(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 108123791 B
(45)授权公告日 2019.03.08

(21)申请号 201711428178.6

(22)申请日 2017.12.26

(65)同一申请的已公布的文献号
申请公布号 CN 108123791 A

(43)申请公布日 2018.06.05

(73)专利权人 衡阳师范学院
地址 421002 湖南省衡阳市珠晖区衡花路16号

(72)发明人 李浪 刘观良 邹祎 焦铬 邓红卫 刘沛林 李永超

(74)专利代理机构 长沙市融智专利事务所 43114
代理人 龚燕妮

(51)Int.Cl.
H04L 9/00(2006.01)
H04L 9/06(2006.01)
H04L 9/08(2006.01)

(56)对比文件
CN 104333446 A,2015.02.04,
CN 103746795 A,2014.04.23,
CN 105959107 A,2016.09.21,
CN 104639312 A,2015.05.20,
CN 106027221 A,2016.10.12,
US 5745577 A,1998.04.28,
陈利科 等."一种基于动态S-盒P-盒的快速分组密码算法——DSP".《计算机科学》.2009,第36卷(第2期),全文.

审查员 李文

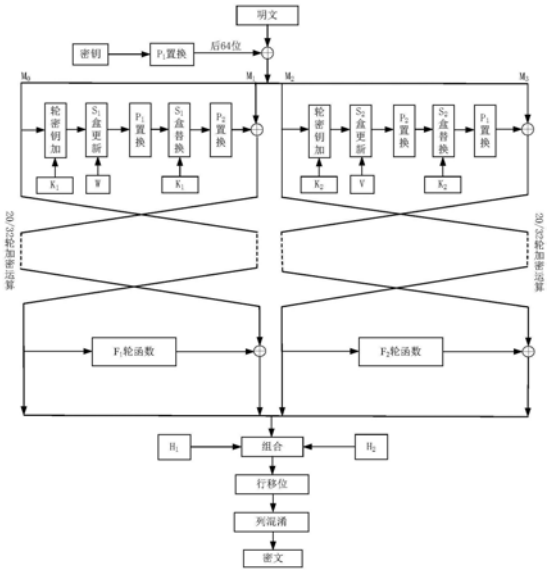
权利要求书3页 说明书16页 附图4页

(54)发明名称

一种轻量级分组密码SCS的实现方法与装置

(57)摘要

本发明公开了一种轻量级分组密码SCS的实现方法与装置,在密钥中划分轮密钥和控制密钥,轮密钥参与轮密钥加操作,控制密钥对每轮S盒的生成进行控制,从而得到随机S盒,控制密钥和轮密钥的更新与上一轮的运算结果有关,不仅每轮所使用的S盒是随机的,而且每轮的运算结果是随机的,能够增加混淆程度;在轮函数中每轮通过梅森旋转算法生成高伪随机P置换来实现扩散,轮函数迭代结束后再通过行移位和列混淆变换,利用这种双重扩散方式加大扩散效果,提高了安全性。本发明所述装置的内部结构相比固定密码结构在消耗资源差别不大的情况下,大幅度提高了该方案本身的安全性,能够在一定程度上增加对线性攻击、差分攻击等攻击的防御系数。



CN 108123791 B

1.一种轻量级分组密码SCS的实现方法,其特征在于,包括以下步骤:

步骤1:利用高伪随机P_1置换对密钥进行置换得到置换后的密钥,并从置换后的密钥中提取初始轮密钥、初始控制密钥和组合数据H_1、H_2;

步骤2:利用经过步骤1置换后的密钥的低64位对64位明文进行异或操作,得到第一中间结果数据,并将第一中间结果数据从高位至低位按16位一组分成4组,得到M_0、M_1、M_2、M_3;

步骤3:将第一中间结果数据的高32位和低32位按照Feistel结构分别进行r轮F_1轮函数和F_2轮函数运算;

F_1轮函数:将M_0作为参与F_1轮函数运算的输入数据,进行F_1轮函数运算,将得到的结果与M_1进行异或,将得到的异或运算结果作为下一轮参与F_1轮函数运算的输入数据M_0,同时将前一轮参与F_1轮函数运算的输入数据M_0作为下一轮的M_1;

F_2轮函数:将M_2作为参与F_2轮函数运算的输入数据,进行F_2轮函数运算,将得到的结果与M_3进行异或,将得到的异或运算结果作为下一轮参与F_2轮函数运算的输入数据M_2,同时将前一轮参与F_2轮函数运算的输入数据的M_2作为下一轮的M_3;

