第 46 卷 第 7 期 计 算 机 工 程 2020 年 7 月
Vol. 46 No. 7 Computer Engineering July 2020

·网络空间安全· 文章编号：1000-3428(2020)07-0136-07 文献标志码：A 中图分类号：TP393

LiCi 算法的基于比特积分攻击

信文倩，孙 兵，李 超

（国防科技大学 文理学院，长沙 410073）

摘 要： 为分析目前 LiCi 算法抵抗积分攻击的能力，利用基于比特的可分性质，结合 MILP 搜索工具对 LiCi 算法的积分区分器进行搜索。搜索得到最长轮数积分区分器为 12 轮积分区分器，利用 12 轮积分区分器对 LiCi 算法进行 13 轮积分攻击。该攻击能够恢复 17 比特密钥信息，攻击的数据复杂度约为 2^{63}，时间复杂度约为 2^{100}次 16 轮加密，存储复杂度约为 2^{41}。为了得到更长轮数的攻击结果，利用 10 轮积分区分器向后攻击 6 轮，对 LiCi 算法进行 16 轮积分攻击，攻击数据复杂度约为 $2^{63.6}$，时间复杂度约为 2^{173}次 16 轮加密，存储复杂度约为 2^{119}。积分攻击实验结果表明，13 轮 LiCi 算法不能抵抗积分攻击。

关键词： 轻量级分组密码算法；LiCi 算法；可分性质；混合整数线性规划；积分攻击

开放科学（资源服务）标志码（OSID）：

中文引用格式： 信文倩，孙兵，李超. LiCi 算法的基于比特积分攻击[J]. 计算机工程，2020，46(7)：136-142.

英文引用格式： XIN Wenqian，SUN Bing，LI Chao. Bit-based integral attack on LiCi algorithm[J]. Computer Engineering，2020，46(7)：136-142.

Bit-based Integral Attack on LiCi Algorithm

XIN Wenqian，SUN Bing，LI Chao

（College of Liberal Arts and Sciences，National University of Defense Technology，Changsha 410073，China）

【Abstract】 To analyze the current ability of LiCi algorithm to resist integral attacks，this paper uses the bit-based division property and the MILP search tool to search for the integral distinguisher of the LiCi algorithm. The obtained longest round of integral distinguisher is 12-round，and is used to perform 13 rounds of integral attacks that can recover 17-bit key information on the LiCi algorithm. The data complexity of the attack is about 2^{63}，the time complexity is about 2^{100} times of 16-round encryption，and the storage complexity is about 2^{41}. In order to obtain a longer round of attack results，a 10-round integral distinguisher is used for 6-round backward attacks，and a 16-round integral attack is performed on the LiCi algorithm. The data complexity of the attack is about $2^{63.6}$，the time complexity is about 2^{173} times of 16-round encryption，and the storage complexity is about 2^{119}. Experimental results of integral attacks show that the 13-round LiCi algorithm cannot resist integral attacks.

【Key words】 lightweight block cipher algorithm；LiCi algorithm；division property；Mixed Integer Linear（MIL）programming；integral attack

DOI： 10.19678/j.issn.1000-3428.0055499

0 概述

随着信息时代的快速发展，物联网作为信息技术的重要组成部分，其通过智能感知、识别技术与普适计算等通信感知技术广泛应用于网络融合中。由于物联网中使用的微型计算设备的计算能力有限，因此为了保证信息安全，轻量级分组密码算法应运而生。轻量级分组密码算法是分组密码算法中的一种，相比普通的分组密码算法，该算法的分组长度相对较短，且算法结构简单，满足低耗能、低成本的需求。目前，轻量级分组密码算法主要有 LED[1]、LBlock[2]、PRESENT[3]、HIGHT[4]、SPECK[5]等算法，这些算法均具有结构简单、加解密一致、容易实现等优点。

2017 年，PATIL 等人[6]提出一个分组长度为 64 比特、密钥长度为 128 比特的轻量级分组密码算法——LiCi 算法。该算法结构类似于 MISTY 结构，在

基金项目： 国家自然科学基金“结构密码分析的原理及应用研究”(61772545)；国家自然科学基金“分组密码算法的安全性分析”(61672530)。

作者简介： 信文倩(1995—)，女，硕士研究生，主研方向为分组密码分析；孙 兵，副教授、博士；李 超，教授、博士、博士生导师。

收稿日期： 2019-07-16 **修回日期：** 2019-08-20 **E-mail：** Wenqian_Xin@163.com

单轮加密结构中,非线性组件 S 盒会影响到加密结构的两支。相较于普通 Feistel 结构分组密码算法,LiCi 算法具有扩散性快等优势。同时,相比于 SP 结构分组密码算法,LiCi 算法对非线性组件输出结果进行复用,使之结构更加简单,具有占用面积小等特性。文献[7]对 LiCi 算法抵抗不可能差分攻击的能力进行了介绍。然而,关于 LiCi 算法抵抗积分攻击的能力目前尚不清楚,因此,本文利用积分攻击方法对该算法进行分析。

积分攻击是 KNUDSEN 等人[8]在总结 Square 攻击、Multiset 攻击、Saturation 攻击的基础上提出的一种密码分析方法。该攻击方法是一种选择明文攻击方法,与差分攻击[9]、线性攻击[10]、代数攻击[11]同为目前密码学界公认的最有效的几种分析方法。结合故障分析的思想,差分故障分析[12]、代数故障分析[13]、积分故障分析[14]等分析方法也受到密码学者们的广泛关注。积分分析方法提出后,其在 AES[15]、Camellia[16]、FOX[17]、PRINCE[18]等算法中进行不同程度的分析应用。文献[19-20]提出比特的可分性质后,结合 MILP 搜索工具,利用可分性质对 MISTY1 进行全轮攻击。同时,文献[21]也基于比特的可分性质结合 MILP 搜索工具,进一步提升 LBlock[2]、PRESENT[3]、SIMECK[22]等算法的积分分析结果。

本文基于比特的可分性质,结合 MILP 搜索工具对 LiCi 算法的积分区分器进行搜索。利用搜索得到的最长轮数的 12 轮积分区分器对 LiCi 算法进行 13 轮积分攻击,恢复 17 比特密钥信息,同时,为了得到更高轮数的攻击结果,利用 10 轮积分区分器向后攻击 6 轮,对 LiCi 算法进行 16 轮积分攻击。

