一种抗错误注入攻击的S盒的构造

柴进晋,吴 暄
(空军工程大学防空反导学院,陕西西安 710051)

摘 要: 分组密码是现代密码学的重要组成部分,而S盒又是分组密码中必不可少的非线性组件,为密码算法提供了很好的混淆作用. 无论是传统分组密码中的S盒还是轻量级分组密码中的S盒都非常容易受到错误注入攻击. 本文通过具有线性或非线性邻域函数的元胞自动机设计了一种可以检测两个字节错误并纠正一个字节错误的S盒,以抵抗错误注入攻击. 对比 Advanced Encryption Standard(AES)中的S盒,虽然密码性能有所下降,但是可以抵抗错误注入攻击. 并且,本文还考虑了回旋镖均匀度这个密码安全性指标,用于衡量S盒抗回旋镖攻击的能力.

关键词: S盒;元胞自动机;错误注入攻击;分组密码
基金项目: 国家自然科学基金(No.62201612)
中图分类号: TP309 **文献标识码:** A **文章编号:** 0372-2112(2023)12-3422-09
电子学报URL:http://www.ejournal.org.cn **DOI:**10.12263/DZXB.20221111

Construction of Fault Injection Attacks Resistant S-Boxes

CHAI Jin-jin, WU Xuan
(*Institute of Air Defense and Anti-missile, Air Force Engineering University, Xi' an, Shaanxi* 710051, *China*)

Abstract: Block cipher is an important part of modern cryptography, and S-box is an essential nonlinear component in block cipher, which provides a good confusion for cryptographic algorithms. Both S-boxes in traditional block ciphers and S-boxes in lightweight block ciphers are vulnerable to fault injection attacks. In this paper, we design some S-boxes that can detect two-byte errors and correct one-byte error based on cellular automata with linear or nonlinear neighborhood functions to resist fault injection attacks. Compared with the S-boxes in Advanced Encryption Standard (AES), although the cryptographic performance has decreased, they can resist fault injection attacks. In addition, this paper considers the security index of boomerang uniformity, which is used to measure the ability of S-box to resist boomerang attacks.

Key words: S-box; cellular automata; fault injection attacks; block cipher
Foundation Item(s): National Natural Science Foundation of China (No.62201612)

1 引言

在1997年欧密会上,Boneh等人[1]首次提出了错误注入攻击的概念,即把错误注入到加密设备的操作中,导致加密设备产生错误的结果,并对错误结果进行分析以获得密钥. 由于在攻击过程中使用了错误的密文,错误攻击会使加密设备运行错误,而不是停止工作. 在攻击期间注入的错误可分为永久错误和临时错误. 永久错误将导致加密设备在执行加密操作时总是输出错误的结果. 这种类型的错误是不可恢复的,并且由于对加密设备的损坏, 这种错误很容易被发现. 临时错误会导致加密设备在错误注入期间发生错误. 错误注入停止后,加密设备将恢复正常操作. 这类错误不会对加密设备造成损害,而且在发生错误注入攻击后也不容易找到.

为了抵抗错误注入攻击,研究人员做了大量的研究[2~4]. 这些对策分为错误检测[5~7]和错误感染[8,9]两种方式. 错误检测是指如果在系统中检测到错误,加密系统将不再输出错误的密文,攻击者将无法使用该密文获取该密钥. 该策略需要一定数量的冗余计算,这增加了系统的存储开销,检测到错误后设备停止工作. 错误感染对策是消除错误密文与密钥之间的依赖关系,使攻击者无法通过密文来分析密钥. 这种对策通常针对特定的应用场景,没有通用的解决方案. 综上所述,无论是错误检测还是错误感染,其目的都是使攻击者无法从密文中获取密钥,而不是考虑纠正错误. S盒是分组密码中必不可少的非线性组件,通常决定了密码算

收稿日期:2022-09-30;修回日期:2023-05-21;责任编辑:李勇锋

法的安全强度. 在S盒的设计方法中,除了传统的代数方法构造之外,采用智能化搜索算法搜索设计也是目前研究热点之一,尤其是随着人工智能的快速发展,这种智能化搜索构造方法越来越受到学者们的关注. 目前,遗传算法、元胞自动机、神经网络等智能算法已经应用于S盒的构造中. Picek等人使用遗传算法分别搜索到含有较小透明阶值的布尔函数[10],4×4 S盒[11]、8×8 S盒[12];Kumar等人[13]提出一种基于可逆元胞自动机规则的AES中S盒的设计方法,降低了实现成本;Picek等人[14]利用遗传规则演化了大量元胞自动机规则,构造了具有良好密码性质的S盒;Ghoshal等人[15]利用元胞自动机规则构造的S盒降低了基于门限实现占用的芯片面积和功耗;关杰等人[16]基于元胞自动机找到一类可以代替Keccak杂凑函数中的新S盒;黄俊君等人[17]研究了基于元胞自动机的S盒的神经网络实现方法,具有更简单的结构盒更小的资源开销. 本文中基于元胞自动机设计出一类S盒,所设计的S盒可以检测错误及纠正错误,出现错误后密码设备可以继续正常工作. 在文献[14,15,18]中分别基于元胞自动机设计了3×3、4×4、5×5的S盒,但这些S盒都没有纠错功能. 文献[13]基于元胞自动机设计了8×8的S盒,可以检错和纠错. 本文利用具有线性或非线性邻域函数的元胞自动机设计出的8×8的S盒,可以检查两个字节的错误并纠正一个字节的错误,相比文献[19]中的设计方法可以为S盒的设计提供参考. 本文设计的方法在运算效率上优于文献[19],所设计的S盒在抵抗差分攻击能力方面优于文献[19]中的S盒. 此外,本文对所设计S盒的抗回旋镖攻击能力进行分析.

2　元胞自动机

元胞自动机[20,21](Cellular Automata,CA)可以看作是有限域内空间、状态和时间上都离散的动态系统. 元胞是CA中最基本的单位. 它们通常是具有相同结构和属性的元素和点. 元胞可以被视为存储介质,其状态为其存储内容. 最简单的CA有两种状态:0和1. 一个元胞的状态只与这个元胞相邻的元胞有关,而与其它元胞无关. 在特定的规则下,元胞状态都会随着相邻元胞的变化而同步更新. 经过一定数量的状态转换后,元胞间会形成复杂的连接,因为CA中的每个元胞都与相邻元胞之间存在计算相关性. 因此,CA虽然具有简单的结构,但可以模拟复杂的演化过程,可以很好地应用于密码学.

