ARIA 密码的积分故障分析

沈煜[1]，李玮[1,2,3,4]，谷大武[2]，吴益鑫[1]，曹珊[1]，刘亚[5]，刘志强[2]，周志洪[4]

（1. 东华大学计算机科学与技术学院，上海 201620；2. 上海交通大学计算机科学与工程系，上海 200240；
3. 上海市可扩展计算与系统重点实验室，上海 200240；4. 上海市信息安全综合管理技术研究重点实验室，上海 200240；
5. 上海理工大学计算机科学与工程系，上海 200093）

摘 要：ARIA 算法作为韩国国家标准分组密码，为信息系统中的软硬件应用实现提供安全保障。在 ARIA 算法的故障攻击研究中，故障导入的范围仅为最后两轮运算。结合应用环境及组件的计算能力，如何扩大故障分析的攻击范围已成为目前研究的难点，为此，提出了针对 ARIA 算法的新型积分故障分析方法、所提方法可以将故障导入扩展到算法的倒数第三轮和第四轮，从而成功地恢复出原始密钥并破译算法。实验结果表明，ARIA 算法的内部轮运算容易受到积分故障攻击的威胁，同时也为其他分组密码标准的安全性分析提供了重要参考。

关键词：密码分析；分组密码；ARIA 算法；积分故障分析

中图分类号：TP309.7

文献标识码：A

doi: 10.11959/j.issn.1000−436x.2019033

Integral fault analysis of the ARIA cipher

SHEN Yu[1], LI Wei[1,2,3,4], GU Dawu[2], WU Yixin[1], CAO Shan[1], LIU Ya[5], LIU Zhiqiang[2], ZHOU Zhihong[4]

1. School of Computer Science and Technology, Donghua University, Shanghai 201620, China
2. Department of Computer Science and Engineering, Shanghai Jiao Tong University, Shanghai 200240, China
3. Shanghai Key Laboratory of Scalable Computing and Systems, Shanghai 200240, China
4. Shanghai Key Laboratory of Integrate Administration Technologies for Information Security, Shanghai 200240, China
5. Department of Computer Science and Engineering, University of Shanghai for Science and Technology, Shanghai 200093, China

Abstract: ARIA is a Korean standard block cipher, which is flexible to provide security for software and hardware implementation. Since its introduction, some research of fault analysis is devoted to attacking the last two rounds of ARIA. It is an open problem to know whether provoking faults at some former rounds of ARIA allowed recovering the secret key. An answer was given to solve this problem by showing a novel integral differential fault analysis on two rounds earlier of ARIA. The mathematical analysis and simulating experiments show that the attack can successfully recover its secret key by fault injections. The results in this study describe that the integral fault analysis is a strong threaten to the security of ARIA. The results are beneficial to the analysis of the same type of other block ciphers.

Key words: crypt analysis, block cipher, ARIA cipher, integral fault analysis

1 引言

分组密码作为实现消息保密、数据完整性保护和实体认证的核心基础体制，其设计、分析与实现方法一直是密码学研究的主流，在信息系统的安全领域发挥着重要的应用。ARIA 算法是韩国国家安全研究所于 2003 年提出的一种符合高级加密算法征集标准的分组密码，并且被商业部、工业部和能源部确认为国家标准分组密码，保障信息系统的数据安全[1-3]。研究表明，ARIA 算法对差分分析法、

收稿日期：2018−06−19；**修回日期**：2019−01−01

通信作者：李玮，liwei.cs.cn@gmail.com

基金项目：国家自然科学基金资助项目（No.61772129）；国家密码发展基金资助项目（No.MMJJ20180101）

Foundation Items: The National Natural Science Foundation of China (No.61772129), The National Cryptography Development Fund (No.MMJJ20180101)

线性分析法、截段–高阶差分分析法、不可能差分分析法、滑动攻击法、积分攻击法等密码分析方法具有足够的安全冗余[4-8]，具有优良的软硬件实现效率，在处理器应用上具有良好的实现性能。

然而，在现实环境中，只从 ARIA 算法的数学模型研究其安全性已经远远不够，必须从实现角度来考虑，即使密码算法在传统密码分析方法下是安全的，如果不能在运行平台上安全地加以实现，也不能正常发挥出预期作用。与传统的保密通信环境相比，如今的运行环境通常面临着更大的安全威胁。譬如银行卡、手持无线设备、汽车遥控锁、交通卡、门禁卡等密码载体的持有者不仅可以是正常用户，也可以是攻击者。攻击者借助异常时钟、微波辐射、激光照射等方式干预密码变换的正常过程，导致运算过程中发生错误，产生错误的输出结果，这种攻击方式比传统密码分析破译速度更快，有效地达到密码算法内关键数据的复制、篡改或伪造的目的，通常被称为故障分析[9-12]，由于其良好的攻击效果和潜在的发展前景，已成为密码分析和密码工程领域发展广受关注的方向之一。

故障分析在发展中逐步衍生出多种攻击方法，Biham 等[13]于 1997 年提出的差分故障分析法是目前分析范围最广、威胁最大且发展最迅速的一种攻击技术，被广泛应用于密码芯片的安全性研究中。之后，无效故障分析法、碰撞故障分析法、不可能故障分析法等故障攻击方法应运而生，这些方法充分结合了传统密码分析方法及软硬件实现技术，在实际运行环境中具有更加广阔的应用前景。但是，这些故障攻击方法的故障导入范围局限于最后若干轮，攻击轮数范围有限。

积分故障分析法是利用积分关系深入密码算法内部达到最深轮数的攻击方法，易于在物联网等应用中实现，攻击者通过分析内部状态的积分关系，便可以较低的计算量和存储量推导出正确的密钥。目前，国内外关于 ARIA 算法抵抗积分故障攻击的安全性尚未有公开发表的成果。因此，本文提出了一种针对 ARIA 算法的新型积分故障分析方法，可以借助异常时钟、微波辐射、激光照射等方式使 ARIA 算法内部发生错误，达到扩大攻击范围并破译原始密钥的目的。所提方法对提升各类高端智能卡系统的防护与破译能力，具有重要的理论和实际意义。

2　研究背景

1997 年，Boneh 等[14-15]利用随机故障法成功破译了公钥密码算法，此后故障分析法在评测密码体制安全性的方法中拥有重要地位。随着故障注入精度及软硬件逆向技术的不断提高，对密码载体成功实施故障攻击所依赖的前提条件不断减弱，成本不断下降，故障分析环境已经可以在一定程度上被控制，这使故障分析法逐步发展为攻击主要密码体制的较为容易实施且最难防御的一种攻击技术[16-19]。

