减轮 LEA 密码算法的积分攻击

李 航[1,2],任炯炯[1,2],陈少真[1,2]
(1. 战略支援部队信息工程大学,河南郑州 450001;2. 数学工程与先进计算国家重点实验室,河南郑州 450001)

摘 要: LEA 密码算法是一类 ARX 型轻量级分组密码,广泛适用于资源严格受限的环境. 本文使用中间相错技术找到 LEA 算法的 86 条 8 轮和 6 条 9 轮零相关区分器,进一步利用零相关区分器和积分区分器的关系,构造出 5 条 8 轮和 1 条 9 轮积分区分器. 在 8 轮积分区分器的基础上,利用密钥扩展算法的性质和部分和技术,首次实现了对 LEA-128 的 10 轮积分攻击,攻击的计算复杂度为 2^{120} 次 10 轮 LEA-128 加密. 进一步,实现了对 LEA-192 的 11 轮积分攻击以及对 LEA-256 的 11 轮积分攻击,计算复杂度分别为 $2^{185.02}$ 次 11 轮 LEA-192 加密和 2^{248} 次 11 轮 LEA-256 加密.

关键词: 轻量级分组密码; LEA 算法; 零相关区分器; 积分攻击
中图分类号: TN918.1 **文献标识码:** A **文章编号:** 0372-2112 (2020)01-0017-11
电子学报 URL: http://www.ejournal.org.cn **DOI:** 10.3969/j.issn.0372-2112.2020.01.003

Integral Attack on Reduced-Round LEA Cipher

LI Hang[1,2], REN Jiong-jiong[1,2], CHEN Shao-zhen[1,2]
(1. *Information Engineering University, Zhengzhou, Henan* 450001, *China*;
2. *State Key Laboratory of Mathematical Engineering and Advanced Computing, Zhengzhou, Henan* 450001, *China*)

Abstract: LEA cipher, a family of ARX lightweight block cipher, is widely used in resource-constrained environments. In this paper, we use the miss-in-the-middle technique to find 86 8-round zero-correlation distinguishers and 6 9-round zero-correlation distinguishers of LEA, and make use of the relationship between the zero-correlation distinguisher and the integral distinguisher to construct 5 8-round distinguishers and 1 9-round integral distinguishers. Based on 8-round integral distinguisher, integral attack on 10-round LEA-128 is implemented firstly by using the property of the key schedule and partial-sum technology, and the attack performs 2^{120} 10-round LEA-128 encryptions. Besides, the integral attack against 11-round LEA-192 with computational complexity of $2^{185.02}$ 11-round LEA-192 encryptions and the integral attack against 11-round LEA-256 with computational complexity of 2^{248} 11-round LEA-256 encryptions are implemented.

Key words: lightweight block cipher; LEA; zero-correlation distinguisher; integral attack

1 引言

LEA 算法[1]由韩国 Electronics and Telecommunications Research Institute 在 2013 年提出的轻量级 ARX 型分组密码算法,其第一设计者与 HIGHT 算法相同. LEA 算法整体采用广义 Feistel 结构,分组长度为 128 比特,对于三种 128 比特,192 比特和 256 比特不同长度密钥,加密轮数分别为 24 轮,28 轮和 32 轮. LEA 算法不仅涉及的运算占用资源少,便于硬件实现,而且提供通用处理器快速软件加密. 出于其潜在应用以及安全性的考虑,对 LEA 算法的分析已备受关注.

对 LEA 密码算法的安全性分析主要集中在传统的差分与线性分析、不可能差分分析、零相关分析、积分分析等. 设计者在文献[1]中提出 LEA 的同时,利用多种密码分析方法对 LEA 的安全性进行了评估. 飞去来器分析方法攻击了 LEA-128 算法 15 轮,差分分析、截断差分分析、不可能差分分析和零相关分析攻击 LEA-128 算法的轮数是 12 轮,积分和零相关线性密码分析攻击 LEA-128 算法 9 轮. 2016 年,文献[2]对零相关线性密码分析的 LEA 族密码的安全级进行了重新评估,给出了 LEA-128/192/256 算法 9/13/14 轮的攻击. 同年,文献[3]对 LEA-128/192/256 给出了 14/14/15 轮的差分分析.

收稿日期:2018-11-05;修回日期:2019-04-22;责任编辑:孙瑶
基金项目:数学工程与先进计算国家重点实验室开放基金课题(No. 2018A03);国家密码发展基金(No. MMJJ20180203);信息保障技术重点实验室开放基金课题(No. KJ-17-002)

零相关分析是继差分密码分析和线性密码分析后,密码学界公认的一种比较有效的密码分析方法. 零相关线性分析的概念由 Bogdanov 和 Rijmen[4] 首次提出. 通过对线性掩码在分组密码算法组件中的传播性质的讨论,Bogdanov 和 Rijmen 展示如何为分组密码算法构建相关度为零的线性逼近以及如何区分分组密码算法与随机置换. 目前,零相关分析经过密码学者的发展[5,6],成为评价新提出密码算法安全性的准则.

积分攻击是另一种比较有效的密码分析方法. 1997 年,Daemen J 等人[7] 提出一种新的方法攻击 SQUARE 算法. 在总结前人工作的基础上, Knudsen L 和 Wagner D[8] 在 FSE2002 上进一步提出了积分攻击的思想. 积分攻击就是选择特定形式的明文进行加密,再对所得密文求和(积分),通过积分值的不随机性将密码算法与随机置换区分开,其对 MISTY1 等主流算法都有很好的攻击效果[9,10]. 积分攻击成功的关键是建立有效的积分区分器,而如何快速寻找到多轮区分器也成为积分攻击的关键点和难点. 在 Eurocrypt 2015 上,Yosuke Todo[11] 提出了寻找积分特征的新技术——可分性(division property). 在 FSE2016 上,Todo 等人[12] 又提出了基于比特的可分性,证明了 SIMON 算法可以抵抗积分攻击. 对于 ARX 型密码,利用基于比特的可分性,文献[13,14]分别基于布尔满足性问题(SAT)和混合整数规划(MILP)构建自动化搜索模型,发现了部分算法更长的积分区分器. 而文献[15]中给出了积分区分器和零相关区分器的转换关系,从而可以利用找到的零相关区分器寻找积分区分器,降低积分区分器的搜索难度.

本文主要研究了针对 LEA 算法的积分攻击. 在寻找积分区分器的研究中,首先考虑线性掩码在 LEA 算法中的传播规律,利用中间相错技术共找到 86 条 8 轮和 6 条 9 轮零相关区分器;而文献[15]中的定理给出了零相关区分器和积分区分器的转化方法,在该定理的基础上结合零相关区分器的形式,进而给出了 5 条 8 轮积分区分器和 1 条 9 轮积分区分器. 在 8 轮积分区分器的基础上,利用部分和技术[16],结合密钥相关信息,首次给出了 LEA-128 算法 10 轮积分攻击,攻击需要 2^{124} 个选择明文,存储量为 2^{37} 个字节,计算复杂度为 2^{120} 次 10 轮 LEA-128 加密. 此外,利用同一个积分区分器,实现了对 LEA-192 的 11 轮积分攻击,攻击需要 2^{124} 个选择明文,存储量为 $2^{142.18}$ 个字节,计算复杂度为 $2^{185.02}$ 次 11 轮 LEA-192 加密;以及实现了对 LEA-256 的 11 轮积分攻击,攻击需要 2^{124} 个选择明文,存储量为 $2^{161.37}$ 个字节,计算复杂度为 2^{248} 次 11 轮 LEA-256 加密. 表 1 和表 2 列出了针对 LEA 算法攻击的主要结果.

