一种基于门限与感染技术的SM4算法综合防护实现

焦志鹏[1,2],姚 富[1,2],陈 华[1,2],王 舰[1,2],匡晓云[3],黄开天[3]
(1. 中国科学院软件研究所可信计算与信息保障实验室,北京 100190;2. 中国科学院大学,北京 100049;
3. 南方电网科学研究院,广东广州 510663)

摘 要: 侧信道攻击和故障攻击对于密码算法的实现安全性有着巨大的威胁.针对这样的现状,本文结合门限实现和乘法感染防护思想构造了一种具有抵抗侧信道攻击和故障攻击能力的综合防护方案,以门限实现思想为基础实现了对于侧信道攻击的防护,以乘法感染思想为基础实现了对于故障攻击的防护,二者相互结合使得综合防护方案同时具有抵抗侧信道攻击和故障攻击的能力.此外以门限实现改善了乘法感染防护中随机数为0的缺陷,并且结合随机置换思想进一步提高了防护方案抵抗故障攻击的能力.随后本文依据以上综合防护理论构造了一种适用于SM4算法的综合防护实现方案,并在现场可编程门阵列(Field Programmable Gate Array,FPGA)上进行了具体的实现,最后通过理论分析和安全性评估实验验证了该综合防护方案的安全性.

关键词: 侧信道攻击;故障攻击;门限实现;感染;综合防护;SM4算法
中图分类号: TN918;TP309 **文献标识码:** A **文章编号:** 0372-2112(2022)05-1066-09
电子学报URL:http://www.ejournal.org.cn **DOI:**10.12263/DZXB.20210223

A Comprehensive Protection Implementation of SM4 Algorithm Based on Threshold and Infection Technology

JIAO Zhi-peng[1,2], YAO Fu[1,2], CHEN Hua[1,2], WANG Jian[1,2], KUANG Xiao-yun[3], HUANG Kai-tian[3]
(1. *Trusted Computing and Information Assurance Laboratory, Institute of Software, Chinese Academy of Sciences, Beijing* 100190, *China*;
2. *University of Chinese Academy of Sciences, Beijing* 100049, *China*;
3. *Electric Power Research Institute, China Southern Power Grid, Guangzhou, Guangdong* 510663, *China*)

Abstract: Side channel attack and fault attack are great threats to the security of cryptography implementation. In view of this situation, this paper combines the threshold implementation(TI) and multiplicative infection protection idea to construct a comprehensive protection scheme with the ability to resist side channel attack and fault attack. Based on the idea of threshold implementation, the protection against side channel attack is realized. Based on the idea of multiplicative infection, the protection against fault attack is realized. The combination of the two theory makes the comprehensive protection scheme capable of resisting side channel attack and fault attack at the same time. In addition, threshold implementation improves the flaw of multiplicative infection when the random number is 0, and the ability of the protection scheme to resist fault attack is further improved by combining the idea of random permutation. Then, based on the above comprehensive protection theory, this paper constructs a comprehensive protection implementation scheme suitable for SM4 algorithm, and carries out a specific implementation on field programmable gate array(FPGA). Finally, the security of the comprehensive protection scheme is verified through theoretical analysis and security evaluation experiments.

Key words: side channel attack; fault attack; threshold implementation; infection; comprehensive protection; SM4 algorithm

1 引言

物联网等技术的发展便利了人们各方面的生活,与此同时也使得个人隐私等秘密信息的保护面临着巨大的挑战.灰盒模型攻击的出现使得密码设备的实现安全性受到了严重的威胁.侧信道攻击[1~4]和故障攻击[5]是灰盒模型攻击中两种主流攻击方法.能量分析

收稿日期:2021-02-04;修回日期:2021-06-03;责任编辑:覃怀银
基金项目:国家重点研发计划(No.2018YFB0904900, No.2018YFB0904901)

攻击是应用最广泛的侧信道攻击手段之一,其中差分能量分析[6]因其实现简单、成本低廉以及功能强大等优点,被广泛用于对实际密码设备的攻击过程. 针对于侧信道攻击的威胁,不同的防护理论陆续被提出,其中门限实现防护理论[7]因其兼具可证明安全以及实现代价相对低廉的优点得到了学术界更为广泛的研究和应用. 除了侧信道攻击之外,故障攻击同样是密码设备实现安全性的重要威胁,差分故障攻击是故障攻击中经典的攻击方法[8]. 乘法感染技术通过随机化故障密文对于故障攻击进行防护,其具有较好的故障攻击防护效果,但是存在着随机数为0时易通过功耗曲线区分的缺陷[9]. 仅仅添加能量攻击防护的电路面对故障攻击时是不安全,反之亦然. 目前,针对两类攻击的防护方案往往是两类防护策略的简单叠加,不仅会带来实现代价的大量增加,也可能产生未知的安全漏洞,因此在实现密码算法时应综合考虑能量和故障攻击方法,并对相应的防护方法进行结合. 为达到综合防护的目的,目前已有一些综合防护方案被提出[10~14],但都存在着一定的问题. 目前还没有一种通用的兼具安全性与实现代价的综合防护方案,仍需要针对具体的密码算法以及实现特点去具体构造防护方案.

SM4算法是我国公布的第一个商用密码算法标准,在我国各个领域的信息保护中起着重要的作用[15]. SM4算法同样面临着能量攻击和故障攻击的威胁. 针对能量攻击的威胁,相应的防护方案也随之出现,谭锐能等基于多路径掩码技术构造了SM4算法抗能量攻击的防护实现[16],裴超等针对SM4算法S盒查表实现提出了一种掩码方案[17],李新超等提出并实现了SM4算法S-box的门限实现防护[18],魏曼等提出了一种SM4算法的门限实现防护方法[19]. 在故障攻击防护方面,辛小霞等基于"校验-阻止"原理提出了SM4算法的一种故障防护方法[20]. 但是对于SM4算法目前还缺乏综合防护方案. 本文基于门限实现与乘法感染的原理提出并实现了针对SM4算法的综合防护方案,该方法可以抵抗2阶差分能量攻击,与此同时也可以抵抗差分故障攻击,并且对于能量与故障的组合攻击也具有对应的抵抗能力.

