Impossible Differential Cryptanalysis of ARIA Reduced to 7 Rounds

Chenghang Du and Jiazhe Chen

Key Lab of Cryptologic Technology and Information Security,
Ministry of Education, Shandong University, Jinan, 250100, P.R. China
{chenghangdu,jiazhechen}@mail.sdu.edu.cn

Abstract. This paper studies the security of the block cipher ARIA against impossible differential cryptanalysis. We find a new impossible differential property of ARIA, and propose an attack against ARIA-256 reduced to 7 rounds based on this property, while previous attacks can only attack ARIA up to 6 rounds. Our new attack needs 2^{125} chosen plaintexts and 2^{238} 7-round encryptions. This is the best result for impossible differential cryptanalysis of ARIA known so far.

Keywords: Block cipher, ARIA, Impossible Differential, Data complexity, Time complexity.

1 Introduction

ARIA [12,15] is a block cipher designed by a group of South Korean experts in 2003. ARIA was established as a Korean Standard block cipher algorithm (KS X 1213) by the Ministry of Commerce, Industry and Energy in 2004. ARIA is a general-purpose involution SPN block cipher algorithm, optimized for lightweight environments and hardware implementation. The interface of ARIA is the same as AES [7]. ARIA has 128-bit block size with 128/192/256-bit key, and in the original version the corresponding round numbers are 10/12/14 respectively [12], while in the current one, ARIA v1.0 [15], the round numbers are altered to 12/14/16 respectively.

The designers, Daesung Kwon et al., gave the initial cryptanalysis of ARIA [12]. It contained differential and linear cryptanalysis [4,14], truncated differential cryptanalysis [10], impossible differential cryptanalysis [1], square attack [3,11], higher order differential cryptanalysis [10], interpolation attack [9], and so on. Later in 2004, Alex Biryukov et al. performed a security evaluation of ARIA in which they focused on dedicated linear cryptanalysis and truncated differential cryptanalysis [5], and found attack on ARIA up to 7 rounds. But they didn't evaluate the security against impossible differential cryptanalysis which is an important attacking method of block cipher. Wenli Wu et al. found a non-trivial 4-round impossible differential path in the first place, which led to an attack on 6-round ARIA requiring about 2^{121} chosen plaintexts and about 2^{112} encryptions [17]. Then Shenhua Li proposed an improved impossible differential attack, which needed 2^{96} 6-round encryptions, and reduced the chosen plaintexts number to 2^{120} [13].

S.-H. Heng, R.N. Wright, and B.-M. Goi (Eds.): CANS 2010, LNCS 6467, pp. 20–30, 2010.

Impossible differential cryptanalysis is a kind of technique that uses differentials with probability 0 to get rid of the wrong keys, in order to obtain the right key. These differentials are called impossible differentials. Since its appearance, researchers discovered that it can be used to analyze many block ciphers, such as AES, and get some good results [2,3,6,8,16].

In this paper, we propose a new impossible differential path, which leads to the attack of ARIA-256 reduced to 7 rounds. We use the "early-abort technique" introduced in [4,17] to reduce the time complexity of our attack. The data complexity is 2^{125}, while the time complexity is less than 2^{238} in our attack.

We organize our paper as follows. Section 2 gives a description of ARIA. A 4-round impossible differential path of ARIA is described in Section 3. In Section 4 we present our impossible differential attack on 7-round ARIA-256. And we conclude our paper in Section 5.

2 Description of ARIA

ARIA is a 128-bit SPN structure block cipher. ARIA-256 supports 256-bit key length, and the corresponding round number is 16. Each round consists of the following three parts:

Round Key Addition(AK): This is done by XORing the 128-bit round key $k_i, 1 \leq i \leq 17$. The round key is derived from the master key (MK) through the key schedule. The detail of the key schedule is in [15].

Substitution Layer(SL): Applying the non-linear 8×8-bit S-boxes in parallel on each byte of the state. ARIA uses 2 S-boxes S_1, S_2 and their inverses S_1^{-1}, S_2^{-1}. Each S-box is defined to be an affine transformation of the inversion function over $GF(2^8)$.

$$S : GF(2^8) \longrightarrow GF(2^8),\ S_1 :\ x \longrightarrow Q \cdot x^{-1} \oplus q,$$

where

$$Q = \begin{pmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \end{pmatrix} \quad \textit{and} \quad q = \begin{pmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \end{pmatrix}.$$

and

$$S_2 : \ x \longrightarrow T \cdot x^{247} \oplus t,$$

where

$$T = \begin{pmatrix} 0 & 1 & 0 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 0 & 1 \\ 1 & 1 & 0 & 1 & 0 & 1 & 1 & 1 \\ 1 & 0 & 0 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 & 1 & 1 & 0 & 1 \\ 1 & 1 & 0 & 1 & 0 & 0 & 1 & 1 \end{pmatrix} \quad and \quad t = \begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 1 \end{pmatrix}.$$

