减轮Simeck算法的积分攻击

任炯炯* 李 航 陈少真
(战略支援部队信息工程大学 郑州 450001)
(数学工程与先进计算国家重点实验室 郑州 450001)

摘 要：该文对轻量级分组密码算法Simeck在积分攻击下的安全性进行了研究。通过向前解密扩展已有的积分区分器，构造了16轮Simeck48和20轮Simeck64算法的高阶积分区分器，并在新区分器的基础上，利用等价子密钥技术和部分和技术，结合中间相遇策略和密钥扩展算法的性质，实现了24轮Simeck48和29轮Simeck64算法的积分攻击。攻击24轮Simeck48的数据复杂度为2^{46}，时间复杂度为2^{95}，存储复杂度为$2^{82.52}$；攻击29轮Simeck64的数据复杂度为2^{63}，时间复杂度为$2^{127.3}$，存储复杂度为$2^{109.02}$。与Simeck算法已有积分攻击的结果相比，该文对Simeck48和Simeck64积分攻击的轮数分别提高了3轮和5轮。

关键词：密码分析；轻量级分组密码；积分攻击；Simeck算法
中图分类号：TN918.1 **文献标识码**：A **文章编号**：1009-5896(2019)09-2156-08
DOI: 10.11999/JEIT180849

Integral Attack on Reduced-round Simeck Algorithm

REN Jiongjiong LI Hang CHEN Shaozhen
(*PLA Information Engineering University*, *Zhengzhou* 450001, *China*)
(*State Key Laboratory of Mathematical Engineering and Advanced Computing*, *Zhengzhou* 450001, *China*)

Abstract: The security of lightweight block cipher Simeck against integral attack is evaluated in this paper. First, a 16-round and a 20-round high-order integral distinguisher of Simeck48 and Simeck64 are constructed by decrypting the existed integral distinguisher forward. Then, combined with the meet-in-the-middle strategy and subkey relationship, the integral attacks on 24-round Simeck48 and 29-round Simeck64 are first proposed utilizing the equivalent-subkey and partial-sum technologies based on the new integral distinguishers. The data, time and memory complexity of attacking 24-round Simeck48 are 2^{46}, 2^{95} and $2^{82.52}$ while the data, time and memory complexity of attacking 29-round Simeck64 are 2^{63}, $2^{127.3}$ and $2^{109.02}$. These new attacks improve greatly the results of the previous integral attack on Simeck. Compared with the known results of the integral attack on Simeck, the number of rounds of the integral attacks on Simeck48 and Simeck64 is increased by 3-round and 5-round, respectively.

Key words: Cryptanalysis; Lightweight block cipher; Integral attack; Simeck algorithm

1 引言

万物联网的需求使得无线传感器、智能卡和RFID标签等受限设备的应用越来越广泛。针对资源受限环境下数据安全和隐私保护问题，兼顾安全和效率的轻量级密码算法的设计和分析成为研究热点。2013年6月，美国国家安全局提出了分别适用于软件和硬件环境的轻量级分组密码算法SIMON和SPECK[1]。这两种算法一经提出，就受到广泛关注。随后，在CHES 2015上，Yang等人[2]提出了一种新的轻量级分组密码Simeck算法。Simeck结合了SIMON和SPECK设计上的优点，在软件及硬件方面都有出色的表现。

针对Simeck算法安全性的分析，最初由设计者给出了Simeck算法差分分析和不可能差分分析的结果[2]。随后，Bagheri[3]对Simeck算法抗线性攻击的能力进行了评估。2016年，文献[4]针对Simeck48

收稿日期：2018-08-31；改回日期：2019-03-14；网络出版：2019-04-01
*通信作者： 任炯炯 jiongjiong_fun@163.com
基金项目：国家密码发展基金(MMJJ20180203)；数学工程与先进计算国家重点实验室开放基金(2018A03)；信息保障技术重点实验室开放基金(KJ-17-002)
Foundation Item: The National Cipher Development Foundation (MMJJ20180203); The State Key Laboratory of Mathematical Engineering and Advanced Computation Open Foundation (2018A03); The Foundation of Science and Technology on Information Assurance Laboratory (KJ-17-002)

和Simeck64，给出了成功概率为1的差分分析的更好结果。随后，Blondeau等人[5]给出了更长轮数带概率的差分分析结果。2018年，Zhang等人[6]给出了Simeck算法零相关线性攻击的结果。对Simeck算法的积分攻击首先由文献[7]提出，Zhang等人[7]构造了Simeck算法的积分区分器，分别给出了21/21/24轮Simeck32/48/64算法的积分攻击。

积分攻击是一种有效的选择明文攻击。1997年，Daemen等人[8]提出一种新的分析方法攻击SQUARE算法。2001年，Lucks[9]提出Saturation攻击，同年，Biryukov等人[10]提出了Multiset攻击。随后，Knudsen等人[11]在FSE 2002上提出了积分攻击的思想。积分攻击的原理是加密特定形式的选择明文，再对所得密文求和，通过积分值的不随机性将密码算法与随机置换区分开。积分攻击成功的关键是建立有效的积分区分器，近年来，提出了很多新的自动化算法搜索积分区分器，大大加快了积分区分器的搜索效率，提高了积分攻击的成功率。

