Biclique Attack of the Full ARIA-256

Shao-zhen Chen
Tian-min Xu
Zhengzhou Information Science and Technology Institute
Zhengzhou 450002, China

January 8, 2012

Abstract

In this paper, combining the biclique cryptanalysis with the MITM attack, we present the first key recovery method for the full ARIA-256 faster than brute-force. The attack requires 2^{80} chosen plaintexts, and the time complexity is about $2^{255.2}$ full-round ARIA encryptions in the processing phase.

keywords: Block Cipher; ARIA-256; Biclique structure

1 Introduction

The ARIA block cipher [1]was designed by a group of Korean experts in 2003, and was standardized as the Korean Standard in 2004. ARIA supports key length of 128,192 and 256 bits, these versions of ARIA are denoted as ARIA-128, ARIA-192 and ARIA-256. The number of rounds for these three versions are 12, 14 and 16, respectively. In ref. [1], the designers of ARIA presented some cryptanalysis including both differential cryptanalysis, linear cryptanalysis, and some other known attacks . Later Biryukov et al. performed an evaluation of ARIA [2], however, they especially focused on truncated differential cryptanalysis and dedicated linear cryptanalysis. In ref. [3], Wu et al. firstly found some non-trivial 4-round impossible differentials which led to a 6-round attack on ARIA. The security of ARIA against boomerang attack was presented by Fleischmann et al. in [4]. Li et al. firstly found some 3-round integral distinguishers by counting methods, which also led up to a 6-round integral attack on ARIA-192 [5], recently, Li et al. [6] presented the meet-in-the-middle(MITM) attacks on the round-reduced ARIA, which led to a 8-round attack on ARIA-256. For ARIA-256, the number of cryptanalyzed round did not increase since then.

ARIA with its wide-trail strategy was designed to withstand differential and linear cryptanalysis. ARIA is an involution SPN cipher and has involution substitution and diffusion layers by using operations such as multiplications in a finite field. In the diffusion layer of ARIA, a 16×16 binary matrix of the maximum branch number 8 is used to avoid some attacks well applied to the reduced round of Rijndael.

The basic concept of the MITM attack was proposed by Diffie and Hellman [7]. So far, this attack has been applied to several block ciphers. Furthermore, over the past few years, this attack has been improved in a line of preimage attacks on hash functions and several novel techniques are introduced.

Biclique cryptanalysis was first introduced for hash cryptanalysis [8]. Andrey Bogdanov* et al. carried over the concept of the independent bicilique to AES cryptanalysis [9], and constructed a 3 round biclique. Combining the biclique with the MITM attack, they presented the first key recovery method for the full AES-128 faster than brute-force. To construct the independent biclique, we need two independent related-key differentials. When it comes to the ciphers whose diffusion operation is fast, it is difficult to construct the biclique.

Our Contribution:

In this paper, we combine the Biclique cryptanalysis with the MITM attack, and present the first key recovery method for the full ARIA-256 faster than brute-force.

1. Four 128-bit values W_0, W_1, W_2, W_3 are generated from the master key MK by using a 3-round 256-bit Feistel cipher. We select those types of differentials of MK such that the difference of W_1 is 0. Then we can construct a 2-round biclique for ARIA-256. However, those types of differentials of MK in other versions of ARIA are not exist.

2. Because the position of active-byte of the master key will affect the number of active-byte of the round key, we found a type of keys such that the round key have the least number of active-bytes in round 3. Hence, the computation complexity can be decreased as well as possible.

3. The attack on ARIA-256 consists of two parts. Firstly, we construct a 2-round biclique of dimension 8 for ARIA-256. Then, we attack the remaining 14 rounds by using MITM method. By matching only on a single byte, the computation complexity can be reduced. Table 1 gives the comparison between our result and earlier best result.

Table 1 Summary of Attacks on ARIA-256

Rounds	Data	Complexity	Type of attack	Paper
8	2^{56}	$2^{251.6}$	MITM Attack	[6]
16	2^{80}	$2^{255.2}$	Biclique Key Recovery Attack	This paper

This paper is organized as follows. We introduce the biclique key recovery attack in Section 2 and give a brief description of ARIA in Section 3. In Section 4, we present a biclique key recovery attack for the full round ARIA-256. Finally, section 5 summarizes this paper.

