30轮LBC-IoT算法的线性分析

李灵琛[1,2]，陈佐甲[1]，韦永壮[1,2]，叶涛[1,2]
（1.桂林电子科技大学计算机与信息安全学院，广西 桂林 541004；2.广西密码学与信息安全重点实验室，广西 桂林 541004）

摘　要：为了评估LBC-IoT算法抵抗线性分析的能力，基于MILP自动化搜索技术，同时采用直接搜索和迭代线性逼近循环构建两种方法求解轮数最长的线性逼近集合，并在扩展轮数尽可能长的情况下得到每一条线性逼近的初始密钥猜测基。进一步结合最小猜测基技术对初始密钥猜测基进行压缩，以此筛选出最优线性逼近进行密钥恢复攻击。结果表明，LBC-IoT算法共有6条线性偏差为2^{-15}的23轮线性逼近，其中存在唯一一条最小猜测基仅为52 bit的最优线性逼近。基于该区分器向上和向下分别扩展3轮和4轮，首次对LBC-IoT算法发起了最长30轮的密钥恢复攻击。该攻击的数据、时间和存储复杂度分别为2^{30}个已知明文、$2^{77.9}$次30轮加密和2^{52}。相比已有结果，攻击轮数整体提升了4轮，导致LBC-IoT算法的安全冗余轮数不足7%，不建议用于实际的通信数据加密。

关键词：轻量级分组密码；LBC-IoT算法；线性分析；最小猜测基
中图分类号：TP309
文献标志码：A
DOI：10.11959/j.issn.1000–436x.2026036

Linear cryptanalysis of 30-round LBC-IoT algorithm

Li Lingchen[1,2], Chen Zuojia[1], Wei Yongzhuang[1,2], Ye Tao[1,2]
1. School of Computer Science and Information Security, Guilin University of Electronic Technology, Guilin 541004, China
2. Guangxi Key Laboratory of Cryptography and Information Security, Guilin 541004, China

Abstract: In order to evaluate the ability of LBC-IoT algorithm to resist linear cryptanalysis, the set of linear approximations with the longest number of rounds was solved based on the MILP automated search technique using both direct search and iterative linear approximation loop construction methods and obtained the initial key guessing basis for each linear approximation with the longest possible number of extended rounds. The initial key guessing basis was further compressed by combining the minimum guessing basis technique, which filtered out the optimal linear approximation for key recovery attack. The results show that the LBC-IoT algorithm had a total of 6 linear approximations of 23 rounds with linear bias of 2^{-15}, among which there existed the only optimal linear approximation with a minimum guessing basis size of only 52 bit. The first key recovery attack of up to 30 rounds was launched against the LBC-IoT algorithm based on the upward and downward expansion of this distinguisher by 3 and 4 rounds, respectively. The data, time, and storage complexity of this attack was 2^{30} KP, $2^{77.9}$ 30 rounds of encryption, and 2^{52}, respectively. Compared with the existing results, the number of rounds of the attack has been increased by 4 rounds as a whole, which leads to less than 7% of the security redundancy rounds of the LBC-IoT algorithm, which is not recommended to use in practical communication data encryption anymore.

Keywords: lightweight block ciphers, LBC-IoT algorithm, linear cryptanalysis, minimum guessing basis

0　引言

随着物联网（Internet of things, IoT）时代的到来，资源受限设备得到了广泛应用，并且为了保证设备通信安全，轻量级分组密码算法由于加解密速度快等优势被广泛部署于资源受限设备。所以，目前针对轻量级分组密码算法的安全性分析问题也备受

收稿日期：2025–07–19；修回日期：2026–01–29
通信作者：李灵琛，814980156@qq.com
基金项目：国家自然科学基金资助项目（No.62162016, No.62402132）
Foundation Items：The National Natural Science Foundation of China (No.62162016, No.62402132)

关注。线性分析属于已知明文攻击，最早由Matsui[1]提出，也是当前应用最广泛的安全性分析方法之一。

针对线性分析中的线性逼近搜索阶段，为了提高线性逼近搜索的效率和准确性，各类自动化分析技术也扩展应用到密码分析中。早期的自动化分析技术是基于Matsui的分支定界法[2]来搜索最优线性逼近，但是该技术手动编程过程十分烦琐且容易出错。而随着技术理论的不断进步，衍生出了一些基于求解器的自动化搜索技术，目前广泛应用的基于求解器的自动化搜索技术有3种，分别为混合整数线性规划[3]（mixed-integer linear programming, MILP）问题、约束规划[4]（constraint programming, CP）问题和布尔可满足性问题[5]（Boolean satisfiability problem/satisfiability modulo theory, SAT/SMT）。其中，MILP问题是一种源于线性规划的组合优化方法，目的是在约束条件下求出线性目标函数的最优解。Mouha等[3]首次提出将MILP自动化搜索技术应用于轻量级分组密码算法的安全性分析，用于计算面向字分组密码的最小活跃S盒数量下界。Sun等[6]进一步对该方法进行扩展，使其可以应用到线性层为面向比特的轻量级分组密码算法，用于自动化搜索最优线性逼近。Fu等[7]首次给出了模加操作的不等式约束，进一步将MILP自动化搜索技术扩展应用到ARX分组密码算法最优线性逼近的自动化搜索。Zong等[8]提出了两步搜索策略，首先搜索出尽可能长轮数的线性逼近集合，再筛选出密钥恢复攻击猜测密钥数量最少的线性逼近作为最优线性逼近，得到GIFT算法目前最好的线性攻击结果。石康康等[9]结合最小活跃S盒搜索模型与最佳相关性搜索模型，有效减少了搜索空间的可行域，并得到了PICO算法的最长轮数线性逼近。