本文对Simeck48和Simeck64在积分攻击下的安全性进行研究。首先，在文献[7]给出的Simeck48算法13轮积分区分器的基础上，标记区分器内部比特的状态，利用文献[12]提出的由内向外的扩展方法，向前解密2轮得到Simeck48算法15轮的高阶积分区分器，再根据Simeck算法Feistel的结构特点，在顶部加1轮得到16轮的积分区分器。然后利用该区分器，给出了24轮Simeck48算法的积分攻击。密钥恢复过程中利用等价子密钥技术和密钥扩展算法的性质减少了密钥猜测量，攻击过程结合中间相遇思想和部分和技术，降低了整个攻击算法的计算复杂度。攻击24轮Simeck48算法的时间复杂度为2^{95}，数据复杂度为2^{46}，存储复杂度为$2^{82.52}$。同样地，对于Simeck64算法，构造了20轮高阶积分区分器，部分解密9轮，猜测得到2^{104}个105 bit等价子密钥，在此基础上穷举23 bit等价子密钥，实现密钥的恢复。攻击29轮Simeck64数据复杂度为2^{63}，时间复杂度为$2^{127.3}$，存储复杂度为$2^{109.02}$。

本文的结构安排如下：第2节简要介绍Simeck算法，给出密钥扩展算法的性质和安全性分析的结果；第3节给出Simeck48算法积分攻击的具体过程和复杂度计算；第4节给出Simeck64算法积分攻击的新结果；第5节总结全文。

2　Simeck算法

文中用到的符号说明如表1所示。

表 1　符号说明

符号	说明
Simeck2n($4n$)	分组为$2n$，密钥为$4n$的Simeck算法
L_i	第i轮左半部分输入$n$bit
$L_i(j)$	L_i第j bit
R_i	第i轮右半部分输入$n$bit
$R_i(j)$	R_i第j bit
K	主密钥$K=(K_3,K_2,K_1,K_0)$
rk_i	第i轮$n$bit的轮子密钥
$\oplus$	异或运算
$\&,\odot$	与运算
$<<<r,>>>r$	循环左移、右移$r$bit

2.1　Simeck算法的轮函数

Simeck算法是文献[2]提出的轻量级分组密码，设计思想结合了SIMON和SPECK算法的优点。为了满足不同的应用需求，Simeck算法按照分组长度和密钥长度的不同分为3个版本Simeck2n($4n$)，n=16，24，32，每个版本的Simeck参数如表2所示。

表 2　Simeck算法的参数

算法	分组长度	密钥长度	字长	轮数
Simeck	32	64	16	32
	48	96	24	36
	64	128	32	44

Simeck算法采用Feistel结构，其轮函数与SIMON算法类似，只有与运算，异或和循环移位操作，唯一的区别是循环移位常数不同。轮函数的迭代过程如图1所示，可以表示如式(1)

$$\left.\begin{aligned} R_{i+1}&=L_i\\ L_{i+1}&=f(L_i)\oplus R_i\oplus \mathrm{rk}_i \end{aligned}\right\} \tag{1}$$

其中，$f(x)=(x\odot(x<<<5))\oplus(x<<<1)$。

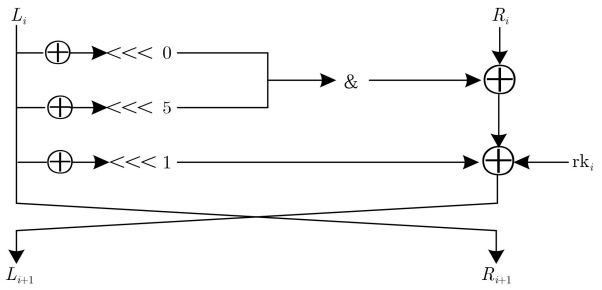


图 1 Simeck算法的轮函数

2.2　Simeck算法的密钥扩展算法

Simeck2n($4n$)的密钥扩展算法借鉴SPECK算法，密钥迭代算法使用线性反馈移位寄存器(LFSR)来产生轮子密钥，初始主密钥K包含4个$n$bit字(K_3,

$K_2, K_1, K_0)$，作为LFSR的初始状态$(t_2, t_1, t_0, \mathrm{rk}_0)$。为了更新寄存器状态以生成轮子密钥，复用轮函数并使用轮常数c_i作为轮密钥。对于$0 \le i < T-1$，状态更新过程如式(2)

$$\left.\begin{aligned} \mathrm{rk}_{i+1} &= t_i \\ c_i &= c \oplus (z_j)_i \\ t_{i+3} &= \mathrm{rk}_i \oplus f(t_i) \oplus c_i \end{aligned}\right\} \tag{2}$$

其中，$c = 2^n - 4$, $(z_j)_i$是序列z_j的第i bit。Simeck32(64)和Simeck48(96)使用同一个m-序列z_0，Simeck64(128)则使用另外一个m-序列z_1。

对于Simeck2n(4n)，通过研究Simeck密钥扩展算法的规律不难发现，开始4轮的密钥rk_0, rk_1, rk_2, rk_3分别等于K_0，K_1，K_2，K_3，且rk_{i+4}只由rk_i，rk_{i+1}和c_i决定，与其它轮子密钥无关。基于上述发现，本文给出密钥扩展算法的性质。

性质1 对于Simeck2n(4n)算法，当$0 \le i \le T-4$时，$\mathrm{rk}_i = \mathrm{rk}_{i+4} \oplus f(\mathrm{rk}_{i+1}) \oplus c_i$。

证明 当$i \ge 3$时，联立Simeck密钥扩展算法的式(2)，可得$\mathrm{rk}_{i+1} = t_i = \mathrm{rk}_{i-3} \oplus f(t_{i-3}) \oplus c_{i-3}$。此外，由式(2)可知，$t_{i-3} = \mathrm{rk}_{i-2}$。因此，$\mathrm{rk}_{i+1}$可以由$\mathrm{rk}_{i-3}$, rk_{i-2}和c_{i-3}唯一表示

$$\mathrm{rk}_{i+1} = \mathrm{rk}_{i-3} \oplus f(\mathrm{rk}_{i-2}) \oplus c_{i-3} \tag{3}$$