表 1 LEA-**128** 攻击结果比较

攻击方法	算法	攻击轮数	复杂度			参考文献
			计算复杂度	存储复杂度	数据复杂度	
差分	LEA-128	12	–	–	–	[1]
差分	LEA-128	14	$2^{124.79}$	2^{22}	$2^{124.79}$	[3]
线性	LEA	11	–	–	–	[1]
飞去来器	LEA-128	15	–	–	–	[1]
截断差分	LEA-128	12	–	–	–	[1]
不可能差分	LEA-128	11	–	–	–	[1]
零相关	LEA-128	9	2^{127}	–	2^{127}	[2]
积分	LEA-128	9	–	–	–	[1]
积分	LEA-128	10	2^{120}	2^{37}	2^{124}	本文

表 2 LEA-**192** 和 LEA-**256** 攻击结果比较

攻击方法	算法	攻击轮数	复杂度			参考文献
			计算复杂度	存储复杂度	数据复杂度	
差分	LEA-192	14	$2^{124.79}$	2^{22}	$2^{124.79}$	[3]
零相关	LEA-192	13	$2^{131.30}$	$2^{60.58}$	2^{128}	[2]
积分	LEA-192	11	$2^{185.02}$	$2^{142.18}$	2^{124}	本文
差分	LEA-256	15	$2^{252.79}$	2^{22}	$2^{124.79}$	[3]
零相关	LEA-256	14	$2^{250.19}$	$2^{142.35}$	2^{128}	[2]
积分	LEA-256	11	2^{248}	$2^{161.37}$	2^{124}	本文

2　LEA 算法

下面定义本文中用到的一些符号.

$P=(P[0],P[1],P[2],P[3])$:128 比特明文,包括 4 个长度为 32 比特的字.

$C=(C[0],C[1],C[2],C[3])$:128 比特密文,包括 4 个长度为 32 比特的字.

$K=(K[0],K[1],\cdots,K[n])$:主密钥,包括 n 个长度为 32 比特的字. 密钥长度为 128 比特,192 比特和 256 比特时,n 的取值分别为 4,6 和 8.

$X_i=(X_i[0],X_i[1],X_i[2],X_i[3])$:128 比特中间状态(第 $i+1$ 轮的输入),包括 4 个长度为 32 比特的字. $X_i[m](j)$ 表示 $X_i[m]$ 的第 j 比特,其中 $0\leqslant m<4$.

$rk_i=(rk_i[0],rk_i[1],rk_i[2],rk_i[3],rk_i[4],rk_i[5])$:第 $i+1$ 轮的子密钥,包括 6 个长度为 32 比特的字. $rk_i[m](j)$ 表示 $rk_i[m]$ 的第 j 比特,其中 $0\leqslant m<6$.

$\boxplus$:模 2^{32} 加. 其逆运算模 2^{32} 减表示为 $\boxminus$.

$\oplus$:按位进行异或.

$\lll$:循环左移.

$\ggg$:循环右移.

2.1　LEA 密码加密算法

分组密码 LEA 算法[1]是广义 Feistel 结构的密码体制,分组长度为 128 比特,支持的密钥长度有 128 比特,192 比特和 256 比特三种,加密轮数分别为 24 轮,28 轮和 32 轮. LEA 的中间状态分成 4 组,每组为 32 比特. 初始化时,把明文 P 赋值给 X_0,调用密钥生成算法由主密钥生成轮子密钥. 加密算法中,轮函数只包括模加运算、异或运算和循环移位,具体过程(如图 1 所示)为:

$$X_{i+1}[0]=((X_i[0]\oplus rk_i[0])\boxplus(X_i[1]\oplus rk_i[1]))\lll 9,$$

$$X_{i+1}[1]=((X_i[1]\oplus rk_i[2])\boxplus(X_i[2]\oplus rk_i[3]))\ggg 5,$$

$$X_{i+1}[2]=((X_i[2]\oplus rk_i[4])\boxplus(X_i[3]\oplus rk_i[5]))\ggg 3,$$

$$X_{i+1}[3]=X_i[0].$$

加密 r 轮后输出的 X_r 即密文 C.

2.2　LEA 算法的密钥生成算法

密钥生成算法在产生轮子密钥的过程中使用以下 8 个常数:

$$\delta[0]=0xc3efe9db,\delta[1]=0x44626b02,$$
$$\delta[2]=0x79e27c8a,\delta[3]=0x78df30ec,$$
$$\delta[4]=0x715ea49e,\delta[5]=0xc785da0a,$$
$$\delta[6]=0xe04ef22a,\delta[7]=0xe5c40957.$$

(1)LEA-128 的密钥生成算法

主密钥 $K=(K[0],K[1],K[2],K[3])$,令 $T[i]=K[i]\ (0\leqslant i<4)$,则轮密钥 $rk_i\ (0\leqslant i<24)$ 生成过程如下:

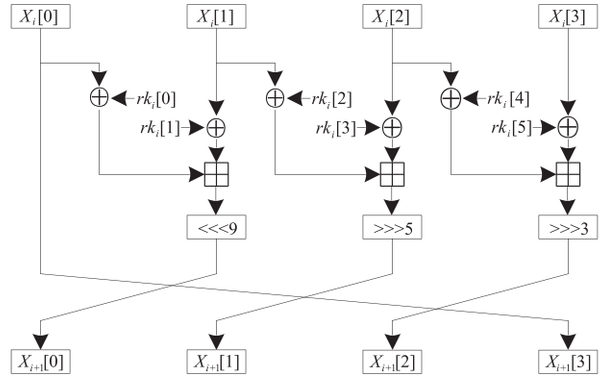

图1　LEA算法的轮函数

$$T[0]=(T[0]\boxplus(\delta[i \bmod 4]\lll i))\lll 1,$$

$$T[1]=(T[1]\boxplus(\delta[i \bmod 4]\lll (i+1)))\lll 3,$$

$$T[2]=(T[2]\boxplus(\delta[i \bmod 4]\lll (i+2)))\lll 6,$$

$$T[3]=(T[3]\boxplus(\delta[i \bmod 4]\lll (i+3)))\lll 11,$$

$$rk_i=(T[0],T[1],T[2],T[1],T[3],T[1]).$$

(2)LEA-192 的密钥生成算法

主密钥 $K=(K[0],K[1],K[2],K[3],K[4],K[5])$,令 $T[i]=K[i]\ (0\leqslant i<6)$,则轮密钥 $rk_i\ (0\leqslant i<28)$ 生成过程如下:

$$T[0]=(T[0]\boxplus(\delta[i \bmod 6]\lll i))\lll 1,$$

$$T[1]=(T[1]\boxplus(\delta[i \bmod 6]\lll (i+1)))\lll 3,$$

$$T[2]=(T[2]\boxplus(\delta[i \bmod 6]\lll (i+2)))\lll 6,$$

$$T[3]=(T[3]\boxplus(\delta[i \bmod 6]\lll (i+3)))\lll 11,$$

$$T[4]=(T[4]\boxplus(\delta[i \bmod 6]\lll (i+4)))\lll 13,$$

$$T[5]=(T[5]\boxplus(\delta[i \bmod 6]\lll (i+5)))\lll 17,$$

$$rk_i=(T[0],T[1],T[2],T[3],T[4],T[5]).$$

(3)LEA-256 的密钥生成算法

主密钥 $K=(K[0],K[1],K[2],K[3],K[4],K[5],K[6],K[7])$,令 $T[i]=K[i]\ (0\leqslant i<8)$,则轮密钥 $rk_i\ (0\leqslant i<32)$ 生成过程如下:

$T[6i \bmod 8]=(T[6i \bmod 8]\boxplus$

$$(\delta[i \bmod 8] \lll i)) \lll 1,$$
$$T[6i+1 \bmod 8] = (T[6i+1 \bmod 8] \boxplus (\delta[i \bmod 8] \lll (i+1))) \lll 3,$$
$$T[6i+2 \bmod 8] = (T[6i+2 \bmod 8] \boxplus (\delta[i \bmod 8] \lll (i+2))) \lll 6,$$
$$T[6i+3 \bmod 8] = (T[6i+3 \bmod 8] \boxplus (\delta[i \bmod 8] \lll (i+3))) \lll 11,$$
$$T[6i+4 \bmod 8] = (T[6i+4 \bmod 8] \boxplus (\delta[i \bmod 8] \lll (i+4))) \lll 13,$$
$$T[6i+5 \bmod 8] = (T[6i+5 \bmod 8] \boxplus (\delta[i \bmod 8] \lll (i+5))) \lll 17,$$
$$rk_i = (T[6i \bmod 8], T[6i+1 \bmod 8], T[6i+2 \bmod 8], T[6i+3 \bmod 8], T[6i+4 \bmod 8], T[6i+5 \bmod 8]).$$

2.3 密钥生成算法的性质

Zhang K 等人在文献[2]中给出了 LEA 密钥生成算法一个性质,该性质揭示了轮子密钥字与主密钥字之间的相关性.