2 Biclique Key Recovery Attack

The main idea behind the biclique key recovery attack is based on the biclique structure which can be used to decrease the complexity of the MITM attack. It is apparent to see that the construction of biclique is an important step. The following sections will describe the steps of the biclique key recovery attack.

2.1 Biclique Structure

Now we introduce the notation of a biclique. Let f be a subcipher that maps a plaintext P to the internal state S: $f_K(P) = S$.

The 3-tuple $[\{P_i\}, \{S_j\}, \{K[i,j]\}]$ is called a *d-dimensional* Biclique, if

$$S_j = f_{K[i,j]}(P_i)$$

2.2 Biclique from Independent Related-Key Differentials

Definition 1.A byte is called an active-byte, if the input differential of it is non-zero. Otherwise, it is called a non-active-byte.

Definition 2.If the trail of a differential do not share active nonlinear components (such as active S-boxes in ARIA) with the trail of another differential, we call them independent related-key differentials.

Let the key $K[0,0]$ map plaintext P_0 to the internal state S_0, and consider two sets of 2^d related-key differentials each over f with respect to the base computation $P_0 \xrightarrow[f]{K[0,0]} S_0$.

- $\bigtriangledown_i$-differentials. A differential in the first set maps input difference $\bigtriangledown_i$ to an output difference 0 under key difference $\bigtriangledown_i^K$:

$$\bigtriangledown_i \xrightarrow[f]{\bigtriangledown_i^K} 0, \text{ with } \bigtriangledown_0^K = 0,\ \bigtriangledown_0 = 0$$

- $\triangle_j$-differentials. A differential in the second set maps input difference 0 to an output difference $\triangle_j$ under key difference $\triangle_j^K$:

$$0 \xrightarrow[f]{\triangle_j^K} \triangle_j, \text{ with } \triangle_0^K = 0,\ \triangle_0 = 0$$

If $\bigtriangledown_i$-differentials and $\triangle_j$-differentials are independent, then we can construct 2^{2d} combined differentials:

$$\bigtriangledown_i \xrightarrow[f]{\bigtriangledown_i^K \oplus \triangle_j^K} \triangle_j, \text{ for all } i,\ j \in \{0, \dots, 2^d - 1\}$$

Substituting P_0, S_0, and $K[0,0]$ to the combined differentials, one obtains:

$$P_0 \oplus \bigtriangledown_i \xrightarrow[f]{K[0,0] \oplus \bigtriangledown_i^K \oplus \triangle_j^K} S_0 \oplus \triangle_j$$

Finally, we put:

$$P_i = P_0 \oplus \bigtriangledown_i,$$
$$S_j = S_0 \oplus \triangle_j,$$
$$K[i,j] = K[0,0] \oplus \bigtriangledown_i^K \oplus \triangle_j^K$$

and get exactly the definition of a *d-dimensional* biclique.

2.3 Biclique Key Recovery Attack

Based on the biclique from independent related-key differentials, we describe the biclique key recovery attack.

For each set of keys the adversary builds a structure of 2^d plaintexts P_i and 2^d intermediate states S_j with respect to the set of keys $K[i,j]$ which satisfies the following condition:

$$S_j = f_{K[i,j]}(P_i)$$

The adversary asks the oracle to encrypt plaintexts P_i under the secret key K_{secret}, and obtains the 2^d ciphertexts C_i:

$$P_i \xrightarrow[e]{oracle} C_i$$

If the texted key $K[i,j]$ satisfies the following condition:

$$S_j \xrightarrow[e_1]{K[i,j]} \overrightarrow{v} = \overleftarrow{v} \xleftarrow[e_2]{K[i,j]} C_i \tag{1}$$

We consider it is a key candidate.

2.4 Matching with Precomputations

Here we describe the idea of matching with precomputations. This can be seen as an efficient way of checking equation (1).

