计算机系统应用 ISSN 1003-3254, CODEN CSAOBN
Computer Systems & Applications,2022,31(10):288−294 [doi: 10.15888/j.cnki.csa.008764]

E-mail: csa@iscas.ac.cn
http://www.c-s-a.org.cn
Tel: +86-10-62661041

# 基于 ESF 密码算法改进的差分故障攻击①

孔　曼[1], 谭　林[1], 王云丽[1], 龙　敏[2]

[1](湖南天河国云科技有限公司, 长沙 410100)
[2](长沙理工大学 计算机与通信工程学院, 长沙 410114)
通信作者: 孔　曼, E-mail: kongman@tianhecloud.com

**摘　要**: 利用置换层结构的特点及差分故障的基本思想, 提出一种针对 ESF 算法的差分故障攻击方法. 在第 30 轮多次注入 1 比特故障, 根据 S 盒的差分特性, 由不同的输入输出差分对, 得到不同的 S 盒的输入值集合, 取其交集可快速确定唯一的 S 盒的可能输入值, 分析得出最后一轮轮密钥. 采用同样的方法, 多次在第 29 轮、28 轮注入 1 比特故障, 结合最后一轮轮密钥, 同样利用 S 盒的差分特性分析得出倒数第 2 轮、第 3 轮轮密钥. 共需约 10 个故障密文, 恢复 3 轮轮密钥后将恢复主密钥的计算复杂度降为 $2^{22}$.

**关键词**: 轻量级分组密码; 差分故障攻击; ESF 算法; 置换层; 计算复杂度

引用格式: 孔曼,谭林,王云丽,龙敏.基于 ESF 密码算法改进的差分故障攻击.计算机系统应用,2022,31(10):288–294. http://www.c-s-a.org.cn/1003-3254/8764.html

## Improved Differential Fault Attack Based on ESF Cryptographic Algorithm

KONG Man[1], TAN Lin[1], WANG Yun-Li[1], LONG Min[2]

[1](Hunan Tianheguoyun Technology Co. Ltd., Changsha 410100, China)
[2](School of Computer and Communication Engineering, Changsha University of Science and Technology, Changsha 410114, China)

**Abstract**: In light of the structural characteristics of the displacement layer and the basic idea of differential fault, this study proposes a differential fault attack method for the eight-sided fortress (ESF) algorithm. In the $30^{th}$ round, a 1-bit fault is injected multiple times. Various input and output differential pairs are used to obtain different input sets for the S-box according to the differential characteristics of the S-box. Taking the intersection of the sets is a quick way to determine the only possible inputs for the S-box. The round key of the last round can then be obtained through analysis. Similarly, a 1-bit fault is injected in the $29^{th}$ and $28^{th}$ rounds many times. With the round key of the last round, the differential characteristics of the S-box are leveraged again to obtain the round keys of the last but one and last but two rounds. About 10 fault ciphertexts are required. After the round keys of three rounds are recovered, the computational complexity of recovering the master key is reduced to $2^{22}$.

**Key words**: lightweight block cipher; differential fault attack; eight-sided fortress (ESF) algorithm; displacement layer; computational complexity

20 世纪 70 年代中期, 分组密码的研究开始兴起, 至今信息安全领域的学者们已经取得了许多丰硕的研究成果. 近年来, 随着物联网的发展, 无线传感器网络[1] 和无线射频技术[2] 的应用愈来愈广泛. 轻量级分组密码的出现解决了微型计算设备计算能力有限、低功耗的问题, 且其加密速度快、易于硬件实现、能够应用于资源受限的环境中, 还具有较高的安全性, 所以自问世以来就一直是密码学界关注的焦点. 然而, 随着信息技术的高速发展, 人们对于信息的价值流通具有高要求, 虽然这些技术通过开放性的计算机网络实现

① 基金项目: 湖南省十大技术攻关项目
收稿时间: 2022-01-19; 修改时间: 2022-02-15, 2022-03-11; 采用时间: 2022-03-18; csa 在线出版时间: 2022-07-14

信息交换和共享, 同时也带来了交互效率低、安全保障低等问题.

此外, 区块链+物联网[3] 是未来的发展方向, 区块链技术可以为物联网提供点对点直接互联的方式来传输数据, 并且, 区块链的密码机制[4] 能够为物联网中的信息传输创造安全的环境. 尤其在资源受限的物联网节点中需要对数据进行加密时, 不可避免地引入具有占用资源少、功耗低、效率高、易于实现等优势的轻量级密码算法, 同时对密码算法的安全性提出了更高的要求.

