DBlock密码算法差分故障分析

李 浪[1,2,3] 邹 祎[1,2] 李株华[1,2] 刘波涛[1,2]

(衡阳师范学院计算机科学与技术学院 衡阳 421002)[1]

(智能信息处理与应用湖南省重点实验室 衡阳 421002)[2] (湖南大学信息科学与工程学院 长沙 410082)[3]

摘 要 DBlock算法是于2015年提出的一种新型分组密码算法,算法分组长度与对应密钥长度为128bit、192bit和256bit,均迭代20轮。基于字节故障模型,并利用基于密钥扩展的差分故障分析方法,在密钥扩展算法运行至第17轮时导入随机故障,对DBlock算法进行差分故障分析。实验结果表明,仅需要4次故障密文便可恢复算法的128bit初始密钥。

关键词 分组密码,DBlock,差分故障分析

中图法分类号 TP309 **文献标识码** A **DOI** 10.11896/j.issn.1002-137X.2017.07.022

Differential Fault Analysis on DBlock Cipher Algorithm

LI Lang[1,2,3] ZOU Yi[1,2] LI Zhu-hua[1,2] LIU Bo-tao[1,2]

(College of Computer Science and Technology, Hengyang Normal University, Hengyang 421002, China)[1]

(Hunan Provincial Key Laboratory of Intelligent Information Processing and Application, Hengyang 421002, China)[2]

(College of Information Science and Engineering, Hunan University, Changsha 410082, China)[3]

Abstract DBlock algorithm is a new type of block cipher algorithm proposed in 2015. The packet length and its corresponding key length are 128bit, 192bit and 256bit, with 20-round iterations of each. Based on byte fault model, the differential fault analysis on the DBlock was used on the key expansion to import a random fault in the 17th round. Experimental results show that the analysis can recover its primitive 128bit key only by introducing four fault ciphertexts.

Keywords Block cipher, DBlock, Differential fault analysis

1 引言

故障攻击是通过引入故障来破解密钥的方法。这一概念由Boneh等人于1996年提出,之后Biham与Shamir将差分分析和故障攻击相结合,提出了差分故障分析方法,并成功地对DES算法进行了分析[1]。利用差分故障分析密码的方法也在不断改进,可以分为对算法加密部分的诱导故障分析和对密钥扩展部分的诱导故障分析,如李瑞林利用单个故障数对SMS4密码算法的分析[2]、王素贞等对MIBS算法进行的宽度差分故障分析[3]、Moradi A等利用多字节故障模型对AES算法的故障分析[4]以及徐朋等采用半字节故障模型对TWINE算法的故障分析[5]属于第一类对算法加密部分的诱导故障分析。王高丽等在密钥扩展部分定向导入半字节的故障分析PRESENT算法[6]以及李玮等在改进的基于密钥扩展的方案下对SMS4的差分故障分析[7]属于第二类对密钥扩展部分的差分故障分析。

DBlock算法是由吴文玲等提出的一种分组密码算法[8],其结合了Feistel与type-2广义Feistel结构的优点。目前未见对DBlock算法差分故障攻击的相关文献。本文的方案在DBlock密钥扩展算法倒数第二轮导入随机字节故障,利用获得的故障密文同时对加密算法的最后两轮进行分析,最终恢复算法的初始密钥。

本文第2节介绍了DBlock算法;第3节概述了差分故障分析的基本思想与分析过程;第4节详细阐述了对DBlock算法差分故障分析的过程;第5节对复杂度与实验结果进行了分析;最后总结全文。

2 DBlock算法

DBlock算法有128bit、192bit与256bit 3种分组长度与密钥长度。本文以128bit的DBlock密码算法为代表来分析其安全性。DBlock加密部分与密钥扩展部分均采用Feistel结构,如图1所示。加密过程为:输入明文 $X=x_1 \| x_0$ (x_1, $x_0 \in F_2^{64}$),其中轮密钥 $k_i \in F_2^{64}$,算法的加密轮函数 $x_i = F_n(x_{i-1} \oplus k_{i-1}) \oplus x_{i-2}$ ($i=2,3,4,\cdots,21$),其中 $F=G(P(x_i))$。