目前已有的针对 ARIA 算法的故障分析方法均为差分故障分析法 [20-22]。在攻击过程中，攻击者首先将故障导入 ARIA 算法的最后两轮来恢复最后一轮的子密钥，然后对正确的密文进行解密，以获得最后一轮的输入，即倒数第 2 轮的输出，重复以上步骤来恢复最后 4 轮的子密钥，直至破译原始密钥为止。国内外的最新研究结果均集中于在 ARIA 算法的倒数第 2 轮导入随机字节故障来恢复最后一轮的子密钥。2008 年，Li 等[20]首次提出了针对 ARIA 算法的面向随机单字节故障模型的差分故障分析法。后来，Park 等[21]在相同故障模型下提出一种改进的差分故障分析法，利用线性层的差异性，实现以较少的故障来恢复最后一轮子密钥，提升了攻击效率。Kim 等[22]提出了面向多字节的差分故障方法，达到以较少故障数破译最后一轮子密钥的目的。这些方法充分结合了传统的差分分析方法以及软硬件实现技术，提高了故障导入的效率。然而，在物联网的应用环境中，故障可以被导入在密码算法的任意轮数，如果故障导入点可以扩展到更大的轮数范围，那么故障分析具有更强的威胁能力，因此，研究 ARIA 算法的新型故障分析方法是非常必要的。

Phan 等[23]于 2006 年首次提出了针对 AES（advanced encryption standard）算法的积分故障攻击法。该方法通过选择明文攻击实现的前提条件，在 AES 算法运行的倒数第 4 轮中加入随机故障，使其执行某些错误的操作，产生错误的密文，利用中间数据的积分关系即可恢复最后一轮子密钥；然后解密密文，破译更多的子密钥，直到破译原始密钥。虽然 ARIA 算法与 AES 算法具有相同的算法结构，但是由于 ARIA 算法中多轮故障路径的扩散性和混乱性更加复杂, 大大增加了攻击的难度，所以针对 ARIA 算法的积分故障分析，国内外还未有文献发表。

本文提出了针对 ARIA 算法的新型积分故障分析，利用故障与子密钥之间的积分关系，构造了 2

个不同的积分区分器，从而使故障可以导入在倒数第 3 轮和第 4 轮运算中，进一步扩展了积分故障分析的攻击范围。由于积分故障路径包含更多的轮数，因此攻击者能将故障导入 ARIA 算法的更深轮数。现有的针对 ARIA 算法的故障分析方法的对比如表 1 所示。

表 1 针对 ARIA 算法的故障分析方法的对比

算法	故障分析	故障模型	首次故障导入轮数
文献[20-21]算法	差分故障分析	字节	第 11 轮
文献[22]算法	差分故障分析	多字节	第 11 轮
本文算法	积分故障分析	字节	第 9~10 轮

3 ARIA 算法简介

ARIA 算法是一种典型的具有代换置换网络结构的迭代型分组密码，其密钥长度为 3 种，分别是 128 bit、192 bit 和 256 bit；分组长度为 128 bit[3]；所对应的轮数分别为 12 轮、14 轮和 16 轮，分别表示为 ARIA-128、ARIA-192 和 ARIA-256，如表 2 所示。

表 2 ARIA 算法的密钥长度和轮数的对应关系

算法	分组长度/bit	密钥长度/bit	轮数/轮
ARIA-128	128	128	12
ARIA-192	128	192	14
ARIA-256	128	256	16

ARIA 算法由加密、解密和密钥编排这 3 个部分组成，其中，加密部分与解密部分的过程相同，不同之处在于子密钥的使用顺序。

3.1 符号说明

以 ARIA-128 算法为例。设 $X \in F_{2^8}^{16}$ 为明文输入，$Y \in F_{2^8}^{16}$ 为密文输出，$K \in F_{2^8}^{16}$ 为主密钥，$l \in \{12,14,16\}$ 为算法轮数，$\mathrm{ek}_d \in F_{2^8}^{16}$ 表示 K 生成的第 d 轮子密钥，其中 $1 \leqslant d \leqslant l+1$。

记 $A_i=(a_{i,0},a_{i,1},a_{i,2},\cdots,a_{i,15}) \in F_{2^8}^{16}$，$B_i=(b_{i,0},b_{i,1},b_{i,2},\cdots,b_{i,15}) \in F_{2^8}^{16}$，$A$ 和 B 分别为第 i 轮置换层的输入和输出，其中 $1 \leqslant i \leqslant l$。

记 $C_i=(c_{i,0},c_{i,1},c_{i,2},\cdots,c_{i,15}) \in F_{2^8}^{16}$ 为第 i 轮扩散层的故障输出，其中 $1 \leqslant i \leqslant l$。

记 SL（substitution layer）和 DL（diffusion layer）分别为置换层运算和扩散层运算，SL^{-1} 和 DL^{-1} 分别为置换层和扩散层的逆运算。W_0、W_1、W_2 和 W_3 是密钥编排方案中初始化部分产生的 4 个数据块。

3.2 ARIA 算法描述

ARIA 算法的结构如图 1 所示。

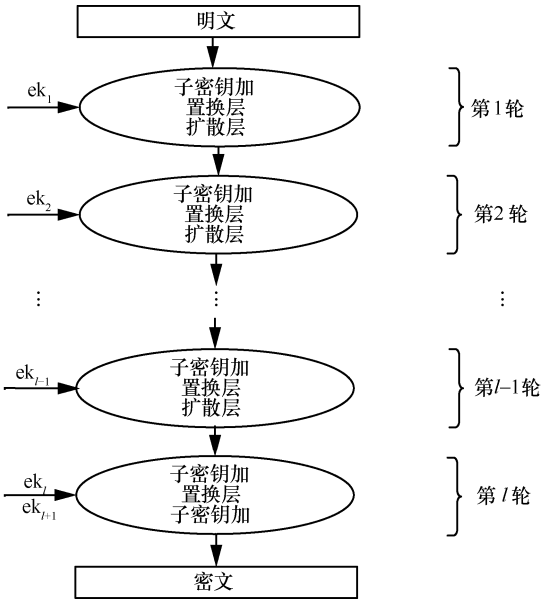


图 1 ARIA 算法的结构

1) 子密钥加（RKA, round key addition）：子密钥与中间状态逐字节进行异或运算。

2) 置换层：中间状态经过 16 个 S 盒进行置换，其中 S 盒有 2 种类型，分别用于奇轮数和偶轮数。

3) 扩散层：对中间状态进行线性映射变换，通过与一个 16×16 的二进制矩阵 $\boldsymbol{C}$ 相乘实现，$\boldsymbol{C}$ 如下所示。最后一轮中没有扩散层，被子密钥加运算所替代。

