Saturnin 算法的不可能差分分析

蒋梓龙，金晨辉
（信息工程大学密码工程学院，河南 郑州 450001）

摘 要：轻量级分组密码算法 Saturnin 是类 AES 算法，在资源受限的环境下，仍具有良好的安全性。对 Saturnin 算法进行了不可能差分分析。首先，基于 Saturnin 算法的结构特性，提出并证明了 Saturnin 算法 3.5 轮不可能差分区分器的充分条件，利用此充分条件可以快速构造 $2^{70.1}$ 个截断式不可能差分区分器。其次，从构造的 $2^{70.1}$ 个区分器中，有针对性地挑选了 64 个区分器并分成了四类。将这四类区分器向前扩展 2 轮可得四条攻击路径。这四条攻击路径不仅具有相同的明文结构，而且具有大量的公共密钥比特，利用这 2 个特性，可以改善攻击方案的复杂度。结合明文早夭等分析技术，提出 Saturnin 算法的 5.5 轮不可能差分攻击方案，其数据、存储和时间复杂度分别为 $2^{176.88}$ 个选择明文、$2^{143.88}$ 算法规模和 $2^{176.91}$ 次 5.5 轮加密，这是目前可见的对 Saturnin 算法的一种不可能差分攻击方案。

关键词：轻量级分组密码；不可能差分；SPN 结构；NIST 竞赛
中图分类号：TN918.1
文献标志码：A
DOI: 10.11959/j.issn.1000−436x.2022045

Impossible differential cryptanalysis of Saturnin algorithm

JIANG Zilong, JIN Chenhui
Department of Cryptogram Engineering, Information Engineering University, Zhengzhou 450001, China

Abstract: A lightweight block cipher, Saturnin, is an AES-like algorithm. In a resource-constrained environment, Saturnin can also provide high security. The impossible differential analysis on Saturnin was proposed. First, based on the structure of Saturnin, the sufficient condition of 3.5-round impossible differential distinguisher of Saturnin was presented and proved, and $2^{70.1}$ truncated impossible differential distinguishers could be quickly constructed by utilizing the sufficient condition. Then, from the constructed $2^{70.1}$ distinguishers, the 64 distinguishers could be picked out pointedly and classified into four types. Four attack trails could be obtained by appending two rounds before the four types of distinguishers. These four attack trails had the same plaintext structure and a number of common subkey bits. With the help of these two properties, the complexity of the attack scheme could be reduced. Combined with the analysis technologies such as early abort, present the 5.5-round impossible differential attack scheme with $2^{176.88}$ chosen plaintexts, $2^{143.88}$ 256-bit blocks, and $2^{176.91}$ 5.5-round encryption. As so far, this is the known attack scheme for Saturnin against impossible differential attack.

Keywords: lightweight block cipher, impossible differential, SPN structure, NIST competition

0 引言

在 NIST 轻量级密码标准竞赛中，Saturnin 算法[1]是进入第二轮的候选算法之一。算法设计者声称该算法不仅适用于小型电子设备，并且可以作为哈希和认证加密的基础算法。该算法在保持高效轻巧的同时具有极高的安全性，甚至可以抵抗量子计算分析。作为一个轻量级可抗量子计算分析的新算法，Saturnin 算法的安全性更值得深入研究。

不可能差分攻击[2-3]是差分攻击的一种特殊变体。

收稿日期：2021−11−29；**修回日期**：2022−02−07
基金项目：国家自然科学基金资助项目（No.61772547, No.61902428, No.61802438）
Foundation Item: The National Natural Science Foundation of China (No.61772547, No.61902428, No.61802438)

与传统的差分攻击利用高概率差分对应完全相反，不可能差分攻击利用不可能出现的差分对应，即差分转移概率为 0 的差分对应，来刻画算法的不完全随机性，并利用此信息泄露做区分攻击或者密钥恢复。关于不可能差分区分器的构造问题一直都是热点课题，在 1999 年的 FSE 会议上，Biham 等[2]详细地阐述了运用“中间相遇找矛盾”的方法构造不可能差分。后来随着自动搜索技术的出现，不可能差分区分器的构造变得更加高效精细，例如，Kim 等[3]提出的 U 方法； Luo 等[4]提出 U 方法的改进版本，称之为 UID 方法； Wu 等[5]提出的线性化方法； Mouha 等[6]将混合整数规划（MILP, mixed integer linear programming）搜索用于差分分析。之后这类工具被广泛用于各类分析方法[7-9]，不可能差分区分器搜索与构造也是研究的热点，Cui 等[10]首次利用 MILP 求解不可能差分区分器；Sasaki 等[11]系统详尽地阐述了如何利用 MILP 方法搜索不可能差分；Hu 等[12]利用传播状态的特性，进一步改进了分析结果；张仕伟等[13]利用 AND 组件特性，基于 SAT 求解器搜索到了更多的不可能差分区分器；Zhang 等[14]提出了“模式运算”方法，实现了 ARX 密码算法不可能差分区分器的自动化搜索。针对轻量级分组密码算法，也有不少关于不可能差分的分析结果，如武小年等[15]给出了 GRANULE 算法的 7 轮不可能差分区分器和 MANTRA 的 9 轮不可能差分区分器；王旭姿等[16]在相关密钥的条件下，首次给出了 SIMON 算法的 13 轮不可能差分区分器。

Saturnin 是类 AES 密码算法，分组长度为 256 bit，由 64 个 4 bit 元胞构成。设计者声称针对 AES 算法[17]的分析方法都可以用来分析 Saturnin 算法，且 Saturnin 算法的安全性基础也受益于 AES 的安全性结果。在设计报告[1]的第 6 节中，基于在单密钥条件下的 AES 安全性分析结果，设计者给出了 Saturnin 算法抵抗经典攻击方法的安全性分析，包括差分攻击、线性攻击、代数攻击、不可能差分、中间相遇等分析方法。在攻击方案的安全性评估上，设计报告只给出了 7.5 轮的中间相遇攻击方案；之后 Hou 等[18]提出了用 Yoyo 方法分析 Saturnin 算法的攻击方案，但是 Yoyo 需要自适应的明密文选择，分析方法所需要的条件较强。针对不可能差分，设计报告[1]中只给出了两条 3.5 轮的不可能差分区分器，设计者声称“我们提出的 2 个不可能差分区分器得到的攻击路径都需要攻击全部密钥比特。我们认为可能可以构造出 4 轮或 4.5 轮的攻击方案，但是至今我们也没有达成这个目标。”截至目前，只有设计报告[1]中给出了 Saturnin 算法抵抗不可能差分的安全性分析，且设计者只给出了两条 3.5 轮的不可能差分区分器，并没有给出完整的不可能差分攻击方案。为弥补这个缺项，本文对 Saturnin 算法进行不可能差分分析，并提出了具体的 5.5 轮不可能差分攻击方案。首先，基于 Saturnin 算法的结构特性，本文提出并证明了 3.5 轮不可能差分区分器的充分条件，即输入差分和输出差分非 0 面（页）的个数和小于或等于 4。利用此充分条件，可以快速构造 $2^{70.1}$ 个不可能差分区分器，可以为攻击方案的设计提供更多的选择。之后，从构造的 $2^{70.1}$ 个区分器中，有针对性地挑选了 64 个区分器并分成了四类。将这四类区分器向前扩展 2 轮各得一条攻击路径，这四条攻击路径不仅具有相同的明密文结构，且具有大量的公共密钥字节，利用这 2 个特性可以改善攻击方案的整体复杂度。基于上述的四条攻击路径，结合一系列分析技术，本文提出了对 Saturnin 算法的 5.5 轮不可能差分攻击方案。作为比较，设计者只构造了两条 3.5 轮的不可能差分区分器，认为可能可以设计出 4 轮或 4.5 轮的不可能差分攻击方案，但他们也没有设计出相应的攻击方案。表 1 总结了经典分析方法对 Saturnin 算法的攻击结果。