Firstly, the adversary computes and stores in memory 2^{d+1} full computations

$$\text{for all } j \; S_j \xrightarrow{K[0,j]} \overrightarrow{v} \text{ and for all } i \; \overleftarrow{v} \xleftarrow{K[i,0]} C_i$$

Then for particular i, j he recomputes only those parts of state that differ from the stored ones. Hence, he can decrease the computational complexity.

3 Description of ARIA-256

The ARIA-256 block cipher is an involution SPN cipher and have involution substitution and diffusion layers. The diffusion layer uses a 16×16 binary matrix in $GF(2^8)$. The substitution layer consists of sixteen 8×8 S-boxes. The 128 bits state can be denoted as 4×4 matrix in $GF(2^8)$ as shown in Fig. 1.

Each round of the cipher consists of three parts, Round key addition(AK), Substitution layer(SL), Diffusion layer(DL). Note that the diffusion layer of the last round is replaced by a round key addition.

x_0	x_4	x_8	x_{12}
x_1	x_5	x_9	x_{13}
x_2	x_6	x_{10}	x_{14}
x_3	x_7	x_{11}	x_{15}

Figure 1: The state of ARIA

S_1	S_2	S_1^{-1}	S_2^{-1}	S_1	S_2	S_1^{-1}	S_2^{-1}	S_1	S_2	S_1^{-1}	S_2^{-1}	S_1	S_2	S_1^{-1}	S_2^{-1}

S-box layer type 1

S_1^{-1}	S_2^{-1}	S_1	S_2	S_1^{-1}	S_2^{-1}	S_1	S_2	S_1^{-1}	S_2^{-1}	S_1	S_2	S_1^{-1}	S_2^{-1}	S_1	S_2

S-box layer type 2

Figure 2: Two types of S-box layer

Substitution layer: nolinear substitution, the Substitution layer consists of two types of S-box layers. Type 1 is used in the odd rounds and type 2 is used in the even rounds as shown in Fig. 2.

Diffusion layer: P: $GF(2^8)^{16} \to GF(2^8)^{16}$ is given by

$$(x_0, x_1, \ldots, x_{15}) \to (y_0, y_1, \ldots, y_{15})$$

where

$$y_0 = x_3 \oplus x_4 \oplus x_6 \oplus x_8 \oplus x_9 \oplus x_{13} \oplus x_{14}, y_8 = x_0 \oplus x_1 \oplus x_4 \oplus x_7 \oplus x_{10} \oplus x_{13} \oplus x_{15},$$

$$y_1 = x_2 \oplus x_5 \oplus x_7 \oplus x_8 \oplus x_9 \oplus x_{12} \oplus x_{15}, y_9 = x_0 \oplus x_1 \oplus x_5 \oplus x_6 \oplus x_{11} \oplus x_{12} \oplus x_{14},$$

$$y_2 = x_1 \oplus x_4 \oplus x_6 \oplus x_{10} \oplus x_{11} \oplus x_{12} \oplus x_{15}, y_{10} = x_2 \oplus x_3 \oplus x_5 \oplus x_6 \oplus x_8 \oplus x_{13} \oplus x_{15},$$

$$y_3 = x_0 \oplus x_5 \oplus x_7 \oplus x_{10} \oplus x_{11} \oplus x_{13} \oplus x_{14}, y_{11} = x_2 \oplus x_3 \oplus x_4 \oplus x_7 \oplus x_9 \oplus x_{12} \oplus x_{14},$$

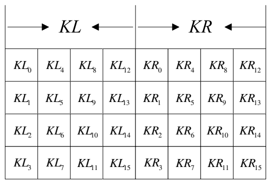
$$y_4 = x_0 \oplus x_2 \oplus x_5 \oplus x_8 \oplus x_{11} \oplus x_{14} \oplus x_{15}, y_{12} = x_1 \oplus x_2 \oplus x_6 \oplus x_7 \oplus x_9 \oplus x_{11} \oplus x_{12},$$