$$\boldsymbol{C}=\begin{bmatrix}
0&0&0&1&1&0&1&0&1&1&0&0&0&1&1&0\\
0&0&1&0&0&1&0&1&1&1&0&0&1&0&0&1\\
0&1&0&0&1&0&1&0&0&0&1&1&1&0&0&1\\
1&0&0&0&0&1&0&1&0&0&1&1&0&1&1&0\\
1&0&1&0&0&1&0&0&1&0&0&1&0&0&1&1\\
0&1&0&1&1&0&0&0&0&1&1&0&0&0&1&1\\
1&0&1&0&0&0&0&1&0&1&1&0&1&1&0&0\\
0&1&0&1&0&0&1&0&1&0&0&1&1&1&0&0\\
1&1&0&0&1&0&0&1&0&0&1&0&0&1&0&1\\
1&1&0&0&0&1&1&0&0&0&0&1&1&0&1&0\\
0&0&1&1&0&1&1&0&1&0&0&0&0&1&0&1\\
0&0&1&1&1&0&0&1&0&1&0&0&1&0&1&0\\
0&1&1&0&0&0&1&1&0&1&0&1&1&0&0&0\\
1&0&0&1&0&0&1&1&1&0&1&0&0&1&0&0\\
1&0&0&1&1&1&0&0&0&1&0&1&0&0&1&0\\
0&1&1&0&1&1&0&0&1&0&1&0&0&0&0&1
\end{bmatrix}$$

ARIA 算法的加密过程如算法 1 所示。

算法 1　ARIA 算法的加密过程

输入　明文 X，密钥 K

输出　密文 Y

1) T=X /将 X 赋值给中间状态 T/

2) for i=1 to l−1

3) T=DL (SL(RKA(T, ek$_i$)))

4) end for

5) i= i+1

6) T=SL(RKA(T, ek$_i$))

7)Y=RKA(T, ek$_{i+1}$)

3.3　密码编排方案

密钥编排由初始化和子密钥生成这 2 个部分组成，具体如下。

1)初始化：4 个 128 bit 的数据块 W_0、W_1、W_2 和 W_3 通过原始密钥 K 基于 3 轮的 Feistel 网络结构而产生。

2)子密钥生成：通过 W_0、W_1、W_2 和 W_3 进行一系列异或运算、右移和左移操作，获得加密变换所需各轮子密钥。

4　ARIA 算法的积分故障分析方法

4.1　故障假设和故障模型

故障假设表明了攻击者具备的能力，本文采用明文攻击，即攻击者可以任意选择明文进行处理，从而获得相对应的密文；故障模型表明了故障的状态，本文采用随机单字节模型，将单字节故障引入运算过程中某些轮的指定位置，并可以随机设定故障值。

4.2　主要过程

攻击者选择随机明文并使用同一个密钥进行加密，在运算过程中的某一轮中导入一组随机故障，从而获得一组错误的密文。故障导入既可以利用异常时钟、异常电流、微波辐射、激光照射、涡流磁场等手段通过真实的密码硬件来实现，也可以通过修改程序等软件模拟技术的方式来实现。攻击者通过构造合适的积分故障区分器，可以恢复最后一轮的子密钥；然后解密正确的密文，获得最后一轮的输入，即倒数第 2 轮的输出。重复上述过程，通过密钥编排方案直至破译原始密钥。

4.3　3 轮攻击方案

针对 ARIA 算法，本文构造了一个 2 轮积分区分器，使得故障位置和子密钥恢复范围在 3 轮之间，从而可以破译 ARIA 算法，具体算法如图 2 所示。

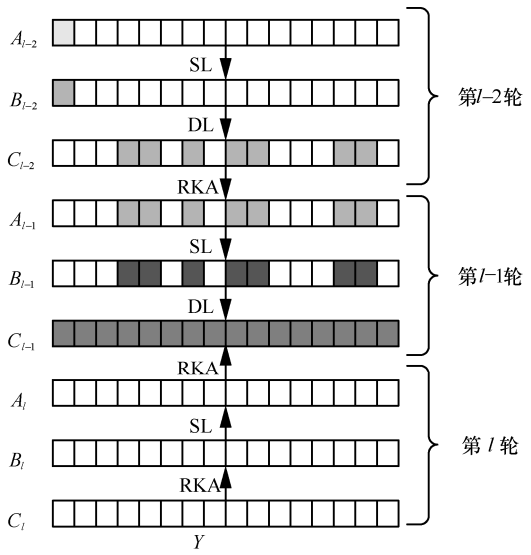


图 2　故障导入在倒数第 3 轮的扩散路径

在构造积分区分器时，需要做以下定义。

定义 1　如果定义在 F_{2^n} 上的集合 $O=\{\alpha^{(u)} \mid 0\leqslant u\leqslant 2^n-1, n\in N\}$，对任意 $0\leqslant u<v\leqslant 2^n-1$，均有 $\alpha^{(u)}\neq\alpha^{(v)}$，则称 O 为 F_{2^n} 上的活跃集[24]。

定义 2　若定义在 F_{2^n} 上的集合 $P=\{\alpha^{(u)} \mid 0\leqslant u\leqslant 2^n-1, n\in N\}$，对任意 $0\leqslant u<v\leqslant 2^n-1$，均有 $\alpha^{(u)}=\alpha^{(v)}$，则称 P 为 F_{2^n} 上的稳定集[24]。

定义 3　若定义在 F_{2^n} 上的集合 $Q=\{\alpha^{(u)} \mid 0\leqslant u\leqslant 2^n-1, n\in N\}$ 满足

$$\sum_{u=0}^{2^n-1}a^{(u)}=0$$

则称 Q 为 F_{2^n} 上的平衡集[24]。

攻击者为了构造 ARIA 算法的积分区分器，需要遵循以下性质：稳定/活跃字集通过双射（如可逆 S 盒，子密钥加）后，仍然是稳定/活跃的；2 个活跃字集的和一定是平衡字集；稳定字集与活跃字集的和为活跃集；2 个平衡字集的和为平衡字集。

结合上述定义和性质，攻击者可以追踪 2 轮运算中活跃字节的位置变化，如图 2 所示。具体路径为：第 1 轮中活跃字节通过扩散层变为 7 个活跃字节，然后这 7 个活跃字节经由第 2 轮扩散层变为 16 个活跃字节，这构成了第 2 轮扩散层输出中的一组集合；通过遍历该集合中所有字节值，