其中,所述F_1轮函数依次包括利用F_1轮函数运算的输入数据对每一轮的轮密钥进行异或操作、S_1盒更新、P_1置换、S_1盒替换以及P_2置换操作;

所述F_2轮函数依次包括利用F_2轮函数运算的输入数据对每一轮的轮密钥进行异或操作、S_2盒更新、P_2置换、S_2盒替换以及P_1置换操作;

所述S_1盒更新和S_2盒更新采用每一轮的控制密钥更新;

每一轮的轮密钥和控制密钥依据每一轮的轮函数运算输入数据分别对初始轮密钥和初始控制密钥进行更新获得;

步骤4:将经过r轮轮函数运算得到的M_1、M_0、M_3、M_2作为第二中间结果数据;

步骤5:把组合数据H_1、H_2分别放在第二中间结果数据的高32位的后面和低32位的后面,得到第三中间结果数据;

所述组合数据H_1和H_2从密钥中选取,且H_1和H_2均为32位;

步骤6:将第三中间结果数据依次进行行移位和列混淆操作,得到明文的加密结果。

2.根据权利要求1所述的方法,其特征在于,所述初始轮密钥、初始控制密钥和组合数据是从利用高伪随机P_1置换对密钥进行置换后的密钥中提取:

将置换后的密钥的第32位至63位作为初始轮密钥;

将置换后的密钥的低32位作为初始控制密钥;

若密钥长度为96位,则将置换后的密钥高32位作为组合数据H_1,H_1的逆序作为组合数据H_2;

若密钥长度为192位,则将置换后的密钥第96位至第159位的前半部分作为组合数据H_1,后半部分作为组合数据H_2;

将所述初始轮密钥的前半部分和后半部分分别作为第一初始轮密钥Lkey和第二初始轮密钥Rkey;

将所述初始控制密钥的前半部分和后半部分分别作为第一初始控制密钥wk和第二初始控制密钥vk;

每一轮F_1轮函数的控制密钥W由第一初始控制密钥wk和参与每一轮F_1轮函数运算的输入数据M_0进行异或得到;

每一轮F_2轮函数的控制密钥V由第二初始控制密钥vk和参与每一轮F_2轮函数运算的输入数据M_2进行异或得到；

每一轮F_1轮函数的轮密钥K_1由第一初始控制密钥wk、第一初始轮密钥Lkey以及参与每一轮F_2轮函数运算的输入数据M_2进行异或得到；

每一轮F_2轮函数的轮密钥K_2由第二初始控制密钥vk、第二初始轮密钥Rkey以及参与每一轮F_1轮函数运算的输入数据M_0进行异或得到。

3.根据权利要求2所述的方法，其特征在于，在每一轮F_1轮函数和F_2轮函数中S_1盒更新和S_2盒更新的过程由每一轮的控制密钥控制，S_1盒替换和S_2盒替换由每一轮的轮密钥控制；

所述S_1盒更新和S_2盒更新的过程相同，包括以下步骤：

步骤1.1：将每一轮的控制密钥的十进制数作为随机数种子，生成16个伪随机数；

步骤1.2：将得到的16个伪随机数相互异或，得到一个异或结果，记为dex；

步骤1.3：将得到的dex再次作为随机数种子，生成16个0到15之间的伪随机数，保存在数组d[i]中，$0\leqslant i\leqslant 15$；

步骤1.4：依次比较i与d[i]，若不相等，则交换初始S盒中i所在位置的值与d[i]所在位置值，i从0取值到15，直到完成所有交换，得到更新后的S盒；

所述S盒的初始值为$S(i)=i$，S_1盒更新和S_2盒更新使用的控制密钥分别为W和V；

所述S_1盒替换和S_2盒替换的过程相同，包括以下步骤：

步骤2.1：将待输入S盒的数据按从高位至低位，每组4位依次分为4组$\{state^j\}$，$0\leqslant j\leqslant 3$；

步骤2.2：将每一轮的轮密钥按从高位至低位，每组4位依次分为4组$\{K^j\}$，$0\leqslant j\leqslant 3$；

步骤2.3：依次进行$(state^j+K^j) \bmod 16$运算，$0\leqslant j\leqslant 3$，得到4个4位结果数据$\{s^j\}$；