针对线性分析中的密钥恢复阶段，Matsui[1]首先提出了Matsui算法1和Matsui算法2，用于线性密钥恢复攻击。基于Matsui算法2进行密钥恢复攻击的时间复杂度为$O(D2^{|GK|})$，其中D为数据复杂度，|GK|为密钥恢复攻击过程中需要猜测密钥比特的数量。同时，研究者发现通过一些技术和策略能够进一步降低Matsui算法2的攻击复杂度，最低可降至$O(D+2^{|GK|})$[10]。例如，Collard等[11]提出使用快速沃尔什变换（fast Walsh transform, FWT）降低计算的时间复杂度。Dunkelman等[12]使用密钥桥技术发现了AES-192的一个8轮密钥桥，成功减少了攻击需要猜测的密钥比特数量。Bogdanov等[13]基于错误密钥与密钥等价假设进一步改进Matsui算法2的数据复杂度。Lin等[14]提出了密钥桥的自动化搜索算法，并应用于LBlock和TWINE算法的不可能差分分析。Cen等[15]基于MILP自动化搜索算法实现了猜测确定攻击的自动化。Hadipour等[16]将密钥桥的自动化搜索算法与猜测确定攻击自动化相结合，提出了工具Autoguess。

为了满足资源受限设备在不同环境中的应用，Ramadan等[17]提出了一种Feistel结构的轻量级分组密码算法LBC-IoT，该算法的硬件实现面积仅为548 GE（等效门），优于相同分组长度的SIMON与SPECK算法[18]。Yasushi等[19]基于MILP自动化搜索算法搜索得到了一个23轮线性逼近表达式，并在此基础上对LBC-IoT算法发起了26轮的密钥恢复攻击。Chan等[20]基于一条18轮差分区分器对LBC-IoT算法发起了19轮的密钥恢复攻击。Grochal等[21]基于18轮的差分中间相遇区分器对LBC-IoT算法发起了26轮的密钥恢复攻击。

本文的主要贡献如下。

1) 基于MILP自动化搜索技术，同时利用直接搜索与迭代线性逼近循环构建两种方法搜索到6条线性偏差为2^{-15}的23轮线性逼近。在区分器前后扩展轮数尽可能长的情况下，首先得出每一条线性逼近的初始密钥猜测基大小，并进一步结合最小猜测基技术，筛选出唯一一条最小猜测基仅为52 bit的最优线性逼近。

2) 基于筛选得到的最优线性逼近向前和向后分别扩展3轮和4轮，对LBC-IoT算法首次成功发起了30轮的密钥恢复攻击。相较已有结果，新攻击的轮数整体提升了4轮，导致LBC-IoT算法的安全冗余轮数不足7%，存在极大的安全风险。

LBC-IoT算法的攻击结果对比如表1所示。

1 基础知识

本节首先给出一些必要的符号说明，然后对LBC-IoT算法的加密流程和密钥编排算法进行详细介绍。

1.1 符号说明

本文中使用的主要符号及其相关含义如表2所示。

表1　　LBC-IoT算法的攻击结果对比

攻击方法	攻击轮数	区分器轮数	数据复杂度	时间复杂度	存储复杂度	参考文献
差分分析	19	18	2^{31} CP	2^{32}	2^{11}	文献[20]
差分中间相遇	26	18	2^{32} CP	2^{67}	2^{65}	文献[21]
线性分析	26	23	2^{30} KP	$2^{56.9}$	—	文献[19]
线性分析	30	23	2^{30} KP	$2^{77.9}$	2^{52}	本文算法

注:CP表示选择明文,KP表示已知明文。

表2　　符号定义

符号	定义
M/C	32 bit的明/密文
X_i,Y_i	第i轮输入状态的左、右分支
ΓX_i,ΓY_i	第i轮输入掩码状态的左、右分支
E	分组密码算法的加密部分
sk_i	第i轮的轮密钥
$k_{i,j}^{t}$	第t轮知识传播第i次密钥寄存器更新的第j比特
$P1$,$P2$	长度为16 bit的置换操作
$x\|\|y$	字符串x与y的连接
$X<<<l$	字X循环左移l比特

1.2　LBC-IoT算法

本节主要介绍LBC-IoT算法的加密流程与密钥编排算法。

LBC-IoT算法整体采用经典的两分支Feistel结构设计，是分组长度为32 bit的轻量级分组密码算法，密钥长度为80 bit，迭代轮数为32轮。LBC-IoT算法轮函数结构如图1所示。

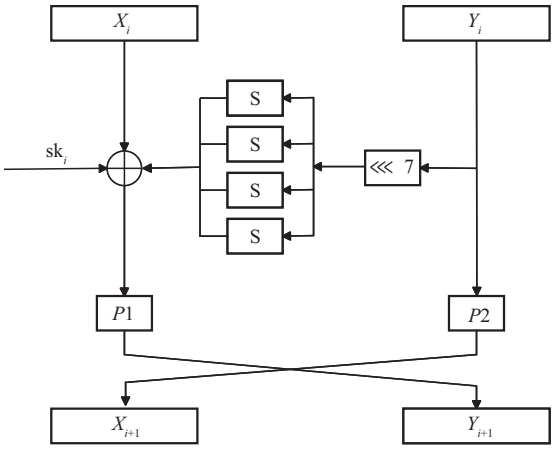

图1　LBC-IoT算法轮函数结构

LBC-IoT算法将32 bit明文M分为左右两分支，$M = X_0||Y_0$，左右分支皆为16 bit。假设(X_i, Y_i)和(X_{i+1}, Y_{i+1})分别为LBC-IoT算法第i轮的输入和输出，且i=0,1,2,…,31，F_{sk_i}为轮函数，sk_i为第i轮的轮密钥，为保证加解密的一致性，在最后一轮加密时分支不进行交换。其第i轮的加密可表示为

$$\begin{cases} X_{i+1} = P2(Y_i) \\ Y_{i+1} = P1\left(X_i \oplus \mathrm{sk}_i \oplus S\left(Y_i <<< 7\right)\right) \end{cases} \tag{1}$$

LBC-IoT算法的轮函数F_{sk_i}由循环移位、线性变换$P1$与$P2$、非线性变换S盒和轮密钥异或等操作组成。其中，线性变换$P1$与$P2$都为16 bit的拉线变换，比特置换规则如表3所示。非线性变换层由4个相同的4 bit可逆S盒组成，S盒具体细节如表4所示。

