EURASIP Journal on
Wireless Communications and Networking
a SpringerOpen Journal

RESEARCH Open Access

Improved differential fault analysis on lightweight block cipher LBlock for wireless sensor networks

Kitae Jeong[1], Changhoon Lee[2*] and Jong In Lim[1]

Abstract

LBlock is a 64-bit lightweight block cipher which can be implemented in both constrained hardware environments, such as wireless sensor network, and software platforms. In this paper, we study the security of LBlock against a differential fault analysis. Based on a random nibble fault model, we propose two versions of the attack on LBlock. In the first attack, we inject random nibble faults to the input register of round 29. As a result, it can recover the secret key of LBlock using an exhaustive search of 2^{25} and five random nibble fault injections on average. This attack can be simulated on a general PC within a few seconds. In the case of second attack, random nibble faults are induced to the input register of round 30. This attack can recover the secret key of LBlock using an exhaustive search of 2^{30} and seven random nibble fault injection on average. This attack can be simulated on a general PC within 1 h. These results are superior to known differential fault analytic result on LBlock.

Introduction

Differential fault analysis (DFA), one of the side channel attacks, was first proposed by Biham and Shamir on DES in 1997 [1]. This attack exploits faults within the computation of a cryptographic algorithm to reveal the secret information. So far, DFAs on many block ciphers such as DES, Piccolo, LED, SEED, and ARIA have been proposed [2-7]. It means that DFA poses a major threat to the security on block ciphers.

LBlock [8] proposed in ACNS 2011 is a 64-bit lightweight block cipher suitable for both constrained hardware environments such as wireless sensor network and software platforms. It is based on the 32-round variant Feistel structure with 64-bit block size and 80-bit key size. There were several cryptanalytic results on LBlock. For example, the proposers of LBlock explored the strength of LBlock against some attacks such as differential cryptanalysis, integral attack, and related-key attack [8]. Also, Karakoc et al. [9] and Liu et al. [10] proposed impossible differential cryptanalysis on a reduced version of LBlock, respectively. On the other hand, in [11], a differential fault analysis on LBlock was proposed. Based on a random bit fault model, the proposed attack needs at least 7 fault injections.

In this paper, we propose a differential fault analysis on LBlock. Based on the random nibble fault model, we consider two fault assumptions. In the first attack (Attack 1), it is assumed that several random nibble faults are injected to the input register of round 29. We can compute the exact fault position by checking the corresponding ciphertext differences. Based on the simulation results, this attack requires an exhaustive search of 2^{25} and five random nibble faults on average, and can recover the 80-bit secret key of LBlock within a few seconds on a general PC. In the case of second attack (Attack 2), to recover the 80-bit secret key of LBlock, we inject several random nibble faults to the input register of round 30. This attack requires an exhaustive search of 2^{30} and seven random nibble faults on average. It can also recover the 80-bit secret key of LBlock within 1 h on a general PC. Considering that the proposed attack in [11] requires at least 7 fault injections, our results are superior to it (see Table 1).

This paper is organized as follows. In the 'Description of LBlock' section, we briefly introduce the structure of LBlock. In the 'Attack 1 - fault position: round 29' and

*Correspondence: chlee@seoultech.ac.kr
[2] Department of Computer Science and Engineering, Seoul National University of Science and Technology, Gongneung-ro, Nowon-gu, Seoul, 139–743, South Korea
Full list of author information is available at the end of the article

Table 1 Comparison between DFA results on LBlock

Reference	Fault assumption	Fault position	Number of fault injection	Exhaustive search
[11]	Random bit	Round 25 to 31	7	-
This paper (Attack 1)	Random nibble	Round 29	5	2^{25}
This paper (Attack 2)	Random nibble	Round 30	7	2^{30}

'Attack 2 - fault position: round 30' sections, our attacks on LBlock are presented. Finally, in the last section, we give our conclusion.

Description of LBlock

In this section, we introduce the structure of LBlock briefly. The notations used in this paper are as follows. Here, a 32-bit value $X = (X_7, X_6, \cdots, X_0)$, where X_i is a nibble value.

- $P = (P^L, P^R)$: a 64-bit plaintext.
- $C = (C^L, C^R)$: a 64-bit ciphertext.
- $I_r = (I_r^L, I_r^R)$: a 64-bit input value of round r $(r = 1, 2, \cdots, 32)$.
- $K_r = (K_{r,7}, K_{r,6}, \cdots, K_{r,0})$: a 64-bit round key of round r.

LBlock is a 64-bit block cipher and supports the 80-bit secret key. As shown in Figure 1, the structure of LBlock is a 32-round iterative structure which is a variant of Feistel network. To generate a 64-bit ciphertext $C = (C^L, C^R)$ from a 64-bit plaintext $P = (P^L, P^R)$, LBlock executes the following procedure. Here, $\lll$ is a left circular rotation.

(1) $I_1 = (I_1^L, I_r^1) \leftarrow (P^L, P^R)$.
(2) For $r = 1, 2, \cdots, 32$, do the following:

$$I_{r+1} = (I_{r+1}^L, I_{r+1}^R) = \left(F(I_r^L, K_r) \oplus (I_r^R \lll 8), I_r^L\right).$$

(3) $(C^L, C^R) \leftarrow (I_{33}^R, I_{33}^L)$.
(4) Output $C = (C^L, C^R)$ as a 64-bit ciphertext.

