积分攻击改进——随机线性区分与密钥恢复攻击

杜少宇
（密码科学技术全国重点实验室，北京 100878）

摘 要：在 4 轮 AES 的积分攻击和碰撞攻击的基础上，提出了一种利用明文和中间状态的某些分组之间线性偏差分布的不均匀性的针对 4 轮 SP 结构分组密码的随机线性区分攻击。进一步结合预计算，提出了对 4 轮 AES 类分组密码的密钥恢复攻击。对 LED-64 算法给出了具体区分攻击和密钥恢复攻击的结果。其中，对于 1-Step 的 LED-64 算法，在数据复杂度为 2^8，计算复杂度为 2^{16} 次基本运算的条件下，区分成功的概率是 85%；对于 2-Step 的 LED-64 算法，相关密钥条件下的密钥恢复攻击的计算复杂度为 2^{14} 次基本运算，数据复杂度为 2^8，预计算存储复杂度为 2^{38} 个半字节。

关键词：积分攻击；区分攻击；分组密码分析；AES；LED
中图分类号：TN92
文献标志码：A
DOI: 10.11959/j.issn.1000− 436x.2023085

Improved integral attack——random linear distinguish and key recovery attack

DU Shaoyu
State Key Laboratory of Cryptology, Beijing 100878, China

Abstract: Based on the integral attack and collision attack of four rounds of AES, a random linear distinguish attack against four rounds of SP block ciphers was proposed, which took advantage of the non-uniformity of linear biases' distribution between some blocks of plaintext and inner state. Combined with precomputation, a key recovery attack against four rounds of AES-like block ciphers was proposed. For LED-64, the results of distinguish attack and key recovery attack were given. Therein for LED-64 of 1-Step, the probability of successful distinguish attack is 85% under the condition that the data complexity is 2^8 and the computational complexity is 2^{16} basic operation. For LED-64 of 2-Step, the calculation complexity of the key recovery attack under the condition of related key is 2^{14} basic operation, the data complexity is 2^8, and the precomputation storage complexity is 2^{38} half bytes.

Keywords: integral attack, distinguish attack, block cipher analysis, AES, LED

0 引言

近年来，为满足射频识别（RFID, radio frequency identification）、智能卡、无线传感器等资源受限环境下的安全需求，轻量级密码随之诞生，并倍受关注。学者提出了一系列轻量分组密码算法，如 PRESENT[1]、LED[2]、LBlock[3]、PRINT-cipher [4]、KLEIN[5]和 SIMON[6]等。

LED[2]是在 CHES 2011 会议提出的，LED 算法分组长度为 64 bit，支持 64/128 bit 的密钥长度，采用 SP 结构，加密轮数为 32 轮。算法第一轮前进行一次轮密钥加，之后每 4 轮进行一次轮密钥加操作。算法的一个 Step 包括 4 轮，每一轮包括轮常量加、S 盒、行移位和列混淆 4 个操作。为提高加密速度及减小硬件实现规模，LED 算法没有密钥扩展算法，轮密钥为初始密钥。实现只需 966 个门电路，是同类分组密码中最少的，适于硬件实现，且保留了合理的软件实现能力。算法状态采用 GF(2^4)上的 4×4 矩阵，每个元素为 4 bit。

积分分析是由 Square 攻击发展而来的，Square

收稿日期：2023−01−09；修回日期：2023−03−21

攻击[7]是对 AES 最有效的攻击之一。近年来，积分分析广泛发展，已经应用于很多密码算法，如 Rijndael[8]、ARIA[9]、Serpent[10]、AES[11]、SIMECK[12]、TWINE[13]、Camellia [14-15]等。随着可分性自动化搜索工具的逐渐成熟[16-17]，积分分析又一次占据了分析方法的主导位置。

已有的针对 LED-64 算法攻击方法包括中间相遇攻击[18]、利用结构特性的 multibridge 攻击[19]和 slidex 攻击[20]、利用差分特性的时空折中攻击[21]等。对于 2-Step 的 LED-64 算法，已有的最好结果的计算复杂度为 2^{56} 次基本运算，数据复杂度为 2^8，存储复杂度为 2^{11} 个半字节。

本文基于 4 轮 AES 的积分攻击和碰撞攻击的思想，提出一种利用明文和中间状态的某些分组之间线性偏差分布的不均匀性的随机线性区分攻击。结合预计算可对 4 轮 SP 结构分组密码进行密钥恢复攻击。对于 1-Step 的 LED-64 算法，在数据复杂度为 2^8，计算复杂度为 2^{16} 次基本运算的条件下，区分成功的概率是 85%；对于 2-Step 的 LED-64 算法，相关密钥条件下的密钥恢复攻击的计算复杂度为 2^{14} 次基本运算，数据复杂度为 2^8，预计算存储复杂度为 2^{38} 个半字节。

1 积分攻击和碰撞攻击回顾

AES 算法 4 轮单 block 积分路径如图 1 所示，轮数为 4 轮。图 1 中，A 表示 Active，即所有取值都取到；B 表示 Balance，即每 1 bit 取 0 和取 1 的总数相等；?表示分布未知，即不具有平衡性或者遍历性。

虽然第四轮 S 盒之后的分布是未知的，不能直接通过求和为零进行区分，但是由于轮数低以及输入之间的不随机性（主要是不平衡性），这些未知分布仍存在许多规律。碰撞攻击就利用了第四轮 S 盒变换之后的未知分布，进行区分攻击和密钥恢复。

