y}_j^r$, and this edge is named as $\mathbb{K}_j^r$.*

In the graph $\mathcal{G}_b$, certain vertices and edges possess special characteristics, which we define as follows.

Definition 5. *For the graph $\mathcal{G}_b$ and its subgraphs, a vertex that does not direct edges to other vertices is referred to as a sink vertex. Conversely, vertices that are not pointed to by other vertices are called source vertices. Additionally, the edge $\mathbb{K}_j^r$ is termed the associated key.*

Undoubtedly, the source vertices are $\mathbb{X}_j^0$ $(j \in J)$, where J denotes the set of indices for which $\mathbb{X}_j^0$ functions as a source vertex within $\mathcal{G}_b$ or its subgraphs. Given that the differences of the sink vertices are known, we can establish a relationship that encompasses all the source vertices influencing a specific sink vertex and then utilize this relationship to filter the keys. In other words, a sink vertex represents a particular condition. To facilitate independent guessing of the keys, we introduce the concept of independent subgraphs, which is defined as follows.

Definition 6. *A subgraph $\mathcal{G}_{bs}$ is referred to as a basic subgraph of $\mathcal{G}_b$ if $\mathcal{G}_{bs}$ contains a source vertex along with all the vertices and edges of $\mathcal{G}_b$ that have an influence on this source vertex. A subgraph $\mathcal{G}_{bi}$ is called an independent subgraph of $\mathcal{G}_b$ if it is composed of basic subgraphs and if, given the values of $IX^0_{0,j}$ and $IX^0_{3,j}$ as well as the associated key of $\mathcal{G}_{bi}$ ($j \in J$), where J is the set of indices for which $\mathbb{X}^0_j$ serves as a source vertex within $\mathcal{G}_{bi}$, the values of $IX^0_{1,j}$ and $IX^0_{2,j}$ can be determined.*

For an independent subgraph $\mathcal{G}_{bi}$, for all vertices $\mathbb{Z}^r_j \in \mathcal{G}_{bi}$ $(0 \leq j \leq t-1)$ and for each $\mathbb{X}^{r+1}_u$ $(0 \leq u \leq t-1)$ that influences $\mathbb{Z}^r_j$, if $\mathbb{X}^{r+1}_u$ is not a vertex of $\mathcal{G}_{bi}$, then $f\mathbb{X}^{r+1}_u \leq 1$. Otherwise, we would encounter unknown differences when attempting to recover $IX^0_{1,j}$ and $IX^0_{2,j}$. Moreover, the independent subgraph $\mathcal{G}_{bi}$ has the following property.

Property 1. The independent subgraph $\mathcal{G}_{bi}$ contains at least one source vertex in the form of $\mathbb{Z}^r_j$ or $\mathbb{Y}^r_j$ $(0 \leq r \leq r_b - 1,\ 0 \leq j \leq t-1)$.

Proof. Suppose that all the source vertices of $\mathcal{G}_{bi}$ are $\mathbb{X}^r_j$ $(0 \leq r \leq r_b - 1,\ 0 \leq j \leq t-1)$. Let r_0 denote the maximum value of r such that $\mathbb{X}^{r_0}_{j_0}$ $(0 \leq r_0 \leq r_b - 1,\ 0 \leq j_0 \leq t-1)$ is a source vertex. Then, there exists a $\mathbb{Z}^{r_0-1}_u$ with $f\mathbb{Z}^{r_0-1}_u = 2$ $(0 \leq u \leq t-1)$ that influences $\mathbb{X}^{r_0}_{j_0}$. Since $f\mathbb{Z}^{r_0-1}_u = 2$, there exists at least one non-source vertex $\mathbb{X}^{r_0}_{j_1}$ $(0 \leq j_1 \leq t-1)$ within $\mathcal{G}_{bi}$ that influences $\mathbb{Z}^{r_0-1}_u$. Consequently, the source vertex that is influenced by $\mathbb{X}^{r_0}_{j_1}$ is also a source vertex of $\mathcal{G}_{bi}$. However, this source vertex cannot be $\mathbb{X}^r_j$ $(0 \leq r \leq r_b - 1,$ $0 \leq j \leq t-1)$, because r_0 is the maximum value of r for which $\mathbb{X}^{r_0}_{j_0}$ is a source vertex. Hence, the property holds.

Based on Property 1, we can construct the independent subgraphs $\mathcal{G}_{bi}$ for all source vertices in the form of $\mathbb{Z}^r_j$ or $\mathbb{Y}^r_j$ $(0 \leq r \leq r_b - 1,\ 0 \leq j \leq t-1)$. Specifically, for each source vertex, we examine whether the corresponding basic subgraph is an independent subgraph. If it is, we retain it. If not, according to our previous analysis, there exist values of r and j such that there exists $\mathbb{X}^r_j$ that is not a vertex of the independent subgraph $\mathcal{G}_{bi}$. In such a case, we simply merge the basic subgraph that contains $\mathbb{X}^r_j$ with $\mathcal{G}_{bi}$, and continue this process analogously until an independent subgraph is formed.

Analogously, for the last r_f rounds, we define $\overline{\mathbb{X}}^r_j = (IX^r_{1,j}, IX^r_{0,j}, \Delta X^r_{12,j}, \Delta X^r_{03,j})$, $\overline{\mathbb{Y}}^r_j = (IY^r_{1,j}, IY^r_{0,j}, \Delta Y^r_{12,j}, \Delta Y^r_{03,j})$, and $\overline{\mathbb{Z}}^r_j = (IZ^r_{1,j}, IZ^r_{0,j}, \Delta Z^r_{12,j}, \Delta Z^r_{03,j})$. Given the output difference β of the distinguisher along with the key differences, we can propagate them for r_f rounds in the forward direction. Consequently, we can define the flag of $\overline{\mathbb{X}}^r_j$ in the following manner:

$$f\overline{\mathbb{X}}^r_j = \begin{cases} 0, & \text{if } \Delta X^r_{12,j} \text{ and } \Delta X^r_{03,j} \text{ are inactive,} \\ 1, & \text{if } \Delta X^r_{12,j} \text{ and } \Delta X^r_{03,j} \text{ are active and known,} \\ 2, & \text{if } \Delta X^r_{12,j} \text{ and } \Delta X^r_{03,j} \text{ are unknown.} \end{cases}$$

In the same format, we define $f\overline{\mathbb{Y}}^r_j$, $f\overline{\mathbb{Z}}^r_j$, and $f\overline{\mathbb{K}}^r_j$. Subsequently, we define a type of directed graph for the key recovery during the last r_f rounds.

Definition 7. *The key recovery graph of the last r_f rounds, denoted as $\mathcal{G}_f$, is a directed graph. For $r = r_b + r_d, \ldots, r_b + r_d + r_f$, its vertices and edges are defined as follows.*

- *AKL_{k_r}: For $0 \leq j \leq t-1$, if $\overline{\mathbb{X}}^r_j$ is a vertex, then add $\overline{\mathbb{Y}}^r_j$ as a vertex to the graph. Also, add an edge directed from $\overline{\mathbb{Y}}^r_j$ to $\overline{\mathbb{X}}^r_j$, and this edge is named as $\overline{\mathbb{K}}^r_j$.*
- *SL: For $0 \leq j \leq t-1$, if $\overline{\mathbb{Y}}^r_j$ is a vertex or $f\overline{\mathbb{Y}}^r_j = 1$, then add $\overline{\mathbb{Z}}^r_j$ and $\overline{\mathbb{Y}}^r_j$ as vertices to the graph. Additionally, add an edge directed from $\overline{\mathbb{Z}}^r_j$ to $\overline{\mathbb{Y}}^r_j$.*
- *LL: For $0 \leq j \leq t-1$, if $\overline{\mathbb{Z}}^r_j$ is a vertex or, although $\overline{\mathbb{Z}}^r_j$ is not a vertex, there exists $\overline{\mathbb{X}}^{r+1}_v$ with $f\overline{\mathbb{X}}^{r+1}_v = 2$ that influences $\overline{\mathbb{Z}}^r_j$, then add $\overline{\mathbb{Z}}^r_j$ and all such $\overline{\mathbb{X}}^{r+1}_v$ that influence $\overline{\mathbb{Z}}^r_j$ as vertices to the graph. Also, add edges directed from each $\overline{\mathbb{X}}^{r+1}_v$ to $\overline{\mathbb{Z}}^r_j$.*

The definitions of sink vertex, source vertex, and associated key for the key recovery graph $\mathcal{G}_b$ are also applicable to $\mathcal{G}_f$. Similarly, the definitions of basic subgraphs and independent subgraphs are analogous.

Definition 8. *A subgraph $\mathcal{G}_{fs}$ is called a basic subgraph of $\mathcal{G}_f$ if $\mathcal{G}_{fs}$ contains a source vertex along with all the vertices and edges of $\mathcal{G}_f$ that have an influence on this source vertex. A subgraph $\mathcal{G}_{fi}$ is called an independent subgraph of $\mathcal{G}_f$ if it is composed of basic subgraphs and if, given the values of $IY_{0,j}^{r_b+r_d+r_f}$ and $IY_{1,j}^{r_b+r_d+r_f}$ as well as the associated key of $\mathcal{G}_{fi}$ ($j \in J$), where J is the set of indices for which $\overline{\mathbb{Y}}_j^{r_b+r_d+r_f}$ serves as a source vertex within $\mathcal{G}_{fi}$, the values of $IY_{2,j}^{r_b+r_d+r_f}$ and $IY_{3,j}^{r_b+r_d+r_f}$ can be determined.*

For an independent subgraph $\mathcal{G}_{fi}$, for all vertices $\overline{\mathbb{X}}_j^{r+1} \in \mathcal{G}_{fi}$ $(0 \leq j \leq t-1)$ and for each $\overline{\mathbb{Z}}_u^r$ $(0 \leq u \leq t-1)$ that influences $\overline{\mathbb{X}}_j^{r+1}$, if $\overline{\mathbb{Z}}_u^r$ is not a vertex of $\mathcal{G}_{fi}$, then $f\overline{\mathbb{Z}}_u^r \leq 1$. Otherwise, we would encounter unknown differences when attempting to recover $IY_{2,j}^0$ and $IY_{3,j}^0$. Moreover, the independent subgraph $\mathcal{G}_{fi}$ has a property similar to that of the independent subgraph $\mathcal{G}_{bi}$, which provides a method for constructing the independent subgraph $\mathcal{G}_{fi}$.

Property 2. The independent subgraph $\mathcal{G}_{fi}$ contains at least one source vertex in the form of $\mathbb{X}_j^r$ or $\mathbb{Y}_j^r$ $(0 \leq r \leq r_b - 1, 0 \leq j \leq t-1)$.

For a block cipher, we can calculate all the independent subgraphs of the key recovery graph of the first r_b rounds $\mathcal{G}_b$ and the key recovery graph of the last r_f rounds $\mathcal{G}_f$. The associated keys of these graphs can be pre-guessed. Assume that we choose to pre-guess the associated keys of l_b independent subgraphs $\mathcal{G}_{bi,i_b}$ $(0 \leq i_b \leq l_b - 1)$ and l_f independent subgraphs $\mathcal{G}_{fi,i_f}$ $(0 \leq i_f \leq l_f - 1)$. Before presenting the attack in detail, we introduce the following new notations.