到稿日期:2016-05-06 返修日期:2016-08-03 本文受国家自然科学基金资助项目(61572174),湖南省自然科学基金资助项目(2017JJ4001,2015JJ4011),湖南省教育厅资助科研项目(15A029,15C0203),衡阳师范学院产学研基金资助项目(16CXYZ01)资助。

李 浪(1971-),男,博士,教授,硕士生导师,CCF高级会员,主要研究领域为嵌入式计算与信息安全,E-mail:lilang911@126.com;邹 祎(1983-),女,硕士,讲师,主要研究领域为嵌入式计算与信息安全;李株华(1996-),主要研究领域为信息安全;刘波涛(1991-),男,硕士生,主要研究领域为信息安全。

在迭代 20 轮后，输出密文 $Y=x_{20}\parallel x_{21}$，其中 F 函数包括 P 置换、S 盒替换与线性函数 A。

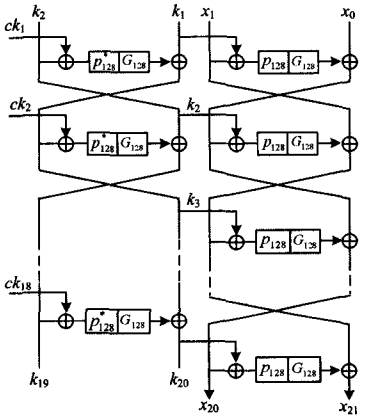

图 1　DBlock 密码算法的结构图

P 置换：将每轮输入的 128bit 的左半部分与轮密钥异或后分成 8 个子块$(y_7,y_6,y_5,y_4,y_3,y_2,y_1,y_0)$，置换后为$(z_7,z_6,z_5,z_4,z_3,z_2,z_1,z_0)$，过程如下：$z_7=y_6,z_6=y_5,z_5=y_3,z_4=y_1,z_3=y_4,z_2=y_7,z_1=y_0,z_0=y_2$。G 函数：将置换后的 8 个子块分成$(z_3,z_2,z_1,z_0)$与$(z_7,z_6,z_5,z_4)$两部分，其分别通过 S 盒替换与线性函数 A 后变为(u_3,u_2,u_1,u_0)与(u_7,u_6,u_5,u_4)，其线性函数 A 为 $x=A(x)=x\oplus(x\lll 8)\oplus(x\lll 10)\oplus(x\lll 18)\oplus(x\lll 26)$。

密钥扩展算法：DBlock 密码的密钥扩展算法与其加密算法一致，均采用 Feistel 结构。对于 128bit 的 DBlock 加密密钥，通过轮函数生成每轮加密子密钥：$k_i(i=1,2,\cdots,19,20)$，$k_i\in F_2^{64}$。$k_i=G(P^*(k_{i-1}\oplus ck_{i-2}))\oplus k_{i-2}(i=3,\cdots,20)$；其中 ck_{i-2} 为轮常数，生成过程如下：$ck_i=(a_{i,m},\cdots,a_{i,0})$，$m=n/16-1$；$a_{i,j}=(16i+j)\times 7 \bmod 256, i=1,2,\cdots,18, j=0,1,\cdots,m$。在子密钥生成过程中，除 P^* 置换与加密轮函数有所差异外，其余部分基本一致。

P^* 置换：将输入的 64bit 分成 8 个子块$(y_7,y_6,y_5,y_4,y_3,y_2,y_1,y_0)$，其置换后子块为$(z_7,z_6,z_5,z_4,z_3,z_2,z_1,z_0)$，过程如下：$z_7=y_1,z_6=y_0,z_5=y_7,z_4=y_6,z_3=y_5,z_2=y_4,z_1=y_2,z_0=y_3$。

