(19)中华人民共和国国家知识产权局

(12)发明专利

(10)授权公告号 CN 106209347 B
(45)授权公告日 2019.03.19

(21)申请号 201610584188.8

(22)申请日 2016.07.22

(65)同一申请的已公布的文献号
申请公布号 CN 106209347 A

(43)申请公布日 2016.12.07

(73)专利权人 武汉大学
地址 430072 湖北省武汉市武昌区珞珈山武汉大学

(72)发明人 唐明 郭志鹏 李煜光 李延斌 王蓬勃

(74)专利代理机构 武汉科皓知识产权代理事务所(特殊普通合伙) 42222
代理人 魏波

(51)Int.Cl.
H04L 9/06(2006.01)
H04L 9/00(2006.01)

(56)对比文件
CN 106788974 A,2017.05.31,
CN 104603745 A,2015.05.06,
郑新建等.抗DPA攻击的AES算法研究与实现.《计算机科学与探索》.2009,(第4期),
Rivain,M..provably secure higher-order masking of AES.《Springer heidelberg》.2010,

审查员 夏晓蕾

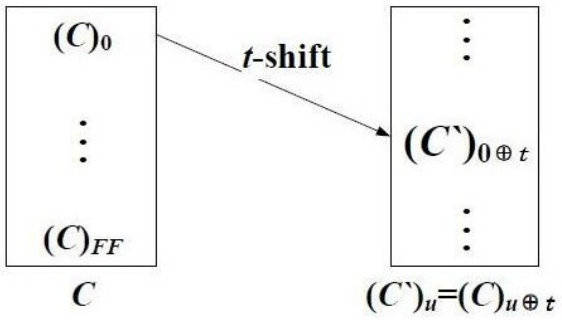
权利要求书2页 说明书6页 附图1页

(54)发明名称

一种复杂度可变的任意阶掩码防护方法

(57)摘要

本发明公开了一种复杂度可变的任意阶掩码防护方法,以k'个长度为2^k比特的逻辑函数配置序列S_t和d个k比特输入共享因子$x_1,x_2,\cdots,x_d$为输入,得到d个k比特输出共享因子$y_1,y_2,\cdots,y_d$;其中,S_t表示原始加密算法中S盒中第t个输出比特对应的逻辑函数配置序列,$1\leqslant t\leqslant k'$;输入共享因子满足$x=x_1\oplus x_2\oplus\ldots\oplus x_d$,输出共享因子满足$y=S(x)=y_1\oplus y_2\oplus\ldots\oplus y_d$;、本发明资源开销相对较低,并且资源开销和运行效率可以根据实际应用环境进行调整;可有效对抗任意阶侧信道攻击,安全性高;本发明实用性强,实现简单,可应用于不同的软硬件载体中,可扩展性强。



CN 106209347 B

1.一种复杂度可变的任意阶掩码防护方法,其特征在于,以k'个长度为2^k比特的逻辑函数配置序列S_t和d个k比特输入共享因子$x_1,x_2,\cdots,x_d$为输入,得到d个k比特输出共享因子$y_1,y_2,\cdots,y_d$;其中,S_t表示原始加密算法中S盒中第t个输出比特对应的逻辑函数配置序列,1≤t≤k';输入共享因子满足$x=x_1\oplus x_2\oplus\ldots\oplus x_d$,输出共享因子满足$y=S(x)=y_1\oplus y_2\oplus\ldots\oplus y_d$;其具体实现包括以下步骤:

步骤1:用整数t作为循环变量,初始值为1;

步骤2:将S_t作为输入进行逻辑函数拆分,得到d个逻辑函数配置序列$C_1,C_2,\ldots,C_d$;用整数i,k作为循环变量,初始值均为1;

所述逻辑函数拆分,是以原始逻辑函数的配置序列为输入,该配置序列记为C;根据掩码阶数d拆分成d个相同长度的逻辑函数配置序列,其中任意d-1个配置序列的异或都与原始逻辑函数配置序列独立,全部d个配置序列的异或与C相等;其具体实现包括以下步骤:

步骤2.1:生成一个长度为2^k比特的整数C_1,令C_1=C;用一个整数i作为循环变量,初始值为2;

步骤2.2:生成一个长度为2^k比特的变量随机整数r,令C_i=r;用C_1和C_i的异或代替C_1,即令$C_1=C_1\oplus C_i$;将循环变量i增加1,即令i=i+1,若i≤d,重复执行步骤2.2;

步骤2.3:输出$C_1,C_2,\ldots,C_d$;其中

$$C_1=C\oplus C_2\oplus\cdots\oplus C_d$$

$$C_1\oplus C_2\oplus\cdots\oplus C_d=C;$$

步骤3:将C_k和x_i作为输入进行逻辑函数调整,得到逻辑函数配置序列C'_k;将循环变量k增加1,即令k=k+1;若k≤n,重复执行步骤3;

所述逻辑函数调整,是以一个2^k比特的逻辑函数配置序列和一个k比特共享因子为输入,该逻辑函数配置序列记为C,该共享因子记为t;根据t调整C中每比特的位置,得到新的逻辑函数配置序列,该逻辑函数配置序列记为C';满足C'中的任意比特$(C')_u$与C中第$u\oplus t$比特$(C)_{u\oplus t}$相等,其中0≤u≤2^k-1;其具体实现包括以下步骤:

步骤3.1:生成一个长度为2^k比特的整数C'=0;用整数i作为循环变量,初始值为0;

步骤3.2:令C'中的第i比特$(C')_i=(C)_{u\oplus t}$;将循环变量i增加1,即令i=i+1;若i≤2^k,重复执行步骤3.2;

步骤3.3:输出新的逻辑函数配置序列C';

步骤4:将步骤3中得到的d个逻辑函数配置序列$C'_1,C'_2,\ldots,C'_d$作为输入进行逻辑函数刷新,并替换原有的$C_1,C_2,\ldots,C_d$;将循环变量i增加1,即令i=i+1;若i≤n-1,重复执行步骤3和步骤4;

所述逻辑函数刷新,d个2^k比特的逻辑函数配置序列$C'_1,C'_2,\ldots,C'_d$为算法的输入,输出新的逻辑函数配置序列$C_1,C_2,\ldots,C_d,tmp_1,tmp_2,\ldots,tmp_{d-1}$表示d-1个$2^k$比特的随机逻辑函数配置序列;其具体实现包括以下步骤:

步骤4.1:生成一个长度为2^k比特的整数C_1,令$C_1=C'_1$;用整数i作为循环变量,初始值为2;

步骤4.2:生成一个长度为2^k比特的随机整数C_i;用C'_1与C_i的异或代替C'_1,即令$C'_1=C'_1\oplus C_i$;将循环变量i增加1,即令i=i+1;若i≤d,重复执行步骤4.2;

步骤4.3,输出$C_1,C_2,...,C_d$;其中d个输入的异或与d个输出的异或相等,并且每个输出C_i都和对应的输入C'_i独立无关,1≤i≤d;

步骤5:用整数j作为循环变量,初始值为1;

步骤6:令y_j的第k个比特等于逻辑函数配置序列C_j中第x_d个比特,即令$(y_j)_k=(C_j)_{xd}$;将循环变量j增加1,即令j=j+1;若j≤k',重复执行步骤6;

步骤7:将循环变量t增加1,即令t=t+1;若t≤k,重复执行步骤2-步骤6;

步骤8:输出$y_1,y_2,\cdots,y_d$。

一种复杂度可变的任意阶掩码防护方法

技术领域

[0001] 本发明属于密码安全技术领域，尤其涉及一种针对不同软硬件载体的高阶掩码方法，可有效对抗任意阶侧信道攻击。

背景技术

[0002] 密码芯片或加密设备的载体形式已受到多种类型的分析与攻击，特别是针对硬件电路的侧信道攻击(Side Channel Attack,SCA)，目前已成为密码算法硬件形式的主要威胁。所谓侧信道攻击是指利用电路工作过程中的各种侧信道泄露信息，如：能耗、时间、故障、电磁辐射等([文献1-6])，通过建立这些泄露信息与密码算法关键信息(如密钥)间的联系，实现对秘密信息的提取。