1　LiCi 算法介绍

1.1　LiCi 算法加密过程

LiCi 算法分组长度为 64 比特,密钥规模为 128 比特,其迭代轮数为 31 轮。LiCi 算法的单轮加密结构如图 1 所示,基本操作包括字节替换、异或、密钥加、循环移位等步骤。字节替换是该算法中唯一的非线性组件,由 8 个并列的 4 进 4 出的 S 盒构成。

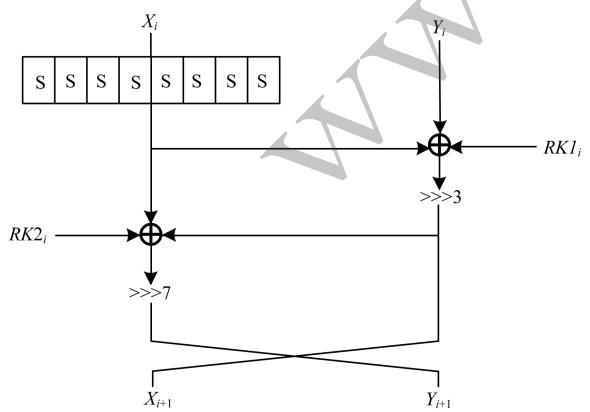

图 1　LiCi 算法单轮加密结构

Fig. 1　Single-round encryption structure of LiCi algorithm

加密过程　设第 i 轮输入为 $X_i, Y_i, i=0,1,2,\cdots,29,30$,分别代表第 i 轮输入的左分支和右分支;输出为 X_{i+1}, Y_{i+1},分别代表输出的左分支和右分支。状态 X_i, Y_i 到状态 X_{i+1}, Y_{i+1} 的迭代过程表示如下:

$$X_{i+1}=[\mathrm{S}[X_i]\oplus Y_i\oplus RK1_i]<<<3$$
$$Y_{i+1}=[\mathrm{S}[X_{i+1}]\oplus X_{i+1}\oplus RK2_i]>>>7 \quad (1)$$

其中,$RK1_i, RK2_i$ 均为轮密钥,$X_{i,(31,30,\cdots,2,1,0)}$ 和 $Y_{i,(31,30,\cdots,2,1,0)}$ 分别表示输入状态的 64 个比特的标号,如 $X_{i,31}$ 表示第 i 轮左支输入的最高比特,$Y_{i,0}$ 表示第 i 轮右支输入的最低比特。

1.2　LiCi 算法密钥扩展方案

密钥扩展方案:种子密钥长度为 128 比特,记为 $K=K_{127}K_{126}\cdots K_2K_1K_0$,$RK1_i, RK2_i$ 均表示第 i 轮轮密钥,其中 $RK1_i$ 应用于第 i 轮右支加密,$RK2_i$ 应用于第 i 轮左支加密。轮密钥生成过程如下:

1) $K=K_{127}K_{126}\cdots K_2K_1K_0$

2) $RK1_i=K_{31}K_{30}K_{29}\cdots K_2K_1K_0$, $RK2_i=K_{63}K_{62}\cdots K_{33}K_{32}, i\in\{0,1,2,\cdots\}$

3) $K_{\mathrm{new}}=K<<<13=K_{114}K_{113}\cdots K_1K_0K_{127}K_{126}\cdots K_{115}$

4) $K=K_{\mathrm{new}}$

5) $[K_3K_2K_1K_0]_{\mathrm{new}}=\mathrm{S}[K_3K_2K_1K_0]$

$[K_7K_6K_5K_4]_{\mathrm{new}}=\mathrm{S}[K_7K_6K_5K_4]$

$[K_{63}K_{62}K_{61}K_{60}K_{59}]_{\mathrm{new}}=[K_{63}K_{62}K_{61}K_{60}K_{59}]\oplus i, i\in\{0,1,2,\cdots\}$

6) $[K_3K_2K_1K_0]=[K_3K_2K_1K_0]_{\mathrm{new}}$

$[K_7K_6K_5K_4]=[K_7K_6K_5K_4]_{\mathrm{new}}$

$[K_{63}K_{62}K_{61}K_{60}K_{59}]=[K_{63}K_{62}K_{61}K_{60}K_{59}]_{\mathrm{new}}$

7) 经过 3) ~ 6) 得到新的 K,返回 1),经过 2) 得到新的轮密钥。

轮密钥分析:若已知第 i 轮密钥 $RK2_i, RK1_i$(其中 $i\geqslant 5$),根据密钥扩展方案可以得知($RK2_{i-4}$, $RK1_{i-4}$),…,($RK2_i, RK1_i$)之间的关系。

假设已知($RK2_i, RK1_i$) = ($K_{63}\cdots K_{33}K_{32}, K_{31}\cdots K_1K_0$),则根据密钥扩展方案中轮密钥生成方案可知($RK2_{i-1}, RK1_{i-1}$)和($RK2_i, RK1_i$)之间的某些比特信息等价,通过密钥生成方案中 3) ~ 6) 可知,($RK2_{i-1}, RK1_{i-1}$)与($RK2_i, RK1_i$)相比,新引入 13 比特信息。同理,可以分析($RK2_{i-2}, RK1_{i-2}$)和($RK2_{i-1}, RK1_{i-1}$),…,($RK2_{i-3}, RK1_{i-3}$)和($RK2_{i-4}, RK1_{i-4}$)之间的关系,5 轮轮密钥总共包含 116 比特密钥信息,6 轮轮密钥总共包含种子密钥 128 比特密钥信息。通过上述轮密钥分析,利用轮密钥之间的信息等价关系,在猜测密钥过程中可以降低密钥猜测量。

1.3　LiCi 算法的 S 盒性质

LiCi 算法 4 比特 S 盒如表 1 所示,输入为 x,输出为 $\mathrm{S}(x)$。

表 1 LiCi 算法 S 盒

Table 1 S box of LiCi algorithm

输入 x	输出 S(x)	输入 x	输出 S(x)
0	3	8	c
1	f	9	4
2	e	a	b
3	1	b	2
4	0	c	9
5	a	d	7
6	5	e	6
7	8	f	d