AES 算法 4 轮碰撞攻击的路径如图 2 所示，其中，Y 表示第一轮行移位之后的状态，Z、R、S、T 分别表示第一轮～第四轮的输出，X 表示积分。经过分析和推导，可以得出 s（S 的第一个 block）关于 y（Y 的第一个 block）的函数 $s^c[y]$ 由 4 个依赖于 c（图 2 中的 c_0,c_1,c_2 记作 c）的字节决定。由生日攻击的原理，给定一个包含 2^{16} 个 c 元素的集合 C（其他 block 的明文全部相同），则以不可忽视的概率存在 2 个不同的三元组 (c',c'')，使 $s^{c'}[y]=s^{c''}[y]$ 对于任意 $y\in\{0,1,\cdots,255\}$ 都成立，即 2 个函数相同。碰撞的检测通过对第四轮输出的 4 个字节 t_0,t_1,t_2,t_3 做逆列混合运算后，对比所得值是否相同。如果对于任意 $y\in\{0,1,\cdots,255\}$，计算出的值都相等，则 (c',c'') 发生了碰撞。经实验验证，16 个随机 y 值的对比结果为相同就能够确定碰撞的发生。

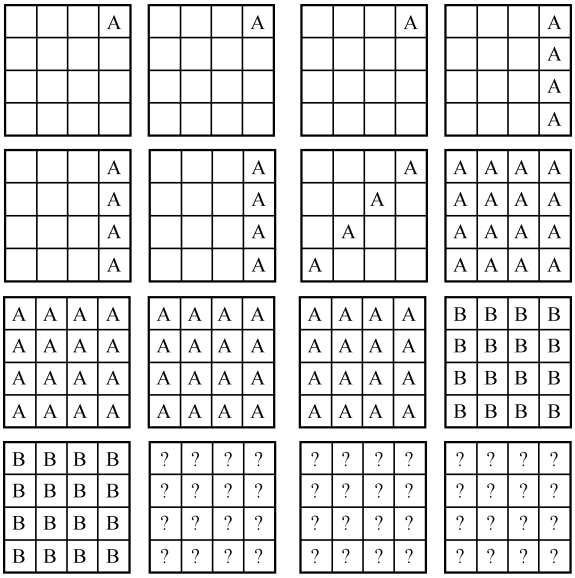

图 1 AES 算法 4 轮单 block 积分路径

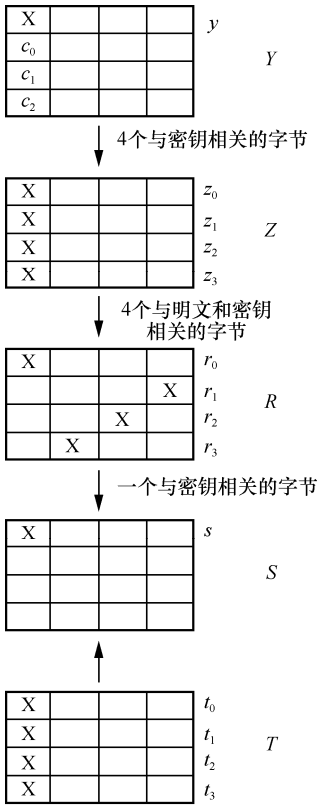

图 2 AES 算法 4 轮碰撞攻击的路径

2　改进方法

碰撞攻击中给出了将第三轮输出中的 s 作为输出、第一轮的输入中的 y 作为输入的函数 $s^c[y]$ 与第一轮输入的 $c(c_0,c_1,c_2)$ 之间的关系。实际上，s 与 y 的之间的函数 $s[y]$ 是由密钥以及明文的剩余 15 B 一起决定的。具体地说，$s[y]$ 是由 2 个只依赖于密钥的字节和 8 个同时依赖于明文和密钥的字节决定的。这 8 B 中有 4 B 完全依赖于 $c(c_0,c_1,c_2)$，另 4 B 依赖于其他部分。本文 2.1 节给出从 y 到第四轮列混淆之前的 block Z 的推导，同时 Z 可以从 4 轮缩减算法的密文 O 逆列混淆求得。则与 y 和 O 有关的函数 $Z(y)$ 是由 11 B 决定的，这意味着共有 2^{88} 种 $Z(y)$ 函数。基于这一改进，本文提出了新的随机线性区分攻击及相应的密钥恢复攻击。

本节所用变量标记如下。

1) $x_i, i=0,\cdots,15$ 表示第一轮输入的 16 个 block。

2) 第一轮行移位之后 block 的标号表示其与行移位前 block 的对应关系。

3) c_1,c_2,c_3,c_4 分别表示与第一轮列混淆之后第四列的第四行、第三行、第二行和第一行的 block 有关的变量。

4) c_5,c_6,c_7,c_8 分别表示与第二轮列混淆之后第四行第三列、第三行第二列、第二行第一列和第一行第二列的 block 有关的变量。

5) Z 表示第四轮行移位后第一行第四列的 block。

6) 记使用的 S 盒为 S，记 4 轮缩减轮算法的密文的 16 个 block 为 $O_i, i=0,\cdots,15$。

2.1　可碰撞函数的推导

4 轮 AES 传播路径如图 3 所示。记第一轮输入的 x_3 位置为 y，即函数的输入。经过一轮加密 Round1 和第二轮的轮密钥加之后，最后一列的分组可表示为 4 个关于 y 的函数，分别为

$$2S(y\oplus k_3^0)\oplus c_4,\ \ S(y\oplus k_3^0)\oplus c_3$$

$$S(y\oplus k_3^0)\oplus c_2,\ \ 3S(y\oplus k_3^0)\oplus c_1$$

其中，c_1,c_2,c_3,c_4 是 4 个由 4,9,14 这 3 个位置的字节决定的变量。这里初始的白话密钥记为 k^0，k_j^i 是第 i 轮轮密钥的第 j 个 block。

经过第二轮加密 Round2 和第三轮的轮密钥加之后，输出的第一行第四列、第二行第一列、第三行第二列和第四行第三列位置的字节仍可以表示为 4 个关于 y 的函数，分别为

$$2S[2S(y\oplus k_3^0)\oplus c_4]\oplus c_8,\ \ S[3S[y\oplus k_3^0]\oplus c_1]\oplus c_7$$

$$2S[S(y\oplus k_3^0)\oplus c_2]\oplus c_6,\ \ S[S[y\oplus k_3^0]\oplus c_3]\oplus c_5$$

