DOI:10.7544/issn1000-1239.2018.20180437
55(10):2174-2184,2018

LBlock 轻量级密码算法的唯密文故障分析

李 玮[1,2,3,4] 吴益鑫[1] 谷大武[2] 曹 珊[1] 廖林峰[1] 孙 莉[1] 刘 亚[5] 刘志强[2]

[1](东华大学计算机科学与技术学院 上海 201620)

[2](上海交通大学计算机科学与工程系 上海 200240)

[3](上海市可扩展计算与系统重点实验室(上海交通大学) 上海 200240)

[4](上海市信息安全综合管理技术研究重点实验室(上海交通大学) 上海 200240)

[5](上海理工大学计算机科学与工程系 上海 200093)

(liwei.cs.cn@gmail.com)

Ciphertext-Only Fault Analysis of the LBlock Lightweight Cipher

Li Wei[1,2,3,4], Wu Yixin[1], Gu Dawu[2], Cao Shan[1], Liao Linfeng[1], Sun Li[1], Liu Ya[5], and Liu Zhiqiang[2]

[1](*School of Computer Science and Technology, Donghua University, Shanghai* 201620)

[2](*Department of Computer Science and Engineering, Shanghai Jiao Tong University, Shanghai* 200240)

[3](*Shanghai Key Laboratory of Scalable Computing and Systems (Shanghai Jiao Tong University), Shanghai* 200240)

[4](*Shanghai Key Laboratory of Integrate Administration Technologies for Information Security (Shanghai Jiao Tong University), Shanghai* 200240)

[5](*Department of Computer Science and Engineering, University of Shanghai for Science and Technology, Shanghai* 200093)

Abstract The lightweight cipher LBlock was proposed at ANCS in 2011. It has the structure of Feistel and is widely applied in the security of Internet of things (IoT). In this paper, a cipher-text fault analysis for LBlock cipher by injecting faults is proposed, and it is analyzed by 6 distinguishers in the last but 3 rounds. On the basis of original distinguishers as SEI, GF, GF-SEI, MLE, we propose GF-MLE and MLE-SEI distinguishers as new distinguishers. The simulation experiments show that the secret key can be recovered with over 99% success probability in a short period of time, and these two new distinguishers can not only improve the attacking efficiency, but also decrease the number of faults. This shows that the ciphertext-only fault analysis poses a great threat to the security of LBlock cipher.

Key words lightweight cipher; LBlock; ciphertext-only fault analysis; Internet of things (IoT); cryptanalysis

摘 要 LBlock 算法是在 2011 年 ANCS 会议上提出来的一种轻量级分组密码算法.它是一种具有 Feistel 结构的典型密码,并且广泛应用于物联网安全中.提出了针对 Feistel 结构的 LBlock 密码算法的新型唯密文故障分析方法,通过在算法的倒数第 4 轮导入故障,分别使用 6 种区分器对算法进行分析.在原有的 SEI 区分器、GF 区分器、GF-SEI 双重区分器、MLE 区分器基础上,提出了 GF-MLE 双重区分器和 MLE-SEI 双重区分器作为新型区分器.仿真实验结果表明:可以在较短的时间内使用较少的故障

收稿日期:2018-06-12;**修回日期**:2018-08-06

基金项目:国家自然科学基金项目(61772129);国家密码发展基金项目(MMJJ20180101)

This work was supported by the National Natural Science Foundation of China (61772129) and the National Cryptography Development Fund (MMJJ20180101).

通信作者:孙莉(sli@dhu.edu.cn)

数且以 99%的成功概率恢复出主密钥并破译算法,其中提出的 2 种新型区分器比原有区分器所需故障数更少、效率更高.由此说明唯密文故障攻击对 LBlock 算法的安全性构成了巨大的威胁.

关键词　轻量级密码;LBlock;唯密文故障攻击;物联网;密码分析

中图法分类号　TP309.7

随着信息技术与计算机技术的不断发展,物联网正在逐渐深入到人们各个领域中.物联网将所有物品通过射频识别和红外感应器等信息传感设备与互联网连接起来,将人、数据与实体相联系,实现了人与物、物与物之间的信息交换与通信,以实现遍及智慧城市、智能家居、环境保护、智能交通和个人健康的智能化识别与管理[1-3].如图 1 所示.轻量级分组密码作为实现消息保密、数据完整性保护和实体认证的核心体制,其设计、分析与实现方法一直是密码学研究的主流,在物联网的安全领域发挥着重要的应用[4].

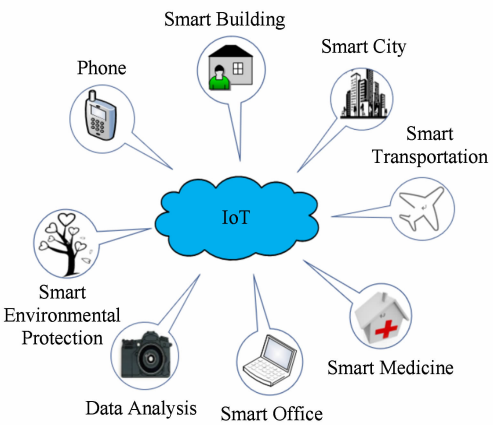

Fig. 1　The IoT scenario

图 1　物联网概况图

LBlock 算法是一种 Feistel 结构的轻量级分组密码算法,具有优良的软硬件实现效率,以及在处理器应用上具有良好的实现性能[5-6],因而可应用于 RFID 标签和其他低资源设备以保护物联网的安全.前人针对 LBlock 算法的安全性分析主要致力于传统密码分析,包括不可能差分分析、相关密钥不可能差分分析、截断差分分析、零相关线性分析、积分分析、Biclique 分析和侧信道立方分析等多种密码分析方法[7-9].然而,在物联网环境中,单纯从 LBlock 算法的算法结构研究安全性已经远远不够,攻击者不仅可以进行传统密码分析,而且可以通过电路剖析和软件逆向分析等方式获取密码算法的硬件结构和内部编码方法,借助异常时钟、异常电流、微波辐射、激光照射、涡流磁场等方式干预密码变换的正常过程使其出错,并有可能实现比传统密码分析更有效地破译,从而实现密码算法内数据的复制、篡改或伪造.人们通常把这种攻击称为"故障分析"[10-19].由于其成功的攻击效果和潜在的发展前景,已经引起了国内外从事密码学和微电子领域的研究学者的极大关注,并成为密码工程和密码分析领域发展最为迅速的方向之一.