性质 1[2] 每一个轮密钥字 $rk_i[j]$ 必然对应一个特定的主密钥字 $K[a]$,即 $K_a \xrightarrow{f} rk_i[j]$,其中 $f:\{0,1\}^{32} \to \{0,1\}^{32}$ 且 f 是双射.

根据性质 1,如果已知对应每一个主密钥字 $K[a]$ 的轮密钥字 $rk_i[j]$,则由 f 可以求出对应的主密钥字,进而得到主密钥.

事实上,LEA 密钥生成算法是可逆变换,对于 LEA-128 算法,我们给出性质 2.

性质 2 $rk_i(1 \leqslant i < 24)$ 是 LEA-128 算法第 $(i+1)$ 轮的子密钥,则第 i 轮的子密钥 rk_{i-1} 可以由 rk_i 唯一求得:

$$T[0] = (rk_i[0] \ggg 1) \boxminus (\delta[i \bmod 4] \lll i)$$
$$T[1] = (rk_i[1] \ggg 3) \boxminus (\delta[i \bmod 4] \lll (i+1))$$
$$T[2] = (rk_i[2] \ggg 6) \boxminus (\delta[i \bmod 4] \lll (i+2))$$
$$T[3] = (rk_i[4] \ggg 11) \boxminus (\delta[i \bmod 4] \lll (i+3))$$
$$rk_{i-1} = (T[0], T[1], T[2], T[1], T[3], T[1]).$$

证明 根据 LEA-128 的密钥生成算法,$rk_i[0] = (rk_{i-1}[0] \boxminus (\delta[i \bmod 4] \lll i)) \lll 1$. 显然,$rk_i[0] \ggg 1 = rk_{i-1}[0] \boxminus (\delta[i \bmod 4] \lll i)$. 因为 i 已知,所以 $\delta[i \bmod 4]$ 为已知常数. 那么,$rk_{i-1}[0]$ 可以由 $rk_{i-1}[0] = (rk_i[0] \ggg 1) \boxminus (\delta[i \bmod 4] \lll i)$ 求得. 同理,可求 $rk_{i-1}[1]$,$rk_{i-1}[2]$,$rk_{i-1}[4]$. 又根据密钥生成算法,$rk_{i-1}[1] = rk_{i-1}[3] = rk_{i-1}[5]$. 综上,$rk_{i-1}$ 可以由 rk_i 唯一求得. 证毕

类似的,对于 LEA-192 算法和 LEA-256 算法,我们分别给出性质 3 和性质 4.

性质 3 $rk_i(1 \leqslant i < 28)$ 是 LEA-192 算法第 $(i+1)$ 轮的子密钥,则第 i 轮的子密钥 rk_{i-1} 可以由 rk_i 唯一求得:

$$rk_{i-1}[0] = (rk_i[0] \ggg 1) \boxminus (\delta[i \bmod 6] \lll i)$$
$$rk_{i-1}[1] = (rk_i[1] \ggg 3) \boxminus (\delta[i \bmod 6] \lll (i+1))$$
$$rk_{i-1}[2] = (rk_i[2] \ggg 6) \boxminus (\delta[i \bmod 6] \lll (i+2))$$
$$rk_{i-1}[3] = (rk_i[3] \ggg 11) \boxminus (\delta[i \bmod 6] \lll (i+3))$$
$$rk_{i-1}[4] = (rk_i[4] \ggg 13) \boxminus (\delta[i \bmod 6] \lll (i+4))$$
$$rk_{i-1}[5] = (rk_i[5] \ggg 17) \boxminus (\delta[i \bmod 6] \lll (i+5))$$
$$rk_{i-1} = (rk_{i-1}[0], rk_{i-1}[1], rk_{i-1}[2], rk_{i-1}[3], rk_{i-1}[4], rk_{i-1}[5]).$$

性质 4 rk_i 和 $rk_{i+1}(1 \leqslant i < 31)$ 分别是 LEA-256 算法第 $(i+1)$ 轮和第 $(i+2)$ 轮的子密钥,则第 i 轮的子密钥 rk_{i-1} 可以由 rk_i 和 rk_{i+1} 唯一求得:

$$rk_{i-1}[0] = (rk_i[2] \ggg 6) \boxminus (\delta[i \bmod 8] \lll (i+2))$$
$$rk_{i-1}[1] = (rk_i[3] \ggg 11) \boxminus (\delta[i \bmod 8] \lll (i+3))$$
$$rk_{i-1}[2] = (rk_i[4] \ggg 13) \boxminus (\delta[i \bmod 8] \lll (i+4))$$
$$rk_{i-1}[3] = (rk_i[5] \ggg 17) \boxminus (\delta[i \bmod 8] \lll (i+5))$$
$$rk_{i-1}[4] = (rk_{i+1}[0] \ggg 1) \boxminus (\delta[(i+1) \bmod 8] \lll i)$$
$$rk_{i-1}[5] = (rk_{i+1}[1] \ggg 3) \boxminus (\delta[(i+1) \bmod 8] \lll (i+1))$$
$$rk_{i-1} = (rk_{i-1}[0], rk_{i-1}[1], rk_{i-1}[2], rk_{i-1}[3], rk_{i-1}[4], rk_{i-1}[5]).$$

利用上述 3 个性质,我们可以在部分解密过程中由后面轮的子密钥计算出前面轮的子密钥,从而减少密钥猜测量,降低计算复杂度.

3 LEA 算法的积分区分器

3.1 中间相错技术寻找 LEA 算法的零相关区分器

LEA 系列算法轮函数的运算较为简单,考虑线性掩码的传播规律时只需研究模加运算、分支运算以及循环移位. 线性掩码通过上述三种运算的规律如图 2 所示.

综合考虑线性掩码传播规律和 LEA 系列算法本身

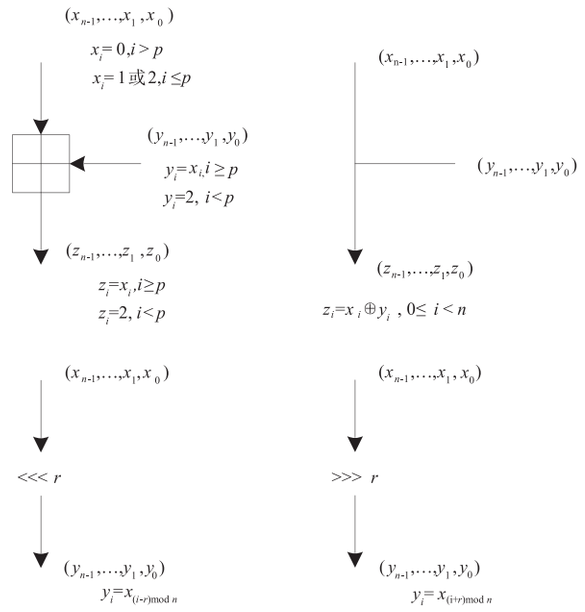

图2　线性掩码的传播规律

轮函数的结构特点,编写出相应的程序,设定输入掩码和输出掩码的形式,将输入掩码向下传播进行加密运算,输出掩码向上传播进行解密运算,直至掩码的值在中间产生一个矛盾. 此时得到的路径就是满足条件的零相关线性路径. 算法 1 给出了寻找 LEA 算法零相关路径的算法.

算法 1　寻找 LEA 算法零相关路径

Input:零相关路径的轮数 r 轮,输入掩码非零比特数 n,输出掩码非零比特数 m

Output:r 轮零相关路径

for $k=1,2,\cdots,r-1$

　for $i=0,1,\cdots,\binom{128}{n}$

　　输入掩码($mask_0[0],mask_0[1],mask_0[2],mask_0[3]$)赋初值

　　正向计算 k 轮,得到对应的输出掩码

　　($mask_k[0],mask_k[1],mask_k[2],mask_k[3]$)

　　for $j=0,1,\cdots,\binom{128}{m}$

　　输出掩码($mask_r[0],mask_r[1],mask_r[2],mask_r[3]$)赋初值

　　反向计算($r-k$)轮,得到对应的输入掩码($mask_k'[0],mask_k'[1],mask_k'[2],mask_k'[3]$)

　　if ($mask_k[0],mask_k[1],mask_k[2],mask_k[3]$)

　　$\neq$($mask_k'[0],mask_k'[1],mask_k'[2],mask_k'[3]$)

　　输出对应的正向和反向掩码传递路径

我们设定输入掩码和输出掩码的值均只有 1 比特非零,利用上述算法,找到了 86 条 8 轮零相关区分器,但是没有找到更长的区分器. 部分结果列在表 3 中,其中 $e_i(0\le i<16)$ 表示第 i 比特的掩码非零,剩余 15 比特的掩码为零.