The round function F is defined as follows (see Figure 2). Here, S and P denote the confusion and diffusion functions.

$$\begin{aligned} F : \{0,1\}^{32} \times \{0,1\}^{32} &\longrightarrow \{0,1\}^{32} \\ (X, K_r) &\longmapsto U = P(S(X \oplus K_r)) \end{aligned}$$

The confusion function S denotes the nonlinear layer of round function F. It consists of eight 4×4 S-boxes S_i in parallel $(i = 1, 2, \cdots, 8)$. The contents of these S-boxes are listed in Table 2.

$$\begin{aligned} S : \quad \{0,1\}^{32} &\longrightarrow \{0,1\}^{32} \\ Y = (Y_7, Y_6, \cdots, Y_0) &\longmapsto Z = (Z_7, Z_6, \cdots, Z_0) \end{aligned}$$

$$\begin{aligned} &Z_7 = S_7(Y_7),\ Z_6 = S_6(Y_6),\ Z_5 = S_5(Y_5),\ Z_4 = S_4(Y_4) \\ &Z_3 = S_3(Y_3),\ Z_2 = S_2(Y_2),\ Z_1 = S_1(Y_1),\ Z_0 = S_0(Y_0). \end{aligned}$$

The diffusion function P is defined as a permutation of eight nibble words, and it can be expressed as the following equations:

$$\begin{aligned} P : \quad \{0,1\}^{32} &\longrightarrow \{0,1\}^{32} \\ Z = (Z_7, Z_6, \cdots, Z_0) &\longmapsto U = (U_7, U_6, \cdots, U_0) \end{aligned}$$

$$\begin{aligned} &U_7 = Z_6,\ U_6 = Z_4,\ U_5 = Z_7,\ U_4 = Z_5 \\ &U_3 = Z_2,\ U_2 = Z_0,\ U_1 = Z_3,\ U_0 = Z_1. \end{aligned}$$

The 80-bit secret key K is stored in a key register and denoted as $K = (k_{79}, k_{78}, k_{77}, k_{76}, \cdots, k_1, k_0)$. Output the leftmost 32 bits of current content of register K as round subkey K_1, and then operate as follows:

For $i = 1, 2, \cdots, 31$, update the key register K as follows:

(1) $K \leftarrow (K \lll 29)$.
(2) $(k_{79}, k_{78}, k_{77}, k_{76}) = S_9\,((k_{79}, k_{78}, k_{77}, k_{76}))$.
(3) $(k_{75}, k_{74}, k_{73}, k_{72}) = S_9\,((k_{75}, k_{74}, k_{73}, k_{72}))$.
(4) $(k_{50}, k_{49}, k_{48}, k_{47}, k_{46}) \leftarrow ((k_{50}, k_{49}, k_{48}, k_{47}, k_{46}) \oplus i)$.
(5) Output the leftmost 32 bits of current content of register K as a round key K_{i+1} of round $i + 1$.

Table 3 presents the partial secret keys used in each round key of LBlock. For example, a round key K_{29} of round 29 includes a 32-bit partial secret key $(k_{67}, k_{66}, \cdots, k_{37}, k_{36})$.

Attack 1 - fault position: round 29

In this section, we propose DFA on LBlock, where the fault position is the input register of round 29. Our fault assumption includes the following assumptions:

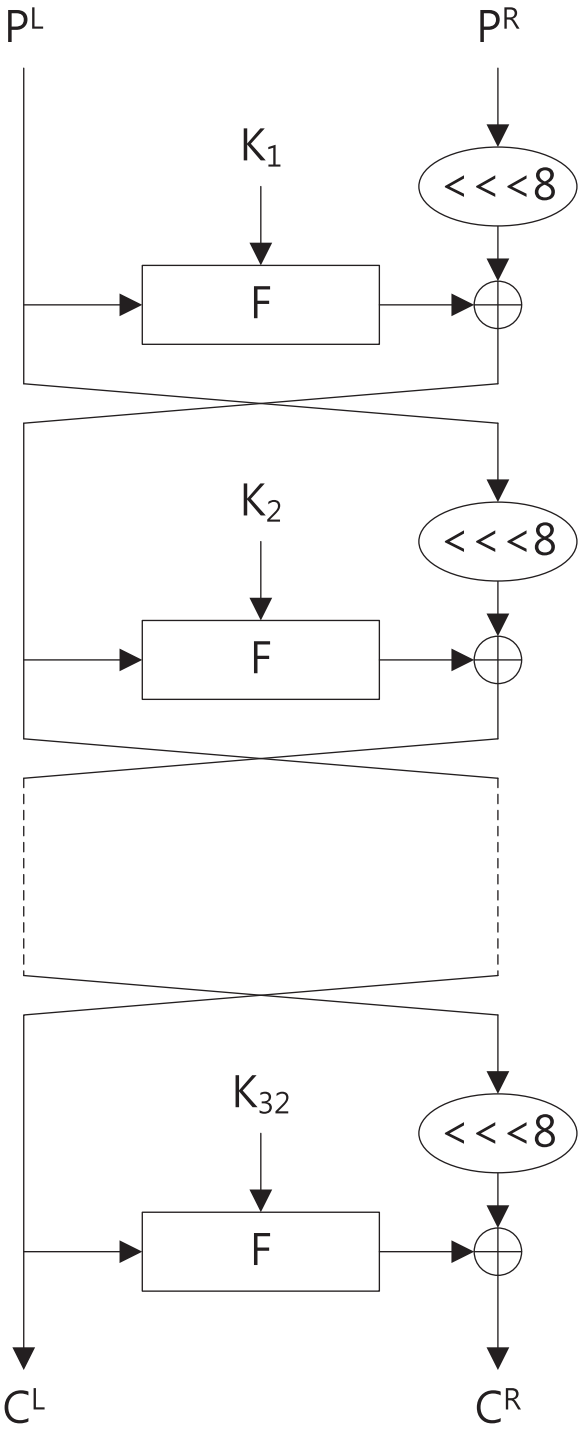

Figure 1 The structure of LBlock.

(1) The attacker has the capability to choose one plaintext to encrypt and obtain the corresponding right/faulty ciphertexts.
(2) The attacker can induce random byte faults to the input register of round 29.
(3) The location and value of faults are both unknown.

From the above assumptions, a random nibble fault can be induced to the input byte register $I^L_{29,i}$ of round 29 ($i = 0, 1, \cdots, 7$). Note that in Attack 1, we do not consider events injecting random nibble faults to $I^R_{29,i}$. They are considered in Attack 2, where random nibble faults are injected to $I^L_{30,i}$. Thus, the number of all possible fault positions is 8. For the simplicity of notations, we denote each case by $E^L_{29,i}$. For example, $E^L_{29,7}$ means an event that a random nibble fault is injected to $I^L_{29,7}$.