表 1 经典分析方法对 Saturnin 算法的攻击结果

分析方法	轮数	时间复杂度	数据复杂度	存储复杂度	文献
中间相遇	7.5	2^{244}	2^{244}	2^{244}	文献[1]
Yoyo	5	2^{46}	$2^{39.1}$	—	文献[8]
不可能差分	5.5	$2^{176.91}$	$2^{176.88}$	$2^{143.88}$	本文

1 预备知识

1.1 符号表示

P：明文。

$\oplus$：逐位异或。

Δx：x 的差分值。

$x_{i,(p,\cdots,r)}^{f}$：第 i 轮 f（f=I 表示输入，f=S 表示元胞替换，f = SR 表示移位变换，f=MC 表示列混合变换，f=AKT 表示轮密钥常数加）后的第$(p,\cdots,r)$元胞值。

$\Delta a \nrightarrow_{i-\text{round}} \Delta b$：差分 Δa 经 i 轮加密后不能得到差分 Δb。

1.2 Saturnin 密码简介

Saturnin 算法是一个 SPN 结构密码算法。算法的主密钥和中间状态都为 256 bit，可以将主密钥和操作中间状态看作 4×4×4 个元胞的立方体，其中

元胞代表一个 4 bit 值。$4\times4\times4$ 个元胞的三维立方状态如图 1 所示，立方状态中的元胞可以由三维坐标 (x,y,z) 表示位置，其中 $x,y,z\in\{0,1,2,3\}$，对应的元胞号为 $(y+4x+16z)$，元胞号为 0~63，0 为最低有效位。举例而言，第 33 号元胞的坐标为 $(0,1,2)$。

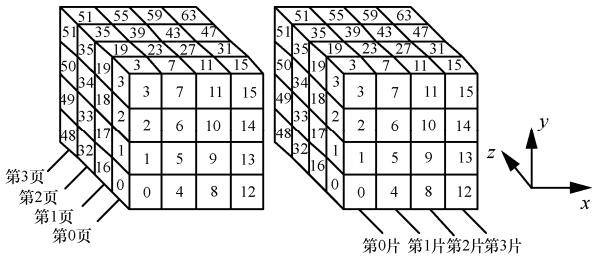


图 1　4×4×4 个元胞算法的三维立方状态

根据 Saturnin 算法的三维立方状态，本文定义了以下 5 种立方状态中的子集。

1) 页：在立方状态中，当 z 坐标值为固定值，x,y 坐标值遍历，对应得到的 16 个元胞集合为一页，记作第 z 页，符号表示为 slice(z)。

2) 片：在立方状态中，当 x 坐标值为固定值，y,z 坐标值遍历，对应得到的 16 个元胞集合为一片，记作第 x 片，符号表示为 sheet(x)。

3) 列：在立方状态中，当 x,z 坐标值都为固定值，y 坐标值遍历，对应得到的 4 个元胞集合为一列，记作第 $x+4z$ 列，符号表示为 column(x+4z)。

4) 页行：在立方状态中，当 y,z 坐标值都为固定值，x 坐标值遍历，对应得到的 4 个元胞集合为一个页行，记作第 $y+4z$ 页行，符号表示为 xrow(y+4z)。

5) 片行：在立方状态中，当 x,y 坐标值都为固定值，z 坐标值遍历，对应得到的 4 个元胞集合为一个片行，记作第 $y+4x$ 片行，符号表示为 zrow(y+4x)。

Saturnin 算法的设计者提出了合并轮[1]的概念。为方便阐述，本文中的一轮均指一个合并轮，加密算法的轮数从 1 开始计数。用户可以输入 2 个参数，分别为加密轮数 $R\in\{10,\cdots,31\}$ 和 4 bit 的域分隔符 D。算法默认加密 10 轮，用户也可以在 $\{10,\cdots,31\}$ 中任选加密轮数。算法轮函数共包括 6 种变换，分别为元胞替换 S、页行移位变换 SR_s、片行移位变换 SR_t、列混合变换 MC、轮子密钥加 AK 和轮常数加 AT。这 6 种变换简述如下。

1) 元胞替换 S。对立方状态中的全部 64 个元胞做查 S 盒的非线性变换。此变换由两类 4 bit 的 S 盒构成，分别记作 S_0、S_1，对元胞号为偶数的元胞查 S_0 盒，对元胞号为奇数的元胞查 S_1 盒。S_0 和 S_1 如表 2 所示。

表 2　Saturnin 算法的两类 4 bit 的 S 盒

x	$S_0(x)$	$S_1(x)$	x	$S_0(x)$	$S_1(x)$
0	0	0	8	9	6
1	6	9	9	8	4
2	14	13	10	12	5
3	1	2	11	5	3
4	15	15	12	2	8
5	4	1	13	10	12
6	7	11	14	3	10
7	13	7	15	11	14

2) 页行移位 SR_s。根据 y 坐标值进行页行循环移位操作，即在第 y 行按 x 坐标方向循环移位 y 元胞（$y=0,1,2,3$），SR_s^{-1} 为 SR_s 的逆变换。以第 0 页为例，具体变换如图 2 所示。

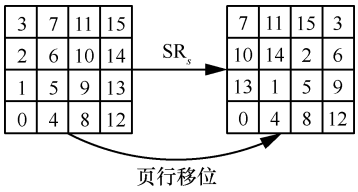


图 2　Saturnin 算法的页行移位样例

3) 片行移位 SR_t。根据 y 坐标值进行片行循环移位操作，即在第 y 行按 z 坐标方向循环移位 y 元胞（$y=0,1,2,3$），SR_t^{-1} 为 SR_t 的逆变换。以第 0 片为例，具体变换如图 3 所示。

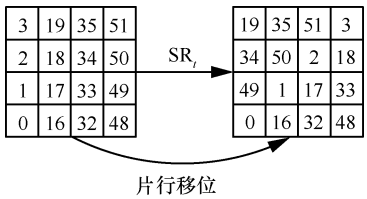


图 3　Saturnin 算法的片行移位样例

4) 列混合变换 MC。对立方状态中的全部十六列做左乘变换 $\boldsymbol{M}$。

$$\boldsymbol{M}\begin{pmatrix}a\\b\\c\\d\end{pmatrix}=\begin{pmatrix}\alpha^2(a)\oplus\alpha^2(b)\oplus\alpha(b)\oplus c\oplus d\\ a\oplus\alpha(b)\oplus b\oplus\alpha^2(c)\oplus c\oplus\alpha^2(d)\oplus\alpha(d)\oplus d\\ a\oplus b\oplus\alpha^2(c)\oplus\alpha^2(d)\oplus\alpha(d)\\ \alpha^2(a)\oplus\alpha(a)\oplus\alpha^2(b)\oplus\alpha(b)\oplus b\oplus c\oplus\alpha(d)\oplus d\end{pmatrix}$$