其中，c_5,c_6,c_7,c_8 是 4 个依赖于第二轮行移位后 12 个字节（灰色和黑色标记）的常数。

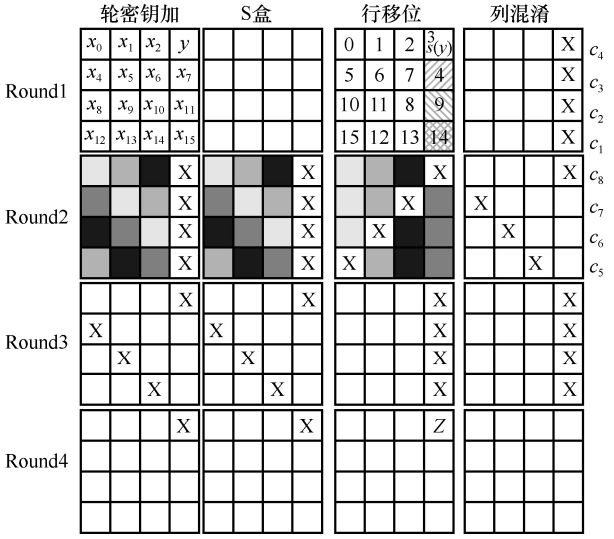

图 3　4 轮 AES 传播路径

经过第三轮加密 Round3，第二轮输出的 4 个标记的字节（第一行第四列、第二行第一列、第三行第二列、第四行第三列的字节）在第三轮行移位后集中在第四列，则第三轮输出的最后一列的 4 B 都可以表示为关于 y 的函数，具体如下。

第一行第四列表示为

$$\begin{aligned}&2S\{2S[2S(y')\oplus c_4]\oplus c_8\}\oplus\\&3S\{S[3S[y']\oplus c_1]\oplus c_7\}\oplus\\&S\{2S[S[y']\oplus c_2]\oplus c_6\}\oplus\\&S\{S[S[y']\oplus c_3]\oplus c_5\}\end{aligned}\tag{1}$$

第二行第四列表示为

$$\begin{aligned}&S\{2S[2S(y')\oplus c_4]\oplus c_8\}\oplus\\&2S\{S[3S[y']\oplus c_1]\oplus c_7\}\oplus\\&3S\{2S[S[y']\oplus c_2]\oplus c_6\}\oplus\\&S\{S[S[y']\oplus c_3]\oplus c_5\}\end{aligned}\tag{2}$$

第三行第四列表示为

$$\begin{aligned}&S\{2S[2S(y')\oplus c_4]\oplus c_8\}\oplus\\&S\{S[3S[y']\oplus c_1]\oplus c_7\}\oplus\\&2S\{2S[S[y']\oplus c_2]\oplus c_6\}\oplus\\&3S\{S[S[y']\oplus c_3]\oplus c_5\}\end{aligned}\tag{3}$$

第四行第四列表示为

$$3S\{2S[2S(y')\oplus c_4]\oplus c_8\}\oplus$$
$$S\{S[3S[y']\oplus c_1]\oplus c_7\}\oplus$$
$$S\{2S[S[y']\oplus c_2]\oplus c_6\}\oplus$$
$$2S\{S[S[y']\oplus c_3]\oplus c_5\} \tag{4}$$

其中，$y'=y\oplus k_3^0$。

对第四轮 Round4，只考虑第一行第四列的输入（即式(1)）。在第四轮列混合之前，这个字节只经过了轮密钥加和一次 S 盒运算，如图 3 所示，记行移位后此字节为 Z，Z 可以表示为与 y 有关的函数，即

$$Z(y)=S(2S\{2S[2S(y\oplus k_3^0)\oplus c_4]\oplus c_8\}\oplus$$
$$3S\{S[3S[y\oplus k_3^0]\oplus c_1]\oplus c_7\}\oplus$$
$$S\{2S[S[y\oplus k_3^0]\oplus c_2]\oplus c_6\}\oplus$$
$$S\{S[S[y\oplus k_3^0]\oplus c_3]\oplus c_5\}\oplus k_3^3) \tag{5}$$

值得注意的是，$c_i, i=1,\cdots,8$ 不仅依赖于明文，同时包含轮密钥的信息。在密钥相同的条件下，c_i 由明文决定。但是在未知密钥的条件下，如果能够求得 c_i，结合已知的明文信息，则可以获得关于密钥的方程，通过求解方程能够恢复部分密钥的信息。

对于缩减为 4 轮的加密算法，行移位之后，Round4 需要经过一次列混合和一次轮密钥加后输出密文。记输出密文的最后一列分别为 O_3、O_7、O_{11}、O_{15}，由第四轮的输出反推 Z 为

$$Z=EO_3\oplus BO_7\oplus DO_{11}\oplus 9O_{15}\oplus k_3^{4'}$$

其中，E、B、D、9 分别表示十六进制数 0xE、0xB、0xD、0x9。则 $k_3^{4'}=Ek_3^4\oplus Bk_7^4\oplus Dk_{11}^4\oplus 9k_{15}^4$。

综上，对 AES 算法决定函数 $Z(y)$ 输入与输出之间关系的字节共有 11 个，即 $c_i, i=1,\cdots,8$、k_3^0、k_3^3 和 $k_3^{4'}$。这意味着从 y 到 Z 的函数总共有 2^{88} 种。假设攻击者有足够的计算能力和存储能力，那么可以进行随机线性区分攻击和相应的密钥恢复攻击。

2.2 随机线性区分攻击

预计算阶段。记函数的输入为 y，输出为 z，对 2^{88} 种函数遍历输入掩码 a 和输出掩码 b，$a,b\in\frac{F_2^8}{\{0\}}$，记录 $\Pr(a\cdot y\oplus b\cdot z=1)$ 与 $\frac{1}{2}$ 之间偏差绝对值的最大值，其中 $\cdot$ 表示内积。然后，对记录的 2^{88} 个最大值进行排序，选择其中的最小值作为偏差的门限。