2　SM4算法二阶综合防护实现

2.1　综合防护方案

本文基于门限实现和乘法感染的思想构造了一种综合防护方案并以SM4算法为例对于综合防护方案进行了实现,其整体框架如图1所示,对于原始SM4进行门限实现,在此基础上实现一路完全相同的冗余SM4门限实现,最终通过乘法感染对故障密文进行随机化,从而实现对于侧信道攻击和故障攻击的综合防护.

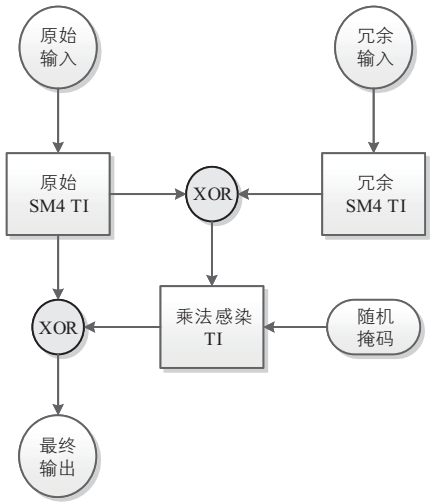


图1　SM4算法综合防护方案

为抵抗d阶差分能量攻击,需要进行d阶的门限实现防护. 关于d阶门限实现,基于不同的考虑有多重变种,其中实现的面积消耗与划分的份额(share)个数正相关,为了达到降低面积消耗的目的,本文选取最低的也就是d+1个share进行门限实现,具体理论可以参考文献[21]. d+1 share实现的d阶门限防护带来实现面积优化的同时也对具体的门限实现有一些细节性的要求,如果实现不当可能会带来一些安全隐患.

对于线性运算来说,d+1 share的d阶门限实现是直接的. 对于非线性运算的d+1 share的d阶门限防护,其实现方法如下所示,单比特的乘法运算是综合防护实现的基本组成单元,因此这里以单比特乘法$y=ab$为例介绍d+1 share的d阶门限实现.

(1) 输入变量的分解

使用布尔掩码将函数输入a和b分解为d+1份;首先利用$2d$份随机数生成前d份输入:$(a_1,b_1,\cdots,a_d,b_d)$,第d+1份输入可以由前d份输入和原始输入的异或生成,具体可以表示为$(a_{d+1},b_{d+1})=(a\oplus(\oplus_{i=1}^{d}a_i),b\oplus(\oplus_{i=1}^{d}b_i))$,最终构成需要的输入$(a_1,b_1,\cdots,a_{d+1},b_{d+1})$,由于随机数的存在,$d$+1份输入满足输入均匀性,也就是说每一种输入掩码都是均匀出现的.

(2) 目标运算的分解

将步骤(1)中分解好的函数输入带入到目标运算$y=ab$中可以得到由掩码输入构成的新的函数表达式,由其组成项a_ib_j, $1\leqslant i,j\leqslant d+1$,构成相应的输出函数$(y_1,\cdots,y_{s_{\text{out}}})$,需要满足正确性:$y=\oplus_{k=1}^{S_{\text{out}}}y_k$,其中$S_{\text{out}}$表示输出函数的个数. 另外为了达到安全防护的目的,d阶门限实现需要满足d阶不完整性,这里需要使得每个输出函数y_k只包含一个组成项a_ib_j,即输出函数的个数满足$S_{\text{out}}=(d+1)(d+1)$. 待防护的密码算法往往包含多轮的运算,因此会出现本阶段运算输出作为下一阶段输

入的情况，为满足下一阶段的输入均匀性的要求，需要对输出函数$(y_1,\cdots,y_{s_{out}})$进行重掩码，另外为了防止毛刺的影响需要在输出之前用寄存器进行存储；最后将$(d+1)(d+1)$份额的输出通过相互异或的方式压缩为$d+1$份作为下一阶段门限防护的输入.

针对故障攻击的威胁，综合防护方案利用乘法感染技术实现相应的防护. 感染防护思想包括多重感染防护和只针对密文输出的单层感染防护，考虑到只针对末轮运算进行故障注入攻击的情况下多重感染将会失效，因此使用单重感染技术从而实现更少的资源占用. 具体的实现方法是在原始SM4门限实现的基础上，冗余实现一路完全相同的SM4门限实现，将原始实现的输出和冗余实现的输出相互异或，然后利用感染函数将异或结果进行随机化，最后将感染函数的输出异或上原始加密的密文形成最终输出. 在感染函数中，为了达到对于故障注入位置以及传播方式的防护，采用了一个随机数控制的置换操作以使得故障注入的错误随机扩散，增加攻击者的攻击难度，然后使用乘法感染技术达到对于故障密文随机化的目的. 此外为了防止乘法感染部分侧信道信息的泄露，在乘法感染部分也进行了门限实现的防护，这种门限实现的方式也有助于改善乘法感染中随机数为0时的缺陷，达到了更好的故障防护效果，具体的更详尽的安全性分析见2.4节.

考虑到侧信道攻击技术的发展和相应设备的更新，针对密码算法实现能量分析攻击的代价越来越低廉，一阶门限实现的防护能力是有限的，因此本文在SM4算法综合防护实例中采用二阶门限实现，也就是3-share的2阶门限实现方案进行侧信道方面的防护，在实际的应用中可以根据不同的安全需求扩展到更高阶的门限实现. SM4算法的综合防护实现主要由SM4算法二阶门限实现部分和乘法感染部分组成，下面分别进行详细的介绍.