即对于任意的$0 \le i \le T-4$，都有$\mathrm{rk}_{i+4} = \mathrm{rk}_i \oplus f(\mathrm{rk}_{i+1}) \oplus c_i$。等式两边同时异或$\mathrm{rk}_{i+4} \oplus \mathrm{rk}_i$，成立

$$\mathrm{rk}_i = \mathrm{rk}_{i+4} \oplus f(\mathrm{rk}_{i+1}) \oplus c_i \tag{4}$$

证毕

更进一步，如果需要求解rk_i的某一特定比特，则有以下性质。

性质2 对于Simeck2n(4n)算法，当$0 \le i \le T-4$，$0 \le j < n$时，$\mathrm{rk}_i(j) = \mathrm{rk}_{i+4}(j) \oplus (\mathrm{rk}_{i+1}(j) \odot \mathrm{rk}_{i+1}((j-5) \bmod n)) \oplus \mathrm{rk}_{i+1}((j-1) \bmod n) \oplus c_i(j)$。

根据Simeck密钥扩展算法的性质，可以利用轮数较高的轮子密钥来求解轮数较低的轮子密钥，进而减少攻击过程中密钥的猜测量。

2.3 Simeck算法的安全性分析

Simeck48和Simeck64算法的安全性分析主要集中在差分和线性攻击，不可能差分和零相关攻击，积分攻击等。表3，表4分别列出了对Simeck48, Simeck64主要攻击结果。

3 24轮Simeck48算法的积分攻击

3.1 Simeck48算法的积分区分器

对于Simeck48算法，文献[7]构造了13轮的积分区分器。具体形式为

$$\begin{aligned} &(\mathrm{ACCC}, \mathrm{ACCC}, \mathrm{AACC}, \mathrm{AAAC}, \mathcal{A}, \mathcal{A}, \mathrm{ACCA}, \\ &\quad \mathrm{ACCA}, \mathrm{AACA}, \mathcal{A}, \mathcal{A}, \mathcal{A}) \\ &\quad \to (U, U, U, U, U, U, *B**, U, U, U, U, U) \end{aligned}$$

表 3 Simeck48的主要攻击结果

算法	攻击方法	攻击轮数	数据复杂度	存储复杂度	时间复杂度	成功概率	参考文献
Simeck48(96)	差分攻击	20	$O(2^{46})$	–	2^{75}	1	文献[2]
Simeck48(96)	差分攻击	26	$O(2^{47})$	–	2^{62}	1	文献[4]
Simeck48(96)	差分攻击	28	$O(2^{46})$	–	$2^{68.3}$	0.468	文献[5]
Simeck48(96)	线性攻击	19	$O(2^{45})$	–	2^{94}	1	文献[3]
Simeck48(96)	不可能差分	24	$O(2^{48})$	$O(2^{74})$	$2^{94.7}$	1	文献[2]
Simeck48(96)	零相关攻击	24	$O(2^{48})$	$O(2^{65.06})$	$2^{91.6}$	1	文献[6]
Simeck48(96)	积分攻击	21	$O(2^{34})$	$O(2^{66})$	2^{95}	1	文献[7]
Simeck48(96)	积分攻击	24	$O(2^{46})$	$O(2^{82.52})$	2^{95}	1	本文

表 4 Simeck64的主要攻击结果

算法	攻击方法	攻击轮数	数据复杂度	存储复杂度	时间复杂度	成功概率	参考文献
Simeck64(128)	差分攻击	26	$O(2^{63})$	–	2^{121}	1	文献[2]
Simeck64(128)	差分攻击	33	$O(2^{63})$	$O(2^{63})$	2^{96}	1	文献[4]
Simeck64(128)	差分攻击	35	$O(2^{63})$	–	$2^{116.3}$	0.555	文献[5]
Simeck64(128)	线性攻击	27	$O(2^{61})$	–	$2^{120.5}$	0.477	文献[3]
Simeck64(128)	不可能差分	25	$O(2^{64})$	$O(2^{79})$	$2^{126.6}$	1	文献[2]
Simeck64(128)	零相关攻击	28	$O(2^{64})$	$O(2^{97.67})$	$2^{123.06}$	1	文献[6]
Simeck64(128)	积分攻击	24	$O(2^{35})$	$O(2^{90.46})$	2^{127}	1	文献[7]
Simeck64(128)	积分攻击	29	$O(2^{63})$	$O(2^{109.02})$	$2^{127.3}$	1	本文

其中，$\mathcal{A}$表示4个活动比特，A表示1个活动比特，C表示1 bit的固定值，U表示4 bit的任意值，$*$表示1 bit的任意值，B表示1个平衡比特。区分器的输入有34个活动比特，区分器输出状态中右半部分第22个比特$C_R(22)$是平衡的。

在该积分区分器的基础上，用“1”标记比特活跃，用“0”标记常数比特，得到向量(1000, 1000, 1100, 1110, 1111, 1111, 1001, 1001, 1101, 1111, 1111, 1111)刻画算法的积分状态，利用文献[12]提出高阶积分的扩展方法，向前解密2轮得到15轮的高阶积分区分器，区分器的输入形式为：$(\mathrm{ACAA}, \mathrm{ACAA}, \mathcal{A}, \mathcal{A}, \mathcal{A}, \mathcal{A}, \mathcal{A}, \mathcal{A}, \mathcal{A}, \mathcal{A}, \mathcal{A}, \mathcal{A})$，有46个活动比特，区分器输出状态中右半部分第22个比特$C_R(22)$仍是平衡的。