表 3　部分 8 轮 LEA 零相关路径

$(0,e_0,0,0,0,0,0,0)\to(0,0,0,e_0,0,0,0,0)$	$(0,e_0,0,0,0,0,0,0)\to(0,0,0,e_1,0,0,0,0)$
$(0,e_1,0,0,0,0,0,0)\to(0,0,0,e_0,0,0,0,0)$	$(0,e_1,0,0,0,0,0,0)\to(0,0,0,e_1,0,0,0,0)$
$(0,e_2,0,0,0,0,0,0)\to(0,0,0,e_0,0,0,0,0)$	$(0,e_2,0,0,0,0,0,0)\to(0,0,0,e_1,0,0,0,0)$
$(0,0,0,e_0,0,0,0,0)\to(0,0,0,e_0,0,0,0,0)$	$(0,0,0,e_0,0,0,0,0)\to(0,0,0,e_1,0,0,0,0)$
$(0,0,0,e_1,0,0,0,0)\to(0,0,0,e_0,0,0,0,0)$	$(0,0,0,e_1,0,0,0,0)\to(0,0,0,e_1,0,0,0,0)$
$(0,0,0,e_2,0,0,0,0)\to(0,0,0,e_0,0,0,0,0)$	$(0,0,0,e_2,0,0,0,0)\to(0,0,0,e_1,0,0,0,0)$

为了找到 9 轮或者更长的零相关区分器,我们设定输入掩码只有 1 比特非零,而输出掩码有 2 个非零比特. 利用上述算法,找到了 6 条 9 轮零相关区分器,但是没有找到更长的区分器. 结果列在表 4 中,其中$(0,0,0,e_0,0,0,0,0)\to(0,e_9,0,0,0,0,0,e_0)$与文献[2]中找到的零相关区分器相同.

表 4　9 轮 LEA 零相关路径

$(0,0,0,e_0,0,0,0,0)\to(0,e_9,0,0,0,0,0,e_0)$	$(0,0,0,e_1,0,0,0,0)\to(0,e_9,0,0,0,0,0,e_0)$
$(0,0,0,e_2,0,0,0,0)\to(0,e_9,0,0,0,0,0,e_0)$	$(0,0,e_0,0,0,0,0,0)\to(0,e_9,0,0,0,0,0,e_0)$
$(0,0,e_1,0,0,0,0,0)\to(0,e_9,0,0,0,0,0,e_0)$	$(0,0,e_2,0,0,0,0,0)\to(0,e_9,0,0,0,0,0,e_0)$

3.2　LEA 算法的积分区分器

在文献[15]中,温隆等人介绍了针对 ARX 型算法零相关线性区分器与积分区分器的转换关系. 分组密码算法 H 输入分成 3 部分,而输出分成 2 部分:

$$\mathrm{H}:F_2^r\times F_2\times F_2^s\to F_2^t\times F_2^u$$

$$\mathrm{H}(x,y,z)=\begin{pmatrix}\mathrm{H}_1(x,y,z)\\ \mathrm{H}_2(x,y,z)\end{pmatrix}$$

定理 1[15] 如果输入掩码 a 和输出掩码 b 相互独立并且对于任意的 $a=(a_1\|1,0)$,$a_1\in F_2^r$ 以及 $b=(b_1,0)$,$b_1\neq 0$,$b_1\in F_2^r$,H 上的线性逼近(a,b)的相关度为 0,那么对于任意的 λ,函数 $\mathrm{H}_1(x,y,z)$的异或和为零(积分特性):

$$\oplus_{y\|z}\mathrm{H}_1(\lambda,y,z)=0$$

根据定理 1 的证明过程,我们给出零相关区分器转换为积分区分器的方法:对于一个 r 轮零相关线性区分器 $a\rightarrow b$,将该区分器的矛盾比特记为 A. 选择输入掩码中合适的比特位置组成的集合 a_1,测试是否对于任意取值的 a_1 和 $b_1\neq 0$,都有$(a_1\|1,0)\rightarrow(b_1,0)$是一个 r 轮的零相关线性区分器,并且矛盾比特为 A. 如果成立,则必然存在一个 r 轮的积分区分器. 区分器的输入形式为 a_1 对应的状态比特为活跃的,其余状态比特为常值;输出形式为 b_1 对应的状态比特为平衡的.

当输出掩码只有 1 比特非零时,定理 1 中关于输出掩码的条件可以忽略. 我们给出推论 1.

推论 1 输入掩码 a 和输出掩码 $b=(b_1,0)$相互独立,$b_1=1$,如果对于任意的 $a=(a_1\|1,0)$,$a_1\in F_2^r$,H 上的线性逼近(a,b)的相关度为 0,那么对于任意的 λ,函数 $\mathrm{H}_1(x,y,z)$的异或和为零(积分特性):

$$\oplus_{y\|z}\mathrm{H}_1(\lambda,y,z)=0$$

利用推论 1 改进转换方法,我们将 3.1 节中找到的 8 轮零相关区分器转换为积分区分器. 对于两条积分区分器,如果积分区分器的输入形式相同,但是输出的平衡比特位置不同,我们将其合并为一条区分器. 最后,我们总共得到了 5 条 8 轮积分区分器,将它们列在表 5 中. 其中,A 表示 1 个活动比特,C 表示 1 比特的固定值,U 表示 1 比特任意取值,B 表示 1 个平衡比特,下标表示对应状态的比特数.

表 5 8 轮 LEA 积分区分器

序号	积分区分器
1	$(A_{30}C_2,A_{30}C_2,A_{32},A_{32})\rightarrow(U_{32},B_5U_{19}B_8,U_{32},U_{32})$
2	$(A_{30}C_2,A_{31}C_1,A_{32},A_{32})\rightarrow(U_{32},B_5U_{18}B_9,U_2B_1U_{29},U_{22}B_1U_9)$
3	$(A_{31}C_1,A_{30}C_2,A_{32},A_{32})\rightarrow(U_{32},B_5U_{18}B_9,U_2B_1U_{29},U_{22}B_1U_9)$
4	$(A_{30}C_2,A_{30}C_1A,A_{32},A_{32})\rightarrow(U_{32},B_5U_{18}B_9,U_2B_1U_{29},U_{22}B_1U_9)$
5	$(A_{30}C_1A,A_{30}C_2,A_{32},A_{32})\rightarrow(U_{32},B_5U_{18}B_9,U_2B_1U_{29},U_{22}B_1U_9)$

当输出掩码有多于 1 位比特非零时,根据定理 1 的证明过程,不难得到以下推论.

推论 2 输入掩码 a 和输出掩码 b 相互独立,如果对于任意的 $a=(a_1\|1,0)$,$a_1\in F_2^r$,H 上的线性逼近(a,b)的相关度为 0,那么对于任意的 λ,函数 $\mathrm{H}_1(x,y,z)$的异或和为零(积分特性):

$$\oplus_{y\|z}(\mathrm{H}_1(\lambda,y,z)\odot b)=0$$

利用推论 2 改进转换方法,我们将 3.1 节中找到的 9 轮零相关区分器转换为积分区分器,得到了 1 条 9 轮积分区分器$(A_{30}C_2,A_{30}C_2,A_{32},A_{32})\xrightarrow{9\text{ 轮}}\oplus(C[0](9)\oplus C[3](0))$.

4 减轮 LEA 算法的积分攻击

在第 3 节找到的多条积分区分器中,区分器$(A_{30}C_2,A_{30}C_2,A_{32},A_{32})\xrightarrow{8\text{ 轮}}(U_{32},B_5U_{19}B_8U_{32},U_{32})$需要的选择明文量最少. 因此,我们使用该区分器对减轮的 LEA 算法进行积分攻击.

