The Journal of Systems and Software 85 (2012) 2096–2103

Contents lists available at SciVerse ScienceDirect

The Journal of Systems and Software

journal homepage: www.elsevier.com/locate/jss

Differential fault analysis of ARIA in multi-byte fault models

Chong Hee Kim

Information Security Group, ICTEAM Institute, Université catholique de Louvain, Place Sainte Barbe, 2, Louvain-la-Neuve, Belgium

ARTICLE INFO

Article history:
Received 21 December 2011
Received in revised form 15 March 2012
Accepted 7 April 2012
Available online 20 April 2012

Keywords:
Cryptanalysis
Security
Differential fault analysis
Block cipher
ARIA

ABSTRACT

Differential fault analysis exploits faults to find secret information stored in a cryptographic device. It utilizes differential information between correct and faulty ciphertexts. We introduce new techniques to improve the previous differential fault analysis of ARIA. ARIA is a general-purpose involutional SPN (substitution permutation network) block cipher and was established as a Korean standard block cipher algorithm in 2004. While the previous method by Li et al. requires 45 faults, our method needs 13 faults to retrieve the 128-bit secret key of ARIA. If access to the decryption oracle is allowed, our method only needs 7 faults. We analyze the characteristics of the diffusion layer of ARIA in detail, which leads us to reduce the number of required faults to find the key.

1. Introduction

Reliable computation is one of the main concerns in many devices. Especially faults occurred during the operations cause many problems such as performance deterioration, unreliable output, etc. Hence, a lot of works to minimize, detect, or prevent faults have been researched. Nowadays, we can easily find cryptographic devices such as smart cards everywhere in our daily lives from banking cards to SIM cards for GSM. These devices are believed to be tamper-resistant. However, if a fault occurs, an adversary may find the secret information stored in the device. Therefore, we are challenging a new type of fault problem.

More precisely, an adversary can find the key of a block cipher using differential information between correct and faulty ciphertexts. This kind of attack is called *differential fault analysis* (DFA). A block cipher is widely used in many cryptographic applications and has been studied extensively in the literature. *Traditional cryptanalysis* of block cipher targets a cipher's design and architecture based on abstract and mathematical approaches. However, in practice a cipher has to be implemented on a real device that is exposed to *physical cryptanalysis* such as side-channel attacks (Dhem et al., 1998; Kocher et al., 1999; Quisquater and Samyde, 2001) and fault attacks (Bar-El et al., 2004; Kim and Quisquater, 2007).

An adversary gets faulty ciphertexts by giving external impact on a device with voltage variation, glitch, laser, etc. (Bar-El et al., 2004). The first DFA presented by Biham and Shamir (1997) targeted DES (National Institute of Standard and Technology, 1993). The ways of exploiting faults to find the key are different according to each algorithm. Therefore, finding an efficient attack for each algorithm is main stream in the research of DFA. Up to now almost all cryptosystems, for example, Triple-DES (Hemme, 2004), RC4 (Biham et al., 2005; Hoch and Shamir, 2004), CLEFIA (Chen et al., 2007; Takahashi and Fukunaga, 2008), RSA (Coron et al., 2010), ElGamal (Bao et al., 1998), IDEA (Clavier et al., 2008), LUC and Demytko (Bleichenbacher et al., 1997), ECC (Blömer et al., 2005; Ciet and Joye, 2005), AES (Piret and Quisquater, 2003; Moradi et al., 2006; Kim and Quisquater, 2008; Takahashi et al., 2007; Barenghi et al., 2010; Kim, 2010), SMS4 and MacGuffin (Li et al., 2009), DSA (Naccache et al., 2005), and ECDSA (Schmidt and Medwed, 2009; Barenghi et al., 2011) have been broken.

The research of DFA can be further diversified into several directions: reducing the number of required faults, applying it to multi-byte fault models, extending to variants, if they exist, or exploring faults induced at an earlier round. In this article, we introduce new fault attacks on ARIA based on multi-byte fault models that needs less faults than the previous one.

ARIA is a general-purpose involutional SPN (substitution permutation network) block cipher algorithm, optimized for lightweight environments and hardware implementation. The name ARIA was taken from the initials of Academia, Research Institute and Agency, acknowledging the co-operative efforts of Korean researchers in designing ARIA. In 2004, ARIA was established as a Korean standard block cipher algorithm (KS X 1213) by the Ministry of Knowledge Economy (ARIA).