其中，α 作用于 4 bit 元胞。

$$\alpha\begin{pmatrix}x_0\\x_1\\x_2\\x_3\end{pmatrix}=\begin{pmatrix}0&1&0&0\\0&0&1&0\\0&0&0&1\\1&1&0&0\end{pmatrix}\begin{pmatrix}x_0\\x_1\\x_2\\x_3\end{pmatrix}$$

5) 轮子密钥加 AK。在每轮输出前进行轮子密钥加变换。记主密钥为 64 元胞 $K_0=(k_0,\cdots,k_{63})$，对第 i 轮加密，若 $i \bmod 2=1$，则用主密钥进行轮子密钥加变换；若 $i \bmod 2=0$，则将 K_0 循环左移 20 元胞，即用密钥 $K_1=(k_{20},\cdots,k_{63},k_0,\cdots,k_{19})$ 进行轮子密钥加变换。算法开始时以 K_0 为白化密钥，先对明文进行一次密钥加变换。

6) 轮常数加 AT。根据 2 个输入参数（加密轮数 R 和域分隔符 D），生成 2 个 16 bit 字 RC_0 和 RC_1，再由 RC_0 和 RC_1 生成轮常数 T_i，在第 i 轮输出前进行轮常数加变换。

Saturnin 算法的轮函数与轮号 i 有关。对第 i 轮加密，当 $i \bmod 2\equiv 1$ 时，轮函数为

$$E_i(m)=K_1\oplus T_i\oplus \mathrm{SR}_s^{-1}\circ \mathrm{MC}\circ \mathrm{SR}_s\circ S\circ \mathrm{MC}\circ S(m)$$

当 $i \bmod 2\equiv 0$ 时，轮函数为

$$E_i(m)=K_0\oplus T_i\oplus \mathrm{SR}_t^{-1}\circ \mathrm{MC}\circ \mathrm{SR}_t\circ S\circ \mathrm{MC}\circ S(m)$$

注意，白化密钥采用主密钥 K_0，需要先用 K_0 与明文 P 做密钥加变换，即第一轮的输入为 $m_0=P\oplus K_0$，再调用轮函数进行加密。

关于 Saturnin 算法的完整详细内容，可以通过查阅文献[1]做进一步了解。

2 Saturnin 密码的不可能差分区分器

为方便对差分路径进行阐述，2.1 节将轮函数进行简化，并给出状态、页、片和列的相关重量定义；2.2 节阐述 Saturnin 算法 3.5 轮不可能差分区分器的充分条件，并利用此充分条件构造了 $2^{70.1}$ 个 3.5 轮不可能差分区分器。

2.1 简化轮和立方状态重量

在单密钥且单常数的条件下，异或加不影响差分值。在忽略轮子密钥加和轮常数加变换的情况下，简化的轮函数也与轮数 $i(i>0)$ 有关。当 $i \bmod 2\equiv 1$ 时，轮函数为

$$F_i(m)=\mathrm{SR}_s^{-1}\circ \mathrm{MC}\circ \mathrm{SR}_s\circ S\circ \mathrm{MC}\circ S(m) \quad (1)$$

当 $i \bmod 2\equiv 0$ 时，轮函数为

$$F_i(m)=\mathrm{SR}_t^{-1}\circ \mathrm{MC}\circ \mathrm{SR}_t\circ S\circ \mathrm{MC}\circ S(m) \quad (2)$$

当轮号为奇数时，记简化轮函数为 $F_1(m)$；当轮号为偶数时，记简化轮函数为 $F_0(m)$。记 $F^{(\mathrm{st},r)}(m)$ 为从第 st 轮开始加密 r 次简化轮，$F^{-(\mathrm{st},r)}(m)$ 为从第 st 轮开始脱密 r 次简化轮，其中 $\mathrm{st},r\in\{1,\cdots,31\}$。例如，算法默认的 10 次简化轮加密记为 $F^{(1,10)}(m)=\left(F_0\circ F_1(m)\right)^5$。

由 1.2 节可知，Saturnin 算法的立方状态 m 有 4 个页、4 个片和 16 个列，每个页、每个片分别有 16 元胞，每个列有 4 元胞。对立方状态、页、片和列而言，元胞重量指包含的非 0 元胞个数，元胞重量的符号定义为 W_{nibble}。例如，$W_{\mathrm{nibble}}(m_{\mathrm{slice}(0)})=3$ 表示在立方状态 m 的第 0 页中有 3 个非 0 元胞。

若页、片、列的全部元胞值都为 0，则元胞重量为 0；反之元胞重量非 0。对一个立方状态 m 而言，W_{slice}、W_{sheet}、W_{column} 分别表示在一个立方状态中重量非 0 的页、片、列的数量。例如，$W_{\mathrm{slice}}(m)=3$ 表示在立方状态 m 中，重量非 0 的页有 3 个；$W_{\mathrm{column}}(m_{\mathrm{slice}(0)})=3$ 表示在立方状态 m 的第 0 页中，重量非 0 的列有 3 个。

2.2 3.5 轮不可能差分区分器的充分条件

基于中间相错的思想，本节用性质 1 刻画 Saturnin 算法 3.5 轮不可能差分区分器的充分条件。

性质 1 在 Saturnin 算法中，记输入差分为 $\Delta\mathrm{in}$，经过 3.5 轮加密后输出差分为 $\Delta\mathrm{out}$。0.5 轮加密变换为在 3 轮加密后再进行 $S\circ \mathrm{MC}\circ S$ 变换。根据起始轮轮号 st 的不同，可得如下 2 个 3.5 轮不可能差分区分器的充分条件。

1) 当 $\mathrm{st} \bmod 2\equiv 1$ 时，有

$$F^{(\mathrm{st},3.5)}(m)=S\circ \mathrm{MC}\circ S\circ F_1\circ F_0\circ F_1(m) \quad (3)$$

可得 $\Delta\mathrm{in}\nrightarrow_{3.5-\mathrm{round}}\Delta\mathrm{out}$ 的充分条件为

$$W_{\mathrm{slice}}(\Delta\mathrm{in})+W_{\mathrm{slice}}(\Delta\mathrm{out})\leqslant 4 \quad (4)$$

2) 当 $\mathrm{st} \bmod 2\equiv 0$ 时，有

$$F^{(\mathrm{st},3.5)}(m)=S\circ \mathrm{MC}\circ S\circ F_0\circ F_1\circ F_0(m) \quad (5)$$

可得 $\Delta\mathrm{in}\nrightarrow_{3.5-\mathrm{round}}\Delta\mathrm{out}$ 的充分条件为

$$W_{\mathrm{sheet}}(\Delta\mathrm{in})+W_{\mathrm{sheet}}(\Delta\mathrm{out})\leqslant 4 \quad (6)$$

证明 以 1) 为例，证明充分性，已知 $W_{\mathrm{slice}}(\Delta\mathrm{in})+W_{\mathrm{slice}}(\Delta\mathrm{out})\leqslant 4$。