Computation of the exact fault position

First, we assume that a random nibble fault was injected to $I^L_{29,7}$, that, is an event $E^L_{29,7}$ was occurred. Figure 3 presents the differential propagation under this assumption.

According to our fault assumption, the input difference ΔI_{29} of I_{29} has the following pattern. Here, $a \neq 0$.

$$\Delta I_{29} = \left[\Delta I^L_{29}, \Delta I^R_{29}\right] = [(a,0,0,0,0,0,0,0), (0,0,0,0,0,0,0,0)].$$

Then, as shown in Figure 3, the output difference of round function F of round 29 is computed as follows. Here, b is an output difference of S-box S_7 taking a as an input difference ($b = S_7(a)$). Thus, the input difference of round 30 has the following pattern:

$$\Delta I_{30} = \left[\Delta I^L_{30}, \Delta I^R_{30}\right] = [(0,0,b,0,0,0,0,0), (a,0,0,0,0,0,0,0)].$$

The input difference ΔI_{31} of round 31 is computed as follows. Here, c is an output difference of S-box S_5 taking b as an input difference ($c = S_5(b)$). Note that, in round 30, a was moved from $\Delta I^R_{30,7}$ to $\Delta I^L_{31,1}$ by an 8-bit left circular rotation.

$$\Delta I_{31} = \left[\Delta I^L_{31}, \Delta I^R_{31}\right] = [(0,0,0,c,0,0,a,0), (0,0,b,0,0,0,0,0)].$$

Similarly, the input difference ΔI_{32} of round 32 has the following pattern. Here, $d = S_4(a)$ and $e = S_1(a)$.

$$\Delta I_{32} = \left[\Delta I^L_{32}, \Delta I^R_{32}\right] = [(b,d,0,0,0,0,0,e), (0,0,0,c,0,0,a,0)].$$

Hence, when a random nibble fault was injected to $I^L_{29,7}$, that is an event $E^L_{29,7}$, the ciphertext difference has the following pattern.

Here, $f = S_6(d)$, $g = S_7(b)$, and $h = S_0(e)$.

$$\Delta C = \left[(b,d,0,0,0,0,0,e), (f,c,g,0,a,h,0,0)\right].$$

Other events $E^L_{29,i}$ can be explained in a similar fashion ($i = 0, 1, \cdots, 6$). Table 4 shows the patterns of ciphertext differences for the positions of fault injections. Here, '?' means a nonzero value. From this table, we can check that the patterns of the ciphertext differences for each event are different from each other. Thus, we can compute the

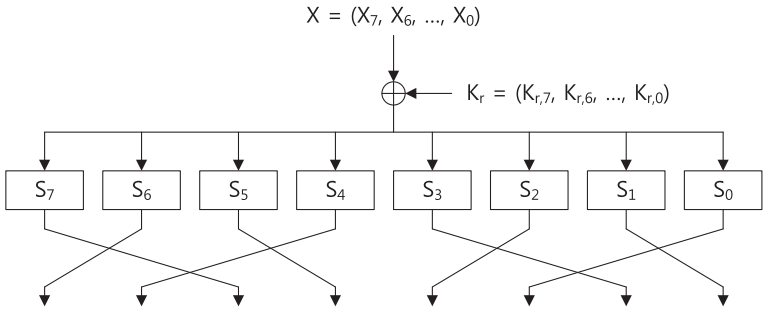

Figure 2 Round function *F* of LBlock.

exact fault position from the patterns of the ciphertext differences.

Computation of round keys for each fault position

We assume that an event $E^L_{29,7}$ has occurred. That is, it is assumed that a random nibble fault was induced to $I^L_{29,7}$. In this case, we can compute 2^{28} candidates of the 56-bit round key by executing the following procedure:

(1) $K_{32,7}$. Guess 4-bit $K_{32,7}$ and compute the output difference of S-box S_7 in round 32 (see red lines in Figure 3). Then, check that this value is equal to ΔC^R_5. The probability passing this test is 2^{-4}. Thus, we can expect that only the right $K_{32,7}$ has survived.
(2) $K_{32,6}$. Guess 4-bit $K_{32,6}$ and compute the output difference of S-box S_6 in round 32 (see red lines in Figure 3). Then, check that this value is equal to ΔC^R_7. Since the filtering probability is 2^{-4}, we can compute the right $K_{32,6}$.
(3) $K_{32,0}$. Guess 4-bit $K_{32,0}$ and compute the output difference of S-box S_0 in round 32 (see red lines in Figure 3). Then, check that this value is equal to ΔC^R_2. The probability passing this test is 2^{-4}. Thus, we can expect that only the right $K_{32,0}$ has survived.
(4) $(K_{31,4}, K_{32,4})$. Guess 8-bit $(K_{31,4}, K_{32,4})$ and compute the output difference of S-box S_4 in round 31 (see blue lines in Figure 3). Then, check that this value is equal to ΔC^L_6. Since the filtering probability is 2^{-4}, we can get 2^4 candidates of $(K_{31,4}, K_{32,4})$.
(5) $(K_{31,1}, K_{32,2})$. Guess 8-bit $(K_{31,1}, K_{32,2})$ and compute the output difference of S-box S_1 in round 31 (see blue lines in Figure 3). Then, check that this value is equal to ΔC^L_0. Since the filtering probability is 2^{-4}, we can get 2^4 candidates of $(K_{31,1}, K_{32,2})$.
(6) $(K_{30,5}, K_{31,6}, K_{32,1})$. Guess 12-bit $(K_{30,5}, K_{31,6}, K_{32,1})$ and compute the output difference of S-box S_5 in round 30 (see green lines in Figure 3). Then, check that this value is equal to ΔC^R_6. Since the filtering probability is 2^{-4}, we can get 2^8 candidates of $(K_{30,5}, K_{31,6}, K_{32,1})$.
(7) $(K_{29,7}, K_{30,3}, K_{31,7}, K_{32,3})$. Guess 16-bit $(K_{29,7}, K_{30,3}, K_{31,7}, K_{32,3})$ and compute the output difference of S-box S_7 in round 29 (see bold black lines in Figure 3). Then, check that this value is equal to ΔC^L_7. Since the filtering probability is 2^{-4}, we can obtain 2^{12} candidates of $(K_{29,7}, K_{30,3}, K_{31,7}, K_{32,3})$.