使得第 2 轮的输出字节的积分之和为零。因此，在 2 轮积分区分器中，第 2 轮输出中的每个字节都是平衡的，并适用于第 1 轮中所有可能的活跃字节位置。

定理 1 设多项式 $f(x)=\sum_{u=0}^{q-1}a^{(u)}x^u\in F_q[x]$，其中 q 是某个素数的方幂，则 $\sum_{x\in F_q}f(x)=-a^{q-1}$。

定理 2 若多项式 $f(x)=\sum_{u=0}^{q-1}a^{(u)}x^u\in F_q[x]$ 是置换多项式，则 $a^{q-1}=0$。

利用上述 2 个定理[25]来证明命题 1 中所构造的 ARIA 算法的 2 轮积分区分器的正确性。

命题 1 对于 ARIA 算法，如果第 1 轮输入中只有一个活跃字节而其他都是稳定字节，那么第 2 轮输出中的所有字节都是平衡字节。

证明 设 $\gamma_0,\gamma_1,\gamma_2,\cdots,\gamma_{15}$ 为第 1 轮的输入字节，其中，只有一个字节 β 为变量，其他字节均为常量。假设 $\beta=\gamma_0$，则第 1 轮的输入表示为

$$(\gamma_0,\gamma_1,\gamma_2,\cdots,\gamma_{15})=(\beta,\gamma_1,\gamma_2,\cdots,\gamma_{14},\gamma_{15})$$

其中，β 是变量，$\gamma_1,\cdots,\gamma_{15}$ 均为常量。根据图 2 和表 3 中 ARIA 算法的运算特点，第 1 轮的输出可以表示为

$$(\delta_0,\delta_1,\delta_2,f_3(\beta),f_4(\beta),\delta_5,f_6(\beta),\delta_7,$$
$$f_8(\beta),f_9(\beta),\delta_{10},\delta_{11},\delta_{12},f_{13}(\beta),f_{14}(\beta),\delta_{15})$$

其中，δ_0、δ_1、δ_2、δ_5、δ_7、δ_{10}、δ_{11}、δ_{12} 和 β_{15} 是常量，$f_3(\beta)$、$f_4(\beta)$、$f_6(\beta)$、$f_8(\beta)$、$f_9(\beta)$、$f_{13}(\beta)$、$f_{14}(\beta)$ 属于 F_{2^8} 域上的置换多项式。

第 2 轮的输出为

$$(h_0(\beta),h_1(\beta),h_2(\beta),\cdots,h_{15}(\beta))$$

其中，$h_0(\beta),\cdots,h_{15}(\beta)$ 属于 F_{2^8} 域上的置换多项式。第 2 轮输出中的每个字节都可以用 F_{2^8} 上 7 个置换多项式的积分和来表示。

假设第 2 轮输出中的第一个字节为

$$p_0(\beta)\oplus p_1(\beta)\oplus p_2(\beta)\oplus\cdots\oplus p_7(\beta)$$

其中，$p_0(\beta),\cdots,p_7(\beta)$ 是 F_{2^8} 上的置换多项式。结合置换多项式的性质，则有

$$\sum_{\beta\in F_{2^8}}p_0(\beta)=\sum_{\beta\in F_{2^8}}p_1(\beta)=$$
$$\sum_{\beta\in F_{2^8}}p_2(\beta)=\cdots=\sum_{\beta\in F_{2^8}}p_7(\beta)=0$$

因此

$$\sum_{\beta\in F_{2^8}}(p_0(\beta)\oplus p_1(\beta)\oplus p_2(\beta)\oplus\cdots\oplus p_7(\beta))=$$
$$\sum_{\beta\in F_{2^8}}p_0(\beta)\oplus\sum_{\beta\in F_{2^8}}p_1(\beta)\oplus\sum_{\beta\in F_{2^8}}p_2(\beta)\oplus\cdots\oplus\sum_{\beta\in F_{2^8}}p_7(\beta)=$$
$$0\oplus0\oplus0\oplus\cdots\oplus0=0$$

证毕。

根据 2 轮积分区分器，实现 3 轮攻击方案的具体步骤如下。

步骤 1 选择任意明文 X，使用原始密钥 K 进行加密，并获得正确密文 Y。

步骤 2 最后 2 轮子密钥 ek_{l+1} 具有如下关系。

$$A_l=\mathrm{SL}^{-1}(Y\oplus\mathrm{ek}_{l+1}),$$
$$C_{l-1}=\mathrm{SL}^{-1}(Y\oplus\mathrm{ek}_{l+1})\oplus\mathrm{ek}_l$$

表 3 ARIA 算法的故障路径扩散

A_{l-2} 中的非零差分字节	A_{l-1} 中的非零差分字节
0	3，4，6，8，9，13，14
1	2，5，7，8，9，12，15
2	1，4，6，10，11，12，15
3	0，5，7，10，11，13，14
4	0，2，5，8，11，14，15
5	1，3，4，9，10，14，15
6	0，2，7，9，10，12，13
7	1，3，6，8，11，12，13
8	0，1，4，7，10，13，15
9	0，1，5，6，11，12，14
10	2，3，5，6，8，13，15
11	2，3，4，7，9，12，14
12	1，2，6，7，9，11，12
13	0，3，6，7，8，10，13
14	0，3，4，5，9，11，14
15	1，2，4，5，8，10，15

攻击者在加密算法的第 $l-2$ 轮中的 A_{l-2} 或 B_{l-2} 中导入随机故障，从而获得错误密文，如图 2 所示。在故障的导入过程中，均可以导致最后更多轮中发生变化，例如 C_{l-2}、A_{l-1}、B_{l-2}、C_{l-1}、A_l、B_l、C_l 等的字节变化会导致出现 ΔC_{l-2}、ΔA_{l-1}、ΔB_{l-2}、ΔC_{l-1}、ΔA_l、ΔB_l、ΔC_l，最后生成错误密文。攻击者可以随机选择一个字节位置改变故障值，取值在[0,255]之间。因此，基于 C_{l-1} 的积分关系如下。

$$\sum_{u=0}^{255} C_{l-1}^{(u)} = \sum_{u=0}^{255} (\mathrm{SL}^{-1}(Y^{(u)} \oplus \mathrm{ek}_{l+1}^{(u)}) \oplus \mathrm{ek}_{l}^{(u)}) = \sum_{u=0}^{255} (\mathrm{SL}^{-1}(Y^{(u)} \oplus \mathrm{ek}_{l+1}) \oplus \mathrm{ek}_{l}) = \sum_{u=0}^{255} (\mathrm{SL}^{-1}(Y^{(u)} \oplus \mathrm{ek}_{l+1}))$$