3　DBlock 差分故障分析

3.1　差分故障模型与基本假设

DBlock 算法采用 8×8 的 S 盒，本文运用的故障模型是面向字节的随机故障模型，模型的基本假设为：

(1)攻击者可以对算法密钥扩展运行过程中某时刻的存储单元诱导单字节故障错误，但是其本身并不知道故障发生的存储单元位置以及具体的错误值。

(2)对于同一个明文，攻击者可以获得在某一密钥下加密的正确密文和错误密文。

3.2　分析的基本过程

步骤 1　任意选择明文，在某一密钥的加密作用下产生对应的正确密文。

步骤 2　恢复最后一轮密钥时，在算法的密钥扩展运算的倒数第二轮中导入随机故障进行诱导，再通过算法的加密过程获得相对应的错误密文；利用算法的性质，将正确密文与错误密文进行差分分析，恢复出这一轮子密钥的部分字节数据；重复该运算过程，直到全部恢复出这一轮的子密钥。

步骤 3　恢复倒数第二轮中密钥时，依然在密钥扩展算法的倒数第二轮中导入随机故障进行诱导，再通过算法的加密过程获得相对应的错误密文。利用步骤 2 中已经获得的最后一轮子密码，将算法最后一轮进行解密运算，由解密获得一个正确的中间值与一个错误的中间值；利用算法的性质，将正确的中间值与错误的中间值进行差分分析，恢复出这一轮子密钥的部分字节数据；重复该运算过程，直到全部恢复出这一轮的子密钥。

步骤 4　利用已经恢复的倒数第二轮与倒数第一轮的子密钥进行算法的密钥扩展逆向运算，恢复出算法中每一轮的子密钥以及原始密钥数据。

4　DBlock 差分故障分析过程详述

4.1　符号说明

ΔA_i 表示在加密算法第 i 轮中 P 置换的输出差分；ΔB_i 表示加密算法中第 i 轮线性函数变换 A 的输入差分；ΔC_i 表示加密算法中第 i 轮线性函数变换 A 的输出差分；$x_i\parallel x_{i+1}$ 表示加密算法中第 i 轮密文；$x_i^*\parallel x_{i+1}^*$ 表示加密算法中第 i 轮故障密文。

4.2　分析过程

(1)在 DBlock 算法的加密过程中，任意选择一组明文 X，在初始密钥 k 下，通过加密算法产生对应的密文 Y。

(2)分析最后一轮，步骤如下：

1)明文 X 在初始密钥 k 的作用下生成密文 $Y=x_{20}\parallel x_{21}$。算法密钥扩展运行至第 17 轮运算时，在其左寄存单元 k_{18} 中导入一个字节的随机故障，由此得到故障密文 $Y^*=x_{20}^*\parallel x_{21}^*$，如图 2 所示。同时，$\Delta x_{20}=x_{20}\oplus x_{20}^*\neq 0$，$\Delta k_{20}=k_{20}\oplus k_{20}^*\neq 0$，$\Delta x_{21}=x_{21}\oplus x_{21}^*\neq 0$。

2)最后一轮中线性函数变换 A 的输出差分为 $\Delta C_{20}=\Delta x_{21}\oplus\Delta x_{19}$，输入差分为 $\Delta B_{20}=\Delta C\cdot A^{-1}$，P 置换的输出差分 $\Delta A_{20}=(\Delta x_{20}\oplus\Delta k_{20})\cdot P$，$A^{-1}$ 记为线性函数 A 的逆运算。