[0003] 掩码对抗方案是一种应用广泛的侧信道对抗方法，掩码对抗方案自提出以来([文献7])，从一阶对抗逐渐发展至高阶对抗阶段([文献8-13])，安全性及通用性也不断提高。最早的一阶掩码方案主要针对DES算法提出，之后出现的一阶掩码方案则大多以AES为防护目标，针对于不同的软硬件平台，同时不断优化时间和空间耗费。但这些对抗方案都只能对抗一阶SCA攻击，一阶掩码方案已不能满足安全性要求，高阶掩码方案便逐渐发展起来。在追求更高安全性的同时，高阶掩码方案也不断朝着通用化的方向发展，主要在于设计通用化的S盒掩码方案，保证可应用于任何S盒设计且可抵抗任意阶SCA攻击，但是高阶掩码会很大程度上增加额外的开销，因此在资源受限的设备上，高阶掩码方案难以得到应用。

[0004] [文献1]P.Kocher.Timing attacks on implementations of Diffie-Hellmann, RSA,DSS,and other systems.CRYPTO'96,LNCS 1109,pp.104-113,1996.

[0005] [文献2]Eli Biham,Adi Shamir.Differential Fault Analysis of Secret Key Cryptosystems.CRYPTO'97

[0006] [文献3]P.Kocher,J.Jaffe,and B.Jun.Differential Power Analysis[A].CRYPTO 1999[C],Berlin Heidelberg:Springer-Verlag,1999:388-397.

[0007] [文献4]Quisquater J.J,Samyde D.Electromagnetic analysis (EMA):Measures and countermeasures for smart cards.Cannes,France:ACM 2001

[0008] [文献5]E.Brier,C.Clavier,and F.Olivier.Correlation Power Analysis with a Leakage Model[A].CHES 2004[C],Berlin Heidelberg:Springer-Verlag,2004:16-29.

[0009] [文献6]B.Gierlichs,L.Batina,P.Tuyls,and B.Preneel.Mutual Information Analysis[A].CHES 2008[C],Berlin Heidelberg:Springer-Verlag,2008:426-442.

[0010] [文献7]S.Chari,C.S.Jutla,J.R.Rao,and P.Rohatgi.Towards Sound Approaches to Counteract Power Analysis Attacks[A].CRYPTO 1999[C],Berlin Heidelberg:Springer-Verlag,1999:398-412.

[0011] [文献8]Akkar,M.-L.,Giraud,C.:An Implementation of DES and AES,Secure against Some Attacks.In:Ko,c,C,.K.,Naccache,D.,Paar,C.(eds.)CHES 2001.LNCS,

vol.2162,pp.309-318.Springer,Heidelberg(2001)

[0012] [文献9]Rivain,M.,Dottax,E.,Prouff,E.:Block ciphers implementations provably secure against second order side channel analysis.In:Nyberg,K.(ed.)FSE2008.LNCS,vol.5086,pp.127-143.Springer,Heidelberg(2008)

[0013] [文献10]Rivain,M.,Prouff,E.:Provably secure higher-order masking of AES.In:Mangard,S.,Standaert,F.-X.(eds.)CHES 2010.LNCS,vol.6225,pp.413-427.Springer,Heidelberg(2010)

[0014] [文献11]Carlet,C.,Goubin,L.,Prouff,E.,Quisquater,M.,Rivain,M.:Higher-order masking schemes for S-Boxes.In:Canteaut,A.(ed.)FSE 2012.LNCS,vol.7549,pp.366-384.Springer,Heidelberg(2012)