元胞自动机可以用如下的四元组来表示:

$$\mathrm{CA}=(A,S_d,\phi,B)$$

(1) 元胞空间 $A=N_1\times N_2\times\cdots\times N_d$ 表示第 i 维空间上的元胞数为 N_i. 元胞的位置可以用一个 d-元组 $i=(i_1\times i_2\times\cdots\times i_d)$ 来表示,并且元胞 i 的邻域被定义为 $\{j=(j_1\times j_2\times\cdots\times j_d):|j_1-i_1|<r_1,|j_2-i_2|<r_2,\cdots,|j_d-i_d|<r_d\}$,向量 $r=(r_1\times r_2\times\cdots\times r_d)$ 称为邻域半径. 本文仅考虑有3个邻域的CA,如图1所示.

图1　3邻域7元胞的CA

(2) S_d 是元胞在时刻 l 的配置,也就是CA状态空间,它是所有元胞状态的集合. 由于CA中每个元胞都是等价的,所以它们都在相同的状态空间 S_d 中取值,即状态空间指定了特定CA中每个元胞的取值范围. 由于CA的状态和时间都具有离散性,元胞在不同时刻的状态迁移由元胞的状态转换规则决定. 本文中状态空间 $S_d=\{0,1\}$.

(3) ϕ 是状态转移规则. 一个3邻域 n 个元胞的CA,它的第 i 个元胞的下一时刻的状态为

$$S_i^{l+1}=\phi_i\left(S_{i-1}^l,S_i^l,S_{i+1}^l\right)\tag{1}$$

式中,S_i^l 表示第 i 个元胞在时刻 l 的状态,S_i^{l+1} 表示第 i 个元胞的下一状态也就是时刻 $l+1$ 时的状态,ϕ_i 是第 i 个元胞的状态转换规则. 元胞的状态转换规则可以用二进制 $a_7a_6a_5a_4a_3a_2a_1a_0$ 来表示,对应的十进制是 $R=\sum_{i=0}^{7}a_i2^i$,称为规则 R. 例如 $(01011010)_2=(90)_{10}$,我们称之为规则90. 因此可以计算得出元胞的状态转移规则一共有 $2^8=256$ 种. 更多关于元胞的状态转换规则可参考文献[22].

(4) B 是CA的边界类型. 对于一个CA,有两种边界类型:无限CA和有限CA. 分类依据是CA中的元胞数量. 一般来说,无限CA只是理论概念,不能应用于实际应用. 在本文中,仅考虑有限CA,并取 $S_{-1}=S_n=0$.

如果一个规则函数只包含一个布尔XOR操作,那么它就被称为线性规则,例如规则60、90、102、150等都是线性规则;如果一个规则函数包含AND或OR操作,那么它被称为非线性规则,例如规则30和规则210都是非线性规则;如果所有元胞都遵循相同的规则,则CA称为均匀CA,否则,它被称为非均匀或混合CA;一个所有元胞都遵循线性规则的CA被称为线性CA(LCA);如果其中某些元胞遵循非线性规则的CA被称为非线性CA(NLCA). CA对应的规则称为该CA的规则向量. 假设第 i 个元胞的状态转换规则分别是30、90、150、86,那么规则30、90、150、86的表达式分别如下:

规则30:$\phi_i=S_{i-1}\oplus S_i\oplus S_{i+1}\oplus\left(S_i\cdot S_{i+1}\right)$

规则90:$\phi_i=S_{i-1}\oplus S_{i+1}$

规则150：$\phi_i=S_{i-1}\oplus S_i\oplus S_{i+1}$

规则86：$\phi_i=S_{i-1}\oplus S_i\oplus S_{i+1}\oplus\left(S_{i-1}\cdot S_i\right)$

将上述规则以特征矩阵[23]的形式来表示．例如，具有规则向量(90,90,90,150,150)的5元胞CA的特征矩阵，表示为

$$\boldsymbol{T}=\begin{bmatrix}0&1&0&0&0\\1&0&1&0&0\\0&1&0&1&0\\0&0&1&1&1\\0&0&0&1&1\end{bmatrix} \tag{2}$$

$\boldsymbol{T}(S)$定义为

$$\boldsymbol{T}(S)=\begin{bmatrix}0&1&0&0&0\\1&0&1&0&0\\0&1&0&1&0\\0&0&1&1&1\\0&0&0&1&1\end{bmatrix}\begin{bmatrix}S_0\\S_1\\S_2\\S_3\\S_4\end{bmatrix}=\begin{bmatrix}S_1\\S_0\oplus S_2\\S_1\oplus S_3\\S_2\oplus S_3\oplus S_4\\S_3\oplus S_4\end{bmatrix} \tag{3}$$

3 S盒的安全性指标

一个n输入m输出的S盒定义为从$\mathbb{F}_2^n$到$\mathbb{F}_2^m$的函数：$F(x)=\left(f_1(x),f_2(x),\cdots,f_m(x)\right)$．分量函数$f_i$，$1\leqslant i\leqslant m$，是布尔函数．显然，当$m=1$时，$F(x)$是单输出布尔函数．S盒也被称为向量布尔函数，或多输出布尔函数．文中将S盒称为(n,m)函数．

一个(n,m)函数的代数标准型定义为

$$F(x)=\sum_{I\in\rho(N)}a_I\left(\prod_{i\in I}x_i\right)=\sum_{I\in\rho(N)}a_Ix^I \tag{4}$$

式中，$\rho(N)$表示集合$N=\{1,2,\cdots,n\}$的幂集，且a_I属于F_2^m．

(n,m)函数的Walsh-Hadamard变换$W_F(\boldsymbol{u},v)$是从F_2^n到$\mathbb{C}$的函数，$W_F(\boldsymbol{u},v)$把任意有序对$(\boldsymbol{u},v)\in F_2^n\times F_2^{m*}$映射到组合函数$v\cdot F(x)$在点$u$处的Walsh-Hadamard变换，即

$$W_F(\boldsymbol{u},v)=\sum_{x\in F_2^n}(-1)^{v\cdot F(x)+\boldsymbol{u}\cdot\boldsymbol{x}} \tag{5}$$