对于任意一个给定的线性逼近 $a\cdot y\oplus b\cdot z$，k_3^0 和 $k_3^{4'}$ 并不影响偏差的大小，只影响偏差的正负。因此，预计算阶段只需要计算 2^{72} 种函数的偏差。

线上阶段。对于给定的密钥 Key，用它加密 2^8 个明文 P，这 2^8 个明文除了 x_3 遍历 0～255 外，其他字节取值全部相同。然后，计算对应的 y（第一轮输入的 x_3 位置的字节）和 Z（第四轮行移位之后的第一行第四列位置的字节）之间的所有线性逼近的偏差，如果最大值大于或等于门限值，则本次区分判定为加密算法，否则本次区分判定为随机算法。改变明文的剩余 15 B 的取值或密钥值，重复实验多次，如果每次区分都判定为加密算法，即每次实验的最大偏差都大于或等于门限值，则区分器判定结果为加密算法，否则判定结果为随机串。由于目前实验能力有限，对于 AES 算法，不能确定偏差的门限值也就不能确定随机线性区分攻击具体的复杂度和成功概率。

在已有有效区分器筛选出密码算法的基础上，能够进行 2.3 节所述的密钥恢复攻击。

2.3 密钥恢复攻击

预计算阶段。类似于随机线性区分攻击，取 k_3^0 和 $k_3^{4'}$ 为 0，存储 2^{72} 种函数对应的 $c_i, i=1,\cdots,8$ 和 k_3^3 为索引。并且对每个函数，存储按照 $a\|b=\{0x00,\cdots,0xff\}$ 的顺序对应的线性偏差 $\varepsilon=\Pr(a\cdot y\oplus b\cdot z=1)-\frac{1}{2}$ 的具体值。记存储每个函数的参数和偏差值的列表为 $\text{Table}_{\text{AES\_1}}$。

建立 $\text{Table}_{\text{AES\_2}}$，对于 $p\|q=\{0x00,\cdots,0xff\}$ 的每一对 $p\|q$ 值，按照 $a\|b=\{0x00,\cdots,0xff\}$ 的顺序，存储 $a\cdot p\oplus b\cdot q$ 的值（0 或者 1）。

此外，在预计算阶段建立 $c_i, i=1,\cdots,8$ 与对应的密钥和明文的方程。

线上阶段。对于给定的密钥 Key，用它加密 2^8 个明文 P，这 2^8 个明文除了 x_3 位置字节遍历 0～255 外，其他字节取值全部相同。然后，按照 $a\|b=\{0x00,\cdots,0xff\}$ 的顺序计算对应的 y 和 Z 之间的所有线性逼近的偏差列表 T_1。按照 T_1 查 $\text{Table}_{\text{AES\_1}}$，找出与 T_1 中每个值绝对值相同的函数，同时记偏差取值符号相反的 $a\|b$ 位置为 1，符号相同的 $a\|b$ 位置为 0，得到 T_2，使用 T_2 查 $\text{Table}_{\text{AES\_2}}$，如果有对应 $p\|q$ 满足 0、1 串相同，记录对应 $\text{Table}_{\text{AES\_1}}$ 的索引以及 p 与 q 的值并返回。否则继续查 $\text{Table}_{\text{AES\_1}}$ 中其他的索引。

在得到索引值 $c_i, i=1,\cdots,8$ 和 k_3^3 以及 p 与 q 的值之后，可以直接得到 k_3^0、k_3^3 和 k_3^4'的值，通过求

解 c_i 与明文和密钥关系的方程，又可以得到另外 8 B 的密钥信息。

对于 y 和 Z 之间对应的函数，可以恢复上述 11 B 的密钥信息。如果想恢复全部密钥信息，可选择明文和第四轮行移位后的其他位置的 block 作为输入和输出，预计算另一组 2^{72} 个函数进行攻击。

3　1-Step 的 LED 算法的随机线性区分

LED 算法使用的是基于 4 bit 分组的 SP 结构。LED 算法分为 LED-64 和 LED-128。下面分析对 LED-64 算法的随机线性攻击和对应的密钥恢复。LED-64 结构如图 4 所示。

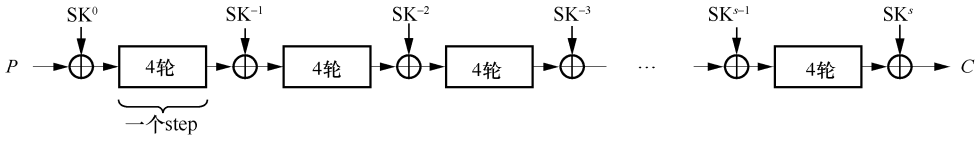

图 4　LED-64 结构

1-Step 的 LED-64 算法的 4 轮传播路径如图 5 所示，其与图 3 的区别在于，将每轮第一步的轮密钥加变为轮常数加，而密钥直接异或在 Round 1 前和 Round 4 后。这导致推导得到的影响 y 和 Z 之间关系的 $c_i, i=1,\cdots,8$ 、k_3^0 、k_3^3 和 $k_3^{4'}$ 变为如下 9 个 4 bit 的 block。

