Ciphertext-only Fault Analysis on the LED Lightweight Cryptosystem in the Internet of Things

Wei Li, Linfeng Liao, Dawu Gu, Chaoyun Li, Chenyu Ge, Zheng Guo, Ya Liu, and Zhiqiang Liu

Abstract—With the enlargement of wireless technology, Internet of Things (IoT) is emerging as a promising approach to realize smart cities and address lots of serious problems such as safety, convenience and efficiency. In order to avoid any possible rancorous attacks, employing lightweight cryptosystems is most effective to implement encryption/decryption, message authentication and digital signature for security of the IoT. LED is such a lightweight cipher with two flexible keysize variants in the IoT. Since its designing, a multitude of fault analysis techniques in chosen plaintext attacks focus on provoking faults on LED to derive the 64-bit and 128-bit secret keys. It is vital to investigate whether injecting faults allows breaking LED while the attackers have the weakest ciphertext-only attacking ability. This study presents ciphertext-only fault analysis with six different distinguishers on LED. The simulating experiments show that our analysis can recover its 64-bit and 128-bit secret keys with over 99% probability using the SEI, GF, GF-SEI, ML, HW and MAP distinguishers. The attack can not only improve the attacking efficiency, but also decrease the number of faults. The fault locations can be injected into the deeper round. It provides vital reference for security analysis of other lightweight ciphers in the IoT.

Index Terms—IoT, lightweight cryptosystem, LED, ciphertext-only fault analysis.

1 Introduction

Internet of Things (IoT) is appearing as a new landscape of mobile ad-hoc networks, with the aim of providing a wide spectrum of safety and comfort applications for animal tracking, smart buildings, health care, military, transportation and logistics, weather forecast, industrial applications, entertainment, environmental monitoring, and precision agriculture etc. It has been tremendously successful and naturally attracted considerable attention from both academia and industry [1-4]. The IoT consists of spatially distributed autonomous devices using sensors to monitor physical or environmental conditions, and incorporates a gateway that provides wireless connectivity back to the wired world and distributed nodes as Fig. 1 shows. However, the IoT is a network with high dynamic topology and their connections is vulnerable to attacks. For instance, the attackers may exploit the IoT to send bogus information to deceive other nodes. Therefore, conservation of security in the IoT is an indispensable demand. Nodes in the IoT should be confident that each communication has been started from a trustworthy source node and messages are not varied by nodes. Although these issues seem similar to those used in traditional communication networks, there is individual characteristics for the IoT. The seriousness of security failures, the selforganized nature of network, the mobility of sensors, the relevance of nodes to their geographic posi-

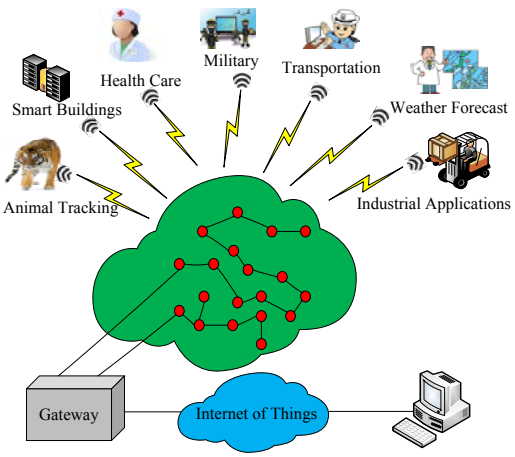


Fig. 1. The IoT scenario.

- *W. Li is with School of Computer Science and Technology, Donghua University, Shanghai 201620, China, with Department of Computer Science and Engineering, Shanghai Jiao Tong University, Shanghai 200240, China, with Shanghai Key Laboratory of Scalable Computing and Systems, Shanghai 200240, China, and also with Shanghai Key Laboratory of Integrate Administration Technologies for Information Security, Shanghai 200240, China.*
  E-mail: liwei.cs.cn@gmail.com.
- *L. Liao and C. Ge is with School of Computer Science and Technology, Donghua University, Shanghai 201620, China.*
  E-mail:lfliaos23@163.com; joygcy@126.com.
- *D. Gu is with Department of Computer Science and Engineering, Shanghai Jiao Tong University, Shanghai 200240, China.*
  E-mail: dwgu@sjtu.edu.cn.
- *C. Li is with Imec-COSIC, KU Leuven, Leuven 3000, Belgium.*
  E-mail: chaoyun.li@esat.kuleuven.be.
- *Z. Guo is with School of Microelectronics, Shanghai Jiao Tong University, Shanghai 200240, China.*
  E-mail: guozheng@sjtu.edu.cn.
- *Y. Liu is with Department of Computer Science and Engineering, University of Shanghai for Science and Technology, Shanghai 200093, China.*
  E-mail: liuyaloccs@gmail.com.
- *Z. Liu (corresponding author) is with Department of Computer Science and Engineering, Shanghai Jiao Tong University, Shanghai 200240, China.*
  E-mail: ilu_zq@sjtu.edu.cn.

tion, and the irregular connectivity among nodes can cause different security issues in the IoT. On the limitation of processing capability, power supply and memory space of highly-constrained devices in nodes, traditional ciphers can not play direct roles in lots of security applications, such as encryption, decryption, message authentication, and digital signature, etc. It is very serious and urgent to implement effective cryptosystems in the IoT, i.e., lightweight ciphers are mostly selected for confidentiality, authentication and integrity [5-12]. Hence, appliance of lightweight cryptosystems can reduce energy consumption for devices, and allow more network communications with lower-resource devices in nodes.

The LED lightweight cryptosystem can be optimized for the RFID tags and other highly-constrained devices for security of nodes in the IoT [12]. Its security has been demonstrated by the designers to be against a linear attack, a different attack, an algebraic attack, a cube tester, an integral attack, a rotational attack and a slide attack. Then Mendel et al. improved the differential attack depending on the mega-boxes and super-boxes [13]. Isobe et al. applied the low key-dependency into the key schedule and presented a meet-in-the-middle attack on the internal rounds of LED [14]. Later Nikolić et al. made use of the multicollision attack and the slidex attack on the round-reduced version of LED [15]. Soleimany presented the probabilistic slide attack on LED-64 [16]. Except the traditional cryptanalysis, much research focuses on LED against fault analysis in recent years [17-21].