式中，$\boldsymbol{u}\cdot\boldsymbol{x}$表示向量$\boldsymbol{u}$和向量$\boldsymbol{x}$的内积．显然，一个$(n,m)$函数的Walsh变换是各个分量函数的所有非零线性组合的Walsh-Hadamard变换．

3.1 平衡性

S盒在Feistel密码中使用时应该是平衡的，在SPN密码中使用时应该是双射的．同时具有单射和满射的映射称为双射．

定理1 设$F(x)$是(n,m)函数，其中$1\leqslant m\leqslant n$，函数$F(x)$平衡时Walsh谱满足：

$$W_F(0,v)=0,v\in F_2^{n*} \tag{6}$$

3.2 非线性度

1993年，Matsui[24]提出了线性密码分析方法．线性密码分析是一种已知明文攻击．线性分析的基本思想是研究S盒的输入位和输出位，并利用两者之间可能的线性关系来找出S盒的特殊性．然后利用这种特殊性对整个密码算法进行攻击．S盒的非线性衡量S盒抵抗线性攻击的重要指标．非线性度越高，S盒的抗线性分析能力越强．

定义1 设$F(x)$是一个(n,m)函数，$1\leqslant m\leqslant n$，它的非线性度为

$$\begin{aligned}\mathrm{NL}(F)&=\min_{l(\boldsymbol{x})\in A_n}d_H(v\cdot F(x),l)\\&=2^{n-1}-\frac{1}{2}\max|W_{F(x)}(\boldsymbol{u},v)|\end{aligned} \tag{7}$$

式中，$v\in F_2^{n*}$，A_n是F_2上所有仿射函数的集合．即非线性度是仿射函数与组合函数$v\cdot F(x)$之间的最小汉明距离．对于一个(n,m)函数，非线性度满足：

$$\mathrm{NL}(F)\leqslant 2^{n-1}-2^{\frac{n}{2}-1} \tag{8}$$

3.3 差分均匀度

1991年，Biham和Shamir[25]提出一种基于差分密码的分析方法．这种分析方法对分组密码构成严重威胁．使用差分均匀度来衡量一个密码系统抵抗差分攻击的能力．差分均匀度越小，其抵抗差分分析的能力就越强．

定义2 设$F(x)$是一个(n,m)函数，$\forall i\in F_2^n,\forall j\in F_2^m$，它的差分分布表是一个$2^n\times 2^m$的矩阵$\Lambda(F)$：

$$\Lambda(F)=\begin{bmatrix}\lambda_{0,0}&\cdots&\lambda_{0,2^{n-1}}\\\vdots&\ddots&\vdots\\\lambda_{2^{n-1},0}&\cdots&\lambda_{2^{n-1},2^{n-1}}\end{bmatrix} \tag{9}$$

式中，$\lambda_{i,j}=\left|\left\{x\in F_2^n:F(x)\oplus F(x\oplus i)=j\right\}\right|$．

定义3 设$F(x)$是一个(n,m)函数，$F(x)$的差分均匀度表示为

$$\delta_F=\max_{i\in F_2^{n*},j\in F_2^m}\lambda_{i,j} \tag{10}$$

3.4 代数次数

函数$F(x)$的代数次数定义为代数标准型ANF中的全局代数次数，如式(11)所示：

$$\deg(F)=\max\left\{|I|:a_i\neq 0,I\in\rho(N)\right\} \tag{11}$$

即函数$F(x)$的代数次数是候选函数f_i中最大的代数次数．S盒的代数次数越高，抵抗高阶差分攻击的能力就越强．

3.5 回旋镖均匀度

由于差分攻击只针对一个S盒，1999年Wagner[26]提出回旋镖攻击作为差分攻击的扩展．但并没有给出衡量密码函数抵抗回旋镖攻击能力的合适指标．2018年欧密会上Cid等人[27]提出回旋镖均匀度，该指标可衡

量分组密码抵抗旋镖攻击的能力. 回旋镖均匀度与回旋镖连接表(Boomerang Connectivity Table, BCT)有关.

定义 4　设 $F(x)$ 是一个 (n,n) 置换, 对于所有的 $a,b\in F_2^{n*}$, 有

$$\begin{aligned}\mathrm{BCT}_F(a,b)=\#\{x\in F_{2^n}: F^{-1}(F(x+a)+b)\\ +F^{-1}(F(x)+b)=a\}\end{aligned}\tag{12}$$

$F(x)$ 的回旋镖均匀度 β_F 定义为 $\beta_F=\max_{a,b\in F_{2^n}^*}\mathrm{BCT}_F(a,b)$.

近年来, 对回旋镖均匀度的研究引起了学者的广泛关注, 可参见文献[28~32].

3.6　自相关

S盒的自相关与三个准则相关, 即严格雪崩准则(Strict Avalanche Criterion, SAC)[33]、扩散准则(Propagation Criterion, PC)[34,35]和全局雪崩准则(Global Avalanche Criterion, GAC)[36].

定义 5　设 $F(x)$ 是一个 (n,m) 函数, 那么 $F(x)$ 的自相关函数定义为

$$C_F(a)=\sum_{x\in F_2^n}(-1)^{D_{a,F}(x)}\tag{13}$$

式中, $a\in F_2^n$, $D_{a,F}(x)=F(x)\oplus F(x\oplus a)$ 是 F 的差分函数.

定义 6　设 $F(x)$ 是一个 (n,m) 函数, 对于任意的 $a\in F_2^n$ 且 $1\leqslant wt(a)\leqslant k$, 函数 $F(x)$ 满足 PC(k), 如果 $D_{a,F}(x)$ 是一个平衡函数. 特别地, 如果函数 $F(x)$ 满足 PC(1), 则函数 $F(x)$ 满足 SAC.