[0015] [文献12]Roy,A.,Vivek,S.:Analysis and improvement of the generic higher-order masking scheme of FSE 2012.In:Bertoni,G.,Coron,J.-S.(eds.)CHES 2013.LNCS,vol.8086,pp.417-434.Springer,Heidelberg(2013)

[0016] [文献13]Coron,J.-S.:Higher Order Masking of Look-Up Tables.In EUROCRYPT 2014,LNCS,vol 8441,2014,pp 441-458.Springer Heidelberg(2014)

发明内容

[0017] 本发明以掩码对抗方法这种具有通用性及可证明安全性的对抗方法为研究目标，提出一种轻量级任意阶掩码对抗方法，旨在使用更低资源开销的情况下，保证任意阶掩码防护方案安全性。

[0018] 本发明所采用的技术方案是：一种复杂度可变的任意阶掩码防护方法，其特征在于，以k'个长度为2^k比特的逻辑函数配置序列S_t和d个k比特输入共享因子$x_1,x_2,\cdots,x_d$为输入，得到d个k比特输出共享因子$y_1,y_2,\cdots,y_d$；其中，S_t表示原始加密算法中S盒中第t个输出比特对应的逻辑函数配置序列，$1\leqslant t\leqslant k'$；输入共享因子满足$x=x_1\oplus x_2\oplus\cdots\oplus x_d$，输出共享因子满足$y=S(x)=y_1\oplus y_2\oplus\cdots\oplus y_d$；其具体实现包括以下步骤：

[0019] 步骤1：用整数t作为循环变量，初始值为1；

[0020] 步骤2：将S_t作为输入进行逻辑函数拆分，得到d个逻辑函数配置序列$C_1,C_2,\ldots,C_d$；用整数i,k作为循环变量，初始值均为1；

[0021] 步骤3：将C_k和x_i作为输入进行逻辑函数调整，得到逻辑函数配置序列C'_k；将循环变量k增加1，即令$k=k+1$；若$k\leqslant n$，重复执行步骤3；

[0022] 步骤4：将步骤3中得到的d个逻辑函数配置序列$C'_1,C'_2,\ldots,C'_d$作为输入进行逻辑函数刷新，并替换原有的$C_1,C_2,\ldots,C_d$；将循环变量i增加1，即令$i=i+1$；若$i\leqslant n-1$，重复执行步骤3和步骤4；

[0023] 步骤5：用整数j作为循环变量，初始值为1；

[0024] 步骤6：令y_j的第k个比特等于逻辑函数配置序列C_j中第x_d个比特，即令$(y_j)_k=(C_j)_{x_d}$；将循环变量j增加1，即令$j=j+1$；若$j\leqslant k'$，重复执行步骤6；

[0025] 步骤7：将循环变量t增加1，即令$t=t+1$；若$t\leqslant k$，重复执行步骤2-步骤6；

[0026] 步骤8：输出$y_1,y_2,\cdots,y_d$。

[0027] 作为优选，步骤2中所述逻辑函数拆分，是以原始逻辑函数的配置序列S_t(为方便

描述，在该步骤中记为C)为输入，根据掩码阶数d拆分成d个相同长度的逻辑函数配置序列，其中任意d-1个配置序列的异或都与原始逻辑函数配置序列独立，全部d个配置序列的异或与C相等；其具体实现包括以下步骤：

[0028] 步骤2.1：生成一个长度为2^k比特的整数C_1，令$C_1=C$；用一个整数i作为循环变量，初始值为2；

[0029] 步骤2.2：生成一个长度为2^k比特的变量随机整数r，令$C_i=r$；用C_1和C_i的异或代替C_1，即令$C_1=C_1\oplus C_i$；将循环变量i增加1，即令i＝i+1，若i≤d，重复执行步骤2.2；