$$y_5 = x_1 \oplus x_3 \oplus x_4 \oplus x_9 \oplus x_{10} \oplus x_{14} \oplus x_{15}, y_{13} = x_0 \oplus x_3 \oplus x_6 \oplus x_7 \oplus x_8 \oplus x_{10} \oplus x_{13},$$

$$y_6 = x_0 \oplus x_2 \oplus x_7 \oplus x_9 \oplus x_{10} \oplus x_{12} \oplus x_{13}, y_{14} = x_0 \oplus x_3 \oplus x_4 \oplus x_5 \oplus x_9 \oplus x_{11} \oplus x_{14},$$

$$y_7 = x_1 \oplus x_3 \oplus x_6 \oplus x_8 \oplus x_{11} \oplus x_{12} \oplus x_{13}, y_{15} = x_1 \oplus x_2 \oplus x_4 \oplus x_5 \oplus x_8 \oplus x_{10} \oplus x_{15},$$

We address two internal states in each round as follows in ARIA-256: #1 is the state after AK in round 1, #2 is the state after DL in round 1, #3



KL				KR			
KL_0	KL_4	KL_8	KL_{12}	KR_0	KR_4	KR_8	KR_{12}
KL_1	KL_5	KL_9	KL_{13}	KR_1	KR_5	KR_9	KR_{13}
KL_2	KL_6	KL_{10}	KL_{14}	KR_2	KR_6	KR_{10}	KR_{14}
KL_3	KL_7	KL_{11}	KL_{15}	KR_3	KR_7	KR_{11}	KR_{15}

Figure 3: MK of ARIA-256

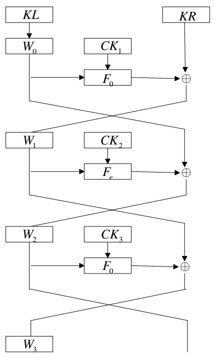


Figure 4: Initialization part of key schedule

is the state after AK in round 2,..., #31 is the state after the first AK in round 16, #32 is the state after the second AK in round 16.

The 256 bites initial key MK of ARIA-256 can be rearranged as a 4×8 matrix as shown in Fig .3.

The key schedule of ARIA-256 consists of two parts, which are initialization and round key generation as follows:

(1) Initialization

In the initialization part, the master key MK denote as follow: $MK = KL \| KR$. Four 128-bit values W_0, W_1, W_2, W_3 are generated from the master key MK, by using a 3-round 256-bit Feistel cipher.

$$W_0 = KL, \qquad W_2 = F_e(W_1, CK_2) \oplus W_0$$
$$W_1 = F_o(W_0, CK_1) \oplus KR, \; W_3 = F_o(W_2, CK_3) \oplus W_1$$

The initialization process is given in Fig. 4.

Here, F_o and F_e are even and odd round functions, respectively, The 128-bit keys CK_i of the round functions are constants.

(2) Round key generation

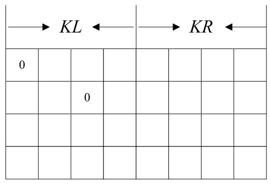


Figure 5: Initialization key $MK[0,0]$

In the generation part, combining the four 18-bits values W_0, W_1, W_2, W_3 to obtain encryption round keys ek_i as follows:

$$\begin{aligned}
&ek_1 = (W_0) \oplus (W_1 >>> 19), &&ek_2 = (W_1) \oplus (W_2 >>> 19),\\
&ek_3 = (W_2) \oplus (W_3 >>> 19), &&ek_4 = (W_0 >>> 19) \oplus (W_3),\\
&ek_5 = (W_0) \oplus (W_1 >>> 31), &&ek_6 = (W_1) \oplus (W_2 >>> 31),\\
&ek_7 = (W_2) \oplus (W_3 >>> 31), &&ek_8 = (W_0 >>> 31) \oplus (W_3),\\
&ek_9 = (W_0) \oplus (W_1 >>> 61), &&ek_{10} = (W_1) \oplus (W_2 >>> 61),\\
&ek_{11} = (W_2) \oplus (W_3 >>> 61), &&ek_{12} = (W_0 >>> 61) \oplus (W_3),\\
&ek_{13} = (W_0) \oplus (W_1 <<< 31), &&ek_{14} = (W_1) \oplus (W_2 <<< 31),\\
&ek_{15} = (W_2) \oplus (W_3 <<< 31), &&ek_{16} = (W_0 <<< 31) \oplus (W_3),\\
&ek_{17} = (W_0) \oplus (W_1 <<< 19).
\end{aligned}$$

4 Biclique Key Recovery Attack for the Full ARIA-256

In this section, combining the Biclique cryptanalysis with the MITM attack, we present the first key recovery method for the full ARIA-256 faster than brute-force.