2.2 二阶门限实现

SM4算法是32轮Feistel结构的分组算法，分组长度和密钥长度都是128 bit，加密结构和解密结构相同，只是加密密钥和解密密钥逆序，并且密钥扩展部分和加密结构类似，因此本文着重关注加密运算的防护实现，其防护实现可以方便地扩展到解密算法和密钥扩展结构. SM4以字为单位进行加密运算，包含32轮相同的轮运算，首轮输入的四个字可以表示为：$(\boldsymbol{X}_0,\boldsymbol{X}_1,\boldsymbol{X}_2,\boldsymbol{X}_3)$；每轮迭代的轮函数可以表示为：$\boldsymbol{X}_{i+4}=F(\boldsymbol{X}_i,\boldsymbol{X}_{i+1},\boldsymbol{X}_{i+2},\boldsymbol{X}_{i+3},\boldsymbol{rk}_i)=\boldsymbol{X}_i\oplus T(\boldsymbol{X}_{i+1}\oplus\boldsymbol{X}_{i+2}\oplus\boldsymbol{X}_{i+3}\oplus\boldsymbol{rk}_i)$，其中$\boldsymbol{rk}_i$为轮密钥，$0\leqslant i\leqslant 31$，由初始密钥扩展得到的，$T$由两部分组成，包括非线性运算部分和线性运算部分，非线性部分由四个相同的S盒运算组成，线性部分为移位操作，本实现中为了减少面积的消耗只实现一个S盒，每轮执行4次S盒运算得到我们的输出. SM4算法2阶门限实现的步骤如下：

（1）使用随机数将输入明文从128 bit扩展为128×3 bit的输入，分别存储在3个128 bit的寄存器中，$(\boldsymbol{X}_0,\boldsymbol{X}_1,\boldsymbol{X}_2,\boldsymbol{X}_3)$代表原始输入4个字的3-share，也就是分别为32×3 bit，整个SM4加密包括32轮的运算，每一轮的运算需要11个时钟周期，下面详细介绍第i轮运算，$0\leqslant i\leqslant 31$.

（2）第0个时钟周期，$\boldsymbol{X}_{i+1},\boldsymbol{X}_{i+2},\boldsymbol{X}_{i+3}$与本轮的轮密钥相互异或生成32×3 bit结果.

（3）第0到第3个时钟周期将步骤（2）生成的32×3 bit的结果构造为本轮所需的4个S盒运算的输入赋给S盒，每个时钟的输入为8×3 bit，5个时钟后产生输出，具体的S盒实现在下文将详细描述.

（4）第6到第9个时钟周期依次取出4个S盒的输出，每个S盒的输出为8×3 bit.

（5）第10个时钟周期将得到的4次S盒的输出组合为32×3 bit的数据，其中每个份额32 bit分别执行线性移位操作，将线性移位后的32×3 bit数据与$\boldsymbol{X}_i$相互异或得到轮运算结果.

（6）经过32轮轮运算后，逆序输出即可得到最终的加密结果.

本方案中基于有限域的方式实现SM4算法的S盒. AES算法S-box可以看作是$f(x)=x^8+x^4+x^3+x+1$上的求逆运算和仿射运算的组合，本文同样参考文献[22]的原理将SM4算法S盒运算看作为此不可约多项式上求逆运算和仿射运算的组合，从而可以方便地将AES S盒的构造应用到SM4 S盒运算中去，具体可以得到SM4算法S盒的代数表达式为：$S(\boldsymbol{X})=\boldsymbol{A}_2(\boldsymbol{A}_1\boldsymbol{X}+\boldsymbol{C}_1)^{-1}+\boldsymbol{C}_2)$其中$\boldsymbol{A}_1$、$\boldsymbol{A}_2$是仿射矩阵，$\boldsymbol{C}_1=(0\ \ 0\ \ 0\ \ 0\ \ 0\ \ 0\ \ 0\ \ 1)$，$\boldsymbol{C}_2=(1\ \ 1\ \ 0\ \ 1\ \ 0\ \ 0\ \ 1\ \ 1)$，$\boldsymbol{A}_1$、$\boldsymbol{A}_2$表示如下.

$$\boldsymbol{A}_1=\begin{bmatrix}0&0&0&1&0&1&0&0\\0&0&1&0&1&1&1&0\\0&0&0&1&0&1&1&0\\1&0&0&0&1&0&1&0\\0&1&1&0&0&0&0&0\\0&0&0&0&1&1&0&1\\1&0&0&1&1&0&1&1\\0&1&1&0&0&1&1&0\end{bmatrix}$$

$$\boldsymbol{A}_2=\begin{bmatrix}0&0&0&0&1&1&1&0\\1&0&1&0&0&1&0&0\\1&0&0&1&1&1&0&0\\0&0&1&0&0&0&1&0\\0&1&1&1&0&0&0&0\\1&1&0&1&1&0&1&1\\1&0&0&0&1&1&1&0\\0&1&0&1&1&0&0&1\end{bmatrix}$$

通过代数表达式可以看到,S盒的非线性运算为GF(2^8)上的求逆运算,为方便门限实现,本文对其进行复合域分解. 参考文献[23]的复合域实现方式,将GF(2^8)上的求逆运算转换为GF(2)上的运算. 为结构图表述更加清晰,将其表示为GF(2^4)上的运算,其大致结构如图2所示,主体部分包含非线性的3个GF(2^4)上乘法器运算和一个GF(2^4)上的求逆运算(图中虚线部分),以及由常数乘运算以及平方运算组成的线性运算$\boldsymbol{L}_1$;对于乘法运算,最终被转换为GF(2)上的乘法运算;对于求逆运算,首先将GF(2^4)上的求逆运算转换为GF(2^2)上的运算,其包含3个GF(2^2)上的乘法运算以及由常数乘以及平方运算组合的线性运算$\boldsymbol{L}_2$,GF(2^2)上的求逆运算是线性运算,表示为$\boldsymbol{L}_3$, GF(2^2)上的乘法运算同样转换为GF(2)上的乘法运算. $\boldsymbol{M}_1$和$\boldsymbol{M}_2$代表仿射运算和逆仿射运算.

本文的S盒门限实现在图2复合域实现的基础上参考文献[24]进行防护设计,为了防止毛刺的影响以及保持门限实现的2阶不完整性,需要使用寄存器存储相应运算的输出结果,根据安全需要S盒的2阶门限实现需要分为6个阶段进行处理,每个阶段进行的操作如下所述.