[0030] 步骤2.3：输出$C_1,C_2,\ldots,C_d$；其中

[0031] $C_1=C\oplus C_2\oplus\cdots\oplus C_d$

[0032] $C_1\oplus C_2\oplus\cdots\oplus C_d=C$。

[0033] 作为优选，步骤3中所述逻辑函数调整，是以一个2^k比特的逻辑函数配置序列C_k(为方便描述，在该步骤中记为C)和一个k比特共享因子x_i(为方便描述，在该步骤中记为t)为输入，根据t调整C中每比特的位置，得到新的逻辑函数配置序列C'_k(为方便描述，在该步骤中记为C')，满足C'中的任意比特$(C')_u$与C中第$u\oplus t$比特$(C)_{u\oplus t}$相等，其中$0\leq u\leq 2^k-1$；其具体实现包括以下步骤：

[0034] 步骤3.1：生成一个长度为2^k比特的整数C'＝0；用整数i作为循环变量，初始值为0；

[0035] 步骤3.2：令C'中的第i比特$(C')_i=(C)_{u\oplus t}$；将循环变量i增加1，即令i＝i+1；若$i\leq 2^k$，重复执行步骤3.2；

[0036] 步骤3.3：输出新的逻辑函数配置序列C'。

[0037] 作为优选，步骤4中所述逻辑函数刷新，d个2^k比特的逻辑函数配置序列$C'_1,C'_2,\ldots,C'_d$为算法的输入，输出新的逻辑函数配置序列$C_1,C_2,\ldots,C_d,tmp_1,tmp_2,\ldots,tmp_{d-1}$表示d-1个$2^k$比特的随机逻辑函数配置序列；其具体实现包括以下步骤：

[0038] 步骤4.1：生成一个长度为2^k比特的整数C_1，令$C_1=C'_1$；用整数i作为循环变量，初始值为2；

[0039] 步骤4.2：生成一个长度为2^k比特的随机整数C_i；用C'_1与C_i的异或代替C'_1，即令$C'_1=C'_1\oplus C_i$；将循环变量i增加1，即令i＝i+1；若i≤d，重复执行步骤4.2；

[0040] 步骤4.3，输出$C_1,C_2,\ldots,C_d$；其中d个输入的异或与d个输出的异或相等，并且每个输出C_i都和对应的输入C'_i独立无关，$1\leq i\leq d$。

[0041] 本发明的特点与优势：

[0042] (1)本发明提出了任意阶掩码防护方案，方案资源开销相对较低，并且方案的资源开销和运行效率可以根据实际应用环境进行调整；

[0043] (2)本发明可有效对抗任意阶侧信道攻击，安全性高；

[0044] (3)本发明实用性强，实现简单，可应用于不同的软硬件载体中，可扩展性强。

附图说明

[0045] 图1为本发明实施例的逻辑函数调整过程示意图。

[0046] 图2为本发明实施例的逻辑函数刷新过程示例图。

具体实施方式

[0047] 为了便于本领域普通技术人员理解和实施本发明，下面结合附图及实施例对本发明作进一步的详细描述，应当理解，此处所描述的实施示例仅用于说明和解释本发明，并不用于限定本发明。

[0048] 本发明提出一种轻量级任意阶掩码防护方案，称为PFD方案(Polynomial Function Division Scheme)，以有效对抗任意阶侧信道攻击。

[0049] 本发明实施例重新构造非线性运算来替换原始加密算法中的非线性部件，包含逻辑函数拆分算法，逻辑函数调整算法，逻辑函数刷新算法以及完整的PFD方案。

[0050] 本发明实施例的加密算法的非线性部件S(x)可定义为：

[0051] $S:\{0,1\}^{k}\rightarrow\{0,1\}^{k'}$

[0052] 其中，输入宽度为k，输出宽度为k'，其原始输入为x，原始输出y，对于安全的掩码方法要求将x拆分为相互独立的d个共享因子，记为$x_1,x_2,\cdots,x_d$，且满足$x=x_1\oplus x_2\oplus\cdots\oplus x_d$。将y拆分成$y_1,y_2,\cdots,y_d$，且满足$y=y_1\oplus y_2\oplus\cdots\oplus y_d$。