现在的轻量级分组密码算法大都受到 DES[5] 和 AES[6] 设计原理的影响, 目前已有许多的轻量级密码算法陆续提出, 有非常经典的轻量级分组加密算法 LBlock[7]、LED[8]、MIBS[9]、KLEIN[10] 等, 还有近两年的新秀, 如 Midori[11]、HBcipher[12]、Surge[13] 等. 然而这些实际用于物联网设备中的轻量级分组密码算法日益凸显出安全漏洞, 所以需要对该类算法进行分析研究, 不断完善攻击过程与方法, 以实现密码算法的更新换代, 提升轻量级分组密码算法的防御能力, 进一步提升区块链中密码技术的安全性.

故障攻击是一种侧信道攻击方法, 于 1997 年被 Boneh 等人[14] 提出, 并用此方法攻击了基于 CRT 算法实现的 RSA 签名密钥, 意味着首次将密码故障应用于密码分析. 同年, Biham 等人首次提出了差分故障攻击[15] 并且成功分析了 DES 算法. 此后, 差分故障攻击被广泛应用于公钥密码算法、分组密码算法等等. 完成差分故障攻击最重要的一点就是在加密设备中引进故障, 如电压瞬变、外部时钟骤变、激光束、X-射线等. 差分故障攻击已经应用到了许多的轻量级密码算法分析中. 文献 [16] 以比特为单位进行故障注入, 有效地攻击 GIFT 算法; 文献 [17] 针对 SKINNY 密码算法提出了恢复主密钥平均需要 10 个半字节故障, 且通过了大量的模拟验证; 文献 [18] 利用 S 盒的差分不均匀性, 在最后一轮注入两次 8 个半字节型的故障, 快速恢复了最后一轮密钥的信息; 文献 [19] 提出了随机注入 2 字节故障的模型, 两对正误密文就可以在不穷举搜索的情况下检索整个 128 位 AES 密钥; 文献 [20] 提出了在密钥调度过程中注入一个字节型故障, 仅需 4 个错误的密文确定整个密钥.

本文通过分析 ESF[21] 的算法结构, 提出了一种差分故障攻击方法. 此方法针对结构中按位进行运算的算法具有通用性. 此攻击方法的主要思想是, 在 S 盒运算前注入错误故障, 结合差分方程与 S 盒在不同故障条件下输出差分不均匀性, 进而获取内部状态信息, 最后分析得出初始密钥. 共需要约 10 个故障密文, 可将计算复杂度降为 $2^{22}$.

## 1 ESF 算法介绍

### 1.1 加密流程

ESF 算法是一个基于变种的 Feistel 结构的加密算法, 轮函数采用的是 SPN 代换置换网络形式, 其整体结构借鉴于 LBlock 算法, 其中的置换层是仿照了 PRESENT 算法的按位置换形式.

算法流程包含了轮密钥异或、非线性变换、线性扩散、中间状态异或与移位操作. ESF 算法的分组长度为 64 比特, 初始密钥 80 比特, 迭代轮数为 32 轮. 加密流程如图 1, 数据输入分成左右两个部分, 令 $P = L_0 \| R_0$ 表示 64 位明文, $C = L_{32} \| R_{32}$ 表示 64 位密文. $K_i$ ($i$=1, 2, ⋯, 32) 是每一轮迭代加密的轮密钥. 轮函数 $F$ 中包含 S 盒非线性变换和 P 盒按位置换.

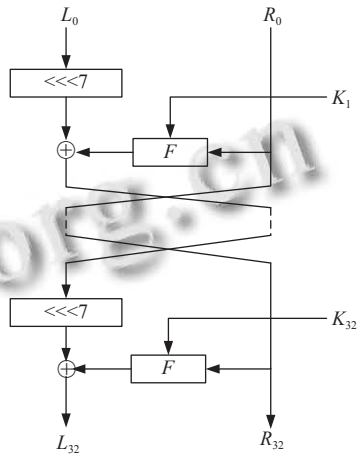


图 1 ESF 算法的加密流程

### 1.2 轮密钥生成

ESF 算法的初始密钥是 80 位, 记为 $K = k_{79}k_{78}\cdots k_0$, 初始密钥的左边 32 位作为第一轮轮密钥 $K_1$, $K^i$ ($i$=1, 2, ⋯, 32) 是 80 位的密钥中间状态, 轮密钥生成算法如算法 1 所示.

算法 1. ESF 算法的密钥扩展方案

输入: 80 比特初始密钥.
输出: 轮密钥.

1) $K^1 \leftarrow K$;
2) for $i$ = 2 to 32 do;
3) $K^i \leftarrow K^{i-1}$<<<13;
4) $[k_{79}k_{78}k_{77}k_{76}] = S_0([k_{79}k_{78}k_{77}k_{76}])$;
5) $[k_{75}k_{74}k_{73}k_{72}] = S_0([k_{75}k_{74}k_{73}k_{72}])$;
6) $[k_{47}k_{46}k_{45}k_{44}k_{43}] = [k_{47}k_{46}k_{45}k_{44}k_{43}] \oplus [i]_2$;
7) $K^i \leftarrow [k_{79}k_{78}\cdots k_0]$;
8) $K_i \leftarrow [k_{79}k_{78}\cdots k_{48}]$.