采用文献[23]中求 S 盒布尔函数表达式的方法来求解 LiCi 算法 4 比特 S 盒代数表达式。

性质 1(S 盒代数表达式) 设 S 盒输入为 $x=(x_3,x_2,x_1,x_0)$,输出为 $y=(y_3,y_2,y_1,y_0)$,则 x 和 y 之间的关系表达式如下:

$$y_3=x_0+x_1+x_3+x_1x_2+x_1x_3$$

$$y_2=x_0+x_1+x_3+x_0x_2+x_0x_3+x_2x_3+x_0x_1x_2$$

$$y_1=1+x_2+x_3+x_0x_1+x_0x_2+x_1x_3+x_2x_3+x_0x_1x_3$$

$$y_0=1+x_1+x_2+x_3+x_0x_1$$

例如,输入 $x_3x_2x_1x_0=0001$,经过 S 盒输出 $y_3y_2y_1y_0=1111$。

2 基于比特的可分性质和 MILP 方法

2.1 基于比特的可分性质

定义 1(比特积函数 $\pi_u(x)$ 和 $\pi_U(X)$[24]) 多重集的可分性质可通过比特积函数进行评估,比特积函数的定义如下:

令 $\pi_u(x):F_2^n\to F_2$ 表示 $u\in F_2^n$ 的比特积函数。令 $x\in F_2^n$ 表示输入,$\pi_u(x)$ 表示满足 $u[i]=1$ 的 $x[i]$ 代数正规型,定义为:

$$\pi_u(x):=\prod_{i=1}^{n}x[i]^{u[i]}$$

其中,$x[i]^1=x[i]$,$x[i]^0=1$。

令 $\pi_U(X):(F_2^{n_1}\times F_2^{n_2}\times\cdots\times F_2^{n_m})\to F_2$ 表示 $\boldsymbol{U}\in(F_2^{n_1}\times F_2^{n_2}\times\cdots\times F_2^{n_m})$ 的比特积函数。令 $\boldsymbol{X}\in(F_2^{n_1}\times F_2^{n_2}\times\cdots\times F_2^{n_m})$ 表示输入,$\pi_U(\boldsymbol{X})$ 定义为:

$$\pi_U(X):=\prod_{i=1}^{n}\pi_{U_i}(X_i)$$

定义 2(可分性[19]) 设多重集 $X\in(F_2^{n_1}\times F_2^{n_2}\times\cdots\times F_2^{n_m})$,若 X 具有可分性 $D_K^{n_1,n_2,\cdots,n_m}$,其中 $\boldsymbol{K}$ 是一个 m 维向量,其第 i 个元素取值为 $0\sim n_i$。若存在 $\boldsymbol{k}\in\boldsymbol{K}$,使得 $W(u)\geqslant k$,则 $\bigoplus_{x\in X}\pi_u(x)$ 为未知,反之,$\bigoplus_{x\in X}\pi_u(x)$ 为 0。

定义 3(可分路径[21]) 考虑可分析性质的传播 $\{\boldsymbol{k}\}\triangleq K_0\to K_1\to K_2\to\cdots\to K_r$,对任意向量 $\boldsymbol{k}_{i+1}\in\boldsymbol{K}_{i+1}$ 使得 $\boldsymbol{k}_i$ 能传播到 $\boldsymbol{k}_{i+1}$,$i\in\{0,1,\cdots,r-1\}$,则 $(\boldsymbol{k}_0\to\boldsymbol{k}_1\to\cdots\to\boldsymbol{k}_r)$ 为一条 r 轮可分路径。

上述内容是关于比特积函数、可分性和可分路径的介绍,下面对基于比特的可分性质经过复制操作、异或操作时的传播规则进行简要介绍,更多详情可参考文献[21,24]。

规则 1(复制操作) 令 x 为复制函数的输入值,y_0,y_1 为复制函数的输出值,其中 $(y_0,y_1)=(x,x)$。令 X 和 Y 分别表示输入多重集和输出多重集,假设多重集 X 有可分性 $D_{\{k\}}^1$,多重集 Y 可分析性为 $D_{K'}^{1\times1}$,则可分性传播只有以下 2 种情况:

$$\begin{cases}\boldsymbol{K}'=\{(0,0)\},k=0\\ \boldsymbol{K}'=\{(0,1),(1,0)\},k=1\end{cases}$$

规则 2(异或操作) 令 y_0,y_1 为异或函数的输入值,x 为异或函数的输出值,其中 $x=y_0\oplus y_1$。令 X 和 Y 分别表示输入多重集和输出多重集,假设多重集 X 有可分性 $D_{K'}^{1\times1}$,多重集 Y 可分析性为 $D_{\{k\}}^1$,则可分性传播有以下 4 种情况:

$$\begin{cases}\boldsymbol{K}'=\{(0)\},\boldsymbol{k}=(0,0)\\ \boldsymbol{K}'=\{(1)\},\boldsymbol{k}=(0,1)\\ \boldsymbol{K}'=\{(1)\},\boldsymbol{k}=(1,0)\\ \boldsymbol{K}'=\varnothing,\boldsymbol{k}=(1,1)\end{cases}$$

2.2 基本操作的 MILP 模型

基于比特的复制模型:假设输入可分性为 a,经过基于比特的复制操作输出可分性为 (a_0,a_1),记作 $a\to(a_0,a_1)$,其中 $a,a_0,a_1\in\{0,1\}$。a,a_0,a_1 之间的关系如下:

$$a-a_0-a_1\geqslant0,0\leqslant a_0\leqslant1,0\leqslant a_1\leqslant1$$

例如:$1\to(0,1)$ 或 $1\to(1,0)$,$0\to(0,0)$。

基于比特的异或模型:假设输入可分性为 (a_0,a_1),经过基于比特的异或操作输出可分性为 a,记作 $(a_0,a_1)\to a$,其中 $a,a_0,a_1\in\{0,1\}$。a,a_0,a_1 之间的关系如下:

$$a-a_0-a_1\geqslant0,0\leqslant a_0\leqslant1,0\leqslant a_1\leqslant1$$

例如:$(0,1)\to1$,$(1,0)\to1$,$(0,0)\to1$,但是输入可分性为 $(1,1)$,经过异或操作可分性传播会中断。

S 盒模型:利用文献[20]中的算法 2 可以得到 LiCi 算法 S 盒的可分性(详见开放科学(资源服务)标志码中附录部分),通过得到的 S 盒的可分性结合 SageMath 软件可得到 S 盒的线性不等式组。再利用文献[20]中的算法 1 对上述 S 盒线性不等式组进行简化,最终得到 LiCi 算法 S 盒的 15 个线性不等式(详见开放科学(资源服务)标志码中附录部分)。