3)根据 S 盒的差分分析模型 $s(x)\oplus s(x\oplus\alpha)=\beta(\alpha,\beta\in F_2^8)$，其中 α 为 S 盒的输入差分，β 为输出差分，定义：$IN_s(\alpha,\beta)=\{x\mid S(x)\oplus S(x\oplus\alpha)=\beta,x\in F_2^8\}$。利用最后一轮中的 S 盒的输入差分 ΔA_{20} 与输出差分 ΔB_{20} 进行计算，可获得 k_{20} 部分字节候选值为 $\{x_{20,j}\oplus e_j:e_j\in IN_s(\Delta A_{20,j},\Delta B_{20,j})\mid e_j\in F_2^8\}$，$0\leqslant j<8$，其中 $IN_s(\Delta A_{20,j},\Delta B_{20,j})=\{x\mid S(x)\oplus S(x\oplus\Delta A_{20,j})=\Delta B_{20,j},x\in F_2^8\}$。重复上述步骤，多次导入随机字节故障，直至候选值唯一时，分析出 k_{20} 的全部字节。

(3)分析倒数第二轮，步骤如下：

1)利用已经确定的最后一轮密钥解密获得第 19 轮加密运算输出的正确密文 $x_{19}\parallel x_{20}$。在密钥扩展第 17 轮中导入字节故障后，解密获得输出的故障密文 $x_{19}^*\parallel x_{20}^*$。

2)如图 2 所示，在 k_{18} 导入字节故障后，可知 $k_{19}^*=G(P^*(k_{18}^*\oplus ck_{17}))\oplus k_{17}$，经过 P^* 置换后一个字节状态发生改变，输入 G 函数中的 S 盒替换和线性函数 A 运算后 4 个字节状态发生改变。通过分析得到在 k_{18} 导入字节故障位置与轮密钥 k_{19} 字节状态发生改变的位置的关系，如表 1 所列。在 $k_{18,0}$，$k_{18,1}$，$k_{18,6}$ 与 $k_{18,7}$ 处导入的字节故障只能对轮密钥 k_{19} 的

左半部分字节的生成产生影响，而余下位置的字节故障只对轮密钥 k_{19} 的右半部分的字节有影响。

表 1 第 17 轮字节故障与故障密钥的关系

k_{18} 故障位置	k_{19} 故障位置
$k_{18,0}$	$k^*_{19,7}k^*_{19,6}k^*_{19,5}k^*_{19,4}$
$k_{18,1}$	$k^*_{19,7}k^*_{19,6}k^*_{19,5}k^*_{19,4}$
$k_{18,2}$	$k^*_{19,3}k^*_{19,2}k^*_{19,1}k^*_{19,0}$
$k_{18,3}$	$k^*_{19,3}k^*_{19,2}k^*_{19,1}k^*_{19,0}$
$k_{18,4}$	$k^*_{19,3}k^*_{19,2}k^*_{19,1}k^*_{19,0}$
$k_{18,5}$	$k^*_{19,3}k^*_{19,2}k^*_{19,1}k^*_{19,0}$
$k_{18,6}$	$k^*_{19,7}k^*_{19,6}k^*_{19,5}k^*_{19,4}$
$k_{18,7}$	$k^*_{19,7}k^*_{19,6}k^*_{19,5}k^*_{19,4}$

3)分析第 19 轮的轮密钥的过程与分析最后一轮的轮密钥的方法相似。利用对应 S 盒的输入差分 ΔA_{19} 与输出差分 ΔB_{19} 进行计算，可获得 k_{19} 部分字节候选值 $\{x_{20,j}\oplus e_j : e_j \in IN_s(\Delta A_{19,j}, \Delta B_{19,j}) \mid e_j \in F_2^8\}$，$0 \leqslant j < 8$，其中 $IN_s(\Delta A_{19,j}, \Delta B_{19,j}) = \{x \mid S(x)\oplus S(x\oplus \Delta A_{19,j}) = \Delta B_{19,j}, x \in F_2^8\}$。同理可以分析出 k_{19} 的全部字节。