定义 7　设 $F(x)$ 是一个 (n,m) 函数, 函数 $F(x)$ 的 GAC 定义为以下两种形式:

(1)平方之和的形式 $\sigma_F=\sum_{a\in F_2^n}C_F^2(a)$;

(2)绝对值的形式 $\Delta_F=\max_{a\in F_2^n}|C_F(a)|$.

4　设计方案

除上一节中提到的安全性指标外, S盒还需满足以下两个条件:(1) 在软件中, S盒函数的变量数是2的次幂;(2) S盒函数应能被快速计算出来, 基于元胞自动机的S盒很容易在硬件中实现.

根据S盒的设计标准, 本文基于元胞自动机的随机演化设计了8输入8输出S盒.

4.1　S盒的设计

本文选择的元胞自动机是一个一维的8位CA, 假设CA的初始状态为

$$S^0=(s_0,s_1,s_2,s_3,s_4,s_5,s_6,s_7)\tag{14}$$

其中, $s_i, 0\leqslant i\leqslant 7$, 是CA的第 i 个元胞的初始状态. 本文选择了状态转移规则90、150、30和86作为CA中元胞的状态更新函数, 其中规则30和规则86都是非线性的.

步骤 1　假设在第一个循环周期中使用的规则向量是(150,150,30,150,150,90,90,86), 得到每个元胞的更新函数如下:

$$\begin{aligned}\phi_0^0&=s_{-1}\oplus s_0\oplus s_1=s_0\oplus s_1\\ \phi_1^0&=s_0\oplus s_1\oplus s_2\\ \phi_2^0&=s_1\oplus s_2\oplus s_3\oplus(s_2\cdot s_3)\\ \phi_3^0&=s_2\oplus s_3\oplus s_4\\ \phi_4^0&=s_3\oplus s_4\oplus s_5\\ \phi_5^0&=s_4\oplus s_6\\ \phi_6^0&=s_5\oplus s_7\\ \phi_7^0&=s_6\oplus s_7\oplus s_8\oplus(s_6\cdot s_7)\\ &=s_6\oplus s_7\oplus(s_6\cdot s_7)\end{aligned}\tag{15}$$

步骤 2　在第 i 个元胞处插入一个非线性函数 $f_{\mathrm{NL}}(\phi_{i-2},\phi_{i+2})=\phi_{i-2}\cdot\phi_{i+2}$. 假设插入位置是(2,3,4,5), 那么就有:

$$\begin{aligned}\phi_2^0&=s_1\oplus s_2\oplus s_3\oplus(s_2\cdot s_3)\oplus f_{\mathrm{NL}}(\phi_0^0,\phi_4^0)\\ &=s_1\oplus s_2\oplus s_3\oplus(s_2\cdot s_3)\oplus(s_0\cdot s_3)\oplus(s_0\cdot s_4)\\ &\quad\oplus(s_0\cdot s_5)\oplus(s_1\cdot s_3)\oplus(s_1\cdot s_4)\oplus(s_1\cdot s_5)\\ \phi_3^0&=s_2\oplus s_3\oplus s_4\oplus f_{\mathrm{NL}}(\phi_1^0,\phi_5^0)\\ &=s_2\oplus s_3\oplus s_4\oplus(s_0\cdot s_4)\oplus(s_0\cdot s_6)\oplus(s_1\cdot s_4)\\ &\quad\oplus(s_1\cdot s_6)\oplus(s_2\cdot s_4)\oplus(s_2\cdot s_6)\\ \phi_4^0&=s_3\oplus s_4\oplus s_5\oplus f_{\mathrm{NL}}(\phi_2^0,\phi_6^0)\\ &=s_3\oplus s_4\oplus s_5\oplus(s_1\cdot s_5)\oplus(s_1\cdot s_7)\\ &\quad\oplus(s_2\cdot s_5)\oplus(s_2\cdot s_7)\oplus(s_3\cdot s_5)\\ &\quad\oplus(s_3\cdot s_7)\oplus(s_2\cdot s_3\cdot s_5)\oplus(s_2\cdot s_3\cdot s_7)\\ \phi_5^0&=s_4\oplus s_6\oplus f_{\mathrm{NL}}(\phi_3^0,\phi_7^0)\\ &=s_4\oplus s_6\oplus(s_2\cdot s_6)\oplus(s_2\cdot s_7)\oplus(s_3\cdot s_6)\\ &\quad\oplus(s_3\cdot s_7)\oplus(s_4\cdot s_6)\oplus(s_4\cdot s_7)\oplus(s_2\cdot s_6\cdot s_7)\\ &\quad\oplus(s_3\cdot s_6\cdot s_7)\oplus(s_4\cdot s_6\cdot s_7)\end{aligned}\tag{16}$$

步骤 3　将 ϕ_i^0 赋值给 s_i^0 后, CA 当前的状态变为 $S^1=(\phi_0^0,\phi_1^0,\phi_2^0,\phi_3^0,\phi_4^0,\phi_5^0,\phi_6^0,\phi_7^0)$, 并根据步骤 2 计算 $\phi_i^1, 0\leqslant i\leqslant 7$.

步骤 4　设 C 为一个非零常数, 且 $C=(c_1,c_2,c_3,c_4,c_5,c_6,c_7)$ 是 C 的二进制展开式. 然后 c_i 与 ϕ_i^7 相加, 得到S盒的分量函数 $f_i=\phi_i^7\oplus c_i, 0\leqslant i\leqslant 7$. 因此, 设计出的S盒是 $F=S^7\oplus C$.

在这个过程中, 规则向量可以是四个规则(规则30、规则90、规则150、规则86)的任意排列, 可以任意选择(选择一个或多个)插入位置(2,3,4,5), 因此有 $4^8\times 15$ 个S盒. 通过计算机仿真, 选择其中20个S盒如表1和表2所示(使用AES中的S盒作为初始输入, 软件是Sagemath). 读者可

在 https://pan.xunlei.com/s/VNJDXbmqcCv3rTmkgXdhkt-boA1?pwd=5kyg#下载并检验这20个S盒.