In the last two decades, fault analysis puts forward a serious threat for cryptographic implementation. It can deduce the secret key by applying the mathematical relations of a cipher resulting from correct and faulty operations. Boneh et al. presented RSA against fault analysis by provoking the faulty bits in 1996 [22, 23]. Later a multitude of fault analysis techniques, including differential fault analysis (DFA), impossible differential fault analysis (IDFA), and meet-in-the-middle fault analysis (MFA), were later proposed to break block ciphers [24-26]. The attackers can inject faults into the running procedure by exploring a glitch on the clock, a spike on the power supply, or implementing the external ways of the laser and electromagnetic radiations. They make advantage of the leaked faulty calculations with mathematical methods. Usually, fault analysis is much stronger than traditional cryptanalysis.

As for LED, recent studies of fault analysis has been published concerning calculations about the secret key by examining the differential, algebraic or impossible differential relations to recover the subkeys, respectively. Three research groups proposed DFA to break LED in the same year [17-19]. They recovered the last subkey by injecting faults into the antepenultimate round of LED. Jeong et al. derived the 64-bit secret key by one random nibble fault injection [17]. Li et al. extended a random nibble-oriented fault model to a random byte-oriented fault model, and break LED-64 and LED-128 with 3 and 6 faults, respectively [18]. Jovanovic et al. applied some techniques of proportional relationships between different layers to reduce the number of faults to 1 and 2, respectively [19]. Then Zhao et al. proposed an algebraic fault analysis (AFA) by inducing the same faults into the antepenultimate round [20]. In 2016, Li et al. presented an IDFA on LED and extended fault locations to the third last round with 48 and 96 faults, respectively [21]. Hence, the previous fault analysis belongs to the chosen plaintext attack while the attackers can derive the right and faulty ciphertexts with any plaintext.

In 2013, Fuhr et al. presented a ciphertext–only fault analysis(CFA) on AES in three different fault models [27]. That is,

- an all-zero byte fault model;
- a half-zero byte fault model;
- a random byte fault model.

In the former two fault models, the attackers can completely or partly control the values of faults. Their ciphertext-only fault analysis can break AES with a maximum likelihood(ML) distinguisher, and a maximal(minimal) mean Hamming weight(HW) distinguisher when faults are injected into the last round. In the third random byte-oriented fault model, the attackers can inject random values into any target byte of the penultimate round. They used an square Euclidean imbalance(SEI) distinguisher with 320 faulty ciphertexts to recover the last subkey of AES. They made the software experiments to implement the attack. Then in 2016, Dobraunig et al. validated the idea in physical experiment and broke a series of nonce-based authenticated encryption schemes on AES [28].

To the best of our knowledge, few studies have been published on the security of LED against the CFA analysis. Although LED adopts the typical AES-like structure, the main difference puts emphasis on the MixColumns layer in the last round between AES and LED. It is well-known that the last round of LED is composed of four layers, including AddConstants, SubCells, ShiftRows and MixColumnSerial. To break the last subkey, two MixColumnSerial layers in the last two rounds should be included in the statistical relationships. However, owing to the diffusion and confusion of the ShiftRows and SubCells layers, the values of the columns in two MixColumnSerial layers have affected each other. It is not really practical to compute since the complexity is up to 2^{64}. Moreover, in the lightweight circumstance such as the IoT, when the attackers can't derive the right and faulty ciphertext pairs from the same plaintext, all the above chosen-plaintext fault analysis fails definitely. In the real applications, the attackers may usually have the weakest ciphertext-only attacking ability and it is not practical to convert some nibbles to all zero or half zero by injecting faults. Hence, the ciphertext-only fault analysis in the random nibble-oriented fault model is practice-oriented. In other words, any vulnerability of a lightweight cryptosystem against fault analysis should be detected as soon as possible. It is the motivation why we investigate novel ciphertext-only fault analysis on LED.

This study proposes CFA with six different distinguishers to break LED successfully in the software experiment. All distinguishers have been applied successfully to attack the deeper round of the LED cipher. They can improve the attacking efficiency and decrease the number of faults. Table 1 shows the comparison of the previous ciphertext-only fault analysis on AES and our work on LED. When random faults are injected into the penultimate round, the attackers can use not only an SEI distinguisher, but also goodness of fit(GF), goodness of fit-square Euclidean

TABLE 1
Comparison of the ciphertext-only fault analysis to recover one column of last subkey of AES and LED.

Cipher / Distinguisher	AES			LED		
	Fault Model	Round	♯Faults	Fault Model	Round	♯Faults
SEI	Byte	r-1	80	Nibble	r-1	70
GF	-	-	-	Nibble	r-1	60
GF-SEI	-	-	-	Nibble	r-1	53
ML	Byte	r	56	Nibble	r-1	40
HW	Byte	r	72	Nibble	r-1	39
MAP	-	-	-	Nibble	r-1	38

imbalance(GF-SEI), maximum likelihood(ML), Hamming weight(HW), and maximum a posteriori(MAP) distinguishers, respectively. By retrieving the related values of the subkey, our CFA method requires about 152 ciphertexts and 304 ciphertexts to recover the 64-bit and the 128-bit secret keys of LED in the best case, respectively. Table 2 shows the summary of ciphertext-only fault analysis on LED.

TABLE 2
Summary of our ciphertext-only fault analysis on LED.

CFA Distinguisher	LED-64		LED-128	
	♯Faults	Time(s)	♯Faults	Time(s)
SEI	280	10.41	560	20.83
GF	240	9.14	480	18.29
GF-SEI	212	7.95	424	15.90
ML	160	6.35	320	12.69
HW	156	5.73	312	11.47
MAP	152	5.62	304	11.24

The remainder of this paper is organized as follows. Section 2 describes the specification of LED. Section 3 proposes our ciphertext-only fault analysis to break LED-64 and LED-128. Then section 4 analyzes the experimental results. The last section concludes the paper.

2 Specification of LED

LED fixes the block length to 64 bits, and supports key lengths of 64 bits and 128 bits [12]. It has 32 and 48 rounds for LED-64 and LED-128 as Fig. 2 shows. The state can be pictured as a rectangular array of nibbles, consisting of four rows and four columns. Each basic step is a sequence of four identical rounds with a subkey addition, denoted as AddRoundKey(ARK). Each round is composed of AddConstants(AC), SubCells(SC), ShiftRows(SR) and MixColumnsSerial(MC) in sequence:

- The AC layer adds constants to the state with a bitwise XOR operation.
- The SC layer applies S-boxes to each nibble of the state independently.
- The SR layer cyclically shifts each row of the state by different offsets.
- The MC layer takes all the columns and multiply their data with a matrix.