基于比特的循环移位模型:假设输入 n 比特可分性为 $\boldsymbol{k}_{n-1},\cdots,\boldsymbol{k}_2,\boldsymbol{k}_1,\boldsymbol{k}_0$,其中 $\boldsymbol{k}_i\in\{0,1\}$。循环左移 j 位,输出为 $\boldsymbol{k}_{(i+n-j)\bmod n}$,$i\in\{n-1,\cdots,2,1,0\}$;循环右移 j 位,输出为 $\boldsymbol{k}_{(i+j)\bmod n}$,$i\in\{n-1,\cdots,2,1,0\}$。

基于比特可分性的初始条件和终止条件:由可分轨迹的定义 $\{\boldsymbol{k}\}\triangleq K_0\to K_1\to K_2\to\cdots\to K_r$,$\boldsymbol{a}^0=(a_{n-1}^0,\cdots,a_1^0,a_0^0)\to(a_{n-1}^r,\cdots,a_1^r,a_0^r)=\boldsymbol{a}^r$ 表示一条 r 轮基于比特的可分轨迹,L 表示变量为 a_i^j,$i=0,1,\cdots,n-1$,$j=0,1,\cdots,r$ 的线性不等式组。

令 $D_k^{1,n}$ 表示初始输入可分性,其中 $\boldsymbol{k}=(\boldsymbol{k}_{n-1},\cdots,\boldsymbol{k}_1,\boldsymbol{k}_0)$,$L$ 求解得到的所有可行解——可分轨迹始于可分性 $\boldsymbol{k}$。由于可分性(0,0,…,0,0,0)经过复制操作、异或操作、S 盒、循环移位等基本操作后,输出可分性仍为 **0**,因此可分性初始输入不能为(0,0,…,0,0,0)。

命题 1[20]　假设 X 是具有可分性 $D_{\boldsymbol{K}}^{1,n}$ 的多重集,当 $\boldsymbol{K}$ 包含 n 个单位向量时,多重集 X 不存在积分。

令目标函数为 Obj: Min$\{a_0^r+a_1^r+\cdots+a_{n-1}^r\}$,MILP 问题转化为限制条件为 L、目标函数为 Obj 的求解问题。最后一轮求解得到可分性 $D_{\boldsymbol{K}^r}^{1,n}$,当 $\boldsymbol{K}^r$ 包含 n 个单位向量时终止求解。

3　MILP 方法在 LiCi 算法中的应用

3.1　LiCi 算法 MILP 模型建立

对 LiCi 算法的基本操作建立 MILP 模型,在模型建立过程中,基于比特的复制模型、异或模型和 S 盒模型与已有文献[20]给出的模型构造基本相同,根据 LiCi 算法左支循环右移 7 位后直接得到输出值的结构特性,在最后一轮利用逆向思维构造 LiCi 算法 MILP 模型。

LiCi 算法单轮可分性传播示意图如图 2 所示。

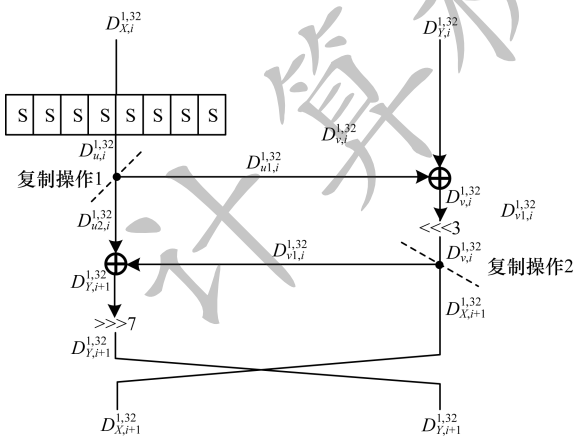

图 2　LiCi 算法单轮可分性传播示意图

Fig. 2　Schematic diagram of single-round separability propagation of LiCi algorithm

第 $i(i\in\{1,2,\cdots,R-1\})$ 轮 LiCi 算法单轮 MILP 模型建立过程如下所示:

1)令可分性 $D_{X,i}^{1,32}$ 经过 S 盒模型,输出可分性为 $D_{u,i}^{1,32}$。

2)令可分性 $D_{u,i}^{1,32}$ 经过“复制操作 1”输出可分性分别为 $D_{u1,i}^{1,32}$、$D_{u2,i}^{1,32}$。

3)令可分性 $D_{u1,i}^{1,32}$ 和 $D_{Y,i}^{1,32}$ 经过异或操作,输出可分性为 $D_{v,i}^{1,32}$。

4)令可分性 $D_{v,i}^{1,32}$ 经过循环左移 3 位后输出可分性为 $D_{vnew,i}^{1,32}$。

5)令可分性 $D_{v,i}^{1,32}=D_{vnew,i}^{1,32}$ 经过“复制操作 2”后,输出可分性分别为 $D_{v1,i}^{1,32}$、$D_{X,i+1}^{1,32}$。

6)令可分性 $D_{u2,i}^{1,32}$ 和 $D_{v1,i}^{1,32}$ 经过异或操作后,输出可分性为 $D_{Y,i+1}^{1,32}$。

7)令可分性 $D_{Y,i+1}^{1,32}$ 经过循环右移 7 位后输出可分性为 $D_{Ynew,i}^{1,32}$,令 $D_{Y,i+1}^{1,32}=D_{Ynew,i}^{1,32}$。

性质 1(基于比特循环移位)　在单轮函数加密结构中,若输入可分性经过循环移位操作后,存在复制操作、异或操作或 S 盒,则直接对输入可分性建立基于比特的循环移位模型,反之,则最后一轮输入可分性经过循环移位操作时利用逆向思维建立基于比特的循环移位模型。以 LiCi 算法为例,最后一轮(第 R 轮)MILP 模型建立过程如下:第 R 轮 MILP 模型建立过程 1)~5)和第 1 轮至第 $R-1$ 轮模型建立过程相同,6)和 7)过程如下:

6)令可分性 $D_{Y,r}^{1,32}$ 经过循环左移 7 位后输出可分性为 $D_{Ynew,r-1}^{1,32}$,令 $D_{Y,r}^{1,32}=D_{Ynew,r-1}^{1,32}$。

7)令可分性 $D_{u2,r-1}^{1,32}$ 和 $D_{v1,r-1}^{1,32}$ 经过异或操作后,输出可分性为 $D_{Y,r}^{1,32}$。

3.2　LiCi 算法基于比特积分区分器搜索结果

目前搜索得到的 LiCi 算法平衡比特数最多,轮数最长的积分区分器为输入 63 比特活跃,输出 43 比特平衡的 12 轮积分区分器。

假设 a 表示活跃,b 表示平衡,c 表示常数,? 表示未知,输入两支的单支为 32 比特,令标号 31 表示最高位,标号 0 表示最低位,输入两支标号表示如下:

$$\begin{cases}X_0:X_{31},X_{30},X_{29},\cdots,X_2,X_1,X_0\\Y_0:Y_{31},Y_{30},Y_{29},\cdots,Y_2,Y_1,Y_0\end{cases}$$

10 轮积分区分器:基于 MILP 搜索得到的平衡比特数最多,轮数最长的积分区分器为输入 62 比特活跃,输出 64 比特平衡的 10 轮积分区分器,输入记为(X_0,Y_0),输出记为(X_{10},Y_{10}),输入输出状态表示如下:

X_0:*aaaa*,*aaaa*,*aaaa*,*aaaa*,*aaaa*,*aaaa*,*aaaa*,*aaaa*

Y_0:*aaaa*,*aaaa*,*aaaa*,*aaaa*,*aaaa*,*aaca*,*aaaa*,*aaca*

X_{10}:*bbbb*,*bbbb*,*bbbb*,*bbbb*,*bbbb*,*bbbb*,*bbbb*,*bbbb*

Y_{10}:*bbbb*,*bbbb*,*bbbb*,*bbbb*,*bbbb*,*bbbb*,*bbbb*,*bbbb*

12 轮积分区分器:目前搜索得到的最长轮数积分区分器为 12 轮积分区分器,共有两个积分区分器,积分区分器 1 是输入 63 比特活跃,输出 43 比特平衡;积分区分器 2 是输入 63 比特活跃,输出 6 比特平衡。输入记为(X_0,Y_0),输出记为(X_{12},Y_{12}),输入输出状态表示如下:

1)12 轮积分区分器 1

X_0:*aaaa*,*aaaa*,*aaaa*,*aaaa*,*aaaa*,*aaaa*,*aaaa*,*aaaa*

Y_0:*aaaa*,*aaaa*,*aaaa*,*aaaa*,*aaaa*,*aaaa*,*aaaa*,*aaac*

X_{12}:*bbbb*,*bbbb*,*bbbb*,*bbbb*,*bbbb*,*bb*??,*bb*??,*bbbb*

Y_{12}:???*b*,??*bb*,*bbbb*,*bbbb*,*bbbb*,*bbbb*,*bbbb*,*b*?? *b*

2)12 轮积分区分器 2

X_0:*aaaa*,*aaaa*,*aaaa*,*aaaa*,*aaaa*,*aaaa*,*aaaa*,*aaac*

Y_0:*aaaa*,*aaaa*,*aaaa*,*aaaa*,*aaaa*,*aaaa*,*aaaa*,*aaaa*

X_{12}:????,??*b*?,*bb*??,*b*???,????,????,????,????

Y_{12}:????,????,????,???*b*,???*b*,????,????,????

4 LiCi 算法的密钥恢复攻击

4.1 13 轮密钥恢复攻击

由于目前基于 MILP 搜索得到的最优 12 轮积分区分器输入活跃比特数为 63 比特,输出平衡比特数为 43 比特,利用 12 轮积分区分器时只能利用 1 组明文,通过猜测 41 比特密钥信息,对第 13 轮 $RK2_{12}$ 的 17 比特密钥信息进行密钥恢复攻击。具体攻击过程和攻击结果如下:

1)对构造 12 轮积分区分器中 2^{63} 个明文进行加密,得到 2^{63} 个密文 $C_0,C_1,\cdots,C_{2^{63}-1}$。

2)猜测第 13 轮 41 比特轮密钥 $RK2_{12,(31,\cdots,13,12,3,2,1,0)}$,$RK1_{12,(28,25,24,23,\cdots,13,12,3,1)}$ 解密第 13 轮,得到第 12 轮 41 比特输出 $X_{12,(31,\cdots,13,12,3,2,1,0)}$ 和 $Y_{12,(23,\cdots,13,12,3,1)}$。如下表示:

$$\begin{aligned}&X_{12,(31,\cdots,13,12,3,2,1\cdots,0)}=\\&S^{-1}((Y_{13}<<<7)\oplus X_{13}\oplus RK2)_{12,(31,\cdots,13,3,\cdots,0)}\\&Y_{12,(28,25,24,23,\cdots,13,12,3,1)}=\\&((Y_{13}<<<7)\oplus X_{13}\oplus RK2_{12})_{(28,25,24,23,\cdots,13,12,3,1)}\oplus\\&(X_{13}>>>3)_{(28,25,24,23,\cdots,13,12,3,1)}\oplus\\&RK1_{12,(28,25,24,23,\cdots,13,12,3,1)}\end{aligned}\tag{2}$$