故障分析在其自身的发展中,逐步衍生出多种攻击方法.1997 年,由 Biham 和 Shamir 提出的差分故障分析是目前分析范围最广、威胁最大且发展最迅速的一种攻击技术,被广泛应用于密码算法的安全性分析中[20].后来,无效故障分析、碰撞故障分析、不可能故障分析、代数故障分析、中间相遇故障分析、线性故障分析以及唯密文故障攻击等应运而生.它们充分结合了传统密码分析技术以及软硬件实现,具备独特的分析优势,不仅扩大了攻击面,而且提升了威胁性,在面向现实的运行环境下具有更加灵活的应用前景.

在故障分析中,唯密文故障攻击是目前唯一只需要知道错误密文就能进行攻击的技术,在唯密文攻击(ciphertext-only attack, COA)假设下,攻击者通过导入故障得到错误密文,解密密文之后得到中间状态,分析得到的中间状态的特性,就能以低的时间复杂度和数据复杂度推导出正确的密钥.目前,国内外的主要研究是基于代换置换网络(substitution permutation network, SPN)结构的唯密文故障分析方法.那么,具有 Feistel 结构的典型密码算法是否能够抵抗唯密文故障分析,已引起国内外研究学者的广泛关注.由此,本文提出了一种针对 LBlock 算法的新型唯密文故障分析方法,在较短时间内以较少故障就能够恢复主密钥,为物联网的安全提供了保障.

1　相关工作

Boneh 等学者于 1997 年利用随机故障成功破译 RSA 算法,此后故障分析在评测密码体制安全性的方法中占据了重要的位置[21].国内外研究学者自 2012 年起针对 LBlock 算法相继提出了故障分析

方法. 2012 年,Zhao 等学者首次使用差分故障攻击(differential fault analysis, DFA)对 LBlock 算法进行攻击,在随机单比特故障模型下,通过在加密算法第 25～31 轮导入 7 次故障,成功恢复倒数 3 轮的子密钥,进而恢复了主密钥[22]. 2013 年,Jeong 等学者对无线传感网络环境下的 LBlock 算法提出了改进的差分故障攻击方法,在随机半字节故障模型下,通过在算法的第 30 轮导入 7 次故障成功恢复了主密钥,同时在算法的第 29 轮导入 5 次故障也成功恢复主密钥,进一步减少了导入故障数[23]. 2018 年,Wei 等学者对 LBlock 算法提出了改进的差分故障攻击方法,在随机半字节故障模型下,通过在算法的第 27～29 轮导入 13.3 次故障,成功恢复主密钥,扩展了该方法的攻击轮数[24]. 2013 年,Chen 等学者针对 LBlock 算法提出了积分故障分析(integral fault attack, IFA),在随机半字节故障模型下,通过在算法的第 24 轮结束时导入 120 次故障成功恢复了主密钥,同时也提出了在半随机的半字节故障模型下,通过在加密的第 23 轮结束时在算法的右分支导入 32 次故障成功恢复了主密钥. 该方法成功实现了对算法更深轮数进行攻击,且能够确定故障导入的具体位置[25]. 上述方法均属于选择明文攻击(chosen-plaintext attack, CPA),而本文针对 LBlock 提出的唯密文故障攻击(Ciphertext-only Fault Analysis, CFA)是唯一一种在只知道错误密文的情况下就能对算法进行攻击的方法. 表 1 列出了现有的针对 LBlock 算法的故障分析方法的对比.

2013 年,Fuhr 等学者基于全零字节模型、半字节模型和随机字节模型,针对 AES 算法首次提出了唯密文故障攻击方法[26]. 该方法在算法运行时导入随机故障,使算法执行某些错误的操作,并产生错误的密文,解密密文得到中间状态,利用统计学的知识统计中间状态的分布律,从而恢复最后一轮子密钥. 重复上述过程,即可恢复更多的子密钥,直至破译原始密钥. 针对 AES 算法,在随机字节故障模型下,将故障导入在倒数第 2 轮的指定位置并用平方欧式距离 SEI 和极大似然估计 MLE 作为区分器,分别以 320 个故障和 224 个故障且 99%的成功概率恢复出了 AES 算法的主密钥. 2017 年,Li 等学者针对 LED 算法提出了新型的唯密文故障分析,除了 Fuhr 等人提出的 SEI 区分器之外,还提出了拟合优度检验 GF 作为新型区分器,又构建了 GF-SEI 双重区分器,以更少的故障数成功求得了 LED 的密钥,提高了故障攻击的效率[27]. 目前发表的唯密文故障分析都是针对 SPN 结构的加密算法. 因而,Feistel 结构密码算法的安全性是目前唯密文故障分析研究的热点.

Table 1 Comparison of Fault Analysis on LBlock

表 1 针对 LBlock 算法的故障分析对比

Type	Assumption	Model	Reference
DFA	CPA	Bit/Nibble	Ref [22-24]
IFA	CPA	Nibble	Ref [25]
CFA	COA	Nibble	This paper

本文首次提出了针对 Feistel 结构的 LBlock 算法的新型唯密文故障分析,并且以更深轮数对算法进行攻击,实现了故障导入在倒数第 4 轮,在原有 SEI 区分器、GF 区分器、MLE 区分器和 GF-SEI 双重区分器的基础上,提出了 GF-MLE 双重区分器和 MLE-SEI 双重区分器统计分析中间状态的分布律,以更少的故障数且 99%的成功率恢复了 LBlock 算法 80 比特的密钥,提高了故障攻击的效率并减少了故障攻击数. 表 2 为统计故障分析 AES 算法、LED 算法和 LBlock 算法的唯密文攻击结果对比. 其中,Byte 和 Nibble 分别表示字节和半字节故障模型,r 表示算法的总轮数. 攻击结果总结如下:

Table 2 Comparison of Ciphertext-Only Fault Attacks on a Subkey of AES, LED and LBlock

表 2 LBlock, LED 和 AES 算法唯密文故障分析部分子密钥结果对比

Distinguisher	SPN						Feistel		
	AES-128			LED-128			LBlock-80		
	Fault Model	Round	# Faults	Fault Model	Round	# Faults	Fault Model	Round	# Faults
SEI	Byte	$r-1$	320	Nibble	$r-1$	136	Nibble	$r-3$	51
GF				Nibble	$r-1$	128	Nibble	$r-3$	47
GF-SEI				Nibble	$r-1$	96	Nibble	$r-3$	36
MLE	Byte	r	224				Nibble	$r-3$	29
GF-MLE							Nibble	$r-3$	28
MLE-SEI							Nibble	$r-3$	22

1) 唯密文攻击不仅适用于 SPN 结构的密码算法,对于 Feistel 结构的密码算法同样可以进行攻击.

