基于 SAT 的 ARX 不可能差分和零相关区分器的自动化搜索

任炯炯,张仕伟,李曼曼,陈少真
(战略支援部队信息工程大学,河南郑州 450001)

摘 要: ARX(Addition,Rotation,Xor)算法基于模整数加,异或加和循环移位三种运算,便于软硬件的快速实现. 不可能差分分析和零相关分析是攻击 ARX 的有效方法,攻击的关键是搜索更长轮数、更多数量的不可能差分和零相关区分器. 目前很多的搜索方法都没有充分考虑非线性组件的性质,往往不能搜索得到更好、更准确的区分器. 本文提出了基于 SAT(Satisfiability)的 ARX 不可能差分和零相关区分器的自动化搜索算法. 通过分析 ARX 算法组件的性质,特别是常规模加和密钥模加这两种非线性运算差分和线性传播的特性,给出了高效简单的 SAT 约束式. 在此基础上,建立 SAT 模型进行区分器的搜索. 作为应用,本文首次给出了 Chaskey 算法 13 条 4 轮不可能差分和 1 条 4 轮零相关区分器;首次给出了 SPECK32 算法 10 条 6 轮零相关区分器和 SPECK48 算法 15 条 6 轮零相关区分器;在较短的时间内,给出了 HIGHT 算法 17 轮的不可能差分和零相关区分器. 与现有结果相比,无论是区分器的条数,还是搜索区分器的时间均有明显的提升. 此外,通过重新封装求解器 STP 的输出接口,建立了自动化的 SAT\SMT 分析模型,能够给出 ARX 算法在特殊输入输出差分和掩码集合下,不可能差分和零相关区分器轮数的上界.

关键词: 不可能差分区分器; 零相关区分器; ARX; Chaskey; SPECK; HIGHT; SAT 求解器
中图分类号: TN918.1 **文献标识码:** A **文章编号:** 0372-2112 (2019)12-2524-09
电子学报 URL: http://www.ejournal.org.cn **DOI:** 10.3969/j.issn.0372-2112.2019.12.010

SAT-Based Automatic Search for Impossible Differentials and Zero-Correlation Linear Approximations in ARX

REN Jiong-jiong, ZHANG Shi-wei, LI Man-man, CHEN Shao-zhen
(*PLA Information Engineering University, Zhengzhou, Henan* 450001, *China*)

Abstract: ARX(Addition, Rotation, Xor) algorithms are based on three operations: modular addition, exclusive-OR and bitwise rotation, which execute very fast in both software and hardware. Impossible differential cryptanalysis and zero-correlation linear cryptanalysis are among the most powerful attacks for ARX ciphers. The key problem for the attacks is searching more and longer impossible differentials and zero-correlation linear approximations. Although there are many automatic search algorithms, these approaches do not fully utilize the non-linear component properties, which cannot reach their potential. A SAT(Satisfiability)-based model to automatic search for impossible differentials and zero-correlation linear approximations in ARX is proposed. By exploiting behaviors of every component, especially the differential and linear propagation properties through general modular addition and key modular addition operations, we generate computable and easily implementable constraints and establish the SAT-based model. As applications, we apply our model to Chaskey, SPECK and HIGHT. For Chaskey, we are able to find 13 4-round impossible differentials and 1 4-round zero-correlation linear approximation for the first time. For SPECK, we first find 10 6-round zero-correlation linear approximations of SPECK32 and 15 6-round zero-correlation linear approximations of SPECK48. Besides, we find 4 17-round impossible differentials and zero-correlation linear approximations of HIGHT in a few minutes. Compared with the existing results, both the number and the search time of distinguishers are significantly improved. In addition, by repackaging the output interface

收稿日期:2018-07-11;修回日期:2019-02-26 责任编辑:梅志强
基金项目:国家密码发展基金 (No. MMJJ20180203);数学工程与先进计算国家重点实验室开放基金(No. 2018A03);信息保障技术重点实验室开放基金(No. KJ-17-002)

of the STP solver, we establish the automated SAT\SMT model, which can give the upper bound of rounds of impossible differentials and zero-correlation linear approximations under special input and output differential and mask sets for ARX algorithm.

Key words: impossible differential; zero-correlation linear approximation; ARX; Chaskey; SPECK; HIGHT; SAT-solver

1 引言

ARX(Addition, Rotation, Xor)算法基于模整数加(Addition),循环移位(Rotation)和异或加(Xor)和三种运算,其中只有模加为非线性运算. 由于 ARX 算法运算简单,便于软硬件的快速实现,出现了很多基于 ARX 结构的密码算法,如 SPECK[1]、LEA[2]、TEA[3]、XTEA[4]、HIGHT[5]、Chaskey[6]和 SPARX[7]等. 对 ARX 算法的设计和安全性的分析具有重要的意义.

不可能差分分析最早由 Biham[8]和 Knudsen[9]提出,与差分分析搜索最大概率的差分特征来恢复正确的密钥不同,不可能差分分析的目的在于寻找概率为零的差分特征来排除错误密钥. 目前不可能差分分析被广泛应用于各种结构的分组密码分析中,如 AES[10]和 QARMA[11]等. 零相关分析方法由 Bogdanov 和 Rijmem[12]于 2012 年提出,自提出以来受到广泛地关注并取得很多好的攻击结果. 与线性分析所利用的高偏差的线性逼近关系不同,零相关线性分析主要利用偏差为零的线性逼近获得部分和全部的密钥信息. 不可能差分和零相关分析的关键是寻找有效的不可能和零相关区分器,区分器的好坏直接关系到密码攻击的效果,只有找到更长轮数和更多数量的区分器,才能得到更好的攻击结果. 最早区分器的搜索基于中间相错的思想,选择特定的输入输出差分(掩码),以差分概率为 1 或非零的偏差从明密文两端同时传递,并在中间相遇,如果在中间位置产生矛盾,则构成一条不可能差分和零相关路径. 随后,2003 年 Kim 等[13]人总结前人方法,提出了新的不可能差分路径的搜索算法 μ 方法,μ 方法只能针对特定结构的密码算法. 为解决这个问题,Luo 等[14]人提出了 UID 方法,Wu 等[15]人提出了扩展方法,可应用于更多结构的密码算法. 但这些方法都没有充分考虑非线性组件的差分和线性传播的性质,导致在搜索区分器时,往往不能得到更好、更准确的结果. 2016 年 Cui 等[16]人基于 MILP 模型提出了不可能差分和零相关路径自动化搜索算法,该算法将差分和线性掩码在轮函数中的传递描述为 MILP 约束式,再通过商用求解器 Gurobi 求解. 随后在 2017 欧密会议上,Yu 等[17]人同样基于 MILP 提出了新的不可能差分区分器的搜索工具,并从设计和分析的角度给出了更多的应用结果.