[0053] 在整个PFD方案中，非线性部件S(x)由k'个逻辑函数配置序列$S_1,\cdots,S_{k'}$表示。每个逻辑函数配置序列S_i($1\leqslant i\leqslant k'$)是一个长度为2^k比特的整数，该整数从低到高第j($0\leqslant j\leqslant 2^k-1$)个比特记为$(S_i)_j$，它的值满足公式(1)

[0054] $(S_i)_j=(S(j)>>(i-1))\&1$；

[0055] 其中>>表示右移运算，&表示逻辑与。

[0056] 本发明实施例的逻辑函数拆分算法，主要用于将原有非线性部件的某个逻辑函数配置序列拆分成d个相互独立的逻辑函数配置序列。其中任意d-1个的异或值与拆分前的配置序列相互独立，d个配置序列的异或值与拆分前相等。

[0057] 所述的逻辑函数调整算法，主要用于将一个2^k比特的逻辑函数配置序列C根据另一个k比特整数t进行调整，得到新的逻辑函数配置序列C'，其中第u($0\leqslant u\leqslant 2^k-1$)个比特满足$(C')_u=(C)_{u\oplus t}$。

[0058] 所述的逻辑函数刷新算法，主要用于刷新d个逻辑函数配置序列，保证每个配置序列之间的独立性，从而确保PFD方案的安全性。逻辑函数刷新算法d个输入的异或与d个输出相等，同时每个逻辑函数配置序列与刷新前的相互独立。

[0059] 本发明实施例的完整PFD方案，主要用于输入$x_1,x_2,\cdots,x_n$，输出独立随机的$y_1,y_2,\cdots,y_n$。PFD方案首先将加密算法的非线性部件的逻辑函数配置序列进行拆分，依次根据$x_1,x_2,\cdots,x_{n-1}$进行逻辑函数调整，每次使用新的x_i调整所有逻辑函数配置序列后，进行一次刷新操作，最后通过x_n查询最终的逻辑函数配置序列，从而得出$y_1,y_2,\cdots,y_n$。

[0060] 本发明实施例所提出的PFD方案中具体的逻辑函数拆分算法，逻辑函数调整算法，逻辑函数刷新算法以及完整PFD方案，具体描述如下。

[0061] (1)逻辑函数拆分算法

[0062] 逻辑函数拆分算法以逻辑函数的配置序列C为输入，根据掩码阶数d拆分成d个相同长度的逻辑函数配置序列，其中任意d-1个配置序列的异或都与原始逻辑函数配置序列独立，全部d个配置序列的异或与C相等。逻辑函数拆分算法包括如下步骤：

[0063] 步骤1，生成一个长度为2^k比特的整数C_1，令$C_1=C$。用一个整数i作为循环变量，初始值为2；

[0064] 步骤2,生成一个长度为2^k比特的变量随机整数r,令$C_i=r$。用C_1和C_i的异或代替C_1,即令$C_1=C_1\oplus C_i$。将循环变量i增加1,即令i=i+1,若$i\leq d$,重复执行步骤2;

[0065] 步骤3,输出$C_1,C_2,\ldots,C_d$。

[0066] 逻辑函数拆分算法的d个输出中C_1满足

[0067] $C_1=C\oplus C_2\oplus\cdots\oplus C_d$;

[0068] 因此,d个输出的异或满足

[0069] $C_1\oplus C_2\oplus\cdots\oplus C_d=C$。

[0070] (2)逻辑函数调整算法;

[0071] 逻辑函数调整算法的过程如图1所示。如图1中所示,算法以一个2^k比特的逻辑函数配置序列C和一个k比特共享因子t为输入,根据t调整C中每比特的位置,得到新的逻辑函数配置序列C',满足C'中的任意比特$(C')_u(0\leq u\leq 2^k-1)$与C中第$u\oplus t$比特$(C)_{u\oplus t}$相等。逻辑函数调整算法包含如下步骤:

[0072] 步骤1,生成一个长度为2^k比特的整数C'=0。用整数i作为循环变量,初始值为0;