进一步，基于Feistel结构，令$(L_1, R_1) = (\mathrm{ACAA}, \mathrm{ACAA}, \mathcal{A}, \mathcal{A}, \mathcal{A}, \mathcal{A}, \mathcal{A}, \mathcal{A}, \mathcal{A}, \mathcal{A}, \mathcal{A}, \mathcal{A})$，可以向前扩展1轮得到16的高阶积分区分器。区分器的输入$(L_0, R_0) = (R_1, f(R_1) \oplus L_1)$，区分器输出状态中右半部分第22个比特$C_R(22)$是平衡比特。利用该区分器向后加8轮，可以攻击24轮的Simeck48算法。攻击过程中利用等价子密钥技术和密钥扩展算法的性质减少密钥猜测量，结合中间相遇思想和部分和技术，降低攻击的复杂度。

3.2 降低复杂度的技术

(1) 等价子密钥技术

等价子密钥技术是由Isobe等人[13]在ASIACRYPT 2013首次提出，结合Simeck算法轮函数的具体运算，把第i轮的子密钥rk_i移动到第$i-1$轮，其中$i = 17, 18, \cdots, 24$，可以得到等效子密钥K'_i。如图2所示，以rk_{23}为例。

图2(a)为Simeck算法本来的结构，为了等效移动rk_{23}而不改变算法结构，rk_{23}需要异或到相应位置，如图2(b)所示。此外，rk_{23}可以进一步转移，与rk_{22}异或，如图2(c)所示。因此，rk_{23}等价于K'_{23}，同样地，可以通过类似的处理进行移动，即$\mathrm{rk}_{22} \oplus (\mathrm{rk}_{23} <<< 1)$等价于$K'_{22}$。以此类推，$K'_i$仅与$\mathrm{rk}_i$有关，并且这种关系是线性的。在攻击过程中，注意到$K'_{16}$在区分器内部，所以恢复密钥时不需要猜测。

(2) 中间相遇策略

根据积分攻击的原理，对猜测的子密钥，如果$\oplus R_{16}(22) = 0$，那么猜测值是候选密钥。因此，进行部分解密的目标是找到使得$\oplus R_{16}(22) = 0$子密钥比特。根据Feistel结构特性式(5)成立

$$\begin{aligned}\oplus R_{16}(22) = \oplus((&L_{16}(22) \odot L_{16}(17)) \\ &\oplus L_{16}(21) \oplus L_{17}(22))\end{aligned} \tag{5}$$

进一步

$$\begin{aligned}\oplus R_{16}(22) = \oplus((&L_{16}(22) \odot L_{16}(17)) \\ &\oplus L_{17}(22) \oplus R_{17}(21))\end{aligned} \tag{6}$$

因此，$\oplus R_{16}(22) = 0$等价于

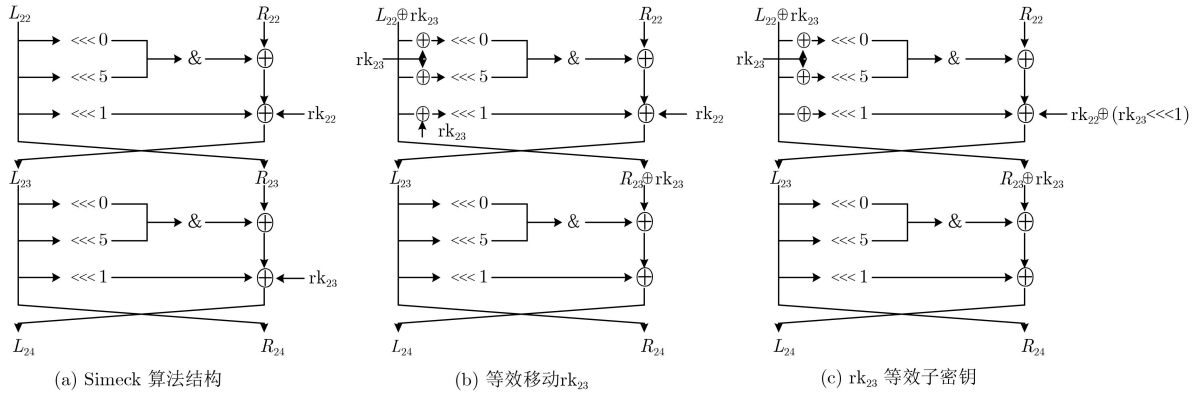
$$\oplus((L_{16}(22) \odot L_{16}(17)) \oplus L_{17}(22) = \oplus R_{17}(21) \tag{7}$$

这个过程如图3所示。

采用中间相遇策略，独立计算$\oplus((L'_{16}(22) \odot L'_{16}(17)) \oplus L'_{17}(22))$，将所有猜测的密钥和结果一起存储在表$\mathrm{Tb}_1$中；独立计算$\oplus R'_{17}(21)$，并将猜测的密钥和结果存储在另一个表$\mathrm{Tb}_2$，其中，$L'_i$和$R'_i$表示使用了等价子密钥技术后的中间状态。最后，通过检查这两个表之间的匹配得到正确的候选密钥，从而降低复杂度。在计算过程中，使用部分和技术来降低时间复杂度。

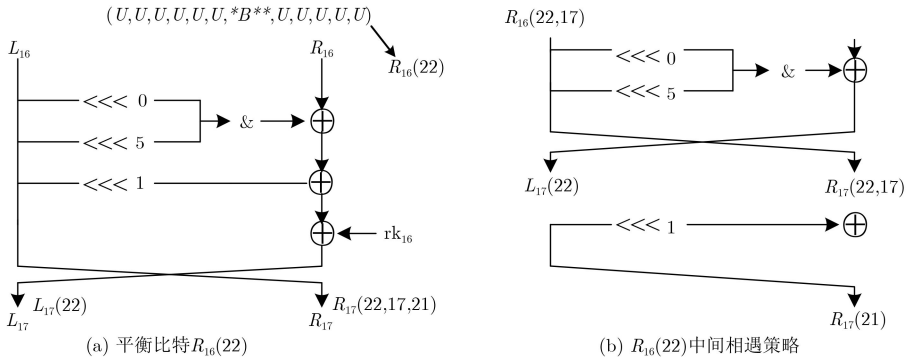
(3) 部分和技术

部分和技术是Ferguson等人[14]在FSE 2000提出的，用以改进对Rijndael算法的攻击结果，随后，部分和技术广泛应用于分组密码的积分攻击[15,16]。下面具体给出利用部分和技术恢复Simeck48密钥的过程。