本文基于 SAT 给出 ARX 算法不可能差分和零相关路径的自动化搜索模型. SAT(Satisfiability)是布尔可满足性问题,作为 NPC 问题,主要考虑一组真值指派是否满足给定的布尔方程组. Mouha 等[18]人首次将 SAT 应用到 Salsa2 最佳差分路径的搜索. 接着 Song 等[19]人利用 Mouha et al. 框架,对 SPECK 等 ARX 算法进行了更多轮数差分路径搜索. Liu 等[20]人首次利用 SAT 实现了 ARX 型密码算法线性路径的搜索. 2015 年美密会,Stefan 等[21]人基于 SAT/SMT(Satisfiability Modulo Theories)工具给出 SIMON 新的差分与线性路径自动化搜索方法. 综合考虑 SAT 求解的优点,本文提出基于 SAT 的 ARX 不可能差分和零相关路径的自动化搜索算法.

首先,我们研究了 ARX 算法组件差分和线性传播的性质,特别针对非线性组件模加运算,给出了常规模加和密钥模加这两种不同情况下,模加运算差分和线性传播性质,进一步给出刻画这两类模加运算高效简单的 SAT 约束式. 其次,针对不同的 ARX 算法,在特定的轮数下,建立算法轮函数差分和线性传播的约束式,利用 SAT\SMT 求解器 STP 将这些等式转化为合取范式(CNF),遍历特殊的输入和输出差分和掩码的集合搜索不可能差分和零相关区分器. 作为应用,我们在个人笔记本(Intel Core i7-6700 (3.4GHz),7.8GBRAM,Ubuntu 14.04.3 LTS)搜索了 ARX 算法 Chaskey、SPECK 和 HIGHT 不可能差分和零相关区分器. 与已有结果相比,我们首次给出了 Chaskey 算法 13 条 4 轮不可能差分区分器和 1 条 4 轮零相关区分器;首次给出了 SPECK32 算法 10 条 6 轮零相关区分器和 SPECK48 算法 15 条 6 轮零相关区分器;给出了 HIGHT 算法 17 轮不可能差分和零相关区分器,且搜索时间明显优于 CUI 等[16]人基于 MILP 的搜索时间. 具体结果如表 1 所示. wt(α)和 wt(β)分别表示输入差分 α 和输出差分 β 的汉明重量,wt(u)和 wt(v)分别表示输入掩码 u 和输出掩码 v 的汉明重量.

值得一提的是,我们利用管道技术,重新封装了求解器 STP 的输出接口,使得建立的 SAT\SMT 模型能够智能化、自动化调用求解器. 可以在短时间内判断对于任意轮数,任意一对输入输出差分和掩码,是否构成可能的差分和线性路径. 利用该性质,我们遍历特殊集合的输入输出差分和掩码,进而可以给出不同 ARX 算法不可能差分和零相关区分器轮数的上界.

表 1 基于 SAT 自动化搜索的结果汇总

算法	不可能差分区分器				零相关区分器			
	轮数	差分类型	数量	时间(秒)	轮数	掩码类型	数量	时间(秒)
Chaskey	4	$wt(\alpha)=1, wt(\beta)=1$	13	400.5	4	$wt(u)=1, wt(v)=1$	1	2866.3
SPECK32	6	$wt(\alpha)=1, wt(\beta)=1$	3	13.7	6	$wt(u)=1, wt(v)=2$	10	288.4
SPECK48	6	$wt(\alpha)=1, wt(\beta)=1$	20	32.9	6	$wt(u)=1, wt(v)=2$	15	1357.4
HIGHT	17	$wt(\alpha)=1, wt(\beta)=1$	4	264	17	$wt(u)=1, wt(v)=1$	4	856.6

2 基于 SAT 的 ARX 不可能差分区分器的搜索

本节分析了 ARX 算法组件,特别是不同类型非线性组件的差分传播性质,给出刻画不同组件性质的 SAT 约束式,构建基于 SAT\SMT 的不可能差分区分器的搜索模型.

2.1 基本符号与定义

设 F_2 代表二元有限域;$\boldsymbol{F}_2^n$ 代表 F_2 上的 n 维向量;对于 $a\in\boldsymbol{F}_2^n$,用 $wt(a)$ 代表 a 的汉明重量;$\oplus$ 表示 $\boldsymbol{F}_2^n$ 中的异或运算;$\boxplus$ 表示 $\boldsymbol{F}_2^n$ 中的模加运算;$\wedge$ 表示 $\boldsymbol{F}_2^n$ 中的与运算;$\lll$ 代表 $\boldsymbol{F}_2^n$ 中的左循环移位运算;对于布尔函数 $f:\boldsymbol{F}_2^n\to\boldsymbol{F}_2^n$,对于给定的输入差分 a 和输出差分 γ,定义差分传播概率:

$$P(\alpha\to\gamma)=2^{-n}\cdot\#\{x: f(x)\oplus f(x\oplus\alpha)=\gamma\},$$

其中#表示集合的元素个数.

定义 1 模 2^n 加运算($\boxplus$)的异或差分转移概率是指输入差分 α 和 β 娄 通过模 2^n 加运算后传播到输出差分 γ 的概率,记为 $xdp^+(\alpha,\beta\to\gamma)$. 设 $x,y\in\{0,1\}^n$,(x,y) 是模 2^n 加运算的输入,概率值 $xdp^+(\alpha,\beta\to\gamma)$ 可以通过以下方式得到:

$$xdp^+(\alpha,\beta\to\gamma)=2^{-2n}\cdot\#\{(x,y):((x\oplus\alpha)+(y\oplus\beta))\oplus(x+y)=\gamma\},$$

若存在输入满足该差分转移,则 $xdp^+(\alpha,\beta\to\gamma)$ 大于零,称该差分转移是有效的.

2.2 ARX 基本组件差分性质传播的约束式

性质 1 对于异或运算($\oplus$),输入差分为 α 和 β,输出差分为 γ,则差分传播概率 $P(\alpha,\beta\to\gamma)=1$ 当且仅当 $\gamma=\alpha\oplus\beta$.

性质 2 对于循环移位运算($\lll t$),输入差分为 α,输出差分为 γ,则差分传播概率 $P(\alpha\to\gamma)=1$ 当且仅当 $\gamma=\alpha\lll t$.

对于非线性组件模加运算($\boxplus$),根据 ARX 算法结构的不同,分为两种:一种是常规模加运算,输入 $x,y\in\boldsymbol{F}_2^n$,模加运算的输出 $z\in\boldsymbol{F}_2^n$,满足 $z=(x+y)\bmod 2^n$,比如 LEA、SPECK 和 Chaskey 算法等;另一种是密钥模加运算,输入 $x,K\in\boldsymbol{F}_2^n$,其中轮子密钥 K 是常数,输出 $z=(x+K)\bmod 2^n$,比如 HIGHT、TEA 和 XTEA 算法等. 为了建立不同结构 ARX 算法不可能差分路径的搜索模型,我们研究不同情况模加运算的差分传播规律,给出刻画组件差分性质高效简单的 SAT 约束式.

对于常规模加运算,Helger Lipmaa 等[22] 人给出了模加运算差分概率的快速计算方法.