首先，因元胞替换和列混合变换不改变重量非 0 列的数量，故 $S\circ \mathrm{MC}\circ S$ 变换和对应的逆变换不会改

变重量非 0 页的数量和非 0 片的数量，即可得 $W_{\text{slice}}(m)=W_{\text{slice}}(S\circ\text{MC}\circ S(m))$；因页行移位变换不改变重量非 0 页的数量，即可得 $W_{\text{slice}}(m)=W_{\text{slice}}(\text{SR}_s(m))$。又因 F_1 变换中只包含页移位变换、元胞替换和列混合变换，故也不改变重量非 0 页的数量，即 $W_{\text{slice}}(m)=W_{\text{slice}}(F_1(m))$。则由上述分析可得，对 $\forall\Delta\text{in}'=S\circ\text{MC}\circ S\circ F_1(\Delta\text{in})$，都有 $W_{\text{slice}}(\Delta\text{in})=W_{\text{slice}}(\Delta\text{in}')$；对 $\forall\Delta\text{out}'=F_1^{-1}\circ S^{-1}\circ\text{MC}^{-1}\circ S^{-1}(\Delta\text{out})$，都有 $W_{\text{slice}}(\Delta\text{out})=W_{\text{slice}}(\Delta\text{out}')$。

其次，因为 Δout 是由 Δin 经过变换 $S\circ\text{MC}\circ S\circ F_1\circ F_0\circ F_1$ 得到的。则可以推得在 2 个立方状态 $\Delta\text{in}''=\text{SR}_t(\Delta\text{in}')$、$\Delta\text{out}''=\text{SR}_t(\Delta\text{out}')$ 之间还存在一个列混合变换 MC。

已知 $W_{\text{slice}}(\Delta\text{in})+W_{\text{slice}}(\Delta\text{out})\leqslant 4$，则由上述分析可得 $W_{\text{slice}}(\Delta\text{in}')+W_{\text{slice}}(\Delta\text{out}')\leqslant 4$，即在 2 个立方状态 Δin'、Δout' 中，至多存在 4 个重量非 0 页，则可得在 2 个立方状态 Δin'、Δout' 中，同一片号 $i\in\{0,1,2,3\}$ 下的两片 $\Delta\text{in}'(\text{sheet}_i)$、$\Delta\text{out}'(\text{sheet}_i)$ 至多有 4 个重量非 0 列，不妨设为第 0 片，则可得 $W_{\text{column}}(\text{sheet}_0)+W_{\text{column}}(\text{sheet}_0)\leqslant 4$。对这两片 $\Delta\text{in}'(\text{sheet}_0)$、$\Delta\text{out}'(\text{sheet}_0)$ 进行片移位变换 SR_t，可得 Δin'' 和 Δout'' 第 0 列的非 0 元胞数最多为 4 个，即 $W(\Delta\text{in}''_{\text{column}(0)})+W(\Delta\text{out}''_{\text{column}(0)})\leqslant 4$，但这与列混合变换 MC 的分支数为 5 矛盾，故 $\Delta\text{in}\not\to_{3.5-\text{round}}\Delta\text{out}$ 成立。

综上所述，$\Delta\text{in}\to_{3.5-\text{round}}\Delta\text{out}$ 的充分条件为 $W_{\text{slice}}(\Delta\text{in})+W_{\text{slice}}(\Delta\text{out})\leqslant 4$。

性质 1 中 2)部分的证明与上述类似。证毕。

由性质 1 可知，Saturnin 算法的不可能差分区分器与输入和输出差分的非 0 页数和非 0 片数有关。以起始轮的轮号为奇数为例，统计 Saturnin 算法 3.5 轮截断式不可能差分区分器数量。

设 $x=(x_0,\cdots,x_{63})\in\text{GF}(2^4)^{64}$ 为 Saturnin 算法的立方状态，令 θ 为 $\text{GF}(2^4)\to\text{GF}(2)$ 的映射，当 $x_i=0$ 时，有 $\theta(x_i)=0$；当 $x_i\neq 0$ 时，有 $\theta(x_i)=1$。则称 $\chi(x)=\chi(x_0,\cdots,x_{63})=(\theta(x_0),\cdots,\theta(x_{63}))$ 为 x 的模式。

在立方状态中，存在一个差分非 0 页时，共有 $A_1=\text{C}_4^1\times(2^{16}-1)\approx 2^{18}$ 个差分模式 $\chi(x)$；存在 2 个差分非 0 页时，共有 $A_2=\text{C}_4^2\times(2^{16}-1)^2\approx 2^{34.6}$ 个差分模式 $\chi(x)$；存在 3 个差分非 0 页时，共有 $A_3=\text{C}_4^3\times(2^{16}-1)^3\approx 2^{50}$ 个差分模式 $\chi(x)$。由性质 1 中充分条件的限制，通过排列组合，可以得到 3.5 轮截断式不可能差分区分器个数分布，如表 3 所示。

表 3　Saturnin 算法截断式不可能差分区分器个数分布

非 0 页数	非 0 片数		
	1	2	3
1	$A_1A_1=2^{36}$	$A_1A_2=2^{52.6}$	$A_1A_3=2^{68}$
2	$A_2A_1=2^{52.6}$	$A_2A_2=2^{69.2}$	0
3	$A_3A_1=2^{68}$	0	0

将表 3 的数据求和可得，由性质 1 构造的 3.5 轮截断式不可能差分区分器的个数约为 $2^{70.1}$，设计报告[1]给出的两条不可能差分区分器也在这 $2^{70.1}$ 个区分器之中。

为便于直观理解，令起始轮的轮号 st 为奇数，输入差分 Δin 的非 0 页数为 3，输出差分 Δout 的非 0 页数为 1。以上述输入差分 Δin、输出差分 Δout 为例，将性质 1 构造的 3.5 轮不可能差分区分器用图 4 展示，其中最左侧的状态代表第 0 页，从左至右的 4 个状态分别代表第 0 页至第 3 页。在列混合变换 MC 两侧，存在对应两列只有 4 个差分非 0 的元胞，这与列混合变换 MC 的分支数为 5 矛盾，故构成了截断式不可能差分区分器。

3　Saturnin 密码的 5.5 轮不可能差分攻击

由性质 1，令输入差分只有 3 个非 0 页，输出差分均只有一个非 0 页，构造了四类从第 3 轮到第 5.5 轮的 3.5 轮截断式不可能差分区分器。为方便阐述，本节以页行为单位，阐述区分器的截断差分及扩展攻击路径中的截断差分，并提出了页行的 3 个状态：满页行状态、空页行状态和单页行状态。满页行状态指页行的 4 个元胞差分值均非 0；空页行状态指页行的 4 个元胞差分值均为 0；单页行状态指在页行中，有且只有一个元胞差分值非 0，其余 3 个元胞的差分值为 0，且要求在一个状态中，单页行状态中差分值非 0 的元胞都在同一片。

3.1　页行视角和四类区分器

由 2.2 节可知，页行定义以页行为单位，则可以将 Saturnin 算法看作由 16 个页行组成。图 5 展示了页行视角的 Saturnin 算法状态，其中数字为页行号，从右至左分别代表第 0 页至第 3 页。