0164-1212/$ – see front matter © 2012 Elsevier Inc. All rights reserved.
http://dx.doi.org/10.1016/j.jss.2012.04.009

S^i_0	S^i_1	S^i_2	S^i_3
S^i_4	S^i_5	S^i_5	S^i_7
S^i_8	S^i_9	S^i_{10}	S^i_{11}
S^i_{12}	S^i_{13}	S^i_{14}	S^i_{15}

Fig. 1. State of ARIA.

Several traditional cryptanalysis (Wu et al., 2007; Li and Song, 2008; Li et al., 2008; Fleischmann et al., 2009) and side-channel analysis of ARIA (Ha et al., 2005; Kim et al., 2008; Park et al., 2007; Yoo et al., 2006) have been proposed. However, there is only one result of DFA of ARIA (Li et al., 2008). While the previous method by Li et al. (2008) requires 45 faults to retrieve the key, our method based on a two-byte fault model needs 13 faults. If access to the decryption oracle is allowed, our method needs 7 faults. Our generalized attack, working with faults corrupting a maximum of 4 bytes, can find the key with 21 faults.

This article is organized as follows: Section 2 introduces the ARIA algorithm. The next section briefly describes the previous work by Li et al. and explains our new techniques. Section 4 discusses possible countermeasures. Finally Section 5 concludes the article.

2. ARIA algorithm

ARIA is a 128-bit SPN block cipher[1] with 128-bit, 192-bit, or 256-bit key, where the number of rounds is 12, 14, and 16, respectively (ARIA, in press; Kwon et al., 2003). In the sequel, we will use the 128-bit key version of ARIA cipher, unless otherwise stated.

2.1. Structure of ARIA

The 128-bit input block passes through a round function, which is iterated 12 times (see Fig. 2). The intermediate cipher result, called *State*, can be represented as a two-dimensional byte array with 4 rows and 4 columns. The $S^i = (S^i_0, \ldots, S^i_{15})$ at round i is thus represented by an array as shown in Fig. 1.

The encryption and decryption processes are identical except the use of round keys. Each round consists of the following three parts:

- Round key addition: the State is XORed with a 128-bit subkey.
- Substitution layer (SL): the State goes through 16 S-boxes. There are 2 different substitutions, types 1 and 2, which alternate between the rounds.
- Diffusion layer (DL): it is a function which maps an input $(x_0, x_1, \ldots, x_{15})$ of 16 bytes into an output $(y_0, y_1, \ldots, y_{15})$. The mapping can also be considered as a 16×16 binary matrix multiplication as follows:

$$\begin{pmatrix} y_0 \\ y_1 \\ y_2 \\ y_3 \\ y_4 \\ y_5 \\ y_6 \\ y_7 \\ y_8 \\ y_9 \\ y_{10} \\ y_{11} \\ y_{12} \\ y_{13} \\ y_{14} \\ y_{15} \end{pmatrix} = \begin{pmatrix} 0&0&0&1&1&0&1&0&1&1&0&0&0&1&1&0 \\ 0&0&1&0&0&1&0&1&1&1&0&0&1&0&0&1 \\ 0&1&0&0&1&0&1&0&0&0&1&1&1&0&0&1 \\ 1&0&0&0&0&1&0&1&0&0&1&1&0&1&1&0 \\ 1&0&1&0&0&1&0&0&1&0&0&1&0&0&1&1 \\ 0&1&0&1&1&0&0&0&0&1&1&0&0&0&1&1 \\ 1&0&1&0&0&0&0&1&0&1&1&0&1&1&0&0 \\ 0&1&0&1&0&0&1&0&1&0&0&1&1&1&0&0 \\ 1&1&0&0&1&0&0&1&0&0&1&0&0&1&0&1 \\ 1&1&0&0&0&1&1&0&0&0&0&1&1&0&1&0 \\ 0&0&1&1&0&1&1&0&1&0&0&0&0&1&0&1 \\ 0&0&1&1&1&0&0&1&0&1&0&0&1&0&1&0 \\ 0&1&1&0&0&0&1&1&0&1&0&1&1&0&0&0 \\ 1&0&0&1&0&0&1&1&1&0&1&0&0&1&0&0 \\ 1&0&0&1&1&1&0&0&0&1&0&1&0&0&1&0 \\ 0&1&1&0&1&1&0&0&1&0&1&0&0&0&0&1 \end{pmatrix} \begin{pmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \\ x_7 \\ x_8 \\ x_9 \\ x_{10} \\ x_{11} \\ x_{12} \\ x_{13} \\ x_{14} \\ x_{15} \end{pmatrix}$$

In the last round, instead of the diffusion layer, there is another key addition.