There are two types of substitution layers to be used so as to make the cipher involution.

$$LS_o = (S_1, S_2, S_1^{-1}, S_2^{-1}, S_1, S_2, S_1^{-1}, S_2^{-1}, S_1, S_2, S_1^{-1}, S_2^{-1}, S_1, S_2, S_1^{-1}, S_2^{-1}),$$
$$LS_e = (S_1^{-1}, S_2^{-1}, S_1, S_2, S_1^{-1}, S_2^{-1}, S_1, S_2, S_1^{-1}, S_2^{-1}, S_1, S_2, S_1^{-1}, S_2^{-1}, S_1, S_2).$$

LS_o is for the odd rounds, while LS_e is for the even rounds.

Diffusion Layer(DL): A 16×16 involution binary matrix with branch number 8 was selected to improve the diffusion effect. It's a simple linear map in which the 128-bit plaintexts are treated as byte matrices of size 4×4.

The 128-bit plaintext includes 16 bytes with every byte numbered as the following:

0	4	8	12
1	5	9	13
2	6	10	14
3	7	11	15

The diffusion layer is given by $DL : X \to Y, \quad Y = AX$

where

$$X = (x_0, x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8, x_9, x_{10}, x_{11}, x_{12}, x_{13}, x_{14}, x_{15})^T,$$
$$Y = (y_0, y_1, y_2, y_3, y_4, y_5, y_6, y_7, y_8, y_9, y_{10}, y_{11}, y_{12}, y_{13}, y_{14}, y_{15})^T,$$

$$A = \begin{pmatrix} 0&0&0&1&1&0&1&0&1&1&0&0&0&1&1&0 \\ 0&0&1&0&0&1&0&1&1&1&0&0&1&0&0&1 \\ 0&1&0&0&1&0&1&0&0&0&1&1&1&0&0&1 \\ 1&0&0&0&0&1&0&1&0&0&1&1&0&1&1&0 \\ 1&0&1&0&0&1&0&0&1&0&0&1&0&0&1&1 \\ 0&1&0&1&1&0&0&0&0&1&1&0&0&0&1&1 \\ 1&0&1&0&0&0&0&1&0&1&1&0&1&1&0&0 \\ 0&1&0&1&0&0&1&0&1&0&0&1&1&1&0&0 \\ 1&1&0&0&1&0&0&1&0&0&1&0&0&1&0&1 \\ 1&1&0&0&0&1&1&0&0&0&0&1&1&0&1&0 \\ 0&0&1&1&0&1&1&0&1&0&0&0&0&1&0&1 \\ 0&0&1&1&1&0&0&1&0&1&0&0&1&0&1&0 \\ 0&1&1&0&0&0&1&1&0&1&0&1&1&0&0&0 \\ 1&0&0&1&0&0&1&1&1&0&1&0&0&1&0&0 \\ 1&0&0&1&1&1&0&0&0&1&0&1&0&0&1&0 \\ 0&1&1&0&1&1&0&0&1&0&1&0&0&0&0&1 \end{pmatrix}.$$

DL is an involution. So we have $DL^{-1} = DL$.

3 4-Round Impossible Differentials of ARIA

Several 4-round impossible differentials of the ARIA were presented in [13,17]. In this section, we propose some new impossible differential paths of 4-round ARIA.

We use X_m^I and X_m^O to denote the input and output of round m, while X_m^S denotes the intermediate value after the application of SL of round m. $X_{m,n}$ denotes the n-th byte of X_m, while R_m denotes the m-th round. We analyze the 4-round impossible differential of R_3 to R_6.

One new impossible differential path states that, given a pair of X_3^I which is equal in all bytes except the 3rd byte, then after 4 rounds encryption the ciphertext differences ΔX_6^O can't be like this $(j, 0, j, 0, 0, 0, 0, 0, j, 0, 0, j, 0, 0, 0, 0)$, i.e., the ciphertext pair has nonzero equal difference at bytes $(0, 2, 8, 11)$, and no difference at the other bytes.

We expressed the property like this:

$$(0, 0, c, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0) \nRightarrow (j, 0, j, 0, 0, 0, 0, 0, j, 0, 0, j, 0, 0, 0, 0) \quad (1)$$

where c and j denote any nonzero value.