## 2 ESF 算法结构分析

### 2.1 ESF 算法置换层结构分析

ESF 共有 8 个 S 盒, 前 4 个 S 盒 ($S_7$–$S_4$) 的输出去往奇数号的 S 盒 ($S_7$、$S_5$、$S_3$、$S_1$), 后 4 个 S 盒 ($S_3$–$S_0$) 的输出去往偶数号的 S 盒 ($S_6$、$S_4$、$S_2$、$S_0$). 具体传播位置置换如图 2 所示, 其中粗线表示有故障的传播, $I_i$ ($i$=0, 1, ⋯, 7) 是右半明文中间状态的半字节表示形式.

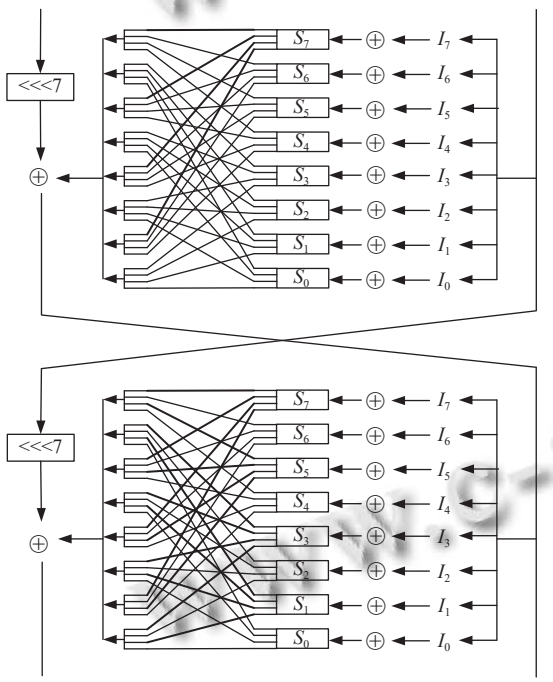


图 2 ESF 的置换图

### 2.2 ESF 算法 S 盒差分分布情况

假设 $a$ 是一个半字节作为某个 S 盒的输入. 在 S 盒输入前, 导入随机故障 $f$, 即为 S 盒的输入差分. $f$*表示为 S 盒的输出差分, 则满足差分公式:

$$f^* = S(a) \oplus S(a \oplus f) \tag{1}$$

已知每一个输入差分都有一个相应的输出差分, 且输入差分有 $2^4$ = 16 种可能, 对应的输出差分有 4–8 种可能, 并且每一对 ($f$,$f$*) 能够得到输入值 $a$ 的集合, 集合内元素的个数为 2 或者 4. 正是由于分布的不均匀性, 成为了差分故障攻击的突破口. 由于篇幅限制, 文中只列出 $S_7$ 盒的差分分布表. ESF 算法 $S_7$ 盒的差分性如表 1 所示 (*ID* 表示输入差分, *OD* 表示输出差分).

表 1 ESF 中 $S_7$ 盒的差分分布表

| *ID* | *OD* | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |
| 1 | 0 | 0 | 4 | 0 | 0 | 4 | 0 | 0 | 2 | 2 | 0 | 2 | 0 | 0 | 2 |
| 2 | 0 | 0 | 2 | 0 | 2 | 0 | 0 | 0 | 0 | 0 | 2 | 4 | 2 | 4 | 0 |
| 3 | 2 | 2 | 0 | 0 | 2 | 2 | 0 | 2 | 0 | 2 | 0 | 0 | 2 | 0 | 2 |
| 4 | 0 | 0 | 0 | 0 | 2 | 0 | 2 | 0 | 2 | 0 | 2 | 2 | 2 | 2 | 2 |
| 5 | 0 | 2 | 2 | 4 | 0 | 0 | 0 | 0 | 2 | 2 | 0 | 0 | 2 | 0 | 2 |
| 6 | 2 | 4 | 2 | 0 | 2 | 2 | 0 | 2 | 2 | 0 | 0 | 0 | 0 | 0 | 0 |
| 7 | 4 | 0 | 2 | 0 | 0 | 0 | 2 | 0 | 0 | 2 | 0 | 0 | 0 | 2 | 4 |
| 8 | 0 | 0 | 2 | 0 | 0 | 2 | 4 | 0 | 2 | 2 | 2 | 0 | 2 | 0 | 0 |
| 9 | 2 | 0 | 0 | 2 | 4 | 0 | 0 | 0 | 0 | 2 | 0 | 2 | 2 | 2 | 0 |
| A | 0 | 0 | 0 | 2 | 0 | 0 | 2 | 4 | 0 | 0 | 4 | 0 | 2 | 2 | 0 |
| B | 4 | 2 | 0 | 0 | 0 | 0 | 2 | 2 | 0 | 0 | 4 | 0 | 2 | 0 | 0 |
| C | 2 | 0 | 0 | 0 | 0 | 2 | 0 | 2 | 2 | 2 | 0 | 2 | 0 | 4 | 0 |
| D | 0 | 2 | 0 | 2 | 2 | 0 | 2 | 2 | 2 | 2 | 0 | 0 | 0 | 0 | 2 |
| E | 0 | 4 | 2 | 2 | 2 | 2 | 0 | 0 | 0 | 0 | 2 | 0 | 0 | 0 | 2 |
| F | 0 | 0 | 0 | 4 | 0 | 2 | 2 | 2 | 2 | 0 | 0 | 4 | 0 | 0 | 0 |