性质 3[22] 对于模加运算 $z=(x+y)\bmod 2^n$,输入差分为 α 和 β,输出差分为 γ,则 $(\alpha,\beta\to\gamma)$ 为一条有效差分传递路径,当且仅当

$$eq(\alpha\ll 1,\beta\ll 1,\gamma\ll 1)\wedge(\alpha\oplus\beta\oplus\gamma\oplus(\beta\ll 1))=0,$$

其中 $eq(x,y,z):=(\neg x\oplus y)\wedge(\neg x\oplus z)$.

根据性质 3,刻画常规模加运算有效差分传播 $(\alpha,\beta\to\gamma)$ 的约束式是:

$$(\neg(\alpha\ll 1)\oplus(\beta\ll 1))\wedge(\neg(\alpha\ll 1)\oplus(\gamma\ll 1))\wedge(\alpha\oplus\beta\oplus\gamma\oplus(\beta\ll 1))=0 \quad (1)$$

对于密钥模加运算,需要考虑保证 $(\alpha,K\to\gamma)$ 为有效的差分传播路径满足的条件,首先成立以下性质.

性质 4 对于密钥模加运算 $z=(x+K)\bmod 2^n$,设输入差分为 α,输出差分为 γ,则 $(\alpha,K\to\gamma)$ 为一条有效的差分传递路径,当且仅当差分 α 和 γ 的最低非零比特位相同.

根据性质 4,为了得到有效的差分路径,需要保证输入差分和输出差分的最低非零比特位相同. 我们首先给出数论函数 $pot_p n$ 的定义,随后给出定理 1,利用 $pot_2 n$ 的值判断输入和输出差分的最低非零比特是否相同.

定义 2 对于给定的素数 p,设 $p^m\parallel n$,即 p^m 整除 n,p^{m+1} 不整除 n,则记 $pot_p n=m$.

定理 1 对于给定的 n 比特的输入差分 α 和输出差分 γ,则 $(\alpha,K\to\gamma)$ 为一条有效的差分传递路径,当且仅当 $pot_2\alpha=pot_2\gamma$.

证明 对于给定 n 比特的输入差分 α 和输出差分 γ,若 $(\alpha,K\to\gamma)$ 为有效的差分传递路径,则 α 和 γ 的最低非零比特位相同,不妨假设为第 t 位,则 α 和 γ 可以表示为:

$$\alpha=2^t+\sum_{i=t+1}^{n-1}\alpha_i 2^i(\alpha_i=0,1),$$

$$\gamma=2^t+\sum_{i=t+1}^{n-1}\gamma_i 2^i(\gamma_i=0,1),$$

则 $2^t\parallel\alpha, 2^t\parallel\gamma$,所以 $pot_2\alpha=pot_2\gamma$.

设 $\alpha=\sum_{i=0}^{n-1}\alpha_i 2^i(\alpha_i=0,1)$, $\gamma=\sum_{i=0}^{n-1}\gamma_i 2^i(\gamma_i=0,1)$, 当 $\mathrm{pot}_2\alpha=\mathrm{pot}_2\gamma=t$ 时,则 $2^t\parallel\alpha, 2^t\parallel\gamma$,因而 $\alpha_i=\gamma_i=0$ $(0\le i<t)$,此时 α 和 γ 的最低非零比特 $\alpha_t=\gamma_t$. 证毕.

根据定理 1,为了判断 $(\alpha,K\rightarrow\gamma)$ 是否为有效的差分传递路径,需要计算 $\mathrm{pot}_2\alpha$ 和 $\mathrm{pot}_2\gamma$ 的值,我们按照比特运算的规律计算 $\mathrm{pot}_2 n$ 的值,进而给出 5 个简单的能被 SAT 求解器识别的约束式刻画密钥模加运算的差分传递性质.

性质 5 设 $(\boldsymbol{\alpha},\boldsymbol{K}\rightarrow\boldsymbol{\gamma})$ 为经过密钥模加的差分传递,其中 $\boldsymbol{\alpha}$ 和 $\boldsymbol{\gamma}$ 分别为为输入和输出差分向量. z 为 $n+1$ 比特的向量. 为了获得一条差分概率大于 0 的有效差分传播路径,差分 $\boldsymbol{\alpha}$ 和 $\boldsymbol{\gamma}$ 需要满足以下约束式

$$\begin{cases} z\wedge(z-1)=0,\\ z\leqslant 2^n,\\ z\geqslant 1,\\ \boldsymbol{\alpha}\wedge(z-1)=0,\boldsymbol{\gamma}\wedge(z-1)=0,\\ \boldsymbol{\alpha}\wedge z>0,\boldsymbol{\gamma}\wedge z>0. \end{cases} \tag{2}$$

2.3 不可能差分区分器的搜索算法

对于不同的 ARX 算法,我们用约束式(1)描述 $(\alpha,\beta\rightarrow\gamma)$ 差分的有效传递,用约束式(2)描述 $(\alpha,K\rightarrow\gamma)$ 差分的有效传递. 通过将轮函数中所有线性和非线性组件的掩码传播约束式结合起来,进而得到刻画算法 r 轮差分传播概率不为 0 的方程式,存放在名为 input. cvc 文件. 通过穷举特定的输入和输出差分集合 (A,B),利用 SAT\SMT 求解器 STP 对文件进行求解,判断输入输出差分是否构成可能的差分路径,若求解结果返回 Invalid,则输出不可能差分区分器. 算法 1 如下所示.

算法 1　不可能差分区分器的搜索算法

输入:输入差分集合 A,输出差分集合 B
输出:不可能差分区分器 $(\Delta a,\Delta b)$
//ARX 算法的分组长度为 n,将刻画 ARX 算法轮函数差分传播性质的约束式写入 input. cvc
1. for $\Delta a_i\in A$ do
2.　for $\Delta b_i\in B$ do
3.　　Modify constraints on the fixed input and output differences of input. cvc
4.　　m = dispose(input. cvc)
5.　　if m = = Invalid　输出不可能差分区分器 $(\Delta a,\Delta b)$
6.　　else then continue
7.　end
8. end

算法 1 为遍历算法,复杂度与输入输出差分对的选取有关,我们通常选取特定集合来遍历. 其中 dispose 函数将所列约束式转化为合取范式(CNF),再将 CNF 送入 SAT 求解器 STP,验证其可否满足.

2.4 算法实现

在具体实现过程,我们利用非、异或、移位等简单运算符,将不同非线性组件的差分传播性质封装成模块,方便不同算法直接拼搭使用. 此外,由于 STP 求解器每次只能单独求解一个 SAT 问题,没有循环操作表示函数,在搜索不可能差分区分器时,无法实现搜索过程的自动化. 我们利用管道技术,重新封装定义了 STP 软件的输出接口,新定义了循环操作函数,方便调用 STP 遍历求解 SAT 问题,实现了 SAT 求解的自动化. 不仅可以判断任意给定的输入和输出差分能否构成可能的路径,而且可以搜索不同算法选定任意轮数和参数时的不可能差分区分器,给出特殊条件下算法不可能差分区分器轮数的上界.