表3　　LBC-IoT算法的比特置换规则

x	$P1(x)$	$P2(x)$	x	$P1(x)$	$P2(x)$
0	12	4	8	4	3
1	9	7	9	15	0
2	6	15	10	14	13
3	11	11	11	3	5
4	8	2	12	0	8
5	13	10	13	5	14
6	2	1	14	10	6
7	1	12	15	7	9

表4　　LBC-IoT算法的S盒

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

密钥编排算法。80 bit的初始主密钥$K=k_{0,79}||k_{0,78}||\cdots||k_{0,1}||k_{0,0}$分别存储在两个长度为40 bit的寄存器KeyMSB和KeyLSB中，其中最左端40 bit寄存于KeyMSB中，最右端40 bit寄存于KeyLSB中。第i轮的16 bit轮密钥为sk_i，前5轮的轮密钥是由主密钥K直接拆分为5个16 bit的轮密钥，即$K=\mathrm{sk}_0||\mathrm{sk}_1||\mathrm{sk}_2||\mathrm{sk}_3||\mathrm{sk}_4$，之后取每次更新之后的KeyMSB最左端16 bit作为轮密钥sk_i，寄存器KeyMSB与KeyLSB的更新流程如图2所示。

2　基于MILP的线性逼近搜索基本模型

分组密码线性自动化分析的MILP模型主要包括两个部分：一是目标函数的设置，一般以最小线性活跃S盒数量或最高线性相关性/偏差为目标函数；二是不等式约束条件的构建，主要针对线性掩码经过线性和非线性操作的传播约束刻画，以及必要的初始条件刻画。构建好MILP模型后，利用求解器（Gurobi[22]）对模型进行求解。

LBC-IoT算法轮函数由S盒、异或、分支和P置换操作构成。下面将对这几种操作的刻画模型进行详细介绍。

1) S盒操作[6]。假设4 bit的S盒操作$\boldsymbol{x}(x_3,x_2,x_1,x_0)\xrightarrow{\mathrm{S}}\boldsymbol{y}(y_3,y_2,y_1,y_0)$，其相对应的输入掩码为$\boldsymbol{\Gamma x}(\Gamma x_3,\Gamma x_2,\Gamma x_1,\Gamma x_0)$，输出掩码为$\boldsymbol{\Gamma y}(\Gamma y_3,\Gamma y_2,\Gamma y_1,\Gamma y_0)$。以LBC-IoT算法的4 bit的S盒为例，根据S盒计算其线性逼近表，该表中绝对值非零取值共包含两种情况$(2^{-1},2^{-2})$，针对每一个S盒引入两个额外变量(b_0,b_1)来表示其输入和输出掩码的相关性，其对应取值的相关性如式(2)所示。

$$(b_0,b_1)=\begin{cases}(0,0)\to 2^0\\(0,1)\to 2^{-1}\\(1,0)\to 2^{-2}\end{cases}\tag{2}$$

同时，将线性逼近表非零点转化为一个离散点集$\boldsymbol{\Phi}$表示。以输入掩码值0x4和输出掩码值0x4的点$(4,4)$为例，该点对应的相关性取值为2^{-1}，将其转化为一个离散点$(0,1,0,0,0,1,0,0,0,1)$。类似地，将S盒所有线性掩码可能传播模式转化为相同形式$(\Gamma x_3,\Gamma x_2,\Gamma x_1,\Gamma x_0,\Gamma y_3,\Gamma y_2,\Gamma y_1,\Gamma y_0,b_0,b_1)\in F_2^{10}$表示。Sun等[6]通过这种方法将线性逼近表中不为0的点转化为离散点集$\boldsymbol{\Phi}$，该离散点集$\boldsymbol{\Phi}$能够完整描述LBC-IoT算法S盒的线性性质。通过调用SageMath中的Polyhedron函数，将离散点集$\boldsymbol{\Phi}$转化为凸包线性不等式组，并通过Sasaki等[23]提出的不等式约减算法进一步减小该不等式组的规模。经过约减之后得到的凸包线性不等式组规模为16，则通过式(3)可以刻画线性掩码在S盒操作中的传播。

$$\begin{pmatrix}(0,0,0,0,0,0,0,0,-1,-1)\\(-1,-1,0,0,-1,0,-1,0,3,4)\\(0,1,0,-,1,0,2,0,-1,0)\\(0,0,0,0,1,0,1,0,-1,0)\\(-1,1,0,1,0,0,-2,-1,4,2)\\(-3,-1,0,1,1,2,1,-1,4,2)\\(-1,-2,3,0,-1,3,-1,0,4,2)\\(0,0,-2,-1,-1,2,1,1,4,2)\\(-2,0,0,-1,1,0,-1,1,4,2)\\(2,1,0,0,1,0,0,-1,-1,0)\\(1,1,2,-1,-1,0,-3,1,4,2)\\(1,-1,0,0,1,0,0,1,0,0)\\(1,2,-1,0,0,0,1,0,-1,0)\\(1,1,2,1,0,0,-1,1,0,-1)\\(-1,2,1,3,-4,-1,2,0,4,2)\\(2,-3,-1,0,-2,-2,1,0,8,4)\end{pmatrix}\times\begin{pmatrix}\Gamma x_3\\\Gamma x_2\\\Gamma x_1\\\Gamma x_0\\\Gamma y_3\\\Gamma y_2\\\Gamma y_1\\\Gamma y_0\\b_0\\b_1\end{pmatrix}+\begin{pmatrix}1\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\end{pmatrix}\geqslant 0\tag{3}$$

2) 异或操作。假设异或操作$x\oplus y=z$，其对应的线性掩码分别为Γx、Γy和Γz，则式(4)可以刻画线性掩码在异或操作中的传播。

$$\begin{cases}\Gamma x-\Gamma y=0\\\Gamma y-\Gamma z=0\end{cases}\tag{4}$$

3) 分支操作[7]。假设分支操作$\boldsymbol{x}\to(\boldsymbol{x},\boldsymbol{x})$，其对应的线性掩码分别为$\boldsymbol{\Gamma x}$、$\boldsymbol{\Gamma y}$和$\boldsymbol{\Gamma z}$，则式(5)可以