(a) Simeck 算法结构　(b) 等效移动rk_{23}　(c) rk_{23} 等效子密钥

图 2 等价子密钥技术



(a) 平衡比特$R_{16}(22)$ (b) $R_{16}(22)$中间相遇策略

图 3 中间相遇策略

3.3 Simeck48算法的密钥恢复过程

攻击最终的目标是恢复Simeck48的96-bit密钥，需要独立计算$\oplus((L'_{16}(22)\odot L'_{16}(17))\oplus L'_{17}(22))$和$\oplus R'_{17}(21)$，分别存储在表$\mathrm{Tb}_1$和表$\mathrm{Tb}_2$中。

(1) 计算表Tb_1的过程

计算$\oplus((L'_{16}(22)\odot L'_{16}(17))\oplus L'_{17}(22))$，总共需要猜测79 bit 密钥，具体过程如图4所示。

(a) 对于2^{24}个猜测密钥K'_{23}和2^{46}个密文，设置2^{41}个1 bit计数器Re_1，计算得到$L'_{22}(1,2,5\sim 7,9\sim 22)||R'_{22}(0\sim 2,4\sim 22)$的值，给相应的计数器的值增加1，保留计数器的值为奇数的那些值。这一步的时间复杂度为$2^{46}\times 2^{24}\times 1/24+2^{46}\times 2^{24}\times 22/(24\times 24)\approx 2^{66.42}$。

(b) 对于2^{22}个猜测密钥$K'_{22}(0\sim 2,4\sim 22)$和保留下来的$\mathrm{Re}_1$的位置，设置$2^{33}$个1 bit计数器$\mathrm{Re}_2$，计算得到$L'_{21}(2,6,7,10\sim 12,14\sim 17,19\sim 22)||R'_{21}$

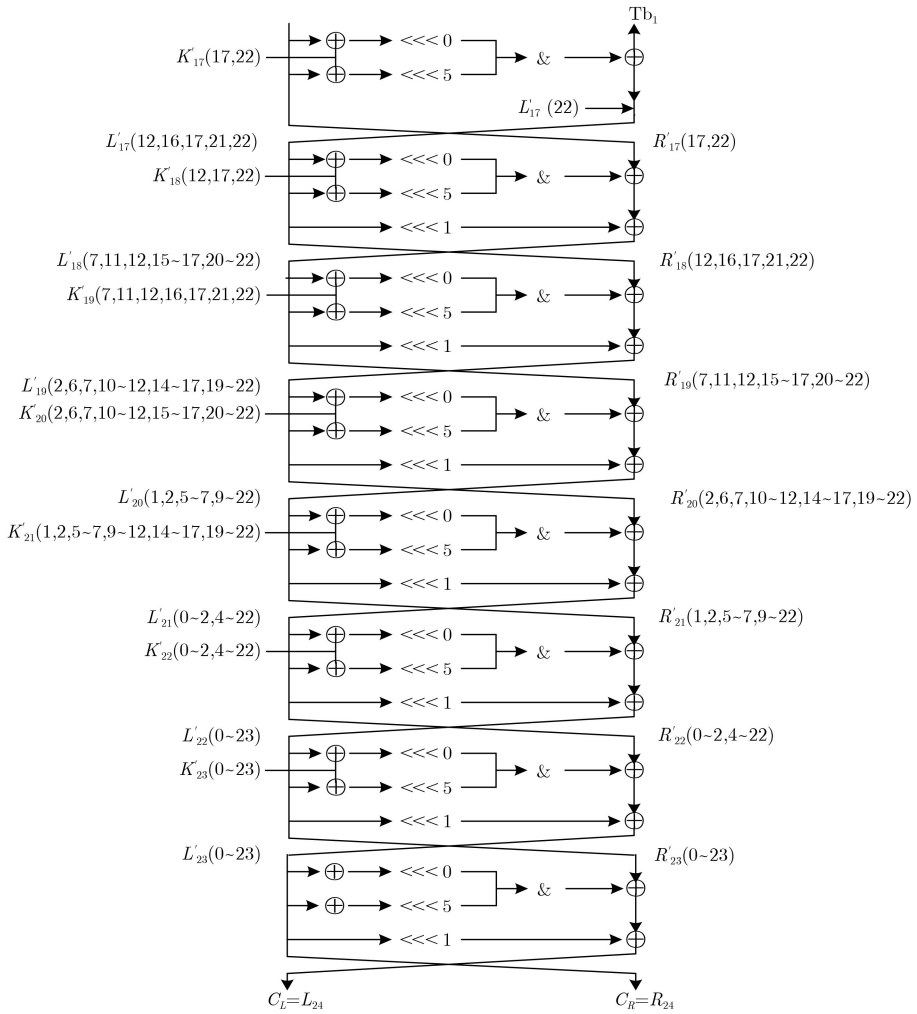


图 4 Tb_1的计算

$(1,2,5\sim7,9\sim22)$的值，保留计数器的值为奇数的那些值。这一步的时间复杂度为$2^{46}\times2^{41}\times19/(24\times24)\approx2^{82.08}$。