(4)利用已恢复的轮密钥 k_{19} 与 k_{20}，根据密钥扩展算法可以推导出初始密钥 k。

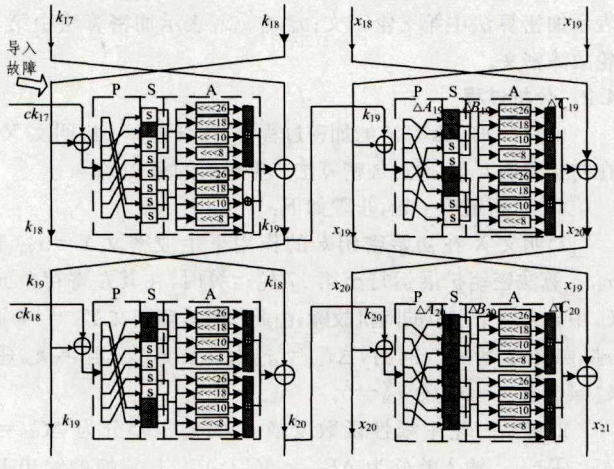

图 2 第 19 轮和第 20 轮故障分析示意图

5 实验结果分析

5.1 复杂度分析

由 S 盒的差分分析模型 $s(x)\oplus s(x\oplus\alpha)=\beta(\alpha,\beta\in F_2^8)$ 可以统计出 DBlock 算法中 S 盒差分分布性质的情况，如表 2 所列。

表 2 S 盒差分分布性质表

$Ns(\alpha,\beta)$ 的数值	出现次数	出现概率	$Ns(\alpha,\beta)>0$ 的概率	均值
0	33150	0.50583		
2	32130	0.49026	0.99210	1.9842
4	255	0.00389	0.00787	0.0315
256	1	0.00002	0.00003	0.00768

根据表 2 可知，方程的解的期望值为 $2.02336(1.9842+0.0315+0.00768)\approx 2^{1.01675}$，因此导入一个字节故障后，其对应的密钥候选值的个数为 2.02336，则一个字节的密钥搜索空间由 2^8 降低到 $2^{1.01675}$，恢复一个字节密钥大约只需要两个字节故障数。由第 4 节可知，如果字节故障被导入到寄存单元 k_{18} 的 $k_{18,0}$，$k_{18,1}$，$k_{18,6}$ 与 $k_{18,7}$ 中，经过 P^* 置换与线性函数 A 后，k_{19} 的右 4 个字节的状态发生变化。当字节故障导入到寄存单元 k_{18} 的其余位置时，k_{19} 的左 4 个字节状态将发生变化。因此，一次字节故障将导致加密过程中 P 置换的输出差分与线性函数 A 输入差分有 4 个字节的状态变化。若要恢复 k_{19} 的全部字节，则需要在寄存单元 k_{18} 的两类位置处各导入 2 个字节故障。在寄存单元 k_{18} 中，任意导入的字节故障经过两轮密钥扩展算法运算后将导致 k_{20} 全部字节的状态发生改变。因此，若要恢复密钥 k_{20}，则需在寄存单元 k_{18} 任意导入 2 个字节故障。理论上，恢复 DBlock 算法的全部密钥大约需要 4 个字节故障数，分析的复杂度为 $2^{11}(2^8\times 2^1\times 2^2)$。

5.2 实验结果分析

在 PC 机(配置：CPU 为 Intel(R)Core(TM) 2.4GHz，内存为 4GB)上使用 C++编程实现上述分析方法，进行 50 次实验，选取其中的 10 次，结果如表 3 所列。实验结果表明，在最好的情况下，仅需要 4 个故障密文便可在该方案下恢复 128bit 初始密钥。表 4 列出了实际差分故障分析的一组实验结果数据。

表 3 DBlock 算法的故障分析实验结果

序列号	分析 k_{19} 所需的故障密文数	分析 k_{20} 所需的故障密文数
1	4	2
2	6	2
3	4	2
4	10	2
5	4	2
6	6	2
7	5	2
8	4	2
9	8	2
10	5	2