According to the above procedure, we can obtain 2^{28} candidates of the following 56-bit round key by using one random nibble fault injected to $I^L_{29,7}$.

(1) Round 29: $K_{29,7}$.
(2) Round 30: $(K_{30,3}, K_{30,5})$.

Table 2 Contents of S-boxes used in LBlock

S-box	Contents
S_0	14, 9, 15, 0, 13, 4, 10, 11, 1, 2, 8, 3, 7, 6, 12, 5
S_1	4, 11, 14, 9, 15, 13, 0, 10, 7, 12, 5, 6, 2, 8, 1, 3
S_2	1, 14, 7, 12, 15, 13, 0, 6, 11, 5, 9, 3, 2, 4, 8, 10
S_3	7, 6, 8, 11, 0, 15, 3, 14, 9, 10, 12, 13, 5, 2, 4, 1
S_4	14, 5, 15, 0, 7, 2, 12, 13, 1, 8, 4, 9, 11, 10, 6, 3
S_5	2, 13, 11, 12, 15, 14, 0, 9, 7, 10, 6, 3, 1, 8, 4, 5
S_6	11, 9, 4, 14, 0, 15, 10, 13, 6, 12, 5, 7, 3, 8, 1, 2
S_7	13, 10, 15, 0, 14, 4, 9, 11, 2, 1, 8, 3, 7, 5, 12, 6
S_8	14, 9, 15, 0, 13, 4, 10, 11, 1, 2, 8, 3, 7, 6, 12, 5
S_9	4, 11, 14, 9, 15, 13, 0, 10, 7, 12, 5, 6, 2, 8, 1, 3

Table 3 Partial secret key used in round keys

Round	Partial secret key
1	$(k_{79}, k_{78}, \cdots, k_{49}, k_{48})$
2	$(k_{50}, k_{49}, \cdots, k_{20}, k_{19})$
⋮	⋮
29	$(k_{67}, k_{66}, \cdots, k_{37}, k_{36})$
30	$(k_{38}, k_{37}, \cdots, k_8, k_7)$
31	$(k_9, k_8, \cdots, k_0, k_{79}, k_{78}, \cdots, k_{58})$
32	$(k_{60}, k_{59}, \cdots, k_{30}, k_{29})$

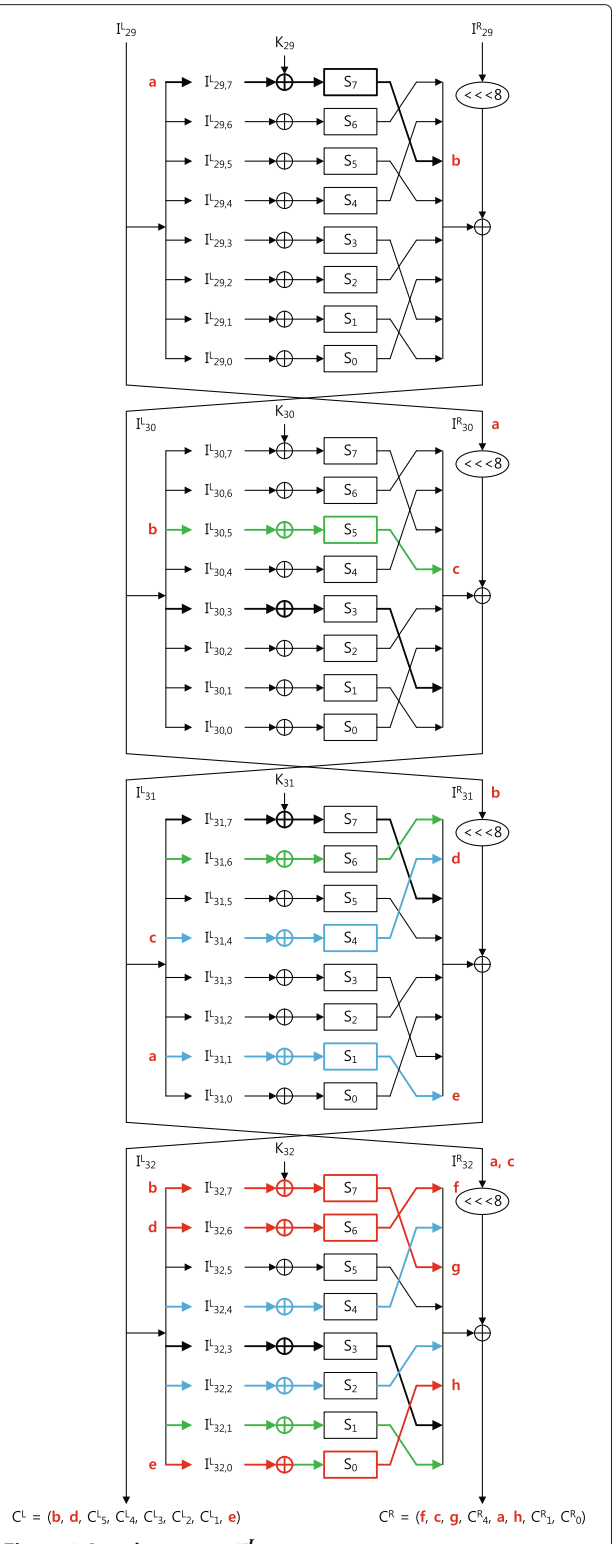

Figure 3 Attack 1 - event $E^L_{29,7}$.