其中，$C_{l-1}^{(u)}$、$Y^{(u)}$、$\mathrm{ek}_{l+1}^{(u)}$和$\mathrm{ek}_{l}^{(u)}$分别表示第 u 次故障导入时C_{l-1}、Y、ek_{l+1}和ek_{l}的值，其中$0 \leqslant u \leqslant 255$。显然，如果$u=0$，那么所有的值都是正确值。基于 2 轮积分区分器的性质，C_{l-1}的每个字集是平衡的，可得

$$\sum_{u=0}^{255} C_{l-1}^{(u)} = 0$$

攻击者通过穷尽搜索ek_{l+1}候选值的每个字节，然后结合同一个密钥和不同明文得到的正确和错误密文，进行重复操作，直到ek_{l+1}候选集合中只有一个元素为止，即可求出正确的子密钥ek_{l+1}。

步骤 3　假设故障被导入到第 l–3 轮，攻击者通过子密钥ek_{l+1}对正确的密文进行解密最后一轮，得到A_l。此时，A_l既是最后一轮的输入，同时也是倒数第 2 轮的输出，因此

$$C_{l-2} = \mathrm{SL}^{-1}(\mathrm{DL}^{-1}(A_l \oplus \mathrm{ek}_l)) \oplus \mathrm{ek}_{l-1} = \mathrm{SL}^{-1}(\mathrm{DL}^{-1}(A_l) \oplus \mathrm{DL}^{-1}(\mathrm{ek}_l)) \oplus \mathrm{ek}_{l-1} = \mathrm{SL}^{-1}(A_l^{'} \oplus \mathrm{ek}_l^{'})) \oplus \mathrm{ek}_{l-1}$$

其中，$A_l^{'} = \mathrm{DL}^{-1}(A_l) = \mathrm{DL}^{-1}(\mathrm{SL}^{-1}(Y \oplus \mathrm{ek}_{l+1}))$，$\mathrm{ek}_l^{'} = \mathrm{DL}^{-1}(\mathrm{ek}_l)$。

攻击者构建C_{l-2}的积分关系如下。

$$\sum_{u=0}^{255} C_{l-2}^{(u)} = \sum_{u=0}^{255} (\mathrm{SL}^{-1}(A_l^{'(u)} \oplus \mathrm{ek}_l^{'(u)}) \oplus \mathrm{ek}_{l-1}^{(u)}) = \sum_{u=0}^{255} (\mathrm{SL}^{-1}(A_l^{'(u)} \oplus \mathrm{ek}_l^{'}) \oplus \mathrm{ek}_{l-1}) = \sum_{u=0}^{255} (\mathrm{SL}^{-1}(A_l^{'(u)} \oplus \mathrm{ek}_l^{'}))$$

其中，$C_{l-2}^{(u)}$、$A_l^{'(u)}$、$\mathrm{ek}_l^{'(u)}$和$\mathrm{ek}_{l-1}^{(u)}$分别表示第 u 次故障导入时C_{l-2}、$A_l^{'}$、$\mathrm{ek}_l^{'}$和ek_{l-1}的值，并且$0 \leqslant u \leqslant 255$。因为$C_{l-2}$的每个字集都是平衡字集，因此可得

$$\sum_{u=0}^{255} C_{l-2}^{(u)} = 0$$

攻击者计算扩散层中C_{l-2}的总和，穷尽搜索$\mathrm{ek}_l^{'}$中的每个字节值。利用$\mathrm{ek}_l = \mathrm{DL}(\mathrm{ek}_l^{'})$，即可推导出子密钥$\mathrm{ek}_l$。同理，通过将故障导入在第 l–4 轮和第 l–5 轮，攻击者可以分别恢复其他 2 个子密钥ek_{l-1}和ek_{l-2}。

步骤 4　结合密钥编排方案，攻击者通过最后 4 个子密钥，即可推导出 ARIA 算法的原始密钥。

4.4　4 轮攻击方案

在上述 3 轮攻击方案中，攻击者可以对步骤 2 中的第 l–2 轮和步骤 3 中的第 l–3 轮、第 l–4 轮和第 l–5 轮分别导入故障，从而恢复最后 4 轮子密钥。然而，在轻量级环境中，随机故障可能发生在算法的其他轮中。如果将故障位置扩展到更多轮，则积分故障分析的实现范围更大。

结合命题 2 构建了 2.5 轮积分区分器，此时故障导入位置分别推广到步骤 2 中的第 l–3 轮和步骤 3 中的第 l–4 轮、第 l–5 轮和第 l–6 轮。此时故障位置和子密钥处于 4 轮运算范围内，因此称为 4 轮攻击方案，具体如图 3 所示。