表1 设计的S盒(1)

S-Box	规则向量	插入位置	平衡性	代数次数	非零线性结构
S-Box1	(150, 90, 150, 30, 86, 150, 90,30)	[3, 4]	TRUE	7	NO
S-Box2	(90, 150, 90, 90, 150, 90, 150,30)	[5]	TRUE	7	NO
S-Box3	(90, 90, 150, 150, 150, 90, 150, 30)	[4]	TRUE	7	NO
S-Box4	(90, 150, 90, 90, 150, 150, 150, 30)	[2, 3]	TRUE	7	NO
S-Box5	(86, 90, 150, 90, 150, 150, 150, 30)	[2, 5]	TRUE	7	NO
S-Box6	(86, 150, 30, 90, 30, 86, 150, 30)	[2, 3]	TRUE	7	NO
S-Box7	(90, 150, 150, 150, 90, 86, 150, 30)	[3, 4]	TRUE	7	NO
S-Box8	(90, 30, 150, 90, 90, 150, 30, 30)	[2, 5]	TRUE	7	NO
S-Box9	(90, 150, 90, 150, 90, 30, 90, 30)	[4]	TRUE	7	NO
S-Box10	(86, 90, 150, 30, 90, 90, 90, 30)	[4]	TRUE	7	NO
S-Box11	(86, 90, 150, 90, 90, 150, 90, 30)	[3, 4]	TRUE	7	NO
S-Box12	(90, 90, 90, 90, 150, 150, 90, 30)	[5]	TRUE	7	NO
S-Box13	(150, 90, 86, 90, 86, 150, 90, 30)	[2, 5]	TRUE	7	NO
S-Box14	(86, 150, 90, 150, 90, 150, 150, 30)	[3]	TRUE	7	NO
S-Box15	(90, 90, 150, 86, 90, 86, 150, 30)	[5]	TRUE	7	NO
S-Box16	(90, 150, 90, 90, 90, 30, 90, 90)	[4]	TRUE	7	NO
S-Box17	(86, 90, 150, 150, 90, 30, 90, 90)	[2, 5]	TRUE	7	NO
S-Box18	(86, 86, 150, 150, 90, 90, 90, 90)	[4]	TRUE	7	NO
S-Box19	(90, 90, 90, 30, 150, 90, 90, 90)	[5]	TRUE	7	NO
S-Box20	(90, 30, 86, 90, 86, 90, 90, 90)	[2, 5]	TRUE	7	NO

表2 设计的S盒(2)

S-Box	双射	非线性度	差分均匀度	回旋镖均匀度	GAC
S-Box1	TRUE	98	14	22	96
S-Box2	TRUE	98	10	20	96
S-Box3	TRUE	98	10	20	112
S-Box4	TRUE	98	10	30	104
S-Box5	TRUE	98	12	20	96
S-Box6	TRUE	98	12	20	96
S-Box7	TRUE	98	10	18	96
S-Box8	TRUE	96	10	18	88
S-Box9	TRUE	96	12	18	88
S-Box10	TRUE	96	12	18	88
S-Box11	TRUE	96	10	18	88
S-Box12	TRUE	96	12	18	88
S-Box13	TRUE	96	12	18	88
S-Box14	TRUE	96	12	18	88
S-Box15	TRUE	96	12	18	88
S-Box16	TRUE	96	10	18	88
S-Box17	TRUE	96	10	18	88
S-Box18	TRUE	96	12	18	88
S-Box19	TRUE	96	10	18	88
S-Box20	TRUE	96	12	18	88

4.2 逆S盒的设计

基于密码算法设计中关于对称性的思想,逆S盒可以基于设计S盒中使用的元胞自动机进行设计. 设一个逆S盒的输入为$Z=S^7\oplus C$, 这里的C与设计S盒过程中步骤4中的C相同. 逆S盒的设计遵循以下步骤.

步骤1 计算$Z\oplus C$的值,结果为S^7.

步骤2 假设用于设计逆S盒的元胞自动机的状态更新函数为ϕ^{-1},它是函数ϕ的逆(ϕ是S盒设计中元胞自动机的状态更新函数). 经过一个时钟周期后,$S^6=\phi^{-1}(S^7)$.

步骤3 将步骤2重复操作8次,就能得到S^0,$S^0=\phi^{-1}(\cdots\phi^{-1}(\phi^{-1}(S^7)))$.

步骤4 逆S盒的输出$S^0=\left(s_0,s_1,s_2,s_3,s_4,s_5,s_6,s_7\right)$.

5 错误检测和校正

对AES的错误注入攻击通常只需要注入单字节的错误,即可通过分析错误的密文来获得密钥. 因此,设计一些校验字节来检测双字节错误和纠正单字节的错误. 假设错误发生在AES加密算法的字节替换层;当NLCA运行8个时钟后,将实现字节替换(此时可能会发生错误). 在每个时钟周期中,NLCA将演变成下一个状态,所设计的三个校验字节也将根据NLCA的状态随之生成. 这些校验字节存储在冗余寄存器中. 此外,其他

三个校验字节将在每个时钟周期的同一时间生成. 通过比较两次生成的校验字节,可以检测到是否发生错误. 如果发生错误,则把正确的字节赋值给CA状态,并且状态更新函数已建立了下一个CA状态. 它还将为下一个CA状态生成新的校验字节. 每个时钟周期执行检测和校正操作,因此在运行8个时钟周期后,S盒输出的所有字节都是正确的字节. 错误检测和校正设计的架构视图如图2所示. $(A_0,A_1,\cdots,A_7)$为8字节的消息输入,状态更新函数为 $\phi(A_i)=T(A_i)\oplus F_{\mathrm{NL}}(A_i),0\leqslant i\leqslant 7$.