4.1 Key Partitioning

We define the key sets respect to the master key MK. The base keys $MK[0,0] = KL\|KR$ are all possible 2^{240} 32-byte values with $KL_0 = KL_9 = 0$ whereas the remaining 30 bytes run through all values as shown in Fig. 5.

The keys $\{MK[i,j]\}$ in a set are enumerated by all possible byte differences i and j with respect to the KL value and $F_o(KL, CK_1) \oplus F_o(KL \oplus$

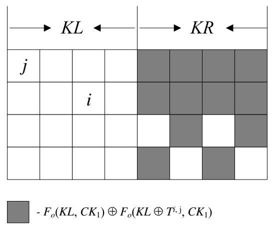


Figure 6: Key set $\{MK[i,j]\}$

$T^{i,j}, CK_1)$ ($T^{i,j}$ denote the input differences of KL) value respect to the KR value of the base key $MK[0,0]$ as shown in Fig. 6.

Hence, the ARIA-256 key space is partitioned into the 2^{240} sets of 2^{16} keys each.

4.2 2-Round Biclique of Dimension 8 for ARIA-256

In this section, we aim to construct a biclique such that the number of rounds contained is as long as possible. Because the diffusion layer of ARIA-256 has the maximum branch number 8, 1 active-byte will transform to 16 active-bytes after 2 round. We only can construct a 2-round biclique from combined related-key differentials as described in section 2.2.

1. The adversary fixes $P_0 = 0$ and derives $S_0 = f_{MK[0,0]}(P_0)$

Note that the diffusion layer of the last round is replaced by a round key addition, it is difficult to construct a long biclique on the bottom of ARIA-256. Hence, we choose the biclique on the top of ARIA-256 which differs from [9].

2. Constructing key differences ∇_i^K and Δ_j^K

The ∇_i-differentials are based on the key difference ∇_i^K, and Δ_j-differentials are based on the key difference Δ_j^K. The choice of ∇_i^K and Δ_j^K is dominated by two aspects. On one hand, in order to construct two independent related-key differentials, the number of active-bytes of the first 2 round key should be as few as possible. Note that the round keys ek_i are generated by W_0, W_1, W_2, W_3. we choose those kind of differences which can make the differences of these four 128-bits values as simple as possible. On the other hand, note that the position of active-byte of the master key will affect the number of active-bytes of the round key, in order to decrease the number of SL and DL bytes which need to be recomputed, we need a type of key that can makes

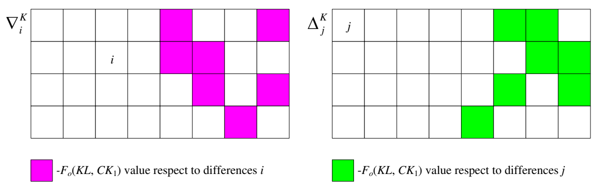

∇_i^K
i
Δ_j^K
j
$-F_o(KL, CK_1)$ value respect to differences i
$-F_o(KL, CK_1)$ value respect to differences j

Figure 7: Initialization Key differentials $\bigtriangledown_i^K$ and $\bigtriangleup_j^K$

the round key have the least number of active-bytes in round 3 after round key generation.