2) GF-MLE 双重区分器与 MLE-SEI 双重区分器减少了故障数,提高了攻击效率.

3) 使用这 6 种区分器进行攻击的轮数在倒数第 4 轮.

本文首先对唯密文故障攻击的研究现状进行了探讨,接着详细介绍了 LBlock 算法,然后给出了故障模型与假设,详述唯密文故障攻击过程,最后分析攻击的实验结果并对本文进行总结,附录给出了实验的全部数据.

2　LBlock 算法

2.1　术语说明

记 $P=X_1 \| X_0$ 为消息明文,其中 $P \in F_{2^8}^{16}$,X_1 为消息明文的左分支,X_0 为消息明文的右分支. $C=X_{32} \| X_{33}$ 为正确密文,$C^*=X_{32}^* \| X_{33}^*$ 为错误密文,其中 $C \in F_{2^8}^{16}$,$C^* \in F_{2^8}^{16}$,l 为算法轮数,$K \in F_{2^8}^{16}$ 为主密钥,$ek_d \in F_{2^8}^{16}$ 为主密钥 K 生成的第 d 轮子密钥,其中 $0 \leqslant d \leqslant l-1$. X_d 为第 d 轮的左分支,也就是第 $d+1$ 轮的右分支. $\lll 8$ 为左移 8 位.

记 F 为轮算法,SL 和 DL 分别表示 S 盒混淆层和 P 扩散层. SL^{-1} 和 DL^{-1} 分别表示 S 盒混淆层的逆运算和 P 置换的逆运算.

2.2　LBlock 密码算法

LBlock 密码算法是一种具有 Feistel 结构的迭代型分组密码,分组长度为 64 b,主密钥长度为 80 b. 迭代轮数为 32 轮. 在加密算法中将消息明文分为 32 b 等长的左右 2 个分支.

该算法由 3 部分组成,分别为加密、解密和密钥编排方案. 其中,解密过程是加密过程的逆. 下面介绍算法的加密过程. 算法结构如图 2 所示.

LBlock 算法前 31 轮都是左右交换操作,最后一轮不包含左右分支交换操作,每一轮变换之后的状态表示为

$$X_i=F(X_{i-1},ek_{i-1}) \oplus (X_{i-2} \lll 8),$$

其中,$2 \leqslant i \leqslant 33$. 在 F 轮函数中,首先将左分支与轮密钥进行异或运算. 其次将左分支的每半个字节经过 8 个 S 盒进行置换变成另外一个半字节. 之后将经过 S 混淆层之后的状态按照特定的交换规则,变换到不同的位置. 最后密文输出为 $C=X_{32} \| X_{33}$.

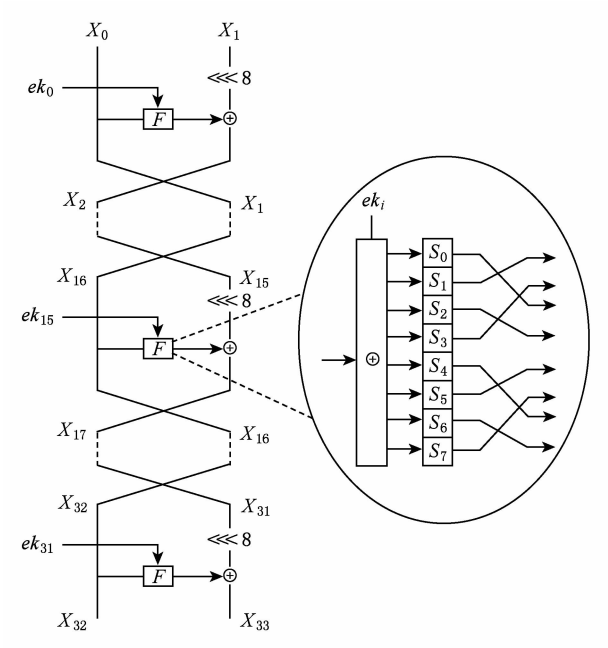

Fig. 2　The structure of LBlock

图 2　LBlock 算法的结构

2.3　密钥编排方案

密钥编排由 3 个步骤组成:

1) 选取 80 b 的主密钥 $K(ek_{79},ek_{78},\cdots,ek_0)$ 的左侧 32 b 密钥($ek_{79}ek_{78}\cdots ek_{47}ek_{46}$)作为第 1 轮加密的轮密钥 ek_0.

2) 对 80 b 的主密钥进行循环左移 29 b 操作,将得到的 80 b 作为新的主密钥.

3) 部分密钥位更新,更新规则如下:

$$[ek_{79}ek_{78}ek_{77}ek_{76}]=s_9[ek_{79}ek_{78}ek_{77}ek_{76}],$$

$$[ek_{75}ek_{74}ek_{73}ek_{72}]=s_8[ek_{75}ek_{74}ek_{73}ek_{72}],$$

$$[ek_{50}ek_{49}ek_{48}ek_{47}ek_{46}]=$$

$$[ek_{50}ek_{49}ek_{48}ek_{47}ek_{46}] \oplus [i]_2,$$

其中,s_8,s_9 均为 4 位 S 盒.

3　唯密文故障分析

3.1　故障模型和基本步骤

本文采用的唯密文故障分析的基本假设为:对随机明文采用同一个密钥进行加密,分析人员可以获得的是密码算法的任意错误密文.

LBlock 算法中采用的 S 盒输入与输出均为4 b,因此本文采用随机半字节故障模型,诱发某一轮的存储单元发生 4 b 的随机错误. 在本节中,导入故障轮数为倒数第 4 轮.

唯密文故障分析的基本步骤如下:

1) 选择随机明文并使用同一个密钥对其进行加密. 当 LBlock 算法的加密过程运行到倒数第 4 轮

时,导入随机故障,从而获得一组错误的密文.其中,故障导入既可以通过控制硬件实现中的异常时钟、异常电流、微波辐射、激光照射、涡流磁场等手段对真实的密码硬件实现来实现,也可以通过修改程序等软件模拟技术的方式来实现.