[0073] 步骤2,令C'中的第i比特$(C')_i=(C)_{u\oplus t}$。将循环变量i增加1,即令i=i+1。若$i\leq 2^k$,重复执行步骤2;

[0074] 步骤3,输出新的逻辑函数配置序列C'。

[0075] (3)逻辑函数刷新算法;

[0076] 逻辑函数刷新算法的过程如图2所示。在图2中,d个2^k比特的逻辑函数配置序列$C_1,C_2,\ldots,C_d$为算法的输入,输出新的逻辑函数配置序列$C'_1,C'_2,\ldots,C'_d,tmp_1,tmp_2,\ldots,tmp_{d-1}$表示d-1个$2^k$比特的随机逻辑函数配置序列。逻辑函数刷新算法满足d个输入的异或与d个输出的异或相等,并且每个输出$C'_i(1\leq i\leq d)$都和对应的输入C_i独立无关。逻辑函数刷新算法包含如下步骤:

[0077] 步骤1,生成一个长度为2^k比特的整数C'_1,令$C'_1=C_1$。用整数i作为循环变量,初始值为2;

[0078] 步骤2,生成一个长度为2^k比特的随机整数C'_i。用C'_1与C'_i的异或代替C'_1,即令$C'_1=C'_1\oplus C'_i$。将循环变量i增加1,即令i=i+1。若$i\leq d$,重复执行步骤2;

[0079] 步骤3,输出$C'_1,C'_2,\ldots,C'_d$。

[0080] (4)PFD方案;

[0081] PFD方案以k'个长度为2^k比特的逻辑函数配置序列$S_t(1\leq t\leq k')$和d个k比特输入共享因子$x_1,x_2,\cdots,x_d$为输入,得到d个k比特输出共享因子$y_1,y_2,\cdots,y_d$。其中,$S_t(1\leq t\leq k')$表示原始加密算法中S盒中第t个输出比特对应的逻辑函数配置序列,输入共享因子满足$x=x_1\oplus x_2\oplus\cdots\oplus x_d$,输出共享因子满足$y=S(x)=y_1\oplus y_2\oplus\cdots\oplus y_d$。PFD方案包含如下步骤:

[0082] 步骤1,用整数t作为循环变量,初始值为1;

[0083] 步骤2,将S_t作为输入执行逻辑函数拆分算法,得到d个逻辑函数配置序列$C_1,C_2,\ldots,C_d$。用整数i,k作为循环变量,初始值均为1;

[0084] 步骤3,将C_k和x_i作为输入执行逻辑函数调整算法,得到逻辑函数配置序列C'_k。将循环变量k增加1,即令k=k+1。若$k\leq n$,重复执行步骤3;

[0085] 步骤4,将步骤3中得到的d个逻辑函数配置序列$C'_1,C'_2,\ldots,C'_d$作为输入执行逻辑函数刷新算法,并替换原有的$C_1,C_2,\ldots,C_d$。将循环变量i增加1,即令i=i+1。若$i\leq n-1$,

重复执行步骤3,4;

[0086] 步骤5,用整数j作为循环变量,初始值为1;

[0087] 步骤6,令y_j的第k个比特等于逻辑函数配置序列C_j中第x_d个比特,即令$(y_j)_k=(C_j)_{xd}$。将循环变量j增加1,即令j=j+1。若j≤k',重复执行步骤6;

[0088] 步骤7,将循环变量t增加1,即令t=t+1。若t≤k,重复执行步骤2,3,4,5,6;

[0089] 步骤8,输出$y_1,y_2,\cdots,y_d$。

[0090] 应当理解的是,本说明书未详细阐述的部分均属于现有技术。

[0091] 应当理解的是,上述针对较佳实施例的描述较为详细,并不能因此而认为是对本发明专利保护范围的限制,本领域的普通技术人员在本发明的启示下,在不脱离本发明权利要求所保护的范围情况下,还可以做出替换或变形,均落入本发明的保护范围之内,本发明的请求保护范围应以所附权利要求为准。