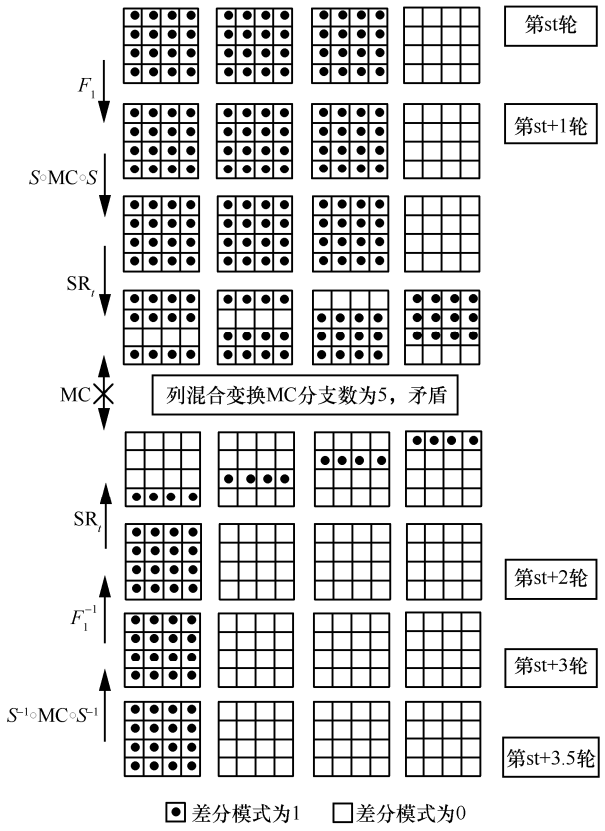


图 4　Saturnin 算法的 3.5 轮不可能差分区分器（起始轮号为奇数）

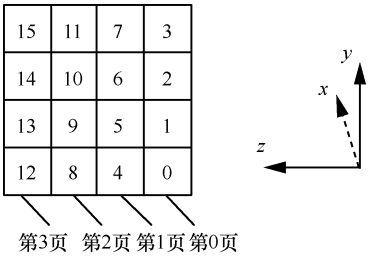


图 5　页行视角的 Saturnin 算法

本节由性质 1 构造了从第 3 轮至第 5.5 轮的四类截断式不可能差分区分器。区分器的四类输入截断差分：每类输入差分各有 4 个截断差分，即第 i 类区分器的第 $i-1$ 片上有 3 列共 12 元胞差分非 0，其余元胞差分为 0，有 4 种情况；区分器的四类输出截断差分：第 i 类区分器在第 $i-1$ 页上共 16 元胞差分非 0，其余元胞差分为 0。表 4 展示了四类截断式不可能差分区分器的具体截断差分，其中的数字是 4 个元胞差分值均非 0 列的列号。表 4 中的任意输入/输出截断差分组合，都可构成 3.5 轮不可能差分区分器，故一共有 64 个不可能差分区分器，每一类分别有 16 个不可能差分区分器。

设计者指出，他们提出的 2 个不可能差分区分器，即使是扩展 1 轮或 0.5 轮，得到的攻击路径都需要攻击全部密钥比特，故设计者没有构造出相应的不可能差分攻击方案。为使构造的区分器能够适用于不可能差分攻击，本文特意选取了表 4 中的 64 个区分器，这 64 个区分器满足以下 2 个特性。

表 4　Saturnin 算法的四类 3.5 轮不可能差分区分器

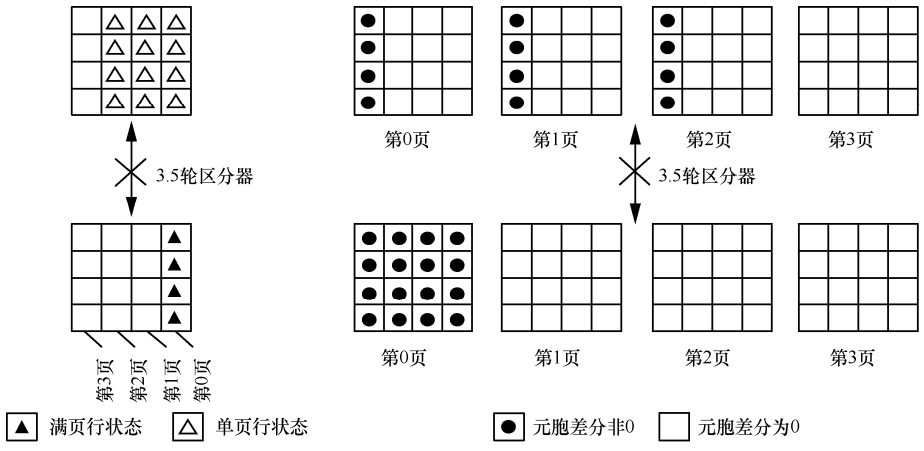
类别	序号	截断式输入差分	截断式输出差分
一	1	(0,4,8)	(0,1,2,3)
	2	(0,4,12)	(4,5,6,7)
	3	(0,8,12)	(8,9,10,11)
	4	(4,8,12)	(12,13,14,15)
二	5	(1,5,9)	(0,1,2,3)
	6	(1,5,13)	(4,5,6,7)
	7	(1,9,13)	(8,9,10,11)
	8	(5,9,13)	(12,13,14,15)
三	9	(2,6,10)	(0,1,2,3)
	10	(2,6,14)	(4,5,6,7)
	11	(2,10,14)	(8,9,10,11)
	12	(6,10,14)	(12,13,14,15)
四	13	(3,7,11)	(0,1,2,3)
	14	(3,7,15)	(4,5,6,7)
	15	(3,11,15)	(8,9,10,11)
	16	(7,11,15)	(12,13,14,15)

1) 区分器输入差分有 3 个非 0 面，且在这 3 个非 0 面中，都有且只有一个非 0 列；同时，这 3 个非 0 列都在一个页上。

2) 区分器的输出差分有且只有一个非 0 面。

将上述的 64 个区分器分为四类，根据特性 1)和 Saturnin 算法的具体结构，构造了四条 5.5 轮的不可能差分攻击路径，每条攻击路径都不需要攻击全部密钥比特。同时，本文仔细考虑了密钥生成算法，进一步减少了这四条攻击路径所需要攻击的密钥比特。

为便于直观理解，以第一个输入截断差分和第一个输出截断差分构成的不可能差分区分器为例，如图 6 所示，以页行、元胞为单位，分别展示此不可能差分区分器样例。图 6(a)是以页行为单位的截断差分示意，其中实心三角代表满页行状态，空心三角代表单页行状态，且此单页行状态中差分非 0 的元胞均在第 0 片；图 6(b)是以元胞为单位的截断差分示意。



(a) 以页行为单位的截断示意　　(b) 以元胞为单位的截断示意

图 6　Saturnin 算法的 3.5 轮不可能差分区分器样例

3.2　5.5 轮不可能差分攻击方案

基于 3.1 节中的四类不可能差分区分器，只向前扩展 2 轮，得到了四条 5.5 轮的不可能差分攻击路径。由第 i 类区分器扩展的攻击路径记作第 i 条攻击路径，以图 6 中的第一类不可能差分区分器为例构造第一条攻击路径样例，利用图 7 展示 Saturnin 算法的 5.5 轮不可能差分攻击路径样例，其中，变换 $S\circ\mathrm{MC}\circ S$ 简记为 MS，第一轮的子密钥加和常数加一起记为 AKT_1。在设计攻击方案时，先使用三条攻击路径只需要攻击 9 列子密钥，第四条路径需要攻击 10 列子密钥，即先使用需要攻击密钥较少的攻击路径，再利用攻击路径中的公共密钥信息，这样可以进一步改善攻击方案的整体复杂度。