The sequence of steps for the decryption is the same as that for the encryption using the same subkeys. The secret key K depends on a key schedule to generate two subkeys k_1 and k_2 for LED as Table 3 shows.

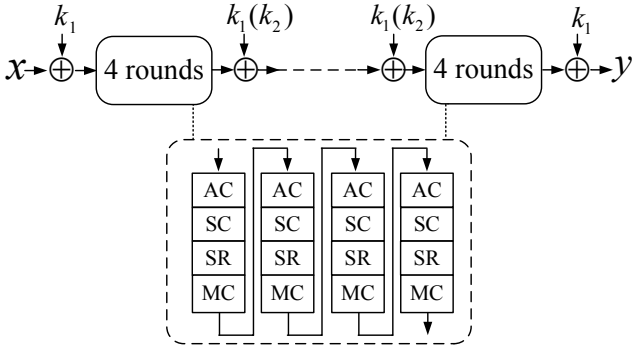


Fig. 2. The structure of LED.

3 Ciphertext-only Fault Analysis on LED

3.1 Notations

Let x and y represent the 64-bit plaintext and ciphertext, respectively.

Let k_1 and k_2 denote the 64-bit subkeys from the secret key K, respectively.

Let r be the number of rounds with $r \in \{32, 48\}$.

Let α_l, β_l, γ_l and δ_l denote the 64-bit output of the AC, SC, SR and MC layers in the l–th round with $1 \leq l \leq r$, respectively.

Let $\hat{\alpha}_l$, $\hat{\beta}_l$, $\hat{\gamma}_l$ and $\hat{\delta}_l$ be the 64-bit faulty output of the above layers in the l-th round with $1 \leq l \leq r$, respectively.

Let AC^{-1}, SC^{-1}, SR^{-1}, and MC^{-1} represent the inverse operation of the above layers, respectively.

Let $\sum$ and $\prod$ denote the sum and multiplication of all elements, respectively.

Let $\sharp$ be the number of elements.

3.2 Basic assumptions

The basic assumption includes ciphertext-only attacks and random nibble-oriented fault model. That is, the attackers have the capability to listen to the encrypted communication. They only know the ciphertexts but not the corresponding plaintexts. This assumption is the weakest in terms of capabilities of the attackers, and thus it is the most practical

TABLE 3
Versions of LED.

Version	Key size	Block size	Rounds	Key schedule
LED-64	64	64	32	$K = k_1$
LED-128	128	64	48	$K = k_1 \| k_2$

in real application. Moreover, the attackers can induce a half byte fault to one layer. However, the value of the fault is unknown. The target nibble can be performed with a bitwise-AND operation by a fault.

3.3 Main procedure

In this subsection, we apply the above basic idea and propose a ciphertext-only fault analysis with six different distinguishers to break the LED cipher. At first, the attackers can induce random errors in some rounds of the encryption, and obtain any faulty ciphertext from any plaintext. They exploit the distribution of a faulty output of the bitwise-AND operation, and then construct a distinguisher. The maximal or minimal value of the distinguisher is connected with the value of the last subkey. The attackers continue to decrypt the right ciphertext and obtain the input of the last round, which is the output of the penultimate round. At last they repeat the above procedure to induce more faults until the secret key is obtained by the key schedule. The detail steps are as follows:

- **Step 1:** The fault injection targets at the (r–1)th round with r=32 in LED-64, or (r–1)th and (r–5)th rounds with r=48 in LED-128. The faulty ciphertexts are derived when random plaintexts are encrypted with the same secret key.
- **Step 2:** This step aims at recovering the subkey k_1 in the last round of LED-64 or LED-128. As Fig. 3 shows, a fault may be induced on either δ_{r-2}, α_{r-1}, β_{r-1} or γ_{r-1}; the approach is identical in either case. Any modification of one nibble in the penultimate round provokes the faulty ciphertext. The original ciphertext y is altered into the faulty ciphertext $\hat{y}$. The attackers have

$$\begin{aligned}&\hat{\gamma}_{r-1}\\ =&MC^{-1}(AC^{-1}(SC^{-1}(SR^{-1}(MC^{-1}(\hat{y}\oplus k_1)))))\\ =&MC^{-1}(AC^{-1}(SC^{-1}(SR^{-1}(MC^{-1}(\hat{y}))\oplus SR^{-1}\\ &(MC^{-1}(k_1)))))\\ =&MC^{-1}(AC^{-1}(SC^{-1}(\hat{y}'\oplus k_1'))),\end{aligned}$$

where

$$\begin{aligned}\hat{y}' &= SR^{-1}(MC^{-1}(\hat{y})),\\ k_1' &= SR^{-1}(MC^{-1}(k_1)).\end{aligned}$$

The attackers can leverage various statistical analysis of the target nibble of $\hat{\gamma}_{r-1}$ to recover four nibbles of k_1'. A list of possible $\hat{\gamma}_{r-1}$ can be deduced by the candidates of k_1'. Then the attackers derive the right k_1' by the maximum or minimum value of a distinguisher. The attackers can take any distinguisher to recover the value of k_1':

- **Square Euclidean Imbalance(SEI)** measures the distance from an unknown distribution to a uniform distribution. The attackers don't need to know the specific distribution of one nibble or byte, which only satisfies a non-uniform distribution. Fuhr et al. applied the SEI distinguisher into security analysis of AES in a random byte-oriented fault model [27]. As for LED, the attackers also have

$$SEI = \sum_{m=0}^{M-1}\left(\frac{\sharp\{\hat{\gamma}_{r-1}|\hat{\gamma}_{r-1}=m,\hat{\gamma}_{r-1}\in\hat{\Upsilon}\}}{N}-\frac{1}{M}\right)^2,$$

where M denotes the total number of one nibble, $m \in [0, M-1]$, N represents the number of all injecting faults, $\hat{\gamma}_{r-1}$ denotes the faulty value of γ_{r-1}, and $\hat{\Upsilon}$ represents the set of all $\hat{\gamma}_{r-1}$. Here, $M = 2^4$. It is the correct k_1' that maximizes the SEI values. Hence, the attackers can compute the maximum value of SEI to distinguish k_1'.