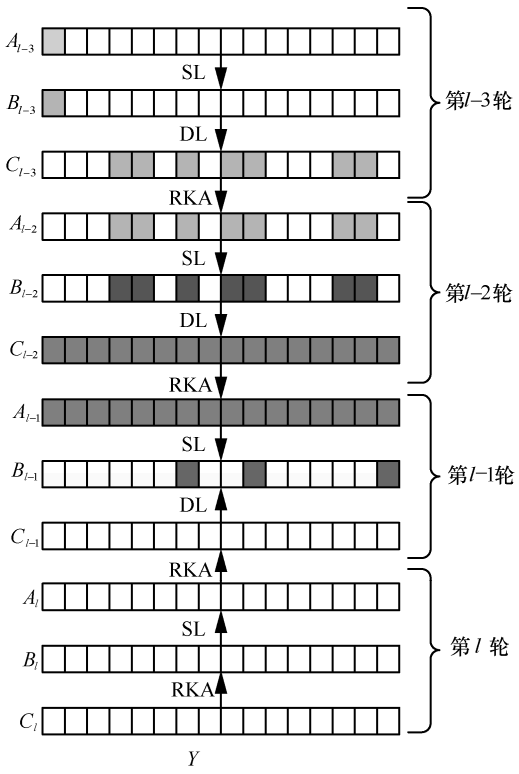


图 3　故障导入在倒数第 3.5 轮的扩散路径

命题 2　对于 ARIA 算法，如果第 1 轮输入中只有一个活跃字节而其他都是稳定字节，则在第 3 轮替代层的输出存在 3 个平衡字节[5]。

4 轮攻击方案采用了与 3 轮攻击方案相同的步骤 1 和步骤 4，不同之处在于步骤 2 与步骤 3。4 轮

攻击方案的步骤 2 故障导入在第 l–3 轮的 A_{l-3} 或 B_{l-3}，可知 B_{l-1} 中有 3 个字节的集合是平衡集，如表 4 所示，则有

$$\sum_{u=0}^{255} B_{l-1,j}^{(u)} = \sum_{u=0}^{255} (\mathrm{DL}^{-1}(\mathrm{SL}^{-1}(Y^{(u)} \oplus \mathrm{ek}_{l+1}^{(u)}) \oplus \mathrm{ek}_{l}^{(u)}))_j =$$
$$\sum_{u=0}^{255} (\mathrm{DL}^{-1}(\mathrm{SL}^{-1}(Y^{(u)} \oplus \mathrm{ek}_{l+1}))_j \oplus (\mathrm{DL}^{-1}(\mathrm{ek}_{l}))_j =$$
$$\sum_{u=0}^{255} (\mathrm{DL}^{-1}(\mathrm{SL}^{-1}(Y^{(u)} \oplus \mathrm{ek}_{l+1})))_j =$$
$$\sum_{u=0}^{255} ((\mathrm{SL}^{-1}(Y^{(u)} \oplus \mathrm{ek}_{l+1}))_{j0} \oplus (\mathrm{SL}^{-1}(Y^{(u)} \oplus \mathrm{ek}_{l+1}))_{j1}$$
$$\oplus \cdots \oplus (\mathrm{SL}^{-1}(Y^{(u)} \oplus \mathrm{ek}_{l+1}))_{j6})$$

其中，$B_{l-1,j}^{(u)}$、$Y^{(u)}$、$\mathrm{ek}_{l+1}^{(u)}$ 和 $\mathrm{ek}_{l}^{(u)}$ 表示第 u 次故障导入中的 $B_{l-1,j}$、Y、ek_{l+1} 和 ek_{l} 的值，j 表示 $B_{l-1}^{(u)}$ 的第 j 个字节，且 $j \in J$。如果 A_{l-3} 中有一个活跃字节，则 B_{l-1} 中有 3 个平衡字节。表 4 列出了 J 的所有字节位置。

表 4　活跃字节和相应的 3 个平衡字节的位置列

活动字节	3 个平衡字节的组合
0	{6, 9, 15}
1	{7, 8, 14}
2	{4, 11, 13}
3	{5, 10, 12}
4	{2, 11, 13}
5	{3, 10, 12}
6	{0, 9, 15}
7	{7, 8, 14}
8	{1, 7, 14}
9	{0, 6, 15}
10	{3, 5, 12}
11	{2, 4, 14}
12	{3, 5, 10}
13	{2, 4, 11}
14	{1, 7, 8}
15	{0, 6, 9}

此时，计算 D_{l-1} 可以转化为 7 个向量的异或，该步运算将数据复杂度由 2^{128} 降到 7×2^8。基于 2.5 轮的积分区分器，B_{l-1} 的第 j 个字集是平衡字集，因此有

$$\sum_{u=0}^{15} B_{l-1,j}^{(u)} = 0$$

其中，$j \in J$。

在步骤 3 中，当故障导入在第 l–4 轮的 A_{l-4} 或 B_{l-4} 时，可知 B_{l-2} 中有 3 个字节的集合是平衡集，并且

$$\sum_{u=0}^{255} B_{l-2,j}^{(u)} = \sum_{u=0}^{255} (\mathrm{DL}^{-1}(\mathrm{SL}^{-1}(\mathrm{DL}^{-1}(A_{l}^{(u)} \oplus \mathrm{ek}_{l}^{(u)}) \oplus \mathrm{ek}_{l-1}^{(u)})))_j =$$
$$\sum_{u=0}^{255} (\mathrm{DL}^{-1}(\mathrm{SL}^{-1}(\mathrm{DL}^{-1}(A_{l}^{(u)} \oplus \mathrm{ek}_{l}^{(u)}))))_j \oplus (\mathrm{DL}^{-1}(\mathrm{ek}_{l-1}^{(u)}))_j =$$
$$\sum_{u=0}^{255} (\mathrm{DL}^{-1}(\mathrm{SL}^{-1}(\mathrm{DL}^{-1}(A_{l}^{(u)} \oplus \mathrm{ek}_{l}^{(u)}))))_j =$$
$$\sum_{u=0}^{255} (\mathrm{DL}^{-1}(\mathrm{SL}^{-1}(A_{l}^{\prime(u)} \oplus \mathrm{ek}_{l}^{\prime})))_j =$$
$$\sum_{u=0}^{255} ((\mathrm{SL}^{-1}(A_{l}^{\prime(u)} \oplus \mathrm{ek}_{l+1}))_{j0} \oplus (\mathrm{SL}^{-1}(Y \oplus \mathrm{ek}_{l+1}))_{j1} \oplus \cdots \oplus$$
$$(\mathrm{SL}^{-1}(A_{l}^{\prime(u)} \oplus \mathrm{ek}_{l+1}))_{j6})$$

其中，$A_{l}^{\prime(u)} = \mathrm{DL}^{-1}(A_{l}^{(u)})$，$\mathrm{ek}_{l+1} = \mathrm{DL}^{-1}(\mathrm{ek}_{l+1})$。

此时，$j \in J$，j_0 到 j_6 的值如表 4 所示。同理有

$$\sum_{u=0}^{255} B_{l-3,j}^{(u)} = \sum_{u=0}^{255} B_{l-4,j}^{(u)} = 0$$

其中，$0 \leqslant j \leqslant 15$。攻击者可以依靠同一个密钥和不同明文得到的不同密文，利用穷尽搜索运算求出最后 4 轮子密钥。

5　理论时间复杂度

在上述分析过程中，为了计算平衡字节的积分和，攻击者至少需要一组密文，包括一个正确的密文和 255 个错误的密文。

设 η 表示一个错误的子密钥可以通过积分和检查的概率，θ 表示攻击过程中穷尽搜索的子密钥空间。当攻击者诱导 ι 个密文集合时，子密钥候选项中剩余的错误子密钥数为 $(\theta-1)\eta^{\iota}$。如果 $(\theta-1)\eta^{\iota}<1$，子密钥候选值集合中只有唯一值，即为正确的子密钥。