本节将变换 $S\circ\mathrm{MC}\circ S$ 整体看作一个非线性变换，即将其看作一列 16 bit 的大 S 盒，用 16 个大 S 盒并置构成了此非线性变换。攻击方案的攻击步骤包括预处理阶段、数据处理阶段和密钥筛选阶段 3 个部分。

预处理阶段。为降低子密钥筛选阶段的时间复杂度，本节先对攻击过程中所需的计算存表，具体表的构造如下。

表 Λ。对 Saturnin 算法的 16 bit 大 S 盒，在给定非 0 输入差分 Δ_{in} 和非 0 输出差分 Δ_{out} 时，方程 $S(x)\oplus S(x\oplus\Delta_{\mathrm{in}})=\Delta_{\mathrm{out}}$ 平均可以求得一个解，构造表 Λ 的索引为 $(2^{16}-1)^2$ 个非 0 差分 $(\Delta_{\mathrm{in}},\Delta_{\mathrm{out}})$，每个 $(\Delta_{\mathrm{in}},\Delta_{\mathrm{out}})$ 存储对应的方程解和对应大 S 盒的输出 $\big(x,S(x)\big)$。

表 H_i。对第 i 条攻击路径，在第一轮 SR_s^{-1} 变换后，只有第 $i-1$ 列和第 $i+3$ 列中的 8 个元胞差分非 0，其余元胞差分均为 0，共有 2^{32} 个差分值 $\Delta x_1^{\mathrm{SR}_s^{-1}}$。对这 2^{32} 个差分值 $\Delta x_1^{\mathrm{SR}^{-1}}$ 做 $\mathrm{SR}_s^{-1}\circ\mathrm{MC}\circ\mathrm{SR}_s$ 变换得到 2^{32} 个差分值 Δx_1^{MS}，将这 2^{32} 个差分值 Δx_1^{MS} 存入对应的表 H_i。

数据处理阶段。在明文的(8,9,10,11,12,13,14,15)这 8 个页行位置取固定值，遍历其他 8 个页行，可以得到 2^{128} 个明文，将其称之为一个明文结构。对

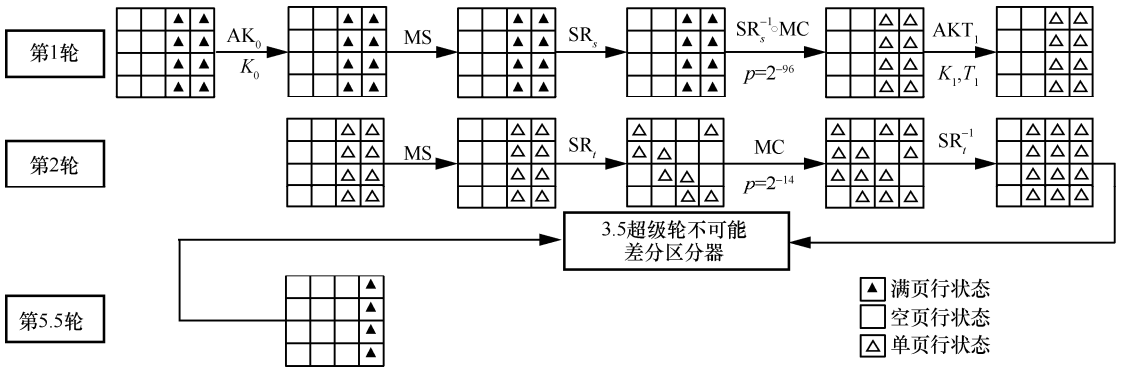


图 7　Saturnin 算法的 5.5 轮不可能差分攻击路径样例

每个明文结构下的 2^{128} 个明文，用快速排序技术[19]对明文做四次排序，即分别以密文的 4 个页行(0,1,2,3)、(4,5,6,7)、(8,9,10,11)和(12,13,14,15)为指标排序，一共可以得到 $4\times 2^{128}\times(2^{128}-1)\div 2\times 2^{-192}\approx 2^{65}$ 个符合此攻击路径的明文对，将其存入表 Ω 中。本节选取 2^{N_s} 个明文结构，故攻击方案中共有 2^{N_s+65} 个明文对，可在四条攻击路径中重复使用。

密钥筛选阶段。为方便理解，本节以列为单位叙述密钥，如四条攻击路径中，所需要攻击的白化密钥 K_0 为第 0 列至第 7 列密钥，记作 $K_{0,\mathrm{col}(0,\cdots,7)}$。四条攻击路径中所需攻击的第一轮子密钥 K_1 分别为 $K_{1,\mathrm{col}(0,4)}$、$K_{1,\mathrm{col}(1,5)}$、$K_{1,\mathrm{col}(2,6)}$ 和 $K_{1,\mathrm{col}(3,7)}$，由 Saturnin 算法的子密钥生成算法可知，其对应的主密钥分别为 $K_{0,\mathrm{col}(5,9)}$、$K_{0,\mathrm{col}(6,10)}$、$K_{0,\mathrm{col}(7,11)}$ 和 $K_{0,\mathrm{col}(8,12)}$，可以看出，除了第四条攻击路径，其余三条攻击路径中都有一列密钥与白化密钥相同，故前三条攻击路径中，只需要攻击 9 列子密钥，共 2^{144} bit 子密钥。攻击方案按攻击路径的序号顺序进行子密钥筛选。以第一条攻击路径为例，用全部 2^{N_s+65} 个明密对查表 H_1 得到密钥 K_0，再用 K_1 部分加密看是否能得到区分器输入差分，若得到，则为错误密钥排除。若在子密钥 $K_{0,\mathrm{col}(0,\cdots,7)}$ 固定的情况下，全部 2^{16} 个第 9 列子密钥 $K_{0,\mathrm{col}(9)}$ 均为错误密钥，则当前子密钥 $K_{0,\mathrm{col}(0,\cdots,7)}$ 为错误密钥，予以排除，在后续的攻击路径中不需要再次检测子密钥 $K_{0,\mathrm{col}(0,\cdots,7)}$，从而改进了此阶段的时间复杂度。经过全部四条攻击路径的筛选后，剩余的候选密钥用加密验证排除错误密钥，直至恢复出正确主密钥。为方便阐述，将变换 $\mathrm{SR}_s^{-1}\circ\mathrm{MC}\circ\mathrm{SR}_s$ 简记为 MR，密钥筛选阶段的具体步骤如下。