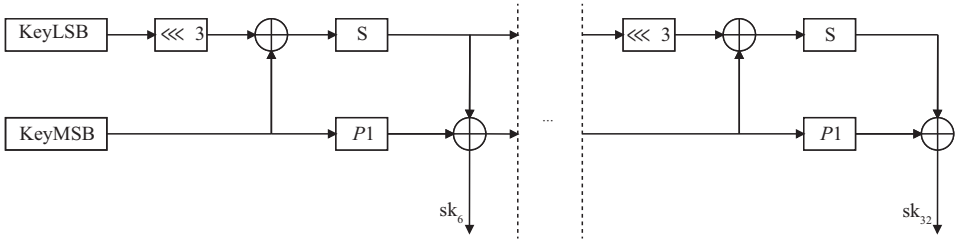

图2　寄存器更新流程

刻画线性掩码在分支操作中的传播。

$$\begin{cases} \boldsymbol{\Gamma x}+\boldsymbol{\Gamma y}-\boldsymbol{\Gamma z}\geqslant 0 \\ \boldsymbol{\Gamma x}-\boldsymbol{\Gamma y}+\boldsymbol{\Gamma z}\geqslant 0 \\ \boldsymbol{\Gamma y}-\boldsymbol{\Gamma x}+\boldsymbol{\Gamma z}\geqslant 0 \\ \boldsymbol{\Gamma x}+\boldsymbol{\Gamma y}+\boldsymbol{\Gamma z}\leqslant 2 \end{cases} \tag{5}$$

4) P置换操作。假设n比特的P置换操作$\boldsymbol{x}\xrightarrow{P}\boldsymbol{y}$，其中$\boldsymbol{x}(x_{n-1},\cdots,x_0)\in F_2^n,\boldsymbol{y}(y_{n-1},\cdots,y_0)\in F_2^n$，其对应的线性掩码分别为$\boldsymbol{\Gamma x}(\Gamma x_{n-1},\cdots,\Gamma x_0)\in F_2^n$和$\boldsymbol{\Gamma y}(\Gamma y_{n-1},\cdots,\Gamma y_0)\in F_2^n$，$P^{-1}$表示$P$置换操作的逆，则式(6)可以刻画线性掩码在$P$置换操作中的传播，其中$0\leqslant j\leqslant 15$。

$$\Gamma x_j-\Gamma y_{P(j)}=0 \tag{6}$$

同时，为了保证模型的有效性以及求出的解为最优解，需要添加额外的初始约束和目标函数。

初始约束。对于分组长度为32 bit的LBC-IoT算法，需要保证初始输入掩码不为0，具体如式(7)所示。

$$\sum_{j=0}^{15}\left(\Gamma X_{0,j}+\Gamma Y_{0,j}\right)>0 \tag{7}$$

目标函数。由于相关性与偏差为线性正相关，将目标函数设置为高相关性，再进一步求取对应的线性偏差。由于LBC-IoT算法每轮共包括4个S盒，所以每轮引进8个辅助变量表示相关性。具体如式(8)所示。

$$\text{Minimize}\sum_{i=0}^{r_m-1}\sum_{j=0}^{3}\left(2b_{i,j}+b_{i,j+1}\right) \tag{8}$$

针对r_1轮的迭代最优线性逼近搜索，需添加初始约束保证线性逼近的输入和输出掩码相同，具体如式(9)所示。

$$\begin{cases} \Gamma X_{0,j}-\Gamma X_{r_1,j}=0 \\ \Gamma Y_{0,j}-\Gamma Y_{r_1,j}=0 \end{cases} \tag{9}$$

基于r_1轮的迭代线性逼近循环构建r_m轮的最优线性逼近，当r_m不为r_1的整数倍时，需要在循环构建的基础上，进一步通过线性逼近的拼接得到r_m轮的最优线性逼近。

3　最优线性逼近的筛选

LBC-IoT算法的分组长度为32 bit，根据线性攻击理论[1]，需要寻找一个绝对值偏差$\varepsilon>2^{-16}$的线性壳，并且基于该线性壳攻击的时间复杂度$O<2^{80}$才是有效攻击。

一般来说，不同的线性逼近在相同扩展情况下所涉及的密钥数量越少，其相应的猜测基数量也越少，对应攻击的时间复杂度也越低。所以在线性逼近轮数相同的情况下，优先选择猜测密钥数量较少的线性逼近来进行密钥恢复攻击[8]，因为攻击轮数可能会更长。因此，采取以下步骤来筛选最优线性逼近。

步骤1　通过第2节中MILP模型同时使用直接搜索和迭代线性逼近循环构建两种方法来获取线性偏差$\varepsilon>2^{-16}$且轮数尽可能长的线性逼近。

步骤2　基于步骤1得到的所有r_m轮线性逼近，在向前扩展r_f轮和向后扩展r_b轮的情况下，先得出每一条线性逼近的初始密钥猜测基。基于该初始密码猜测基预估密钥恢复攻击的时间复杂度O，在时间复杂度$O<2^{80}$的条件下，扩展轮数r_f+r_b应尽可能长。随后，结合最小猜测基技术筛选最优线性逼近，并进行密钥恢复攻击。

3.1　初始密钥猜测基的确定

通过线性逼近表达式对LBC-IoT算法进行攻击，首先将输入和输出掩码分别沿解密和加密方向扩展，得到涉及的明密文和密钥。随后，在此基础上对该算法进行密钥恢复攻击。其中，所涉及的密钥即初始密钥猜测基。

线性掩码的前后扩展等同于以概率1进行传播，在此过程中1 bit线性掩码的取值共有3种状态，具体如表5所示。

表5　线性掩码状态表示

掩码状态	掩码取值
0	0
1	1
*	未知

由于异或与P置换操作不存在掩码之间的运算，因此传播模式并未改变，而S盒与分支操作的扩展可能使传播模式发生改变，具体如下。

1) S盒操作的扩展可能传播模式。利用非受干扰比特[24]的性质，基于S盒的线性逼近表给出所有可能的线性掩码传播模式。以LBC-IoT算法的S盒为例，沿加密和解密方向传播的所有可能传播模式如表6所示。