- **Goodness of Fit(GF)** describes how well it fits a set of observations. It summarizes the discrepancy between the observed values and the expected values under the model in question. In our analysis, the distribution of an injected faulty nibble can be deduced on the

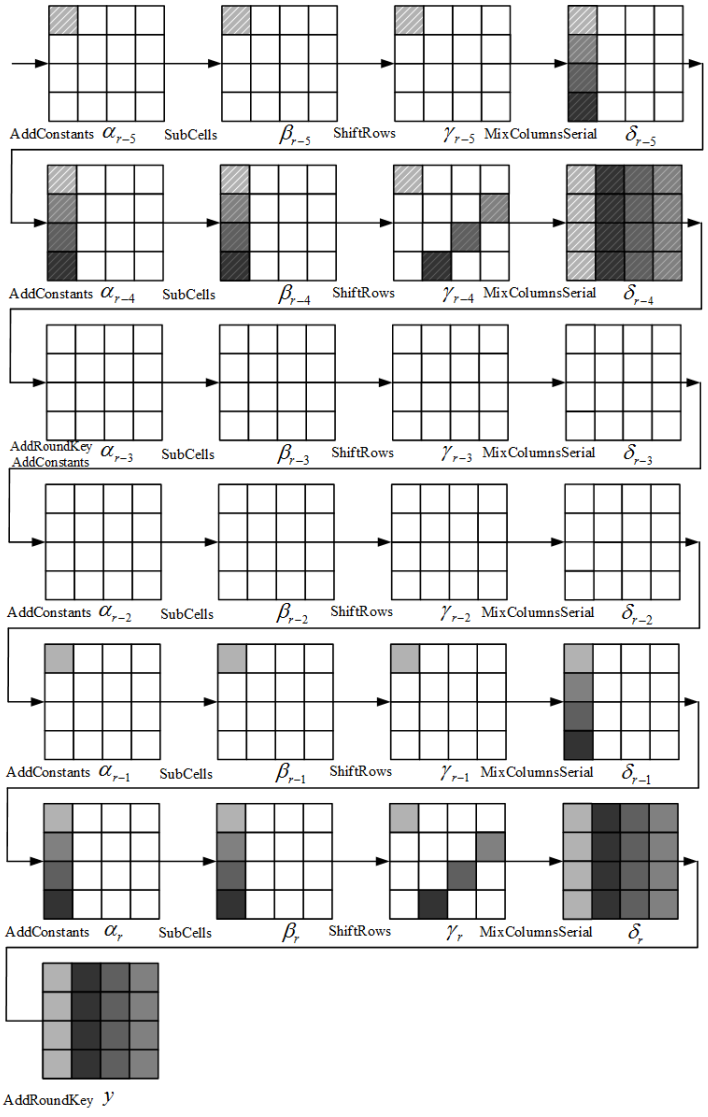


Fig. 3. The faulty attacking paths in the last six rounds in LED-64 and LED-128.

TABLE 4
The output of the nibble bitwise-AND operation.

	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1
2	0	0	2	2	0	0	2	2	0	0	2	2	0	0	2	2
3	0	1	2	3	0	1	2	3	0	1	2	3	0	1	2	3
4	0	0	0	0	4	4	4	4	0	0	0	0	4	4	4	4
5	0	1	0	1	4	5	4	5	0	1	0	1	4	5	4	5
6	0	0	2	2	4	4	6	6	0	0	2	2	4	4	6	6
7	0	1	2	3	4	5	6	7	0	1	2	3	4	5	6	7
8	0	0	0	0	0	0	0	0	8	8	8	8	8	8	8	8
9	0	1	0	1	0	1	0	1	8	9	8	9	8	9	8	9
a	0	0	2	2	0	0	2	2	8	8	a	a	8	8	a	a
b	0	1	2	3	0	1	2	3	8	9	a	b	8	9	a	b
c	0	0	0	0	4	4	4	4	8	8	8	8	c	c	c	c
d	0	1	0	1	4	5	4	5	8	9	8	9	c	d	c	d
e	0	0	2	2	4	4	6	6	8	8	a	a	c	c	e	e
f	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f

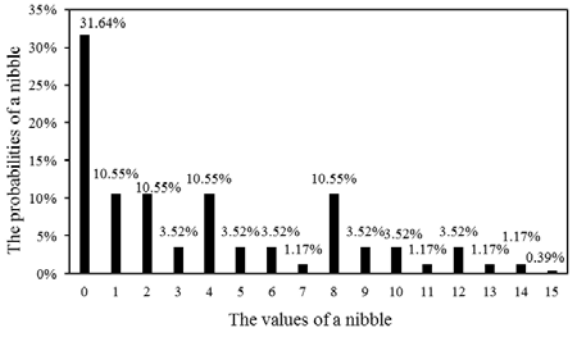
basis of the bitwise-AND operation as Table 4 and Fig. 4 shows. The attackers can do brute-force search on the bitwise-AND operation of two nibbles. There is

$$GF = \sum_{m=0}^{M-1} \frac{(O_m - E_m)^2}{E_m},$$

where

$$O_m = \sharp\{\hat{\gamma}_{r-1} | \hat{\gamma}_{r-1} = m, \hat{\gamma}_{r-1} \in \hat{\Upsilon}, m \in [0, M-1]\},$$
$$E_m = \sharp\{\dot{\gamma}_{r-1} | \dot{\gamma}_{r-1} = m, \dot{\gamma}_{r-1} \in \dot{\Upsilon}, m \in [0, M-1]\},$$

Fig. 4. The bitwise-AND distribution of a nibble after fault injections.



M denotes the total number of one nibble, O_m is an expected number for m, E_m represents a theoretical number for m, $\hat{\gamma}_{r-1}$ is the observed faulty value of γ_{r-1}, $\dot{\gamma}_{r-1}$ represents the expected value of γ_{r-1}, $\hat{\Upsilon}$ denotes the set of all observed $\hat{\gamma}_{r-1}$, and $\dot{\Upsilon}$ represents the set of all observed $\dot{\gamma}_{r-1}$, respectively. Here, $M = 16$. It is the correct k_1' that minimizes the GF values.

- **Goodness of Fit-Square Euclidean Imbalance** (**GF-SEI**) is a novel double distinguisher to combine the advantages of the above single GF distinguisher and single SEI distinguisher. Precisely, if $GF > \chi_a^2$, then $\hat{\gamma}_{r-1}$ can reject the know distribution. χ_a^2 can be deduced by the known degree of freedom df and the defined a significance level of the χ^2-distribution. Here, $M = 16$, and $df = M-1 = 15$. In our analysis, the GF distinguisher is rather effective when $N \geq 50$ and $E_m = \sharp\{\dot{\gamma}_{r-1} | \dot{\gamma}_{r-1} = m, \dot{\gamma}_{r-1} \in \dot{\Upsilon}, m \in [0, M-1]\} \geq 5$. The attackers can exclude the wrong candidate of k_1' by a GF distinguisher, and then deduce the correct k_1' by an SEI distinguisher. The correct k_1' first satisfies $GF \leq \chi_a^2$ and then maximizes the SEI value.