2) 解密错误密文得到中间状态.找到中间状态、子密钥和错误密文之间的关系,用关系式正确表达出来.

3) 利用构造的区分器统计中间状态值的分布律.从而恢复倒数 3 轮的子密钥中的部分位.因为在故障导入之后,加密过程中的某些中间状态值会呈现非均匀分布.

4) 重复上述过程,可以恢复倒数 3 轮的子密钥的全部位,最后结合密钥编排方案直至破译原始密钥.

3.2 具体过程

本文首次提出了用唯密文故障分析的方法对 Feistel 结构算法进行破译.针对 LBlock 算法,本文在前人提出的 SEI 区分器、MLE 区分器、GF 区分器和 GF-SEI 双重区分器的基础上又提出了 2 种新型区分器用于唯密文故障分析.具体过程如下:

1) 使用同一个密钥对随机明文进行加密,在运算过程中的某一轮中导入随机半字节故障,故障导入的位置在倒数第 4 轮的左分支上 8 个半字节中的任意半个字节,根据故障导入的位置不同,则扩散路径也各不同,从而根据不同的故障扩散路径求得子密钥的不同的比特值.如图 3 所示,故障可以导入在 X_{29} 的任意半个字节,图中阴影表示受故障影响的半字节.图 3 表示故障导入在 X_{29} 的第 1 个半字节后的故障扩散路径.

2) 在运算过程中,中间状态 X_{29} 的每个半字节都能用子密钥 ek_{31} 的 2 个半字节、ek_{30} 的 1 个半字节、ek_{29} 的 1 个半字节和错误密文的 4 个半字节来表示:

$$\begin{cases} X_{31}=[DL(SL(X_{32}^{*},ek_{31}))\oplus X_{33}^{*}]\ggg 8 \\ X_{30}=[DL(SL(X_{31},ek_{30}))\oplus X_{32}^{*}]\ggg 8, \\ X_{29}=[DL(SL(X_{30},ek_{29}))\oplus X_{31}]\ggg 8 \end{cases}$$

其中,X_{32}^{*} 和 X_{33}^{*} 是已知的,所以通过该公式可以将中间状态 X_{29} 表示出来.

3) 通过区分器对中间状态进行分析,计算出 ek_{29} 的 1 个半字节、ek_{30} 的 1 个半字节、ek_{31} 的 2 个半字节.对于每一个子密钥的候选值,根据导入的故障数都能求出一组中间状态值.选择一个区分器,以这组数据为样本,计算出每个密钥候选值所对应的区分器的值,经过比较,区分器值最大或最小所对应的候选值,即为正确子密钥.

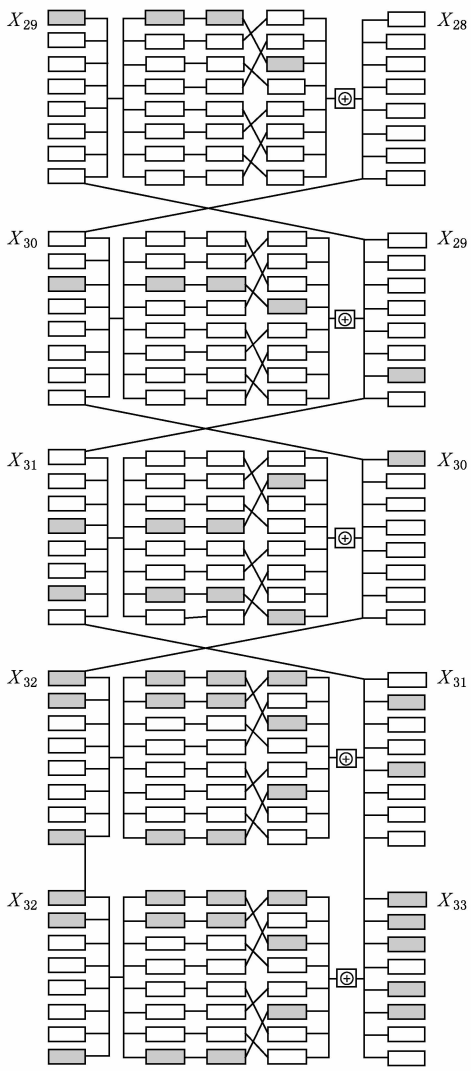

Fig. 3 Fault diffusion path in the fifth countdown round

图 3 故障导入在倒数第 4 轮后的扩散路径

4) 将上述过程重复 4 次,即可恢复出子密钥 ek_{31} 的 8 个半字节、ek_{30} 的 4 个半字节、ek_{29} 的 4 个半字节.上述过程重复 8 次,即可恢复出子密钥 ek_{29}、ek_{30} 和 ek_{31} 的全部位,之后可通过密钥编排方案求出主密钥.

3.3 区分器介绍

在 3.2 节的步骤 3 中所使用的区分器一共有 6 种,前 4 种是以前学者所提出的,后 2 种是本文所提出的新型区分器.

1) 平方欧氏距离(square euclidean imbalance, SEI)区分器

SEI 区分器用于计算未知分布到均匀分布的距离,其中越不服从均匀分布的那一组样本值所对应

的密钥候选值,即为所求的子密钥的部分位的解. 表达式为

$$SEI = \sum_{m=0}^{M-1}\left(\frac{\gamma[m]}{N}-\frac{1}{M}\right)^2,$$

其中,M 表示半个字节的所有可能值的个数,因此 $M=16$. $\gamma[m]$表示实际得到的中间状态半字节值为 m 的样本个数;N 为导入的故障数;每个密钥候选值都可以通过中间状态呈现的不同的分布律求出对应的 SEI 值. 计算故障导入之后实际得到中间状态半字节值的分布与理论半字节值均匀分布的距离,即为当前样本的 SEI 值. 当得到的 SEI 数值越大,表示这组样本越不服从均匀分布. 因此,攻击者可以计算 SEI 的最大值来求出正确的候选值.