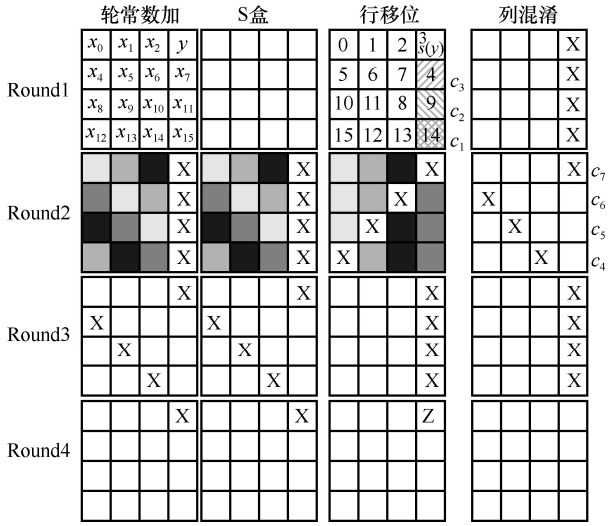

图 5　LED-64 算法的 4 轮传播路径

①c_1, c_2, c_3：分别依赖于 $x_4 \oplus k_4, x_9 \oplus k_9, x_{14} \oplus k_{14}$。

②c_4, c_5, c_6, c_7：同时依赖于 $x_i \oplus k_i$，$i = 0,1,2,5,6,7,10,11,8,15,12,13$。

③$M^{-1}(k_3, k_7, k_{11}, k_{15})_0$（$M^{-1}$ 表示列混淆的逆）和 k_3。

由于 $M^{-1}(k_3, k_7, k_{11}, k_{15})_0$ 和 k_3 仅影响线性逼近偏差的符号，并不影响偏差的大小。因此，对于所有的 $c_i, i=1,\cdots,7$ 的 $2^4 \times 7 = 2^{28}$ 种取值的每个对应的以 y 为输入、Z 为输出的 $Z(y)$ 函数，遍历输入输出掩码 a,b 计算对应的偏差，记录偏差绝对值的最大值。而后对 2^{32} 个最大偏差排序，最小值为门限值，记为 θ。

攻击的复杂度和成功概率不仅与实验得到的偏差门限 θ 相关，而且依赖于对随机函数进行随机线性分析时偏差的分布，即对于随机函数，遍历所有输入输出掩码 a,b（$a,b \in \frac{F_2^4}{\{0\}}$）时，最大偏差小于 θ 的概率。基于此，有以下 2 种攻击策略，它们复杂度和成功概率的计算略有不同，具体如 3.1 节和 3.2 节所述。

3.1　利用不同函数与随机函数做区分

Step1　对于给定的密钥 Key，用它加密 2^4 个明文 P，这 2^4 个明文除了 x_3 位置的 block 遍历 0～16 外，其他 block 取值全部相同。然后，计算对应的 y 和 Z 之间的所有 225 个线性逼近（$a,b \in \frac{F_2^4}{\{0\}}$）的偏差，如果最大值大于或等于门限值 θ，则本次区分判定为加密算法；否则判定为随机函数。

Step2　改变明文除了 x_3 之外的 15 个 block 的取值，重复 Step1。改变 n 次，每次剩余 15 个 block 的取值都不同。

Step3　如果每次区分都判定为加密算法，即每次实验的最大偏差都大于或等于 θ，则最终判定算法为 LED-64 加密算法；否则判定为随机函数。

3.2　相同函数与随机串的区分

另一种提高数据复杂度以减小错误概率 p_α 的方式如下，只对同一个函数（即相同 $c_i, i=1,\cdots,7$ 决定函数）统计线性偏差，这可通过同时改变 x_i 和 k_i 的值（$i \neq 3,7,11,15$），使 $x_i \oplus k_i$ 的值相同来保证。

Step1′　对于给定的密钥 Key，用它加密 2^8 个明文 P，这 2^8 个明文除了 x_3 遍历 0～255 之外，其他字节取值全部相同。同时存储所有的明文和密文。

Step2′　同时改变 x_i 和 k_i 的值（$i \neq 3,7,11,15$）使

$x_i \oplus k_i$ 的值相同，重复 Step1′。改变 n 次。

Step3′　对所有的明文和密文，计算对应的 y 和 Z 之间的 225 个线性逼近（$a,b \in \frac{F_2^4}{\{0\}}$）的偏差。如果每个逼近的偏差都小于门限值 θ，则判定为随机函数；否则判定为 LED-64 算法。

4　1-Step 和 2-Step 的 LED 算法的密钥恢复

4.1　1-Step 的 LED-64 算法单密钥的密钥恢复

1) 预计算阶段

Step0.1　设置 $M^{-1}(k_3,k_7,k_{11},k_{15})_0$ 和 k_3 为 0，遍历 $c_i, i=1,\cdots,7$，得到 2^{28} 个输入为 y、输出为 Z 的函数。以 $c_i, i=1,\cdots,7$ 作为索引，每个函数存储按照 $a\|b=\{0\text{x}00,\cdots,0\text{x}ff\}$ 的顺序对应的线性偏差为 $\varepsilon=\Pr(a\cdot y\oplus b\cdot Z=1)-\frac{1}{2}$ 的序列。生成的表记为 Table_{1_1}。

Step0.2　将逆列混合变换后的等价密钥的其他 block（位置为 U）设置为 0，同时将 k_3 设置为 0，能够得到 2^{28} 个输入为 y、输出为 U 的函数。同样以 $c_i^U, i=1,\cdots,7$ 为索引，重复 Step0.1，生成的表记为 Table_{1_2}。变换输出 block 的位置 n 次，依次记生成的表为 Table_{1_i}，$i=1,\cdots,n$。

Step0.3　对于 $p\|q=\{0\text{x}00,\cdots,0\text{x}ff\}$ 的每一个 $p\|q$，按照 $a\|b=\{0\text{x}00,\cdots,0\text{x}ff\}$ 的顺序，存储 $a\cdot p\oplus b\cdot q$（0 或者 1）的序列。生成的表记为 Table_2。

2) 线上阶段

Step1″　对于给定的密钥 Key，用其加密 2^4 个明文 P，这 2^4 个明文除 x_3 遍历 0～15 外，其他字节取值全部相同。

Step2″　按照 $a\|b=\{0\text{x}00,\cdots,0\text{x}ff\}$ 的顺序计算对应的 y 和 Z 之间的所有线性逼近的偏差列表 T_{1_1}。