步骤2.4：将步骤2.3得到的4个4位结果数据$\{s^j\}$均输入S盒进行变换，将得到的变换结果按照从高位至低位进行合并，得到S盒替换结果。

4.根据权利要求1-3任一项所述的方法，其特征在于，依据密钥长度，确定进行轮运算的轮数r；

若密钥长度为96位，轮数r为20；密钥长度为192位，轮数r为32。

5.根据权利要求4所述的方法，其特征在于，在对密文进行解密时，先将密文进行逆列混淆，再进行逆行移位，接着进行拆分操作，将得到的拆分结果采用广义Feistel结构进行相应轮数r的迭代，将迭代后结果利用经过P_1置换后的密钥的低64位进行异或运算，得到解密后的明文；

所述迭代过程与加密过程中的轮函数运算相同；

所述逆列混淆和逆行移位与加密过程中的列混淆和行移位运算互逆；

所述拆分操作是指将经过逆行移位运算后的结果的第64位到第95位和低32位取出后，剩余数据按照从高位至低位，每组16位，拆分成4组，依次为C_0、C_1、C_2、C_3，将拆分得到的数据作为迭代过程中的输入数据。

6.一种轻量级分组密码SCS的实现装置，其特征在于，包括：

初始化单元，利用高伪随机P_1置换对密钥进行置换得到置换后的密钥，并从置换后的密钥中提取初始轮密钥、初始控制密钥和组合数据H_1、H_2；

数据拆分单元，利用初始化单元中置换后的密钥的低64位对64位明文进行异或操作，

得到第一中间结果数据，并将第一中间结果数据从高位至低位按16位一组分成4组，得到M_0、M_1、M_2、M_3；

轮函数迭代单元，采用权利要求1-5任一项所述的方法将第一中间结果数据的高32位和低32位按照Feistel结构分别进行r轮F_1轮函数和F_2轮函数运算；

F_1轮函数模块：将M_0作为参与F_1轮函数运算的输入数据，进行F_1轮函数运算，将得到的结果与M_1进行异或，将得到的异或运算结果作为下一轮参与F_1轮函数运算的输入数据M_0，同时将前一轮参与F_1轮函数运算的输入数据M_0作为下一轮的M_1；

F_2轮函数：将M_2作为参与F_2轮函数运算的输入数据，进行F_2轮函数运算，将得到的结果与M_3进行异或，将得到的异或运算结果作为下一轮参与F_2轮函数运算的输入数据M_2，同时将前一轮参与F_2轮函数运算的输入数据的M_2作为下一轮的M_3；

其中，所述F_1轮函数依次包括利用F_1轮函数运算的输入数据对每一轮的轮密钥进行异或操作、S_1盒更新、P_1置换、S_1盒替换以及P_2置换操作；

所述F_2轮函数依次包括利用F_2轮函数运算的输入数据对每一轮的轮密钥进行异或操作、S_2盒更新、P_2置换、S_2盒替换以及P_1置换操作；

所述S_1盒更新和S_2盒更新采用每一轮的控制密钥更新；

每一轮的轮密钥和控制密钥依据每一轮的轮函数运算输入数据分别对初始轮密钥和初始控制密钥进行更新获得；

合并单元：将经过r轮轮函数运算得到的M_1、M_0、M_3、M_2作为第二中间结果数据，把组合数据H_1、H_2分别放在第二中间结果数据的高32位的后面和低32位的后面，得到第三中间结果数据；

所述组合数据H_1和H_2从密钥中选取，且H_1和H_2均为32位；

行列操作单元：将合并单元输出的第三中间结果数据依次进行行移位和列混淆操作，得到明文的加密结果。

一种轻量级分组密码SCS的实现方法与装置

技术领域

[0001] 本发明属于信息安全领域,特别涉及一种轻量级分组密码SCS的实现方法与装置。

背景技术

[0002] 近年来,随着网络、微电子和信息技术的飞跃性发展,物联网作为新一代信息化的典型代表,正在深入到人类生产和生活的各个方面,例如智能城市和交通、现代物流和环境监控等方面。而物联网领域的数据安全由于其自身资源受限、计算能力较弱和存储能力有限等问题不能由传统分组密码来解决,所以适应资源受限的轻量级分组密码算法应运而生。相关领域的学者也开始对轻量级密码进行大量研究,这些研究主要集中在轻量级密码的设计、安全性能的分析以及性能评估等方面。由于轻量级分组密码算法的研究刚刚起步,尚无统一标准与绝对领先地位的轻量级密码算法出现,同时,密码技术必须靠自主研发,针对我国信息安全实际情况,研发一种拥有自主知识产权的轻量级分组密码有着重要意义。