The path is illustrated in Fig.1.

Proof: To start with the first 2 rounds, suppose the difference of inputs satisfies the left part of (1). The first 2-round differential is obtained as follows:

The input difference $\Delta X_3^I = (0, 0, c, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)$ is preserved through the AK operation of R_3. This difference is in a single byte, so the difference after the SL of R_3 is still in a single byte, i.e., $\Delta X_3^S = (0, 0, d, 0, 0, 0, 0, 0, 0, 0,$

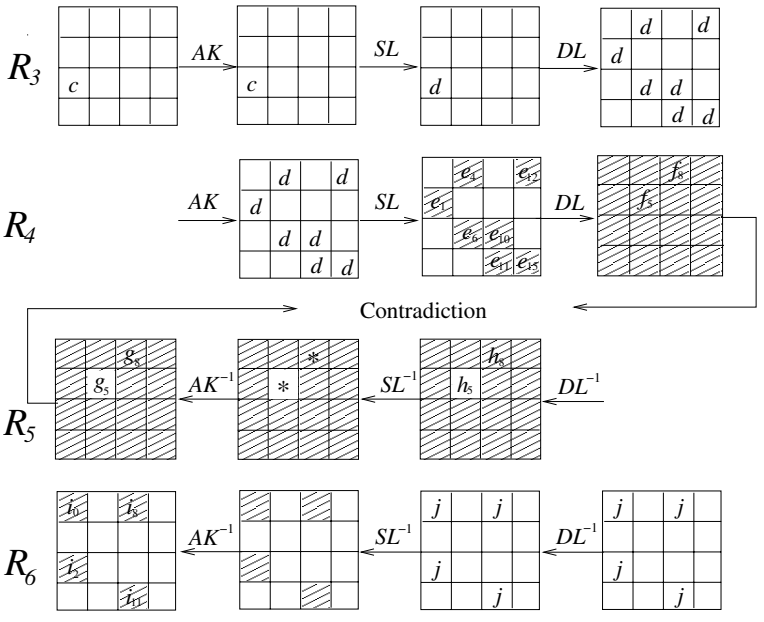

Fig. 1. 4-round impossible differential path of ARIA

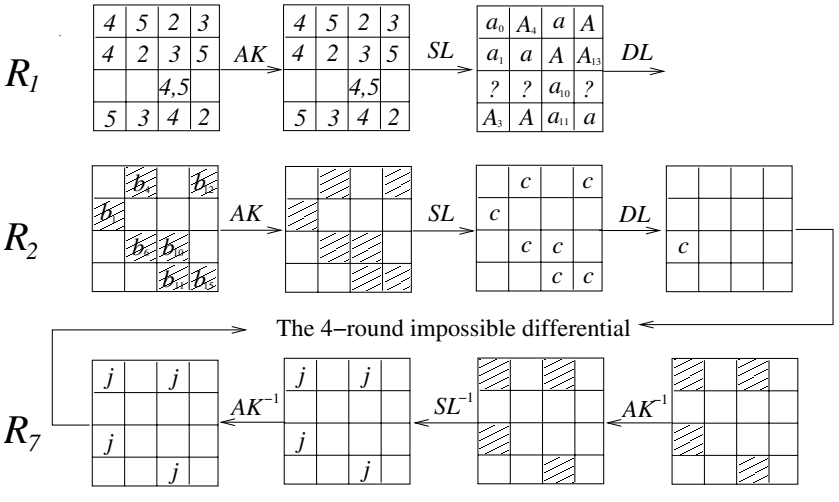

Fig. 2. Impossible Differential Cryptanalysis of 7-round ARIA

$0, 0, 0, 0, 0, 0, 0)$, where d is an unknown nonzero byte. And the DL of R_3 makes the differential become $\Delta X_3^O = (0, d, 0, 0, d, 0, d, 0, 0, 0, d, d, d, 0, 0, d)$.

After AK and SL of R_4, the difference is $\Delta X_4^S = (0, e_1, 0, 0, e_4, 0, e_6, 0, 0, 0, e_{10}, e_{11}, e_{12}, 0, 0, e_{15})$.

Finally, after the DL of R_4, the difference evolves into $\Delta X_4^O = (f_0, f_1, f_2, f_3, f_4, f_5, f_6, f_7, f_8, f_9, f_{10}, f_{11}, f_{12}, f_{13}, f_{14}, f_{15})$, where we have $f_5 = f_8 = e_1 \oplus e_4 \oplus e_{10} \oplus e_{15}$. Hence, $\Delta X_3^I = (0, 0, c, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)$ evolves with probability one into ΔX_4^O, which has same value in bytes 5 and 8.