3 基于 SAT 的 ARX 零相关区分器的搜索

本节给出刻画常规模加和密钥模加运算线性传播性质的 SAT 约束式,构建基于 SAT\SMT 零相关区分器的搜索模型.

3.1 基本符号与定义

对于 n 维向量 $\boldsymbol{x}$ 和 $\boldsymbol{y}$,定义 $\boldsymbol{x}\preceq\boldsymbol{y}\Leftrightarrow x_i\le y_i$, $\forall i\in\{0,1,\cdots,n-1\}$;定义特征函数 $1_{x\preceq y}=\begin{cases}1,\text{当 }\boldsymbol{x}\preceq\boldsymbol{y}\\0,\text{其他}\end{cases}$. 对于布尔函数 $f:\boldsymbol{F}_2^n\rightarrow\boldsymbol{F}_2^n$,对于给定的输入掩码 $\boldsymbol{u}$ 和输出掩码 $\boldsymbol{w}$,定义线性相关度:

$$C(\boldsymbol{u}\rightarrow\boldsymbol{v})=2^{-n}\hat{f}(\boldsymbol{u},\boldsymbol{w}),$$

其中 Walsh 谱值 $\hat{f}(\boldsymbol{u},\boldsymbol{w})=\sum_{\boldsymbol{x}\in\boldsymbol{F}_2^n}(-1)^{(\boldsymbol{w}\cdot f+\boldsymbol{u}\cdot\boldsymbol{x})}$.

定义 3 设 $x,y,z\in\{0,1\}^n$, (x,y) 是模 2^n 加运算的输入,z 是模 2^n 加的输出,$z=(x+y)\bmod 2^n$,对于给定的输入掩码 u 和 v 和输出掩码 w,线性相关度 $\mathrm{xlc}^+(u,v\rightarrow w)$ 通过以下方式得到:

$$\mathrm{xlc}^+(u,v\rightarrow w)=2^{-2n}\cdot\sum_{x\in\boldsymbol{F}_2^n,y\in\boldsymbol{F}_2^n}(-1)^{w\cdot z\oplus u\cdot x\oplus v\cdot y},$$

若 $\mathrm{xlc}^+(u,v\rightarrow w)$ 不等于 0,则称该线性传播有效.

3.2 ARX 基本组件线性性质传播的约束式

性质 6 对于异或运算(⊕),输入掩码为 u 和 v,输出掩码为 w,则线性相关度 $C(u,v\rightarrow w)\neq 0$ 当且仅当 $u=v=w$.

性质 7 对于循坏移位运算($<<<t$),输入掩码为 u,输出掩码为 w,则线性相关度 $C(u,v\rightarrow w)\neq 0$ 当且仅当 $w=u<<<t$.

对于非线性组件模加运算(田),需要研究常规模加和密钥模加运算两种不同情况的线性传播性质,并给出相应的 SAT 约束式. 对于常规模加运算,Schulte-Geers[23] 给出了直接计算常规模加运算线性相关度的方法.

性质 8[23] 设$(\boldsymbol{u},\boldsymbol{v}\rightarrow\boldsymbol{w})$为经过模加 2^n 的掩码传递,其中为 $\boldsymbol{u},\boldsymbol{v}$ 为输入掩码向量,$\boldsymbol{w}$ 为输出掩码向量. z 是一个满足下式的 n 维向量

$$z\oplus(z\ll 1)\oplus((\boldsymbol{u}\oplus\boldsymbol{v}\oplus\boldsymbol{w})\ll 1)=0, z_{n-1}=0,$$

则

$$\mathrm{cor}(\boldsymbol{u},\boldsymbol{v},\boldsymbol{w})=1_{\boldsymbol{u}\oplus\boldsymbol{w}\leq z}1_{\boldsymbol{v}\oplus\boldsymbol{w}\leq z}(-1)^{(\boldsymbol{u}\oplus\boldsymbol{w})\cdot(\boldsymbol{w}\oplus\boldsymbol{v})}2^{-\mathrm{wt}(z)}.$$

在性质 8 的基础上,为了获得一条有效的掩码传播路径,Liu 等[20]人给出了 5 个刻画常规模加运算线性传播性质的约束式:

$$\begin{cases} z_{n-1}=0,\\ z_{n-2}=u_{n-1}\oplus v_{n-1}\oplus w_{n-1},\\ z_j=z_{j+1}\oplus u_{j+1}\oplus v_{j+1}\oplus w_{j+1},\\ z_i\geq w_i\oplus v_i,\\ z_i\geq u_i\oplus w_i. \end{cases} \tag{3}$$

其中 $0\leq i\leq n-1, 0\leq j\leq n-3$.

对于密钥模加运算的线性传播,我们根据 Bogdanov 等[24]人提出的刻画模加运算线性逼近关系$(u,v\rightarrow w)$的规律,给出以下性质.

性质 9 对于密钥模加运算 $z=(x+K)\bmod 2^n$,输入掩码为 u,输出掩码为 w,则$(u,K\rightarrow w)$为一条有效的线性传递路径,当且仅当掩码 u 和 w 的最高非零比特位相同.

根据性质 9,为了得到有效的线性逼近,需要保证输入掩码和输出掩码的最高非零比特位相同. 为了将该线性传播性质转化为相应的 SAT 约束式,我们给出以下定理.

定理 2 对于给定的 n 比特的输入掩码 u 和输出掩码 w,我们将闭集$[0,2^n-1]$分为 $n+1$ 个区间$\{[0,1),[1,2^1),[2^1,2^2),\cdots,[2^{n-2},2^{n-1}),[2^{n-1},2^n-1]\}$,则掩码 u 和 w 的最高非零比特位相同当且仅当 u 和 w 落在某个相同的区间.

证明 对于给定的 n 比特掩码 u 和 w,将其表示为十进制的整数,则 u 和 w 可以写成:

$$u=\sum_{i=0}^{n-1}u_i2^i(u_i=0,1), w=\sum_{i=0}^{n-1}w_i2^i(w_i=0,1),$$

其中 $u_i,w_i,0\leq i\leq n$ 代表 n 比特 u 和 w 第 i 比特值.

当 u 和 w 的最高非零比特位相同时,不妨假设为第 t 位,则 u 和 w 可以表示为:

$$u=2^t+\sum_{i=0}^{t-1}u_i2^i(u_i=0,1), w=2^t+\sum_{i=0}^{t-1}w_i2^i(w_i=0,1),$$

则 u 和 w 落在相同的区间$[2^t,2^{t+1})$.

当 u 和 w 落在某个区间$[2^t,2^{t+1})$,u 和 w 的最高非零比特位 $u_t=w_t$. 证毕.

根据定理 2,为了判断 u 和 w 的最高非零比特位是否相同,需要不断的循环判断 u 和 w 是否落在区间$\{[2^{i-1},2^i),(0\leqslant i\leqslant n)\}$,这对于 SAT 求解器来说,会产生很大的复杂度,不利于实际应用. 为了避免产生递归和判断语句,我们按照比特运算的规律同样给出 5 个简单的约束式.