2) 拟合优度(goodness of fit, GF)区分器

GF 区分器的使用条件为已知样本的分布律. 攻击者通过比较得到的样本分布律是否满足所讨论模型下的理论分布律. 若满足,则可求出子密钥的部分位. 其中,4 b 故障以按位与方式导入后,分布律如图 4 所示,并在 GF 区分器中使用该分布律. 区分器表达式为

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

其中,M 表示半个字节的所有可能值的个数;O_m 表示实际得到的值为 m 的样本个数;H_m 表示在样本总数相同的情况下,值为 m 中的理论个数. 通过区分器表达式可求出样本分布律是否满足理论分布律. 其中,$M=16$,GF 所求出的值越小越满足分布,即正确候选值.

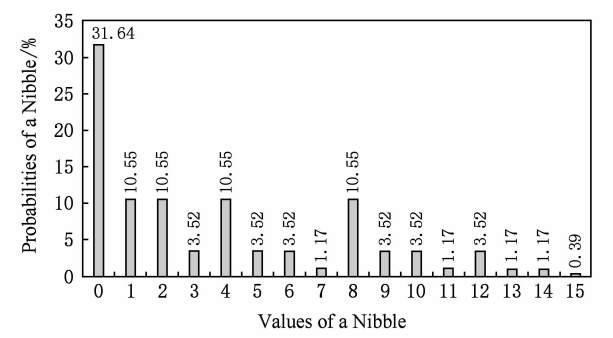

Fig. 4　The bitwise-AND distribution of a nibble after fault injections

图 4　半字节故障按位与导入后的分布律

3) GF-SEI 双重区分器

GF-SEI 双重区分器结合了 GF 区分器与 SEI 区分器的优点,先用拟合优度检验筛选出满足分布的样本. 再用 SEI 区分器从中寻找最优样本,从而恢复一定的密钥比特. 该方法进一步减少了故障数,提高了故障攻击的效率.

4) 极大似然估计(maximum likelihood estimate, MLE)区分器

MLE 区分器作为唯密文故障分析的区分器之一适用于已知样本满足某种概率分布的情况下,攻击者计算得到样本出现的概率,当概率最大时,即为所求的子密钥:

$$MLE = \prod_{n=0}^{N-1}P(\widetilde{\Psi}_{l-3}),$$

其中,N 表示故障数,$0\leqslant n\leqslant N-1$;$\widetilde{\Psi}_{l-3}$ 表示倒数第 4 轮错误的中间状态值;P 表示中间状态值出现的概率. 概率分布如图 4 所示. 若计算结果越大,则表示该组样本越接近实际概率.

5) GF-MLE 区分器

为了减少故障,本文将 GF 区分器与 MLE 区分器相结合,构建了一个 GF-MLE 双重区分器,先用拟合优度检验筛选出符合中间状态值分布的所有样本组. 再用 MLE 区分器从中选出更优样本,即为正确子密钥:

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

$$GF(k')\leqslant\chi_\alpha^2,$$

其中,k 表示子密钥候选值,$0\leqslant k\leqslant 2^{16}$,$m$ 表示每个候选值样本中所对应的所有中间状态值. 求出每组样本对应的 GF 的值之后,我们可以通过自由度 df 查找上侧位卡方分布表 χ^2 中对应的临界值 χ_α^2,规定当 $\chi^2\leqslant\chi_\alpha^2$ 时,样本服从该已知分布. 因而本文中 $M=16$,$df=M-1=15$,使用 GF 区分器时的样本数量 $N=50$ 并且 $H_m\geqslant 15$ 时,N 越大,恢复子密钥的成功率越高. 如果 $GF>\chi_\alpha^2$,则该候选值不是正确的子密钥的值;攻击者通过 GF 区分器筛选出满足 $GF\leqslant\chi_\alpha^2$ 的候选值 k',其中,$k'\subseteq k$. 之后用 MLE 区分器从中寻找最优样本,从而恢复一定的密钥位.

$$MLE(k') = \prod_{n=0}^{N-1}P(\widetilde{\Psi}_{l-3}),$$

$$RK = \max(MLE(k')),$$

其中,k' 表示 GF 区分器筛选之后的候选值,$MLE(k')$表示每组样本对应的 MLE 值;RK 表示正确的候选值. 从中选取 MLE 区分器最大的值,就是正确的子密钥.

6) MLE-SEI 区分器

为了更进一步减少故障数,本文构建了一个 MLE-SEI 双重区分器,将 MLE 区分器与 SEI 区分器结合,先用 MLE 区分器筛选出符合中间状态值

分布的一部分样本,再用 SEI 区分器选出更优样本:

$$MLE(k)=\prod_{n=0}^{N-1}P(\widetilde{\Psi}_{l-3}),$$

$$k''=\max(MLE(k)),$$

同上,k 表示子密钥候选值,$0\leqslant k\leqslant 2^{16}$,$\widetilde{\Psi}_{l-3}$ 表示每个 k 所对应的所有中间状态值;k''表示筛选出符合中间状态值分布的一部分样本.

$$SEI(k'')=\sum_{m=0}^{M-1}\left(\frac{\gamma[m]}{N}-\frac{1}{M}\right)^2,$$

$$RK=\max(SEI(k'')),$$

其中,m 表示每个 k''所对应的所有中间状态值,$SEI(k'')$表示部分样本分别对应的 SEI 值;从中选取 SEI 区分器最大的值,就是正确的子密钥.

4 实验分析

本实验在普通 PC 机器(CPU 为 Inter Core I7-7700K,4.2 GHz 内存 16 GB)上使用 Java 语言编程进行算法的加解密实现和攻击操作,利用计算机软件来模拟故障的产生,从而得到错误密文,用编程实现从故障导入到唯密文攻击再到最后得到主密钥的全部过程. 本文一共进行了 1 000 次实验,记录了恢复 ek_{31},ek_{30}和 ek_{29}部分比特的 SEI 区分器、GF 区分器、MLE 区分器、GF-SEI 和 GF-MLE 双重区分器所需的故障数和耗费时间. 图 5 表示不同区分器导入不同的故障数所恢复正确密钥部分位的成功概率. 其中,横坐标轴表示导入的故障数,纵坐标轴表示恢复出正确子密钥的成功率. 图 5 中有 6 种不同线段,分别表示使用 SEI 区分器、GF 区分器、GF-SEI 双重区分器、MLE 区分器、GF-MLE 双重区分器和 MLE-SEI 双重区分器进行唯密文攻击的结果. 从实验结果可知,在半字节随机故障模型下,当以 99%的概率恢复出过程 3 中的子密钥时,SEI 区分器、GF 区分器、GF-SEI 双重区分器、MLE 区分器、GF-MLE 双重区分器和 MLE-SEI 双重区分器分别需要 51,47,36,29,28 和 22 个故障数. 由此分析,本文所提出的 2 种新型区分器可以减少攻击时使用的故障数,并且使用 MLE-SEI 双重区分器时故障数减少明显.