在 3 轮攻击方案中，存在

$$\begin{cases} \theta = 2^8 \\ \eta = 2^{-8} \end{cases}$$

如果 $\iota > 2$，那么攻击者可以计算出真正的子密钥。此时穷尽搜索一组密文的时间复杂度是

$$2^8 \times 2^8 \times 16 = 2^{20}$$

因而，破译 ARIA 算法的最少理论时间复杂度为

$$2 \times 4 \times 2^{20} = 2^{23}$$

在 4 轮攻击方案中，存在

$$\begin{cases} \theta = 7 \times 2^8 \\ \eta = 2^{-8} \end{cases}$$

如果 $\iota > 2$，则攻击者可以推导出真正的子密钥。此时穷尽搜索一组密文的时间复杂度是

$$2^8 \times 2^8 \times 16 \times 7 \approx 2^{23}$$

因而，恢复 ARIA 算法密钥的最少理论时间复杂度为

$$2 \times 4 \times 2^{23} = 2^{26}$$

6　实验结果分析

实验环境为 Inter Core I7-7700K@4.2GHz、内存为 32 GB 的计算机，使用 Java 语言编程进行 ARIA 算法的加解密和攻击操作。利用软件模拟故障产生，从而得到错误密文，本文共进行了 1 000 次实验，将这些实验平均分成了 5 组，记为 G_1、G_2、G_3、G_4 和 G_5。在积分故障分析过程中，攻击者至少需要一组密文，包含一个正确密文和 255 个错误密文。本节采用准确性、可靠性和耗费时间对实验结果进行了详细描述。

准确性是指候选密钥数与真实密钥数之间接近的程度。数值越接近，则实验结果就越准确。采用均方根误差（RMSE, root mean square error）计算式来进行计算。

$$\text{RMSE}=\sqrt{\frac{1}{m}\sum_{e=1}^{m}(\phi'(e)-\phi)}$$

其中，m 为实验次数，e 为第 e 次实验，$\phi'(e)$ 为在第 e 次实验中已恢复出的候选子密钥的比特数，ϕ 为真正子密钥的比特数。均方根值越接近于 0，则实验结果就越准确，候选子密钥的均方根值如表 5 与表 6 所示，其中 $m=200$，$\phi=128$、$e=\{1,\cdots,200\}$。实验结果表明，在 3 轮和 4 轮攻击方案中，分别最多仅需要 3 组和 16 组密文集合即可恢复子密钥。

可靠性是指成功实验在所有实验中的比例。如表 5 与表 6 所示，在 3 轮攻击方案中，子密钥恢复的平均比例分别为 0、94.8%和 100%。在四轮攻击

表 5　　**在 3 轮攻击方案中恢复子密钥的准确性和可靠性**

集合	准确性					可靠性				
	G_1	G_2	G_3	G_4	G_5	G_1	G_2	G_3	G_4	G_5
1	8.08	8.16	8.09	8.11	8.96	0	0	0	0	0
2	0.72	0.60	0.45	0.72	0.80	94.0%	96.0%	97.5%	93.5%	93.0%
3	0	0	0	0	0	100%	100%	100%	100%	100%

表 6　　**在 4 轮攻击方案中恢复子密钥的准确性和可靠性**

集合	准确性					可靠性				
	G_1	G_2	G_3	G_4	G_5	G_1	G_2	G_3	G_4	G_5
1	11.31	11.37	11.34	11.28	11.35	0	0	0	0	0
2	11.13	11.19	11.16	11.10	11.17	0	0	0	0	0
3	10.81	10.86	10.81	10.85	10.87	0	0	0	0	0
4	10.44	10.43	10.38	10.42	10.39	0	0	0	0	0
5	9.95	9.95	9.95	9.89	9.98	0	0	0	0	0
6	9.39	9.42	9.43	9.40	9.40	0	0	0	0	0
7	8.81	8.88	8.81	8.80	8.87	0	0	0	0	0
8	8.23	8.24	8.22	8.15	8.27	0	0	0	0	0
9	7.55	7.55	7.56	7.54	7.51	0	0	0	0	0
10	6.92	6.91	6.89	6.90	6.89	0	0	0	0	0
11	6.25	6.26	6.23	6.24	6.14	0	0	0	0	0
12	5.52	5.57	5.52	5.58	5.45	0	0	0	0	0
13	4.78	4.77	4.77	4.81	4.76	0	0	0	0	0
14	3.94	3.93	3.92	3.95	3.94	0	0	0	0	0
15	2.83	2.82	2.81	2.84	2.83	0	0	0	0	0
16	0	0	0	0	0	100%	100%	100%	100%	100%

方案中，子密钥恢复的平均比例分别为 0、0、…、0 和 100%。

耗费时间是指从故障导入到恢复子密钥的时间。图 4 显示了 1 000 次实验的耗时。在 3 轮攻击方案中，97.7%的耗费时间处于 5~10 s 之间，而在 4 轮攻击方案中，87%的耗费时间处于 30~40 s 之间。

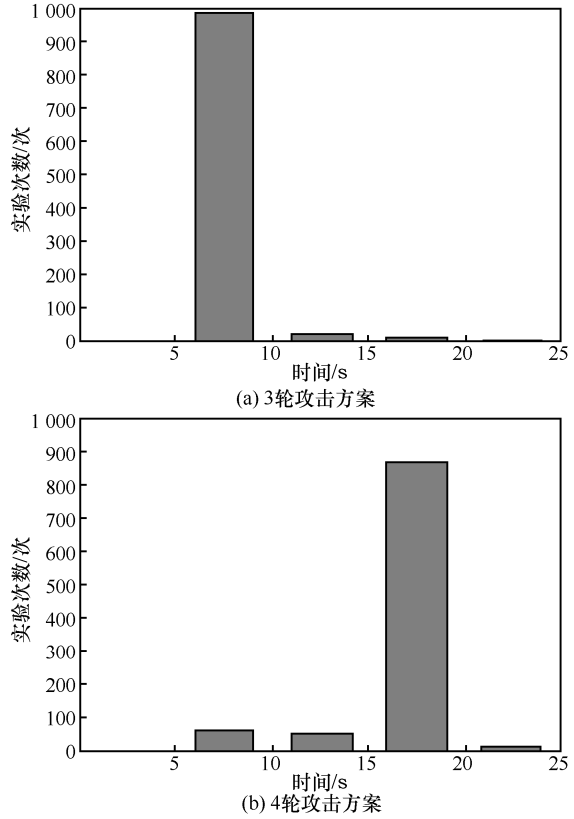


图 4 在攻击方案中恢复子密钥的耗费时间

因此，在 3 轮攻击和 4 轮攻击方案中，穷尽搜索一组密文的实际时间复杂度分别为

$$2^8 \times 2^8 \times 16 = 2^{20}$$

$$2^8 \times 2^8 \times 16 \times 7 \approx 2^{24}$$

则恢复 ARIA 原始密钥的最少实际时间复杂度分别为

$$3 \times 4 \times 2^{20} \approx 2^{24}$$

$$16 \times 4 \times 2^{24} \approx 2^{30}$$

7 结束语

本文提出了基于 ARIA 密码的积分故障分析方法。理论分析和实验结果表明，在面向单字节的故障模型中，分别可以构建 2 个积分区分器进行故障导入来破译 ARIA 算法。相较于其他故障分析方法，积分故障分析可以进一步深入算法内部更深轮数进行攻击，在攻击范围上均具有显著的优势。因此，以 ARIA 为代表的标准密码在现实环境中实现时必须加以软硬件防护措施进行保护，否则极易受到积分故障分析的威胁。