- Δ_{pin}, d_{pin}: Δ_{pin} represents the set of the part of plaintext differences that is involved in $\mathcal{G}_{bi,i_b}$ $(0 \leq i_b \leq l_b - 1)$, and $d_{pin} = \log_2 |\Delta_{pin}|$.
- Δ_{rin}, d_{rin}: Δ_{rin} represents the set of the part of plaintext differences that is not involved in $\mathcal{G}_{bi,i_b}$ $(0 \leq i_b \leq l_b - 1)$, and $d_{rin} = \log_2 |\Delta_{rin}|$.
- Δ_{pout}, d_{pout}: Δ_{pout} represents the set of the part of ciphertext differences that is involved in $\mathcal{G}_{fi,i_f}$ $(0 \leq i_f \leq l_f - 1)$, and $d_{pout} = \log_2 |\Delta_{pout}|$.
- Δ_{rout}, d_{rout}: Δ_{rout} represents the set of the part of ciphertext differences that is not involved in $\mathcal{G}_{fi,i_f}$ $(0 \leq i_f \leq l_f - 1)$, and $d_{rout} = \log_2 |\Delta_{rout}|$.
- K_{pin}, K_{pout}: the associated keys involved in $\mathcal{G}_{bi,i_b}$ $(0 \leq i_b \leq l_b - 1)$ and $\mathcal{G}_{fi,i_f}$ $(0 \leq i_f \leq l_f - 1)$.
- K_{rin}, K_{rout}: $K_{rin} = K_{in}/K_{pin}$, and $K_{rout} = K_{out}/K_{pout}$.
- c_{rin}, c_{rout}: $2^{-c_{rin}}$ and $2^{-c_{rout}}$ denote the probabilities of reaching the distinguisher differences α and β from differences in Δ_{rin} and Δ_{rout} under the values of Δ_{pin} and Δ_{pout}, respectively.

Take the boomerang style key recovery as an example. The overall improved attack process is detailed as follows. The overall improved attack can be divided into four steps: **BS.1′**, **BS.2′**, and **BS.3′**. The step **BS.1′** and **BS.3′** are the same as **BS.1** and **BS.3** of the boomerang style. The step **IDS.2′** is detailed as follows.

-BS.2': Guess all the keys $K_{pin} \cup K_{pout}$:

- **-BS.2a':** For all possible values of $(IX_{0,j}^0, IX_{3,j}^0)$ in the source vertex $\mathbb{X}_j^0$ of $\mathcal{G}_{bi,i_b}$ $(0 \leq i_b \leq l_b - 1)$, obtain the corresponding $(IX_{1,j}^0, IX_{2,j}^0)$ and store them in the table PT. Similarly, for all possible values of $(IY_{1,j}^{r_b+r_d+r_f}, IY_{0,j}^{r_b+r_d+r_f})$ in the source vertex $\mathbb{Y}_j^{r_b+r_d+r_f}$ of $\mathcal{G}_{fi,i_f}$ $(0 \leq i_f \leq l_f - 1)$, obtain the corresponding $(IY_{2,j}^{r_b+r_d+r_f}, IY_{3,j}^{r_b+r_d+r_f})$ and store them in the table CT.
- **-BS.3b':** Under the $2^{s+d_{in}}$ plaintexts, for the index j where $\mathbb{X}_j^0$ is the source vertex of $\mathcal{G}_{b,i}^{kv}$ $(0 \leq i \leq l_b - 1)$, look up PT to obtain the value of $(IX_{1,j}^0, IX_{2,j}^0)$. For the remaining positions of active differences, enumerate all values of difference. Finally, we can construct $\mathcal{P} = 2^{s+d_{pin}+2d_{rin}}$ pairs.

- **-BS.3c':** Store $(IX_0^0, IY_0^{r_b+r_d+r_f}, IX_1^0, IY_1^{r_b+r_d+r_f})$ into a hash table H_0 indexed by $n - |\Omega_{out}|$ bits of C_0 and $n - |\Omega_{out}|$ bits of C_1 and indexed by the index j where $\mathbb{Y}_j^{r_b+r_d+r_f}$ is the source vertex of $\mathcal{G}_{f,i}^{kv}$ $(0 \leq i \leq l_f - 1)$. For each $(IX_2^0, IY_2^{r_b+r_d+r_f}, IX_3^0, IY_3^{r_b+r_d+r_f})$, look up the hash table H_0 to find the corresponding $(IX_0^0, IY_0^{r_b+r_d+r_f}, IX_1^0, IY_1^{r_b+r_d+r_f})$. For the index j where $\mathbb{Y}_j^{r_b+r_d+r_f}$ is the source vertex of $\mathcal{G}_{f,i}^{kv}$ $(0 \leq i \leq l_f - 1)$, look up the table CT to obtain the possible value of $(IY_{1,j}^{r_b+r_d+r_f}, IY_{0,j}^{r_b+r_d+r_f})$. On average, we can find $2^{s+d_{pin}+2d_{rin}-(2(n-d_out)+2d_{pout})}$ (P_0, C_0, P_1, C_1) corresponding to each (P_2, C_2, P_3, C_3). Finally, we obtain $\mathcal{Q} = 2^{s+d_{pin}+2d_{rin}+s+d_{pin}+2d_{rin}-(2(n-d_out)+2d_{pout})} = 2^{2(s+d_{pin}+2d_{rin})+2(d_{out}-d_{pout}-n)}$ quartets that have differences in Ω_{in} and Ω_{out}.
- **-BS.4c':** Adopt the early abort technique to recover the remaining keys K_{rin} and K_{rout} for the $\mathcal{Q}$ quartets.

Complexity. The data complexity is $\mathcal{DC}_{BS'} = 2^{2+s+d_{in}}$. For the time complexity, it consists of five parts:

- Cost of data generation: $\mathcal{D} = 2^{2+s+d_{in}}$.
- Cost of building pairs: $2^{|K_{pin} \cup K_{pout}|} \times 2\mathcal{P} \times C'_E$, where $\mathcal{P} = 2^{s+d_{pin}+2d_{rin}}$ and C'_E is the cost of the partial encryption and decryption of building pairs.
- Cost of producing quartets: $2^{|K_{pin} \cup K_{pout}|} \times \mathcal{Q}$.
- Cost of adopting the early abort technique to recover keys: The time complexity of this step is estimated to be $2^{|K_{pin} \cup K_{pout}|} \times (\mathcal{Q} \times 2^{|K_{rin} \cup K_{rout}|-2(c_{rin}+c_{rout})}) C''_E$, where C''_E is the ratio of the cost for one partial encryption to the full encryption.
- Cost of final exhaustive search: Let N_b^r be the number of active S-boxes at round r that are not in the independent graph. In the early abort technique, the probability that the difference Δ_{rin} propagates to α is $p_{rin} = 2^{s \cdot N_b^1}/2^{s \cdot N_b^0} \times \cdots \times 1/2^{s \cdot N_b^{r_b-1}} = 2^{-s \cdot N_b^0} = 2^{-d_{rin}}$. Analogously, it holds $p_{rout} = 2^{-d_{rout}}$. Thus, $p = (1 - (p_{rin} p_{rout})^2)^{\mathcal{Q}} = e^{-2^{(2s+2d_{in}-2n)}}$. Therefore, the cost of this step is $p \cdot 2^k = e^{-2^{(2s+2d_{in}-2n)}} \times 2^k$.

The memory complexity will be determined by the cost of storing the data, pairs, quartets and remaining keys: $\mathcal{MC}_{BS'} = \mathcal{D} + 2\mathcal{P} + \mathcal{Q} + \mathcal{K}$, where $\mathcal{K} = 2^{|K_{in} \cup K_{out}|}$.

3.3 Precise complexity evaluation technique

Currently, automated methods enable the automatic search for the optimal key recovery strategy by modeling each step of the attack. Specifically, the attacker employs an approximate formula to assess the time complexity of the early abort technique. However, there are two issues. Firstly, if the time complexity of the early abort technique preponderates over the overall complexity, the outcome obtained using the approximation formula may not necessarily be the optimal one. Secondly, even if the approximation formula is utilized to derive the optimal solution, it is still necessary to manually deduce the specific key recovery process in order to present the detailed attack steps, which is undeniably complex and tedious.

First, we combine the pre-sieving technique with the partial pre-guess key technique and provide a unified account of the key recovery attack. It should be noted that the impossible differential style and boomerang style key recovery attacks are particular instances of our attack. The configuration of the attack is detailed as follows.

- **Pre-guessed key K_{pin}.** Compute all the independent subgraphs of the directed graphs for key recovery in the first r_b rounds. Then, select the associated keys of l_b independent subgraphs $\mathcal{G}_{bi,i_b}$ $(0 \leq i_b \leq l_b - 1)$ as K_{pin}.
- **Pre-guessed set K_{pout}.** Calculate all the independent subgraphs of the directed graphs of the key recovery in the last r_f rounds. Subsequently, choose the associated keys of l_f independent subgraphs $\mathcal{G}_{fi,i_f}$ $(0 \leq i_f \leq l_f - 1)$ as K_{pin}.
- **With/without the pre-sieving technique in the first r_b rounds.** Let Ω_{rin}^r denote the set of differences at round r that are obtained by propagating from the differences α and the key difference and are not involved in $\mathcal{G}_{bi,i_b}$ $(0 \leq i_b \leq l_b - 1)$, and let $d_{rin}^0 = \log_2 |\Omega_{rin}^0|$. Let N_b^r be the number of active S-boxes

at round r that are not in the independent graph and $J_b^r = \{j_0^r, \ldots, j_{N_b^r-1}^r\}$ be the index of the active S-boxes. If the early abort technique with the pre-sieving technique is used in the first r_b rounds, we pre-calculate the set of difference Ω_{rin}^0. Otherwise, this set Ω_{rin}^0 traverses all possible values of d_{rin} bits.

In the absence of the pre-sieving technique during the first r_b rounds, the key recovery attack is essentially the improved boomerang style key recovery attack. Therefore, in this context, we will only elaborate on the key recovery attack that incorporates the pre-sieving technique within the first r_b rounds.

-1: This step is identical to step IDS.1 of the impossible differential style.

-2: Guess all the keys $K_{pin} \cup K_{pout}$:

-2a: This step is the same as step BS.2a' of the improved boomerang style.

-2b: Under the $2^{s+d_{in}}$ plaintexts, for the index j of the source vertex $\mathbb{X}_j^0$ of $\mathcal{G}_{bi,i_b}$ $(0 \leq i_b \leq l_b - 1)$, we look up PT to obtain the value of $(IX_{1,j}^0, IX_{2,j}^0)$. For the remaining positions of active differences, we traverse all possible values of the difference of Ω_{rin}^0. Eventually, we can construct $\mathcal{P} = |\Omega_{rin}^0| 2^{s+d_{in}} = 2^{s+d_{rin}^0+d_{in}}$ pairs.

-2c: Insert $(IX_0^0, IY_0^{r_b+r_d+r_f}, IX_1^0, IY_1^{r_b+r_d+r_f})$ into a hash table H_0 which is indexed by $n - |\Omega_{out}|$ bits of C_0 and $n - |\Omega_{out}|$ bits of C_1 and also indexed by the index j where $\mathbb{Y}_j^{r_b+r_d+r_f}$ is the source vertex of $\mathcal{G}_{f,i}^{kv}$ $(0 \leq i \leq l_f - 1)$. For each $(IX_2^0, IY_2^{r_b+r_d+r_f}, IX_3^0, IY_3^{r_b+r_d+r_f})$, look up the hash table H_0 to find the corresponding $(IX_0^0, IY_0^{r_b+r_d+r_f}, IX_1^0, IY_1^{r_b+r_d+r_f})$. When the index j where $\mathbb{Y}_j^{r_b+r_d+r_f}$ is the source vertex of $\mathcal{G}_{f,i}^{kv}$ $(0 \leq i \leq l_f - 1)$, we look up the table CT to get the possible value of $(IY_{1,j}^{r_b+r_d+r_f}, IY_{0,j}^{r_b+r_d+rf})$. On average, we can find $2^{s+d_{rin}^0+d_{in}-(2(n-d_{out})+2d_{pout})}$ (P_0, C_0, P_1, C_1) corresponding to each (P_2, C_2, P_3, C_3). Finally, we obtain $\mathcal{Q} = 2^{s+d_{rin}^0+d_{in}+s+d_{rin}^0+2d_{in}-(2(n-d_{out})+2d_{pout})} = 2^{2(s+d_{rin}^0+d_{in})+2(d_{out}-d_{pout}-n)}$ quartets that have differences in Ω_{in} and Ω_{out}.