步骤 1 利用第一条攻击路径，求解 $K_{0,\mathrm{col}(0,\cdots,7)}$。

对于表 Ω 中的全部 2^{N_s+65} 个明文对，查表 H_1 可得 2^{32} 个差分值 Δx_1^{MS}，全部明文对差分和 Δx_1^{MS} 查表 Λ 可得 2^{N_s+97} 个 $(x_1^{\mathrm{AK}},x_1^{\mathrm{MS}})$，则白化密钥 $K_{0,\mathrm{col}(0,\cdots,7)}=P\oplus x_{1,\mathrm{col}(0,\cdots,7)}^{\mathrm{AK}}$。同时，在差分 Δx_1^{MS} 的条件下，计算中间状态对 $(x_{1,\mathrm{col}(0,4)}^{\mathrm{SR}_s^{-1}},x'^{\mathrm{SR}_s^{-1}}_{1,\mathrm{col}(0,4)})$，其中 $x_{1,\mathrm{col}(0,4)}^{\mathrm{SR}_s^{-1}}=\mathrm{MR}(x_{1,\mathrm{col}(0,\cdots,7)}^{\mathrm{MS}})$，另一个中间状态为 $x'^{\mathrm{SR}_s^{-1}}_{1,\mathrm{col}(0,4)}=\mathrm{MR}(x_{1,\mathrm{col}(0,\cdots,7)}^{\mathrm{MS}}\oplus\Delta x_{1,\mathrm{col}(0,\cdots,7)}^{\mathrm{MS}})$。以白化密钥 $K_{0,\mathrm{col}(0,\cdots,7)}$ 为指标，存储中间状态对，平均每个密钥 $K_{0,\mathrm{col}(0,\cdots,7)}$ 可以存储 $2^{N_s+97}\times 2^{-128}=2^{N_s-31}$ 个中间状态对 $(x_{1,\mathrm{col}(0,4)}^{\mathrm{SR}_s^{-1}},x'^{\mathrm{SR}_s^{-1}}_{1,\mathrm{col}(0,4)})$。

步骤 2 在第一条攻击路径的条件下，对每个 $K_{0,\mathrm{col}(0,\cdots,7)}$，求解 $K_{0,\mathrm{col}(9)}$。由每个密钥 $K_{0,\mathrm{col}(0,\cdots,7)}$ 下的 2^{N_s-31} 个中间状态对 $(x_{1,\mathrm{col}(0,4)}^{\mathrm{SR}_s^{-1}},x'^{\mathrm{SR}_s^{-1}}_{1,\mathrm{col}(0,4)})$，遍历全部 2^{16} 个子密钥 $K_{0,\mathrm{col}(9)}$，令中间状态对 $(x_{1,\mathrm{col}(0,4)}^{\mathrm{SR}_s^{-1}},x'^{\mathrm{SR}_s^{-1}}_{1,\mathrm{col}(0,4)})$ 异或加轮常数 T_1 和子密钥 $K_{0,\mathrm{col}(5,9)}$，经部分加密可得中间状态差分 $\Delta x_{1,\mathrm{col}(0,4)}^{\mathrm{MS}}$。若 $\mathrm{MR}(\Delta x_{1,\mathrm{col}(0,4)}^{\mathrm{MS}})$ 为第一类区分器的输入差分，则可以判定当前子密钥 $K_{0,\mathrm{col}(0,\cdots,7,9)}$ 为错误子密钥，予以排除。若在密钥 $K_{0,\mathrm{col}(0,\cdots,7)}$ 固定时，对应的全部 2^{16} 个子密钥 $K_{0,\mathrm{col}(9)}$ 都为错误密钥，则当前的 $K_{0,\mathrm{col}(0,\cdots,7)}$ 为错误密钥，予以排除，在后续的攻击路径中不需要再做检测。

步骤 3 由第二、三和四条攻击路径，利用对应的表 H_i，进行与上述两步类似的筛选。在四条攻击路径都筛选完毕后，最后可以得到 13 列中的候选密钥 $K_{0,\mathrm{col}(0,\cdots,12)}$。

步骤 4 穷举剩余的 3 列子密钥 $K_{0,\mathrm{col}(13,\cdots,15)}$，即得到了全部主密钥，进行加密验证，直至得到最后的正确主密钥。

3.3 复杂度分析

本节选取 2^{N_s} 个明文结构，则攻击方案所需要的数据复杂度是 2^{N_s+128}。

预处理阶段复杂度相较整体攻击方案而言较少，可忽略不计。

数据处理阶段主要为明文加密得到密文的时间，故数据处理阶段的时间复杂度为 2^{N_s+128} 次 5.5 轮加密。明文对只存储前两页的值，故所需的存储复杂度为 2^{N_s+65} 算法规模。

密钥筛选阶段需要对四条攻击路径的复杂度进行分析。

第一条攻击路径筛选子密钥的复杂度分析如下。

第一步所需的时间复杂度为 2^{N_s+97} 次查表，每个子密钥 $K_{0,\mathrm{col}(0,\cdots,7)}$ 索引下只存储两列中间状态对 $(x_{1,\mathrm{col}(0,4)}^{\mathrm{SR}_s^{-1}},x'^{\mathrm{SR}_s^{-1}}_{1,\mathrm{col}(0,4)})$，故所需的存储复杂度为 2^{N_s+95} 算法规模。

第二步只需要在一片上进行部分加密计算，故认为其为 0.25 轮加密。则第二步所需的时间复杂度为 $2^{128}\times 2^{N_s-31}\times 2^{16}\div(5.5\times 4)=2^{N_s+108.54}$ 次 5.5 轮加密。由上述分析可知，对一条攻击路径而言，时间复杂度主要耗时在第二步。

下面分析经过第一条攻击路径筛选后，候选子密钥的个数。

由一条攻击路径可知，一个错误子密钥不通过一个有效明文对的检测概率是 2^{-110}，则一个错误子密钥通过一条攻击路径的检测概率是 $p_L=(1-2^{-110})^{2^{N_s+65}}$。在第一条攻击路径中，经过 2^{N_s+65} 个有效明文对的检测后，可得候选密钥的个数为 $2^{144}\times p_L$。

第一条攻击路径共涉及 9 列密钥 $K_{0,\mathrm{col}(0,\cdots,7,9)}$。经过第一步的筛选，平均每个密钥 $K_{0,\mathrm{col}(0,\cdots,7)}$ 可以存储 2^{N_s-31} 个中间状态对 $(x_{1,\mathrm{col}(0,4)}^{\mathrm{SR}_s^{-1}},x'^{\mathrm{SR}_s^{-1}}_{1,\mathrm{col}(0,4)})$。在固定 8 列密钥密钥 $K_{0,\mathrm{col}(0,\cdots,7)}$ 的条件下，$K_{0,\mathrm{col}(9)}$ 通过第一条攻击路径的检测概率为 $(1-2^{-14})^{2^{N_s-31}}$，则全部 2^{16} 个 $K_{0,\mathrm{col}(9)}$ 都没有通过第一条攻击路径的检测概率为 $P_k=[1-(1-2^{-14})^{2^{N_s-31}}]^{2^{16}}\approx \mathrm{e}^{-2^{16-1.4425\times 2^{N_s-45}}}$，$P_k$ 也是当前固定密钥 $K_{0,\mathrm{col}(0,\cdots,7)}$ 是错误子密钥的概率，故经过第一条攻击路径筛选后 $K_{0,\mathrm{col}(0,\cdots,7)}$ 的候选密钥的个数为 $2^{128}\times(1-P_k)$。