验证 $X_{12,(31,\cdots,13,12,3,2,1,0)}$ 和 $Y_{12,(28,25,24,23,\cdots,13,12,3,1)}$ 是否为平衡比特,若为平衡比特,则猜测密钥为正确密钥,否则为错误密钥。密钥恢复攻击分为两步,第一步对轮密钥 $RK2_{12,(31,\cdots,13,12,3,2,1,0)}$ 进行密钥恢复攻击,错误轮密钥使 $X_{12,(31,\cdots,13,12,3,2,1,0)}$ 为平衡比特的概率为 2^{-24},经过 1 组明文后剩余错误密钥数目为 $(2^{24}-1)\times 2^{-24}\approx 1$。第二步对轮密钥 $RK2_{12,(31,\cdots,13,12,3,2,1,0)}$ 和 $RK1_{12,(28,25,24,23,\cdots,13,12,3,1)}$ 进行密钥恢复攻击,错误密钥使 $Y_{12,(28,25,24,23,\cdots,13,12,3,1)}$ 为平衡比特的概率为 2^{-17}。由于第一步已经对 $RK2_{12,(28,25,24,23,\cdots,13,12,3,1)}$ 进行筛选,经过 1 组明文后 $RK2_{12,(28,25,24,23,\cdots,13,12,3,1)}$ 剩余错误密钥数目为 1,经过第二步筛选后 $RK2_{12,(28,25,24,23,\cdots,13,12,3,1)}$ 剩余错误密钥数目为 $2^{-17}<<1$,可以恢复 $RK2_{12,(28,25,24,23,\cdots,13,12,3,1)}$ 共 17 比特密钥信息。由于经过第二步筛选,$RK1_{12,(28,25,24,23,\cdots,13,12,3,1)}$ 错误密钥量为 1,因此无法对 $RK1_{12,(23,\cdots,13,12)}$ 进行正确恢复。从而攻击数据复杂度为 2^{63} 个明文,时间复杂度为 $2^{63}\times 2^{41}=2^{104}$ 次查 32 比特S 盒大表,相当于 $2^{104}/16=2^{100}$ 次 16 轮加密。为猜测密钥,攻击需要对猜测密钥进行存储,存储复杂度为 2^{41}。

针对式(2)的计算,可以通过“部分和”[25]技术对其进行改进,具体方式如下:

步骤 1 猜测 $RK2_{12,(31,\cdots,13,12,3,2,1,0)}$ 的一种可能值,并计算 72 比特三元组 $(Z_{12,(31,\cdots,12,3,\cdots,0)},(Y_{13}<<<7)_{(31,\cdots,13,3,\cdots,0)},X_{13,(31,\cdots,13,3,\cdots,0)})$,其中 $Z_{12,(31,\cdots,13,12,3,\cdots,0)}=((Y_{13}<<<7)\oplus X_{13}\oplus RK2_{12})_{(31,\cdots,13,3,\cdots,0)}$,用表 T 记录出现奇数次的三元组 $(Z_{12,(31,\cdots,12,3,\cdots,0)},(Y_{13}<<<7)_{(31,\cdots,13,3,\cdots,0)},X_{13,(31,\cdots,13,3,\cdots,0)})$,求得最终结果 $\oplus Z_{12,(31,\cdots,12,3,\cdots,0)}$。

步骤 2 猜测 $(RK2,RK1)_{12,(28,25,24,23,\cdots,13,12,3,1)}$ 的一种可能值,对表 T 中标记的每一个三元组,计算 $W_{12,(28,25,24,23,\cdots,13,12,3,1)}$,其中 $W_{12,(28,25,24,23,\cdots,13,12,3,1)}=Z_{12,(28,25,24,23,\cdots,13,12,3,1)}\oplus(_{X13}\oplus RK1_{12,(28,25,24,23,\cdots,13,12,3,1)}$,求得最终结果 $\oplus W_{12,(28,25,24,23,\cdots,13,12,3,1)}$。

若步骤 1 中求得的值 $\oplus Z_{12,(31,\cdots,12,3,\cdots,0)}=0$,则此次猜测的 $RK2_{12,(31,\cdots,13,12,3,2,1,0)}$ 有可能是正确密钥,否则一定是错误密钥。一个错误密钥满足 $\oplus Z_{12,(31,\cdots,12,3,\cdots,0)}=0$ 的概率为 2^{-24}。若步骤 2 中求得的值 $\oplus W_{12,(28,25,24,23,\cdots,13,12,3,1)}=0$,则此次猜测的 $RK2_{12,(28,25,24,23,\cdots,13,12,3,1)}$,$RK1_{12,(28,25,24,23,\cdots,13,12,3,1)}$ 有可能是正确密钥,否则一定是错误密钥。一个错误密钥满足 $\oplus W_{12,(28,25,24,23,\cdots,13,12,3,1)}=0$ 的概率为 2^{-17},因此,1 个包含 2^{63} 个明文的明文组可以唯一确定 17 比特轮密钥 $RK2_{12,(28,25,24,23,\cdots,13,12,3,1)}$。

求解 $RK2_{12,(28,25,24,23,\cdots,13,12,3,1)}$ 的时间复杂度步骤如下:

步骤 1 对于 2^{24} 种 $RK2_{12,(31,\cdots,13,12,3,2,1,0)}$ 的可能值,需要处理的密文有 2^{63} 个,因此需要进行 2^{87}次 32 比特S 盒查表操作。

步骤 2 由于一共猜测了 34 比特密钥信息 $RK2_{12,(28,25,24,23,\cdots,13,12,3,1)}$,$RK1_{12,(28,25,24,23,\cdots,13,12,3,1)}$,对于三元组中的每个 $(Z_{12},Y_{13}<<<7,X_{13})_{28,25,24,23,\cdots,13,12,3,1}$ 计算 $W_{12,(28,25,24,23,\cdots,13,12,3,1)}$,且 $(Z_{12},Y_{13}<<<7,X_{13})_{(28,25,24,23,\cdots,13,12,3,1)}$ 至多有 2^{51} 种可能,需要大约进行 2^{85} 次 32 比特 S 盒查表操作。综上,攻击时间复杂度为 $(2^{87}+2^{85})\approx 2^{87}$ 次查 32 比特 S 盒查表,相当于约 2^{83} 次 16 轮加密,相比于 2^{100} 次 16 轮加密结果有了较大改进。

4.2 16 轮密钥恢复攻击

为了得到更长轮数的攻击结果,结合密钥扩展方案的特性,本文选择利用 10 轮积分区分器向后攻击 6 轮,对 16 轮 LiCi 算法进行密钥恢复攻击。攻击过程至少需要猜测 119 比特密钥信息。第 11 轮攻击过程如图 3 所示。

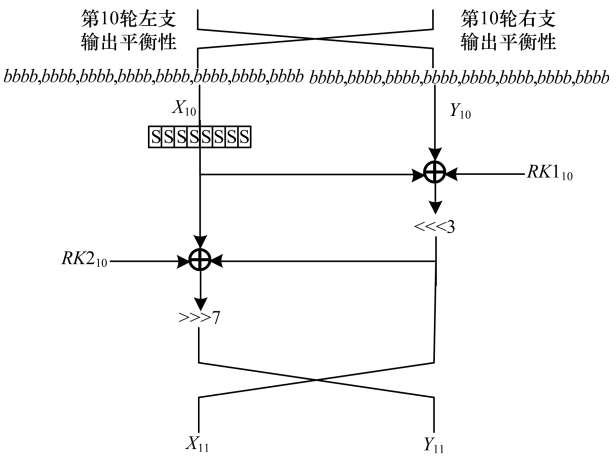

图 3 第 11 轮密钥恢复攻击

Fig. 3 The eleventh round of key recovery attack

具体攻击过程如下:

步骤 1　对构造 10 轮积分区分器中 2^{62} 个明文进行加密,得到 2^{62} 个密文 $C_0, C_1, \cdots, C_{2^{62}-1}$。

步骤 2　猜测第 16 轮 64 比特轮密钥($RK2_{15}$, $RK1_{15}$),解密第 16 轮,得到第 15 轮 64 比特输出 $X_{15,(31,30,\cdots,2,1,0)}$ 和 $Y_{15,(31,30,\cdots,2,1,0)}$。

步骤 3　根据步骤 2 的结果,猜测第 15 轮 64 比特轮密钥($RK2_{14}$,$RK1_{14}$),结合密钥扩展方案和轮密钥的关系可以得知,这一步只需要猜测 13 比特信息。解密第 15 轮,得到第 14 轮 64 比特输出。

步骤 4　根据步骤 3 的结果,猜测第 14 轮 64 比特轮密钥($RK2_{13}$,$RK1_{13}$),结合密钥扩展方案和轮密钥的关系可以得知,这一步只需要猜测 13 比特信息。解密第 14 轮,得到第 13 轮 64 比特输出。

步骤 5　根据步骤 4 的结果,猜测第 13 轮 64 比特轮密钥($RK2_{12}$,$RK1_{12}$),结合密钥扩展方案和轮密钥的关系可以得知,这一步只需要猜测 13 比特信息。解密第 13 轮,得到第 12 轮 64 比特输出。

步骤 6　根据步骤 5 的结果,猜测第 12 轮 64 比特轮密钥($RK2_{11}$,$RK1_{11}$),结合密钥扩展方案和轮密钥的关系可以得知,这一步只需要猜测 13 比特信息。解密第 12 轮,得到第 11 轮 64 比特输出。

步骤 7　根据步骤 6 的结果,猜测第 11 轮 44 比特轮密钥($RK2_{10,(21,20,\cdots,2,1,0)}$,$RK1_{10,(21,20,\cdots,2,1,0)}$),结合密钥扩展方案和轮密钥的关系可以得知,这一步只需要猜测 3 比特信息。解密第 11 轮,得到第 10 轮 42 比特输出($X_{10,(19,\cdots,2,1,0)}$,$Y_{10,(21,20,\cdots,2,1,0)}$),具体表示如下:

$$\begin{aligned} X_{10,(19,\cdots,2,1,0)} &= \mathrm{S}^{-1}((Y_{11} <<<7) \oplus X_{11} \oplus RK2_{10})_{(19,\cdots,2,1,0)} \\ Y_{10,(21,20,\cdots,2,1,0)} &= ((Y_{11} <<<7) \oplus X_{11} \oplus RK2_{10})_{(21,20,\cdots,2,1,0)} \oplus \\ &\quad (X_{11} >>>3)_{(21,20,\cdots,2,1,0)} \oplus \\ &\quad RK1_{10,(21,20,\cdots,2,1,0)} \end{aligned} \tag{3}$$

验证 $X_{10,(19,18,\cdots,2,1,0)}$ 和 $Y_{10,(21,20,\cdots,2,1,0)}$ 是否为平衡比特,若为平衡比特,则猜测密钥为正确密钥,否则为错误密钥。

步骤 8　重新选择一组构造 10 轮积分区分器的明文,重复步骤 1 ~ 步骤 7 直至密钥唯一确定。

结合密钥扩展方案和轮密钥的分析,上述攻击共需要猜测 119 比特密钥信息。对于正确密钥可以保证 $X_{10,(27,26,\cdots,1,0)}$ 和 $Y_{10,(27,26,\cdots,1,0)}$ 为平衡比特,错误密钥使 $X_{10,(27,26,\cdots,1,0)}$ 和 $Y_{10,(27,26,\cdots,1,0)}$ 为平衡比特的概率为 2^{-42},经过 1 组明文后剩余错误密钥数目为 $(2^{119}-1)\times 2^{-42} \approx 2^{77}$,为确定唯一密钥需要 3 组明文,剩余错误密钥数量为 $(2^{119}-1)\times 2^{-42\times 3} \approx 2^{-7}$。从而攻击数据复杂度为 $2^{62}\times 3 = 2^{63.6}$ 个明文,时间复杂度为 $2^{62}\times(2^{119}+2^{77}+2^{35}) \approx 2^{177}$ 次查 32 比特 S 盒大表,相当于 $2^{177}/16 = 2^{173}$ 次 16 轮加密。为猜测密钥,攻击需要对猜测密钥进行存储,存储复杂度为 2^{119}。由于利用 10 轮积分区分器向后扩展 6 轮对 LiCi 算法进行 16 轮积分攻击,第 10 轮输出和第 16 轮密文信息以及第 12 ~ 第 16 轮的轮密钥的每一比特信息均几乎相关,因此未能利用“部分和”技术降低时间复杂度。

根据攻击步骤 1 ~ 步骤 8 结合 LiCi 算法密钥扩展算法可知,利用 10 轮积分区分器向后扩展 2 轮 ~ 6 轮时,攻击时间复杂度均大于 2^{128}。

5　结束语

本文将基于比特的积分性质和 MILP 搜索工具相结合,得到平衡比特数目最多、轮数最长的积分区分器为 12 轮积分区分器,利用 12 轮积分区分器对 LiCi 算法进行 13 轮积分攻击,攻击数据复杂度约为 2^{63},时间复杂度约为 2^{100} 次 16 轮加密,存储复杂度约为 2^{41}。利用“部分和”技术可以将时间复杂度降为 2^{83} 次 16 轮加密。为得到更长轮数的攻击结果,利用构造的 10 轮积分区分器向后攻击 6 轮,对 16 轮算法实施密钥恢复攻击,攻击数据复杂度约为 $2^{63.6}$,时间复杂度约为 2^{173} 次 16 轮加密,存储复杂度约为 2^{119}。本文在积分攻击层面对 LiCi 算法进行分析,结果表明,13 轮 LiCi 算法不能抵抗积分攻击。下一步将基于比特的可分性,在搜索积分区分器时对输入可分性的初始值和积分区分器的轮数与平衡比特数目之间的关系进行研究。

参考文献

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

[2] WU Weiling, ZHANG Lei. LBlock: a lightweight block cipher[C]//Proceedings of International Conference on Applied Cryptography and Network Security. Berlin, Germany: Springer, 2011: 327-344.