4.1 10 轮 LEA-128 的积分攻击

4.1.1 攻击过程

在积分区分器的基础上后加 2 轮,对 10 轮 LEA-128 算法进行积分攻击. 为了降低攻击的计算复杂度,我们同时考虑区分器输出的 8 个平衡比特 $X_8[1](0\sim7)$. 在部分解密过程中,我们利用了部分和技术以及密钥生成算法的性质,从而进一步降低了计算复杂度.

解密 2 轮的过程如下(见图 3).

(1)对于 2^{124} 个满足以下形式$(A_{30}C_2,A_{30}C_2,A_{32},A_{32})$的明文,加密获得相应的密文,建立 2^{40} 个 1 比特计数器,统计 $X_{10}[0](9\sim16)\|X_{10}[1](0\sim2,27\sim31)\|X_{10}[2](0\sim4,29\sim31)\|X_{10}[3](0\sim7,9\sim16)$每个值出现的次数,保留计数器为奇数的值.

(2)对于 2^8 个猜测密钥 $rk_9[0](0\sim7)$和保留值,计算 8 比特 $S_9[1](0\sim7)$,建立 2^{32} 个 1 比特计数器,统计 $X_{10}[1](0\sim2,27\sim31)\|X_{10}[2](0\sim4,29\sim31)\|S_9[1](0\sim7)\|X_9[0](9\sim16)$每个值出现的次数,保留计数器为奇数的值,这一步的计算复杂度为 $2^{40}\times2^8\times1/10\approx2^{44.68}$.

(3)对于 2^{16} 个猜测密钥 $rk_9[1](0\sim7)\|rk_9[2](0\sim7)$和保留值,计算 8 比特 $S_9[2](0\sim7)$,建立 2^{24} 个 1 比特计数器,统计 $X_{10}[2](0\sim4,29\sim31)\|S_9[2](0\sim7)\|X_9[0](9\sim16)$每个值出现的次数,保留计数器为奇数的值,这一步的计算复杂度为 $2^{32}\times2^{16+8}\times\frac{1}{10}\approx2^{52.68}$.

(4)猜测 2^8 个密钥 $rk_9[4](0\sim7)$,根据密钥生成算法,$rk_9[5](0\sim7)=rk_9[3](0\sim7)=rk_9[1](0\sim7)$,计算 8 比特 $X_9[3](0\sim7)$,建立 2^{16} 个计数器,统计 $X_9[0](9\sim16)\|X_9[3](0\sim7)$每个值出现的次数,保留计数器为奇数的值,这一步的计算复杂度为 $2^{24}\times2^{16+8+8}\times\frac{1}{10}\approx2^{52.68}$.

(5)利用性质 2,由 $rk_9[0](0\sim7)\|rk_9[1](0\sim7)$,计算复杂度可以忽略. 计算得到$\oplus X_8[1](0\sim7)$的值,若 8 比特均为零,则保留该候选密钥;否则,舍去该

密钥. 这一步的计算复杂度为 $2^{16}\times 2^{16+8+8}\times\frac{1}{10}\approx 2^{44.68}$.

解密过程需要猜测 32 比特密钥. 因为我们考虑 8 个平衡比特,所以总的密钥候选空间从 2^{32} 减少到 $2^{32-8}=2^{24}$. 再加上剩余的 96 比特密钥,仍然有 $2^{24}\times 2^{96}=2^{120}$ 个候选密钥. 针对 2^{120} 个候选密钥检测明密文是否匹配,正确的主密钥可以解出.

攻击程序可以概括如下.

首先,我们构造一个包含 2^{124} 个明文的集合,该集合中的明文满足以下形式($A_{30}C_2$,$A_{30}C_2$,A_{32},A_{32}). 进一步,加密明文获得相应的密文.

然后,猜测子密钥 rk_9 的部分比特,并且部分解密 2^{124} 个密文 2 轮来计算 $\oplus X_8[1](0\sim 7)$,若 8 比特均为零,则为候选密钥;否则,舍去该密钥.

最后,对每一个候选的子密钥,猜测 rk_9 剩余的 96 比特密钥,使用 1 组明密文对检查密钥的正确性,筛选得到正确的密钥.

4.1.2　复杂度分析

数据复杂度为 2^{124} 选择明文.

解密过程中计数器的存储复杂度为 2^{37} 字节,2^{24} 个候选子密钥的存储复杂度为 2^{26} 字节. 所以,总的存储复杂度为 $2^{37}+2^{26}\approx 2^{37}$ 字节.

攻击过程中,第二步的计算复杂度为 $2^{52.68}+2^{52.68}+2^{44.68}+2^{44.68}\approx 2^{53.68}$ 次 10 轮 LEA-128 加密,第三步的计算复杂度为 2^{120} 次 10 轮 LEA-128 加密. 所以,总的计算复杂度为 $2^{120}+2^{53.68}\approx 2^{120}$ 次 10 轮 LEA-128 加密.

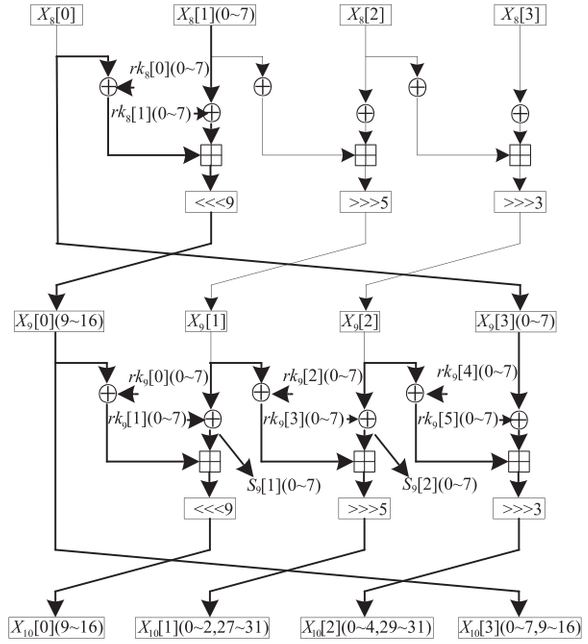

图3　LEA-128部分解密2轮

4.2　11 轮 LEA-192 的积分攻击

4.2.1　攻击过程

在积分区分器的基础上后加 3 轮,对 11 轮 LEA-192 算法进行积分攻击. 为了降低攻击的计算复杂度,我们同时考虑区分器输出的 7 个平衡比特 $X_8[1](0\sim 6)$. 在部分解密过程中,我们利用了部分和技术以及密钥生成算法的性质,从而进一步降低了计算复杂度.

解密 3 轮的过程如下(见图 4).

(1)对于 2^{124} 个满足以下形式($A_{30}C_2$,$A_{30}C_2$,A_{32},A_{32})的明文,加密获得相应的密文,建立 2^{110} 个 1 比特计数器,统计 $X_{11}[0](0\sim 31)\parallel X_{11}[1](0\sim 31)\parallel X_{11}[2](0\sim 3,6\sim 12,29\sim 31)\parallel X_{11}[3](0\sim 31)$ 每个值出现的次数,保留计数器为奇数的值.

(2)猜测 2^{32} 个密钥 $rk_{10}[0](0\sim 31)$,计算 32 比特 $S_{10}[1](0\sim 31)$,建立 2^{85} 个 1 比特计数器,统计 $X_{11}[1](0\sim 31)\parallel X_{11}[2](0\sim 3,6\sim 12,29\sim 31)\parallel S_{10}[1](0\sim 31)\parallel X_{10}[0](9\sim 15)$ 每个值出现的次数,保留计数器为奇数的值,这一步的计算复杂度为 $2^{108}\times 2^{32}\times\frac{1}{11}\approx 2^{136.54}$.

(3)猜测 2^{64} 个密钥 $rk_{10}[1](0\sim 31)\parallel rk_{10}[2](0\sim 31)$,计算 19 比特 $S_{10}[2](0\sim 15,29\sim 31)$,建立 2^{47} 个 1 比特计数器,统计 $X_{11}[2](0\sim 3,6\sim 12,29\sim 31)\parallel S_{10}[2](0\sim 15,29\sim 31)\parallel X_{10}[0](9\sim 15)\parallel X_{10}[1](0,1,27\sim 31)$ 每个值出现的次数,保留计数器为奇数的值,这一步的计算复杂度为 $2^{85}\times 2^{32+64}\times\frac{1}{11}\approx 2^{177.54}$.