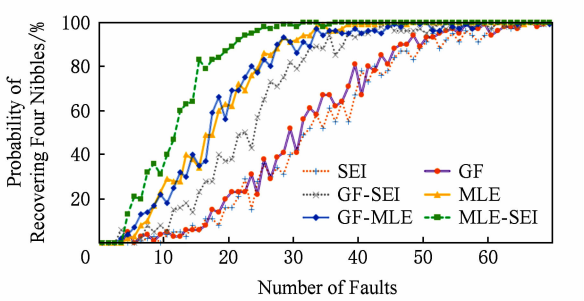

Fig. 5 The probability of recovering a partial subkey using different faults

图 5 不同故障数对应恢复出部分子密钥的概率

图 6 表示分别使用 SEI 区分器、GF 区分器、GF-SEI 双重区分器、MLE 区分器、GF-MLE 双重区分器和 MLE-SEI 双重区分器破译密钥时间消耗图. 其中纵坐标轴表示消耗时间,以秒为单位,横坐标表示导入故障数. 从实验结果可知,在半字节的随机故障模型下,6 个区分器以 99%的成功率破译密钥所需时间分别为 130.4 s,87.4 s,50.0 s,43.2 s,37.0 s 和 28.4 s. 不同的区分器所需时间相差较大,其中消耗时间最多的 GF 区分器,消耗时间是 MLE-SEI 消耗时间的 5 倍多. 从结果可知,本文中所提出的双重区分器并不会增加消耗的时间,反而减少了消耗时间.

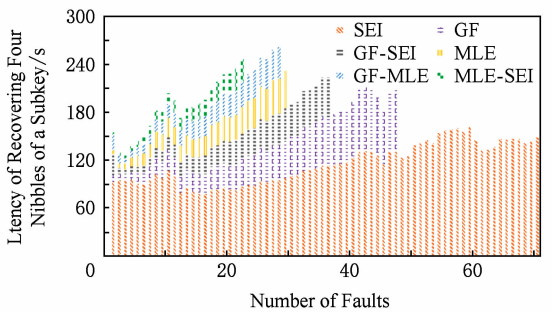

Fig. 6 The time shown with stacked charts of using different faults

图 6 不同故障数对应消耗时间的堆积柱形图

由图 5 和图 6 可得,从各个区分器对算法的破译情况进一步分析,GF 区分器的攻击效果比 SEI 区分器更佳,而 GF-SEI 双重区分器的攻击效果比 GF 和 SEI 区分器效果更佳,本文中所提出的 2 种新型区分器的攻击效果都比之前的 4 种区分器的攻击效果更佳. MLE-SEI 双重区分器所需要的导入故障数和耗费的时间比前面 5 种区分器均明显减少.

5 结束语

本文提出并讨论了 LBock 密码算法抗唯密文故障攻击的安全性. 研究结果表明:在面向半字节的故障模型中,与原有的 4 种区分器相比,本文所提出的 2 种新型区分器所需故障数更少,故障攻击效率

更高;以 LBlock 为代表的 Feistel 结构的密码算法易受到统计故障分析的威胁,此方法有助于优化唯密文故障攻击方法的攻击性能,提高故障攻击的效率和实用性.在下一步工作中,我们将进一步深入分析双重区分器的效率以及注入故障的分布律.

参 考 文 献

[1] Alioto M, Shahghasemi M. The Internet of things on its edge: Trends toward its tipping point [J]. IEEE Consumer Electronics Magazine. 2018, 7(1): 77-87

[2] Ning Huansheng, Liu Hong, Yang L T. Cyberentity security in the Internet of things [J]. IEEE Computer, 2013, 46(4): 46-43

[3] Mayer C P. Security and privacy challenges in the Internet of things [J]. Electronic Communications of the European Association of Software Science and Technology, 2009, 17 (3): 11-22

[4] Hong D, Sung J, Hong S, et al. HIGHT: A new block cipher suitable for low-resource device [G] //LNCS 4249: Proc of the 8th Int Workshop on Cryptographic Hardware and Embedded Systems. Berlin: Springer, 2006: 46-59

[5] Wu Wenling, Zhang Lei. LBlock: A lightweight block cipher [G] //LNCS 6715: Proc of the 9th Int Conf on Applied Cryptography and Network Security. Berlin: Springer, 2011: 327-344

[6] Kelsey J, Schneier B, Wagner D A, et al. Side channel cryptanalysis of product ciphers [G] //LNCS 1485: Proc of the 5th Int Conf on Computer Security. Berlin: Springer, 1998: 97-110

[7] Minier M, Naya-Plascencia M. A related key impossible differential attack against 22 rounds of the lightweight block cipher LBlock [J]. Information Processing Letters, 2012, 112(16): 624-629

[8] Soleimany H, Nyberg K. Zero-correlation linear cryptanalysis of reduced-round LBlock [J]. Designs, Codes and Cryptography, 2014, 73(3): 683-698

[9] Zhan Yingjie, Guan Jie, Ding Lin, et al. Related-key impossible differential attack on reduced round LBlock [J]. Journal of Electronic and Information Technology, 2012, 34 (9): 2161-2166 (in Chinese)
(詹英杰, 关杰, 丁林, 等. 对简化版 LBlock 算法的相关密钥不可能差分攻击[J]. 电子与信息学报, 2012, 34(9): 2161-2166)

[10] Boneh D, Demillo R A, Lipton R J. On the importance of checking cryptographic protocols for faults [G] //LNCS 1233: Proc of the 16th Int Conf on Theory and Application of Cryptographic Techniques. Berlin: Springer, 1997: 37-51

[11] Boneh D, Demillo R A, Lipton R J. On the importance of eliminating errors in cryptographic computations [J]. Journal of Cryptology, 2001, 14(2): 101-119

[12] Biehl I, Meyer B, Müller V. Differential fault attacks on elliptic curve cryptosystems [G] //LNCS 1880: Proc of the 20th Int Conf on Advances in Cryptology. Berlin: Springer, 2000: 131-146