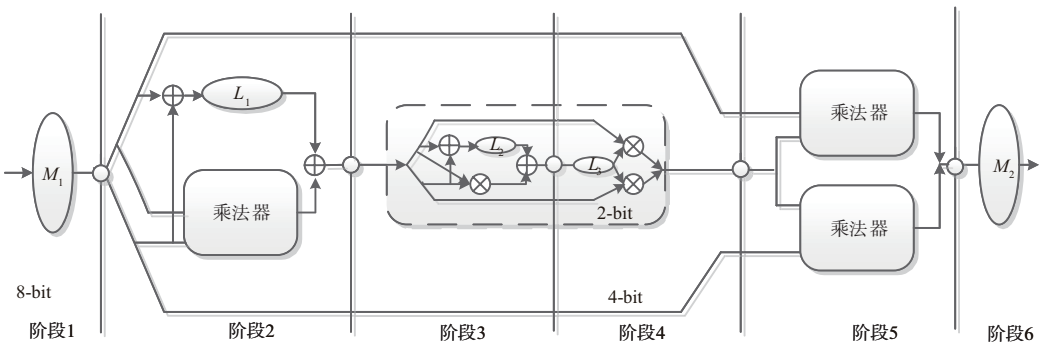


图2　SM4算法S盒复合域实现

阶段1　此阶段包含两个线性操作,首先是$\boldsymbol{A}_1\boldsymbol{X}+\boldsymbol{C}_1$仿射操作,其次是基转换的转置矩阵,为了降低实现电路的复杂度,这里两种线性操作合二为一,为了安全性的需要,本阶段的输出需要使用寄存器存储以供下一阶段运算的使用.

阶段2　此阶段主要包含一个GF(2^4)上的乘法运算以及一个线性运算,线性运算包含乘以常数操作以及平方操作两个运算,这里将线性运算和非线性运算相结合以减少输出份额的个数,从而减少存储所需寄存器的个数;这里输入为3-share的,输出是9-share,$\boldsymbol{L}_1$代表这一阶段的线性运算,以$(\boldsymbol{a}_1,\boldsymbol{a}_2,\boldsymbol{a}_3)$,$(\boldsymbol{b}_1,\boldsymbol{b}_2,\boldsymbol{b}_3)$代表此阶段输入$\boldsymbol{a}$, $\boldsymbol{b}$的3-share随机掩码,以$(\boldsymbol{c}_1,\boldsymbol{c}_2,\boldsymbol{c}_3,\boldsymbol{c}_4,\boldsymbol{c}_5,\boldsymbol{c}_6,\boldsymbol{c}_7,\boldsymbol{c}_8,\boldsymbol{c}_9)$代表门限输出,那么此阶段的运算可以表示为如下的表达:

$$\boldsymbol{c}_1=\boldsymbol{a}_1\otimes\boldsymbol{b}_1\oplus\boldsymbol{L}_1(\boldsymbol{a}_1\oplus\boldsymbol{b}_1);\ \boldsymbol{c}_2=\boldsymbol{a}_1\otimes\boldsymbol{b}_2;\ \boldsymbol{c}_3=\boldsymbol{a}_1\otimes\boldsymbol{b}_3;\ \boldsymbol{c}_4=\boldsymbol{a}_2\otimes\boldsymbol{b}_1;\ \boldsymbol{c}_5=\boldsymbol{a}_2\otimes\boldsymbol{b}_2\oplus\boldsymbol{L}_1(\boldsymbol{a}_2\oplus\boldsymbol{b}_2);\ \boldsymbol{c}_6=\boldsymbol{a}_2\otimes\boldsymbol{b}_3;\ \boldsymbol{c}_7=\boldsymbol{a}_3\otimes\boldsymbol{b}_1;\ \boldsymbol{c}_8=\boldsymbol{a}_3\otimes\boldsymbol{b}_2;\ \boldsymbol{c}_9=\boldsymbol{a}_3\otimes\boldsymbol{b}_3\oplus\boldsymbol{L}_1(\boldsymbol{a}_3\oplus\boldsymbol{b}_3).$$

其中⊕代表4bit的异或运算,⊗代表4bit的乘法运算,由GF(2)上的乘法组合而成. 一方面为了满足下一阶段输入的均匀性,另一方面为了达到抵抗多变量攻击的目的,这里对于$(\boldsymbol{c}_1,\boldsymbol{c}_2,\boldsymbol{c}_3,\boldsymbol{c}_4,\boldsymbol{c}_5,\boldsymbol{c}_6,\boldsymbol{c}_7,\boldsymbol{c}_8,\boldsymbol{c}_9)$使用随机数$(\boldsymbol{r}_1,\cdots,\boldsymbol{r}_6)$进行重掩码,形成输出$(\boldsymbol{O}_1,\cdots,\boldsymbol{O}_9)$,掩码方式如图3所示,并且对于输出使用寄存器暂存.

阶段3　此阶段在GF(2^2)上进行运算,类似于阶段2的运算,不过参与相应运算的数值从4bit变为了2bit.

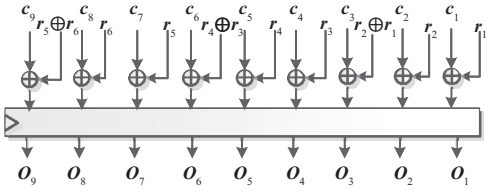


图3　重掩码

这里需要对上一阶段产生的9-share输出压缩为3-share,这里需要小心操作,避免出现脱掩码情况的出现,本文实现中以$\boldsymbol{O}_1\otimes\boldsymbol{O}_4\otimes\boldsymbol{O}_7$,$\boldsymbol{O}_2\otimes\boldsymbol{O}_5\otimes\boldsymbol{O}_8$和$\boldsymbol{O}_3\otimes\boldsymbol{O}_6\otimes\boldsymbol{O}_9$的形式构造3-share的输入,此阶段的输出同样需要进行类似于阶段2的重掩码,并对于输出使用寄存器进行暂存.