表 4 DBlock 算法差分故障分析的一组实验结果数据

明文	00112233445566778899AABBCCDDEEFF
初始密钥	00112233445566778899AABBCCDDEEFF
正确密文	D8EB9A38335E282E465C33F979365B3E
故障密文 1	1C1B59073DB66BC8BCFF9E6295721B85
故障密文 2	E835F78B9CA12AD3E6CADA0C7E94AB38
故障密文 3	D1899F5F087CBA9E7A5123A7181F83C8
故障密文 4	890EA2E504ABD6259A9D355183B6F67E
故障密文 5	179DADC926EEE751DDAF727889044B44
故障密文 6	566C8D3A3E5C482103DEFF8FEFD46FDC
恢复密钥 k_{19}	CE57B31B04EC0C76
恢复密钥 k_{20}	DB5DAB50502D166E

结束语 我们对 DBlock 算法进行了差分故障分析，实验结果表明，在该方案中导入的字节故障有较好的利用率，在最好的情况下，仅需要 4 个故障密文便可恢复出初始密钥。后续工作是将算法组成部分的特性与差分故障分析相结合，通过定向导入故障，提高故障的利用率与对密码算法分析的效率。

参 考 文 献

[1] BIHAM E, SHAMIR A. Differential fault analysis of secret key cryptosystems[C]// Proc of Advances in Cryptology-Crpto'97. 1997: 513-525.

[2] LI R L, SUN B, LI C, et al. Differential fault analysis on SMS4 using a single fault[J]. Information Processing Letters, 2011, 111(4): 156-163.

[3] WANG S Z, ZHAO X J, WANG T, et al. Wide Differentail Fault

Analysis on MIBS[J]. Computer Science, 2011, 38(4): 122-124. (in Chinese)
王素贞，赵新杰，王韬，等. 针对 MIBS 的宽度差分故障分析[J]. 计算机科学，2011，38(4)：122-124.

[4] MORADI A, SHALMANI M T M, SALMASIZDEH M. A generalized method of differential fault attack against AES cryptosystem[C]// Proc of Cryptographic Hardware and Embedded System 2006. 2006: 91-100.

[5] XU P, WEI Y C, PAN X Z. Differential fault attack on TWINE [J]. Application Research of Computer, 2015, 32(6): 1796-1800. (in Chinese)
徐朋，魏悦川，潘晓中. 轻量级分组密码 TWINE 的差分故障攻击[J]. 计算机应用研究，2015，32(6)：1796-1800.

[6] WANG G I, WANG S H. Differential Fault Analysis on PRESENT Key Schedule[C]// Proc of the 2010 International Conference on Computational Intelligence and Security. 2010: 362-366.

[7] LI W. A Improved Method of Differential Fault Analysis on SMS4 key Schedule[C]// Proc of the 2010 2nd International Conference on Future Computer and Communication. 2010: 95-99.

[8] WU W L, ZHANG L, YU X L. The DBlock family of block ciphers[J]. Science China Information Sciences, 2015, 58(3): 1-14.

(上接第 110 页)

[4] ZHENG X, YU J, SHUAI Y. A novel authentication scheme based on chaos[C]// 2013 8th International Conference on Computer Science & Education. 2013: 879-882.

[5] XIAO D, LIAO X, DENG S. A novel key agreement protocol based on chaotic maps[J]. Information Sciences, 2007, 177(4): 1136-1142.

[6] USAMA M, KHAN M K, ALGHATHBAR K, et al. Chaos-based secure satellite imagery cryptosystem[J]. Computers & Mathematics with Applications, 2010, 60(2): 326-337.

[7] LEUNG H Y, CHENG L M, CHENG L L. Robust watermarking schemes using selective curvelet coefficients based on a hvsmodel[J]. International Journal of Wavelets, Multiresolution and Information Processing, 2010, 8(6): 941-959.

[8] XIAO D, LIAO X, WANG Y. Parallel keyed hash function construction based on chaotic neural network[J]. Neurocomputing, 2009, 72(10): 2288-2296.

[9] GUO X, ZHANG J. Secure group key agreement protocol based on chaotic Hash[J]. Information Sciences, 2010, 180(20): 4069-4074.