(4)猜测 2^{35} 个密钥 $rk_{10}[3](0\sim 15,29\sim 31)\parallel rk_{10}[4](0\sim 15)$,计算 14 比特 $S_{10}[3](0\sim 6,9\sim 15)$,建立 2^{35} 个 1 比特计数器,统计 $X_{10}[0](9\sim 15)\parallel X_{10}[1](0,1,27\sim 31)\parallel X_{10}[2](0\sim 3,29\sim 31)\parallel S_{10}[3](0\sim 6,9\sim 15)$ 每个值出现的次数,保留计数器为奇数的值,这一步的计算复杂度为 $2^{47}\times 2^{32+64+35}\times\frac{1}{11}\approx 2^{174.54}$.

(5)猜测 2^{14} 个密钥 $rk_{10}[5](0\sim 6,9\sim 15)$,计算 14 比特 $X_{10}[3](0\sim 6)\parallel X_9[0](9\sim 15)$,建立 2^{35} 个 1 比特计数器,统计 $X_{10}[0](9\sim 15)\parallel X_{10}[1](0,1,27\sim 31)\parallel X_{10}[2](0\sim 3,29\sim 31)\parallel X_{10}[3](0\sim 6)\parallel X_9[0](9\sim 15)$ 每个值出现的次数,保留计数器为奇数的值,这一步的计算复杂度为 $2^{35}\times 2^{32+64+35+14}\times\frac{1}{11}\approx 2^{176.54}$.

(6)利用性质 3,由 $rk_{10}[0](0\sim 6)\parallel rk_{10}[1](0\sim 6)\parallel rk_{10}[2](0\sim 6)\parallel rk_{10}[3](0\sim 6)\parallel rk_{10}[4](0\sim 6)$ 计算得到 $rk_9[0](0\sim 6)\parallel rk_9[1](0\sim 6)\parallel rk_9[2](0\sim 6)\parallel rk_9[3](0\sim 6)\parallel rk_9[4](0\sim 6)$,计算复杂度可以忽略. 计算得到 $X_9[3](0\sim 6)$,建立 2^{14} 个计数器,统计

$X_9[0](9\sim15)\parallel X_9[3](0\sim6)$每个值出现的次数,保留计数器为奇数的值,这一步的计算复杂度为$2^{35}\times 2^{32+64+35+14}\times\frac{1}{11}\approx 2^{176.54}$.

(7)利用性质3,由$rk_9[1](0\sim6)\parallel rk_9[1](0\sim6)$计算得到$rk_8[0](0\sim6)\parallel rk_8[1](0\sim6)$,计算复杂度可以忽略. 计算得到$\oplus X_8[1](0\sim6)$,若7比特均为零,则保留该候选密钥;否则,舍去该密钥. 这一步的计算复杂度为$2^{14}\times 2^{32+64+35+14}\times\frac{1}{11}\approx 2^{155.54}$.

解密过程需要猜测145比特密钥. 因为我们考虑7个平衡比特,所以总的密钥候选空间从2^{145}减少到$2^{145-7}=2^{138}$. 再加上剩余的47比特密钥,仍然有$2^{138}\times 2^{47}=2^{185}$个候选密钥. 针对$2^{185}$个候选密钥检测明密文是否匹配,正确的主密钥可以解出.

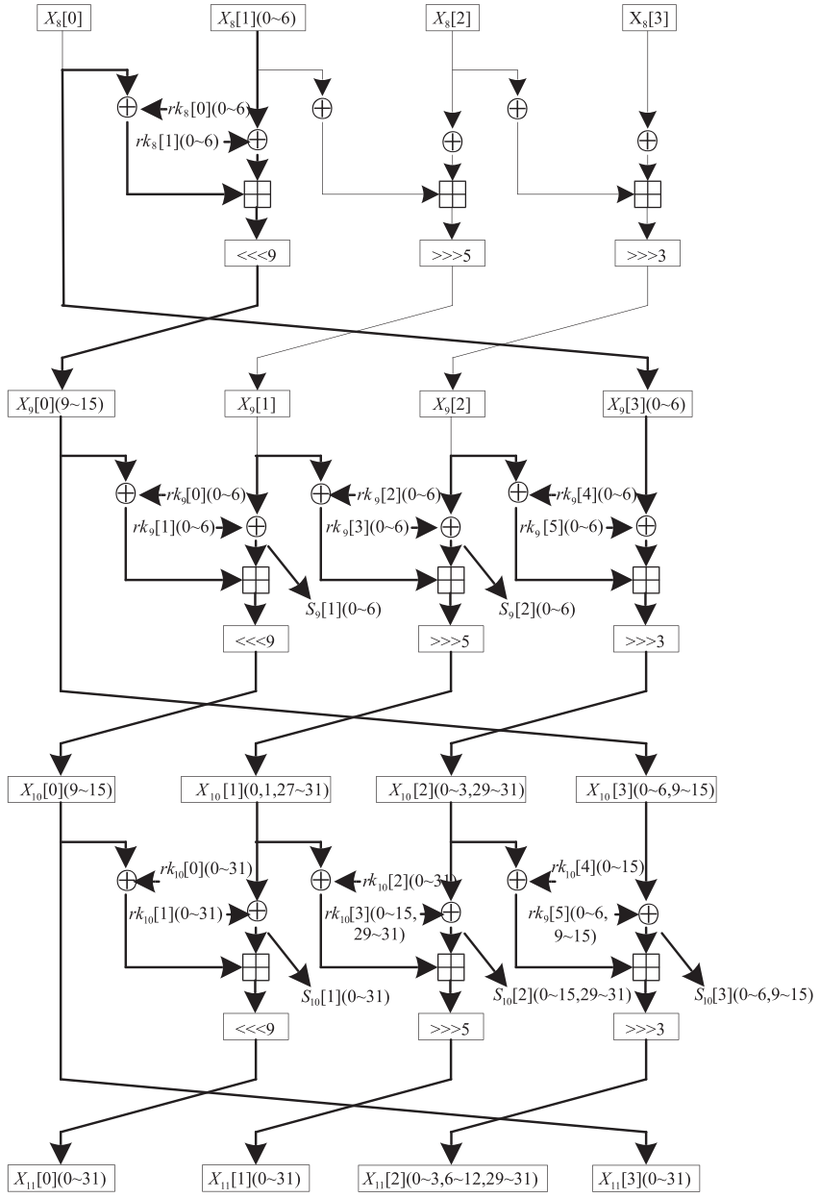

图4 LEA-192部分解密3轮

攻击程序可以概括如下.

首先,我们构造一个包含2^{124}个明文的集合,该集合中的明文满足以下形式$(A_{30}C_2,A_{30}C_2,A_{32},A_{32})$. 进一步,加密明文获得相应的密文.

然后,猜测子密钥rk_{10}的部分比特,并且部分解密2^{124}个密文3轮来计算$\oplus X_8[1](0\sim6)$,若7比特均为

零,则为候选密钥;否则,舍去该密钥.

最后,对每一个候选的子密钥,猜测 rk_{10} 剩余的 47 比特密钥,使用 2 组明密文对检查密钥的正确性,筛选得到正确的密钥.

4.2.2　复杂度分析

数据复杂度为 2^{124} 选择明文.

解密过程中计数器的存储复杂度为 2^{107} 字节,2^{138} 个 145 比特候选子密钥的存储复杂度为 $2^{142.18}$ 字节. 所以,总的存储复杂度为 $2^{105}+2^{142.18}\approx 2^{142.18}$ 字节.

攻击过程中,第二步的计算复杂度为 $2^{136.54}+2^{177.54}+2^{174.54}+2^{176.54}+2^{176.54}+2^{155.54}\approx 2^{178.63}$ 次 11 轮 LEA-192 加密,第三步的计算复杂度为 2^{185} 次 11 轮 LEA-192 加密. 所以,总的计算复杂度为 $2^{185}+2^{178.63}\approx 2^{185.02}$ 次 11 轮 LEA-192 加密.