阶段4　此阶段包含两个GF(2^2)乘法运算,首先需要将上阶段的9-share压缩为3-share,然后经过一个GF(2^2)上的求逆运算$\boldsymbol{L}_3$,GF(2^2)上的求逆运算是线性运算只需要拉线操作计算产生3-share输出作为乘法运算的一个公共输入,另外结合阶段3的输入构造两个GF(2^2)乘法的其他输入,两个乘法运算的9-share输出相互结合形成9-share的4 bit输出,对这个9-share输出进行与阶段2相同的重掩码,并将掩码结果存储在寄存器中.

阶段5　此阶段类似于阶段4,只不过相应运算从GF(2^2)上的乘法运算变为GF(2^4)上的乘法运算;首先需要将上阶段的9-share按照阶段3的方式压缩为3-share作为两个乘法的公共输入,另外结合阶段2的输

入构造两个GF(2^4)上的乘法的其他输入,两个乘法运算的9-share输出相互结合形成9-share的8 bit输出,对这个9-share输出进行阶段2相同的重掩码,存储在寄存器中.

阶段6 此阶段包含三个线性运算,首先将上阶段的9-share输出按照阶段3的方式压缩为3-share,其次是基的转置矩阵,最后是SM4算法S盒的仿射运算,类似于阶段1,本阶段同样将转置操作和仿射运算相互结合以降低实现电路的复杂度.

2.3 感染防护

在SM4算法原始二阶门限实现的基础上实现一路与原始门限实现完全相同的冗余SM4算法二阶门限实现,冗余实现的输入和采用的随机数与原始实现完全相同. 将原始实现的输出和冗余实现的输出相互异或形成128×3 bit的输出. 如果没有故障注入时,异或输出为0,经过感染结构后的输出为0,最后异或到原始密文形成最终输出,不改变密文输出结果,保证了无故障注入时算法的正确性. 当有故障注入时,异或输出不为0,然后将128×3 bit的输出以32×3 bit为一组分为4组,分别进行感染防护,其中被故障注入影响的字节表现为随机数,与原始实现的输出相互异或形成最终的密文输出,相应的可被利用的故障密文也被随机化了,从而实现了相应的故障防护. 其中感染防护由随机置换和乘法感染两部分组成,具体结构如图4所示.

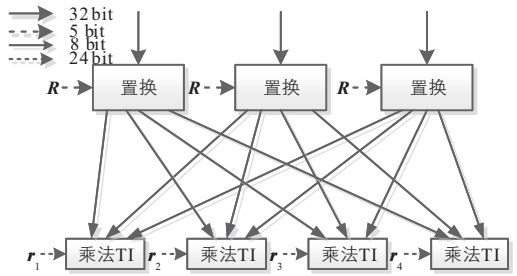


图4 感染函数

为了隐藏故障注入的位置以及故障的传播路径,在进行感染操作之前进行一个32 bit的随机置换操作. 具体的在进行乘法感染操作之前,先进行32 bit的置换操作,初始置换操作(位置标号从0开始)如下:

$$\begin{bmatrix}25 & 17 & 9 & 1 & 27 & 19 & 11 & 3 & 29 & 21 & 13 & 5 & 31 & 23 & 15 & 7\\ 24 & 16 & 8 & 0 & 26 & 18 & 10 & 2 & 28 & 20 & 12 & 4 & 30 & 22 & 14 & 6\end{bmatrix}$$

此置换由一个5 bit的随机数$\boldsymbol{R}$控制,以此增加其随机性,具体是在每一次感染的时候将初始置换进行一个随机的循环移位操作后作为此次感染的置换操作.

经过置换操作后的32×3 bit的输出以8×3 bit为一组和随机数($\boldsymbol{r}_1,\boldsymbol{r}_2,\boldsymbol{r}_3,\boldsymbol{r}_4$)进行GF($2^8$)上的以$f(x)=x^8+x^4+x^3+x+1$为不可约多项式的有限域乘法运算,这里的乘法运算为了防止侧信道信息的泄露也使用了二阶门限实现进行防护,能够有效改善乘法感染中随机数为0时的缺陷. 此外感染防护中的随机置换对于故障传播途径进行了随机化,使得攻击者无法直接进行攻击,从而进一步增强了故障防护效果.

2.4 安全性分析

2.4.1 针对侧信道攻击的安全性

综合防护方案对于侧信道攻击的防护能力继承于门限实现. 一个d阶的门限实现满足输入的均匀性,d阶不完整性;d阶能量攻击可以同时利用d个综合防护方案的中间值的能量信息进行攻击,而门限实现的d阶不完整性使得这d个中间值至少和原始中间值的一个分量相互独立,也就是和原始中间值成相互独立的关系,使得d阶能量攻击者能够利用的信息与原始中间状态相互独立,从而实现了d阶能量攻击下安全性.

具体到本文的SM4算法的综合防护实现,本文的综合防护实现中加密部分和乘法感染部分都基于2阶门限实现进行了防护,其满足2阶不完整性,一个2阶能量攻击者可以同时利用2个综合防护的中间值的能量信息,这2个中间值和原始中间值呈现相互独立的关系,使得2阶能量攻击者能够利用的信息与原始中间状态相互独立,从而实现了2阶能量攻击的安全性.

2.4.2 针对故障攻击的安全性

数值类的故障攻击如差分故障攻击,其攻击条件首先是需要知道故障密文,其次需要分析故障传播路径,因此可以从这两方面实现相应的防护. 在本文综合防护中,一方面随机置换的添加使得故障密文扩散呈现随机性,增加了攻击者对于故障攻击位置以及传播逻辑分析的难度,另一方面乘法感染技术使得输出的故障密文呈现随机性,破坏了输出的故障密文和原始故障密文的相关性,使得需要故障密文的故障攻击无法顺利进行,从而实现了对于数值类故障攻击的防护.