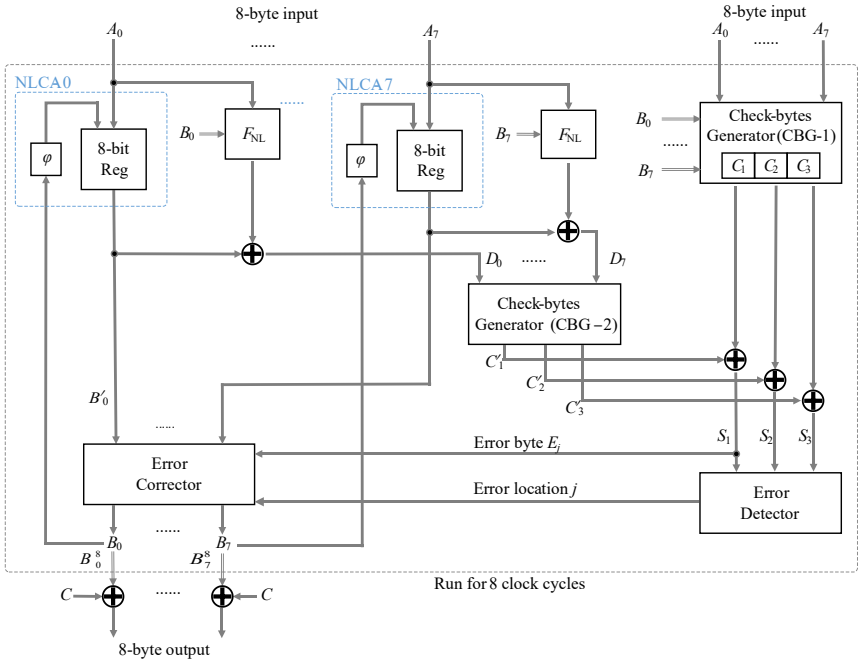

图2　错误检测和校正的架构视图

校验字节生成器CBG-1生成三个校验字节 C_1,C_2,C_3,如图3所示. C_1通过XOR所有输入字节获得,然后CA-1加载结果并运行一个周期,实现功能 T. 对于 C_2,CA-2使用 A_7作为初始种子,运行8个周期. A_i的输入顺序为 $A_7,A_6,\cdots,A_0$. 对于 C_3,CA-3也使用 A_7作为初始种子,输入的顺序是 $A_7,A_6,\cdots,A_1$. 在运行7个周期后,CA-4加载 A_0和CA-3的结果,再运行一个周期,得到式(17).

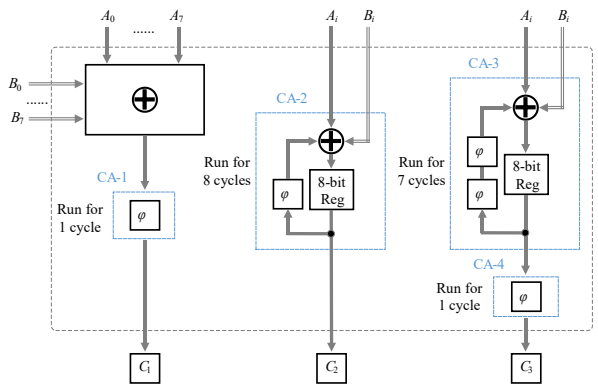

图3　错误检测和校正的架构视图

$$
\begin{aligned}
C_1&=T\left(A_0\oplus A_1\oplus A_2\oplus A_3\oplus A_4\oplus A_5\oplus A_6\oplus A_7\right)\\
C_2&=T\left(A_0\oplus T\left(A_1\oplus T\left(\left(\left(\left(\left(\left(A_2\oplus T\right)\right.\right.\right.\right.\right.\right.\\
&\quad\left.\left.\left.\left.\left.\left.\left(A_3\oplus T\right)\right)\right)\right)\right)\right)\right)\\
&=T\left(A_0\right)\oplus T^2\left(A_1\right)\oplus T^3\left(A_2\right)\oplus T^4\left(A_3\right)\oplus T^5\left(A_4\right)\\
&\quad\oplus T^6\left(A_5\right)\oplus T^7\left(A_6\right)\oplus T^8\left(A_7\right)\\
C_3&=T\left(A_0\oplus T^2\left(A_1\oplus T^2\left(\left(\left(\left(\left(\left(A_2\oplus T^2\right)\right.\right.\right.\right.\right.\right.\right.\\
&\quad\left.\left.\left.\left.\left.\left.\left(A_3\oplus T^2\right)\right)\right)\right)\right)\right)\right)\\
&=T\left(A_0\right)\oplus T^3\left(A_1\right)\oplus T^5\left(A_2\right)\oplus T^7\left(A_3\right)\oplus T^9\left(A_4\right)\\
&\quad\oplus T^{11}\left(A_5\right)\oplus T^{13}\left(A_6\right)\oplus T^{15}\left(A_7\right)
\end{aligned}
\tag{17}
$$

设 $B'=\left(B'_0,B'_1,\cdots,B'_7\right)$为8个NLCA的输出字节(可

能携带错误的字节），$E=(E_0,E_1,\cdots,E_7)$为相应的错误字节．那么正确的输出字节是$B_i=B_i'\oplus E_i$，$B_i=(B_0,B_1,\cdots,B_7)$，因为$B_i'=T(A_i)\oplus F_{\mathrm{NL}}(A_i),0\leqslant i\leqslant 7$，有$B_i=T(A_i)\oplus F_{\mathrm{NL}}(A_i)\oplus E_i$.

其余三个校验字节C_1',C_2',C_3'（由校验字节生成器CBG-2生成，如图4所示）是根据$D_i=B_i'\oplus F_{\mathrm{NL}}(A_i),0\leqslant i\leqslant 7$计算得出．

CBG-2中，C_1'将$(D_0,D_1,\cdots,D_7)$作为8个字节输入，通过XOR获得结果．C_2'中CA-5使用D_7作为初始种子运行7个时钟周期，然后加载D_0得到结果．D_i的输入顺序$D_7,D_6,\cdots,D_0$．C_3'与C_2'计算方法相同，但CA-6实现T^2．假设在第j个字节中有一个错误，则有式(18)．