-2d: Recover the keys K_{rin} and K_{rout}. Assume that the keys in K_{rin} are recovered first, and then the keys in K_{rout} are recovered.

-2d.1: At round 0, let η^0 and η'^0 be the output difference of the S - box layer in round 0 and be derived from α and the corresponding key differences. Then there are $|\Omega_{rin}^1|^2$ possible values of (η^0, η'^0). For each value of (η^0, η'^0), there are $(\prod_{j \in J_b^0} \mathcal{N}_j^0(\eta_j^0)\mathcal{N}_j^0(\eta_j'^0))2^{2s+2d_{in}+2d_{out}-2d_{pout}-2n}$ quartets that may propagate to it. For those quartets and each $j \in J_b^0$, we guess the 2^s possible keys $IK_{0,j}^0$ and filter the quartets according to the output difference $(\eta_j^0, \eta_j'^0)$ in turn. Finally, for each guessed key, $|\Omega_{rin}^1|^2 2^{2s+2d_{in}+2d_{out}-2d_{pout}-2n}$ quartets remain.

-2d.2 - 2d.$(r_b - 1)$: At round r $(r \geq 1)$, let η^r and η'^r be the output difference of the S - box layer in round r and be derived from α and the corresponding key differences. Then there are $|\Omega_{rin}^{r+1}|^2$ possible values of (η^r, η'^r) under each guessed key. Under each guessed key, for each value of (η^r, η'^r), there are $(\prod_{j \in J_b^r} \mathcal{N}_j^r(\eta_j^0)\mathcal{N}_j^r(\eta_j'^r))2^{2s+2d_{in}+2d_{out}-2d_{pout}-2n}$ quartets that may propagate to it. For those quartets and each $j \in J_b^r$, we guess the 2^s possible keys $IK_{0,j}^r$ and the necessary unguessed keys in round $0 - (r-1)$, and filter the quartets according to the output difference $(\eta_j^r, \eta_j'^r)$ in turn. Finally, for each guessed key, $|\Omega_{rin}^{r+1}|^2 2^{2s+2d_{in}+2d_{out}-2d_{pout}-2n}$ quartets remain.

-2d.r_b : At round $r_b - 1$, let η^{r_b-1} and η'^{r_b-1} be the output difference of the S - box layer in round $r_b - 1$ and be derived from α and the corresponding key differences. Then the values of η^r and η'^r are uniquely determined by the input difference of the IBD. Under each guessed key, there are $(\prod_{j \in J_b^{r_b-1}} \mathcal{N}_j^{r_b-1}(\eta_j^{r_b-1})\mathcal{N}_j^{r_b-1}(\eta_j'^{r_b-1}))2^{2s+2d_{in}+2d_{out}-2d_{pout}-2n}$ quartets that may propagate to it. For those quartets and each $j \in J_b^{r_b-1}$, we guess the 2^s possible keys $IK_{0,j}^{r_b-1}$ and the necessary unguessed keys in round $0 - (r_b - 2)$, and filter the quartets according to the output difference $(\eta_j^{r_b-1}, \eta_j'^{r_b-1})$ in turn. Finally, for each guessed key, $\mathcal{Q}' = 2^{2s+2d_{in}+2d_{out}-2d_{pout}-2n}$ quartets remain.

-2d.$(r_b + 1)$: Adopt the early abort technique to recover the remaining keys K_{rout} for the $\mathcal{Q}'$ quartets.

-3: Exhaustively search for the remaining keys.

Complexity. The date complexity is $\mathcal{DC} = 2^{2+s+d_{in}}$. For the time complexity, it consists of five parts:

- Cost of data generation: $\mathcal{D} = 2^{2+s+d_{in}}$.
- Cost of building pairs: $2^{|K_{pin} \cup K_{pout}|} \times 2\mathcal{P} \times C'_E$, where $\mathcal{P} = 2^{s+d_{in}+d_{rin}}$ and C'_E is the cost of the partial encryption and decryption of building pairs.
- Cost of producing quartets: $2^{|K_{pin} \cup K_{pout}|} \times \mathcal{Q}$.
- Cost of recovering keys: Unlike the traditional methods that use estimation, we will present a method below to directly obtain the detailed key recovery steps, and then obtain the globally optimal time complexity.
- Cost of final exhaustive search: the cost of this step is $p \cdot 2^k$, where $p = (1 - 2^{-2c_{pout}})^{\mathcal{Q}'} = e^{-2^{2s+2d_{in}-2n}}$.

The memory complexity will be determined by the cost of storing the data, pairs, quartets and remaining keys: $\mathcal{MC} = \mathcal{D} + 2\mathcal{P} + \mathcal{Q} + \mathcal{K}$, where $\mathcal{K} = 2^{|K_{in} \cup K_{out}|}$.

The complexity of the other steps is very clear. Thus, we put forward the method for obtaining the time complexity associated with recovering keys, and then obtain an accurate representation of the overall complexity. Similar to the proof of Lemma 1-Lemma 3, we have the following results.

Lemma 4. *Regarding step 2d.1, its time complexity is mainly determined by* $T_1^2 = |\Omega_{rin}^1|^2 2^q (\prod_{j \in J_b^0} \mathcal{N}_j^0(\eta_j^0)\mathcal{N}_j^0(\eta_j'^0))2^{2s+2d_{in}+2}$ *where* $C'_{E,1}$ *represents the ratio of the cost for two S-box operations to the cost of a full encryption.*

Lemma 5. *For the step 2d.r between 2d.2 and 2d.*$(r_b - 1)$*, its time complexity is bounded by*

$$T_r^2 = |\Omega_{rin}^{r+1}|^2 2^q 2^{NK_{r,0}^2 + NK_{r,1}^2} (\prod_{j \in J_b^r} \mathcal{N}_j^r(\eta_j^r)\mathcal{N}_j^r(\eta_j'^r))2^{2s+2d_{in}+2d_{out}-2d_{pout}-2n} C'_{E,1},$$

where $C'_{E,1}$ *is the ratio of the cost for two S-box operations to the full encryption,* $NK_{r,0}^2$ *is the number of bits of the keys that has been guessed in the prior* $(r-1)$ *rounds already, and* $NK_{r,1}^2$ *is the number of bits of the keys that need to guessed for round* r *in the prior* $(r-1)$ *rounds.*

Lemma 6. *For the step 2d.*r_b*, let* $NK_{r_b-1,0}^2$ *is the number of bits of the keys that has been guessed in the prior* $(r_b - 2)$ *rounds already, and* $K^2_{1,j_i^{r_b-1}}$ *is the set of keys that need to addition guessed in the prior* $(r_b - 2)$ *rounds for the* $j_i^{r_b-1}$*-th S-box at round* $r_b - 1$ $(j_i^{r_b-1} \in J_b^{r_b-1}, 0 \le i \le N_b^{r_b-1} - 1)$*. For a given key recovery order* $j_{i_0}^{r_b-1}, \ldots, j_{i_{N_b^{r_b-1}-1}}^{r_b-1}$*, the time complexity of this step is*

$$T_{r_b}^2 = \sum_{u=0}^{N_b^{r_b-1}-1} T_{r_b-1,i_u}^2,$$

where $T_{r_b-1,i_u}^2 = (2^q)^{u+1} 2^{NK} (\prod_{j \in J_b^{r_b-1}/\{j_{i_0}^{r_b-1},\ldots,j_{i_{u-1}}^{r_b-1}\}} \mathcal{N}_j^{r_b-1}(\eta_j^{r_b-1})\mathcal{N}_j^{r_b-1}(\eta_j'^{r_b-1}))2^{2s+2d_{in}+2d_{out}-2d_{pout}-2n} C'_{E,1}$ *and* $NK = NK_{r_b-1,0}^2 + | \cup_{p=0}^{u} K^2_{1,j_{i_u}^{r_b-1}}|$.

In the final r_f rounds, we employ the early abort technique to recover the remaining keys K_{rout}. Recall that, based on the key recovery graph of the last r_f rounds $\mathcal{G}_f$, each sink vertex represents a condition. According to the early abort technique, there are c_{rout}/q $(2q)$-bit conditions. Specifically, for each sink vertex v, let C_v^f be the set of source vertices that affect v, and K_v^f be the set of corresponding keys. Then, we can establish a relationship among C_v^f, K_v^f and the differences of v. Such a relationship is, in fact, a $(2q)$-bit condition. To obtain all c_{rout}/q $(2q)$-bit conditions, we can iterate through all sink vertices v. If not all keys in K_v have been guessed, then we set a condition and mark all keys in K_v as guessed. To determine the time complexity, we have the following lemma.

Lemma 7. *For the step 2d.*$(r_b + 1)$*, let* $NK_{r_b,0}^2$ *is the number of bits of the keys that has been guessed in the prior* r_b *steps already,* $v_0, \ldots, v_{c_{rout}/q-1}$ *are the sink vertexes that derive the* c_{rout}/q $(2q)$*-bit, and* $K_{v_i}^f$ *be* *the set of the associated keys of $v_i (0 \leq i \leq c_{rout}/q - 1)$. For a given key recovery order $i_0, \ldots, i_{c_{rout}/q-1}$, the time complexity of this step is*

$$T^2_{r_b+1} = \sum_{u=0}^{c_{rout}/q-1} T^2_{r_b+1,i_u},$$

where $T^2_{r_b+1,i_u} = \mathcal{Q}'(2^{-2q})^u 2^{NK}$, where $NK = NK^2_{r_b,0} + | \cup_{l=0}^{u} K^f_{v_{i_l}} |$.

Taking into account Lemma 4-Lemma 7, the overall time complexity of key recovery can be summarized as follows.

Theorem 3. *Let $j^{r_b-1}_{i_0}, \ldots, j^{r_b-1}_{i_{N_b^{r_b-1}-1}}$ be the key recovery order of step r_b and $i_0, \ldots, i_{c_{rout}/q-1}$ be the key recovery order of step $r_b + 1$. In the case of using the pre-sieve technique and the keys in K_{rin} are recovered first, the time cost of recovery keys is $T^2 = \sum_{i=1}^{r_b+1} T^2_i$, where the time complexity $T^2_i (1 \leq i \leq r_b + 1)$ and other notations are defined as Lemma 4-Lemma 7.*

Undoubtedly, we can also choose to recover the keys of the last r_f rounds first. In such case, we will have guessed $|K_{out}|$ bits of the keys, and the number of remaining quartets is $\mathcal{Q}'' = |\Omega^0_{rin}|^2 2^{2s+2d_{in}-2n}$, which is reduced by a factor of $2^{2d_{out}}$ for the first r_b rounds. All things considered, we obtain the following results.