Table 4 Attack 1 - ciphertext differences for the positions of fault injections

Event	Ciphertext difference
$E^L_{29,7}$	[(?, ?, 0, 0, 0, 0, 0, ?), (?, ?, ?, 0, ?, ?, 0, 0)]
$E^L_{29,6}$	[(0, 0, 0, ?, 0, ?, ?, 0), (?, ?, 0, 0, ?, ?, 0, ?)]
$E^L_{29,5}$	[(?, ?, ?, 0, 0, 0, 0, 0), (?, 0, ?, ?, 0, 0, ?, ?)]
$E^L_{29,4}$	[(?, 0, ?, 0, 0, 0, 0, ?), (0, 0, ?, ?, 0, ?, ?, ?)]
$E^L_{29,3}$	[(0, 0, 0, ?, ?, ?, 0, 0), (?, ?, 0, 0, ?, ?, ?, 0)]
$E^L_{29,2}$	[(0, ?, ?, 0, 0, 0, 0, ?), (?, ?, 0, ?, ?, ?, 0, 0)]
$E^L_{29,1}$	[(0, 0, 0, 0, ?, ?, ?, 0), (0, 0, ?, ?, ?, 0, ?, ?)]
$E^L_{29,0}$	[(0, 0, 0, ?, ?, 0, ?, 0), (0, ?, ?, ?, 0, 0, ?, ?)]

(3) Round 31: $(K_{31,1}, K_{31,4}, K_{31,6}, K_{31,7})$.
(4) Round 32: $(K_{32,0}, K_{32,1}, K_{32,2}, K_{32,3}, K_{32,4}, K_{32,6}, K_{32,7})$.

Other events $E^L_{29,i}$ can be explained in a similar fashion ($i = 0, 1, \cdots, 6$). In detail, in each event, we can get 2^{28} candidates of the 56-bit round key from one random nibble fault injection.

- $E^L_{29,6}$
  - Round 29: $K_{29,6}$.
  - Round 30: $(K_{30,1}, K_{30,7})$.
  - Round 31: $(K_{31,0}, K_{31,2}, K_{31,3}, K_{31,5})$.
  - Round 32: $(K_{32,0}, K_{32,1}, K_{32,2}, K_{32,4}, K_{32,5}, K_{32,6}, K_{32,7})$.
- $E^L_{29,5}$
  - Round 29: $K_{29,5}$.
  - Round 30: $(K_{30,4}, K_{30,6})$.
  - Round 31: $(K_{31,1}, K_{31,4}, K_{31,6}, K_{31,7})$.
  - Round 32: $(K_{32,1}, K_{32,2}, K_{32,3}, K_{32,4}, K_{32,5}, K_{32,6}, K_{32,7})$.
- $E^L_{29,4}$
  - Round 29: $K_{29,4}$.
  - Round 30: $(K_{30,4}, K_{30,6})$.
  - Round 31: $(K_{31,1}, K_{31,4}, K_{31,6}, K_{31,7})$.
  - Round 32: $(K_{32,0}, K_{32,1}, K_{32,2}, K_{32,3}, K_{32,4}, K_{32,5}, K_{32,7})$.
- $E^L_{29,3}$
  - Round 29: $K_{29,3}$.
  - Round 30: $(K_{30,1}, K_{30,7})$.
  - Round 31: $(K_{31,0}, K_{31,2}, K_{31,3}, K_{31,5})$.
  - Round 32: $(K_{32,0}, K_{32,2}, K_{32,3}, K_{32,4}, K_{32,5}, K_{32,6}, K_{32,7})$.
- $E^L_{29,2}$
  - Round 29: $K_{29,2}$.
  - Round 30: $(K_{30,3}, K_{30,5})$.
  - Round 31: $(K_{31,1}, K_{31,4}, K_{31,6}, K_{31,7})$.

  - Round 32: $(K_{32,0}, K_{32,1}, K_{32,2}, K_{32,3}, K_{32,4}, K_{32,5}, K_{32,6})$.

- $E^L_{29,1}$
  - Round 29: $K_{29,1}$.
  - Round 30: $(K_{30,0}, K_{30,2})$.
  - Round 31: $(K_{31,0}, K_{31,2}, K_{31,3}, K_{31,5})$.
  - Round 32: $(K_{32,0}, K_{32,1}, K_{32,2}, K_{32,3}, K_{32,5}, K_{32,6}, K_{32,7})$.

- $E^L_{29,0}$
  - Round 29: $K_{29,0}$.
  - Round 30: $(K_{30,0}, K_{30,2})$.
  - Round 31: $(K_{31,0}, K_{31,2}, K_{31,3}, K_{31,5})$.
  - Round 32: $(K_{32,0}, K_{32,1}, K_{32,3}, K_{32,4}, K_{32,5}, K_{32,6}, K_{32,7})$.

Recovery of the secret key from candidates of round keys

In the previous subsection, we presented the method to obtain the candidates of round keys by injecting random nibble faults to the input register of round 29. In this subsection, we explain the method to recover candidates of the secret key of LBlock using candidates of round keys.

As shown in Table 3, the partial secret key used in $(K_{29}, K_{30}, K_{31}, K_{32})$ is as follows:

- K_{29}: $(k_{67}, k_{66}, \cdots, k_{37}, k_{36})$.
- K_{30}: $(k_{38}, k_{37}, \cdots, k_8, k_7)$.
- K_{31}: $(k_9, k_8, \cdots, k_0, k_{79}, k_{78}, \cdots, k_{58})$.
- K_{32}: $(k_{60}, k_{59}, \cdots, k_{30}, k_{29})$.

From the above relation, $(K_{29}, K_{30}, K_{31}, K_{32})$ includes all 80-bit secret key information. Thus, from the keyschedule of LBlock, we can easily compute candidates of the secret key of LBlock by using candidates of round keys computed in the previous subsection. However, in the case that the number of candidates of round keys is very large, we require the exhaustive search with the large computational complexity. On the other hand, from the above relation, we can check that each round key include the common partial secret key information. Thus, if equations are constructed by using this property, we can decrease the number of candidates of the secret key of LBlock.

To decrease the number of candidates of the secret key, we consider equations as shown in Table 5. The total filtering probability is 2^{-41}. Here, '&' means 'AND' operation, and S_9^{-1} and S_8^{-1} are the inverse functions of the S-boxes S_9 and S_8, respectively.