### 2.3 ESF 算法差分扩散规律

已经知道 ESF 算法的置换层的基本运算单位是比特. 在 S 盒运算前随机注入 1 比特故障, 那么无论在哪个 S 盒, 输入差分只可能为 0001、0010、0100、1000 这 4 个中的任何一个, 由这 4 个输入差分, 可列出每个 S 盒对应的输出差分, 以 $S_7$ 盒为例, 表 2 列出了 $S_7$ 盒的输入所对应的所有可能出现的输出差分 (*OD* 表示输出差分).

表 2 ESF 中 $S_7$ 盒每一个输出差分可能的输入

| *OD* | 0011 | 0101 | 0110 | 0111 | 1001 | 1010 | 1011 | 1100 | 1101 | 1110 | 1111 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 可能的输入 | 1000 | 1101 | 0100 | 1001 | 0110 | 1100 | 1000 | 0000 | 0001 | 0000 | 0010 |
| | 1001 | 1111 | 0101 | 1101 | 0111 | 1101 | 1010 | 0001 | 0011 | 0010 | 0011 |
| | 1110 | 0001 | 1010 | 0100 | 1010 | 0011 | 0011 | 0100 | 0010 | 1001 | 0000 |
| | 1111 | 0101 | 1011 | 0110 | 1110 | 1011 | 0111 | 0110 | 0110 | 1011 | 0100 |
| | 0101 | — | 0000 | 1110 | 0001 | — | 0101 | 1100 | 0111 | 1000 | — |
| | 0111 | — | 1000 | 1010 | 1001 | — | 1101 | 1110 | 1111 | 1100 | — |
| | 0010 | — | — | — | — | — | — | 1011 | — | — | — |
| | 1010 | — | — | — | — | — | — | 1111 | — | — | — |

观察表 2 的输出差分, 可知, 当输入差分仅为 0001、0010、0100、1000 时, 经过 S 盒后, 至少能发生 2 比特的故障错误. 再分析每个 S 盒的可能的输入, 可以发现经过 S 盒后, 发生 2 比特和 3 比特故障错误

的概率较大, 且发生 2 比特故障错误的概率大于发生 3 比特故障错误的概率, 发生 4 比特故障错误的概率最大也才 1/8, 因此, 在 S 盒运算前注入 1 比特故障, 至少将导致 2 个半字节出现错误. 本文将以 2 比特故障错误进行保守分析.

在 S 盒运算前注入 1 比特故障, 经过 3 轮迭代运算后, 只导致两个 S 盒的输出半字节出现故障错误的平均概率仅为 0.019 5, 所以在分析过程中可以忽略此种情况. 本文将以最后一轮 S 盒输出半字节出现故障的个数为 3 的情况进行保守分析.

## 3 ESF 的差分故障攻击

### 3.1 攻击条件

(1) 攻击者有能力选择一个明文进行加密, 并获得相应的正确\故障密文;

(2) 攻击者能够诱导 1 比特故障输入到加密的第 30 轮寄存器中;

(3) 故障位置和值均未知.

### 3.2 攻击过程及分析

#### 3.2.1 攻击流程

(1) 选择明文 $P$ 进行加密, 获得正确密文 $C$.

(2) 恢复最后一轮轮密钥, 步骤如下:

1) 对同样的明文 $P$ 进行加密, 在第 30 轮轮函数运行前注入 1 比特随机故障到寄存器 $B^{30}N^j$ ($1\leqslant j\leqslant 8$, 表示在第 30 轮、第 $j$ 个 S 盒位置) 中, 获得错误密文 $D1$. 将正确密文 $C$ 与错误密文 $D1$ 进行异或, 得到密文差分 $\Delta D1$ (此外, 密文差分还受上一轮输出的左边 32 比特的差分影响, 但可直接观察最后一轮右边 32 比特密文差分, 得到准确的进入逆运算的密文差分), 密文差分经过逆 P 盒置换, 得到最后一轮 S 盒的输出差分.

2) 查找表 2, 由最后一轮 S 盒的输出差分列出相应 S 盒正确输入的候选值.

3) 在第 30 轮轮函数运算前, 多次注入 1 比特的随机故障, 并重复步骤 1) 和步骤 2), 不断缩小 S 盒正确输入的候选值的个数, 直到剩下唯一一个. 此时得到的就是 S 盒的正确输入值.