性质 10 设$(\boldsymbol{u},\boldsymbol{K}\rightarrow\boldsymbol{w})$为经过密钥模加的掩码传递,其中为 $\boldsymbol{u}$ 和 $\boldsymbol{w}$ 分别为输入和输出掩码向量. $\boldsymbol{u}$ 和 $\boldsymbol{w}$ 为 n 比特向量,z 为 $n+1$ 比特向量. 为了获得有效的掩码传播路径,掩码 $\boldsymbol{u}$ 和 $\boldsymbol{w}$ 需要满足以下约束式:

$$\begin{cases} z\wedge(z-1)=0,\\ z\leqslant 2^n,\\ z\geqslant 1,\\ \boldsymbol{u},\boldsymbol{w}\geqslant(z\gg 1),\\ \boldsymbol{u},\boldsymbol{w}<z. \end{cases} \tag{4}$$

3.3 零相关区分器的搜索算法

对于 ARX 密码算法,我们用约束式(3)描述$(\boldsymbol{u},\boldsymbol{v}\rightarrow\boldsymbol{w})$掩码的有效传递,用约束式(4)描述$(\boldsymbol{u},\boldsymbol{K}\rightarrow\boldsymbol{w})$掩码的有效传递. 通过将轮函数中所有线性和非线性组件的掩码传播约束式结合起来,进而得到刻画算法 r 轮线性相关度不为 0 的掩码传播的方程式,存放在名为 input. cvc 的文件. 通过穷举特定的输入输出掩码集合(Δ,Γ),利用 SAT\SMT 求解器 STP 对文件进行求解,判断每一对输入输出掩码是否构成一条有效线性路径,若求解结果返回 Invalid,则输出一条零相关路径. 算法 2 如下所示.

算法 2 零相关区分器的搜索算法

输入:输入掩码集合 Δ,输出掩码集合 Γ
输出:零相关区分器(Δ_i,τ_i)
//ARX 算法的分组长度为 n,将刻画 ARX 算法轮函数线性传播特性的约束式写入 input. cvc
1. for $\Delta_i\in\Delta$ do
2. for $\tau_i\in\Gamma$ do
3. Modify constraints on the fixed input and output masks of model. cvc
4. m = dispose(input. cvc)
5. if m = = Invalid 输出零相关区分器(Δ_i,τ_i)
6. else then continue
7. end
8. end

同样算法 2 的复杂度与输入输出掩码对的选取集合有关. 具体实现过程与 2.4 节类似,将不同非线性组件的线性传播性质封装成模块,方便调用. 通过封装 STP 软件输出接口,实现了 SAT 求解的自动化. 不仅可以判断任意给定的输入和输出掩码能否构成可能的线性路径,而且可以给出不同 ARX 算法在特殊条件下零相关区分器轮数的上界.

4 具体算法区分器的搜索

本节我们利用基于 SAT 的模型自动化搜索不同类

型 ARX 算法 Chaskey、SPECK 和 HIGHT 算法的不可能差分和零相关区分器.

4.1 Chaskey 算法区分器的搜索

Chaskey 算法是由 Mouha 等[6]人在 SAC 2014 提出的可在微型处理器上高效运行的 MAC 算法. 随后被选为 ISO/IEC JTC 1/SC 27/WG 2 的标准候选算法. 该算法基于置换设计,共 8 轮. 每一轮设计基于 ARX 结构,使用模加和循环移位运算,轮密钥的形式以异或的方式包含在结构里,轮函数具体如图 1 所示.

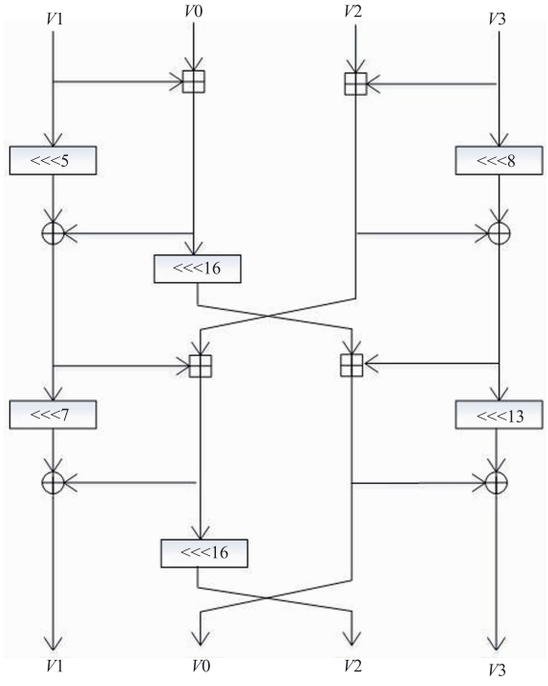

图1　Chaskey算法的轮函数

文献[20]利用 SAT\SMT 模型搜索得到了 Chaskey 算法 3 轮线性相关度最优的线性路径. 我们基于 SAT 首次给出 Chaskey 算法不可能差分区分器和零相关区分器的搜索. Chaskey 算法的轮函数包括 4 个模加,6 个循环移位运算和 4 个异或运算. 常规模加运算是 Chaskey 唯一的非线性运算,利用 2.2 节和 3.2 节的约束式分别刻画 Chaskey 算法轮函数差分和线性传播性质. 由于 Chaskey 算法的分组长度是 128 比特,无法遍历所有的输入输出差分和掩码集合,因而仅仅遍历输入和输出差分和掩码的汉明重量为 1 的情况. 利用 2.3 节和 3.3 节的搜索算法,首次找到了 Chaskey 算法 13 条 4 轮的不可能差分区分器和 1 条 4 轮的零相关区分器,如表 2 和表 3 所示. 其中 $e_i(0\leqslant i\leqslant 7)$ 代表 8 比特的字节中第 i 位为 1 其余位为 0,“ 0 ”则代表一个字节中所有比特均为 0.