- **Maximum Likelihood** (**ML**) estimation is a method of estimating the parameters of a

given distribution model by finding the parameter value that maximizes the likelihood. Fuhr et al. applied this ML distinguisher to attack AES in all fault models, but just in last round [27]. Here, we can extend the ML distinguisher to the random byte-oriented fault model in the deeper round as follows:

$$ML = \prod_{n=0}^{N-1} p(\hat{\gamma}_{r-1}),$$

where N represents the number of faults, $n \in [0, N-1]$, p is the probability of the element, and $\hat{\gamma}_{r-1}$ represents the observed faulty value of γ_{r-1}. It is the correct k'_1 that maximizes the ML value.

- **Hamming Weight** (**HW**) represents the number of symbols that are different from the zero-symbol of the same length. In our attack, the Hamming weight is the number of nonzero bits of a nibble. Fuhr et al. applied the minimal (maximal) mean HW distinguisher in attacking AES with all fault models [27]. Our attack can apply the HW distinguisher into the random byte-oriented fault model in the deeper round. There is

$$HW = \frac{1}{N} \sum_{n=0}^{N-1} hw(\hat{\gamma}_{r-1}),$$

where N denotes the number of faults, $n \in [0, N-1]$, hw represents the hamming weight of the element, and $\hat{\gamma}_{r-1}$ represents the observed faulty value of γ_{r-1}, respectively. On the basis of the bitwise-AND operation in our fault model, the attackers can compute the minimum value of HW to distinguish k'_1.

- **Maximum a Posteriori** (**MAP**) probability estimate is an estimate of an unknown quantity, that equals the mode of the posterior distribution. It employs an augmented optimization objective which incorporates a prior distribution over the quantity one wants to estimate. There is

$$MAP = \frac{p(\hat{\Upsilon}|k'^t_1) \cdot \pi(k'^t_1)}{\sum_{t=0}^{T-1} p(\hat{\Upsilon}|k'^t_1) \cdot \pi(k'^t_1)},$$

where T denotes the total number of four nibbles in a subkey, $t \in [0, T-1]$, $\pi(k'^t_1)$ represents the prior distribution of k'^t_1, and $p(\hat{\Upsilon}|k'^t_1)$ denotes the conditional probability of $\hat{\Upsilon}$ when the parameter is k'^t_1, respectively. Here, $T = 2^{16}$. It is the correct k'_1 that maximizes the MAP values.

The above distinguishers, in conjunction with multiple faulty ciphertexts $\hat{y}$, allow to collect a list of possible candidates for k'_1. The attackers can do brute-force search for each four nibbles of k'_1, until the set of k'_1 candidates has only one element. Hence, all nibbles of K in the 64-bit secret key version can be derived as follows:

$$K = k_1 = MC(SR(k'_1)).$$

- **Step 3:** This step aims at recovering the subkey k_2 in the last round of LED-128, the attackers can decrypt the last four rounds using the subkey k_1 to obtain the input of the (r–3)th round, represented as α_{r-3}. They can take the above attacking procedure to derive all nibbles of k'_2 when random faults are injected before δ_{r-5} in the (r–5)th round. They have

$$\begin{aligned}\hat{\beta}_{r-4} &= SR^{-1}(MC^{-1}(\hat{\alpha}_{r-3} \oplus \hat{k}_2)) \\ &= SR^{-1}(MC^{-1}(\hat{\alpha}_{r-3})) \oplus SR^{-1}(MC^{-1}(k_2)) \\ &= \alpha'_{r-3} \oplus k'_2,\end{aligned}$$

where

$$\begin{aligned}\hat{\alpha}'_{r-3} &= SR^{-1}(MC^{-1}(\hat{\alpha}_{r-3})), \\ k'_2 &= SR^{-1}(MC^{-1}(k_2)).\end{aligned}$$

Hence, the attackers can take any of the above distinguishers to derive all nibbles of k'_2. The secret key K is deduced as

$$K = k_1||k_2 = k_1||MC(SR(k'_2)).$$

4 Simulation

We implemented the attack on a PC using the Java language with 64GB memory. The fault injections are simulated with 1000 process units by computer software. The number of faults, latency, and time complexity to recover four nibbles of a subkey are taken into consideration to evaluate the experimental results.

Fig. 5 illustrates the possibility of recovering four nibbles of a subkey with different faults, where the x-coordinate represents the number of faults, and the y-coordinate denotes the probability of recovering four nibbles of a subkey, respectively. The colored lines reflect the trend of six distinguishers among SEI, GF, GF-SEI, ML, HW and MAP, respectively. To retrieve four nibbles of a subkey with 99% probability, the faults are between 70, 60, 53, 40, 39 and 38 among different distinguishers in Table 2. Referring to the experimental results, breaking LED-64 requires at most 280 faults and at least 152 faults. And breaking the LED-128 requires at most 560 faults and at least 304 faults.

Latency is the time from the first fault injection to the recovery of four nibbles of a subkey in our software simulation. Fig. 6 shows that the latency of one experiment, which are measured in seconds. The time of one successful experiment with 99% probability is between 1.41s and 2.81s. According to the experimental results in Table 2, the whole attacking procedure requires 5.62s and 11.24s to break LED-64 and LED-128 in the best case, respectively.

On the basis of the number of faults in Table 2, the time complexities of all distinguishers are listed in Table 5, where $T = 2^{16}$, $M = 2^4$, and N represents the number of all injecting faults.

Both Table 2 and Table 5 show that the probability, latency and time complexity of the GF, GF-SEI, ML, HW and MAP distinguishers are better than those of the SEI

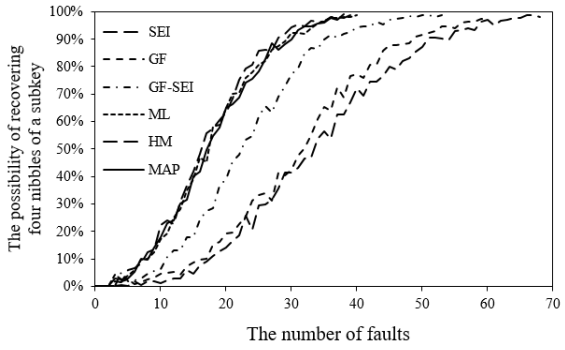


Fig. 5. The recovery of four nibbles on possibility.