2) 分支操作的扩展可能传播模式。当输入掩码中存在未知状态的比特时，其输出掩码一定是未知的，所有掩码的可能传播模式如表7所示。

表6　LBC-IoT算法S盒操作的可能传播模式

S（加密方向）		S^{-1}（解密方向）	
输入掩码	输出掩码	输入掩码	输出掩码
0000	0000	0000	0000
0001	1*0*	0001	01**
0010	**10	0100	1**0
0011	**1*	0101	1***
其他	****	其他	****

表7　LBC-IoT算法分支操作的可能传播模式

输入掩码	输出掩码
0 0	0
0 1	1
1 0	1
1 1	0
0 *	*
* 0	*
1 *	*
* 1	*
* *	*

结合表6与表7给出的可能传播模式和LBC-IoT算法结构，可以搜索到扩展过程中涉及的所有密钥，即初始密钥猜测基。

3.2　最小猜测基的搜索

在密钥恢复攻击过程中，通过最小猜测基模型[15]搜索等价密钥，减少初始密钥猜测基的大小，从而在一定程度上降低攻击的时间复杂度。

LBC-IoT算法前5轮的轮密钥$\mathrm{sk}_i(0 \leqslant i < 5)$是由主密钥$K$直接拆分而来的，所以从第5轮开始构建密钥之间的关系。设i轮$(5 \leqslant i < 32)$的密钥为$k_{i-4,79-0}$，其中$k_{i-4,79-40}$为寄存器KeyMSB的状态，$k_{i-4,39-0}$为寄存器KeyLSB的状态，$k_{i-4,79-64}$为第i轮$(5 \leqslant i < 32)$的轮密钥sk_i。每一次寄存器状态更新，密钥之间所形成的关系如下。

1) 循环移位操作。将寄存器KeyMSB和KeyLSB每轮状态最右端的16 bit输出，剩余的高24 bit右移16 bit，所以$k_{i-4,79-56}$和$k_{i-4,39-16}$分别等于$k_{i-3,63-40}$和$k_{i-3,23-0}$，即存在等价关系，如式(10)所示。

$$\begin{cases} k_{i-4,79-56} \Leftrightarrow k_{i-3,63-40} \\ k_{i-4,39-16} \Leftrightarrow k_{i-3,23-0} \end{cases} \tag{10}$$

2) S盒操作。由于寄存器KeyMSB和KeyLSB状态的低16 bit输出，经过循环左移3 bit和异或操作后的结果再经过S盒操作，形成的推导关系如式(11)所示。

$$\begin{cases} k_{i-4,13-10}, k_{i-4,55-52} \Rightarrow k_{i-3,39-36} \\ k_{i-4,9-6}, k_{i-4,51-48} \Rightarrow k_{i-3,35-32} \\ k_{i-4,5-2}, k_{i-4,47-44} \Rightarrow k_{i-3,31-28} \\ k_{i-4,\{1,0,15,14\}}, k_{i-4,43-40} \Rightarrow k_{i-3,27-24} \end{cases} \tag{11}$$

3) 异或操作。这里仅考虑更新寄存器KeyMSB的异或操作，对应形成的推导关系如式(12)所示。

$$\begin{cases} k_{i-3,39-24}, k_{i-4,\{P1(j-40)\}} \Rightarrow k_{i-3,79-64} \\ k_{i-3,39-24}, k_{i-3,79-64} \Rightarrow k_{i-4,\{P1(j-40)\}} \\ k_{i-3,79-64}, k_{i-4,\{P1(j-40)\}} \Rightarrow k_{i-3,39-24} \end{cases} \tag{12}$$

同时，每一次的知识传播都会形成一条推导关系，如式(13)所示。

$$k_{i-4,j}^{t} \Rightarrow k_{i-4,j}^{t+1} \tag{13}$$

其中，$\mathrm{Path}(k_{i,j}^{t})(0 \leqslant t < \alpha, 5 \leqslant i < 32, 0 \leqslant j < 80)$为对应密钥变量所涉及的推导关系，$\mathrm{Relation}(\mathrm{Path}(k_{i,j}^{t}))$为相应推导关系所形成的约束。

基于上述关系，结合文献[11]构建最小猜测基搜索模型并进行有效求解。模型构建流程如算法1所示。

算法1　LBC-IoT算法最小猜测基搜索模型构建流程

输入　初始密钥猜测基K_{guess}，知识传播次数α，攻击轮数$r = r_f + r_m + r_b$

输出　最小猜测基$K_{\min}$

1)for $t = 0 \rightarrow \alpha - 1$ do #α次的知识传播

2)　for $i = 0 \rightarrow r - 5$ do #$r - 5$次更新密钥寄存器的状态

3)　　for $j = 0 \rightarrow 79$ do # 密钥寄存器状态

4)　　　M.var $\leftarrow k_{i,j}^{t}$

5)　　　M.con $\leftarrow \mathrm{Relation}(\mathrm{Path}(k_{i,j}^{t}))$

6)　　end for

7)　end for

8)for $k \in K_{\mathrm{guess}}$ do

9)　M.con $\leftarrow k^{\alpha} = 1$ # 在第α次的知识传播中，初始密钥猜测基中的密钥状态均已知

10)　M.con $\leftarrow \left|K_{\min}\right| \leqslant \left|K_{\mathrm{guess}}\right|$

11)　M.obj $\leftarrow \min \sum_{i=0}^{r-5} \sum_{j=0}^{79} k_{i,j}^{0}$ # 目标函数

12) $K_{\min} \leftarrow$ M.optimize() # 求解模型

13)end for

14)end for

15)return $K_{\min}$

3.3　最优线性逼近的筛选结果

为了筛选出最优线性逼近，首先使用MILP自动化搜索技术，同时采用直接搜索和迭代线性逼近循环构建两种方法得到6条线性偏差为2^{-15}的23轮线性逼近。在此基础上，分别向前扩展3轮和向后扩展4轮，得到每一条线性逼近的初始密钥猜测基，再进一步求取相应的最小猜测基，具体结果如表8所示。

表8　LBC-IoT线性偏差为2^{-15}的23轮线性逼近

输入掩码	输出掩码	构建方法	初始密钥猜测基大小	最小猜测基大小
0x00408001	0x01400001	直接搜索	71	66
0x00000001	0x09400001	直接搜索	67	66
0x00000001	0x08400001	直接搜索	63	60
0x0040c001	0x00400000	迭代线性逼近循环构建	67	59
0x00408001	0x00400000	迭代线性逼近循环构建	61	57
0x00401000	0x00400000	迭代线性逼近循环构建	56	52