Corollary 2. *Let $j^{r_b-1}_{i_0}, \ldots, j^{r_b-1}_{i_{N_b^{r_b-1}-1}}$ be the key recovery order of step r_b and $i_0, \ldots, i_{c_{rout}/q-1}$ be the key recovery order of step $r_b + 1$. In the case of using the pre-sieve technique and the keys in K_{rout} are recovered first, the time cost of recovery keys is $T^3 = \sum_{i=1}^{r_b+1} T^3_i$, where $T^3_{r_b+1} = \sum_{u=0}^{c_{rout}/q-1} T^3_{r_b+1,i_u}$ with $T^3_{r_b+1,i_u} = \mathcal{Q}(2^{-2q})^u 2^{|K^f_{v_{i_l}}|}$, and $T^3_i = (2^{|K_{out}|-2d_{out}}) T^2_i (1 \leq i \leq r_b)$.*

Certainly, when we don't use the pre-sieve technique, we will obtain $(c_{rin} + c_{rout})/q$ $(2q)$-bit conditions for $(c_{rin} + c_{rout})/q$ source vertices.

Corollary 3. *Let K^4_v be the set of corresponding keys that on the path from plaintexts or ciphertexts to the vertices v, and $i_0, \ldots, i_{(c_{rin}+c_{rout})/q-1}$ be the key recovery order. Then the time cost of recovery keys is*

$$T^4 = \sum_{u=0}^{(c_{rin}+c_{rout})/q-1} T^4_{i_u},$$

where $T^4_{i_u} = \mathcal{Q}(2^{-2q})^u 2^{|\cup_{l=0}^{u} K^4_{v_{i_l}}|}$.

Based on the previous discussion, we have furnished an accurate depiction of the complexity of the entire attack. This enables us to automatically search for the optimal key recovery strategy. It should be noted that the time complexity of key recovery is one component of the overall time complexity. Thus, if for a particular given key recovery order, the time complexity of key recovery does not surpass the time complexity of other steps, then the overall complexity is not strongly correlated with this specific item. Consequently, it can also be inferred that this key recovery order exhibits a globally optimal time complexity. Therefore, we can employ the *greedy algorithm* to recover the keys. That is, in each step, we select the keys that demand the fewest guesses for recovery. If the final time complexity of key recovery indeed does not exceed the time complexity of other steps, then the key recovery order is considered optimal.

When the final time complexity of key recovery exceed the time complexity of other steps, we need to utilize optimization methods to search for the optimal key recovery strategy.

4 The Full-round Related-key Impossible Boomerang Attack on ARADI

In this section, we first propose the 11-round RK-IBDs. Then, we add 2 rounds before the distinguisher and 3 rounds after the distinguishers to launch the 16 rounds IBA.

4.1 Specification of the block cipher ARADI

The block cipher ARADI is a low-latency block cipher, which is based on Toffoli gates and has a 128-bit block size and a 256-bit key size [BGG+23]. The overall encryption function is defined as follows:

$$E = \tau_{k_{16}} \circ (\Lambda_{15}\pi\tau_{k_{15}}) \circ \cdots \circ (\Lambda_2\pi\tau_{k_2}) \circ (\Lambda_1\pi\tau_{k_1}) \circ (\Lambda_0\pi\tau_{k_0}).$$

Where,

- π: π is the S-box layer, it uses 32 identical 4-bit S-boxes in parallel.
- $\Lambda_r (0 \leq r \leq 15)$: Λ_r is the r-th linear map, and the indices of the Λ_r are reduced modulo four.
- $\tau_{k_r} (0 \leq r \leq 15)$: τ_{k_r} is the key addition layer, the 128-bit round key k_r is XORed with the internal state.

One round of ARADI is shown in Figure

The internal state at r-th step of the key schedule of ARADI is represented by an array of eight 32-bit words $(K_0^r, K_1^r, \ldots, K_7^r)$, and

$$k_r = \begin{cases} K_0^r \| K_1^r \| K_2^r \| K_3^r, \ r \bmod 2 = 0, \\ K_4^r \| K_5^r \| K_6^r \| K_7^r, \ r \bmod 2 = 1. \end{cases}$$

In each step, $K_0^r \| K_1^r$ and $K_4^r \| K_5^r$ are processed through a 64-bit linear transformation M_0, while $K_2^r \| K_3^r$ and $K_6^r \| K_7^r$ undergo a 64-bit linear transformation M_1. This is followed by a word-level permutation $P_{r \bmod 2}$, where $P_0 = (1,2)(5,6)$ and $P_1 = (1,4)(3,6)$. The linear transformations M_0 and M_1 operate on the 32-bit inputs (a, b) as follows:

$$M_0((a,b)) = \left(S_{32}^1(a) \oplus b, S_{32}^3(b) \oplus S_{32}^1(a) \oplus b\right),$$
$$M_1((a,b)) = \left(S_{32}^9(a) \oplus b, S_{32}^{28}(b) \oplus S_{32}^9(a) \oplus b\right),$$

where S_{32}^j donates the left circular shift j-bit on a 32-bit word.

4.2 The 11 rounds RK-IBDs of ARADI

To enable the addition of as many rounds as possible before and after the distinguisher, we expect the weights of the input and output differences of the distinguisher to be as small as possible. Therefore, by adopting the same concept in [BCL+24,?], we propose a more concise way to obtain the RK-IBDs.

- Define the flag of difference. Partition the difference into blocks according to the size of the S-box, for a given difference, the flag of the difference is defined as fs with $0 \leq fs \leq 2^q$, where $0 \leq fs \leq 2^q - 1$ represents the real values of the difference of this S-box and $fs = 2^q$ represents unknown differences.
- Define the propagation rule of the flag of difference through each operation.
  - S-box: Let fs_0 and fs_1 be the types of input and output differences. Then,

$$fs_1 = \begin{cases} 0, fs_0 = 0, \\ 2^q, fs_0 > 0. \end{cases}$$

  - Xor: Let fs_0 and fs_1 be the two types of two input differences of Xor, and fs_2 be the type of output differences. Then,

$$fs_2 = \begin{cases} fs_0 \oplus fs_1, fs_0 \leq 2^s - 1, fs_1 \leq 2^s - 1, \\ 2^s, \ \text{otherwise}. \end{cases}$$

  For other linear transformations, its propagation rules can be derived according to Xor, and it will not be elaborated here.

- Detect the RK-IBDs. For the input difference α, output difference β, the master key difference ΔK in the upper trail, and the master key difference ∇K in the lower trail, propagate these differences for r rounds in the forward and backward directions respectively using a Python or C program. Subsequently, detect the S-box where the flag of the input difference $fs_0 \leq 2^s - 1$ and the output difference $fs_0 \leq 2^s - 1$. It should be noted that, in this scenario, the flag of the difference is the difference value. Therefore, we can examine the BCT to ascertain whether $((\alpha, \alpha), (\beta, \beta))$ is an r-round RK-IBD under the key difference $(\Delta K, \nabla K)$ or not.

Although in [BCL+24] the difference is divided into four types, the types of any non-zero difference and any difference essentially play the same role. As a result, they can be combined into a single category. Moreover, the propagation rules of our method are identical to those of the original method. Consequently, our method is simply a simplified rendition of the original one. Its advantage lies in the fact that it does not depend on a third-party solver and can be implemented both quickly and concisely.

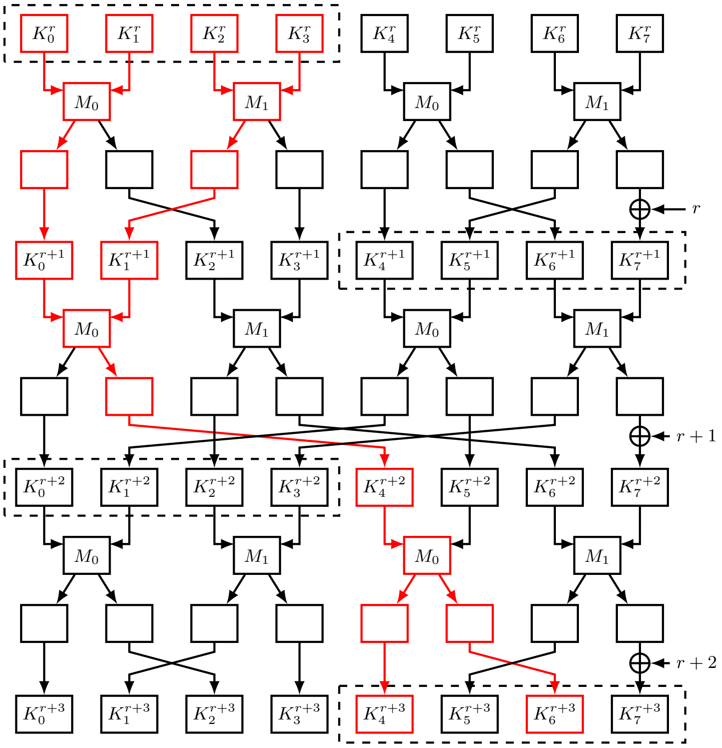

Fig. 4. One type of 3-round key difference of ARADI

We use our method to search for the RK-IBDs of ARADI. Firstly, it has been found that there exist 3-round probability-1 related-key differentials in ARADI.

Theorem 4. *For $\forall \kappa \in \mathbb{F}_2^{32}$, let Δk^r be the key difference at round r. Then $\Delta k^r = (\lambda_0, \lambda_1, \lambda_2, \lambda_3)$, $\Delta k^{r+1} = \Delta k^{r+2} = 0$, and $\Delta k^{r+3} = (\omega_0, 0, \omega_1, 0)$ for $r \mod 2 = 0$, where $(\chi_0, 0) = M_0(\lambda_0, \lambda_1)$, $(\chi_1, 0) = M_1(\lambda_2, \lambda_3)$ $(0, \kappa) = M_0(\chi_0, \chi_1)$, and $(\omega_0, \omega_1) = M_0(\kappa, 0)$. Meanwhile, the difference of the master key ΔK be got by calculating $(\lambda_0, \lambda_1, \lambda_2, \lambda_3, 0, 0, 0, 0)$ r rounds in the backward direction.*

Proof. As shown in Figure 4, let $K_4^{i+2} = \kappa$, then this theorem holds.

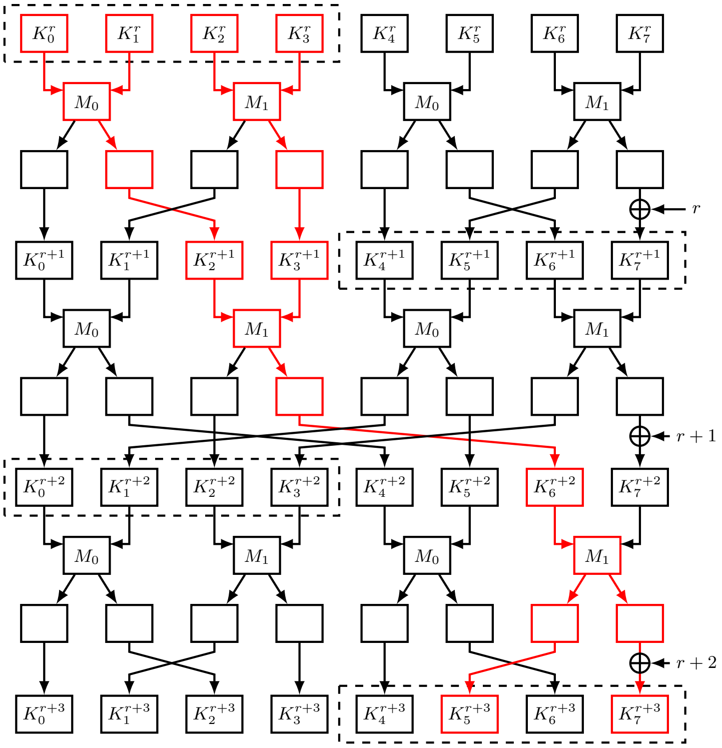

Fig. 5. One type of 3-round key difference of ARADI

Let $\varDelta x^r$ be the difference at round r before the operation key addition, and $\varDelta y^r$ be the difference at round r after the operation key addition, then $\varDelta x^r = \varDelta k^r \to 0 \to 0 \to \varDelta y^{r+2} = \varDelta k^{r+2}$ is an 3-round related-key differential with probability of 1 under the master key difference $\varDelta mk$.