Now, we investigate how the inverse of the last 2 rounds works on the right part of (1).

The second differential ends after R_6 with difference $\Delta X_6^O = (j, 0, j, 0, 0, 0, 0, 0, j, 0, 0, j, 0, 0, 0, 0)$.

After rolling back this difference through DL, we get $\Delta X_6^S = (j, 0, j, 0, 0, 0, 0, 0, j, 0, 0, j, 0, 0, 0, 0)$. Then after the transformation SL^{-1} and AK^{-1}, the difference is evolved into $\Delta X_6^I = (i_0, 0, i_2, 0, 0, 0, 0, 0, i_8, 0, 0, i_{11}, 0, 0, 0, 0)$ where i_0, i_2, i_8, i_{11} are unknown nonzero byte values.

After DL^{-1} of R_5, the difference is changed to $\Delta X_5^S = (h_0, h_1, h_2, h_3, h_4, h_5, h_6, h_7, h_8, h_9, h_{10}, h_{11}, h_{12}, h_{13}, h_{14}, h_{15})$. Here $h_5 = i_1 \oplus i_3 \oplus i_4 \oplus i_9 \oplus i_{10} \oplus i_{14} \oplus i_{15} = 0 \oplus 0 \oplus 0 \oplus 0 \oplus 0 \oplus 0 \oplus 0 = 0$, and $h_8 = i_0 \neq 0$.

Therefore, when rolling back this difference through SL and AK of R_5, we get $\Delta X_5^I = (g_0, g_1, g_2, g_3, g_4, g_5, g_6, g_7, g_8, g_9, g_{10}, g_{11}, g_{12}, g_{13}, g_{14}, g_{15})$. And we know $g_5 = SL^{-1}(X_5^S) \oplus SL^{-1}(X_5^S \oplus h_5) = 0$, and $g_8 = SL^{-1}(X_5^S) \oplus SL^{-1}(X_5^S \oplus h_8) \neq 0$.

So we have $g_5 \neq g_8$. And also this property stands with probability one.

This differential contradicts the first differential with probability one, which has $f_5 = f_8$.

This contradiction is emphasized in Fig.1.

Some other impossible differential paths like (1) can also be found either. It's just the position has been altered. For instance,

$$(0, 0, 0, c, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0) \not\Rightarrow (0, j, 0, 0, j, j, 0, 0, j, 0, 0, 0, 0, 0, 0, 0),$$
$$(c, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0) \not\Rightarrow (0, 0, 0, 0, j, 0, 0, 0, 0, 0, j, j, 0, j, 0, 0),$$
$$(c, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0) \not\Rightarrow (0, j, 0, 0, j, j, 0, 0, j, 0, 0, 0, 0, 0, 0, 0),$$
$$(0, c, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0) \not\Rightarrow (0, j, 0, j, 0, 0, 0, 0, 0, j, j, 0, 0, 0, 0, 0),$$
$$(0, c, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0) \not\Rightarrow (0, 0, 0, 0, 0, 0, 0, 0, 0, j, j, 0, j, j, 0, 0).$$

4 7-Round Impossible Differential Attack on ARIA-256

In this section, we present an impossible differential cryptanalysis of ARIA-256 reduced to 7 rounds, using the 4-round impossible differential path, which was described in previous section, with additional two rounds at the beginning and one round at the end as shown in Fig.2. Best previous impossible differential attacks could only apply to ARIA reduced to 6-round. Note that the last round of ARIA doesn't have the diffusion layer, but an additional AK.

4.1 Four Equations

We discover some amazing properties of DL transformation, which lead to this cryptanalysis. As illustrated in Fig.2, 4 significant equations of bytes in ΔX_1^S

are found, which make ΔX_1^S evolve into ΔX_2^I with 9 bytes $(0, 2, 3, 5, 7, 8, 9, 13, 14)$ equaling to zero with probability $p = 2^{-24}$, while in the random case the probability is $p = 2^{-72}$. The four equations are:

$$\Delta X_{1,5}^S = \Delta X_{1,8}^S = \Delta X_{1,15}^S \tag{2}$$

$$\Delta X_{1,7}^S = \Delta X_{1,9}^S = \Delta X_{1,12}^S \tag{3}$$

$$\Delta X_{1,0}^S \oplus \Delta X_{1,1}^S \oplus \Delta X_{1,10}^S \oplus \Delta X_{1,11}^S = 0 \tag{4}$$

$$\Delta X_{1,3}^S \oplus \Delta X_{1,4}^S \oplus \Delta X_{1,10}^S \oplus \Delta X_{1,13}^S = 0 \tag{5}$$