所以，本文选择最小猜测基的一条线性逼近进行密钥恢复攻击，该线性逼近的初始密钥猜测基通过5次知识传播，找到了大小为52 bit的最小猜测基，包含的最少密钥比特位置具体如表9所示。

表9　需要猜测的52 bit密钥位置

密钥寄存器更新次数/次	密钥寄存器中需要猜测的比特位置
0	79,78,77,75,73,72,71,70,68,67,66,65,64,61,55,54,52,50,36
22	70
23	75,72,70,64
24	78,77,75,74,73,72,71,70,69,68,65,64
25	79,78,76,75,74,73,72,71,70,69,68,67,66,65,64

在该最小猜测基中，已知密钥变量与可推出的未知目的猜测密钥变量之间的推导关系，具体如表10所示。

表10　密钥推导关系

知识传播次数α	已知密钥	目的密钥
0	$k_{25,\{79,78,76,75,73,72,71,70\}}$	$k_{0,\{76\}}$
3	$k_{25,\{73,72,71,70\}}$	$k_{0,\{74\}}$
3	$k_{25,\{79,78,76,75,67,66,65,64\}}$	$k_{0,\{69\}}$
5	$k_{0,\{73,72,71,70,55,54,52,50\}}$	$k_{24,\{67\}}$

4　LBC算法的30轮密钥恢复攻击

基于筛选得到的最优23轮线性逼近，给出23轮的线性逼近表达式，在此基础上对LBC-IoT算法进行30轮的密钥恢复攻击。

$$(0x0040||0x1000)(X_3||Y_3) = (0x0040||0x0000)(X_{26}||Y_{26}) \tag{14}$$

依据该线性逼近表达式向前和向后分别扩展3轮和4轮，对LBC-IoT算法进行30轮线性攻击，线性掩码的传播过程如图3所示。在线性掩码的扩展传播过程中，轮密钥的异或操作不会影响线性掩码的传播。

令$M = X_0||Y_0$，$C = X_{30}||Y_{30}$。从$X_0||Y_0$加密得到$X_3||Y_3$的过程中，需要知道sk_0中对应的16 bit，sk_1中对应的5 bit和sk_2中对应的1 bit。从$X_{30}||Y_{30}$解密得到$X_{26}||Y_{26}$的过程中，需要知道sk_{29}中对应的16 bit和sk_{28}中对应的13 bit，sk_{27}中对应的4 bit和sk_{26}中对应的1 bit。因此，在密钥恢复攻击过程中，总共涉及56 bit的密钥即初始密钥猜测基。通过最小猜测基技术将猜测密钥量压缩至52 bit，减少了4 bit需要猜测的密钥量，可降低攻击的时间复杂度。

LBC-IoT算法的攻击过程如表11所示，下面对该过程进行详细说明。

步骤1　将攻击过程中涉及的明密文比特状态设为$x_0 = X_{0,15-0}||Y_{0,15-0}$，猜测$\mathrm{sk}_0$中的16 bit密钥即$k_{0,\{79-64\}}$，但利用等价密钥实际猜测13 bit密钥$k_{0,\{79-77,75,73,-70,68-64\}}$，利用密钥加密对应位置的值，可将62 bit的$x_0$压缩成51 bit的$x_1$，这一步的时间复杂度为$2^{13} \times 2^{62} \div 30 \approx 2^{70.1}$。

步骤2　猜测sk_{29}即$k_{25,79-64}$，利用密钥加密对应位置的值，可将51 bit的x_1压缩成47 bit的x_2，这一步的时间复杂度为$2^{29} \times 2^{51} \div 30 \approx 2^{75.1}$。

步骤3　猜测密钥sk_1即$k_{0,\{61,55,54,52,50\}}$，利用密钥解密对应位置的值，可将47 bit的x_2压缩成36 bit的x_3，这一步的时间复杂度为$2^{34} \times 2^{47} \div 30 \approx 2^{76.1}$。