4) 最后将正确 S 盒的输入值与正确密文的左边 32 比特异或, 即可得到最后一轮轮密钥.

(3) 恢复第 31 轮轮密钥, 步骤如下:

1) 结合最后一轮轮密钥和最后一轮输出, 可逆推得到第 31 轮正确输出. 此时, 在第 29 轮注入 1 比特的随机故障到位置 $B^{29}N^j$ ($1\leqslant j\leqslant 8$) 中, 获得错误密文 $D2$, 结合最后一轮轮密钥, 可逆推得到第 31 轮故障输出. 将第 31 轮的正确输出与故障输出异或, 得到中间状态密文差分 (此外, 密文差分还受上一轮输出的左边 32 比特的差分影响, 但可直接观察第 31 轮的输出的右 32 比特密文差分, 得到准确的进入逆运算的密文差分). 密文差分经过逆 P 盒置换, 得到第 31 轮 S 盒的输出差分.

2) 查找表 2, 由第 31 轮 S 盒的输出差分列出相应 S 盒正确输入的候选值.

3) 在第 29 轮轮函数运算前, 多次注入 1 比特的随机故障, 并重复步骤 1) 和步骤 2), 不断缩小 S 盒正确输入的候选值的个数, 直到剩下唯一一个. 此时得到的就是 S 盒的正确输入值.

4) 最后将正确 S 盒的输入值与正确密文的左边 32 比特异或, 即可得到第 31 轮轮密钥.

(4) 恢复第 30 轮轮密钥, 步骤如下:

1) 结合第 31 轮轮密钥和第 31 轮输出, 可逆推得到第 30 轮正确输出. 此时, 在第 28 轮注入 1 比特的随机故障到寄存器 $B^{28}N^j$ ($1\leqslant j\leqslant 8$) 中, 获得错误密文 $D3$, 结合第 31 轮轮密钥, 可逆推得到第 30 轮故障输出. 将第 30 轮的正确输出与故障输出异或, 得到中间状态密文差分 (此外, 密文差分还受上一轮输出的左边 32 比特的差分影响, 但可直接观察第 30 轮的输出的右边 32 比特密文差分, 得到准确的进入逆运算的密文差分). 密文差分经过逆 P 盒置换, 得到第 30 轮 S 盒的输出差分.

2) 通过与 (3) 类似的方法分析出第 30 轮轮密钥.

#### 3.2.2 差分故障攻击方法分析

由第 3.2.1 节已经知道, 在第 30 轮 S 盒运算前随机注入 1 比特故障, 在第 32 轮至少能得到 3 个错误的 S 盒输出, 在这样的情况下, 恢复第 32 轮的轮密钥所需要的错误密文则减少到 6 个 (平均每两对 S 盒的输入输出差分所对应的可能输入值的集合能确定唯一的 S 盒半字节输入值).

由于是在第 30 轮的 S 盒运算前注入的故障错误, 因此, 在恢复最后一轮轮密钥的同时, 也可以得到第 31 轮中至少 12 个错误的 S 盒输出, 那么只需要在第 29 轮的 S 盒运算前再注入 2 比特的故障错误, 得到 2 个错误密文就可以恢复第 31 轮轮密钥; 在恢复最后一轮轮密钥时, 已经得到第 30 轮的 6 个错误的 S 盒输

出, 在恢复第 31 轮轮密钥时, 已经得到第 30 轮的至少 4 个错误的 S 盒输出, 那么只需在第 28 轮 S 盒运算前注入 2 比特故障, 即可恢复第 30 轮轮密钥. 综上所述, 理论上只需要约 10 个错误密文就可以恢复最后 3 轮轮密钥 $K_{32}$、$K_{31}$、$K_{30}$.

3.2.3 密钥推断过程

以下步骤通过恢复出的最后 3 轮轮密钥 $K_{32}$、$K_{31}$、$K_{30}$, 来推断 80 比特的完整子密钥, 其中“||”表示连接符.

(1) 根据密钥的扩展方案, $K_{30}$ 是 $K^{30}$ 的左 32 位, $K^{30} = K_{30}[0{:}31] \,||\, K^{30}[47{:}0]$;

(2) 移位之后:

$K^{30} = K_{30}[13{:}31] \,||\, K^{30}[47{:}0] \,||\, K_{30}[0{:}12]$;

(3) 过 S 盒之后:

$K^{30} = S_0(K_{30}[13{:}16]) \,||\, S_0(K_{30}[17{:}20]) \,||\, K_{30}[21{:}31] \,||\, K^{30}[47{:}0] \,||\, K_{30}[0{:}12]$;

(4) 加轮常量之后:

$K^{31} = S_0(K_{30}[13{:}16]) \,||\, S_0(K_{30}[17{:}20]) \,||\, K_{30}[21{:}31] \,||\, K^{30}[47{:}34] \,||\, K^{30}[33{:}29] \oplus i \,||\, K^{30}[28{:}0] \,||\, K_{30}[0{:}12]$.