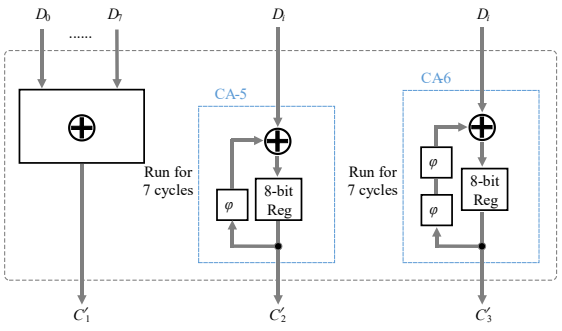

图4 校验字节生成器CBG-2

$$
\begin{aligned}
C_1'&=D_0\oplus D_1\oplus\cdots\oplus D_7\\
&=\left(B_0'\oplus F_{\mathrm{NL}}(A_0)\right)\oplus\left(B_1'\oplus F_{\mathrm{NL}}(A_1)\right)\oplus\cdots\oplus\left(B_7'\oplus F_{\mathrm{NL}}(A_7)\right)\\
&=T(A_0)\oplus T(A_1)\oplus\cdots\oplus T(A_6)\oplus T(A_7)\\
&=T(A_0\oplus A_1\oplus\cdots\oplus A_7)\oplus E_j\\
C_2'&=D_0\oplus T\left(D_1\oplus T\left(\cdots\oplus T\left(D_6\oplus T(D_7)\right)\right)\right)\\
&=D_0\oplus T(D_1)\oplus\cdots\oplus T^7(D_7)\\
&=\left(B_0'\oplus F_{\mathrm{NL}}(A_0)\right)\oplus T\left(B_1'\oplus F_{\mathrm{NL}}(A_1)\right)\oplus\cdots\oplus T^7\left(B_7'\oplus F_{\mathrm{NL}}(A_7)\right)\\
&=T(A_0)\oplus T\left(T(A_1)\right)\oplus\cdots\oplus T^j\left(T(A_j)\oplus E_j\right)\oplus\cdots\oplus T^6\left(T(A_6)\right)\oplus T^7\left(T(A_7)\right)\\
&=T(A_0)\oplus T^2(A_1)\oplus T^3(A_2)\oplus\cdots\oplus T^{j+1}(A_j)\oplus T^j(E_j)\oplus\cdots\oplus T^7(A_6)\oplus T^8(A_7)\\
C_3'&=D_0\oplus T^2\left(D_1\oplus T^2\left(D_2\oplus T^2\left(\cdots\oplus T^2\left(D_5\oplus T^2\left(D_6\oplus T^2(D_7)\right)\right)\cdots\right)\right)\right)\\
&=D_0\oplus T^2(D_1)\oplus T^4(D_2)\oplus\cdots\oplus T^{14}(D_7)\\
&=\left(B_0'\oplus F_{\mathrm{NL}}(A_0)\right)\oplus T^2\left(B_1'\oplus F_{\mathrm{NL}}(A_1)\right)\oplus T^4\left(B_2'\oplus F_{\mathrm{NL}}(A_2)\right)\oplus\cdots\oplus T^{14}\left(B_7'\oplus F_{\mathrm{NL}}(A_7)\right)\\
&=T(A_0)\oplus T^2\left(T(A_1)\right)\oplus\cdots\oplus T^{2j}\left(T(A_j)\oplus E_j\right)\oplus\cdots\oplus T^{13}(A_6)\oplus T^{15}(A_7)\\
&=T(A_0)\oplus T^3(A_1)\oplus T^5(A_2)\oplus\cdots\oplus T^{2j+1}(A_j)\oplus T^{2j}(E_j)\oplus\cdots\oplus T^{13}(A_6)\oplus T^{15}(A_7)
\end{aligned}
\tag{18}
$$

设S_i表示C_i与C_i'的XOR操作的结果，其中$S_i=C_i\oplus C_i',1\leqslant i\leqslant 3$.

（1）如果S_1,S_2,S_3中的两个不为零，1个为零，那么校验字节本身就是错误的．

（2）如果没有错误，则$S_1=S_2=S_3=0$.

（3）在错误字节$E_j,0\leqslant j\leqslant 7$，那么$S_1,S_2,S_3$必须满足$S_1=E_j$，$S_2=T^j[E_j]$，$S_3=T^{2j}[E_j]$.

这时，错误检测器模块（Error Detector）试图找到一个j，满足$S_2=T^j[S_1]$；$S_3=T^{2j}[S_1]$.

同时，将满足条件的j输入到错误校正器模块（Error Corrector）．如果不存在满足条件的，则存在两个字节错误．

（4）如果想要纠正一个错误字节$B_j',0\leqslant j\leqslant 7$，那么错误校正器模块可以通过$B_j=B_j'\oplus S_1$计算正确的字节$B_j$．其余字节保持不变，最后输出字节是正确的字节．

6 结论

错误注入攻击是一种针对分组密码的强有力攻击方法．为了防止密码系统受到错误注入攻击，本文基于非线性（或线性）元胞自动机设计了一些S盒，这些S盒在软件及硬件上都易于实现．通过添加校验字节可以检测双字节错误和纠正单字节错误．与AES中的S盒相比，其密码性能有所降低，但能抵抗错误注入攻击．与文献[19]中的S盒相比，抗差分攻击能力增强，设计方法的运算效率较高．同时本文还考虑了差分分析的一种变形分析方法——回旋镖攻击，用回旋镖均匀度这个指标来衡量S盒抗回旋镖攻击的能力．

参考文献

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

[2] BOUSSELAM K, NATALE G D, FLOTTES M, et al. On countermeasures against fault attacks on the advanced encryption standard[C]//Fault Analysis in Cryptography, Information Security and Cryptography. Berlin: Springer, 2012: 89-108.

[3] ROY I, REBEIRO C, HAZRA A, et al. Safari: Automatic synthesis of fault-attack resistant block cipher implementations[J]. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 2020, 39(4): 752-765.

[4] SCHMIDT J M, MEDWED M. Countermeasures for symmetric key ciphers[C]//Fault Analysis in Cryptography. Berlin: Springer, 2012: 73-87.

[5] AKDEMIR K D, ZHEN W, KARPOVSKY M, et al. Design of Cryptographic Devices Resilient to Fault Injection Attacks Using Nonlinear Robust Codes[M]. Berlin Heidelberg: Springer, 2012.

[6] BARENGHI A, BREVEGLIERI L, KOREN I, et al. Fault injection attacks on cryptographic devices: Theory, practice, and countermeasures[J]. Proceedings of the IEEE, 2012, 100(11): 3056-3076.

[7] GIVEN-WILSON T, JAFRI N, LEGAY A. Combined software and hardware fault injection vulnerability detection [J]. Innovations in Systems and Software Engineering, 2020, 16(2): 101-120.