In Fig.2, as we can see, in R_1 we use $a, A, a_i, A_j, (i, j \in \{0, 1, ..., 15\})$ to present $\Delta X_{1,k}^S, (k \in \{0, 1, ..., 15\})$, i.e., $a_1 = \Delta X_{1,1}^S$, $A_4 = \Delta X_{1,4}^S$, $a = \Delta X_{1,5}^S = \Delta X_{1,8}^S = \Delta X_{1,15}^S$, etc., and each number in the states before and after AK of R_1 corresponds with an equation. So the 4 equations become:

$$\Delta X_{1,5}^S = \Delta X_{1,8}^S = \Delta X_{1,15}^S = a \tag{2}$$

$$\Delta X_{1,7}^S = \Delta X_{1,9}^S = \Delta X_{1,12}^S = A \tag{3}$$

$$a_0 \ \oplus a_1 \ \oplus a_{10} \ \oplus a_{11} = 0 \tag{4}$$

$$A_3 \oplus A_4 \oplus A_{10} \oplus A_{13} = 0 \tag{5}$$

Now we prove that $p = 2^{-24}$.

Proof: We use structure here to make our argument much easier and more explicit. A structure is defined as a set of 2^{56} differential values of plaintexts which equal to zero in all but 7 bytes $(1, 4, 6, 10, 11, 12, 15)$.

Randomly choose ΔX_2^I from this structure, and through DL^{-1} transformation, we get all the values of $\Delta X_{1,i}^S, 0 \le i \le 15$ as in Fig.2. And we have $\Delta X_{1,5}^S = \Delta X_{2,1}^I \oplus \Delta X_{2,4}^I \oplus \Delta X_{2,10}^I \oplus \Delta X_{2,15}^I$, $\Delta X_{1,8}^S = \Delta X_{2,1}^I \oplus \Delta X_{2,4}^I \oplus \Delta X_{2,10}^I \oplus \Delta X_{2,15}^I$ and $\Delta X_{1,15}^S = \Delta X_{2,1}^I \oplus \Delta X_{2,4}^I \oplus \Delta X_{2,10}^I \oplus \Delta X_{2,15}^I$. Therefore, no matter what the values of $\Delta X_{2,1}^I$, $\Delta X_{2,4}^I$, $\Delta X_{2,10}^I$, $\Delta X_{2,15}^I$ are, equation (2) stands with probability one.

Likewise, equation (3) always holds with probability one.

At the same time, we get $a_0 = \Delta X_{2,4}^I \oplus \Delta X_{2,6}^I, a_1 = \Delta X_{2,4}^I \oplus \Delta X_{2,12}^I, a_{10} = \Delta X_{2,6}^I \oplus \Delta X_{2,15}^I, a_{11} = \Delta X_{2,4}^I \oplus \Delta X_{2,12}^I, A_3 = \Delta X_{2,10}^I \oplus \Delta X_{2,11}^I, A_4 = \Delta X_{2,11}^I \oplus \Delta X_{2,15}^I, A_{13} = \Delta X_{2,6}^I \oplus \Delta X_{2,10}^I$.

It's easy to verify that equations (4) and (5) stand with probability one as well.

It suggests that, if we demand the values of ΔX_2^I are all in the structure defined above, the corresponding values of ΔX_1^S must fulfill all the equations $(2) - (5)$. So we can eliminate all the values of ΔX_1^S that can't satisfy all the equations without deleting a right one. We use p_2, p_3, p_4, p_5 to denote the probability of equations $(2) - (5)$ respectively. We can easily find out that $p_2 = 2^{-16}$, $p_3 = 2^{-16}$, $p_4 = 2^{-8}$, $p_5 = 2^{-8}$. So the number of ΔX_1^S is narrowed down to: $N = 2^{128} \times \prod_{i=2}^{5} p_i = 2^{80}$.

Since DL is a linear transformation, and there are 2^{56} values of ΔX_2^I in the structure, the number of corresponding values of ΔX_1^S which make ΔX_2^I be elements of the structure is also 2^{56}. Spontaneously, any ΔX_1^S which satisfies all the 4 equations evolves ΔX_2^I into the structure with probability $p = \frac{2^{56}}{2^{80}} = 2^{-24}$. □

4.2 The Procedure of 7-Round Attack on ARIA-256

The procedure of this attack is as follows. We use $k_{m,n}$ to denote the n-th byte of k_m.