DFA on LBlock (Attack 1)

Now, we are ready to propose a differential fault analysis on LBlock under an assumption that random nibble faults are injected to the input register of round 29. Our attack procedure is as follows:

(1) *Collection of right ciphertext.* Choose a plaintext P and obtain the corresponding right ciphertext $C = (C^L, C^R)$.
(2) *Collection of faulty ciphertext.* After inducing an ith random nibble fault to $I^L_{29} = (I^L_{29,7}, I^L_{29,6}, \cdots, I^L_{29,0})$ of round 29, get the corresponding faulty ciphertext C^i $(i = 1, \cdots, n)$.
(3) *Computation of fault positions.* Compute ΔC^i by using (C, C^i) and then compute the exact fault positions from Table 4.
(4) *Computation of the candidates of* $(K_{29}, K_{30}, K_{31}, K_{32})$. According to the fault positions computed in step 3, compute the candidates of $(K_{29}, K_{30}, K_{31}, K_{32})$ by using the method in 'Computation of round keys for each fault position' section.
(5) *Recovery of the 80-bit secret key.* Using the method in 'Recovery of the secret key from candidates of round keys' section, compute the candidates of the secret key by using the candidates of $(K_{29}, K_{30}, K_{31}, K_{32})$. Then, recover the 80-bit secret key of LBlock by using one trial encryption.

We simulated our attack on a general PC 10,000 times. Based on the simulation results, we can obtain about 2^{25} candidates of the secret key by using five fault injections on average. Thus, we do an exhaustive search for them. Since the filtering probability is 2^{-64}, the expected number of wrong secret keys passing our attack algorithm is $2^{-39} (= 2^{25} \cdot 2^{-64})$. It means that the possibility that a wrong key can pass our attack algorithm is very low. Based on the simulation results, we can always recover the 80-bit secret key of LBlock within a few seconds by using five fault injections on average.

Attack 2 - fault position: round 30

In this section, we propose the second attack (Attack 2) where random nibble faults are induced to the input register of round 30. Since the attack procedure of Attack 2 is similar to that of Attack 1, we briefly discuss the attack procedure of Attack 2.

Our fault assumption is as follows.

- The attacker has the capability to choose one plaintext to encrypt and obtain the corresponding right/faulty ciphertexts.
- The attacker can induce random byte faults to the input register of round 30.
- The location and value of faults are both unknown.

From the above assumptions, a random nibble fault can be induced to the input byte register $I^L_{30,i}$ of round 30 $(i = 0, 1, \cdots, 7)$. Note that, similarly to Attack 1, we do not also

Table 5 Attack 1 - equations to decrease the number of candidates of the secret key

Related secret key	Equation	Filtering probability
(k_{58}, k_{59}, k_{60})	$\{(S_9^{-1}[K_{32,7}]) \gg 1\} = (K_{32,0}\&0x7)$	2^{-3}
(k_{37}, k_{38})	$(K_{32,2}\&0x3) = (K_{30,7} \gg 2)$	2^{-2}
(k_{35}, k_{36})	$(K_{32,1} \gg 2) = (K_{30,7}\&0x3)$	2^{-2}
(k_{33}, k_{34})	$(K_{32,1}\&0x3) = (K_{30,6} \gg 2)$	2^{-2}
(k_{31}, k_{32})	$\{(K_{32,0} \gg 2) \oplus 0x1\} = (K_{30,6}\&0x3)$	2^{-2}
(k_{29}, k_{30})	$\{(K_{32,0}\&0x3) \oplus 0x3\} = (K_{30,5} \gg 2)$	2^{-2}
(k_7, k_8, k_9)	$\{(S_9^{-1}[K_{31,7}]) \gg 1\} = (K_{30,0}\&0x7)$	2^{-3}
k_{60}	$[\{(S_9^{-1}[K_{32,7}]) \gg 3\} \oplus 0x1] = (K_{29,6}\&0x1)$	2^{-1}
(k_{57}, k_{58}, k_{59})	$[\{(S_9^{-1}[K_{32,7}])\&0x7\} \oplus 0x7] = (K_{29,5} \gg 1)$	2^{-3}
k_{56}	$\{(S_8^{-1}[K_{32,6}]) \gg 3\} = (K_{29,5}\&0x1)$	2^{-1}
(k_{53}, k_{54}, k_{55})	$\{(S_8^{-1}[K_{32,6}])\&0x7\} = (K_{29,4} \gg 3)$	2^{-3}
k_{52}	$(K_{32,5} \gg 3) = (K_{29,4}\&0x1)$	2^{-1}
(k_{49}, k_{50}, k_{51})	$(K_{32,5}\&0x7) = (K_{29,3} \gg 1)$	2^{-3}
k_{48}	$(K_{32,4} \gg 3) = (K_{29,3}\&0x1)$	2^{-1}
(k_{45}, k_{46}, k_{47})	$(K_{32,4}\&0x7) = (K_{29,2} \gg 1)$	2^{-3}
k_{44}	$(K_{32,3} \gg 3) = (K_{29,2}\&0x1)$	2^{-1}
(k_{41}, k_{42}, k_{43})	$(K_{32,3}\&0x7) = (K_{29,1} \gg 1)$	2^{-3}
k_{40}	$(K_{32,2} \gg 3) = (K_{29,1}\&0x1)$	2^{-1}
k_{39}	$\{(K_{32,2} \gg 2)\&0x1\} = (K_{29,0} \gg 3)$	2^{-1}
(k_{36}, k_{37}, k_{38})	$\{(S_9^{-1}[K_{30,7}]) \gg 1\} = (K_{29,0}\&0x7)$	2^{-3}

consider events injecting random nibble faults to $I_{30,i}^R$ in Attack 2. Thus, the number of all possible fault positions is 8.

Table 6 shows the patterns of ciphertext differences for the positions of fault injections in Attack 2. Here, '?' means a nonzero value. For example, the differential propagation under an event $E_{30,7}^L$ is shown in Figure 4. From this table, we can check that the patterns of the ciphertext differences for each event are different from each other. Thus, in Attack 2, we can compute the exact fault position from the patterns of ciphertext differences.