(c) 对于2^{10}个猜测密钥$K'_{21}(1,2,6,7,11,12,16,17,21,22)$和保留下来的$\mathrm{Re}_2$的位置，设置$2^{30}$个1 bit计数器$\mathrm{Re}_3$，计算$L'_{20}(5,7,9\sim22)||R'_{20}(2,6,7,11,12,16,17,21,22)||L'_{21}(10,14,15,19,20)$的值，保留计数器的值为奇数的那些值。这一步的时间复杂度为$2^{56}\times2^{33}\times9/(24\times24)\approx2^{83.00}$。

(d) 对于2^7个猜测密钥$K'_{21}(5,9,10,14,15,19,20)$和保留下来的$\mathrm{Re}_3$的位置，设置$2^{23}$个1 bit计数器$\mathrm{Re}_4$，计算得到$L'_{20}(7,11,12,15\sim17,20\sim22)||R'_{20}(2,6,7,10\sim12,14\sim17,19\sim22)$的值，保留计数器的值为奇数的那些值。这一步的时间复杂度为$2^{63}\times2^{30}\times5/(24\times24)\approx2^{86.15}$。

(e) 对于2^7个猜测密钥$K'_{20}(6,10,11,15,16,20,21)$和保留下来的$\mathrm{Re}_4$的位置，设置$2^{18}$个1 bit计数器$\mathrm{Re}_5$，计算得到$L'_{19}(2,6,7,11,12,16,17,21,22)||R'_{19}(11,15,16,20,21)||R'_{20}(7,12,17,22)$的值，保留计数器的值为奇数的那些值。这一步的时间复杂度为$2^{70}\times2^{23}\times5/(24\times24)\approx2^{86.15}$。

(f) 对于2^5个猜测密钥$K'_{20}(2,7,12,17,22)$和保留下来的Re_5的位置，设置2^{14}个1 bit计数器Re_6，计算得到$L'_{19}(12,16,17,21,22)||R'_{19}(7,11,12,15\sim17,20\sim22)$的值，保留计数器的值为奇数的那些值。这一步的时间复杂度为$2^{75}\times2^{18}\times4/(24\times24)\approx2^{85.83}$。

(g) 对于2^7个密钥$K'_{19}(7,11,12,16,17,21,22)$和保留下来的$\mathrm{Re}_6$的位置，由密钥扩展算法的性质1和性质2可知，因为$K'_{19}$=$\mathrm{rk}_{19}\oplus(\mathrm{rk}_{20}<<<1)$，$\mathrm{rk}_{19}$=$\mathrm{rk}_{23}\oplus f(\mathrm{rk}_{20})\oplus c_{19}$，又根据等价子密钥与原始子密钥的线性关系，$\mathrm{rk}_{23}$和$\mathrm{rk}_{20}$可以由已经猜测的密钥比特直接算出，所以$K'_{19}(7,11,12,16,17,21,22)$不需要猜测。而且计算过程与中间状态无关，所以相较于解密过程，计算复杂度忽略不计。设置2^7个1 bit计数器Re_7，计算得到$L'_{18}(17,22)||R'_{18}(12,16,17,21,22)$的值，给相应的计数器的值增加1，保留计数器的值为奇数的那些值。这一步的时间复杂度为$2^{75}\times2^{14}\times5/(24\times24)\approx2^{82.15}$。

(h) 对于2^3个密钥$K'_{18}(12,17,22)$和保留下来的Re_7的位置，与第(g)步类似，根据密钥扩展算法的性质和等价子密钥与原始子密钥的关系，可知计算$K'_{18}(12,17,22)$的复杂度忽略不计。设置2^3个1 bit计数器Re_8，计算得到$L'_{17}(22)||R'_{17}(17,22)$的值，保留计数器的值为奇数的那些值。这一步的时间复杂度为$2^{75}\times2^7\times2/(24\times24)\approx2^{73.83}$。

(i) 对于密钥$K'_{17}(17,22)$以及保留下来的Re_8的位置，猜测4-bit $K'_{20}(14,19)$和$K'_{21}(13,18)$，计算出$K'_{17}(17,22)$这2-bit 的密钥值，计算复杂度约为3轮Simeck48加密运算。计算得到$((L'_{16}(22)\oplus K'_{17}(22))\odot(L'_{16}(17)\oplus K'_{17}(17)))\oplus L'_{17}(22)$的值，和猜测的密钥比特一起存储在表$\mathrm{Tb}_1$。这一步的时间复杂度为$2^{79}\times2^3\times1/(24\times24)+2^{79}\times3\times2/(24\times24)\approx2^{73.64}$。

(2) 计算表Tb_2的过程

计算$\oplus R'_{17}(21)$，总共需要猜测65 bit密钥，具体过程下所示。

(a) 对于2^{22}个猜测密钥$K'_{23}(0,1,3\sim21,23)$和2^{46}个密文，设置2^{33}个1 bit计数器Re_1，计算得到$L'_{22}(1,5,6,9\sim11,13\sim21)||R'_{22}(0,1,4\sim6,8\sim21)$的值，给相应的计数器加1，保留计数器的值为奇数的值。这一步时间复杂度为$2^{46}\times2^{22}\times22/(24\times24)+2^{46}\times2^{22}\times19/(24\times24)\approx2^{63.29}$。

(b) 对于2^{19}个猜测密钥$K'_{22}(0,1,4\sim6,8\sim21)$和保留下来的$\mathrm{Re}_1$的位置，设置$2^{25}$个1 bit计数器$\mathrm{Re}_2$，计算得到$L'_{21}(6,10,11,14\sim16,18\sim21)||R'_{21}(1,5,6,9\sim11,13\sim21)$的值，给相应计数器的值增加1，保留计数器的值为奇数的值。这一步时间复杂度为$2^{41}\times2^{33}\times15/(24\times24)\approx2^{68.74}$。