According to the two aspects above, we choose the key as shown in Fig. 7. Key differences $\bigtriangledown_i^K$ are enumerated by all possible byte differences i with respect to the KL value and the $F_o(KL, CK_1) \oplus F_o(KL \oplus T^i, CK_1)$ (T^i denote the input differences of KL) value respect to the KR value of the base key $MK[0,0]$. Hence, the difference of W_1 is 0 and the difference of W_2 is the same as the difference of W_0. A similar property hold for $\bigtriangleup_j^K$. Furthermore, the number of active-bytes of ek_3 in $\bigtriangledown_i$-differentials are determined by the position of the 7 active-bytes of W_3, and the position of differences i affect the position of the 7 active-bytes of W_3. We found KL_9 is the best position of active-byte that can make ek_3 have the least number of active-bytes.

3. Constructing $\bigtriangledown_i$-differentials and $\bigtriangleup_j$-differentials

We can obtain two independent related-key differentials $\bigtriangledown_i$-differentials and $\bigtriangleup_j$-differentials from key differentials $\bigtriangledown_i^K$ and $\bigtriangleup_j^K$ respectively. Both sets of differentials are depicted in Fig .8.

We put:

$$P_i = P_0 \oplus \bigtriangledown_i,$$

$$S_j = S_0 \oplus \bigtriangleup_j$$

$$MK[i,j] = MK[0,0] \oplus \bigtriangledown_i^K \oplus \bigtriangleup_j^K$$

and get a 2-round biclique of dimension 8 $[\{P_i\}, \{S_j\}, \{MK[i,j]\}]$ for ARIA-256.

Since the $\bigtriangledown_i$-differentials affects only 10 bytes of the plaintext, all the plaintexts share the same values in $P_{0,5,8,10,13,15}$. As a result, the data complexity does not exceed 2^{80}.

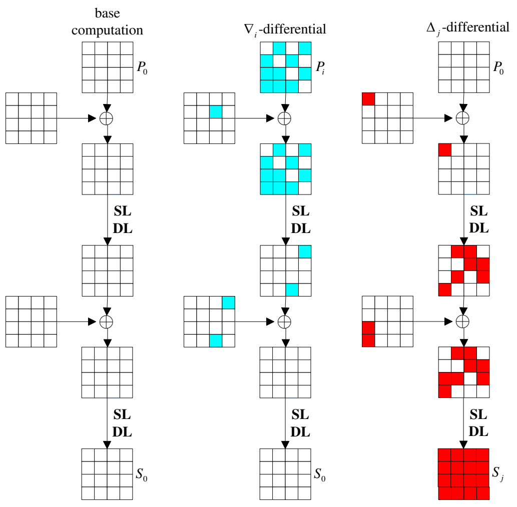


Figure 8: $\bigtriangledown_i$-differentials and $\bigtriangleup_j$-differentials

4.3 Matching over 14 Rounds

Firstly, as shown in Fig. 11, we expand $\bigtriangledown_i$-differentials and $\bigtriangleup_j$-differentials to full-round differentials, and derive the overlapping byte from them. The tested key $MK[i,j]$ is a key candidate of the secret key K_{secret} if

$$S_j \xrightarrow[e_1]{MK[i,j]} \overrightarrow{v}(\#25_0) = \overleftarrow{v}(\#25_0) \xleftarrow[e_2]{MK[i,j]} C_i \tag{2}$$

($\#25_0$ denote the first byte of #25). Here, only matching a single byte, the computation complexity can be decreased.

We first precompute $S_j \xrightarrow{K[0,j]} \overrightarrow{v}$ and $\overleftarrow{v} \xleftarrow{K[i,0]} C_i$ for $0 \leq i, j \leq 2^8 - 1$, and store the 2^9 values as well as the intermediate states and subkeys in memory. Then let i and j run through $\{0, 1, \ldots, 2^8 - 1\}$ respectively and check whether the condition (2) holds. In the matching process, we need only recompute those variables that differ from the ones stored in memory.

Next we evaluate the amount of recomputation in both directions. Because the round from 3 to 11 does not decrease the computation complexity, we omit these rounds as shown in Fig. 9 and Fig. 11.