[3] KNUDSEN L R, LEANDER G. PRESENT-block cipher[C]//Proceedings of International Workshop on Cryptographic Hardware and Embedded Systems. Berlin, Germany: Springer, 2011: 39-59.

[4] HONG D, SUNG J, HONG S, et al. HIGHT: a new block cipher suitable for low-resource device[C]//Proceedings of International Workshop on Cryptographic Hardware and Embedded Systems. Berlin, Germany: Springer, 2006: 46-59.

[5] BEAULIEU R, SHORS D, SMITH J, et al. The SIMON and SPECK lightweight block ciphers[C]//Proceedings of the 52nd Annual Design Automation Conference. New York, USA: ACM Press, 2015: 1-6.

[6] PATIL J, BANSOD G, KANT K S. LiCi: a new ultra-lightweight block cipher[C]//Proceedings of 2017 International Conference on Emerging Trends & Innovation in ICT. Washington D. C., USA: IEEE Press, 2017: 40-45.

[7] WEI Yongzhuang, SHI Jiali, LI Lingchen. Impossible differential cryptanalysis of LiCi block cipher[J]. Journal of Electronics & Information Technology, 2019, 41(7): 1610-1617. (in Chinese)
韦永壮,史佳利,李灵琛. LiCi 分组密码算法的不可能差分分析[J]. 电子与信息学报, 2019, 41(7): 1610-1617.

[8] KNUDSEN L, WAGNER D. Integral cryptanalysis[C]//

Proceedings of Fast Software Encryption. Berlin, Germany: Springer, 2002: 112-127.
[9] BIHAM E, SHAMIR A. Differential cryptanalysis of DES-like cryptosystems[J]. Journal of Cryptology, 1991, 4(1): 3-72.
[10] MATSUI M. Linear cryptanalysis method for DES cipher[C]// Proceedings of EUROCRYPT'93. Berlin, Germany: Springer, 1993: 386-397.
[11] COURTOIS N T, MEIER W. Algebraic attacks on stream ciphers with linear feedback[C]//Proceedings of Lecture Notes in Computer Science. Berlin, Germany: Springer, 2003: 345-359.
[12] ZHAO Xinjie, WANG Tao, GUO Shize. An improved differential fault analysis on Camellia[J]. Chinese Journal of Computers, 2011, 34(4): 613-627. (in Chinese)
赵新杰, 王韬, 郭世泽. 一种针对 Camellia 的改进差分故障分析[J]. 计算机学报, 2011, 34(4): 613-627.
[13] HUANG Changyang, WANG Tao, WANG Xiaohan, et al. Algebraic fault attack against SIMECK cipher based on optimized fault location[J]. Computer Engineering, 2019, 45(8): 7-13, 21. (in Chinese)
黄长阳, 王韬, 王晓晗, 等. 基于优化故障定位的 SIMECK 密码代数故障攻击[J]. 计算机工程, 2019, 45(8): 7-13, 21.
[14] SHEN Yu, LI Wei, GU Dawu, et al. Integral fault analysis of the ARIA cipher[J]. Journal on Communications, 2019, 40(2): 164-173. (in Chinese)
沈煜, 李玮, 谷大武, 等. ARIA 密码的积分故障分析[J]. 通信学报, 2019, 40(2): 164-173.
[15] DAEMEN J, RIJMEN V. The design of rijndael[M]. New York, USA: ACM Press, 2002: 634-639.
[16] AOKI K, ICHIKAWA T, KANDA M, et al. Camellia: a 128-bit block cipher suitable for multiple platforms — design and analysis[C]//Proceedings of International Conference on Selected Areas in Cryptography. Berlin, Germany: Springer, 2001: 39-56.
[17] JUNOD P, VAUDENAY S. FOX: a new family of block ciphers[C]// Proceedings of International Workshop on Selected Areas in Cryptography. Berlin, Germany: Springer, 2004: 114-129.
[18] BORGHOFF J L, CANTEAUT A N, GÜNEYSU T, et al. PRINCE-A low-latency block cipher for pervasive computing applications[C]//Proceedings of the 18th International Conference on the Theory and Application of Cryptology and Information Security. Berlin, Germany: Springer, 2012: 208-225.
[19] TODO Y. Structural evaluation by generalized integral property[C]//Proceedings of EUROCRYPT'15. Berlin, Germany: Springer, 2015: 287-314.
[20] TODO Y. Integral cryptanalysis on full MISTY1[J]. Journal of Cryptology, 2015, 30: 920-959.
[21] XIANG Zejun, ZHANG Wentao, BAO Zhenzhen, et al. Applying MILP method to searching integral distinguishers based on division property for 6 lightweight block ciphers[C]// Proceedings of ASIACRYPT'16. Berlin, Germany: Springer, 2016: 648-678.
[22] YANG G Q, ZHU B, SUDER V, et al. The simeck family of lightweight block ciphers[C]//Proceedings of International Workshop on Cryptographic Hardware and Embedded Systems. Berlin, Germany: Springer, 2015: 307-329.
[23] CHEN Qin, CHEN Wei, ZHOU Lv. An algorithm of calculating boolean function formers[J]. Computer Engineering and Application, 2002, 38(8): 87-88. (in Chinese)
陈勤, 陈伟, 周律. 布尔函数表达式的求解算法[J]. 计算机工程与应用, 2002, 38(8): 87-88.
[24] TODO Y, MORII M. Bit-based division property and application to simon family[C]//Proceedings of the 23rd International Conference on Fast Software Encryption. Berlin, Germany: Springer, 2016: 357-377.
[25] FERGUSON N, KELSEY J, LUCKS S, et al. Improved cryptanalysis of rijndael[C]//Proceedings of International Workshop on Fast Software Encryption. Berlin, Germany: Springer, 2001: 213-230.

编辑 刘继娟