后三条攻击路径筛选子密钥的复杂度分析如下。

第一步均与第一条攻击路径的第一步相同。在第二步中，只需要攻击 7 列白化密钥 $K_{0,\mathrm{col}(0,\cdots,7)}$ 中的候选密钥，故第二条攻击路径所需的时间复杂度为 $2^{N_s+108.54}\times(1-P_k)$ 次 5.5 轮加密；第三条攻击路径所需的时间复杂度为 $2^{N_s+108.54}\times(1-P_k)^2$ 次 5.5 轮加密。由于第四条攻击路径在第二步需多攻击一列子密钥，因此第四条攻击路径所需的时间复杂度为 $2^{N_s+108.54}\times(1-P_k)^3\times 2^{16}=2^{N_s+124.54}\times(1-P_k)^3$ 次 5.5 轮加密。

加密验证阶段，即第四步，经过四条攻击路径的检测后，通过的候选子密钥个数为 $N_k=(2^{208}-1)\times(p_L)^4$，需穷举 2^{48} 子密钥 $K_{0,\mathrm{col}(13,\cdots,15)}$，故所需的时间复杂度为 $2^{48}\times N_k$ 次 5.5 轮加密。

综上，本节攻击方案的数据复杂度为 2^{N_s+128}；存储复杂度为 2^{N_s+95} 算法规模；时间复杂度为各个阶段时间复杂度之和，其中数据处理阶段的时间复杂度是攻击方案中最耗时的部分。本文取 $N_s=48.88$，故本文攻击方案所需的数据复杂度为 $2^{176.88}$ 个选择明文，存储复杂度为 $2^{143.88}$ 算法规模，时间复杂度为 $2^{176.91}$ 次 5.5 轮加密。

4　结束语

本文对 Saturnin 算法进行不可能差分分析。首先，提出并证明了 Saturnin 算法 3.5 轮不可能差分区分器的充分条件，利用此条件可快速构造 $2^{70.1}$ 个截断式不可能差分区分器，从而为可以攻击方案的设计提供更多的选择。之后，构造了四类 3.5 轮区分器，向前扩展 2 轮得到了四条有相同的明文结构的攻击路径，利用这四条攻击路径，提出了 Saturnin 算法的 5.5 轮不可能差分攻击方案，数据、存储和时间复杂度分别为 $2^{176.88}$ 个选择明文、$2^{143.88}$ 算法规模和 $2^{176.91}$ 次 5.5 轮加密，这是目前可见的对 Saturnin 算法的一种不可能差分攻击方案。

参考文献：

[1] CANTEAUT A, DUVAL S, LEURENT G, et al. Saturnin: a suite of lightweight symmetric algorithms for post-quantum security[J]. IACR Transactions on Symmetric Cryptology, 2020(S1): 160-207.

[2] BIHAM E, BIRYUKOV A, SHAMIR A. Cryptanalysis of skipjack reduced to 31 rounds using impossible differentials[C]//International Conference on the Theory and Applications of Cryptographic Techniques. Berlin: Springer, 1999: 12-23.

[3] KIM J, HONG S, SUNG J, et al. Impossible differential cryptanalysis for block cipher structures[C]//International Conference on Cryptology in India. Berlin: Springer, 2003: 82-96.

[4] LUO Y Y, LAI X J, WU Z M, et al. A unified method for finding impossible differentials of block cipher structures[J]. Information Sciences, 2014, 263: 211-220.

[5] WU S B, WANG M S. Automatic search of truncated impossible differentials for word-oriented block ciphers[C]//Progress in Cryptology - INDOCRYPT 2012. Berlin: Springer, 2012: 283-302.

[6] MOUHA N, WANG Q J, GU D W, et al. Differential and linear cryptanalysis using mixed-integer linear programming[C]// International Conference on Information Security and Cryptology. Berlin: Springer, 2011: 57-76.

[7] XIANG Z J, ZHANG W T, BAO Z Z, et al. Applying MILP method to searching integral distinguishers based on division property for 6 lightweight block ciphers[C]//International Conference on the Theory and Application of Cryptology and Information Security. Berlin: Springer, 2016: 648-678.

[8] SHI D P, SUN S W, DERBEZ P, et al. Programming the demirci-

selçuk meet-in-the-middle attack with constraints[C]//International Conference on the Theory and Application of Cryptology and Information Security. Berlin: Springer, 2018: 3-34.

[9] LIU Y, SUN S, LI C. Rotational cryptanalysis from a differential-linear perspective[C]//Annual International Conference on the Theory and Applications of Cryptographic Techniques. Berlin: Springer, 2021: 741-770.

[10] CUI T T, CHEN S Y, FU K, et al. New automatic tool for finding impossible differentials and zero-correlation linear approximations[J]. Science China Information Sciences, 2020, 64(2): 1-3.

[11] 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.

[12] HU X C, LI Y Q, JIAO L, et al. Mind the propagation of states: new automatic search tool for impossible differentials and impossible polytopic transitions[C]//Advances in Cryptology - ASIACRYPT 2020. Berlin: Springer, 2020: 415-445.

[13] 张仕伟, 陈少真. SIMON 不可能差分及 0 相关路径自动化搜索算法[J]. 软件学报, 2018, 29(11): 3544-3553.
ZHANG S W, CHEN S Z. Automatic search algorithm for impossible differential trials and zero-correlation linear trials in SIMON[J]. Journal of Software, 2018, 29(11): 3544-3553.

[14] ZHANG K, GUAN J, HU B. Automatic search of impossible differentials and zero-correlation linear hulls for ARX ciphers[J]. China Communications, 2018, 15(2): 54-66.

[15] 武小年, 李迎新, 韦永壮, 等. GRANULE 和 MANTRA 算法的不可能差分区分器分析[J]. 通信学报, 2020, 41(1): 94-101.
WU X N, LI Y X, WEI Y Z, et al. Impossible differential distinguisher analysis of GRANULE and MANTRA algorithm[J]. Journal on Communications, 2020, 41(1): 94-101.

[16] 王旭姿, 吴保峰, 侯林, 等. SIMON 算法相关密钥不可能差分特征搜索[J]. 密码学报, 2021, 8(5): 881-893.
WANG X Z, WU B F, HOU L, et al. Searching for related-key impossible differentials for SIMON[J]. Journal of Cryptologic Research, 2021, 8(5): 881-893.

[17] DAEMEN J, RIJMEN V. Reijndael: the advanced encryption standard[J]. Dr. Dobb's Journal: Software Tools for the Professional Programmer, 2001, 26(3): 137-139.

[18] HOU T, CUI T, ZHANG J Y. Practical attacks on reduced-round 3D and Saturnin[J]. The Computer Journal, 2021, doi:10.1093/comjnl/bxab174.

[19] 张庆贵. 不可能差分攻击中的明文对筛选方法[J]. 计算机工程, 2010, 36(2): 127-129.
ZHANG Q G. Plaintext pair sieve methods in impossible differential attack[J]. Computer Engineering, 2010, 36(2): 127-129.

[作者简介]

蒋梓龙（1992− ），男，江苏南通人，信息工程大学博士生，主要研究方向为对称密码设计与分析。

金晨辉（1965− ），男，河南扶沟人，博士，信息工程大学教授、博士生导师，主要研究方向为密码学与信息安全。