参考文献：

[1] ALIOTO M, SHAHGHASEMI M. The Internet of things on its edge: trends toward its tipping point[J] .IEEE Consumer Electronics Magazines, 2018, 7(1): 77-87.

[2] BAKER T, UGLJANIN E, FACI N, et al. Everything as a resource: foundations and illustration through Internet-of-things[J] .Computers in Industry, 2018, 94(1):62-74.

[3] KWON D, KIM J, PARK S, et al. New block cipher: ARIA[C]// International Conference of Information Security and Cryptology. 2003: 432-445.

[4] BIRYUKOV A, CANNIERE D C, LANO J, et al. Security and performance analysis of ARIA[J]. Internal Report, KU Leuven ESAT/SCD-COSIC, 2004: 1-55.

[5] LI P, SUN B, LI C. Integral cryptanalysis of ARIA[C]//International Conference of Information Security and Cryptology. 2009: 1-14.

[6] LIU Z, GU D, LIU Y, et al. Linear cryptanalysis of ARIA block cipher[C]//International Conference of Information and Communications Security, 2011: 242-254.

[7] LI Y, WU W, ZHANG L. Integral attacks on reduced-round ARIA block cipher[C]// International Conference of Information Security, Practice and Experience. 2010: 19-29.

[8] WU W, ZHANG W, FENG D. Impossible differential cryptanalysis of reduced–round ARIA and Camellia[J]. Journal of Computer Science and Technology, 2007, 22(3): 449-456.

[9] HESS E, JANSSEN N, MEYER B, et al. Information leakage attacks against smart card implementations of cryptographic algorithms and countermeasures–a survey[C]//International Conference on Research in Smart Cards.2000: 55-64.

[10] JOYE M, QUISQUATER J J, YEN S M, et al. Observability analysis-detecting when improved cryptosystems fail[C]//The Cryptographer's Track at the RSA Conference on Topics in Cryptology. 2002: 17-29.

[11] KELSEY J, SCHNEIER B, WAGNER D, et al. Side channel cryptanalysis of product ciphers[C]//European Symposium on Research in Computer Security. 1998: 97-110.

[12] LIN I C, CHANG C C. Security enhancement for digital signature schemes with fault tolerance in RSA[J]. Information Sciences, 2007, 177(19): 4031-4039.

[13] BIHAM E, SHAMIR A. Differential fault analysis of secret key cryptosystems[C]//Annual International Cryptology Conference. 1997: 513-525.

[14] BONEH D, DEMILLO R A, LIPTON R J. On the importance of checking cryptographic protocols for faults[C]//International Conference on Theory and Application of Cryptographic Techniques. 1997: 37-51.

[15] BONEH D, DEMILLO R A, LIPTON R J. On the importance of

eliminating errors in cryptographic computations[J].Journal of Cryptology, 2001, 14(2): 101-119.

[16] BIEHL I, MEYER B, MULLER V. Differential fault attacks on elliptic curve cryptosystems[C]//International Cryptology Conference on Advances in Cryptology. 2000: 131-146.

[17] FISCHER W, REUTER C A. Differential fault analysis on Grøstl[C]//Workshop on Fault Diagnosis and Tolerance in Cryptography. 2012: 44-54.

[18] HEMME L, HOFFMANN L. Differential fault analysis on the SHA1 compression function[C]//Workshop on Fault Diagnosis and Tolerance in Cryptography. 2011: 54-62.

[19] HOCH J J, SHAMIR A. Fault analysis of stream ciphers[C]// International Workshop on Cryptographic Hardware and Embedded Systems.2004: 240-253.

[20] LI W, GU D, LI J. Differential fault analysis on the ARIA algorithm[J]. Information Sciences, 2008, 178(19): 3727-3737.

[21] PARK J H, HA J C. Improved differential fault analysis on block cipher ARIA[C]//International Workshop on Information Security Applications.2012: 82-95.

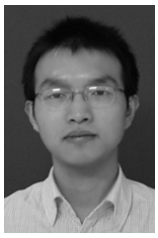
[22] KIM H C. Differential fault analysis of ARIA in multi-byte fault models[J]. Journal of Systems and Software, 2012, 85(9): 2096-2103.

[23] PHAN R C W, YEN M. Amplifying side-channel attacks with techniques from block cipher cryptanalysis[J]. International Conference on Smart Card Research and Advanced Applications, 2006: 135-150.

[24] DAEMEN J, KNUDSEN L R, RIJMEN V. The block cipher square[C]//International Workshop on Fast Software Encryption.1997: 149-165.

[25] LIDL R, NIEDERREITER H. Finite fields[M]. Cambridge: Cambridge University Press, 1997.

[作者简介]



沈煜（1980− ），男，湖南湘潭人，博士，东华大学助理研究员，主要研究方向为密码学、小波分析。

李玮（1980− ），女，安徽寿县人，博士，东华大学教授、博士生导师，主要研究方向为密码分析。

谷大武（1970− ），男，河南漯河人，博士，上海交通大学教授、博士生导师，主要研究方向为密码学与计算机安全。

吴益鑫（1995− ），女，浙江湖洲人，东华大学硕士生，主要研究方向为分组密码的安全性分析。

曹珊（1995− ），女，湖南株洲人，东华大学硕士生，主要研究方向为轻量级密码的安全性分析。

刘亚（1983− ），女，安徽安庆人，博士，上海理工大学副教授，主要研究方向为对称密码的安全性分析。

刘志强（1970− ），男，江西南昌人，博士，上海交通大学副研究员，主要研究方向为密码学与计算机安全。

周志洪（1981− ），男，上海人，博士，上海交通大学副总工程师，主要研究方向为物联网安全。