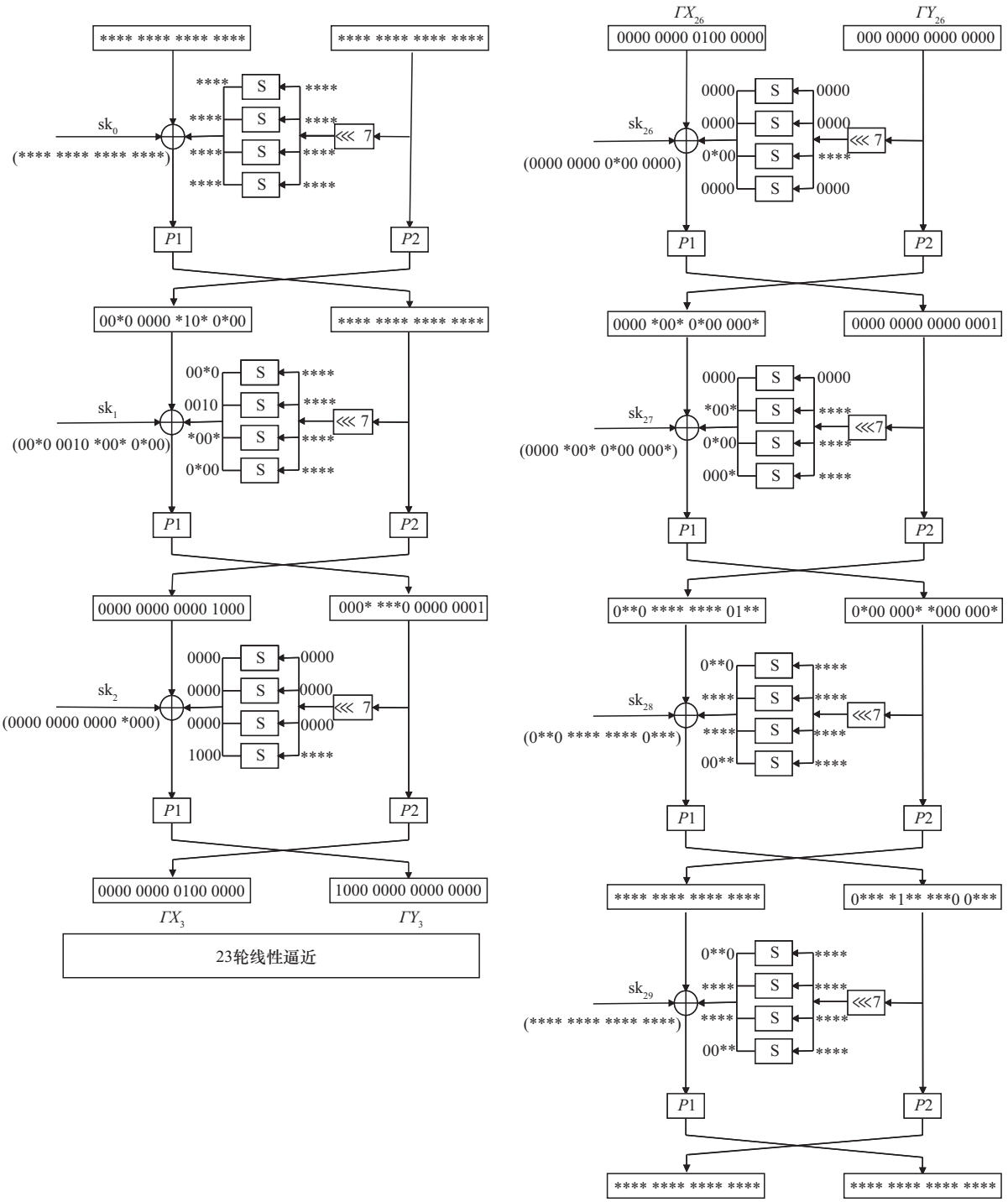

图3　LBC-IoT算法的30轮线性攻击

步骤4　猜测sk_{28}中的13 bit密钥即$k_{24,\{78,77,75-68,65,64\}}$与等价密钥$k_{24,\{67\}}$，利用密钥加密对应位置的值，可将36 bit的$x_3$压缩成23 bit的$x_4$，这一步的时间复杂度为$2^{46}\times 2^{36}\div 30\approx 2^{77.1}$。

步骤5　猜测sk_{27}中的4 bit密钥即$k_{23,\{75,72,70,64\}}$，利用密钥解密对应位置的值，可将23 bit的x_4压缩成11 bit的x_5，这一步的时间复杂度为$2^{50}\times 2^{23}\div 30\approx 2^{68.1}$。

步骤6　猜测密钥sk_2中的1 bit密钥即$k_{0,\{36\}}$，利用密钥解密对应位置的值，可将11 bit的x_5压缩成7 bit的x_6，这一步的时间复杂度为$2^{51}\times 2^{11}\div 30\approx 2^{57.1}$。

表11　　LBC-IoT算法的攻击过程

步骤	猜测密钥	中间状态	时间复杂度
1	$k_{0,\{79-77,75,73,-70,68-64\}}$	$x_1 = X_{1,\{13,7,6,4,2\}} \| Y_{1,\{15-5,3,0\}} \| X_{30,15-0} \| Y_{30,\{15-0\}}$	$2^{13} \times 2^{62} \div 30 \approx 2^{70.1}$
2	$k_{25,79-64}$	$x_2 = X_{1,\{13,7,6,4,2\}} \| Y_{1,\{15-5,3,0\}} \| X_{29,15-0} \| Y_{29,\{14-11,9-5,2-0\}}$	$2^{29} \times 2^{51} \div 30 \approx 2^{75.1}$
3	$k_{0,\{61,55,54,52,50\}}$	$x_3 = X_{2,\{4\}} \| Y_{2,\{12-8,0\}} \| X_{28,\{14,13,11-4,1,0\}} \| Y_{28,\{14,8,7,0\}}$	$2^{34} \times 2^{47} \div 30 \approx 2^{76.1}$
4	$k_{24,\{78,77,75-68,65,64\}}$	$x_4 = X_{2,\{4\}} \| Y_{2,\{12-8,0\}} \| X_{27,\{11,8,6,0\}} \| Y_{27,\{0\}}$	$2^{46} \times 2^{36} \div 30 \approx 2^{77.1}$
5	$k_{23,\{75,72,70,64\}}$	$x_5 = X_{2,\{4\}} \| Y_{2,\{12-8,0\}} \| X_{27,\{11,8,6,0\}} \| Y_{27,\{0\}}$	$2^{50} \times 2^{23} \div 30 \approx 2^{68.1}$
6	$k_{0,\{36\}}$	$x_6 = X_{3,\{6\}} \| Y_{3,\{15\}} \| X_{27,\{11,8,6,0\}} \| Y_{27,\{0\}}$	$2^{51} \times 2^{11} \div 30 \approx 2^{57.1}$
7	$k_{22,\{70\}}$	$x_7 = X_{3,\{6\}} \| Y_{3,\{15\}} \| X_{26,\{6\}}$	$2^{52} \times 2^{7} \div 30 \approx 2^{54.1}$

步骤7　猜测密钥sk_{26}中的1 bit密钥即$k_{22,\{70\}}$，利用密钥解密对应位置的值，可将7 bit的x_6压缩成3 bit的x_7，这一步的时间复杂度为$2^{52} \times 2^{7} \div 30 \approx 2^{54.1}$。

攻击复杂度分析。由于攻击选择的23轮线性逼近的线性偏差为2^{-15}，所以攻击的数据复杂度约为2^{30}个已知明文；其攻击的整体时间复杂度约为$2^{70.1} + 2^{75.1} + 2^{76.1} + 2^{77.1} + 2^{68.1} + 2^{57.1} + 2^{54.1} \approx 2^{77.9}$次30轮加密；攻击原本需要猜测的密钥数量为56 bit，通过最小猜测基技术减少了4 bit的密钥猜测数量，所以攻击的存储复杂度由2^{56}降低至2^{52}。

5　结束语

本文基于MILP自动化搜索技术，同时采用直接搜索和迭代线性逼近循环构建两种方法共得到LBC-IoT算法的6条线性偏差为2^{-15}的23轮线性逼近，结合最小猜测基技术从中筛选出了唯一一条猜测基大小为52 bit的最优23轮线性逼近。在此基础上，向上向下分别扩展3轮和4轮，进行30轮的密钥恢复攻击，其中攻击的数据复杂度为2^{30}KP，时间复杂度约为$2^{77.9}$次30轮加密，存储复杂度为2^{52}。这是LBC-IoT算法目前已知最长轮数的攻击，相比之前的线性攻击轮数提升了4轮。对于已有的线性攻击结果，未来可考虑进一步借助FWT等技术进一步优化攻击的时间复杂度。

参考文献：

[1] Matsui M. Linear cryptanalysis method for DES cipher[C]//Advances in Cryptology-EUROCRYPT'93. Berlin: Springer, 1994: 386-397.