Table 6 Attack 2 - ciphertext differences for the positions of fault injections

Event	Ciphertext difference
$E_{30,7}^L$	[(0, 0, 0, ?, 0, 0, ?, 0), (?, ?, 0, 0, 0, 0, 0, ?)]
$E_{30,6}^L$	[(0, 0, ?, 0, 0, 0, 0, ?), (0, 0, 0, ?, 0, 0, ?, ?)]
$E_{30,5}^L$	[(?, ?, 0, 0, 0, 0, 0, 0), (?, ?, ?, 0, 0, 0, 0, 0)]
$E_{30,4}^L$	[(?, ?, 0, 0, 0, 0, 0, 0), (?, 0, ?, 0, 0, 0, 0, ?)]
$E_{30,3}^L$	[(0, 0, ?, 0, 0, 0, 0, ?), (0, 0, 0, ?, ?, ?, 0, 0)]
$E_{30,2}^L$	[(0, 0, 0, ?, 0, 0, ?, 0), (0, ?, ?, 0, 0, 0, 0, ?)]
$E_{30,1}^L$	[(0, 0, 0, 0, ?, ?, 0, 0), (0, 0, 0, 0, ?, ?, ?, 0)]
$E_{30,0}^L$	[(0, 0, 0, 0, ?, ?, 0, 0), (0, 0, 0, ?, ?, 0, ?, 0)]

Recall that, in Attack 1, we get 2^{28} candidates of the 56-bit round key from one random nibble fault injection. In Attack 2, we obtain 2^{12} candidates of the 28-bit round key from one random nibble fault injection.

- $E_{30,7}^L$
  - Round 30: $K_{30,7}$.
  - Round 31: $(K_{31,3}, K_{31,5})$.
  - Round 32: $(K_{32,1}, K_{32,4}, K_{32,5}, K_{32,6})$.
- $E_{30,6}^L$
  - Round 30: $K_{30,6}$.
  - Round 31: $(K_{31,1}, K_{31,7})$.
  - Round 32: $(K_{32,0}, K_{32,2}, K_{32,3}, K_{32,5})$.
- $E_{30,5}^L$
  - Round 30: $K_{30,5}$.
  - Round 31: $(K_{31,4}, K_{31,6})$.
  - Round 32: $(K_{32,1}, K_{32,4}, K_{32,6}, K_{32,7})$.
- $E_{30,4}^L$
  - Round 30: $K_{30,4}$.
  - Round 31: $(K_{31,4}, K_{31,6})$.
  - Round 32: $(K_{32,1}, K_{32,4}, K_{32,6}, K_{32,7})$.

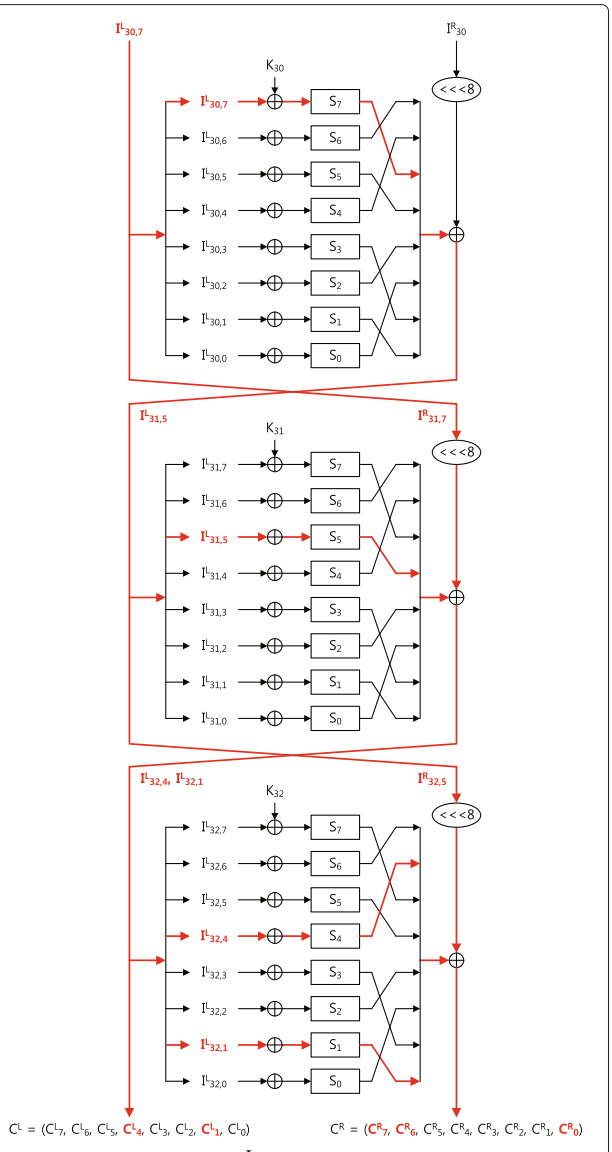

Figure 4 Attack 2 - event $E^L_{30,7}$.

- $E^L_{30,3}$
  - Round 30: $K_{30,3}$.
  - Round 31: $(K_{31,1}, K_{31,5})$.
  - Round 32: $(K_{32,0}, K_{32,2}, K_{32,5}, K_{32,6})$.
- $E^L_{30,2}$
  - Round 30: $K_{30,2}$.
  - Round 31: $(K_{31,3}, K_{31,5})$.
  - Round 32: $(K_{32,1}, K_{32,4}, K_{32,6}, K_{32,7})$.
- $E^L_{30,1}$
  - Round 30: $K_{30,1}$.
  - Round 31: $(K_{31,0}, K_{31,2})$.
  - Round 32: $(K_{32,0}, K_{32,2}, K_{32,3}, K_{32,5})$.
- $E^L_{30,0}$
  - Round 30: $K_{30,0}$.
  - Round 31: $(K_{31,0}, K_{31,2})$.
  - Round 32: $(K_{32,0}, K_{32,2}, K_{32,3}, K_{32,5})$.

Using the candidates of round keys, the method to compute the candidates of the secret key of LBlock in Attack 2 is similar to that in Attack 1. Recall that the partial secret key used in (K_{30}, K_{31}, K_{32}) is as follows:

- K_{30}: $(k_{38}, k_{37}, \cdots, k_8, k_7)$.
- K_{31}: $(k_9, k_8, \cdots, k_0, k_{79}, k_{78}, \cdots, k_{58})$.
- K_{32}: $(k_{60}, k_{59}, \cdots, k_{30}, k_{29})$.