2.2. Key expansion of ARIA

The ARIA key expansion consists of two parts: *initialization* and *subkey generation*. In the initialization part, four 128-bit values, W_0, W_1, W_2, and W_3, are generated from the master key by using a 3-round Feistel cipher. Then the subkeys are generated by a sequence of XOR, rotate-right and rotate-left operations as follows:

$$\begin{aligned} ek_1 &= W_0 \oplus W_1^{\gg 19}, ek_2 = W_1 \oplus W_2^{\gg 19}, ek_3 = W_2 \oplus W_3^{\gg 19}, \\ ek_4 &= W_0^{\gg 19} \oplus W_3, ek_5 = W_0 \oplus W_1^{\gg 31}, ek_6 = W_1 \oplus W_2^{\gg 31}, \\ ek_7 &= W_2 \oplus W_3^{\gg 31}, ek_8 = W_0^{\gg 31} \oplus W_3, ek_9 = W_0 \oplus W_1^{\ll 61}, \\ ek_{10} &= W_1 \oplus W_2^{\ll 61}, ek_{11} = W_2 \oplus W_3^{\ll 61}, \\ ek_{12} &= W_0^{\ll 61} \oplus W_3, ek_{13} = W_0 \oplus W_1^{\ll 31}. \end{aligned}$$

The subkeys for decryption are derived from the subkeys for encryption as follows:

$$\begin{aligned} dk_1 &= ek_{13}, dk_2 = DL(ek_{12}), \ldots, \\ dk_{12} &= DL(ek_2), dk_{13} = ek_1. \end{aligned}$$

2.3. Notations

We use the following notations to describe ARIA. We denote by $X \in (\{0,1\}^8)^{16}$ the plaintext and by $Y \in (\{0,1\}^8)^{16}$ the ciphertext. The ith subkey is denoted by $ek_i \in (\{0,1\}^8)^{16}$, $1 \le i \le 13$. We denote by $A_i = (a_{0,i}, a_{1,i}, a_{2,i}, \ldots, a_{15,i})$ and $B_i = (b_{0,i}, b_{1,i}, b_{2,i}, \ldots, b_{15,i})$ the input and the output of the substitution layer at round i, $1 \le i \le 12$, respectively. We denote by $C_i = (c_{0,i}, c_{1,i}, c_{2,i}, \ldots, c_{15,i})$ the output of the linear layer at round i, $1 \le i \le 12$ (see Fig. 2). We denote by $A^*_i = (a^*_{0,i}, a^*_{1,i}, a^*_{2,i}, \ldots, a^*_{15,i})$ and $B^*_i = (b^*_{0,i}, b^*_{1,i}, b^*_{2,i}, \ldots, b^*_{15,i})$ the faulty input and output of the substitution layer at round i, $1 \le i \le 12$, respectively. The faulty output of the linear layer at round i is denoted by $C^*_i = (c^*_{0,i}, c^*_{1,i}, c^*_{2,i}, \ldots, c^*_{15,i})$, $1 \le i \le 12$.

Let $\Delta A_i = (\Delta a_{0,i}, \Delta a_{1,i}, \Delta a_{2,i}, \ldots, \Delta a_{15,i})$ be the difference between A_i and A^*_i, $1 \le i \le 12$. We denote by ΔB_i and ΔC_i the difference between B_i and B^*_i, and C_i and C^*_i, $1 \le i \le 12$, respectively. We denote by $SL(A_i)$ the output of 128-bit substitution layer for the input A_i, $1 \le i \le 12$. We denote by $SL^{-1}(B_i)$ the output of the inversion of 128-bit substitution layer for the input B_i, $1 \le i \le 12$. Let $DL(B_i)$ be the output of 128-bit diffusion layer for the input

[1] ARIA has three versions according to slightly different key expansion methods: V0.8, V0.9 and V1.0. We use the latest standard version, V1.0 (ARIA).