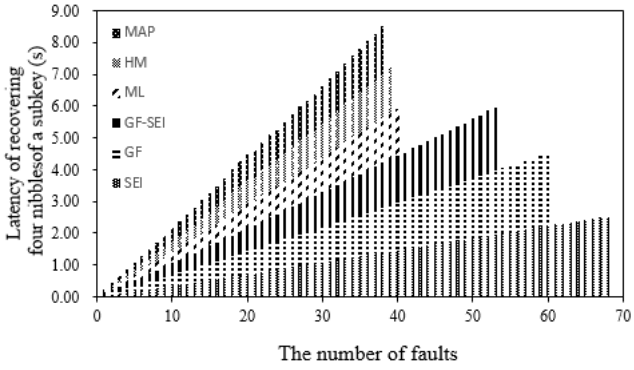


Fig. 6. The recovery of four nibbles on latency with stacked charts.

TABLE 5
Summary of time complexities of attacking LED.

CFA	Time complexity	LED-64	LED-128
SEI	$T*(M+N)$	$2^{24.21}$	$2^{25.17}$
GF		$2^{24.00}$	$2^{24.95}$
GF-SEI		$2^{23.83}$	$2^{24.78}$
ML	$T*N$	$2^{23.32}$	$2^{24.32}$
HW		$2^{23.29}$	$2^{24.39}$
MAP	$T*(N+1)$	$2^{23.26}$	$2^{24.25}$

distinguisher. Compared with a single SEI distinguisher or a single GF distinguisher, the double GF-SEI distinguisher has higher probability, less latency and time complexity. Furthermore, the experimental results of the MAP, HW and ML distinguishers are very close, and those of the MAP distinguisher is in the best case. All experimental results of possibility and latency for each fault are listed in the Appendix.

5 Conclusions

This paper presents the ciphertext-only fault analysis with six distinguishers on the LED cryptosystem in the random nibble-oriented fault model. The analysis could break the 64-bit and 128-bit secret keys of LED by at least 152 and 304 faults in the best case, respectively. It shows that the ciphertext-only fault analysis is a strong threaten to the LED cipher in the IoT. We expect that our research will provide deeper understanding of the security of AES-like lightweight cryptosystems.

Acknowledgments

The authors wish to acknowledge Prof. Vincent Rijmen for helpful suggestions. This work is supported by the Research Council KU Leuven under Grant No. OT/13/071, and European Unions Horizon 2020 research and innovation programme under Grant agreement No. H2020-MSCA-ITN-2014-643161 ECRYPTNET, the National Natural Science Foundation of China under Grant No. 61772129, No. 61472250, No. 61672347, No.61402288, No. 61402286, No. 61572192, Shanghai Natural Science Foundation under Grant No. 15ZR1400300, No. 16ZR1401100, and Opening Project of Shanghai Key Laboratory of Integrate Administration Technologies for Information Security under Grant No. AGK201703, Opening Project of Shanghai Key Laboratory of Scalable Computing and Systems, National Cryptography Development Fund, Fundamental Research Funds for the Central Universities, and Foundation of Science and Technology on Information Assurance Laboratory under Grant No. KJ-17-008.

References

[1] Z. Liu, X. Huang, S. Hu, M. K. Khan, H. Seo and L. Zhou, "On emerging family of elliptic curves to secure Internet of Things: ECC comes of age", in *IEEE T DEPEND SECURE*, vol. 14, no, 3, pp. 237-248, 2017.

[2] J. Hamblen and G. Bekkum, "An embedded systems laboratory to support rapid prototyping of robotics and the Internet of Things," in *IEEE T EDUC*, vol. 56, no. 1, pp. 121-128, 2013.

[3] D. Guinard, V. Trifa, S. Karnouskos, P. Spiess and D. Savio, "Interacting with the SOA-based Internet of Things: discovery, query, selection, and on-demand provisioning of web services," in *IEEE T SERV COMPUT*, vol. 3, no. 3, pp. 223-235, 2010.

[4] D. Zhang, S. Zhao, L. T. Yang, M. Chen, Y. Wang and H. Liu, "NextMe: localization using cellular traces in Internet of Things," in *IEEE TRANS IND INFORM*, vol. 11, no. 2, pp. 302-312, 2015.

[5] C. Beierle, J. Jean, S. Kölbl, G. Leander, A. Moradi, T. Peyrin, Y. Sasaki, P. Sasdrich and S. M. Sim, "The SKINNY family of block ciphers and its low-latency variant MANTIS," in *Proc CRYPTO*, vol. 9815, pp. 123-153, 2016.

[6] R. Beaulieu, S. T. D. Shors, B. Weeks J. Smith and L. Wingers, "The SIMON and SPECK lightweight block ciphers," in *Proc. DAC*, vol. 52, pp. 175-180, 2015.

[7] A. Bogdanov, L. Knudsen, G. Leander, C. Paar, A. Poschmann, M. Robshaw, Y. Seurin and C. Vikkelsoe, "PRESENT: an ultra-lightweight block cipher," in *Proc. CHES*, vol. 4727, pp. 450-466, 2007.

[8] S. Banik, A. Bogdanov, T. Isobe, K. Shibutani, H. Hiwatari, T. Akishita and F. Regazzoni, "Midori: A block cipher for low energy," in *Proc. ASIACRYPT*, vol. 9543, pp. 411-436, 2015.

[9] D. Engels, M. J. O. Saarinen, P. Schweitzer and E. M. Smith, "The Hummingbird-2 lightweight authenticated encryption algorithm," in *Proc. RFIDSec*, vol. 7055, pp. 19-31, 2012.

[10] C. Lim and T. Korkishko, "mCrypton-a lightweight block cipher for security of low-cost RFID tags and sensors," in *Proc. WISA*, vol. 3786, pp. 243-258, 2006.

[11] D. Hong, J. Sung, S. Hong, J. Lim, S. Lee, B. S. Koo, C. Lee, D. Chang, J. Lee, K. Jeong, H. Kim, J. Kim and S. Chee, "HIGHT: a new block cipher suitable for low-resource device," in *Proc. CHES*, vol. 4249, pp. 46-59, 2006.

[12] J. Guo, T. Peyrin, A. Poschmann and M. Robshaw, "The LED block cipher," in *Proc. CHES*, vol. 6917, pp. 326-341, 2011.