4.3　11 轮 LEA-256 的积分攻击

4.3.1　攻击过程

在积分区分器的基础上后加 3 轮,对 11 轮 LEA-256 算法进行积分攻击. 为了降低攻击的计算复杂度,我们同时考虑区分器输出的 8 个平衡比特 $X_8[1](0\sim7)$. 在部分解密过程中,我们利用了部分和技术以及密钥生成算法的性质,从而进一步降低了计算复杂度.

攻击的目标是恢复 256 比特主密钥,所以我们需要由猜测的轮子密钥求出主密钥. 为了清楚地表示猜测的轮子密钥和主密钥的相关关系,我们首先利用性质 4 求出和第 9,10,11 轮子密钥相关的主密钥,将它们列在表 6 中.

表 6　LEA-256 部分轮子密钥相关的主密钥

轮数($i+1$)	$rk_i[0]$	$rk_i[1]$	$rk_i[2]$	$rk_i[3]$	$rk_i[4]$	$rk_i[5]$
9	$K[0]$	$K[1]$	$K[2]$	$K[3]$	$K[4]$	$K[5]$
10	$K[6]$	$K[7]$	$K[0]$	$K[1]$	$K[2]$	$K[3]$
11	$K[4]$	$K[5]$	$K[6]$	$K[7]$	$K[0]$	$K[1]$

解密 3 轮的过程如下(见图 5).

(1)对于 2^{124} 个满足以下形式($A_{30}C_2$,$A_{30}C_2$,A_{32},A_{32})的明文,加密获得相应的密文,建立 2^{112} 个 1 比特计数器,统计 $X_{11}[0](0\sim31)\parallel X_{11}[1](0\sim31)\parallel X_{11}[2](0\sim4,6\sim13,29\sim31)\parallel X_{11}[3](0\sim31)$ 每个值出现的次数,保留计数器为奇数的值.

(2)猜测 2^{32} 个密钥 $rk_{10}[0](0\sim31)$,计算 32 比特 $S_{10}[1](0\sim31)$,建立 2^{88} 个 1 比特计数器,统计 $X_{11}[1](0\sim31)\parallel X_{11}[2](0\sim4,6\sim13,29\sim31)\parallel S_{10}[1](0\sim31)\parallel X_{10}[0](9\sim16)$ 每个值出现的次数,保留计数器为奇数的值,这一步的计算复杂度为 $2^{112}\times 2^{32}\times\frac{1}{11}\approx 2^{140.54}$.

(3)猜测 2^{64} 个密钥 $rk_{10}[1](0\sim31)\parallel rk_{10}[2](0\sim31)$,计算 20 比特 $S_{10}[2](0\sim16,29\sim31)$,建立 2^{52} 个 1 比特计数器,统计 $X_{11}[2](0\sim4,6\sim13,29\sim31)\parallel S_{10}[2](0\sim16,29\sim31)\parallel X_{10}[0](9\sim16)\parallel X_{10}[1](0\sim2,27\sim31)$ 每个值出现的次数,保留计数器为奇数的值,这一步的计算复杂度为 $2^{88}\times 2^{32+64}\times\frac{1}{11}\approx 2^{180.54}$.

(4)猜测 2^{37} 个密钥 $rk_{10}[3](0\sim16,29\sim31)\parallel rk_{10}[4](0\sim16)$,计算 16 比特 $S_{10}[3](0\sim7,9\sim16)$,建立 2^{40} 个 1 比特计数器,统计 $X_{10}[0](9\sim16)\parallel X_{10}[0](9\sim16)\parallel X_{10}[1](0\sim2,27\sim31)\parallel X_{10}[2](0\sim4,29\sim31)\parallel S_{10}[3](0\sim7,9\sim16)$ 每个值出现的次数,保留计数器为奇数的值,这一步的计算复杂度为 $2^{52}\times 2^{32+64+37}\times\frac{1}{11}\approx 2^{181.54}$.

(5)猜测 2^{16} 个密钥 $rk_{10}[5](0\sim7,9\sim16)$,利用性质 4,由 $rk_{10}[2](0\sim7)\parallel rk_{10}[3](0\sim7)\parallel rk_{10}[4](0\sim7)\parallel rk_{10}[5](0\sim7)$ 计算得到 $rk_9[1](0\sim7)\parallel rk_9[2](0\sim7)\parallel rk_9[3](0\sim7)$,计算复杂度可以忽略. 计算 16 比特 $X_9[0](9\sim16)\parallel X_9[2](0\sim7)$,建立 2^{24} 个 1 比特计数器,统计 $X_{10}[2](0\sim4,29\sim31)\parallel X_9[0](9\sim16)\parallel X_9[2](0\sim7)$ 每个值出现的次数,保留计数器为奇数的值,这一步的计算复杂度为 $2^{40}\times 2^{32+64+37+16}\times\frac{1}{11}\approx 2^{185.54}$.

(6)猜测 2^{16} 个密钥 $rk_9[4](0\sim7)\parallel rk_9[5](0\sim7)$,计算得到 8 比特 $X_9[3](0\sim7)$,建立 2^{16} 个 1 比特计数器,统计 $X_9[0](9\sim16)\parallel X_9[3](0\sim7)$ 每个值出现的次数,保留计数器为奇数的值,这一步的计算复杂度为 $2^{24}\times 2^{32+64+37+16+16}\times\frac{1}{11}\approx 2^{185.54}$.

(7)利用性质 4,由 $rk_9[2](0\sim7)\parallel rk_9[3](0\sim7)$ 计算得到 $rk_8[0](0\sim7)\parallel rk_8[1](0\sim7)$,计算复杂度可以忽略. 计算得到$\oplus X_8[1](0\sim7)$,若 8 比特均为零,则保留该候选密钥;否则,舍去该密钥. 这一步的计算复杂度为 $2^{16}\times 2^{32+64+37+16+16}\times\frac{1}{11}\approx 2^{177.54}$.

解密过程需要猜测 165 比特密钥. 因为我们考虑 8 个平衡比特,所以总的密钥候选空间从 2^{165} 减少到 $2^{165-8}=2^{157}$. 再加上剩余的 91 比特密钥,仍然有 $2^{157}\times 2^{91}=2^{248}$ 个候选密钥. 针对 2^{248} 个候选密钥检测明密文是否匹配,正确的主密钥可以解出.

攻击程序可以概括如下.

首先,我们构造一个包含 2^{124} 个明文的集合,该集合中的明文满足以下形式($A_{30}C_2$,$A_{30}C_2$,A_{32},A_{32}). 进一步,加密明文获得相应的密文.

然后,猜测子密钥 rk_{10},$rk_9[0]$ 和 $rk_9[1]$ 的部分比特,并且部分解密 2^{124} 个密文 3 轮来计算$\oplus X_8[1](0\sim7)$,若 8 比特均为零,则为候选密钥;否则,舍去该密钥.