Step 1. Randomly select 2^{125} plaintexts, and such plaintexts proposes $2^{125} \times 2^{125} \times \frac{1}{2} = 2^{249}$ pairs.

Step 2. Select pairs whose ciphertext pairs have zero difference at the twelve bytes $(1, 3, 4, 5, 6, 7, 9, 10, 12, 13, 14, 15)$. The expected number of such pairs is $2^{249} \times 2^{-96} = 2^{153}$.

Step 3. Guess the 4-byte value $(k_{8,0}, k_{8,2}, k_{8,8}, k_{8,11})$ of the last round key k_8. For each ciphertext pair (C, C'), compute $\Delta X_7^I = SL^{-1}(C \oplus k_8) \oplus SL^{-1}(C' \oplus k_8)$, and choose pairs whose difference ΔX_7^I are same at the 4 bytes $\Delta X_{7,0}^I, \Delta X_{7,2}^I, \Delta X_{7,8}^I, \Delta X_{7,11}^I$. The expected number of the remaining pairs is $2^{153} \times 2^{-24} = 2^{129}$.

Step 4. Next guess all 16 bytes of k_1. But we don't guess all the 16 bytes values at once, we separate them into 5 parts, using the "four equations" presented in the previous subsection.

- **Step 4.1** Guess 3-byte value $(k_{1,5}, k_{1,8}, k_{1,15})$ of the first round key k_1, and for those plaintext pairs (P, P') with such ciphertext pairs, compute $\Delta X_1^S = SL(P \oplus k_1) \oplus SL(P' \oplus k_1)$ at the above 3 bytes. Choose pairs whose difference ΔX_1^S are same at these 3 bytes. The expected number of such pairs is $2^{129} \times 2^{-16} = 2^{113}$.
- **Step 4.2** Guess 3-byte value $(k_{1,7}, k_{1,9}, k_{1,12})$ of k_1, and for the remaining pairs (P, P') compute like above, $\Delta X_1^S = SL(P \oplus k_1) \oplus SL(P' \oplus k_1)$ at the 3 bytes $(7, 9, 12)$. And discard those pairs which have different values at bytes $(7, 9, 12)$. The number of the remaining pairs is $2^{113} \times 2^{-16} = 2^{97}$.
- **Step 4.3** Guess 4-byte value $(k_{1,0}, k_{1,1}, k_{1,10}, k_{1,11})$ of k_1, and compute $\Delta X_1^S = SL(P \oplus k_1) \oplus SL(P' \oplus k_1)$ at the 4 bytes $(0, 1, 10, 11)$. Choose pairs which satisfy the equation : $\Delta X_{1,0}^S \oplus \Delta X_{1,1}^S \oplus \Delta X_{1,10}^S \oplus \Delta X_{1,11}^S = 0$. So there are $2^{97} \times 2^{-8} = 2^{89}$ pairs left.
- **Step 4.4** Guess 3-byte value $(k_{1,3}, k_{1,4}, k_{1,13})$ of k_1, and compute $\Delta X_1^S = SL(P \oplus k_1) \oplus SL(P' \oplus k_1)$ at the 3 bytes $(3, 4, 13)$. Get rid of pairs which don't satisfy the equation : $\Delta X_{1,3}^S \oplus \Delta X_{1,4}^S \oplus \Delta X_{1,10}^S \oplus \Delta X_{1,13}^S = 0$. The number of the remaining pairs is $2^{89} \times 2^{-8} = 2^{81}$.
- **Step 4.5** Guess the last 3-byte value $(k_{1,2}, k_{1,6}, k_{1,14})$ of k_1, and compute $\Delta X_1^S = SL(P \oplus k_1) \oplus SL(P' \oplus k_1)$ at the those 3 bytes like above.
- **Step 4.6** For all 16 bytes values of ΔX_1^S, compute $\Delta X_2^I = DL(\Delta X_1^S)$, pick up pairs whose difference ΔX_2^I are zero at 9 bytes $(0, 2, 3, 5, 7, 8, 9, 13, 14)$. The probability is $p = 2^{-24}$. So the number of the remaining pairs is $2^{81} \times 2^{-24} = 2^{57}$.

Step 5. Guess 7-byte value at $(k_{2,1}, k_{2,4}, k_{2,6}, k_{2,10}, k_{2,11}, k_{2,12}, k_{2,15})$ of k_2, and compute $\Delta X_2^S = SL(X_2^I \oplus k_1) \oplus SL(X_2'^I \oplus k_1)$ at the 7 bytes $(1, 4, 6, 10, 11, 12, 15)$. Choose pairs whose difference ΔX_2^S are same at the 7 bytes $(1, 4, 6, 10, 11, 12, 15)$. The probability is 2^{-48}.