表 2　Chaskey 算法不可能差分区分器

输入差分	输出差分
$(0,0,0,0,0,0,0,0,e_6,0,0,0,0,0,0,0)$	$(0,0,0,e_6,0,0,0,0,0,0,0,0,0,0,0,0)$
$(0,0,0,0,0,0,0,0,e_7,0,0,0,0,0,0,0)$	$(0,0,0,0,0,0,0,0,e_2,0,0,0,0,0,0,0)$
$(0,0,0,0,0,0,0,0,e_7,0,0,0,0,0,0,0)$	$(0,0,0,0,0,0,0,0,e_3,0,0,0,0,0,0,0)$
$(0,0,0,0,0,0,0,0,e_7,0,0,0,0,0,0,0)$	$(0,0,0,e_1,0,0,0,0,0,0,0,0,0,0,0,0)$
$(0,0,0,0,0,0,0,0,e_7,0,0,0,0,0,0,0)$	$(0,0,0,e_2,0,0,0,0,0,0,0,0,0,0,0,0)$
$(0,0,0,0,0,0,0,0,e_7,0,0,0,0,0,0,0)$	$(0,0,0,e_3,0,0,0,0,0,0,0,0,0,0,0,0)$
$(0,0,0,0,0,0,0,0,e_7,0,0,0,0,0,0,0)$	$(0,0,0,e_4,0,0,0,0,0,0,0,0,0,0,0,0)$
$(0,0,0,0,0,0,0,0,e_7,0,0,0,0,0,0,0)$	$(0,0,0,e_5,0,0,0,0,0,0,0,0,0,0,0,0)$
$(0,0,0,0,0,0,0,0,e_7,0,0,0,0,0,0,0)$	$(0,0,0,e_6,0,0,0,0,0,0,0,0,0,0,0,0)$
$(0,0,0,0,0,0,0,0,e_7,0,0,0,0,0,0,0)$	$(0,0,e_1,0,0,0,0,0,0,0,0,0,0,0,0,0)$
$(0,0,0,0,0,0,0,0,e_7,0,0,0,0,0,0,0)$	$(0,0,e_2,0,0,0,0,0,0,0,0,0,0,0,0,0)$
$(0,0,0,0,0,0,0,0,e_7,0,0,0,0,0,0,0)$	$(0,0,e_3,0,0,0,0,0,0,0,0,0,0,0,0,0)$
$(0,0,0,0,e_7,0,0,0,0,0,0,0,0,0,0,0)$	$(0,0,0,e_3,0,0,0,0,0,0,0,0,0,0,0,0)$

表 3　Chaskey 算法零相关区分器

输入掩码	输出掩码
$(0,0,0,0,0,0,0,0,0,0,0,0,e_1,0,0)$	$(0,0,0,0,0,0,0,0,e_0,0,0,0,0,0,0)$

我们总结对比了 Chaskey 算法已有的自动化搜索结果,如表 4 所示。

表 4　Chaskey 算法自动化搜索结果对比

区分器类型	轮数	文献
线性区分器	3	文献[20]
不可能差分区分器	4	本文
零相关区分器	4	本文

4.2 SPECK 算法区分器的搜索

SPECK 算法[1]是由美国安全局(NSA)在 2013 年提出的,可在资源受限的软件环境下高效运行的 ARX 算法. SPECK 是类 Feistel 结构,轮函数由常规模加,异或和循环移位运算组成,每一轮中左右两个分支都做了变化. 对于 SPECK2n 算法,轮函数如图 2 所示. $X_{r-1,L}$、$X_{r-1,R}$分别对应于第 r 轮加密输入分组的左半部分和右半部分,k_r是作用于第 r 轮加密的子密钥,长度均为 n 比特. 那么第 r 轮加密输出 X_{rL},$X_{r,R}$通过以下方式得到:

$$X_{r,L}=((X_{r-1,L}>>>r_1)\boxplus X_{r-1,R})\oplus k_r,$$

$$X_{r,R}=(X_{r-1,R}<<<r_2)\oplus X_{r,L},$$

其中左右循环移位常数 r_1、r_2规定如下:分组长度为 32 的 SPECK32 算法 r_1=7,r_2=2,其他分组长度的算法 r_1=8,r_2=3.

对于 SPECK 算法不可能差分区分器,Lee 等[25]人利用 MILP 给出了 SPECK64 算法一些 6 轮的不可能差分区分器. 我们利用 2.2 节的约束式刻画 SPECK 算法

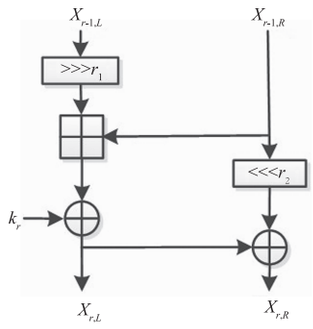

图2 SPECK算法的轮函数

轮函数的差分传播性质,分别遍历汉明重量为 1 的输入差分和输出差分集合,发现 SPECK32 和 SPECK48 算法最长的不可能差分区分器为 6 轮,攻击者如果想搜索得到更长轮数的不可能差分区分器,需要考虑其他的输入输出差分的类型. 我们分别在 13.7 秒和 32.9 秒内搜索到 SPECK32 的 3 条和 SPECK48 的 20 条不可能差分区分器,分别如表 5 和表 6 所示.

表 5 SPECK32 算法不可能差分区分器

$(0,e_6,0,0)\to(0,e_0,0,0)$	$(0,e_6,0,0)\to(0,e_1,0,0)$
$(0,e_6,0,0)\to(e_7,0,0,0)$	

表 6 SPECK48 算法不可能差分区分器

$(0,0,0,e_2,0,0)\nrightarrow(0,0,e_2,0,0,0)$	$(0,0,0,e_3,0,0)\nrightarrow(0,0,e_1,0,0,0)$
$(0,0,0,e_3,0,0)\nrightarrow(0,0,e_2,0,0,0)$	$(0,0,0,e_4,0,0)\nrightarrow(0,0,e_2,0,0,0)$
$(0,0,0,e_5,0,0)\nrightarrow(0,0,e_2,0,0,0)$	$(0,0,0,e_6,0,0)\nrightarrow(0,0,e_2,0,0,0)$
$(0,0,0,e_7,0,0)\nrightarrow(0,0,e_2,0,0,0)$	$(0,0,e_3,0,0,0)\nrightarrow(0,0,e_2,0,0,0)$
$(0,0,e_4,0,0,0)\nrightarrow(0,0,e_2,0,0,0)$	$(0,0,e_5,0,0,0)\nrightarrow(0,0,e_2,0,0,0)$
$(0,0,e_6,0,0,0)\nrightarrow(0,0,e_2,0,0,0)$	$(0,0,e_7,0,0,0)\nrightarrow(0,0,0,e_5,0,0)$
$(0,0,e_7,0,0,0)\nrightarrow(0,0,0,e_7,0,0)$	$(0,0,e_7,0,0,0)\nrightarrow(0,0,e_0,0,0,0)$
$(0,0,e_7,0,0,0)\nrightarrow(0,0,e_1,0,0,0)$	$(0,0,e_7,0,0,0)\nrightarrow(0,0,e_2,0,0,0)$
$(0,0,e_7,0,0,0)\nrightarrow(0,e_1,0,0,0,0)$	$(0,0,e_7,0,0,0)\nrightarrow(e_5,0,0,0,0,0)$
$(0,0,e_7,0,0,0)\nrightarrow(e_6,0,0,0,0,0)$	$(0,0,e_7,0,0,0)\nrightarrow(e_7,0,0,0,0,0)$

对于零相关区分器的搜索,利用 3.3 节零相关区分器的搜索算法,分别在 288 秒和 1357 秒内首次得到了 SPECK32 算法 10 条 6 轮零相关区分器和 SPECK48 算法 15 条 6 轮零相关区分器,分别如表 7 和表 8 所示.