Recomputation cryptanalysis of forward direction:

As shown in Fig. 9, compute $S_j \xrightarrow{MK[i,j]} \overrightarrow{v}$ differs from the stored bytes of $S_j \xrightarrow{MK[0,j]}$ for all i, j. It is determined by the influence of the difference

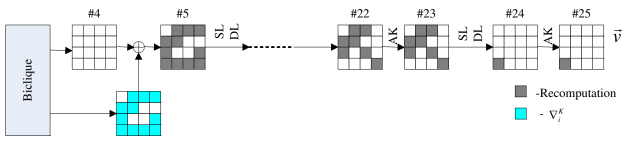


Figure 9: Forward direction

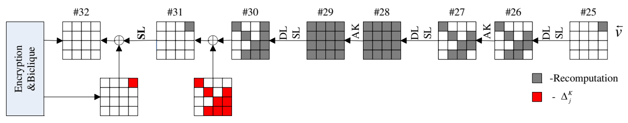


Figure 10: Backward direction

between keys $MK[i,j]$ and $MK[0,j]$. The number byte of internal state which need not to be recomputed is $5+9=14$.

Recomputation cryptanalysis of backward direction:

As shown in Fig. 10, compute $\overleftarrow{v} \xleftarrow{MK[i,j]} C_i$ differs from the stored bytes of $\overleftarrow{v} \xleftarrow{MK[i,0]} C_i$ for all i, j. It is determined by the influence of the difference between keys $MK[i,j]$ and $MK[i,0]$. The number byte of internal state which need not to be recomputed is $15+9+15=39$.

4.4 Complexities

Let us evaluate the full complexity of this approach. Since the full key space partition into the 2^{240} sets, we get the following equation:

$$C_{full} = 2^{n-2d}(C_{Biclique} + C_{precomp} + C_{recomp} + C_{falsepas})$$

where:

- $C_{Biclique}$ is the complexity of constructing a single biclique.
- $C_{precomp}$ is the complexity of the precomputation.
- C_{recomp} is the complexity of the internal variable v 2^{16} times. It strongly depends on the diffusion properties.

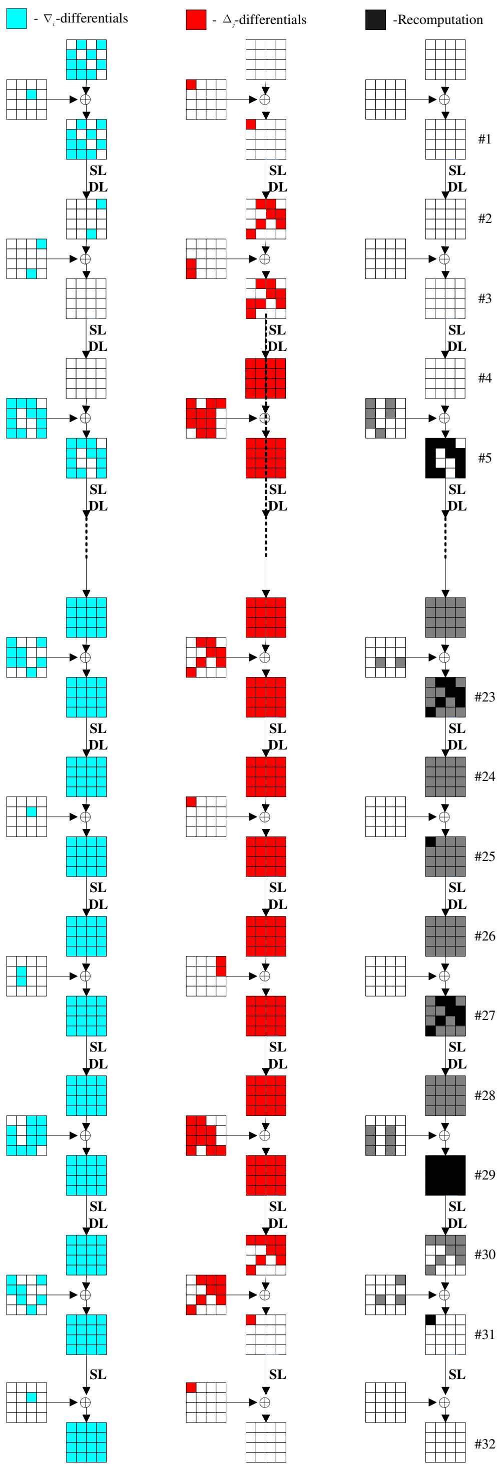


Figure 11: Biclique differentials and matching in ARIA-256

- $C_{falsepas}$ is the complexity generated by false positives, which have to be matched on other variables. If we match on a single byte, the number of false positives is about 2^8. Each requires only a few operations to re-check.