Step3″　按照 T_{1_1} 查询 Table_{1_1}，找出与 T_{1_1} 中数据绝对值相同的序列，同时记与序列中偏差取值符号相反的 $a\|b$ 位置为 1，相同的为 0，得到 T_{2_1}。使用 T_{2_1} 查询 Table_2，如果存在 $p\|q$ 对应的序列与 T_{2_1} 相同，则同时记录 Table_{1_1} 的索引和 p、q 的值并返回；否则，继续查询 Table_{1_1}。重复上述步骤，直到找到符合条件的值。

Step4″　得到索引值 $c_i, i=1,\cdots,7$ 以及 p、q 的值之后，可以建立密钥和明文的方程，求解方程能够得到 36 bit 的密钥信息。

Step5″　改变输出 block 的位置，重复 Step2″～Step4″，直到恢复出所有 64 bit 的密钥信息。

函数输出位置选择的不同能够恢复出的密钥信息不同。对于同一列的输出，它们对应的以 y 为输入的 4 个函数分别依赖的 k_3 和 $c_i, i=1,\cdots,7$ 是完全相同的，也就是说将输出变换到同一列的另外一个位置时，只能额外获得 4 bit 的密钥信息。并且由于输入引入积分的位置固定为 x_3，因此即使改变输出位置到其他列，函数依赖的 $c_i, i=1,\cdots,7$ 和 k_3 也完全相同，此时只能额外获得 20 bit 的信息。这种常数的相关性虽然降低了获得的信息量，但是在实际操作中可以有效地减少查表的复杂度。平衡预计算阶段和线上阶段查表与解方程的复杂度，选择 $n=4$，其中最后一列选择 3 个 block 作为输出，其他三列选择一个 block。

如果明文的具体值未知，那么恢复出来的 64 bit 信息就是明文异或密钥之后的状态。

4.2　2-Step 的 LED-64 算法的相关密钥恢复

2-Step 的 LED-64 算法结构如图 6 所示。2-Step 算法的相关密钥恢复攻击的预计算过程与 1-Step 相同。线上阶段的攻击过程如下。

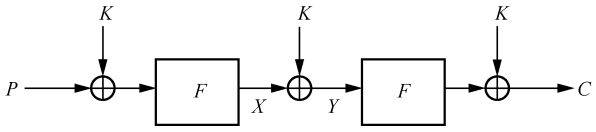

图 6　2-Step 的 LED-64 算法结构

Step1*　对明文和密钥同时选择相同的积分，积分位置在 x_3。2-Step LED-64 算法的积分传播如图 7 所示。由于明文和密钥的积分相同，1-Step 后输出的 16 个 X 完全相同。

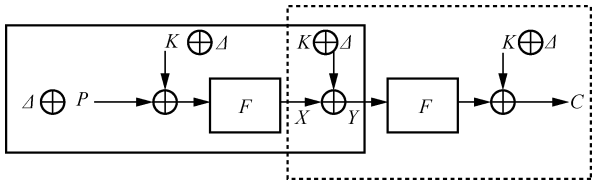

图 7　2-Step LED-64 算法的积分传播

Step2*　图 7 中虚线框的部分可以等价为图 8 中的单轮 LED-64。已知 Δ 和 C'，就可以利用单轮的 LED-64 算法的密钥恢复攻击恢复 $X\oplus K=Y$ 的值。

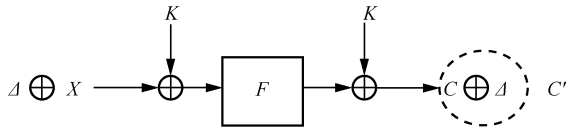

图 8　图 7 中虚线框的近似

Step3* 得到 Y 之后，图 7 中实线框部分等价于图 9 中的单轮 LED-64 加密。

Step4* 遍历 P 在 x_3 的取值，重复 Step1*～Step3*。得到 $P\oplus\Delta$ 用 K 加密一轮的所有输出 Y，再利用单轮的密钥恢复攻击来恢复 K。

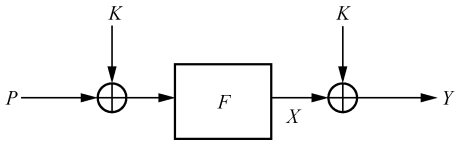

图 9　图 7 中实线框的近似

5　复杂度分析与实验验证

LED 算法的函数 $Z(y)$以 y 为输入，z 为输出，其由 $c_i, i=1,\cdots,7$ 决定，共有 2^{28} 种。$Z(y)$对应的 2^{32} 个偏差的最小值经实验验证为 2^{-2}，即门限值 $\theta=2^{-2}$。

5.1　区分攻击复杂度与实验验证

针对 3.1 节的区分策略，对于一个取自 LED-64 算法的由 7 个常数 $c_i, i=1,\cdots,7$ 确定的函数而言，计算出的 $(2^4-1)(2^4-1)$ 种偏差中一定存在偏差大于门限值 2^{-2}，也就是说，按照这种判定方式，将 LED-64 的输出误判为随机串的错误概率 p_β=0。

而将随机函数误判为 LED-64 的输出的错误概率 p_α 取决于重复次数 n 和 p^*，其中 p^* 表示随机选取一个布尔函数 $F:F_2^4\to F_2$，函数的 15 个（$a\in\dfrac{F_2^4}{\{0\}}$）线性逼近的偏差全部小于 2^{-2} 的概率，记为 $p*$。实验可得，$p^*=\dfrac{16\,626}{65\,536}\approx 0.25$。