2.4.3 针对组合攻击的安全性

如前文提到的乘法感染在随机数为0时存在缺陷,此时能耗信息较小,然后攻击者可以依据这一点判断出随机数为0时的故障注入,筛选出故障攻击需要的故障密文,从而实施正常的依赖故障密文的故障攻击,此类攻击方法属于一种组合攻击. 普通的GF(2^8)的乘法运算中,只有8 bit的随机数参与运算,找到一个8 bit的为0随机数的情况,相对容易,在本文的综合防护中对于乘法感染同样实现了门限实现防护,由于进行3-share的门限实现防护,只有在3×8 bit的随机数同

时为零的情况下,功耗才容易被分辨出来,实现的困难性大大增加;同时即使找到了3×8 bit随机数为0的情况,其故障攻击仍然存在困难性,因为本文的综合防护的乘法感染防护中进行了随机置换操作,随机化了故障传播的路径,使得实际的故障攻击的逻辑分析仍然存在着困难性,因此可以实现对于此类组合攻击的防护.

另外本文的综合防护方案对于故障注入-探测类型的组合攻击同样有着抵抗能力,在此类攻击中,攻击者通常先通过故障注入的方式将某些中间值置位为0,然后通过探测的方式实现对于原始中间值的获取. 在本文的综合防护方案中,即使注入了对应的故障,如将某些中间值分量置位为0,这同样不会导致敏感信息的泄露,因为门限实现的d阶不完整性使得d阶探测的攻击者仍然无法获取原始中间值的有效信息.

3　实验结果

3.1　实验环境

本文在SAKURA-X评估板上实现了我们的综合防护方案,SAKURA-X是一种专门用来进行侧信道防护方案实现和评估的开发板,主要包含两个FPGA(Field Programmable Gate Array),一个是评估FPGA,用于实现待测方案,我们的综合防护方案实现在其中;另一个是控制FPGA,用来控制评估FPGA和上位机的通信,为避免噪声的影响,随机数发生器实现在其中,使用时传输相应的随机数到评估FPGA中.

3.2　安全性评估

3.2.1　侧信道安全性评估实验

在评估环境设置方面,我们采用便于侧信道攻击的方式进行评估实验,比如:(1)为了降低采集到的功耗信息的噪声,提高信噪比,本文将随机数的产生逻辑和综合防护的实现放在了不同的FPGA中;(2)为了能够使得采集到的功耗曲线更加清晰,我们使得综合防护实现工作在低频状态,具体在本文的实现是工作在375 kHz的时钟下. 在如此便利于攻击者的情况下,如果可以验证本文防护方案的安全性,那么在实际的攻击环境中,综合防护的安全性是可以得到保障的.

我们采用测试向量泄露评估技术(Test Vector Leakage Assessment, TVLA)[25]进行侧信道安全性的评估,它是一种从信息泄露角度评估防护方案安全性的一种技术. 其中不针对具体中间值的non-specific TVLA评估技术是应用最广泛的评估技术,需要收集两组分别是固定输入和随机输入的功耗曲线,当评估高阶泄露时,需要进行合适的预处理,预处理后的曲线作为TVLA待测曲线;其通过t-test技术评估两组曲线分布均值的差异性,从而判断是否有泄露的产生. 本文选择正负4.5作为统计量的阈值,其置信度大于99.999%,当其统计值低于阈值时,则表明通过了测试,当统计值超过阈值时,则表明其存在泄露点,相应的防护方案可能存在易损点,d阶TVLA与d阶差分能量分析相对应,当其通过TVLA,则可认为相应的防护方案是d阶差分能量分析安全的. S盒是防护方案中主要的非线性组件,S盒的侧信道安全性可以反映整个防护方案的安全性,本文主要测试S盒的侧信道安全性.

本文使用PicoScope 3206D示波器在125 MHz的时钟下对于待测方案进行功耗采集,分别在随机数关闭(无防护)和随机数开启(有防护)状态下采集100万功耗曲线进行了TVLA安全性评估. 其一阶TVLA评估结果如图5所示,其二阶TVLA评估结果如图6所示.

图中横轴代表S盒执行过程中不同时刻采集到的样本点,纵轴代表t测试统计量. 从实验结果中,我们看到未添加防护的情况下,统计量都超过了4.5,添加防护后统计量都未超过4.5,验证了我们的防护方案在侧信道安全方面具备抵抗一阶差分能量攻击和二阶差分能量攻击的防护能力.

3.2.2　故障攻击安全性评估

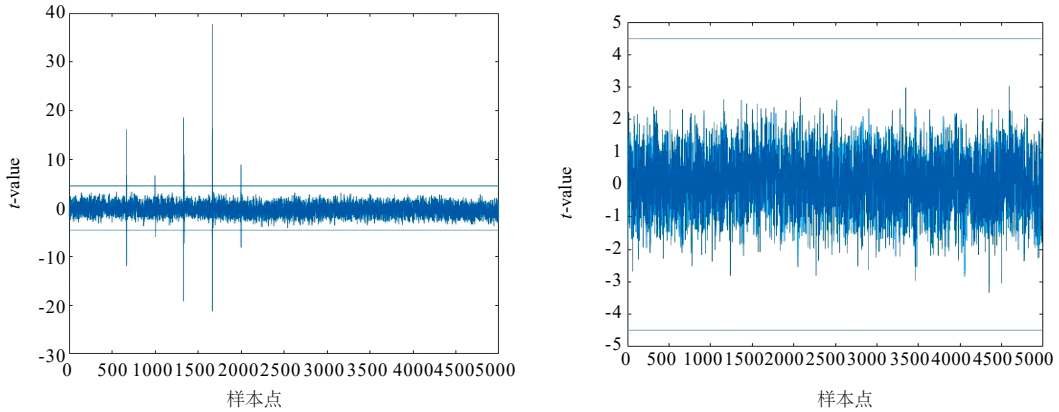
我们通过对SM4算法的最后一轮的S-box输入注入故障来检测综合防护方案的故障防护的安全性,通过分析经过感染结构的故障密文的随机性验证综合防护方案的故障防护的能力. 具体的,选择SM4算法最后一轮运算中的任意一个S-box运算,在综合防护实现代码中将选定的S-box输入置位为常数,从而实现对于故障注入的模拟,然后将此代码生成的流文件下载到FPGA中去,然后执行100万次明文、密钥相同的运算,然后记录密文输出中被故障注入影响到的故障密文,也就是共400万字节的故障密文,最后统计故障密文的随机性,如果故障密文呈现良好的随机性,那么证明本文的防护方案起到了针对故障攻击的防护效果.