[2] Matsui M. On correlation between the order of S-boxes and the strength of DES[C]//Advances in Cryptology-EUROCRYPT'94. Berlin: Springer, 1995: 366-375.

[3] Mouha N, Wang Q J, Gu D W, et al. Differential and linear cryptanalysis using mixed-integer linear programming[C]//Conference on Information Security and Cryptology. Berlin: Springer, 2012: 57-76.

[4] Gerault D, Minier M, Solnon C. Constraint programming models for chosen key differential cryptanalysis[C]//Principles and Practice of Constraint Programming. Berlin: Springer, 2016: 584-601.

[5] Mouha N, Preneel B. Towards finding optimal differential characteristics for ARX: application to Salsa20[J]. IACR Cryptology ePrint Arch, 2013, 328: 1-29.

[6] Sun S W, Hu L, Wang P, et al. Automatic security evaluation and (related-key) differential characteristic search: application to SIMON, PRESENT, LBlock, DES(L) and other bit-oriented block ciphers[C]//Advances in Cryptology-ASIACRYPT 2014. Berlin: Springer, 2014: 158-178.

[7] Fu K, Wang M Q, Guo Y H, et al. MILP-based automatic search algorithms for differential and linear trails for speck[C]//Fast Software Encryption. Berlin: Springer, 2016: 268-288.

[8] Zong R, Dong X Y, Chen H F, et al. Towards key-recovery-attack friendly distinguishers: application to GIFT-128[J]. IACR Transactions on Symmetric Cryptology, 2021: 156-184.

[9] 石康康, 任炯炯, 陈少真. 基于MILP的PICO算法差分和线性区分器的搜索[J]. 密码学报, 2023, 10(5): 910-921.
Shi K K, Ren J J, Chen S Z. MILP-based search for differential and linear distinguishers of PICO algorithm[J]. Journal of Cryptologic Research, 2023, 10(5): 910-921.

[10] Flórez-Gutiérrez A, Naya-Plasencia M. Improving key-recovery in linear attacks: application to 28-round PRESENT[C]//Advances in Cryptology-EUROCRYPT 2020. Berlin: Springer, 2020: 221-249.

[11] Collard B, Standaert F X, Quisquater J J. Improving the time complexity of matsui's linear cryptanalysis[C]//Information Security and Cryptology-ICISC 2007. Berlin: Springer, 2007: 77-88.

[12] Dunkelman O, Keller N, Shamir A. Improved single-key attacks on 8-round AES-192 and AES-256[C]//Advances in Cryptology-ASIACRYPT 2010. Berlin: Springer , 2010: 158-176.

[13] Bogdanov A, Tischhauser E. On the wrong key randomisation and key equivalence hypotheses in matsui's algorithm 2[C]//Fast Software En-

cryption. Berlin: Springer, 2014: 19-38.

[14] Lin L, Wu W L, Zheng Y F. Automatic search for key-bridging technique: applications to LBlock and TWINE[C]//Fast Software Encryption. Berlin: Springer, 2016: 247-267.

[15] Cen Z, Feng X T, Wang Z Y, et al. Minimizing deduction system and its application[J]. arXiv Preprint, arXiv: 2006.05833, 2020.

[16] Hadipour H, Eichlseder M. Autoguess: a tool for finding guess-and-determine attacks and key bridges[C]//Applied Cryptography and Network Security. Berlin: Springer, 2022: 230-250.

[17] Ramadan R A, Aboshosha B W, Yadav K, et al. LBC-IoT: lightweight block cipher for IoT constraint devices[J]. Computers, Materials & Continua, 2021, 67(3): 3563-3579.

[18] 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: ACM Press, 2015: 1-6.

[19] Yasushi S, Igarashi Y. MILP-based linear attack on lightweight block cipher LBC-IoT[C]//Proceedings of the International Conference on on Cryptography and Information Security. 2022: 1-11.

[20] Chan Y Y, Khor C Y, Khoo B T, et al. On the resistance of new lightweight block ciphers against differential cryptanalysis[J]. Heliyon, 2023, 9(4): e15257.

[21] Grochal P, Stanek M. Differential MITM attacks on SLIM and LBC-IoT[J]. Cryptology ePrint Archive, 2024, 1877: 1-18.

[22] Gurobi. Gurobi optimization[R]. 2025.

[23] Sasaki Y, Todo Y. New impossible differential search tool from design and cryptanalysis aspects[C]//Advances in Cryptology-EUROCRYPT 2017. Berlin: Springer, 2017: 185-215.

[24] Tezcan C. Improbable differential attacks on present using undisturbed bits[J]. Journal of Computational and Applied Mathematics, 2014, 259: 503-511.

[作者简介]

李灵琛（1988– ），女，广西桂林人，博士，桂林电子科技大学讲师、硕士生导师，主要研究方向为密码学与信息安全。

陈佐甲（2000– ），男，湖南长沙人，桂林电子科技大学硕士生，主要研究方向为分组密码的安全性分析。

韦永壮（1976– ），男，广西田阳人，博士，桂林电子科技大学教授、博士生导师，主要研究方向为密码学与信息安全。

叶涛（1991– ），男，黑龙江伊春人，博士，桂林电子科技大学讲师、硕士生导师，主要研究方向为对称密码算法的设计与分析。