Step 6. Since such a difference is impossible, every value of k_2 which satisfies the difference is wrong value. After we analyze 2^{57} pairs, there are only $2^{56} \times (1 - 2^{-48})^{2^{57}} \approx 2^{-662.3}$ wrong value of k_2 left.

Unless the assumptions on k_8 and k_1 are both correct, it is expected that we can get rid of the whole 56-bit values of k_2 for each 160-bit value of (k_8, k_1), since the number of remaining wrong value of (k_8, k_1, k_2) is about $2^{32} \times 2^{128} \times 2^{-662} = 2^{-512} \approx 0$ [2]. Hence if there remains a value of k_2, we can assume the value of (k_8, k_1, k_2) is right.

4.3 Time Complexity

Next we analyze the time complexity of our attack.

In Step 3, if we compute all the values of those 4 bytes at once, the time complexity of this step will be $2 \times (2^{153} \times 2^{32} \times \frac{4}{16}) = 2^{184}$. But actually we only need 3×2^{168}. Because we can first compute $\Delta X_{7,0}^I$ and $\Delta X_{7,2}^I$, and check if they are equal. And get rid of the pairs which don't. For the rest pairs, continue to compute $\Delta X_{7,8}^I$, and compare with the value of $\Delta X_{7,0}^I$. If they are equal, remain the corresponding pairs, and so on. This is what is called the "early-abort technique". Since we only need to compute 4 bytes here. Thus this step requires $2 \times (2^{153} \times 2^{16} + 2^{145} \times 2^{24} + 2^{137} \times 2^{32}) \times \frac{4}{16} = 3 \times 2^{168}$ one round operations.

And we also use the "early-abort technique" in all the rest steps.

In Step 4.1, because we just compute 3 bytes of plaintext pairs, and only AK and SL are operated, so we consider it as $\frac{3}{16} \times \frac{2}{3}$ one round operations. So just like Step 3, this step requires $2^{32} \times 2 \times (2^{129} \times 2^{16} + 2^{121} \times 2^{24}) \times \frac{3}{16} \times \frac{2}{3} = 2^{176}$ one round operations.

Similarly, Step 4.2 needs $2^{32} \times 2 \times 2^{24} \times (2^{113} \times 2^{16} + 2^{105} \times 2^{24}) \times \frac{3}{16} \times \frac{2}{3} = 2^{184}$ one round operations.

In Step 4.3, we encrypt the 4 bytes $(0, 1, 10, 11)$ of plaintext pairs, and also only AK and SL are operated. So this step demands $2^{32} \times 2 \times 2^{48} \times (2^{97} \times 2^{32}) \times \frac{4}{16} \times \frac{2}{3} = \frac{1}{3} \times 2^{209}$ one round operations.

Just like Step 4.3, Step 4.4 requires $2^{32} \times 2 \times 2^{80} \times (2^{89} \times 2^{24}) \times \frac{3}{16} \times \frac{2}{3} = 2^{223}$ one round operations.

Step 4.5 needs $2^{32} \times 2 \times 2^{104} \times (2^{81} \times 2^{24}) \times \frac{3}{16} \times \frac{2}{3} = 2^{239}$ one round operations.

In Step 4.6, as we considered in Step 4.1, because only DL is operated, we consider it as $\frac{1}{3}$ one round operations.

So here we require $2^{32} \times 2^{128} \times 2 \times 2^{81} \times \frac{1}{3} = \frac{1}{3} \times 2^{242}$ one round operations.

And in Step 5, like Step 3, we demand $2^{32} \times 2^{128} \times 2 \times (2^{57} \times 2^{16} + 2^{49} \times 2^{24} + 2^{41} \times 2^{32} + 2^{33} \times 2^{40} + 2^{25} \times 2^{48} + 2^{17} \times 2^{56}) \times \frac{7}{16} = 21 \times 2^{231}$ one round operations.

Therefore, the total time complexity is $(3 \times 2^{168} + 2^{176} + 2^{184} + \frac{1}{3} \times 2^{209} + 2^{223} + \frac{1}{3} \times 2^{242} + 21 \times 2^{231}) \times \frac{1}{7} = 2^{238}$ encryptions of ARIA-256 reduced to 7 rounds.

Consequently, our attack requires about 2^{125} chosen plaintexts and less than 2^{238} encryptions of 7-round ARIA-256.