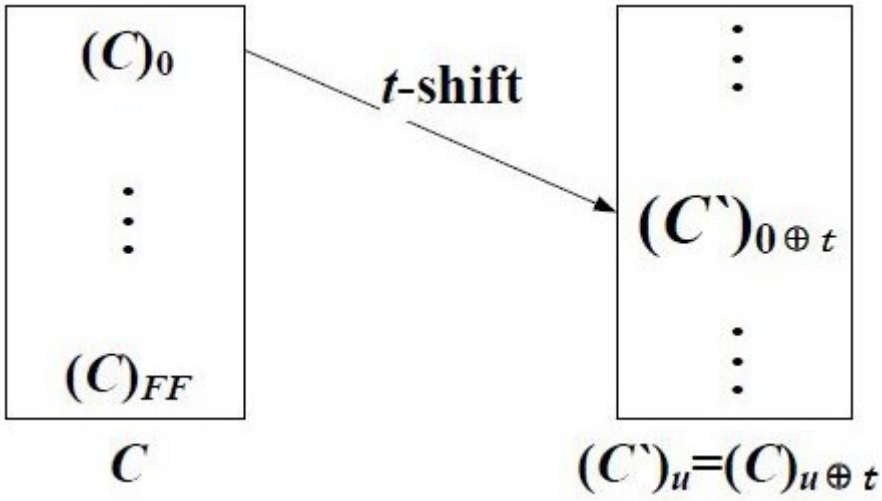

(C)0
t-shift
(C)FF
C
(C`)0⊕t
(C`)u=(C)u⊕t

图 1

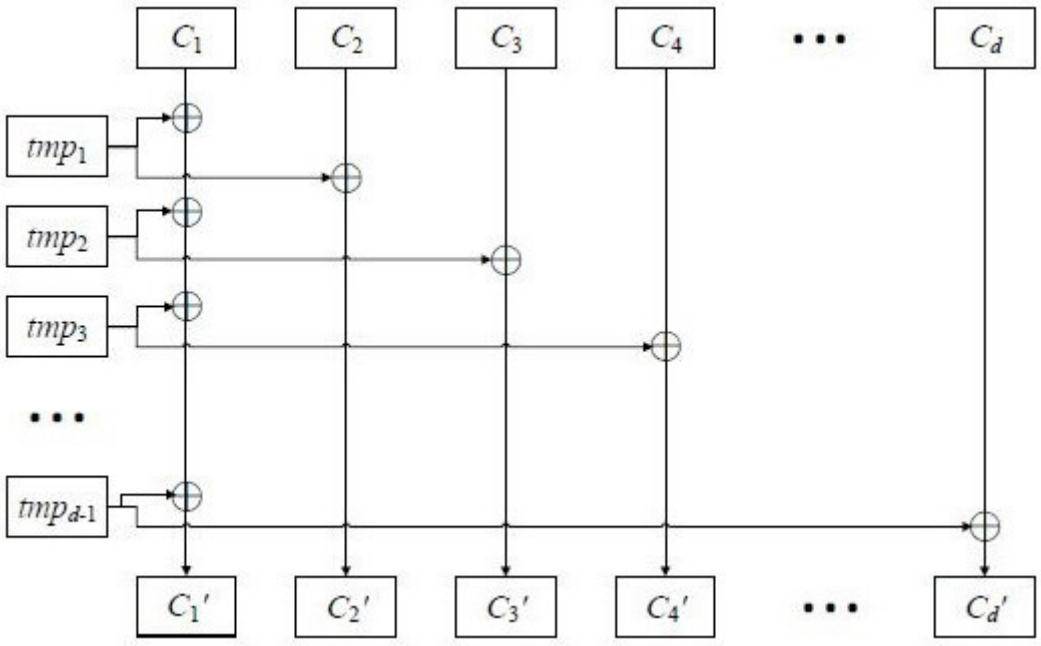

C1
C2
C3
C4
Cd
tmp1
tmp2
tmp3
tmpd-1
C1′
C2′
C3′
C4′
Cd′

图 2