[0003] 从2011年开始,国际学术界就陆续发表了一些有关轻量级分组密码算法论文,学术界研究的一个前沿热点问题就是轻量级分组密码算法。一系列轻量级分组密码算法相继被提出,我国学者吴文玲和张蕾在ACNS 2011上提出的LBIock,以及龚征在RFID Sec 2011提出的KLEIN,密码硬件与嵌入式系统国际会议提出的Piccolo和LED(CHES 2011),国际信息安全应用会议(IWISA 2013)提出的LEA,国际安全、隐私和应用密码学工程会议(SPACE 2014)提出的Khudra等。适应资源约束环境下的轻量级分组加密算法正在受到全球密码研究人员和学者的广泛关注和研究。

[0004] 目前轻量级分组密码算法存在的问题有以下几个方面:

[0005] (1)分组密码主要采用以下两种结构:一是Feistel结构,其结构高度对称,一定程度上能够保证其加/解密的相似性且消耗资源较少,但该结构密码算法扩散速度慢,一轮迭代只能改变一半的分组数据,尤其以超级计算机为代表的一系列计算能力优越的机器和方法的实现,被破解的可能性增加,安全性能低;二是SPN结构,其结构清晰,混淆和扩散效果优越,但是由于其结构不对称,加/解密不相似,所以其消耗资源较多,效率相比之下也要低。

[0006] (2)当前轻量级分组密码存在追求低资源消耗而损失安全性的问题,例如很多轻量级算法为了追求更小的资源消耗而将轮函数中运算和置换模块设计的过于单一和简单,这样导致算法根本不具备抵抗现有多种技术相结合的旁路攻击方法,从而带来安全隐患。

[0007] (3)现有的一些轻量级算法采用一种固定密钥的方式,在轮函数中采用固定位的密钥去参与运算,即使提供了不同位数的密钥,但是在加/解密过程中除了运算轮数不一样外运算和置换的模块还是相同的,这样不仅没有大幅提高安全性,反倒大幅降低了算法的性能和效率,同时还加大了硬件的资源消耗。

[0008] (4)在轻量级算法一些加密模式中,加密的过程和运算变换模块是高度确定的,这种高度的确定性会给算法带来安全隐患。比如一些轻量级算法的S盒替换模块直接用固定的S盒去参与运算变换,同时其他运算模块也是按照固定的方式运算或者变换,在一定程度

上增加了被破解的可能性,降低了算法的安全性

发明内容

[0009] 本发明提供了一种轻量级分组密码SCS的实现方法与装置,其目的在于,克服现有技术中Feistel网络结构算法一轮迭代运算只能改变部分分组数据,扩散和混淆程度不高;密钥参与模块运算的控制过程和方法过于简单,密钥不同位数间的实现过程或者加密轮数不一样导致资源消耗太大;扩散和混淆方式过于简单且步骤繁杂,效率不高;算法加密过程和运算变换模块固定,导致被攻击和分析的可能性增加,安全性不高的问题。

[0010] 一种轻量级分组密码SCS的实现方法,包括以下步骤:

[0011] 步骤1:利用高伪随机P_1置换对密钥进行置换得到置换后的密钥,并从置换后的密钥中提取初始轮密钥、初始控制密钥和组合数据H_1、H_2;

[0012] 步骤2:利用经过步骤1置换后的密钥的低64位对64位明文进行异或操作,得到第一中间结果数据,并将第一中间结果数据从高位至低位按16位一组分成4组,得到M_0、M_1、M_2、M_3;

[0013] 步骤3:将第一中间结果数据的高32位和低32位按照Feistel结构分别进行r轮F_1轮函数和F_2轮函数运算;

[0014] F_1轮函数:将M_0作为参与F_1轮函数运算的输入数据,进行F_1轮函数运算,将得到的结果与M_1进行异或,将得到的异或运算结果作为下一轮参与F_1轮函数运算的输入数据M_0,同时将前一轮参与F_1轮函数运算的输入数据M_0作为下一轮的M_1;

[0015] F_2轮函数:将M_2作为参与F_2轮函数运算的输入数据,进行F_2轮函数运算,将得到的结果与M_3进行异或,将得到的异或运算结果作为下一轮参与F_2轮函数运算的输入数据M_2,同时将前一轮参与F_2轮函数运算的输入数据的M_2作为下一轮的M_3;