(5) $K_{31}$ 是 $K^{31}$ 的左 32 位, 移位之后:

$K^{31} = K_{30}[26{:}31] \,||\, K^{30}[47{:}34] \,||\, K^{30}[33{:}29] \oplus i \,||\, K^{30}[28{:}0] \,||\, K_{30}[0{:}12] \,||\, S_0(K_{30}[13{:}16]) \,||\, S_0(K_{30}[17{:}20]) \,||\, K_{30}[21{:}25]$;

(6) 过 S 盒之后:

$K^{31} = S_0(K_{30}[26{:}29]) \,||\, S_0(K_{30}[30{:}31] \,||\, K^{30}[47{:}46]) \,||\, K^{30}[45{:}34] \,||\, K^{30}[33{:}29] \oplus i \,||\, K^{30}[28{:}0] \,||\, K_{30}[0{:}12] \,||\, S_0(K_{30}[13{:}16]) \,||\, S_0(K_{30}[17{:}20]) \,||\, K_{30}[21{:}25]$;

(7) 加轮常量之后:

$K^{32} = S_0(K_{30}[26{:}29]) \,||\, S_0(K_{30}[30{:}31] \,||\, K^{30}[47{:}46]) \,||\, K^{30}[45{:}34] \,||\, K^{30}[33{:}29] \oplus i \,||\, K^{30}[28{:}22] \,||\, K^{30}[21{:}17] \oplus i \,||\, K^{30}[16{:}0] \,||\, K_{30}[0{:}12] \,||\, S_0(K_{30}[13{:}16]) \,||\, S_0(K_{30}[17{:}20]) \,||\, K_{30}[21{:}25]$.

$K_{32}$ 是 $K^{32}$ 左 32 位, 可以看出 $K^{32}$ 已知 32+13+4+4+5 = 58 位, 剩余 22 位未知, 可以通过穷举的方式获得初始密钥, 即初始密钥搜索空间降至 $2^{22}$.

## 4 实验结果与分析

### 4.1 实验配置

本文使用一台普通的笔记本电脑进行实验, 处理器为 AMD A4-Series A4-5000 四核, 操作系统为 64 位 Windows 7 旗舰版 SP1, 内存为 4 GB. 采用使用 DVE-C++ 5.1 软件编程.

### 4.2 攻击实验结果

本文 ESF 算法中的故障注入, 是通过编程修改相关语句来实现的. 实验过程中的由于样本选取的随机性及样本空间有限, 在恢复最后一轮轮密钥的过程中, 实际需要的故障密文数量会在理论值周围波动. 我们进行了多组实验, 现仅列出其中 10 组如表 3 所示, 表 4 列举了序号 1 的实验结果.

表 3 差分故障攻击 ESF 算法的实验数据

| 序号 | 恢复第$n$轮密钥 | | | |
|---|---|---|---|---|
| | 32 | 31 | 30 | 总计 |
| 1 | 6 | 2 | 3 | 11 |
| 2 | 6 | 4 | 3 | 13 |
| 3 | 5 | 3 | 3 | 11 |
| 4 | 3 | 3 | 3 | 9 |
| 5 | 2 | 3 | 3 | 8 |
| 6 | 5 | 3 | 3 | 11 |
| 7 | 4 | 2 | 3 | 9 |
| 8 | 5 | 2 | 3 | 10 |
| 9 | 4 | 3 | 3 | 10 |
| 10 | 6 | 3 | 2 | 11 |

表 4 恢复轮密钥的一组实验数据

| 编号 | 错误密文 | 当前S盒输入值及恢复的子密钥 |
|---|---|---|
| 1 | 442FBBD24248F970 | |
| 2 | AF20CA7B6B039353 | |
| 3 | 18CABE23C0C0D178 | $S^{32}$: 3A0E7C8E |
| 4 | 714654D33F578603 | $K_{32}$: 504CAFDC |
| 5 | CA197DA0E8C0D150 | |
| 6 | 3B681A632A02D213 | |
| 1 | 068C6DE5BEE77BF6 | $S^{31}$: DDD10582 |
| 2 | 8AD6CF2A89D04DE7 | $K_{31}$: 76236A65 |
| 1 | 578A2691037B37DD | $S^{30}$: A02C0AD3 |
| 2 | 04BC2CEA026A63DC | $K_{30}$: F61629EB |
| 3 | 39662836173B2288 | |

固定明文和密钥, 明文取 0123456789ABCDEF, 密钥取 0123456789ABCDEFFEDC, 加密后, 获得的正确密文为 9F68B9406A42D352. 采用本文方法, 计算出第 32 轮的子密钥 504CAFDC、 第 31 轮子密钥 76236A65 以及第 30 轮子密钥 F61629EB. 这与未注入故障前, 正确运行密码算法程序时得出的最后 3 轮子密钥一致, 验证了本方法的正确性.