对于一个(4,4)的向量值布尔函数 f，如果要求其 225 个非平凡的线性逼近的偏差都小于 2^{-2}，相当于要求由它构成的 2^4-1 个 $F_2^4\to F_2$ 的布尔函数 $f_v=v\cdot f$，$v\in\dfrac{F_2^4}{\{0\}}$ 的所有 15 个线性逼近的偏差全部小于 2^{-2}。如果选择 n 次不同的函数实验后将随机函数判断为 LED-64 算法，这意味着 n 次子区分中每次都将随机函数判定为 LED-64 算法，也就是说每次子区分用到的 15 个布尔函数中至少存在一个布尔函数的 15 个线性逼近中存在偏差大于 2^{-2} 的逼近。因此，错误概率为 $p_\alpha<(1-p^{*15})^n=\left(1-\left(\dfrac{16\,626}{65\,536}\right)^{15}\right)^n$。当 $n=2^{31}$ 时，$p_\alpha<0.08$。则当选择 2^{31} 个不同的函数，即随机选择 2^{31} 种明文和密钥组合时，成功概率为 $1-p_\alpha-p_\beta=0.92$，复杂度为 $2^{31}(2^4+2^8)=O(2^{39})$ 次基本运算，数据复杂度为 $2^{31}2^4=2^{35}$。

针对 3.2 节的区分策略，可以用定理 1 卡方分布计算攻击的成功概率和数据复杂度，具体推导过程参见文献[22]。

定理 1　已知上述区分器对长度为 $N=n\cdot 2^4$ 的样本进行区分，区分的概率空间的维度 $k=2$，偏差门限的大小 $\varepsilon_{\text{thr}}=2^{-2}$，则区分所有的显著性参数 χ_{thr}、错误概率和需要的样本数 N 有如下关系

$$\chi^2_{\text{thr}}\in(\mu;\mu+2C\sigma)$$

χ^2_{thr} 的选择要保证两类错误概率相近。

$$p_\alpha=1-(Q_{\chi^2}(\chi^2_{\text{thr}};\mu))^m$$

$$p_\beta\leqslant Q_{NC\chi^2}\left(\frac{\chi^2_{\text{thr}}}{\xi_1};\mu;2C\sigma\right)$$

$$N>\frac{\xi_0 C\sqrt{k-1}}{\sqrt{2}\varepsilon^2_{\text{thr}}}$$

其中，$\mu=k-1$；$\sigma=\sqrt{2(k-1)}$；$\xi_\tau=1+(-1)^\tau k\varepsilon$；$m=225$；$\varepsilon=\dfrac{1}{4}$；$C>0$ 为常数，且 C 的取值越大，错误概率越低。$Q_{\chi^2}(\chi^2_{\text{thr}};\mu)$ 由 MATLAB 中 $\text{chi2cdf}(\chi^2_{\text{thr}};\mu)$ 计算得到，$Q_{NC\chi^2}\left(\dfrac{\chi^2_{\text{thr}}}{\xi_1};\mu;2C\sigma\right)$ 由 MATLAB 中的 $\text{ncx2cdf}\left(\dfrac{\chi^2_{\text{thr}}}{\xi_1};\mu;2C\sigma\right)$ 计算得到。

当 $C=15$ 时，选择 $\chi^2_{\text{thr}}=13$，则 $p_\alpha=0.067$，$p_\beta=0.083$，$N=2^4\times 2^4=2^8$。成功概率 $1-p_\alpha-p_\beta=0.85$。数据复杂度为 2^8，计算复杂度为 $O(2^8\times 2^8)=O(2^{16})$ 次基本运算。

区分攻击结果对比如表 1 所示。LED 算法的设计文档[2]指出，攻击者可以区分 3.75-Step 的 LED-64 算法与随机置换，但并未给出成功概率的计算方法和实验验证。本文基于 LED-64 算法门限值确定的实验结果，精确地给出了成功概率计算方法，与已有的攻击结果相比，明显提升了分析可靠性。

5.2　密钥恢复攻击复杂度分析

在单密钥条件下，对于 1-Step 的 LED-64 算法的密钥恢复攻击，预计算阶段的计算复杂度为

表 1　区分攻击结果对比

攻击假设	Step	计算复杂度	数据复杂度	存储复杂度/半字节	攻击成功概率	文献
单密钥	1	2^{39}	2^{35}	0	0.92	本文
选择密钥	1	2^{16}	2^{8}	0	0.85	本文
选择密钥	3.75	2^{16}	—	2^{16}	—	文献[2]
选择密钥	4	$2^{33.5}$	—	2^{32}	—	文献[20]
选择密钥	5	$2^{60.2}$	—	2^{61}	—	文献[20]
选择密钥	5	$2^{57.7}$	—	$2^{58.5}$	—	文献[23]

表 2　密钥恢复攻击结果对比

攻击假设	Step	计算复杂度	数据复杂度	存储复杂度/半字节	攻击成功概率	文献
单密钥	1	2^{10}	2^{4}	2^{38}	1	本文
选择密钥	2	2^{14}	2^{8}	2^{38}	1	本文
选择密钥	2	2^{56}	2^{8}	2^{11}	—	文献[18]
选择密钥	3	$2^{60.2}$	2^{49}	2^{60}	—	文献[21]
选择密钥	4	$2^{62.7}$	$2^{62.7}$	$2^{62.7}$	—	文献[19]
选择密钥	4	$2^{60.8}$	$2^{60.8}$	$2^{60.8}$	—	文献[23]

$4\times 2^{28}\times(2^4+2^8)+2^8\times 2^8=O(2^{38})$ 次基本运算；存储复杂度为 $4\times 2^{28}\times 2^8+2^8\times 2^8=O(2^{38})$，单位为半字节。线上阶段的数据复杂度为 2^4，计算复杂度为 $4\times 2^8=2^{10}$ 次基本运算（忽略查表和解方程的复杂度）。

在相关密钥条件下，对于 2-Step 的 LED-64 算法的密钥恢复攻击，在预计算阶段的复杂度及存储与 1-Step 的单密钥的恢复攻击相同。线上阶段的数据复杂度为 $2^4\times 2^4=2^8$，计算复杂度为 $2^4\times(4\times 2^8)+4\times 2^8=O(2^{14})$ 次基本运算（忽略查表和解方程的复杂度）。