The values $C_{Biclique}$ and $C_{precomp}$ together do not exceed 2^8 calls of the full AIRA-256. The full key recovery complexity is dominated by $2^{n-2d} \cdot C_{recomp}$. SL and DL are the major part of the round transformation and the key schedule. The number of bytes which need to be recomputed is $256 - 32 - 14 - 39 = 171$ in the round transformation and 0 in the key schedule. And the total number of bytes is $16 * 16 + 3 * 16 = 304$. As a result, C_{recomp} is equivalent to $2^{16} * 171/304 = 2^{15.17}$ runs of the full AIRA-256.

Hence, the full computational complexity amounts to about

$$C_{full} = 2^{240}(2^8 + 2^{15.17} + 2^8) = 2^{255.2}.$$

5 Conclusion

In this paper, we construct a 2-round biclique of dimension 8 for ARIA-256 by using a type of special master keys and tools like independent related-key differentials et al. We put forward the first key recovery attack on the full ARIA-256, and it has an advantage of $2^{0.8}$ over brute-force. The data complexity does not exceed 2^{80}. Alternatively, this paper may inspire work on other block ciphers if the key schedule of them diffuse slowly.

References

[1] D. Kwon, J. Kim, S. Park and S.H. Sung et al. New Block Cipher: ARIA. ICISC 2003, LNCS 2971, pp.432-445, Springer-Verlag 2004.

[2] A. Biryukov, C.D. Canniere, J. Lano, S.B. Ors and B. Preneel. Security and Performance Analysis of Aria. Version 1.2. Jan 7, 2004.

[3] W.L. Wu, W.T. Zhang and D.G. Feng. Impossible differential cryptanalysis of Reduced-Round ARIA and Camellia. Journal of Computer Science and Technology 22(3), pp. 44-456, 2007.

[4] E. Fleischmann, M. Gorski and S. Lucks. Attacking Reduced Rounds of the ARIA Block Cipher. Cryptology ePrint Archive, Report 2009/334, 2009. http://eprint.iacr.org/.

[5] P. Li, B. Sun and C. Li. Integral Cryptanalysis of ARIA. In pre-proceeding of Inscrypt 2009.

[6] X.H. Tang, B. Sun, R.L. Li and C. Li. A Meet-in-the-Middle Attack on ARIA. Cryptology ePrint Archive, Report 2010, 2010. http://eprint.iacr.org/.

[7] W. Diffie and M.E. Hellman. Exhaustive Cryptanalysis of the NBS Data Encryption Standard. Computer 10, 74-84 , 1977.

[8] D. Khovratovich, C. Rechberger and A. Savelieva. Bicliques for preimages: attacks on Skein-512 and the SHA-2 family. Cryptology ePrint Archive, Report 2011, 2011 http://eprint.iacr.org/.

[9] A. Bogdanov*, D. Khovratovich and C. Rechberger*. Biclique Cryptanalysis of the Full AES. Cryptology ePrint Archive, Report 2011, 2011 http://eprint.iacr.org/.

[10] H. Demirci, A.A. Selcuk and E. Ture. A new meet-in-the-middle attack on the IDEA Block Cipher. SAC 2003. LNCS 3006, pp. 117C129. Springer, 2004.

[11] H. Demirci and A.A. Selcuk. A Meet-in-the-Middle Attack on 8-Round AES, FSE 2008, LNCS 5086, pp. 116C126, Springer, 2008.

[12] O. Dunkelman, N. Keller and A. Shamir. Improved Single-Key Attacks on 8-round AES. Cryptology ePrint Archive, Report 2010, 2010. http://eprint.iacr.org/.