表 7 SPECK32 算法零相关区分器

$(0,0,0,e_5)\nrightarrow(0,e_2,0,e_2)$	$(0,0,0,e_5)\nrightarrow(0,e_3,0,e_3)$
$(0,0,0,e_5)\nrightarrow(0,e_4,0,e_4)$	$(0,0,0,e_5)\nrightarrow(0,e_5,0,e_5)$
$(0,0,0,e_5)\nrightarrow(0,e_6,0,e_6)$	$(0,0,0,e_5)\nrightarrow(0,e_7,0,e_7)$
$(0,0,0,e_6)\nrightarrow(0,e_2,0,e_2)$	$(0,0,0,e_6)\nrightarrow(0,e_4,0,e_4)$
$(0,0,0,e_7)\nrightarrow(0,e_2,0,e_2)$	$(0,0,0,e_7)\nrightarrow(0,e_3,0,e_3)$

表 8 SPECK48 算法零相关区分器

$(0,0,0,0,0,e_5)\nrightarrow(0,0,e_3,0,0,e_3)$	$(0,0,0,0,0,e_5)\nrightarrow(0,0,e_4,0,0,e_4)$
$(0,0,0,0,0,e_5)\nrightarrow(0,0,e_6,0,0,e_6)$	$(0,0,0,0,0,e_5)\nrightarrow(0,0,e_7,0,0,e_7)$
$(0,0,0,0,0,e_5)\nrightarrow(0,e_0,0,0,e_0,0)$	$(0,0,0,0,0,e_5)\nrightarrow(0,e_2,0,0,e_2,0)$
$(0,0,0,0,0,e_6)\nrightarrow(0,0,e_3,0,0,e_3)$	$(0,0,0,0,0,e_6)\nrightarrow(0,0,e_4,0,0,e_4)$
$(0,0,0,0,0,e_6)\nrightarrow(0,0,e_5,0,0,e_5)$	$(0,0,0,0,0,e_6)\nrightarrow(0,0,e_6,0,0,e_6)$
$(0,0,0,0,0,e_6)\nrightarrow(0,0,e_7,0,0,e_7)$	$(0,0,0,0,0,e_6)\nrightarrow(0,e_0,0,0,e_0,0)$
$(0,0,0,0,0,e_6)\nrightarrow(0,e_1,0,0,e_1,0)$	$(0,0,0,0,0,e_7)\nrightarrow(0,0,e_7,0,0,e_7)$
$(0,0,0,0,0,e_7)\nrightarrow(0,e_0,0,0,e_0,0)$	

我们可以利用表 5、6 列出的多条不可能差分区分器,改进 SPECK 算法不可能差分的分析结果;也可以利用表 7、8 列出的多条区分器,结合多维零相关分析模型,降低 SPECK 算法的分析的复杂度.

4.3 HIGHT 算法区分器的搜索

HIGHT 由 Hong 等[5]人在 CHES 2006 上提出,并被采纳为 ISO/IECI 8033-3 的标准密码算法. 该算法采用 32 轮加密,每轮包含两个线性函数($F_0:\{0,1\}^9\to\{0,1\}^8$,$F_1:\{0,1\}^9\to\{0,1\}^8$),一个内部置换函数(IP:$\{0,1\}^{64}\to\{0,1\}^{64}$),四个模加(⊞)运算,模加运算有常规模加,也有密钥模加. 算法的分组长度为 64 比特,密钥长度为 128 比特. HIGHT 算法的轮函数如图 3 所示.

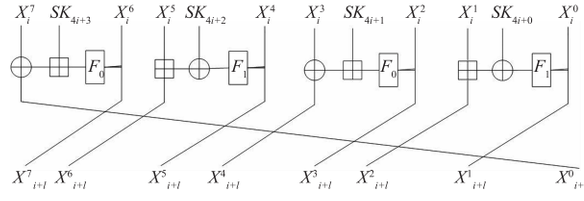

图3 HIGHT算法的轮函数

轮函数中的 F_0 和 F_1 定义如下:

$$F_0(x)=(x\lll 1)\oplus(x\lll 2)\oplus(x\lll 7),$$
$$F_1(x)=(x\lll 3)\oplus(x\lll 4)\oplus(x\lll 6).$$

利用自动化搜索工具之前,HIGHT 算法最长的不可能差分和零相关区分器均为 16 轮,Cui 等[16]人基于 MILP 模型搜索了 HIGHT 算法 17 轮的不可能差分和零相关区分器. 我们考虑了 HIGHT 算法常规模加和密钥模加运算的性质,利用 2.2 节和 3.2 节的约束式分别刻画 HIGHT 算法轮函数的差分和线性传播性质,建立了基于 SAT 的搜索模型. 我们分别遍历输入和输出差分和掩码的汉明重量为 1 的情况,发现 HIGHT 算法不可能差分和零相关区分器轮数的上界为 17 轮. 在 2^{12} 的搜索空间里,得到了 HIGHT 算法 4 条 17 轮的不可能差分区分器和零相关区分器,如表 9 和表 10 所示.

Cui 等人建立的 MILP 模型,在服务器(Intel(R) Xeon(R) CPU E5-2620 (2.00GHz,47GB RAM,Ubuntu 14.04.3))上同时使用 12 个线程并行运行,搜索 4 条不

可能差分区分器花费 4445 秒,约 74 分钟;搜索 4 条零相关区分器花费 4786 秒,约 80 分钟. 我们建立的基于 SAT 的模型,在个人笔记本上(Intel Core i7-6700 (3.4GHz),7.8GBRAM,Ubuntu 14.04.3 LTS)单线程运行,搜索 HIGHT 算法不可能差分区分器的时间只需要 264 秒,约 4.4 分钟,搜索零相关区分器的时间需要 856.6 秒,约 14 分钟. 表 11 列出了 HIGHT 算法区分器搜索结果的对比.

表 9　HIGHT 算法不可能差分区分器

$$(0,0,0,0,0,0,e_7,0)\nrightarrow(0,0,0,0,0,e_7,0,0)$$
$$(0,0,0,0,e_7,0,0,0)\nrightarrow(0,0,0,e_7,0,0,0,0)$$
$$(0,0,e_7,0,0,0,0,0)\nrightarrow(0,e_7,0,0,0,0,0,0)$$
$$(e_7,0,0,0,0,0,0,0)\nrightarrow(0,0,0,0,0,0,0,e_7)$$

表 10　HIGHT 算法零相关区分器

$$(0,0,0,0,0,0,0,e_0)\nrightarrow(0,0,0,0,0,0,e_0,0)$$
$$(0,0,0,0,0,e_0,0,0)\nrightarrow(0,0,0,0,e_0,0,0,0)$$
$$(0,0,0,e_0,0,0,0,0)\nrightarrow(0,0,e_0,0,0,0,0,0)$$
$$(0,e_0,0,0,0,0,0,0)\nrightarrow(e_0,0,0,0,0,0,0,0)$$

表 11　HIGHT 算法区分器搜索结果对比

区分器类型	轮数	时间(s)	出处
不可能差分区分器	16	—	文献[26]
不可能差分区分器	17	4445	文献[16]
不可能差分区分器	17	264	本文
零相关区分器	16	—	文献[27]
零相关区分器	17	4786	文献[16]
零相关区分器	17	856.6	本文