(c) 对于2^{15}个$K'_{21}(1,5,6,9\sim11,13\sim21)$和保留下来的$\mathrm{Re}_2$位置，设置$2^{16}$个1 bit计数器$\mathrm{Re}_3$，计算$L'_{20}(11,15,16,19\sim21)||R'_{20}(6,10,11,14\sim16,18\sim21)$的值，保留计数器的值为奇数的值。时间复杂度为$2^{56}\times2^{25}\times10/(24\times24)\approx2^{75.15}$。

(d) 对于2^9个猜测密钥$K'_{20}(6,10,11,14\sim16,19\sim21)$和保留下来的$\mathrm{Re}_3$的位置，设置$2^9$个1 bit计数器$\mathrm{Re}_4$，计算得到$L'_{19}(10,16,21)||R'_{19}(11,15,16,19\sim21)$的值，给相应的计数器的值增加1，保留计数器的值为奇数的那些值。这一步的时间复杂度为$2^{65}\times2^{16}\times6/(24\times24)\approx2^{74.42}$。

(e) 对于2^5个密钥$K'_{19}(11,15,16,20,21)$和保留下来的Re_4的位置，因为K'_{19}=$\mathrm{rk}_{19}\oplus(\mathrm{rk}_{20}<<<1)$，由性质1可知$\mathrm{rk}_{19}$=$\mathrm{rk}_{23}\oplus f(\mathrm{rk}_{20})\oplus c_{19}$，又根据等价子密钥与原始子密钥的关系，$\mathrm{rk}_{23}$和$\mathrm{rk}_{20}$可以由已经猜测的密钥比特直接算出，所以$K'_{19}(11,15,16,20,21)$不需要猜测。计算过程与中间状态无关，所以相较于解密过程，计算复杂度可以忽略不计。设置2^4个1 bit计数器Re_5，计算得到$L'_{18}(21)||R'_{18}(10,16,21)$的值，给相应的计数器的值增加1，保留计数器的值为奇数的那些值。这一步的时间复杂度为$2^{65}\times2^9\times3/(24\times24)\approx2^{66.42}$。

(f) 对于密钥$K'_{18}(16,21)$和保留下来的Re_5的位置，密钥$K'_{18}(16,21)$可以由已经猜测的密钥比特唯一解出，计算复杂度约为加密2轮Simeck48运算。计算得到$\oplus R'_{17}(21)$的值，和猜测的密钥一起存储在表Tb_2中。时间复杂度为$2^{65}\times 2^4\times 1/(24\times 24)+2^{65}\times(2\times 2)/(24\times 24)\approx 2^{60.15}$。

因为只考虑1个平衡比特，所以总的密钥候选空间从$2^{65+14+0}=2^{79}$减少到2^{78}。再加上剩余的$K'_{20}(0,1,3,4,5,7,8,13,18,23)$，$K'_{21}(0,3,4,8,23)$和$K'_{22}(1,23)$这17 bit的密钥，仍然有$2^{95}$个候选密钥。针对这$2^{95}$个候选密钥检测明密文是否匹配，进一步根据等价子密钥和轮子密钥的关系和密钥扩展算法，可以由轮子密钥解出正确的主密钥。

24轮Simeck48算法的积分攻击的过程可以概括如下：

步骤 1　构造一个包含2^{46}个明文的集合，该集合中的明文满足以下性质：加密1轮后的输出形式为 $(\mathrm{ACAA},\mathrm{ACAA},\mathcal{A},\mathcal{A},\mathcal{A},\mathcal{A},\mathcal{A},\mathcal{A},\mathcal{A},\mathcal{A},\mathcal{A},\mathcal{A})$。进一步，加密明文获得相应密文；

步骤 2　猜测子密钥$K'_i(17\le i\le 23)$的部分比特，并且部分解密2^{46}个密文8轮来计算$\oplus((L'_{16}(22)\odot L'_{16}(17))\oplus L'_{17}(22))$，计算结果和79 bit猜测密钥一起存储在表$\mathrm{Tb}_1$中；

步骤 3　猜测$K'_i(18\le i\le 23)$的部分比特，并且部分解密2^{46}个密文7轮来计算$\oplus R'_{17}(21)$，计算结果和65 bit猜测密钥一起存储在表Tb_2中；

步骤 4　对每个候选的子密钥，猜测$K'_i(20\le i\le 23)$剩余的17 bit密钥，根据性质1计算得到主密钥，利用两组明密文对检查密钥的正确性，筛选得到正确的主密钥。

3.4 复杂度计算

24轮Simeck48积分攻击的数据复杂度为2^{46}个选择明文。

构造Tb_1和Tb_2的时间复杂度分别为$2^{87.75}$和$2^{75.83}$，所以攻击过程中步骤2和步骤3总的时间复杂度为$2^{87.75}+2^{75.83}\approx 2^{87.75}$次24轮Simeck48加密，而步骤4的时间复杂度为2^{95}次24轮Simeck48加密。所以，总的攻击时间复杂度约为2^{95}次24轮Simeck48加密。

存储的内容包括Tb_1和Tb_2。对于Tb_1，存储复杂度为$2^{82.52}$ Byte。对于Tb_2，存储复杂度为$2^{68.19}$ Byte。所以，总的存储复杂度约为$2^{82.52}$ Byte。