[8] FENG J, CHEN H, LI Y, et al. A framework for evaluation and analysis on infection countermeasures against fault attacks[J]. IEEE Transactions on Information Forensics and Security, 2020, 15: 391-406.

[9] TUPSAMUDRE H, BISHT S, MUKHOPADHYAY D. Destroying fault invariant with randomization-a countermeasure for AES against differential fault attacks[C]//International Workshop on Cryptographic Hardware and Embedded Systems. Berlin: Springer, 2014: 93-111.

[10] PICEK S, BATINA L, JAKOBOVIC D. Evolving DPA-resistant Boolean functions[C]//International Conference on Parallel Problem Solving from Nature. Berlin: Springer, 2014: 812-821.

[11] PICEK S, EGE B, PAPAGIANNOPOULOS K, et al. Optimality and beyond: The case of 4×4 S-boxes[C]//2014 IEEE International Symposium on Hardware-Oriented Security and Trust. Piscataway: IEEE, 2014: 80-83.

[12] PICEK S, EGE B, BATINA L, et al. On using genetic algorithms for intrinsic side-channel resistance: The case of AES S-box[C]//Workshop on Cryptography and Security in Computing Systems. New York: ACM, 2014: 13-18.

[13] KUMAR K J J, KARTHICK V. AES S-box construction using one dimensional cellular automata rules[J]. International Journal of Computer Applications, 2015, 110(12): 35-39.

[14] PICEK S, MARIOT L, YANG B, et al. Design of S-boxes defined with cellular automata rules[C]//Computing Frontiers Conference. New York: ACM, 2017: 409-414.

[15] GHOSHAL A, SADHUKHAN R, PATRANABIS S, et al. Lightweight and side-channel secure 4×4 S-boxes from cellular automata rules[J]. IACR Transactions on Symmetric Cryptology, 2018: 311-334.

[16] GUAN J, HUANG J. Research on cryptographic properties of a new S-box based on cellular automaton [J]. Journal of Communications, 2019, 40(5): 192-200.

[17] 黄俊君, 关杰. 基于元胞自动机的S盒的性质与神经网络实现研究[J]. 电子学报, 2020, 48(12): 2462-2468.
HUANG J J, GUAN J. Research on properties and neural networks implementation of cellular automata based S-boxes[J]. Acta Electronica Sinica, 2020, 48(12): 2462-2468. (in Chinese)

[18] MARIOT L, PICEK S, LEPORATI A, et al. Cellular automata based S-boxes[J]. Cryptography and Communications, 2019, 11(1): 41-62.

[19] MAITI S, CHOWDHURY D R. Design of fault-resilient S-boxes for AES-like block ciphers[J]. Cryptography and Communications, 2021, 13: 71-100.

[20] NEUMANN J V. Theory of Self-reproducing Automata [M]. Urbana: University of Illinois, 1966.

[21] SCHNEIER B. Applied Cryptography: Protocols, Algorithms, and Source Code in C[M]. Hoboken: Wiley, 1995.

[22] NANDI S, KAR B K, CHAUDHURI P PAL. Theory and applications of cellular automata in cryptography[J]. IEEE Transactions on Computers, 1994, 43(12): 1346-1357.

[23] DIHIDAR K, CHOUDHURY P P. Matrix algebraic formulae concerning some exceptional rules of two-dimensional cellular automata [J]. Information Sciences, 2004, 165(1-2): 91-101.

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

[25] BIHAM E, SHAMIR A. Differential cryptanalysis of des-

like cryptosystems [J]. Journal of Cryptology, 1991, 4(1): 3-72.

[26] WAGNER D. The boomerang attack[C]//International Workshop on Fast Software Encryption. Berlin: Springer, 1999: 156-170.

[27] CID C, HUANG T, PEYRIN T, et al. Boomerang connectivity table: A new crypt-analysis tool[C]//Proceeding of Eurocrypt'18—Advances in Cryptology. Berlin: Springer, 2018: 683-714.

[28] DUNKELMAN O, KELLER N, RONEN E, et al. The retracing boomerang attack[C]//Proceeding of Eurocrypt 2020—Advances in Cryptology. Cham: Springer International Publishing, 2020: 280-309.

[29] LI K, QU L, SUN B, et al. New results about the boomerang uniformity of permutation polynomials[J]. IEEE Transactions on Information Theory, 2019, 65(11): 7542-7553.

[30] MESNAGER S, TANG C, XIONG M. On the boomerang uniformity of quadratic permutations[J]. Designs, Codes and Cryptography, 2020, 88(10): 2233-2246.

[31] TIAN S, BOURA C, PERRIN L. Boomerang uniformity of popular S-box constructions[J]. Designs, Codes and Cryptography, 2020, 88(9): 1959-1989.

[32] TU Z, LI N, ZENG X, et al. A class of quadrinomial permutations with boomerang uniformity four[J]. IEEE Transactions on Information Theory, 2020, 66(6): 3753-3765.

[33] WEBSTER A F, TAVARES S E. On the design of S-boxes[C]//Proceedings of CRYPTO'85 Lecture Notes in Computer Science. Berlin: Springer, 2007: 523-534.

[34] VANLEEKWIJCK W, PRENEEL B, VANLINDEN L, et al. Propagation characteristics of Boolean functions[C]//Proceedings of EUROCRYPT'90 Lecture Notes in Computer Science. Berling: Springer, 1990: 161-173.

[35] PRENEEL B, GOVAERTS R, VANDEWALLE J. Boolean functions satisfying higher order propagation criteria [C]//Proceedings of EUROCRYPT'91. Lecture Notes in Computer Science. Berling: Springer, 1991: 141-152.

[36] ZHANG X, ZHENG Y. GAC-the criterion for global avalanche characteristics of cryptographic functions[C]//The Journal of Universal Computer Science. Berlin: Springer, 1996: 320-337.

作者简介

柴进晋 女,1992年5月出生于陕西省榆林市. 现为空军工程大学讲师. 主要研究方向为密码基础理论与应用研究.
E-mail: jj_chai@163.com

吴 暄 女,1994年12月出生于山西省晋中市. 现为空军工程大学博士研究生. 主要研究方向为人工智能模型应用研究.