对于记录的故障密文使用NIST SP800-22给出的项目和参数进行随机性检测,检测中单样本大小是20000 B,总样本数是200,其检测结果如表1所示.

检测结果表明故障密文通过了随机性检测,经过感染后的故障密文具有良好的随机性,可以达到相应的防护效果,验证了综合防护方案对于故障攻击的安全性.

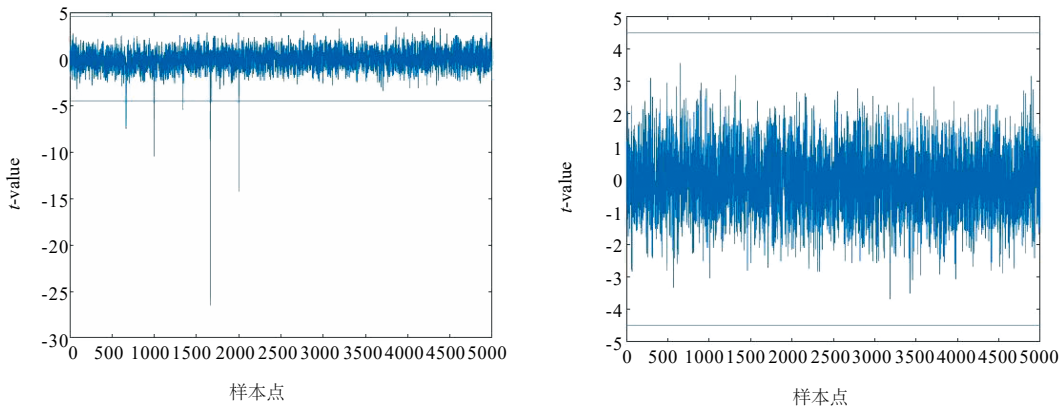
3.3　实现代价评估

我们使用ISE 14.7验证了SM4算法综合防护实现的功能性,使用Synopsys 2016.03在NanGate 45 nm公



(*a*) 未添加防护 (*b*) 添加防护

图5 一阶TVLA测试结果



(*a*) 未添加防护 (*b*) 添加防护

图6 二阶TVLA测试结果

表1 随机性检测的统计结果如下(通过检测的百分比：阈值下限96.889)

检测项目	通过率(%)
重叠子序列(2)-1	99.00
重叠子序列(2)-2	100.00
单比特频数	98.50
近似熵(5)	97.50
矩阵秩	99.50
块内频数(100)	99.00
块内最大游程(128)	99.00
累加和	98.50
离散傅里叶	97.50
线性复杂度(500)	99.00
游程总数	99.50

开元件库下评估了相应的面积消耗,无防护情况下的SM4算法的面积消耗如表2所示;SM4算法综合防护的面积消耗如表3所示,其中“SM4原始门限实现”与“SM4冗余门限实现”完全相同,面积消耗完全一样,因此只展开列出了“SM4原始门限实现”组成部分的面积消耗情况.

表2 SM4算法无防护实现面积消耗

评估目标	面积消耗(GE:等价门)
S盒	505
线性移位	153
密钥扩展	2074
其他寄存器与控制逻辑	2548
总的消耗	5280

本文中SM4算法综合防护方案每轮使用11个时钟周期,共32轮,所以整个防护方案使用了352个时钟周期;在随机数方面,每个S盒运算使用了108 bit的随机数,感染结构中的随机置换部分使用了20 bit的随机数,乘法感染部分使用了384 bit的随机数.

表3　SM4算法综合防护实现面积消耗

评估目标		面积消耗（GE:等价门）
SM4原始门限实现	S盒	3076
	线性移位	460
	密钥扩展	2074
	其他寄存器与控制逻辑	5315
SM4冗余门限实现		10925
SM4算法故障随机置换		3982
SM4算法感染乘法		16368
综合防护控制逻辑与其他寄存器		750
总的面积消耗		42950

4　结论

考虑到灰盒模型攻击的巨大威胁,本文结合门限实现和感染防护的思想提出了一种综合防护方案,通过门限实现弥补了乘法感染故障防护中随机数为0时的缺陷,另外通过添加随机置换的方式掩盖了故障注入位置和故障传播逻辑,进一步强化了对于故障攻击的防护能力. 并且本文以SM4算法为例在FPGA上实际实现了一个二阶综合防护方案,为SM4算法针对侧信道攻击和故障攻击的综合防护提供了一种方案. 最后本文通过安全性分析以及实际的实验验证了SM4防护方案的有效性,并评估了相应的实现代价.

参考文献

[1] KOCHER P C. Timing attacks on implementations of Diffie-Hellman, RSA, DSS, and other systems[C]//Advances in Cryptology—CRYPTO'96. Berlin, Heidelberg: Springer-Verlag, 1996: 104-113.

[2] MESSERGES T S. Using second-order power analysis to attack DPA resistant software[C]//Cryptographic Hardware and Embedded Systems—CHES 2000. Berlin, Heidelberg: Springer-Verlag, 2000: 238-251.

[3] FERRIGNO J, HLAVÁČ M. When AES blinks: introducing optical side channel[J]. IET Information Security, 2008, 2(3): 94-98.

[4] GENKIN D, SHAMIR A, TROMER E. RSA key extraction via low-bandwidth acoustic cryptanalysis[C]//Advances in Cryptology—CRYPTO 2014. Berlin, Heidelberg: Springer-Verlag, 2014: 444-461.