Similarly, according to Figure 5, we can construct another type of 3-round related-key differential with probability of 1. We only detail the key differences here.

Theorem 5. *For $\forall \kappa \in \mathbb{F}_2^{32}$, let $\varDelta k^r$ be the key difference at round r. Then $\varDelta k^r = (\lambda_0, \lambda_1, \lambda_2, \lambda_3)$, $\varDelta k^{r+1} = \varDelta k^{r+2} = 0$, and $\varDelta k^{r+3} = (0, \omega_0, 0, \omega_1)$ for $r \mod 2 = 0$, where $(\chi_0, \chi_1) = M_1(0, \kappa)$, $(\omega_0, \omega_1) = M_1(\kappa, 0)$, and $(\lambda_0, \lambda_1) = M_0(0, \omega_0)$, $(\lambda_2, \lambda_3) = M_1(0, \omega_1)$. Meanwhile, the difference of the master key $\varDelta mk$ be got by calculating $(\lambda_0, \lambda_1, \lambda_2, \lambda_3, 0, 0, 0, 0)$ r rounds in the backward direction.*

With the 3-round related-key differential with probability of 1, we can construct the IBD as follows.
Construction. Let (α, α_{core}) be an 3-round related-key differential with probability of 1 under the master key difference $\varDelta K$ form round r_0 to $r_0 + 3$, and (β_{core}, β) be an 3-round related-key differential with probability of 1 under the master key difference ∇K form round $r_1 - 3$ to r_1 ($r_0 \mod 2 = 0, (r_1 - 3) \mod 2 = 0$), if $((\alpha_{core}, \alpha_{core}), (\beta_{core}, \beta_{core}))$ is an $(r_1 - r_0 - 6)$-round IBD under the master key difference $(\varDelta K, \nabla K)$, then $((\alpha, \alpha), (\beta, \beta))$ is an $(r_1 - r_0)$-round IBD from round r_0 to round r_1 under the master key difference $(\varDelta K, \nabla K)$. Note that, for either of the two construction methods of the 3-round related-key differential with probability of 1, the values of α, α_{core} and $\varDelta K$ is fully determined by a 32-bit value α_v, and the values of β, β_{core} and ∇K is fully determined by another 32-bit value β_v. Therefore, we we can search for IBD within given ranges of α_v and β_v.
Result. We set $r_0 = 2$ and $r_1 = 13$, with α_v and β_v each having only 1 bit activated. The overall search space amounts to $32 \times 32 \times 4 = 2^{12}$. By employing our search method, we obtain 97 11-round RK-IBDs within 258.06 seconds. One of these RK-IBDs is presented as follows and is utilized to implement our subsequent attacks.

Theorem 6. *Let α be a 32-dimensional tuple with $\alpha_0 = 8$, $\alpha_{15} = 1$, $\alpha_{16} = 2$, $\alpha_{23} = 4$ and 0 otherwise, and β be a 32-dimensional tuple with $\beta_0 = 10$ and 0 otherwise. Then $((\alpha, \alpha), (\beta, \beta))$ is an 11-round IBD from round 2 to round 13.*

Proof. We prove this theorem by contradiction. Let α_{core} be a 32-dimensional tuple with $\alpha_{core,0} = 10$ and 0 otherwise, and β_{core} be a 32-dimensional tuple with $\beta_{core,0} = 8$, $\beta_{core,15} = 1$, $\beta_{core,16} = 2$, $\beta_{core,23} = 4$ and 0 otherwise. As shown in Figure 6, (α, α_{core}) and (β_{core}, β) both are 3-round related-key differential with probability of 1. By propagating the α_{core} 3 rounds in forward direction, and β_{core} 2 rounds in backward direction. Then, in the 13-th S-box at round 8, it holds $BCT(4, 2) \neq 0$. However, according to the property of S-box, it holds $BCT(4, 2) = 0$. This is a contradiction.

4.3 Full-round Related-key Impossible Boomerang Attack on ARADI

We add 2 round before and 3 round after the 97 11-round RK-IBDs, and attempt to utilize our novel technique to identify a full-round attack of ARADI. That is, the sum of the data complexity, time complexity and memory complexity is less than 2^{256}. Finally, a full-round attack based on the distinguisher in Theorem 6 is detected. In this full-round attack, we pre-guess 12-bit key in K_{in} and 8-bit key in K_{out}, apply the pre-sieving technique in the first r_b rounds, and adopt the greedy algorithm to recover the keys. The overview of the full-round attack is shown in Figure 7 and Figure 8. In the last 3 rounds, we exchange the operation τ_{k_i} and $\Lambda_{i-1}(i = 14, 15, 16)$. Before we launch an attack, we introduce some notations.

Notation 1 *For determined differences $\Delta K^0_{01} = \Delta K^0_{23}$, $\Delta K^1_{01} = \Delta K^1_{23}$ and $\Delta Z^1_{01} = \Delta Z^1_{23}$, define*

$$\begin{aligned}
&\mathcal{S}_{Y_1} = \{\epsilon | \Delta Z^1_{01} \xrightarrow{\pi^{-1}} \epsilon\},\\
&\mathcal{S}_{X_1} = \{\theta | \theta = \epsilon \oplus \Delta K^1_{01}, \epsilon \in \mathcal{S}_{Y_1}\},\\
&\mathcal{S}_{Z_0} = \{\lambda | \lambda = \Lambda_0(\theta), \theta \in \mathcal{S}_{X_1}\},\\
&\mathcal{S}_{Y_0} = \{\sigma | \overline{\lambda} \xrightarrow{\pi^{-1}} \sigma,\ where\ \overline{\lambda}_i = 0, i \in I = \{7, 12, 13\}, \overline{\lambda}_j = \lambda_j, j \in \mathbb{Z}_{32}/I,\ and\ \lambda \in \mathcal{S}_{Z_0}\},\\
&\mathcal{S}_{X_0} = \{\omega | \omega = \sigma \oplus \Delta K^0_{01}, \sigma \in \mathcal{S}_{Y_0}\}.
\end{aligned}$$

Let $|\mathcal{S}|$ donate the size of the set $\mathcal{S}$, then

$$|\mathcal{S}_{Z_0}| = |\mathcal{S}_{X_1}| = |\mathcal{S}_{Y_1}| = \prod_{j \in J} \mathcal{N}^1_j(\Delta Z^1_{01,j}),$$

where $J = \{0, 9, 10, 15, 16, 17, 23, 26, 27\}$. And

$$|\mathcal{S}_{X_0}| = |\mathcal{S}_{Y_0}| = \sum_{\gamma \in \mathcal{S}_{Z_0}} \prod_{j \in J} \mathcal{N}^0_j(\gamma_j),$$

where $J = \{0, 2, 3, 4, 5, 8, 9, 10, 11, 15, 16, 17, 18, 21, 22, 23, 24, 26, 27, 28, 29, 30\}$. For the block cipher ARADI, and the fixed differences $\Delta K^0_{01}, \Delta K^1_{01}, \Delta Z^1_{01} = \Delta Z^1_{23}$, we obtain $|\mathcal{S}_{X_0}| \approx 2^{80.07}$ by exhaustive search.

Data Collection.

- **Data generation.** For all 2^n plaintexts P, we get the corresponding ciphertexts (C_0, C_1, C_2, C_3) under four keys $(K, K \oplus \Delta K, K \oplus \Delta K \oplus \nabla K, K \oplus \nabla K)$. Then, we guess $K^{16}_{0,7}$ and $K^{16}_{0,29}$, and partial decrypt the ciphertexts to obtain $W_i = \Lambda_{15}(C_i)(0 \leq i \leq 3)$, and

$$V_{i,j} = \begin{cases} S^{-1}(W_{i,j} \oplus K^{16}_{i,j}), j \in J = \{7, 29\}, \\ W_{i,j}, j \in Z_{32}/J, \end{cases} \qquad (0 \leq i \leq 3).$$

Finally, for each guessed key, we create four tables $T_i = \{(P, V_i) | P \in \mathbb{F}_2^{128}\}(0 \leq i \leq 3)$.

- **Pair generation.** For $j \in \mathbb{Z}_{32}$, let $IX^0_{0,j}$ and $IX^0_{3,j}$ traverse through all possible values.
  - For $j = 1, 6, 14, 19, 20, 25, 31$, we set $IX^0_{1,j} = IX^0_{0,j}$ and $IX^0_{2,j} = IX^0_{3,j}$.
  - For $j = 7, 12, 13$, we guess $IK^0_{0,j}$ and calculate

$$\begin{aligned} IX^0_{1,j} &= S^{-1}((S(IX^0_{0,j} \oplus IK^0_{0,j}) \oplus \Delta Z_{0,j})) \oplus IK^0_{0,j} \oplus \Delta K_{0,j}, \\ IX^0_{2,j} &= S^{-1}((S(IX^0_{3,j} \oplus IK^0_{0,j} \oplus \nabla K_{0,j}) \oplus \Delta Z_{0,j})) \oplus IK^0_{0,j} \oplus \Delta K_{0,j} \oplus \nabla K_{0,j}. \end{aligned}$$

  - For the remained position j and all differences $\omega, \omega' \in \mathcal{S}_{X_0}$, we set $IX^0_{1,j} = IX^0_{0,j} \oplus \omega_j$ and $IX^0_{2,j} = IX^0_{3,j} \oplus \omega'_j$.

  Finally, we guess 12-bit keys, and for each key, we get $2^n|\mathcal{S}_{X_0}| = 2^{208.07}$ pairs (X^0_0, X^0_1) and $2^n|\mathcal{S}_{X_0}| = 2^{208.07}$ pairs (X^0_2, X^0_3).
- **Quartets generation.** For each guessed keys $IK^{16}_{0,7}$, $IK^{16}_{0,29}$, $IK^0_{0,7}$, $IK^0_{0,12}$, and $IK^0_{0,13}$, we adopt the following method to construct quartets. For all $2^n|\mathcal{S}_{X_0}|$ pairs (IX^0_0, IX^0_1), we get the corresponding (V_0, V_1) by lookup table T_0 and T_1. Then, we insert those values to a hash table H_0 index by the j-th ($j \in J$) column of V_0 and the j-th ($j \in J$) column of V_1, where $J = \mathbb{Z}_{32}/\{0, 1, 4, 7, 10, 11, 21, 25, 28, 29, 31\}$. For each $2^n|\mathcal{S}_{X_0}|$ pairs (IX^0_2, IX^0_3), we get the corresponding (V_2, V_3) by lookup table T_2 and T_3, and lookup the hash table H_0 to find the corresponding $(IX^0_0, V_0, IX^0_1, V_1)$ such that

$$V_{0,j} \oplus V_{3,j} = V_{1,j} \oplus V_{2,j} = \begin{cases} 4, j = 7, \\ 2, j = 29, \\ 0, j \in Z_{32}/\{0, 1, 4, 7, 10, 11, 21, 25, 28, 29, 31\}. \end{cases}$$

  Finally, we get $Q = |\mathcal{S}_{X_0}|^2 \cdot 2^{2d_{rout}} = 2^{232.14}$ quartets, where $d_{rout} = 4 \times 9 = 36$.