[13] F. Mendel, V. Rijmen, D. Toz and K. Varici, "Differential analysis of the LED block cipher," in *Proc. ASIACRYPT*, vol. 7658, pp. 190-207, 2012.

[14] T. Isobe and K. Shibutani, "Security analysis of the lightweight block ciphers XTEA, LED and Piccolo," in *Proc. ACISP*, vol. 7372, pp. 71-86, 2012.

[15] I. Nikolić, L. Wang and S. Wu, "Cryptanalysis of round-reduced LED," in *Proc. FSE*, vol. 8424, pp. 112-129, 2014.

[16] H. Soleimany, "Probabilistic slide cryptanalysis and its applications to LED-64 and Zorro," in *Proc. FSE*, vol. 8540, pp. 373-389, 2014.
[17] K. Jeong and C. Lee, "Differential fault analysis on block cipher LED-64," in *FUTURE INF TECHNOL APPL SERV*, vol. 164, pp. 747-775, 2012.
[18] W. Li, D. Gu, X. Xia, C. Zhao, Z. Liu, Y. Liu and Q. Wang, "Single byte differential fault analysis on the LED lightweight cipher in the wireless sensor network," in *INT J COMP INTELL SYS*, vol. 5, no. 8, pp. 896-904, 2012.
[19] P. Jovanovic, M. Kreuzer and I. Polian, "A fault attack on the LED block cipher," in *Proc. COSADE*, vol. 7275, pp. 120-134, 2012.
[20] X. Zhao, S. Guo and F. Zhang, "Improving and evaluating differential fault analysis on LED with algebraic techniques," in *Proc. FDTC*, pp. 41-51, 2013.
[21] W. Li, W. Zhang, D. Gu, Q. Cao, Z. Tao, Z. Zhou, Y. Liu and Z. Liu, "Impossible differential fault analysis on the LED lightweight cryptosystem in the vehicular ad–hoc networks," in *IEEE T DEPEND SECURE*, vol. 13, no. 1, pp. 84-92, 2016.
[22] D. Boneh, R. A. DeMillo and R. J. Lipton, "On the importance of eliminating errors in cryptgraphic computations," in *J CRYPTOL*, vol. 14, no. 2, pp. 101-119, 2001.
[23] D. Boneh, R. A. DeMillo, R. J. Lipton and M. Yung, "On the importance of checking cryptographic protocols for faults," in *Proc. EUROCRYPTO*, vol 1233, pp. 37-51, 1997.
[24] P. Derbez, P.-A. Fouque and D. Lereateux, "Meet-in-the-middle and impossible differential fault analysis on AES," in *Proc. CHES*, vol. 6917, pp. 274-291, 2011.
[25] P. Dusart, G. Letourneux and O. Vivolo, "Differential fault analysis on A.E.S," in *Proc. ACNS*, vol. 2846, pp. 293-306, 2003.
[26] J. Blömer, and J. P. Seifert, "Fault based cryptanalysis of the advanced encryption standard (AES)," in *Proc. FC*, vol. 2846, pp. 162-181, 2003.
[27] T. Fuhr, E. Jaulmes, V. Lomné and A. Thillard, "Fault attacks on AES with faulty ciphertexts only," in *Proc. FDTC*, pp. 108-118, 2013.
[28] C. Dobraunig, M. Eichlseder, T. Korak, V. Lomneé and F. Mendel, "Statistical fault attacks on nonce-based authenticated encryption schemes," in *Proc. ASIACRYPT*, vol. 10031, pp. 369-395, 2016.

APPENDIX

TABLE 6
The probability of breaking LED using different distinguishers (%)

♯Faults	SEI	GF	GF-SEI	ML	HW	MAP
0	0.00	0.00	0.00	0.00	0.00	0.00
1	0.00	0.00	0.00	0.00	0.00	0.00
2	0.00	0.00	0.00	0.00	0.00	0.00
3	0.00	0.00	0.00	4.40	2.00	3.00
4	0.60	1.80	5.20	2.40	1.60	2.40
5	0.00	2.40	2.00	3.80	6.00	3.00
6	1.80	1.00	2.20	6.00	6.60	5.40
7	0.40	1.60	2.80	9.60	8.20	10.00
8	1.80	2.40	4.60	10.20	12.60	9.80
9	1.80	3.20	5.60	12.60	13.60	14.20
10	1.20	4.60	6.40	17.20	22.40	17.80
11	2.20	5.00	10.60	19.40	24.00	23.00
12	3.00	5.40	13.00	25.00	22.80	24.20
13	3.00	4.40	13.20	28.00	30.80	30.20
14	6.00	7.20	17.80	35.00	36.80	31.60
15	4.60	8.60	18.00	39.00	41.60	40.00
16	7.80	9.80	24.40	47.00	48.20	41.80
17	9.00	10.20	27.60	45.20	55.80	49.60
18	11.00	15.00	28.60	58.00	57.60	54.80
19	12.80	16.20	36.60	58.80	60.60	57.80
20	14.00	19.40	39.00	65.00	63.20	65.00
21	16.60	19.60	45.40	70.00	68.40	66.20
22	18.60	22.60	47.80	70.40	75.20	68.80
23	25.60	25.40	53.20	76.00	79.20	74.20
24	20.60	31.20	54.60	77.80	80.80	75.60
25	29.40	33.40	62.00	80.60	85.80	78.40
26	29.80	34.00	65.60	82.00	86.20	82.80
27	31.60	31.20	63.20	86.00	85.60	88.60
28	35.80	41.40	69.00	87.60	90.60	86.20
29	41.40	40.40	72.80	89.20	91.80	88.00
30	41.40	42.60	77.40	92.20	94.20	90.00
31	43.20	47.60	79.20	92.20	95.20	93.40
32	46.60	50.20	84.60	91.80	94.80	94.80
33	46.80	55.00	86.80	94.00	96.60	94.60
34	53.80	60.80	86.40	95.60	96.40	95.80
35	56.40	65.40	89.80	96.80	96.20	96.00
36	53.80	63.80	91.60	97.80	98.00	97.60
37	62.60	72.40	90.40	97.80	98.20	97.00
38	62.60	68.60	92.00	98.00	97.80	99.00
39	67.00	76.40	92.60	98.00	98.60	98.40
40	72.00	77.40	94.40	98.60	98.80	98.60
41	69.40	75.80	94.60	99.20	99.80	99.40
42	74.40	80.60	94.60	98.60	99.60	98.80
43	74.40	80.80	96.20	99.20	99.80	99.20
44	76.60	85.60	95.20	99.20	100.00	99.60
45	79.80	87.80	97.00	99.20	99.60	99.60
46	79.00	87.80	97.40	100.00	99.60	100.00
47	82.00	88.20	97.60	99.40	100.00	99.60
48	83.00	88.40	98.40	99.60	99.60	99.60
49	85.40	90.80	98.20	99.80	99.80	100.00
50	87.00	91.40	98.80	100.00	100.00	99.40
51	90.40	92.20	98.80	100.00	99.60	99.80
52	90.40	93.00	98.20	99.80	99.80	100.00
53	90.20	94.60	98.60	99.80	100.00	100.00
54	89.40	94.80	98.60	100.00	99.80	100.00
55	93.00	93.80	98.60	100.00	100.00	100.00
56	93.20	96.00	98.40	100.00	100.00	100.00
57	93.80	96.40	99.20	100.00	100.00	100.00
58	93.20	96.20	99.00	100.00	100.00	100.00
59	96.20	97.40	99.00	100.00	100.00	100.00
60	97.00	97.40	99.40	100.00	100.00	100.00
61	95.00	99.80	99.20	100.00	99.80	100.00
62	96.80	99.20	99.20	100.00	100.00	100.00
63	97.00	98.20	99.20	100.00	100.00	99.80
64	97.60	98.80	99.80	100.00	100.00	100.00
65	97.80	99.00	100.00	100.00	100.00	100.00
66	98.60	99.00	99.60	100.00	100.00	100.00
67	98.80	98.80	99.80	100.00	100.00	100.00
68	98.20	99.80	99.80	100.00	100.00	100.00
69	99.60	99.20	100.00	100.00	100.00	100.00
70	99.40	99.80	99.80	100.00	100.00	100.00