[0016] 其中,所述F_1轮函数依次包括利用F_1轮函数运算的输入数据对每一轮的轮密钥进行异或操作、S_1盒更新、P_1置换、S_1盒替换以及P_2置换操作;

[0017] 所述F_2轮函数依次包括利用F_2轮函数运算的输入数据对每一轮的轮密钥进行异或操作、S_2盒更新、P_2置换、S_2盒替换以及P_1置换操作;

[0018] 所述S_1盒更新和S_2盒更新采用每一轮的控制密钥更新;

[0019] 每一轮的轮密钥和控制密钥依据每一轮的轮函数运算输入数据分别对初始轮密钥和初始控制密钥进行更新获得;

[0020] 步骤4:将经过r轮轮函数运算得到的M_1、M_0、M_3、M_2作为第二中间结果数据;

[0021] 步骤5:把组合数据H_1、H_2分别放在第二中间结果数据的高32位的后面和低32位的后面,得到第三中间结果数据;

[0022] 所述组合数据H_1和H_2从密钥中选取,且H_1和H_2均为32位;

[0023] 步骤6:将第三中间结果数据依次进行行移位和列混淆操作,得到明文的加密结果。

[0024] 进一步地,所述初始轮密钥、初始控制密钥和组合数据是从利用高伪随机P_1置换对密钥进行置换后的密钥中提取:

[0025] 将置换后的密钥的第32位至63位作为初始轮密钥;

[0026] 将置换后的密钥的低32位作为初始控制密钥;

[0027] 若密钥长度为96位，则将置换后的密钥高32位作为组合数据H_1，H_1的逆序作为组合数据H_2；

[0028] 若密钥长度为192位，则将置换后的密钥第96位至第159位的前半部分作为组合数据H_1，后半部分作为组合数据H_2；

[0029] 将所述初始轮密钥的前半部分和后半部分分别作为第一初始轮密钥Lkey和第二初始轮密钥Rkey；

[0030] 将所述初始控制密钥的前半部分和后半部分分别作为第一初始控制密钥wk和第二初始控制密钥vk；

[0031] 每一轮F_1轮函数的控制密钥W由第一初始控制密钥wk和参与每一轮F_1轮函数运算的输入数据M_0进行异或得到；

[0032] 每一轮F_2轮函数的控制密钥V由第二初始控制密钥vk和参与每一轮F_2轮函数运算的输入数据M_2进行异或得到；

[0033] 每一轮F_1轮函数的轮密钥K_1由第一初始控制密钥wk、第一初始轮密钥Lkey以及参与每一轮F_2轮函数运算的输入数据M_2进行异或得到；

[0034] 每一轮F_2轮函数的轮密钥K_2由第二初始控制密钥vk、第二初始轮密钥Rkey以及参与每一轮F_1轮函数运算的输入数据M_0进行异或得到。

[0035] 进一步地，在每一轮F_1轮函数和F_2轮函数中S_1盒更新和S_2盒更新的过程由每一轮的控制密钥控制，S_1盒替换和S_2盒替换由每一轮的轮密钥控制；

[0036] 所述S_1盒更新和S_2盒更新的过程相同，包括以下步骤：

[0037] 步骤1.1：将每一轮的控制密钥的十进制数作为随机数种子，生成16个伪随机数；

[0038] 步骤1.2：将得到的16个伪随机数相互异或，得到一个异或结果，记为dex；

[0039] 步骤1.3：将得到的dex再次作为随机数种子，生成16个0到15之间的伪随机数，保存在数组d[i]中，$0\leqslant i\leqslant 15$；

[0040] 步骤1.4：依次比较i与d[i]，若不相等，则交换初始S盒中i所在位置的值与d[i]所在位置值，i从0取值到15，直到完成所有交换，得到更新后的S盒；

[0041] 所述S盒的初始值为$S(i)=i$，S_1盒更新和S_2盒更新使用的控制密钥分别为W和V；

[0042] 所述S_1盒替换和S_2盒替换的过程相同，包括以下步骤：

[0043] 步骤2.1：将待输入S盒的数据按从高位至低位，每组4位依次分为4组$\{state^j\}$，$0\leqslant j\leqslant 3$；

[0044] 步骤2.2：将每一轮的轮密钥按从高位至低位，每组4位依次分为4组$\{K^j\}$，$0\leqslant j\leqslant 3$；

[0045] 步骤2.3：依次进行$(state^j+K^j)\bmod 16$运算，$0\leqslant j\leqslant 3$，得到4个4位结果数据$\{s^j\}$；