Guess-and-Filter. For each 20-bit keys $IK^{16}_{0,7}$, $IK^{16}_{0,29}$, $IK^0_{0,7}$, $IK^0_{0,12}$, and $IK^0_{0,13}$, we make use of the Q quartets to eliminate wrong key bits, and then exhaust the remaining key bits to recover the full key.

1. **Guess the keys of the active difference in the first round.** For $\epsilon \in \mathcal{S}_{Y_1}$ and $\epsilon' \in \mathcal{S}_{Y_1}$, (ϵ, ϵ') correspond to $\prod_{j \in J^0} \mathcal{N}^0_j(\lambda_j)\mathcal{N}^0_j(\lambda'_j) \cdot 2^{2d_{rout}}$ quartets, where $\lambda = \Lambda_0(\epsilon \oplus \Delta K^1_{01})$, $\lambda' = \Lambda_0(\epsilon' \oplus \Delta K^1_{01})$ and $J^0 = \{0, 2, 3, 4, 5, 8, 9, 10, 11, 15, 16, 17, 18,$
   $21, 22, 23, 24, 26, 27, 28, 29, 30\}$. Let

$$\begin{aligned} J^{0,0} = J^0, J^{0,1} &= J^{0,0}/\{J^{0,0}_0\}, J^{0,2} = J^{0,1}/\{J^{0,1}_0\}, \\ &\cdots \\ J^{0,21} &= J^{0,20}/\{J^{0,20}_0\}, J^{0,22} = J^{0,21}/\{J^{0,21}_0\} = \emptyset. \end{aligned}$$

   Then, for $i \in \{1, 2, \ldots, 22\}$, we repeat the following steps to recover the keys.
   - $1.i(p = J^0_{i-1})$: Guess 2^4 possible values of $IK^0_{0,p}$, partially encrypt $(IX^0_{0,p}, IX^0_{1,p}, IX^0_{2,p}, IX^0_{3,p})$ one S-box, then use the known difference (λ_p, λ'_p) to filter the quartets. There are about $\prod_{j \in J^{0,i-1}} \mathcal{N}^0_j(\lambda_j)\mathcal{N}^0_j(\lambda'_j) \cdot 2^{2d_{rout}} \cdot (1/(\mathcal{N}^0_j(\lambda_j)\mathcal{N}^0_j(\lambda'_j))) = \prod_{j \in J^{0,i}} \mathcal{N}^0_j(\gamma_j)\mathcal{N}^0_j(\gamma'_j) \cdot 2^{2d_{rout}}$ remaining quartets. The time complexity of this step is

$$2^{20} \times \prod_{j \in J^{0,i-1}} \mathcal{N}^0_j(\lambda_j)\mathcal{N}^0_j(\lambda'_j) \cdot 2^{2d_{rout}} \times (2^4)^i \times 4/(32 \cdot 16) = 2^{13+2d_{rout}+4i} \prod_{j \in J^{0,i-1}} \mathcal{N}^0_j(\lambda_j)\mathcal{N}^0_j(\lambda'_j).$$

   Finally, for each possible (ϵ, ϵ') and $IK^0_{0,p}(p \in J^0)$, there remain $2^{2d_{rout}}$ quartets. Since we have $\prod_{j \in J^1}(\mathcal{N}^1_j(\Delta Z^1_{01,j}))^2$ pairs (ϵ, ϵ'), where $J^1 = \{0, 9, 10, 15, 16, 17, 23, 26, 27\}$. Thus, for each $IK^0_{0,p}(p \in J^0)$, there remain $2^{2d_{rout}} \prod_{j \in J^1}(\mathcal{N}^1_j(\Delta Z^1_{01,j}))^2$ quartets.

2. **Guess the remained keys in the first two rounds.** Let

$$\begin{aligned}
&J^{1,0}=\{0,9,10,15,16,17,23,26,27\}, P_0=\{0,5,24\},\\
&J^{1,1}=\{9,10,15,16,17,23,26,27\}, P_1=\{10,15,18\},\\
&J^{1,2}=\{9,15,16,17,23,26,27\}, P_2=\{4,15,23\},\\
&J^{1,3}=\{9,16,17,23,26,27\}, P_3=\{2,16,21\},\\
&J^{1,4}=\{9,17,23,26,27\}, P_4=\{3,17,22\},\\
&J^{1,5}=\{9,23,26,27\}, P_5=\{9,23,28\},\\
&J^{1,6}=\{9,26,27\}, P_6=\{13,16,27\},\\
&J^{1,7}=\{9,26\}.
\end{aligned}$$

Then, for $i \in \{1,2,\dots,7\}$ and $J^{1'}=\{0,10,15,16,17,23,27\}$, we repeat the following steps to recover the keys.

- 2.$i(q=J^{1'}_{i-1})$: Guess 2^4 possible values of $IK^1_{0,q}$, partially encrypt $(IX^1_{0,q}, IX^1_{1,q}, IX^1_{2,q}, IX^1_{3,q})$ one S-box, then use the known difference $(\Delta Z^1_{01,q}, \Delta Z^1_{01,q})$ to filter the quartets. There are about $2^{2d_{rout}}\prod_{j\in J^{1,i-1}}(\mathcal{N}^1_j(\Delta Z^1_{01,j}$ $(1/(\Delta Z^1_{01,j})^2) = 2^{2d_{rout}}\prod_{j\in J^{1,i}}(\mathcal{N}^1_j(\Delta Z^1_{01,j}))^2$ remaining quartets. The time complexity of this step is

$$2^{108}\times 2^{2d_{rout}}\prod_{j\in J^{1,i-1}}(\mathcal{N}^1_j(\Delta Z^1_{01,j}))^4\times(2^4)^i\times 2/(32\cdot 16) = 2^{100+2d_{rout}+4i}\prod_{j\in J^{1,i-1}}(\mathcal{N}^1_j(\Delta Z^1_{01,j}))^2.$$

After that, we adopt the following two steps to recover the keys.

- 2.8: Guess 2^8 possible values of $IK^0_{0,14}, IK^1_{0,9}$, partially encrypt $(IX^0_{0,14}, IX^0_{1,14}, IX^0_{2,14}, IX^0_{3,14})$ and $(IX^1_{0,p}, IX^1_{1,p}, IX^1_{2,p}, IX^1_{3,p})(p=9,17)$ one S-box, then use the known difference $(\Delta Z^1_{01,9}, \Delta Z^1_{01,9})$ to filter the quartets. There are about $2^{2d_{rout}}\prod_{j\in J^{1,7}}(\mathcal{N}^1_j(\Delta Z^1_{01,j}))^2\cdot(1/(\Delta Z^1_{01,9})^2) = 2^{2d_{rout}}(\mathcal{N}^1_9(\Delta Z^1_{01,26}))^2$ remaining quartets. The time complexity of this step is

$$2^{136}\times 2^{2d_{rout}}\prod_{j\in J^{1,7}}(\mathcal{N}^1_j(\Delta Z^1_{01,j}))^2\times 2^8\times 8/(32\cdot 16) = 2^{138+2d_{rout}}\prod_{j\in J^{1,7}}(\mathcal{N}^1_j(\Delta Z^1_{01,j}))^2.$$

- 2.9: Guess 2^8 possible values of $IK^0_{0,31}, IK^1_{0,26}$, partially encrypt $(IX^0_{0,31}, IX^0_{1,31}, IX^0_{2,31}, IX^0_{3,31})$ and $(IX^1_{0,p}, IX^1_{1,p}, IX^1_{2,p}, IX^1_{3,p})(p=12,26)$ one S-box, then use the known difference $(\Delta Z^1_{01,26}, \Delta Z^1_{01,26})$ to filter the quartets. There are about $2^{2d_{rout}}(\mathcal{N}^1_{26}(\Delta Z^1_{01,26}))^2\cdot(1/(\Delta Z^1_{01,26})^2) = 2^{2d_{rout}}$ remaining quartets. The time complexity of this step is

$$2^{144}\times 2^{2d_{rout}}(\mathcal{N}^1_{26}(\Delta Z^1_{01,26}))^2\times 2^8\times 8/(32\cdot 16) = 2^{146+2d_{rout}}(\mathcal{N}^1_{26}(\Delta Z^1_{01,26}))^2.$$

3. **Guess the keys in the last three rounds.**
- 3.1: Guess 2^8 possible values of $IK^{16'}_{0,0}, IK^{16'}_{0,28}$, partially encrypt $(IX^{16'}_{0,p}, IX^{16'}_{1,p}, IX^{16'}_{1,p}, IX^{16'}_{1,p})$ one S-box $(p=0,28)$, then use the known difference $(\nabla X^{15'}_{12,0}, \nabla X^{15'}_{03,0})$ to filter the quartets. There are about $2^{2d_{rout}-8}$ remaining quartets. The time complexity of this step is $2^{152}\times 2^{2d_{rout}}\times 2^8\times 8/(32\cdot 16) = 2^{154+2d_{rout}}$.
- 3.2: Guess 2^4 possible values of $IK^{16'}_{0,21}$, partially encrypt $(IX^{16'}_{0,21}, IX^{16'}_{1,21}, IX^{16'}_{1,21}, IX^{16'}_{1,21})$ one S-box, then use the known difference $(\nabla X^{15'}_{12,21}, \nabla X^{15'}_{03,21})$ to filter the quartets. There are about $2^{2d_{rout}-16}$ remaining quartets. The time complexity of this step is $2^{152}\times 2^{2d_{rout}-8}\times 2^8\times 2^4\times 4/(32\cdot 16) = 2^{149+2d_{rout}}$.
- 3.3: Guess 2^8 possible values of $IK^{16'}_{0,4}$ and $IK^{16'}_{0,11}$, partially encrypt $(IX^{16'}_{0,p}, IX^{16'}_{1,p}, IX^{16'}_{1,p}, IX^{16'}_{1,p})$ one S-box $(p=4,11)$, then use the known difference $(\nabla X^{15'}_{12,4}, \nabla X^{15'}_{03,4})$ to filter the quartets. There are about $2^{2d_{rout}-24}$ remaining quartets. The time complexity of this step is $2^{152}\times 2^{2d_{rout}-16}\times 2^{12}\times 2^8\times 8/(32\cdot 16) = 2^{150+2d_{rout}}$.