密钥恢复攻击结果对比如表 2 所示。本文密钥恢复攻击基于对缩减轮 LED-64 算法某些位置的输入输出函数的全部刻画，意味着查表成功后恢复的密钥信息是确定的，与已有的基于结构或者差分路径的概率统计类的密钥恢复攻击相比具有更高的准确度。与文献[19]相似的相关密钥假设下，本文给出了对 2-Step 算法攻击成功概率为 1 的密钥恢复攻击，与已有的 2-Step 攻击结果相比，所需时间、存储复杂度和数据复杂度极大降低，线上阶段仅需 256 个明文，计算复杂度仅为 2^{14} 次基本运算。

6　结束语

本文对采用 SP 结构的 AES 和 LED 分组密码算法的安全性进行分析。在 4 轮 AEC 的积分攻击和碰撞攻击思想的基础上，提出了一种利用明文和中间状态的某些分组之间线性偏差分布的不均匀性的随机线性区分攻击和密钥恢复攻击。分析结果表明，本文提出的密钥恢复攻击对 2-Step 的 LED-64 算法具有更优的时间和存储复杂度。

参考文献：

[1] BOGDANOV A, KNUDSEN L R, LEANDER G, et al. PRESENT: an ultra-lightweight block cipher[C]//Proceedings of Cryptographic Hardware and Embedded Systems. Berlin: Springer, 2007: 450-466.

[2] GUO J, PEYRIN T, POSCHMANN A, et al. The LED block cipher[C]//Proceedings of Cryptographic Hardware and Embedded Systems. Berlin: Springer, 2011: 326-341.

[3] WU W L, ZHANG L. LBlock: a lightweight block cipher[C]// Proceedings of Applied Cryptography and Network Security. Berlin: Springer, 2011: 327-344.

[4] KNUDSEN L, LEANDER G, POSCHMANN A. PRINT cipher: a block cipher for IC-printing[C]//Proceedings of Cryptographic Hardware and Embedded Systems. Berlin: Springer, 2010: 16-32.

[5] LALLEMAND V, NAYA-PLASENCIA M. Cryptanalysis of KLEIN[C]//Proceedings of Fast Software Encryption. Berlin: Springer, 2014: 451-470.

[6] BEAULIEU R, TREATMAN-CLARKS, SHORS D, et al. The SIMON and SPECK lightweight block ciphers[C]//Proceedings of 2015 52nd ACM/EDAC/IEEE Design Automation Conference. Piscataway: IEEE Press, 2015: 1-6.

[7] DAEMEN J, KNUDSEN L, RIJMEN V. The block cipher square[C]//Proceedings of Fast Software Encryption. Berlin: Springer, 1997: 149-165.

[8] LI Y J, WU W L. Improved integral attacks on Rijndael[J]. Journal of Information Science and Engineering, 2011, 27(6): 2031-2045.

[9] LI Y J, WU W L, ZHANG L. Integral attacks on reduced-round ARIA block cipher[C]//Proceedings of Information Security Practice and

Experience. Berlin: Springer, 2010: 19-29.

[10] ZABA M R, RADDUM H, HENRICKSEN M, et al. Bit-pattern based integral attack[C]//Proceedings of Fast Software Encryption. Berlin: Springer, 2008: 363-381.

[11] XIA T F, ZHAO Z Y, LI W, et al. A new kind of integral cryptanalysis for the round-reduced AES[J]. Electrical Engineering and Computer Science, 2019, 3: 6-9.

[12] LI H, REN J J, CHEN S Z. Improved integral attack on reduced-round SIMECK[J]. IEEE Access, 2019, 7: 118806-118814.

[13] ELSHEIKH M, YOUSSEF A M. Integral cryptanalysis of reduced-round tweakable TWINE[C]//Proceedings of Cryptology and Network Security. Berlin: Springer, 2020: 485-504.

[14] DUO L, LI C, FENG K Q. Square like attack on Camellia[C]//Proceedings of Cryptology and Network Security. Berlin: Springer, 2007: 269-283.

[15] LI Y J, WU W L, ZHANG L T, et al. Improved integral attacks on reduced round Camellia[EB]. [2011-04-11].

[16] CHEN L L, WANG G L, ZHANG G Y. MILP-based related-key rectangle attack and its application to GIFT, Khudra, MIBS[J]. The Computer Journal, 2019, 62(12): 1805-1821.

[17] XIANG Z J, ZHANG W T, BAO Z Z, et al. Applying MILP method to searching integral distinguishers based on division property for 6 lightweight block ciphers[C]//Proceedings of Advances in Cryptology. Berlin: Springer, 2016: 648-678.

[18] ISOBE T, SHIBUTANI K. Security analysis of the lightweight block ciphers XTEA, LED and Piccolo[C]//Proceedings of Information Security and Privacy. Berlin: Springer, 2012: 71-86.

[19] MENDEL F, RIJMEN V, TOZ D, et al. Differential analysis of the LED block cipher[C]//Proceedings of Advances in Cryptology. Berlin: Springer, 2012: 190-207.

[20] NIKOLIĆ I, WANG L, WU S. Cryptanalysis of round-reduced LED[C]//Proceedings of Fast Software Encryption. Berlin: Springer, 2013: 112-129.

[21] DINUR I, DUNKELMAN O, KELLER N, et al. Key recovery attacks on 3-roundeven-mansour, 8-Step LED-128, and full AES2[C]//Proceedings of Advances in Cryptology. Berlin: Springer, 2013: 337-356.

[22] MAXIMOV A, JOHANSSON T. A linear distinguishing attack on scream[J]. IEEE Transactions on Information Theory, 2007, 53(9): 3127-3144.

[23] SUN L, WANG W, WANG M. More accurate differential properties of LED64 and Midori64[J]. IACR Transactions on Symmetric Cryptology, 2018(3): 93-123.

[作者简介]

杜少宇（1988– ），女，山东龙口人，博士，密码科学技术全国重点实验室助理研究员，主要研究方向为对称密码的侧信道分析。