5　总结

本文主要针对 ARX 密码算法,提出了基于 SAT 的不可能差分和零相关区分器的搜索模型. 通过分析 ARX 算法组件的性质,特别是常规模加和密钥模加这两种非线性运算差分和线性传播的特性,给出了高效简单的 SAT 约束式. 在此基础上,建立轮函数差分和线性传播的约束式,进而利用 SAT\SMT 求解器 STP 进行区分器的搜索. 我们首次给出了 Chaskey 算法 13 条 4 轮不可能差分和 1 条 4 轮零相关区分器;首次给出了 SPECK32 算法 10 条 6 轮零相关区分器和 SPECK48 算法 15 条 6 轮零相关区分器;给出了 HIGHT 算法 17 轮的不可能差分和零相关区分器,搜索时间明显优于现有结果. 此外,除了用于自动化搜索区分器,我们通过重新封装求解器 STP 的输出接口,建立了自动化的 SAT\SMT 分析模型,能够判断算法在任意轮数、任意输入输出差分和掩码条件下,是否构成不可能差分和零相关路径,进而可以给出 ARX 算法不可能差分和零相关区分器轮数的上界. 下一步将研究如何利用搜索得到的区分器,改进具体算法不可能差分和零相关分析的结果.

参考文献

[1] R Beaulieu, D Shors, J Smith, et al. The SIMON and SPECK families of lightweight block ciphers[DB/OL]. http://eprint.iacr.org/2013/404/2013-06-20.

[2] D Hong, J K Lee, D C Kim, et al. LEA: A 128-bit block cipher for fast encryption on common processors[A]. International Workshop on Information Security Applications [C]. Heidelberg: Springer, 2013. 3 – 27.

[3] D J Wheeler, R M Needham. TEA, a tiny encryption algorithm[A]. FSE 1995[C]. Heidelberg: Springer, LNCS, 1995. 363 – 366.

[4] R M Needham, D J Wheeler l. TEA extensions[R]. Cambridge: University of Cambridge, 1997.

[5] D Hong, J Sung, S Hong, J Lim, et al. HIGHT: A new block cipher suitable for low-resource device[A]. Cryptographic Hardware and Embedded Systems-CHES 2006 [C]. Heidelberg: Springer, 2006. 46 – 59.

[6] N Mouha, B Mennink, A Van Herrewege, D Watanabe, B Preneel, I Ver-bauwhede. Chaskey: An efficient MAC algorithm for 32-bit microcontrollers[A]. Selected Areas in Cryptography-SAC 2014[C]. Heidelberg: Springer, 2014. 306 – 323.

[7] D Dinu, L Perrin, A Udovenko, V Velichkov, et al. Design strategies for ARX with provable bounds: Sparx and LAX [A]. ASIACRYPT 2016[C]. Heidelberg: Springer, 2016. 484 – 513.

[8] L R Knudsen. DEAL a 128-bit block cipher[R]. Bergen: Department of Informatics, University of Bergen, 1998.

[9] E Biham, A Biryukov, A Shamir. Cryptanalysis of Skipjack reduced to 31 rounds using impossibledifferentials[A]. EUROCRYPT 1999[C]. Heidelberg: Springer, 1999. 291 – 311.

[10] C W Phan. Impossible differential cryptanalysis of 7-round Advanced Encryption Standard (AES)[J]. Information Processing Letters, 2004, 91(1), 33 – 38.

[11] D Yang, W Qi, H Chen. Impossible differential attack on QARMA family of block ciphers[DB/OL]. http://eprint.iacr.org/2018/334, 2018-04-11.

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

[13] J Kim, S Hong, J Lim. Impossible differential cryptanalysis using matrix method[J]. Discrete Mathematics, 2010, 310(5): 988 – 1002.

[14] Y Luo, X Lai, Z Wu, G Gong. A unified method for finding impossible differentials of block cipher structures[J]. Information Sciences, 2014, 263(1): 211 – 220.
[15] S Wu, M Wang. Automatic search of truncated impossible differentials for word-oriented block ciphers[A]. INDOCRYPT 2012[C]. Heidelberg: Springer, 2012. 283 – 302.
[16] T Cui, K Jia, K Fu, et al. New automatic search tool for impossible differentials and zero-correlation linearapproximations[DB/OL]. http://eprint. iacr. org/2016/689, 2018-11-21.
[17] Y Sasaki, Y Todo. New impossible differential search tool from design and cryptanalysis aspects revealing structural properties of several ciphers[A]. EUROCRYPT 2017[C]. Heidelberg: Springer, 2017, 185 – 215.
[18] N Mouha, B Preneel. Towards finding optimal differential characteristics for ARX: Application to Salsa20[DB/OL]. http://eprint. iacr. org/2013/ 328, 2013-11-23.
[19] L Song, Z Huang, Q Yang. Automatic differential analysis of ARX block ciphers with application to SPECK and LEA[A]. ACISP 2016[C]. Heidelberg: Springer, 2016. 379 – 394.
[20] Y Liu, Q Wang, V Rijmen. Automatic search of linear trails in ARX with applications to SPECK and Chaskey[A]. ACNS 2016[C]. Heidelberg: Springer, 2016. 485 – 499.
[21] S Kölbl, G. Leander, T Tiessen. Observations on the SIMON block cipher family[A]. CRYPTO 2015[C]. Heidelberg: Springer, 2015. 161 – 185.
[22] H Lipmaa, S Moriai. Efficient algorithms for computing differential properties of addition[A]. FSE 2002[C]. Heidelberg: Springer, 2002. 336 – 350.
[23] E Schulte-Geers. On CCZ-equivalence of addition mod 2^n[J]. Designs, Codes and Cryptography, 2013, 66(1 – 3): 111 – 127.
[24] A Bogdanov, M Wang. Zero correlation linear cryptanalysis with reduced data complexity[A]. FSE 2012[C]. Heidelberg: Springer, 2012. 29 – 48.
[25] H Lee, H Kang, D Hong et al. New impossible differential characteristic of SPECK64 using MILP[DB/OL]. http://eprint. iacr. org/2016/1137, 2016-12-21.
[26] J Lu. Cryptanalysis of reduced versions of the HIGHT block cipher from CHES 2006[A]. ICISC 2007[C]. Heidelberg: Springer, 2007. 11 – 26.
[27] L Wen, M Wang, A Bogdanov, H Chen. Multidimensional zero-correlation attacks on lightweight block cipher HIGHT: Improved cryptanalysis of an ISO standard[J]. Information Processing Letters, 2014, 114(6): 322 – 330.

作者简介

任炯炯 男,1994 年生于甘肃天水,信息工程大学博士研究生,研究方向为对称密码设计与分析.
E-mail: jiongjiong_fun@163. com

张仕伟 男,1988 年生于河北唐山,信息工程大学硕士生,究方向为对称密码设计与分析.

李曼曼 女,1986 年出生于河南开封,信息工程大学讲师,研究方向为对称密码设计与分析.

陈少真 女,1967 年生于江苏无锡,信息工程大学教授, 研究方向为密码学与信息安全.