- 3.4: Guess 2^4 possible values of $IK^{16'}_{0,25}$, partially encrypt $(IX^{16'}_{0,p}, IX^{16'}_{1,p}, IX^{16'}_{1,p}, IX^{16'}_{1,p})$ one S-box ($p = 25$), then use the known difference $(\nabla X^{15'}_{12,25}, \nabla X^{15'}_{03,25})$ to filter the quartets. There are about $2^{2d_{rout}-32}$ remaining quartets. The time complexity of this step is $2^{152} \times 2^{2d_{rout}-24} \times 2^{20} \times 2^4 \times 4/(32 \cdot 16) = 2^{145+2d_{rout}}$.
- 3.5: Guess 2^8 possible values of $IK^{16'}_{0,1}$ and $IK^{16'}_{0,10}$, partially encrypt $(IX^{16'}_{0,p}, IX^{16'}_{1,p}, IX^{16'}_{1,p}, IX^{16'}_{1,p})$ one S-box ($p = 1, 0$), then use the known difference $(\nabla X^{15'}_{12,10}, \nabla X^{15'}_{03,10})$ to filter the quartets. There are about $2^{2d_{rout}-40}$ remaining quartets. The time complexity of this step is $2^{152} \times 2^{2d_{rout}-32} \times 2^{24} \times 2^8 \times 8/(32 \cdot 16) = 2^{146+2d_{rout}}$.
- 3.6: Guess 2^4 possible values of $IK^{16'}_{0,31}$, partially encrypt $(IX^{16'}_{0,p}, IX^{16'}_{1,p}, IX^{16'}_{1,p}, IX^{16'}_{1,p})$ one S-box ($p = 31$), then use the known difference $(\nabla X^{15'}_{12,31}, \nabla X^{15'}_{03,31})$ to filter the quartets. There are about $2^{2d_{rout}-48}$ remaining quartets. The time complexity of this step is $2^{152} \times 2^{2d_{rout}-40} \times 2^{32} \times 2^4 \times 4/(32 \cdot 16) = 2^{141+2d_{rout}}$.
- 3.7: Guess 2^{16+16} possible values of $IK^{16'}_{0,p}(p = 8, 14, 19, 24)$ and $IK^{15'}_{0,p}(p = 1, 7, 24, 28)$, partially encrypt $(IX^{16'}_{0,p}, IX^{16'}_{1,p}, IX^{16'}_{1,p}, IX^{16'}_{1,p})$ one S-box ($p = 8, 14, 19, 24$), and $(IX^{15'}_{0,p}, IX^{15'}_{1,p}, IX^{15'}_{1,p}, IX^{15'}_{1,p})$ one S-box ($p = 1, 7, 24, 28$), then use the known difference $(\nabla X^{14'}_{12,28}, \nabla X^{14'}_{03,28})$ to filter the quartets. There are about $2^{2d_{rout}-56}$ remaining quartets. The time complexity of this step is $2^{152} \times 2^{2d_{rout}-48} \times 2^{36} \times 2^{32} \times 32/(32 \cdot 16) = 2^{168+2d_{rout}}$.
- 3.8: Guess 2^{8+4} possible values of $IK^{16'}_{0,p}(p = 2, 23)$ and $IK^{15'}_{0,p}(p = 11)$, partially encrypt $(IX^{16'}_{0,p}, IX^{16'}_{1,p}, IX^{16'}_{1,p}, IX^{16'}_{1,p})$ one S-box ($p = 2, 23$), and $(IX^{15'}_{0,p}, IX^{15'}_{1,p}, IX^{15'}_{1,p}, IX^{15'}_{1,p})$ one S-box ($p = 11$), then use the known difference $(\nabla X^{14'}_{12,11}, \nabla X^{14'}_{03,11})$ to filter the quartets. There are about $2^{2d_{rout}-64}$ remaining quartets. The time complexity of this step is $2^{152} \times 2^{2d_{rout}-56} \times 2^{68} \times 2^{12} \times 12/(32 \cdot 16) \approx 2^{171+2d_{rout}}$.
- 3.9: Guess 2^4 possible values of $IK^{14'}_{0,p}(p = 1)$, partially encrypt $(IX^{14'}_{0,p}, IX^{14'}_{1,p}, IX^{14'}_{2,p}, IX^{14'}_{3,p})$ one S-box ($p = 1$), then use the known difference $(\nabla Y^{13}_{12,1}, \nabla Y^{13}_{03,1})$ to filter the quartets. There are about $2^{2d_{rout}-72}$ remaining quartets. The time complexity of this step is $2^{152} \times 2^{2d_{rout}-64} \times 2^{80} \times 2^4 \times 4/(32 \cdot 16) = 2^{165+2d_{rout}}$.

After above process, there remains $2^{256}/e \approx 2^{254.56}$ keys, we can exhaustively search for these remaining keys.
Complexity. This attack use the full-codebook, thus the data complexity is 2^{130}.

- Cost of data generation: we guess 2^8 keys for 2^{128} under four keys and store those plaintexts-ciphertexts. The time complexity is $2^{128} \times 2^8 \times 4 \times 8/(32 \cdot 16) = 2^{134}$, and the memory complexity is $2^{128} \times 2^8 \times 4 \times 2 = 2^{139}$.
- Cost of pair generation: we guess 2^{12} keys for 2^{128}, then we lookup table to construct the pairs store them. The time complexity is $2^{128} \times 2^{12} \times 12/(32 \cdot 16) + 2^{12} \times 2 \times 2^n |\mathcal{S}_{X_0}| \approx 2^{221.07}$, and the memory complexity is $2^{12} \times 2 \times 2^n |\mathcal{S}_{X_0}| = 2^{221.07}$.
- Cost of pair generation: for 2^{20} guessed keys and $2^n |\mathcal{S}_{X_0}|$ pairs (X^0_2, X^0_3), we lookup the $2^{2d_{rout}-n}|\mathcal{S}_{X_0}|$ pairs (X^0_0, X^0_1). The time complexity and memory complexity are both $2^{20} \times |\mathcal{S}_{X_0}|^2 \cdot 2^{2d_{rout}} = 2^{252.14}$.
- Cost of Step 1 in Guess-and-Filter: Since $\mathcal{N}^0_j(\gamma_j) > 4$, the time complexity of $1.i$ decreases successively as the value of i increases. Thus, the time complexity and memory complexity are both $2^{13+2d_{rout}} \times (\sum_{\lambda \in \mathcal{S}_{Z_0}} \sum_{\lambda' \in \mathcal{S}_{Z_0}} \prod_{j \in J^{0,0}} \mathcal{N}^0_j(\lambda_j) \mathcal{N}^0_j(\lambda'_j)) = 2^{13+2d_{rout}} |\mathcal{S}_{X_0}|^2 = 2^{245.14}$.
- Cost of Step 2 in Guess-and-Filter: The time complexity of $2.i$ decreases successively as the value of $i(\in \{1, 2, \ldots, 7\})$ increases. Thus, the time complexity and memory complexity are both $2^{100+2d_{rout}} \prod_{j \in J^{1,0}} (\mathcal{N}^1_j(\varDelta Z^1_{01,j}))^2 = 2^{218.2}$. For step 2.8, the time complexity and memory complexity are both $2^{138+2d_{rout}} \prod_{j \in J^{1,7}} (\mathcal{N}^1_j(\varDelta Z^1_{01,j}))^2 = 2^{220.34}$. For step 2.9, the time complexity and memory complexity are both $2^{146+2d_{rout}} (\mathcal{N}^1_{26}(\varDelta Z^1_{01,26}))^2 = 2^{223.18}$.
- Cost of Step 3 in Guess-and-Filter: The time complexity and memory complexity are both $2^{171+2d_{rout}} = 2^{243}$.
- Cost of exhaustively search: the time complexity is $2^{254.56}$

All in all, the data complexity is 2^{130}, the time complexity is $2^{254.56} + 2^{252.14} = 2^{254.81}$, and the memory complexity is $2^{252.14} + 2^{236} = 2^{252.14}$. Since $2^{128} + 2^{254.81} + 2^{252.14} = 2^{255.02} < 2^{256}$, we can recovery the key within the complexity is less than exhaustive keys.

5 Conclusion

In this paper, we propose the pre-sieving technique, partial pre-guess key technique and precise complexity evaluation technique to improve the key recovery of impossible boomerang attack. As a result, we apply those techniques to the block cipher ARADI, and propose the first full-round attack on ARADI.

References

[BCL+24] Xavier Bonnetain, Margarita Cordero, Virginie Lallemand, Marine Minier, and María Naya-Plasencia. On impossible boomerang attacks application to simon and skinnyee. *IACR Trans. Symmetric Cryptol.*, 2024(2):222–253, 2024.

[BGG+23] Emanuele Bellini, David Gérault, Juan Grados, Yun Ju Huang, Rusydi H. Makarim, Mohamed Rachidi, and Sharwan K. Tiwari. CLAASP: A cryptographic library for the automated analysis of symmetric primitives. In Claude Carlet, Kalikinkar Mandal, and Vincent Rijmen, editors, *Selected Areas in Cryptography - SAC 2023 - 30th International Conference, Fredericton, Canada, August 14-18, 2023, Revised Selected Papers*, volume 14201 of *Lecture Notes in Computer Science*, pages 387–408. Springer, 2023.

[BHL+20] Hamid Boukerrou, Paul Huynh, Virginie Lallemand, Bimal Mandal, and Marine Minier. On the feistel counterpart of the boomerang connectivity table introduction and analysis of the FBCT. *IACR Trans. Symmetric Cryptol.*, 2020(1):331–362, 2020.

[BJK+16] Christof Beierle, Jérémy Jean, Stefan Kölbl, Gregor Leander, Amir Moradi, Thomas Peyrin, Yu Sasaki, Pascal Sasdrich, and Siang Meng Sim. The SKINNY family of block ciphers and its low-latency variant MANTIS. In Matthew Robshaw and Jonathan Katz, editors, *Advances in Cryptology - CRYPTO 2016 - 36th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 14-18, 2016, Proceedings, Part II*, volume 9815 of *Lecture Notes in Computer Science*, pages 123–153. Springer, 2016.

[BL23] Xavier Bonnetain and Virginie Lallemand. On boomerang attacks on quadratic feistel ciphers new results on KATAN and simon. *IACR Trans. Symmetric Cryptol.*, 2023(3):101–145, 2023.

[CHP+18] Carlos Cid, Tao Huang, Thomas Peyrin, Yu Sasaki, and Ling Song. Boomerang connectivity table: A new cryptanalysis tool. In Jesper Buus Nielsen and Vincent Rijmen, editors, *Advances in Cryptology - EUROCRYPT 2018 - 37th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Tel Aviv, Israel, April 29 - May 3, 2018 Proceedings, Part II*, volume 10821 of *Lecture Notes in Computer Science*, pages 683–714. Springer, 2018.

[DDV20] Stéphanie Delaune, Patrick Derbez, and Mathieu Vavrille. Catching the fastest boomerangs application to SKINNY. *IACR Trans. Symmetric Cryptol.*, 2020(4):104–129, 2020.

[DKS10] Orr Dunkelman, Nathan Keller, and Adi Shamir. A practical-time related-key attack on the KASUMI cryptosystem used in GSM and 3g telephony. In Tal Rabin, editor, *Advances in Cryptology - CRYPTO 2010, 30th Annual Cryptology Conference, Santa Barbara, CA, USA, August 15-19, 2010. Proceedings*, volume 6223 of *Lecture Notes in Computer Science*, pages 393–410. Springer, 2010.

[DKS14] Orr Dunkelman, Nathan Keller, and Adi Shamir. A practical-time related-key attack on the KASUMI cryptosystem used in GSM and 3g telephony. *J. Cryptol.*, 27(4):824–849, 2014.

[DR02] Joan Daemen and Vincent Rijmen. *The Design of Rijndael: AES - The Advanced Encryption Standard.* Information Security and Cryptography. Springer, 2002.

[HSE23] Hosein Hadipour, Sadegh Sadeghi, and Maria Eichlseder. Finding the impossible: Automated search for full impossible-differential, zero-correlation, and integral attacks. In Carmit Hazay and Martijn Stam, editors, *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*, volume 14007 of *Lecture Notes in Computer Science*, pages 128–157. Springer, 2023.

[LKKD08a] Jiqiang Lu, Jongsung Kim, Nathan Keller, and Orr Dunkelman. Improving the efficiency of impossible differential cryptanalysis of reduced camellia and MISTY1. In Tal Malkin, editor, *Topics in Cryptology - CT-RSA 2008, The Cryptographers' Track at the RSA Conference 2008, San Francisco, CA, USA, April 8-11, 2008. Proceedings*, volume 4964 of *Lecture Notes in Computer Science*, pages 370–386. Springer, 2008.