[13] Fischer W, Reuter C A. Differential fault analysis on Grøstl [C] //Proc of the 7th Int Workshop on Fault Diagnosis and Tolerance in Cryptography. Piscataway, NJ: IEEE, 2012: 44-54

[14] Hemme L, Hoffmann L. Differential fault analysis on the SHA1 compression function [C] //Proc of the 6th Int Workshop on Fault Diagnosis and Tolerance in Cryptography. Piscataway, NJ: IEEE, 2011: 54-62

[15] Hoch J J, Shamir A. Fault analysis of stream ciphers [G] //LNCS 3156: Proc of the 6th Int Workshop on Cryptographic Hardware and Embedded Systems. Berlin: Springer, 2004: 240-253

[16] Cnudde T, Nikova S. Securing the PRESENT block cipher against combined side-channel analysis and fault attacks [J]. IEEE Trans on Very Large Scale Integration Systems, 2017, 25(12): 3291-3301

[17] Ghosh S, Saha D, Sengupta A, et al. Preventing fault attacks using fault randomisation with a case study on AES [J]. International Journal of Applied Cryptography, 2017, 3 (3): 225-235

[18] Wang An, Zhang Yu, Tian Weina, et al. Right or wrong collision rate analysis without profiling: Full-automatic collision fault attack [J]. Science China Information Sciences, 2018, 61(3): 032101:1-032101:11

[19] Breier J, Hou Xiaolu, Liu Yang. Fault attacks made easy: Differential fault analysis automation on assembly code [J]. IACR Trans on Cryptographic Hardware and Embedded Systems, 2018, 2018(2): 96-122

[20] Biham E, Shamir A. Differential fault analysis of secret key cryptosystems [G] //LNCS 1294: Proc of the 17th Int Conf on Advances in Cryptology. Berlin: Springer, 1997: 513-525

[21] Lin I C, Chang C C. Security enhancement for digital signature schemes with fault tolerance in RSA [J]. Information Sciences, 2007, 177(19): 4031-4039

[22] Zhao Liang, Nishide T, Sakurai K. Differential fault analysis of full LBlock [G] //LNCS 7275: Proc of the 3rd Int Workshop on Constructive Side-Channel Analysis and Secure Design. Berlin: Springer, 2012: 135-150

[23] Jeong K, Lee C, Lim J. Improved differential fault analysis on lightweight block cipher LBlock for wireless sensor networks [J]. EURASIP Journal on Wireless Communications and Networking, 2013, 2013(1): 151-159

[24] Wei Yuechuan, Rong Yisheng, Fan Cunyang. Differential fault attacks on lightweight cipher LBlock [J]. Fundamenta Informaticae, 2018, 157(1/2): 125-139

[25] Chen Hua, Fan Limin. Integral based fault attack on LBlock [G] //LNCS 8565: Proc of the 16th Int Conf on Information Security and Cryptology. Berlin: Springer, 2013: 227-240

[26] Fhur T, Jaulmes E, Lomne V, et al. Fault attacks on AES with faulty ciphertexts only [C] //Proc of the 8th Int Workshop on Fault Diagnosis and Tolerance in Cryptography. Piscataway, NJ: IEEE, 2013: 108-118

[27] Li Wei, Ge Chenyu, Gu Dawu, et al. Research on the LED lightweight cipher against the statistical fault analysis in Internet of Things [J]. Journal of Computer Research and Development, 2017, 54(10): 2205-2214 (in Chinese)
(李玮, 葛晨雨, 谷大武, 等. 物联网环境中 LED 轻量级密码算法的统计故障分析研究[J]. 计算机研究与发展, 2017, 54(10): 2205-2214)

Li Wei, born in 1980. PhD, professor. Senior member of CCF. Her main research interests include the design and analysis of symmetric ciphers.

Wu Yixin, born in 1995. Master candidate. Her main research interests include security analysis of block ciphers.

Gu Dawu, born in 1970. PhD, professor and PhD supervisor. His research interests cover cryptology and computer security.

Cao Shan, born in 1995. Master candidate. Her main research interests include security analysis of lightweight ciphers.

Liao Linfeng, born in 1993. Master candidate. His main research interests include security analysis of symmetric ciphers.

Sun Li, born in 1964. PhD, associate professor. Her main research interests include Internet of things.

Liu Ya, born in 1983. PhD, lecturer. Her main research interests include the design and analysis of symmetric ciphers and computational number theory.

Liu Zhiqiang, born in 1977. PhD, associate professor. His main research interests include cryptanalysis and design of block ciphers and hash functions.

附录 A. 实验数据及结果.

明文:随机生成.

密钥:01 23 45 67 89 AB CD EF FE DC

实验结果:6 种区分器均能恢复主密钥,实验数据如表 A1 和表 A2 所示.

Table A1　Probability of Breaking LBlock Using Different Distinguishers

表 A1　不同区分器破译 LBlock 的成功概率

Faults	Beaking Probablity/%					
	SEI	GF	GF-SEI	MLE	GF-MLE	MLE-SEI
0	0	0	0	0	0	0
1	0	0	0	0	0	0
2	0	0	0	0	0	0
3	0	0	0	0	0	0
4	0	0	0	2	6	0
5	0	5	2	4	2	13
6	0	0	3	7	0	21
7	1	3	8	13	2	20
8	2	4	10	14	8	32
9	3	1	17	17	5	36
10	0	4	23	22	8	31
11	4	5	29	18	5	40
12	5	3	28	25	15	47
13	5	3	28	32	16	60
14	3	6	40	30	18	63
15	8	6	38	40	14	64
16	6	6	34	35	23	83
17	11	8	49	37	28	79
18	11	15	49	59	28	83
19	8	14	60	66	40	84
20	16	20	63	56	37	87
21	16	23	62	69	38	89
22	21	23	72	69	49	92
23	29	23	69	75	50	94
24	15	31	76	80	43	95
25	28	22	79	77	57	96
26	36	38	86	83	65	98
27	30	29	85	80	73	97
28	34	39	88	90	71	98
29	31	41	93	93	75	99
30	40	52	91	91	82	99
31	42	41	92	86	79	98

Continued(Table A1)