[10] ZHAO G, FANG J Q. Modern information safety and advances in application research of chaos-based security communication [J]. Progress in Physics, 2003, 23(2): 212-255. (in Chinese)
赵耿，方锦清. 现代信息安全与混沌保密通信应用研究的进展[J]. 物理学进展，2003，23(2)：212-255.

[11] JAKIMOSKI G, KOCAREV L. Chaos and cryptography: block encryption ciphers based on chaotic maps[J]. IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications, 2001, 48(2): 163-169.

[12] WEBSTER A F, TAVARES S E. On the design of S-boxes[M]// Advances in Cryptology—CRYPTO'85 Proceedings. Springer Berlin Heidelberg, 1985: 523-534.

[13] YI X, CHENG S X, YOU X H, et al. A method for obtaining cryptographically strong 8×8 S-boxes[C]//Global Telecommunications Conference, 1997(GLOBECOM'97). IEEE, 1997: 689-693.

[14] TANG G, LIAO X, CHEN Y. A novel method for designing S-boxes based on chaotic maps[J]. Chaos, Solitons & Fractals, 2005, 23(2): 413-419.

[15] KOHDA T, TSUNEDA A. Statistics of chaotic binary sequences [J]. IEEE Transactions on Information Theory, 1997, 43(1): 104-112.

[16] CHEN G. A novel heuristic method for obtaining S-boxes[J]. Chaos, Solitons & Fractals, 2008, 36(4): 1028-1036.

[17] HUSSAIN I, SHAH T, GONDAL M A, et al. A novel method for designing nonlinear component for block cipher based on TD-ERCS chaotic sequence[J]. Nonlinear Dynamics, 2013, 73(1/2): 633-637.

[18] KHAN M, SHAH T, MAHMOOD H, et al. A novel technique for the construction of strong S-boxes based on chaotic Lorenz systems[J]. Nonlinear Dynamics, 2012, 70(3): 2303-2311.

[19] HUSSAIN I, SHAH T, MAHMOOD H, et al. A projective general linear group based algorithm for the construction of substitution box for block ciphers[J]. Neural Computing and Applications, 2013, 22(6): 1085-1093.

[20] ÖZKAYNAK F, YAVUZ S. Designing chaotic S-boxes based on time-delay chaotic system[J]. Nonlinear Dynamics, 2013, 74(3): 551-557.

[21] GUESMI R, AMINE BEN FARAH M, KACHOURI A, et al. Chaos-based designing of a highly nonlinear S-box using Boolean functions[C]// 2015 12th International Multi-Conference on Systems, Signals & Devices (SSD). IEEE, 2015: 1-5.

[22] TIAN Y, LU Z. S-box: LL Cascade Chaotic Map and Line Map [M]// Image and Graphics. Springer International Publishing, 2015: 297-309.

[23] FENG Y, LI L, HUANG F. A symmetric image encryption approach based on line maps[C]// 1st International Symposium on Systems and Control in Aerospace and Astronautics, 2006 (ISSCAA 2006). IEEE, 2006: 1362-1367.

[24] QIN J, WANG P. A method to construct Dynamic S-Box based on Chaotic Map[J]. Computer Science, 2007, 34(5): 89-91. (in Chinese)
邱劲，王平. 基于混沌映射的动态 S 盒构造方法[J]. 计算机科学，2007，34(5)：89-91.

[25] HASSANI M. Derangements and applications[J]. Journal of Integer Sequences, 2003, 6(2): 1-8.

[26] LIU Y, TIAN S. Design and statistical analysis of a new chaos block cipher for WSN[C]// 2010 IEEE International Conference on Information Theory and Information Security. 2010: 327-330.

[27] BENJEDDOU A, TAHA A, FOURNIER-PRUNARET D, et al. A new cryptographic hash function based on chaotic S-Box[C]// CSNDSP. Austria, 2008: 23-25.

[28] BLANCHARD P, DEVANEY R L, HALL G R. Differential Equations[M]. London: Thompson, 2006: 96-111.