5 Conclusion

In this paper, we present a new impossible differential attack against ARIA-256 reduced to 7 rounds. This attack requires 2^{125} chosen plaintexts and 2^{238} encryptions. Our result is the best impossible differential cryptanalysis result on ARIA as far as we know to date. In Table 1, we compare the new attack with the previous impossible differential attacks.

Table 1. Comparison of impossible differential cryptanalysis of ARIA variants

Variant	Number of Rounds	Chosen Plaintexts	Time Complexity	Source
ARIA-128	6	2^{121}	2^{112}	Ref.[17]
ARIA-128	6	2^{120}	2^{96}	Ref.[13]
ARIA-256	7	2^{125}	2^{238}	This paper

Acknowledgments

The authors would like to thank Professor Xiaoyun Wang for her valuable instructions and suggestions. The authors also thank Chengliang Tian and Keting Jia for their useful help. This research is supported by the National 973 Program of China (Grant No.2007CB807902) and the National Natural Science Foundation of China (Grant No.60910118).

References

1. Biham, E., Biryukov, A., Shamir, A.: Cryptanalysis of Skipjack Reduced to 31 Rounds Using Impossible Differentials. In: Stern, J. (ed.) EUROCRYPT 1999. LNCS, vol. 1592, pp. 12–23. Springer, Heidelberg (1999)
2. Biham, E., Dunkelman, O., Keller, N.: Related-Key Impossible Differential Attacks on 8-round AES-192. In: Pointcheval, D. (ed.) CT-RSA 2006. LNCS, vol. 3860, pp. 21–33. Springer, Heidelberg (2006)
3. Biham, E., Keller, N.: Cryptanalysis of Reduced Variants of Rijndael. In: The Third AES Candidate Conference (2000)
4. Biham, E., Shamir, A.: Differential Cryptanalysis of DES-like Cryptosystems. Journal of Cryptology 4(1), 3–72 (1991)
5. Biryukov, A., De Canniere, C., Lano, J., Ors, S.B., Preneel, B.: Security and Performance Analysis of Aria. Version 1.2 (Janaury 7, 2004)

6. Cheon, J.H., Kim, M., Kim, K., et al.: Improved Impossible Differential Cryptanalysis of Rijndael and Crypton. In: Kim, K. (ed.) ICISC 2001. LNCS, vol. 2288, pp. 39–49. Springer, Heidelberg (2002)
7. Daemen, J., Rijmen, V.: The Design of Rijndael. In: Information Security and Cryptography. Springer, Heidelberg (2002)
8. Jakimoski, G., Desmedt, Y.: Related-Key Differential Cryptanalysis of 192-bit key AES Variants. In: Matsui, M., Zuccherato, R. (eds.) SAC 2003. LNCS, vol. 3006, pp. 208–221. Springer, Heidelberg (2004)
9. Jakobsen, T., Knudsen, L.R.: The Interpolation Attack against Block Ciphers. In: Biham, E. (ed.) FSE 1997. LNCS, vol. 1267, pp. 28–40. Springer, Heidelberg (1997)
10. Knudsen, L.R.: Truncated and Higher Order Differentials. In: Preneel, B. (ed.) FSE 1994. LNCS, vol. 1008, pp. 196–211. Springer, Heidelberg (1995)
11. Knudsen, L.R., Wagner, D.: Integral Cryptanalysis (extended abstract). In: Daemen, J., Rijmen, V. (eds.) FSE 2002. LNCS, vol. 2365, pp. 629–632. Springer, Heidelberg (2002)
12. Kwon, D., Kim, J., Park, S., et al.: New Block Cipher: ARIA. In: Lim, J.-I., Lee, D.-H. (eds.) ICISC 2003. LNCS, vol. 2971, pp. 432–445. Springer, Heidelberg (2004)
13. Li, S., Song, C.: Improved Impossible Differential Cryptanalysis of ARIA. In: ISA 2008, pp. 129–132. IEEE Computer Society, Los Alamitos (April 2008)
14. Matsui, M.: Linear Cryptanalysis Method for DES Cipher. In: Helleseth, T. (ed.) EUROCRYPT 1993. LNCS, vol. 765, pp. 386–397. Springer, Heidelberg (1994)
15. National Security Research Institute: Specification of ARIA, Version 1.0 (January 2005), `http://www.nsri.re.kr/ARIA/doc/ARIAspecification-e.pdf`
16. Phan, R.C.: Impossible Differential Cryptanalysis of 7-round AES. Inf. Process. Lett. 91(1), 33–38 (2004)
17. Wu, W., Zhang, W., Feng, D.: Impossible Differential Cryptanalysis of Reduced-Round ARIA and Camellia. Journal of Computer Science and Technology 22(3), 449–456 (2007)