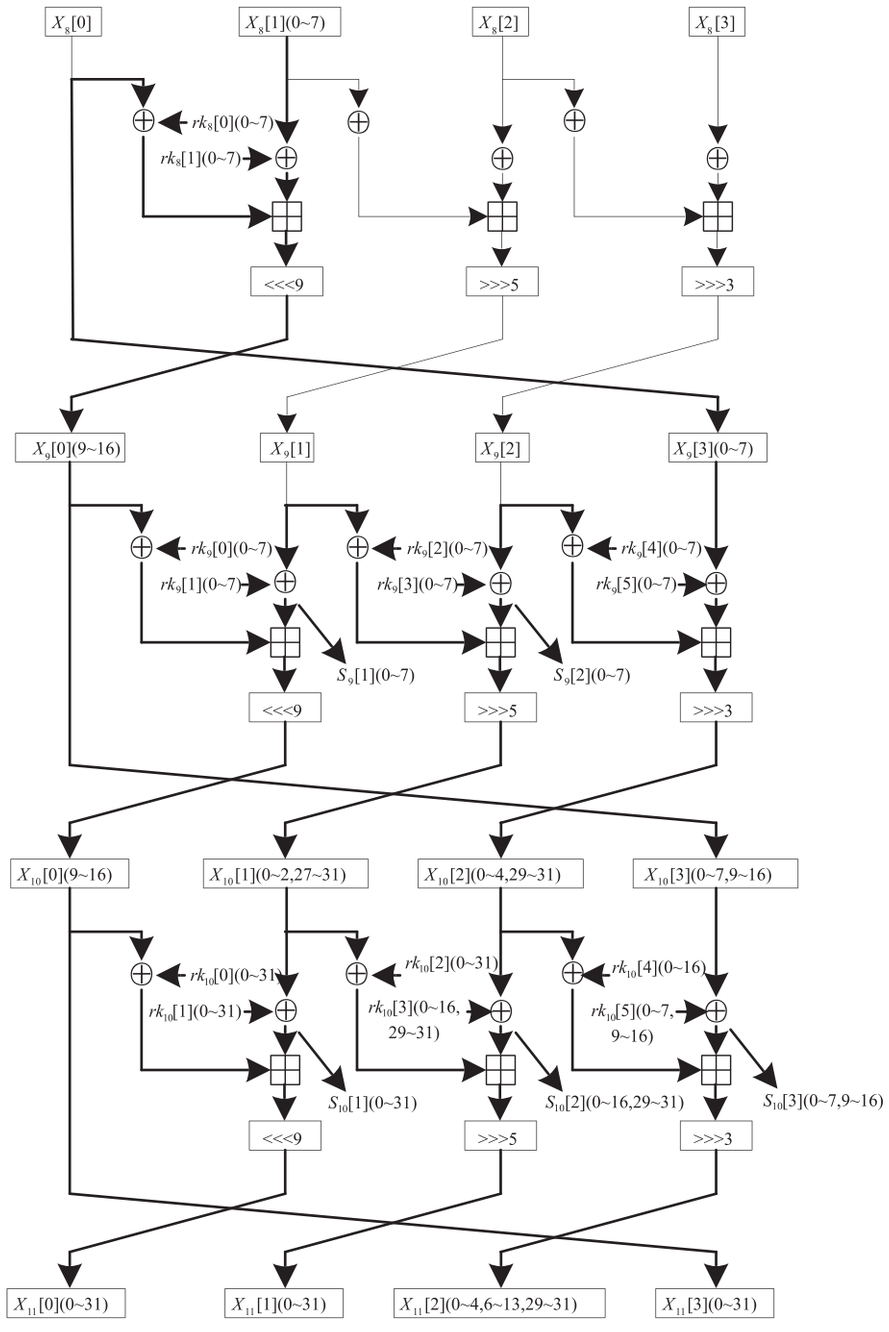

图5 LEA-256部分解密3轮

最后,对每一个候选的子密钥,猜测 rk_{10},$rk_9[0]$和$rk_9[1]$剩余的 91 比特密钥,使用 2 组明密文对检查密钥的正确性,筛选得到正确的密钥.

4.3.2 复杂度分析

数据复杂度为 2^{124}选择明文.

解密过程中计数器的存储复杂度为 2^{109}字节,2^{157}个 165 比特候选子密钥的存储复杂度为 $2^{161.37}$字节. 所以,总的存储复杂度为 $2^{109}+2^{161.37}\approx2^{161.37}$字节.

攻击过程中,第二步的计算复杂度为 $2^{140.54}+2^{180.54}+2^{181.54}+2^{185.54}+2^{185.54}+2^{177.54}\approx2^{186.61}$次 11 轮 LEA-256 加密,第三步的计算复杂度为 2^{248}次 11 轮 LEA-256 加密. 所以,总的计算复杂度为 $2^{248}+2^{186.61}\approx2^{248}$次 11 轮 LEA-256 加密.

5 总结

本文评估了 LEA 抵抗积分攻击的安全性. 使用中

间相错技术找到 LEA 算法的 8 轮和 9 轮零相关区分器,进一步利用零相关区分器和积分区分器的关系,构造出 5 条 8 轮积分区分器和 1 条 9 轮积分区分器. 为了减少攻击的计算复杂度和选择明文量,从 8 轮积分区分器中选择较好的一条,实现了对 LEA-128 的 10 轮积分攻击,改进了文献[1]中积分攻击的结果. 攻击过程利用了密钥扩展算法的性质和部分和技术,减少了密钥猜测量,降低了计算复杂度. 进一步,在上述区分器的基础上,首次实现了对 LEA-192/256 的积分攻击,分别攻击了 11 轮 LEA-192 和 11 轮 LEA-256. 而如何构造较长轮数的区分器,并加大密钥筛选的力度将是下一步值得研究的工作.

参考文献

[1] DEUKJO Hong, et al. LEA: A 128-bit block cipher for fast encryption on common processors[A]. Proceedings of WISA 2013[C]. Cham: Springer, 2013. 3 – 27.

[2] ZHANG Kai, et al. Zero correlation linear cryptanalysis on LEA family ciphers[J]. Journal of Communications, 2016, 11(7): 677 – 685.

[3] SONG Ling, et al. Automatic differential analysis of ARX block ciphers with application to SPECK and LEA[A]. Proceedings of the 21st Australasian Conference on Information Security and Privacy[C]. New York: Springer, 2016. 379 – 394.

[4] BOGDANOV A, RIJMEN V. Linear hulls with correlation zero and linear cryptanalysis of block ciphers[J]. Designs, Codes and Cryptography, 2014, 70(3): 369 – 383.

[5] BOGDANOV A, et al. Integral and multidimensional linear distinguishers with correlation zero[A]. Proceedings of ASIACRYPT 2012[C]. Berlin Heidelberg: Springer, 2012. 244 – 261.

[6] 付立仕,等. 嵌套 SP 网络的 New-Structure 系列结构的零相关线性逼近与不可能差分性质研究[J]. 电子学报, 2017, 45(6): 1367 – 1374.
FU Li-shi, et al. Zero correlation linear approximations and impossible differentials of new-structure series with SP networks[J]. Acta Electronica Sinica, 2017, 45(6): 1367 – 1374. (in Chinese)

[7] DAEMEN J, KNUDSEN L, RIJMEN V. The block cipher square[A]. Proceedings of FSE 1997[C]. Berlin Heidelberg: Springer, 1997. 149 – 165.

[8] KNUDSEN L, WAGNER D. Integral cryptanalysis[A]. Proceedings of FSE 2002[C]. Berlin Heidelberg: Springer, 2002. 112 – 127.

[9] SUN Bing, et al. Higher order integral cryptanalysis of Zodiac[J]. Chinese Journal of Electronics, 2013, 22(3): 589 – 593.

[10] TODO Y. Integral cryptanalysis on full MISTY1[A]. Proceedings of CRYPTO 2015[C]. Berlin Heidelberg: Springer, 2015. 413 – 432.

[11] TODO Y. Structural evaluation by generalized integral property[A]. Proceedings of EUROCRYPT 2015[C]. Berlin Heidelberg: Springer, 2015. 287 – 314.

[12] TODO Y, MORII M. Bit-based division property and application to Simon family[A]. Proceedings of FSE 2016[C]. New York: Springer, 2016. 357 – 377.

[13] SUN L, et al. Automatic search of bit-based division property for ARX ciphers and word-based division property[A]. Proceedings of ASIACRYPT 2017[C]. Cham: Springer, 2017. 128 – 157.

[14] SUN L, et al. MILP-aided bit-based division property for ARX ciphers[J]. Science China Information Sciences, 2018, 61(11): 118102.

[15] Wen L, Wang M. Integral zero-correlation distinguisher for ARX block cipher, with application to SHACAL-2[A]. Proceedings of Information Security and Privacy 2014[C]. Cham: Springer, 2014. 454 – 461.

[16] Ferguson N, et al. Improved cryptanalysis of Rijndael[A]. Proceedings of FSE 2000[C]. Berlin Heidelberg: Springer, 2000. 213 – 230.

作者简介

李　航　男,1995 年 3 月出生,山东菏泽人. 现为战略支援部队信息工程大学硕士研究生,主要研究方向为密码学与信息安全.
E – mail: lih_student@163.com

任炯炯　男,1995 年 1 月出生,甘肃天水人. 现为战略支援部队信息工程大学博士研究生,主要研究方向为密码学与信息安全.

陈少真　女,1967 年 3 月出生,河南郑州人. 战略支援部队信息工程大学教授,主要研究方向为密码学与信息安全.