[LKKD08b] Jiqiang Lu, Jongsung Kim, Nathan Keller, and Orr Dunkelman. Improving the efficiency of impossible differential cryptanalysis of reduced camellia and misty1. In Tal Malkin, editor, *Topics in Cryptology – CT-RSA 2008*, pages 370–386, Berlin, Heidelberg, 2008. Springer Berlin Heidelberg.

[Lu] Jiqiang Lu. Cryptanalysis of block ciphers. *mat.uniroma3.it.*

[Lu11] Jiqiang Lu. The (related-key) impossible boomerang attack and its application to the AES block cipher. *Des. Codes Cryptogr.*, 60(2):123–143, 2011.

[Mur11] Sean Murphy. The return of the cryptographic boomerang. *IEEE Trans. Inf. Theory*, 57(4):2517–2521, 2011.

[NSS22] Yusuke Naito, Yu Sasaki, and Takeshi Sugawara. Secret can be public: Low-memory AEAD mode for high-order masking. In Yevgeniy Dodis and Thomas Shrimpton, editors, *Advances in Cryptology - CRYPTO 2022 - 42nd Annual International Cryptology Conference, CRYPTO 2022, Santa Barbara, CA, USA, August 15-18, 2022, Proceedings, Part III*, volume 13509 of *Lecture Notes in Computer Science*, pages 315–345. Springer, 2022.

[WP19] Haoyang Wang and Thomas Peyrin. Boomerang switch in multiple rounds. application to AES variants and deoxys. *IACR Trans. Symmetric Cryptol.*, 2019(1):142–169, 2019.

[WWS23] Dachao Wang, Baocang Wang, and Siwei Sun. Sat-aided automatic search of boomerang distinguishers for ARX ciphers. *IACR Trans. Symmetric Cryptol.*, 2023(1):152–191, 2023.

[ZWT24] Jianing Zhang, Haoyang Wang, and Deng Tang. Impossible boomerang attacks revisited applications to deoxys-bc, joltik-bc and SKINNY. *IACR Trans. Symmetric Cryptol.*, 2024(2):254–295, 2024.

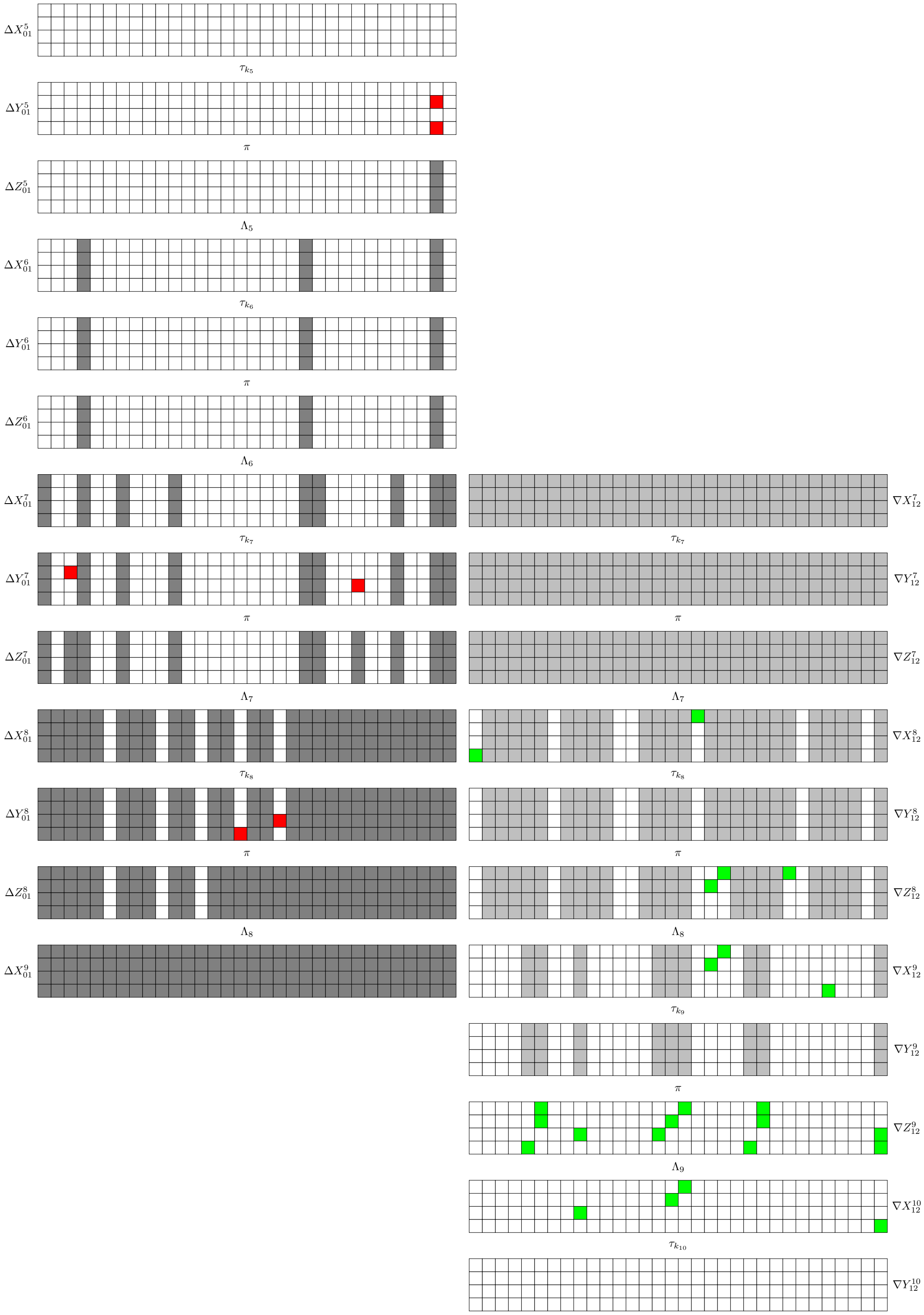

Fig. 6. The core of 11-round IBD.

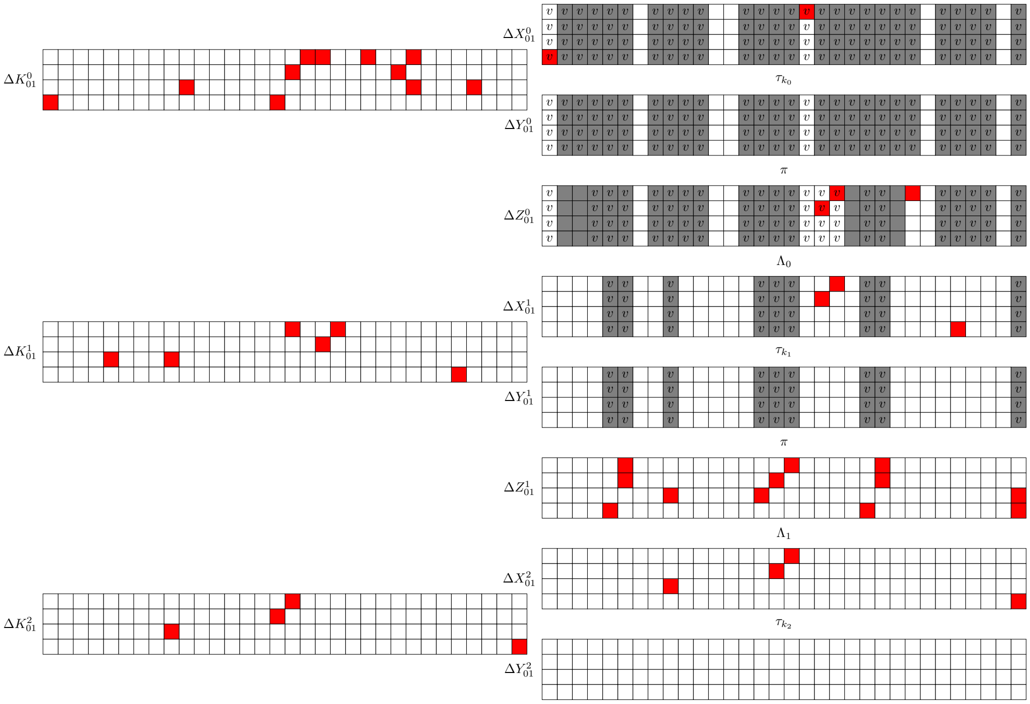

Fig. 7. Top 2 rounds added for key recovery in full-round attack on ARADI.

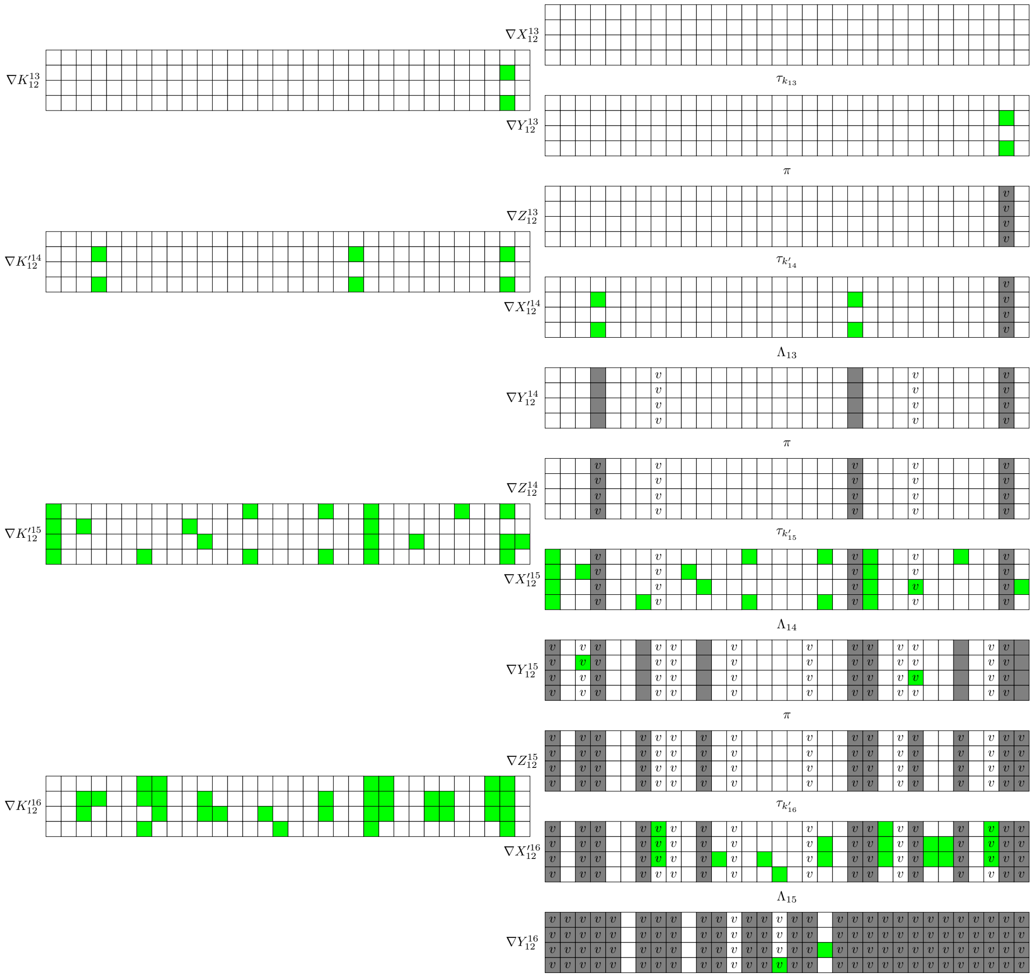

Fig. 8. Bottom 3 rounds added for key recovery in full-round attack on ARADI.