From the above relation, (K_{30}, K_{31}, K_{32}) includes all 80-bit secret key information. Thus, from the keyschedule of LBlock, we can easily compute candidates of the secret key of LBlock by using candidates of the round keys. To decrease the number of candidates of the secret key, we consider seven equations related to round 30 to 32 as shown in Table 5. The total filtering probability is 2^{-16}.

The attack procedure of Attack 2 is as follows:

(1) *Collection of right ciphertext.* Choose a plaintext P and obtain the corresponding right ciphertext $C = (C^L, C^R)$.
(2) *Collection of faulty ciphertext.* After inducing an ith random nibble fault to $I^L_{30} = (I^L_{30,7}, I^L_{30,6}, \cdots, I^L_{30,0})$ of round 30, get the corresponding faulty ciphertext C^i $(i = 1, \cdots, n)$.
(3) *Computation of fault positions.* Compute ΔC^i by using (C, C^i) and then compute the exact fault positions from Table 6.
(4) *Computation of candidates of* (K_{30}, K_{31}, K_{32}). According to the fault positions computed in step 3, compute candidates of (K_{30}, K_{31}, K_{32}).
(5) *Recovery of the 80-bit secret key.* Compute candidates of the secret key by using the candidates of (K_{30}, K_{31}, K_{32}). Then, recover the 80-bit secret key of LBlock by using one trial encryption.

We simulated our attack on a general PC 10,000 times. Based on the simulation results, we can obtain about 2^{30} candidates of the secret key by using seven fault injections on average. Recall that we obtain 2^{12} candidates of the 28-bit round key from one random nibble fault injection under Attack 2. Thus, to get the small number of candidates of the secret key, we need more fault injections than Attack 1. We do an exhaustive search for these candidates. Since the filtering probability is 2^{-64}, the expected number of wrong secret keys

passing our attack algorithm is $2^{-34}(= 2^{30} \cdot 2^{-64})$. It means that the possibility that a wrong key can pass our attack algorithm is very low. Based on the simulation results, we can always recover the 80-bit secret key of LBlock within 1 h by using seven fault injections on average.

Conclusion

In this paper, we have presented DFA on LBlock suitable for wireless sensor networks. The proposed attack has two versions, Attack 1 and Attack 2. To recover the 80-bit LBlock, Attack 1 requires an exhaustive search of 2^{25} and five random nibble fault injections on average. It is executed within a few seconds on a general PC. In the case of Attack 2, this attack is executed within 1 h by using seven random nibble faults. These results are superior to known differential fault analytic result on LBlock.

Competing interests
The authors declare that they have no competing interests.

Acknowledgements
This work was supported by a Korea University grant for author KJ. Also, this research was supported by the The Ministry of Knowledge Economy (MKE), Korea, under the Information Technology Research Center (ITRC) support program (NIPA-2013-H0301–13-3007) supervised by the National IT Industry Promotion Agency (NIPA).

Author details
[1]Center for Information Security Technologies (CIST), Korea University, Anam-dong, Seongbuk-gu, Seoul, 136–713, South Korea . [2] Department of Computer Science and Engineering, Seoul National University of Science and Technology, Gongneung-ro, Nowon-gu, Seoul, 139–743, South Korea .

Received: 3 April 2013 Accepted: 22 May 2013
Published: 3 June 2013

References

1. E Biham, A Shamir, in *Proceedings of Advances in Cryptology - CRYPTO 1997*, ed. by B Kaliski. Differential fault analysis of secret key cryptosystems (Springer Berlin, 1997), pp. 513–525
2. K Jeong, J Sung, S Hong, C Lee, A new approach of differential fault analysis on block ciphers with s-box. Inf. - An Int. Interdiscip. J. **16**(3(A)), 1915–1928 (2013)
3. K Jeong, Security analysis of block cipher piccolo suitable for wireless sensor networks. Peer-to-Peer Netw. Appl (2013). doi:10.1007/s12,083–012–0196–9
4. K Jeong, C Lee, in *Proceedings of The 6th International Symposium on Digital Forensics and Information Security(DFIS-12)*, ed. by J Park, V Leung, C Wang, and Shon T. Differential fault analysis on block cipher led-64 (Springer Netherlands, 2012), pp. 747–755
5. K Jeong, Y Lee, J Sung, S Hong, Differential fault analysis on block cipher seed. Math. Comput. Model. **55**(1–2), 26–34 (2012)
6. W Li, D Gu, J Li, Differential fault analysis on the aria algorithm. Inf. Sci. **178**(19), 3727–3737 (2008)
7. R Malhotra, A Jain, Fault prediction using statistical and machine learning methods for improving software quality. JIPS. **8**(2), 241–262 (2012)
8. W Wu, L Zhang, in *Proceedings of ACNS 2011*, ed. by J Lopez, Tsudik G. Lblock: A lightweight block cipher (Springer Berlin, 2011), pp. 327–344
9. F Karakoc, J Demirci, A Harmanci, in *Proceedings of WISTP 2012*, ed. by I Askoxylakis, H Pohls, and Posegga J. Impossible differential cryptanalysis on reduced-round Lblock (Springer Berlin, 2012), pp. 179–188
10. Y Liu, D Gu, Z Liu, W Li, in *Proceedings of ISPEC 2012*, ed. by M Ryan, B Smyth, and G Wang. Impossible differential attacks on reduced-round Lblock (Springer Berlin, 2012), pp. 97–108
11. L Zhao, T Nishide, K Sakurai, in *Proceedings of COSADE 2012*, ed. by W Schindler, Huss S. Differential fault analysis of full Lblock (Springer Berlin, 2012), pp. 135–150

doi:10.1186/1687-1499-2013-151
Cite this article as: Jeong *et al.*: Improved differential fault analysis on lightweight block cipher LBlock for wireless sensor networks. *EURASIP Journal on Wireless Communications and Networking* 2013 **2013**:151.