TABLE 7
The latency in breaking LED using different distinguishers (seconds)

♯Faults	SEI	GF	GF-SEI	ML	HW	MAP
0	0.000	0.000	0.000	0.000	0.000	0.000
1	0.037	0.042	0.049	0.039	0.039	0.043
2	0.068	0.074	0.081	0.073	0.072	0.078
3	0.102	0.107	0.115	0.106	0.105	0.121
4	0.134	0.139	0.148	0.137	0.136	0.159
5	0.170	0.174	0.183	0.170	0.167	0.195
6	0.201	0.205	0.215	0.205	0.202	0.249
7	0.239	0.246	0.254	0.239	0.236	0.289
8	0.270	0.276	0.286	0.274	0.270	0.321
9	0.305	0.308	0.318	0.310	0.307	0.371
10	0.345	0.350	0.360	0.347	0.343	0.409
11	0.382	0.385	0.397	0.382	0.378	0.449
12	0.416	0.422	0.431	0.417	0.413	0.488
13	0.447	0.452	0.463	0.452	0.448	0.540
14	0.488	0.495	0.504	0.489	0.483	0.577
15	0.526	0.533	0.542	0.523	0.517	0.619
16	0.559	0.570	0.580	0.561	0.555	0.666
17	0.601	0.611	0.618	0.607	0.601	0.697
18	0.633	0.641	0.649	0.689	0.682	0.732
19	0.670	0.678	0.685	0.726	0.719	0.774
20	0.710	0.722	0.726	0.767	0.759	0.817
21	0.745	0.757	0.761	0.758	0.749	0.860
22	0.784	0.797	0.801	0.793	0.783	0.904
23	0.823	0.835	0.837	0.838	0.828	0.936
24	0.855	0.868	0.867	0.871	0.858	0.927
25	0.899	0.912	0.913	0.896	0.885	0.970
26	0.931	0.948	0.948	0.958	0.946	1.006
27	0.970	0.987	0.988	1.004	0.990	1.052
28	1.010	1.028	1.027	1.013	1.002	1.086
29	1.044	1.060	1.058	1.047	1.032	1.134
30	1.085	1.103	1.099	1.090	1.076	1.163
31	1.123	1.144	1.139	1.146	1.126	1.214
32	1.159	1.182	1.178	1.172	1.156	1.261
33	1.196	1.219	1.216	1.222	1.209	1.288
34	1.238	1.259	1.250	1.268	1.250	1.343
35	1.279	1.299	1.295	1.297	1.278	1.379
36	1.310	1.337	1.330	1.338	1.321	1.425
37	1.349	1.371	1.364	1.373	1.358	1.455
38	1.389	1.417	1.406	1.405	1.387	1.509
39	1.423	1.451	1.444	1.453	1.433	1.547
40	1.461	1.490	1.482	1.495	1.473	1.587
41	1.497	1.529	1.516	1.486	1.465	1.628
42	1.541	1.576	1.566	1.523	1.502	1.665
43	1.572	1.601	1.592	1.561	1.537	1.711
44	1.617	1.647	1.637	1.601	1.578	1.756
45	1.657	1.689	1.677	1.636	1.615	1.792
46	1.686	1.720	1.709	1.677	1.651	1.830
47	1.726	1.761	1.752	1.716	1.686	1.866
48	1.769	1.806	1.790	1.763	1.738	1.905
49	1.794	1.834	1.821	1.788	1.763	1.951
50	1.850	1.891	1.878	1.828	1.797	1.992
51	1.882	1.924	1.909	1.867	1.836	1.926
52	1.931	1.970	1.955	1.901	1.871	1.993
53	1.959	2.007	1.987	1.944	1.911	2.015
54	2.001	2.046	2.029	1.978	1.947	2.079
55	2.049	2.092	2.078	2.019	1.987	2.143
56	2.074	2.114	2.102	2.056	2.025	2.166
57	2.125	2.172	2.154	2.094	2.061	2.206
58	2.154	2.207	2.184	2.132	2.103	2.263
59	2.196	2.246	2.226	2.172	2.136	2.327
60	2.236	2.286	2.265	2.214	2.176	2.359
61	2.278	2.333	2.312	2.248	2.212	2.414
62	2.293	2.343	2.326	2.289	2.250	2.570
63	2.343	2.397	2.376	2.327	2.289	2.487
64	2.370	2.427	2.409	2.363	2.328	2.511
65	2.404	2.453	2.449	2.404	2.367	2.546
66	2.451	2.502	2.493	2.440	2.407	2.570
67	2.486	2.533	2.532	2.485	2.453	2.609
68	2.519	2.576	2.572	2.523	2.482	2.648
69	2.566	2.613	2.614	2.549	2.518	2.693
70	2.603	2.653	2.652	2.598	2.560	2.734