Faults	Beaking Probablity/%					
	SEI	GF	GF-SEI	MLE	GF-MLE	MLE-SEI
32	49	56	94	91	83	100
33	52	61	94	89	88	100
34	58	58	98	97	89	100
35	52	67	99	94	88	98
36	61	67	96	96	94	99
37	55	62	96	96	85	100
38	64	64	97	95	89	100
39	55	71	99	95	95	100
40	67	81	99	97	93	100
41	78	67	99	95	95	100
42	73	80	99	96	95	100
43	80	78	99	96	98	100
44	76	85	100	95	98	100
45	78	81	100	98	100	100
46	84	88	100	99	96	100
47	87	90	99	98	96	100
48	87	90	99	100	96	100
49	83	94	100	100	96	100
50	88	89	100	99	97	100
51	91	93	100	100	95	100
52	95	93	100	96	98	100
53	88	95	100	96	99	100
54	92	96	100	100	97	100
55	95	97	100	98	98	100
56	91	95	100	97	98	100
57	96	99	100	99	99	100
58	94	97	100	97	99	100
59	91	97	100	99	99	100
60	94	99	100	100	99	100
61	97	94	100	100	97	100
62	100	96	100	100	100	100
63	99	98	100	100	98	100
64	97	98	100	100	99	100
65	99	97	100	99	98	100
66	99	99	99	100	99	100
67	100	100	100	100	100	100
68	99	98	100	100	100	100
69	99	99	100	100	99	100
70	100	100	100	99	100	100

Table A2 Time of Breaking LBlock Using Different Distinguishers

表 A2 不同区分器破译 LBlock 的消耗时间

Faults	Latency Time/s					
	SEI	GF	GF-SEI	MLE	GF-MLE	MLE-SEI
0	0.0	0.0	0.0	0.0	0.0	0.0
1	7.5	93.6	26.7	4.9	19.8	2.8
2	7.6	94.9	8.4	5.3	9.6	3.6
3	7.6	95.0	6.0	7.6	6.8	4.8
4	10.1	94.9	7.0	11.3	8.5	5.9
5	12.5	90.8	8.5	14.4	10.5	7.1
6	13.9	89.9	9.8	15.2	11.6	8.3
7	16.8	94.9	11.1	17.1	13.5	9.7
8	19.8	102.8	12.6	20.9	14.6	10.9
9	20.3	97.9	13.8	20.3	15.6	12.1
10	26.0	107.3	15.0	25.0	16.8	13.4
11	23.1	99.7	16.5	22.4	18.2	14.5
12	19.7	79.9	17.9	20.7	19.3	15.8
13	23.4	84.9	19.2	23.6	20.6	17.2
14	22.5	80.7	20.6	22.3	21.8	18.5
15	23.7	78.5	22.0	23.6	23.1	19.9
16	24.4	78.2	23.5	24.1	24.6	20.8
17	27.7	83.7	24.7	27.6	25.6	21.9
18	29.0	83.5	26.3	28.9	27.0	23.3
19	30.3	85.8	27.5	30.8	28.1	24.6
20	30.5	83.8	28.7	30.4	29.2	25.6
21	31.2	83.7	30.5	31.1	30.9	26.9
22	34.2	87.1	31.6	33.4	31.9	28.4
23	36.1	89.0	33.1	35.6	33.5	29.5
24	37.1	89.1	34.4	37.2	34.6	30.8
25	41.4	94.2	35.7	40.8	35.8	31.9
26	39.9	92.2	37.2	40.2	37.0	33.1
27	43.1	95.2	38.3	43.2	38.2	34.5
28	43.6	94.9	39.7	43.5	39.6	35.6
29	45.6	98.0	40.7	47.1	40.5	36.7
30	48.5	100.8	41.9	49.0	41.8	38.1
31	48.2	101.6	43.3	48.8	43.2	39.3
32	54.2	107.0	44.7	53.6	44.5	40.6
33	53.9	107.5	46.1	54.7	46.0	41.7
34	56.9	109.5	47.6	57.6	47.5	42.8

Continued(Table A2)

Faults	Latency Time/s					
	SEI	GF	GF-SEI	MLE	GF-MLE	MLE-SEI
35	61.2	113.6	48.8	62.6	48.7	44.2
36	60.3	113.5	50.0	60.9	49.9	46.0
37	62.2	115.2	51.4	62.5	51.2	47.2
38	63.8	116.2	52.7	64.1	52.6	47.9
39	65.4	118.2	54.3	65.8	54.1	49.1
40	70.0	123.0	55.4	70.6	55.3	50.3
41	77.9	130.3	56.7	79.4	56.5	51.7
42	79.3	131.4	58.6	80.2	58.3	52.9
43	78.0	130.2	59.4	79.6	59.2	54.2
44	75.6	126.2	60.8	76.4	60.6	55.4
45	65.5	114.8	62.4	65.8	62.2	56.7
46	76.5	130.1	63.7	77.1	63.5	57.9
47	77.1	130.4	64.7	77.9	64.7	59.6
48	72.0	123.6	66.4	72.3	66.2	60.7
49	72.9	125.2	67.8	73.3	67.7	61.5
50	83.1	139.0	68.8	83.6	68.7	62.8
51	84.7	141.2	70.9	85.3	71.1	64.1
52	87.3	144.6	71.9	87.9	71.8	65.2
53	86.7	141.1	72.9	87.8	73.1	66.6
54	94.5	154.3	74.4	95.3	74.5	68.2
55	97.2	157.3	76.0	98.5	75.8	69.7
56	98.4	158.4	77.0	99.3	76.8	71.9
57	99.9	160.5	78.7	100.8	78.3	71.4
58	100.6	155.6	80.3	101.4	80.4	72.2
59	101.1	161.2	81.5	102.1	81.4	73.6
60	93.6	145.0	82.8	95.5	82.8	75.6
61	82.7	131.8	84.1	83.0	83.8	144.9
62	84.2	133.3	85.4	84.8	85.3	119.8
63	86.5	136.5	86.9	86.4	86.5	125.0
64	93.9	146.6	88.5	94.1	88.4	115.6
65	94.1	145.7	90.1	95.5	90.1	103.3
66	95.2	146.3	92.1	95.9	91.9	99.4
67	95.0	146.6	110.3	95.8	109.9	100.4
68	91.7	141.5	112.8	92.5	112.2	102.2
69	92.8	143.1	117.2	93.9	116.3	103.4
70	94.1	148.7	118.1	95.9	117.7	105.0