[5] BONEH D, DEMILLO R A, LIPTON R J. On the importance of checking cryptographic protocols for faults[C]//Advances in Cryptology—EUROCRYPT'97. Berlin, Heidelberg: Springer-Verlag, 1997: 37-51.

[6] KOCHER P, JAFFE J, JUN B. Differential power analysis [C]//Advances in Cryptology—CRYPTO'99. Berlin, Heidelberg: Springer-Verlag, 1999: 388-397.

[7] NIKOVA S, RECHBERGER C, RIJMEN V. Threshold implementations against side-channel attacks and glitches [C]//Information and Communications Security. Berlin, Heidelberg: Springer-Verlag, 2006: 529-545.

[8] PIRET G, QUISQUATER J J. A differential fault attack technique against SPN structures, with application to the AES and KHAZAD[C]//Cryptographic Hardware and Embedded Systems—CHES 2003. Berlin, Heidelberg: Springer-Verlag, 2003: 77-88.

[9] FENG J Y, 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.

[10] ISHAI Y, PRABHAKARAN M, SAHAI A, et al. Private circuits II: keeping secrets in tamperable circuits[C]//Advances in Cryptology—EUROCRYPT 2006. Berlin, Heidelberg: Springer-Verlag, 2006: 308-327.

[11] DE CNUDDE T, NIKOVA S. More efficient private circuits II through threshold implementations[C]//2016 Workshop on Fault Diagnosis and Tolerance in Cryptography(FDTC). New Jersey: IEEE, 2016: 114-124.

[12] SCHNEIDER T, MORADI A, GÜNEYSU T. ParTI-towards combined hardware countermeasures against side-channel and fault-injection attacks[C]//Advances in Cryptology—CRYPTO 2016. Berlin, Heidelberg: Springer-Verlag, 2016: 302-332.

[13] REPARAZ O, DE MEYER L, BILGIN B, et al. CAPA: the spirit of beaver against physical attacks[C]//Advances in Cryptology—CRYPTO 2018. Berlin, Heidelberg: Springer-Verlag, 2018: 121-151.

[14] MEYER L D, ARRIBAS V, NIKOVA S, et al. M&M: Masks and macs against physical attacks[J]. IACR Transactions on Cryptographic Hardware and Embedded Systems, 2019, 2019(1): 25-50.

[15] 吕述望, 苏波展, 王鹏, 等. SM4分组密码算法综述[J]. 信息安全研究, 2016, 2(11): 995-1007.
LÜ Shu-wang, SU Bo-zhan, WANG Peng, et al. Overview on SM4 algorithm[J]. Journal of Information Security Research, 2016, 2(11): 995-1007. (in Chinese)

[16] 谭锐能,卢元元,田椒陵. 抗侧信道攻击的SM4多路径乘法掩码方法[J]. 计算机工程, 2014, 40(05): 103-108, 114.
TAN Rui-neng, LU Yuan-yuan, TIAN Jiao-ling. SM4 multi-path multiplicative masking method against side-channel attack[J]. Computer Engineering, 2014, 40(05): 103-108, 114. (in Chinese)

[17] 裴超. 一种SM4掩码方法和抗DPA攻击分析[J]. 密码学报, 2016, 3(01): 79-90.
PEI Chao. A method of masking SM4 and analysis against DPA attacks[J]. Journal of Cryptologic Research, 2016, 3(01): 79-90. (in Chinese)

[18] 李新超,钟卫东,张帅伟,等. 一种SM4算法S盒的门限实现方案[J]. 密码学报, 2018, 5(06): 641-650.
LI Xin-chao, ZHONG Wei-dong, ZHANG Shuai-wei, et al. A New Threshold Implementation of the S-box in SM4[J]. Journal of Cryptologic Research, 2018, 5(06): 641-650. (in Chinese)

[19] WEI Man, SUN Siwei, WEI Zihao , HU Lei. Unbalanced sharing: a threshold implementation of SM4[J]. Science China(Information Sciences), 2021, 64(05): 218-220.

[20] 辛小霞. 抗故障攻击的硬件密码算法研究与实现[D]. 湖南长沙: 湖南大学, 2015.
XIN Xiao-xia. The Research and Implementation of Hardware Cryptographic Algorithms to Resist Fault Attack[D]. Changsha, Hunan: Hunan University, 2015. (in Chinese)

[21] REPARAZ O, BILGIN B, NIKOVA S, GIERLICHS B, VERBAUWHEDE I. Consolidating masking schemes in CRYPTO[C]//Advances in Cryptology—CRYPTO 2015. Berlin, Heidelberg: Springer-Verlag, 2015: 764-783.

[22] MAO W, BAI X, WEN L. Methods and apparatus for secure and efficient implementation of block ciphers: CN2017/080318[P]. 2017-04-12.

[23] CANRIGHT, D. A very compact S-Box for AES[C]//Cryptographic Hardware and Embedded Systems—CHES 2005. Berlin, Heidelberg: Springer-Verlag, 2005: 441-455.

[24] CNUDDE T D, REPARAZ O, BILGINBEGÜL, et al. Masking AES with d+1 shares in hardware[C]//Cryptographic Hardware and Embedded Systems—CHES 2016. Berlin, Heidelberg: Springer-Verlag, 2016: 194-212.

[25] SCHNEIDER T, MORADI A. Leakage assessment methodology[C]//Cryptographic Hardware and Embedded Systems—CHES 2015. Berlin, Heidelberg: Springer-Verlag, 2015: 495-513.

作者简介

焦志鹏 男,1992年生于河南省平顶山市,现为中国科学院软件研究所博士研究生,研究方向为侧信道分析与防护.
E-mail: zhipeng2017@iscas.ac.cn

姚 富 男,1990年生于山西省朔州市,现为中国科学院软件研究所博士研究生,研究方向为侧信道分析与防护.
E-mail: yaofu2020@iscas.ac.cn

陈 华 女,1976年生于山东省日照市,现为中国科学院软件研究所正高级工程师,博士生导师,研究方向为侧信道分析与防护、密码检测.
E-mail: chenhua@iscas.ac.cn