根据表 3 所列的实验数据, 当恢复第 32、31、30 轮子密钥时, 分别需要进行多次不等的故障注入, 但最终的总计结果体现了计算攻击时所需故障密文数量为 10.3 个, 就能够恢复最后 3 轮子密钥. 接近理论所需

的故障密文数量.

### 4.3 对比分析

本文针对密码算法 ESF 改进的差分故障攻击, 其明密文对数量约为 10 个, 时间复杂度为 $2^{22}$, 表 5 描述了其他针对 ESF 算法的攻击方法以供对比.

表 5 针对 ESF 算法的攻击方法对比表

| 论文 | 复杂度 | | 方法 |
|---|---|---|---|
| | *data* | *time* | |
| 文献[22] | $2^{53}$ | $2^{60.23}$ | 不可能差分攻击 |
| 文献[23] | $2^{47}$ | $2^{66}$ | 相关密钥差分攻击 |
| 文献[24] | $2^{61.99}$ | $2^{77.39}$ | 截断不可能差分攻击 |
| 文献[25] | 24 | $2^{22}$ | 差分故障攻击 |
| 本文 | 10 | $2^{22}$ | 差分故障攻击 |

高红杰等人[22] 研究了 ESF 算法抵抗不可能差分攻击的能力, 基于一条 8 轮不可能差分路径, 根据轮密钥之间的关系, 对 12 轮 ESF 算法进行了攻击, 攻击 12 轮 ESF 算法所需的数据复杂度为 $2^{53}$, 时间复杂度为 $2^{60.23}$. 尹军等人[23] 通过建立相关密钥下的 MILP 模型, 利用搜索到的 11 轮相关密钥差分特征, 提出了 13 轮的相关密钥差分攻击, 攻击的数据复杂度为 $2^{47}$, 时间复杂度为 $2^{66}$. 李明明等人[24] 利用密钥编排算法部分子密钥间存在的依赖关系, 给出了 ESF 算法的 13 不可能差分分析, 其时间复杂度为 $2^{77.39}$, 数据复杂度为 $2^{61.99}$ 个选择明文. 徐朋[25] 通过在 28 到 32 轮右半部分导入共约 24 次故障, 将密钥搜索空间降至 $2^{22}$, 此方法故障注入要求难度大, 需要在轮函数输入前的 8 个半字节状态上同时发生故障.

本文方法相较于前 3 种方法, 有故障注入操作这一技术要求, 但是时间复杂度和数据复杂度相对来说降低不少, 这得益于差分故障攻击自身的优势; 相较于第 4 种同样的差分故障攻击, 本文不需要有高要求的故障注入手段, 例如, 本文不需要限定在一轮的某个位置或者某一部分注入故障, 而是可以任意在一轮中随机注入故障, 然后充分完整地分析其故障扩散的规律, 找到此攻击方法. 而文献 [25] 中, 恢复最后 3 轮密钥时, 将注入故障限定在右半部分密文的每个半字节上, 极大地增大了实现难度. 因此本文针对 ESF 的攻击方法, 是目前存在的研究中最优的方法.

## 5 结束语

本文提出了一种针对置换层为拉线型的密码算法 ESF 的改进的差分故障攻击. 通过选定最后 3 轮, 分析故障扩散的程度, 分别注入 6 次、2 次、2 次故障, 共 10 次故障, 并结合差分表能够分析出最后 3 轮轮密钥. 结合最后 3 轮轮密钥和密钥编排, 可将恢复主密钥的计算复杂度降至 $2^{22}$. 在本章的差分故障分析中, 为了保证方法的通用性, 且避免最优情况的偶然性, 分析过程都是利用概率较大的情况进行分析, 例如本文分析中, 如果故障影响的比特数越多, 密钥分析难度越低, 但由于故障传播路径中至少产生 2 个比特故障, 且产生 2 比特故障的概率较大, 所以会优先利用实验中产生 2 比特故障的情况进行保守分析. 这种办法通用性强, 还可以用于其他具有类似置换层的密码算法中, 分析置换层故障的传播特性和 S 盒的差分分布, 可得到全部或部分密钥. 另外, 由于 S 盒的一些抗差分性质, 导致分析出来的某处 S 盒的输入值不唯一的现象. 所以, 下一步, 将针对 S 盒的差分分布和其差分均匀度进行研究并改善, 加强抵抗此类差分故障分析.

## 参考文献


1 马莹, 文波. 无线传感器网络数据隐私保护技术. 网络安全技术与应用, 2021, (12): 75–76.