4 29轮Simeck64算法的积分攻击

文献[7]中给出Simeck64算法15轮积分区分器，区分器的输入形式为：$(\mathrm{ACCC},\mathrm{CCCC},\mathrm{CCCC},\mathrm{ACCC},\mathrm{AACC},\mathrm{AAAC},\mathcal{A},\mathcal{A},\mathrm{ACCC},\mathrm{CCCA},\mathrm{CCCA},\mathrm{ACCA},\mathrm{AACA},\mathcal{A},\mathcal{A},\mathcal{A})$，区分器输出状态中右半部分第28个比特$C_R(28)$是平衡的。在该积分区分器基础上，向前做高阶积分扩展，得到19轮的积分区分器$(\mathrm{AAAC},\mathcal{A},\mathcal{A},\mathcal{A},\mathcal{A},\mathcal{A},\mathcal{A},\mathcal{A},\mathcal{A},\mathcal{A},\mathcal{A},\mathcal{A},\mathcal{A},\mathcal{A},\mathcal{A},\mathcal{A},\mathcal{A})$，进一步，基于算法 Feistel结构，可以向前扩展1轮得到20轮的高阶积分区分器。

利用该区分器，攻击29轮的Simeck64算法。攻击过程与3.3节类似。在部分解密9轮Simeck64算法恢复密钥的计算过程中，使用部分和技术和密钥扩展算法的性质来降低时间复杂度。猜测得到2^{104}个105 bit等价子密钥，在此基础上穷举23 bit等价子密钥，从而实现全密钥的恢复。29轮Simeck64积分攻击的数据复杂度为2^{63}，时间复杂度为$2^{127.3}$次，存储复杂度为$2^{109.02}$。

5 结束语

本文给出了轻量级分组密码算法Simeck积分攻击的新结果。在已有积分区分器的基础上，通过向前做高阶积分扩展，分别得到16轮Simeck48和20轮Simeck64算法的积分区分器。利用等价子密钥技术、中间相遇策略，结合部分和技术和密钥扩展算法的性质，实现了24轮Simeck48和29轮Simeck64的积分攻击。与已有积分攻击的结果相比，本文明显提高了Simeck48和Simeck64算法积分攻击的轮数，分别提高了3轮和5轮。下一步如何利用减少复杂度的技术推广积分攻击算法的应用范围，将是值得研究的工作。

参 考 文 献

[1] BEAULIEU R, SHORS D, SMITH J, *et al*. The SIMON and SPECK families of lightweight block ciphers[EB/OL]. http://eprint.iacr.org/2013/404, 2013.

[2] YANG Gangqiang, ZHU Bo, SUDER V, *et al*. The Simeck family of lightweight block ciphers[C]. Proceedings of the 17th International Workshop on Cryptographic Hardware and Embedded Systems 2015, Saint-Malo, France, 2015: 307–329.

[3] BAGHERI N. Linear cryptanalysis of reduced-round Simeck variants[C]. The 16th International Conference on Cryptology in India, Bangalore, India, 2015: 140–152.

[4] KÖLBL S and ROY A. A brief comparison of Simon and Simeck[C]. The 5th International Workshop on Lightweight Cryptography for Security and Privacy, Aksaray, Turkey, 2016: 69–88.

[5] BLONDEAU C, BOGDANOV A and WANG M. On the (In)equivalence of impossible differential and zero-correlation distinguishers for Feistel- and Skipjack-type

ciphers[C]. The 12-th International Conference on Applied Cryptography and Network Security, Lausanne, Switzerland, 2014: 271–288.

[6] ZHANG Kai, GUAN Jie, HU Bin, *et al*. Security evaluation on Simeck against zero-correlation linear cryptanalysis[J]. *IET Information Security*, 2018, 12(1): 87–93. doi: 10.1049/iet-ifs.2016.0503.

[7] ZHANG Kai, GUAN Jie, HU Bin, *et al*. Integral cryptanalysis on Simeck[C]. The Sixth International Conference on Information Science and Technology, Dalian, China, 2016: 216–222.

[8] DAEMEN J, KNUDSEN L R, and RIJMEN V. The block cipher square[C]. The 4th International Workshop on Fast Software Encryption, Haifa, Israel, 1997: 149–165.

[9] LUCKS S. The saturation attack—A bait for Twofish[C]. The 8th International Workshop on Fast Software Encryption, Yokohama, Japan, 2001: 1–15.

[10] BIRYUKOV A and SHAMIR A. Structural cryptanalysis of SASAS[C]. International Conference on the Theory and Application of Cryptographic Techniques: Advances in Cryptology, Innsbruck, Austria, 2001: 395–405.

[11] KNUDSEN L and WAGNER D. Integral cryptanalysis[C]. The 9th International Workshop on Fast Software Encryption, Leuven, Belgium, 2002: 112–127.

[12] ZHANG Wentao, SU Bozhan, WU Wenling, *et al*. Extending higher-order integral: An efficient unified algorithm of constructing integral distinguishers for block ciphers[C]. The 10th International Conference on Applied Cryptography and Network Security, Singapore, 2012: 117–134.

[13] ISOBE T and SHIBUTANI K. Generic key recovery attack on Feistel scheme[C]. The 19th International Conference on the Theory and Application of Cryptology and Information Security, Bengaluru, India, 2013: 464–485.

[14] FERGUSON N, KELSEY J, LUCKS S, *et al*. Improved cryptanalysis of Rijndael[C]. The 7th International Workshop on Fast Software Encryption, New York, USA, 2000: 213–230.

[15] YI Wentan, WU Baofeng, CHEN Shaozhen, *et al*. Improved integral and zero-correlation linear cryptanalysis of CLEFIA block cipher[C]. The 12th International Conference on Information Security and Cryptology, Beijing, China, 2016: 33–46.

[16] FUNABIKI Y, TODO Y, ISOBE T, *et al*. Improved integral attack on HIGHT[C]. The 22nd Australasian Conference on Information Security and Privacy, Auckland, New Zealand, 2017: 363–383.

任炯炯：男，1994年生，博士生，研究方向为对称密码设计与分析.

李 航：男，1995年生，硕士生，研究方向为对称密码设计与分析.

陈少真：女，1967年生，教授，研究方向为密码学与信息安全.