2 周喜, 王会珍, 赵娟萍. 基于 RFID 技术的门禁管理系统设计. 科学技术创新, 2021, (3): 66–67. [doi: 10.3969/j.issn.1673-1328.2021.03.031]

3 姚中原, 潘恒, 祝卫华, 等. 区块链物联网融合: 研究现状与展望. 应用科学学报, 2021, 39(1): 174–184. [doi: 10.3969/j.issn.0255-8297.2021.01.015]

4 蔡晓晴, 邓尧, 张亮, 等. 区块链原理及其核心技术. 计算机学报, 2021, 44(1): 84–131.

5 Leander G, Paar C, Poschmann A, *et al*. New lightweight DES variants. Proceedings of the 14th International Workshop on Fast Software Encryption. Luxembourg: Springer, 2007. 196–210.

6 柴绍杰, 张彩珍. AES 加密算法的改进及 FPGA 实现. 兰州交通大学学报, 2020, 39(3): 47–53. [doi: 10.3969/j.issn.1001-4373.2020.03.008]

7 Wu WL, Zhang L. LBlock: A lightweight block cipher. Proceedings of the 9th International Conference on Applied Cryptography and Network Security. Nerja: Springer, 2011. 327–344.

8 Guo J, Peyrin T, Poschmann A, *et al*. The LED block cipher. Proceedings of the 13th International Workshop on Cryptographic Hardware and Embedded Systems. Nara: Springer, 2011. 326–341.

9 Izadi M, Sadeghiyan B, Sadeghian SS, *et al*. MIBS: A new lightweight block cipher. Proceedings of the 8th International Conference on Cryptology and Network Security. Kanazawa: Springer, 2009. 334–348.

10 Gong Z, Nikova S, Law YW. KLEIN: A new family of lightweight block ciphers. Proceedings of the 7th International Workshop on Security and Privacy. Amherst: Springer, 2011. 1–18.

11 Banik S, Bogdanov A, Isobe T, *et al*. Midori: A block cipher for low energy. Proceedings of the 21st International Conference on the Theory and Application of Cryptology and Information Security. Auckland: Springer, 2015. 411–436.

12 李浪, 郭影, 刘波涛, 等. HBcipher: 一种高效的轻量级分组密码. 密码学报, 2019, 6(3): 336–352. [doi: 10.13868/j.cnki.jcr.000306]

13 李浪, 刘波涛. Surge: 一种新型、低资源、高效的轻量级分组密码算法. 计算机科学, 2018, 45(2): 236–240. [doi: 10.11896/j.issn.1002-137X.2018.02.041]

14 Boneh D, DeMillo RA, Lipton RJ. On the importance of checking cryptographic protocols for faults. International Conference on the Theory and Applications of Cryptographic Techniques. Berlin, Heidelberg: Springer, 1997. 37–51.

15 Biham E, Shamir A. Differential fault analysis of secret key cryptosystems. Proceedings of the 17th Annual International Cryptology Conference Santa Barbara. California: Springer, 1997. 513–525.

16 冯天耀, 韦永壮, 史佳利, 等. 轻量级分组密码 GIFT 的差分故障攻击. 密码学报, 2019, 6(3): 324–335.

17 Vafaei N, Bagheri N, Saha S, *et al*. Differential fault attack on SKINNY block cipher. Proceedings of the 8th International Conference on Security, Privacy, and Applied Cryptography Engineering. Kanpur: Springer, 2018. 177–197.

18 王永娟, 张诗怡, 王涛, 等. 对 MIBS 分组密码的差分故障攻击. 电子科技大学学报, 2018, 47(4): 601–605. [doi: 10.3969/j.issn.1001-0548.2018.04.020]

19 Zhang JB, Wu N, Li JH, *et al*. A novel differential fault analysis using two-byte fault model on AES key schedule. IET Circuits, Devices & Systems, 2019, 13(5): 661–666.

20 Gruber M, Selmke B. Differential fault attacks on KLEIN. Proceedings of the 10th International Workshop on Constructive Side-Channel Analysis and Secure Design. Darmstadt: Springer, 2019. 80–95.

21 Liu X, Zhang WY, Liu XZ, *et al*. Eight-sided fortress: A lightweight block cipher. The Journal of China Universities of Posts and Telecommunications, 2014, 21(1): 104–108, 128. [doi: 10.1016/S1005-8885(14)60275-2]

22 高红杰, 卫宏儒. 用不可能差分法分析 12 轮 ESF 算法. 计算机科学, 2017, 44(10): 147–149, 181. [doi: 10.11896/j.issn.1002-137X.2017.10.028]

23 尹军, 宋健, 曾光, 等. 轻量级分组密码算法 ESF 的相关密钥差分分析. 密码学报, 2017, 4(4): 333–344.

24 李明明, 郭建胜, 崔竞一, 等. ESF 算法的截断不可能差分分析. 密码学报, 2019, 6(5): 585–593.

25 徐朋. 轻量级分组密码 ESF 的差分故障攻击. 网络安全技术与应用, 2016, (1): 99–100. [doi: 10.3969/j.issn.1009-6833.2016.01.067]

(校对责编: 孙君艳)