[0046] 步骤2.4：将步骤2.3得到的4个4位结果数据$\{s^j\}$均输入S盒进行变换，将得到的变换结果按照从高位至低位进行合并，得到S盒替换结果。

[0047] 进一步地，依据密钥长度，确定进行轮运算的轮数r；

[0048] 若密钥长度为96位，轮数r为20；密钥长度为192位，轮数r为32。

[0049] 进一步地，在对密文进行解密时，先将密文进行逆列混淆，再进行逆行移位，接着进行拆分操作，将得到的拆分结果采用广义Feistel结构进行相应轮数r的迭代，将迭代后结果利用经过P_1置换后的密钥的低64位进行异或运算，得到解密后的明文；

[0050] 所述迭代过程与加密过程中的轮函数运算相同；

[0051] 所述逆列混淆和逆行移位与加密过程中的列混淆和行移位运算互逆；

[0052] 所述拆分操作是指将经过逆行移位运算后的结果的第64位到第95位和低32位取出后，剩余数据按照从高位至低位，每组16位，拆分成4组，依次为C_0、C_1、C_2、C_3，将拆分得到的数据作为迭代过程中的输入数据。

[0053] 一种轻量级分组密码SCS的实现装置，包括：

[0054] 初始化单元，利用高伪随机P_1置换对密钥进行置换得到置换后的密钥，并从置换后的密钥中提取初始轮密钥、初始控制密钥和组合数据H_1、H_2；

[0055] 数据拆分单元，利用初始化单元中置换后的密钥的低64位对64位明文进行异或操作，得到第一中间结果数据，并将第一中间结果数据从高位至低位按16位一组分成4组，得到M_0、M_1、M_2、M_3；

[0056] 轮函数迭代单元，采用上述的方法将第一中间结果数据的高32位和低32位按照Feistel结构分别进行r轮F_1轮函数和F_2轮函数运算；

[0057] F_1轮函数模块：将M_0作为参与F_1轮函数运算的输入数据，进行F_1轮函数运算，将得到的结果与M_1进行异或，将得到的异或运算结果作为下一轮参与F_1轮函数运算的输入数据M_0，同时将前一轮参与F_1轮函数运算的输入数据M_0作为下一轮的M_1；

[0058] F_2轮函数：将M_2作为参与F_2轮函数运算的输入数据，进行F_2轮函数运算，将得到的结果与M_3进行异或，将得到的异或运算结果作为下一轮参与F_2轮函数运算的输入数据M_2，同时将前一轮参与F_2轮函数运算的输入数据的M_2作为下一轮的M_3；

[0059] 其中，所述F_1轮函数依次包括利用F_1轮函数运算的输入数据对每一轮的轮密钥进行异或操作、S_1盒更新、P_1置换、S_1盒替换以及P_2置换操作；

[0060] 所述F_2轮函数依次包括利用F_2轮函数运算的输入数据对每一轮的轮密钥进行异或操作、S_2盒更新、P_2置换、S_2盒替换以及P_1置换操作；

[0061] 所述S_1盒更新和S_2盒更新采用每一轮的控制密钥更新；

[0062] 每一轮的轮密钥和控制密钥依据每一轮的轮函数运算输入数据分别对初始轮密钥和初始控制密钥进行更新获得；

[0063] 合并单元：将经过r轮轮函数运算得到的M_1、M_0、M_3、M_2作为第二中间结果数据，把组合数据H_1、H_2分别放在第二中间结果数据的高32位的后面和低32位的后面，得到第三中间结果数据；

[0064] 所述组合数据H_1和H_2从密钥中选取，且H_1和H_2均为32位；

[0065] 行列操作单元：将合并单元输出的第三中间结果数据依次进行行移位和列混淆操作，得到明文的加密结果。

[0066] 所述广义Feistel结构，在加密和解密过程中使用的结构相同。

[0067] 算法采用广义Feistel结构，加解密相似，结构对称提高效率，并且解密不需要构造逆S盒，易于实现。解密只需要将拆分、逆行移位和逆列混淆放到广义Feistel结构位置之前。

[0068] 有益效果

[0069] 本发明提供了一种轻量级分组密码SCS的实现方法与装置，采用了一种新的加密模式，将密钥划分出轮密钥和控制密钥两种，轮密钥参与轮函数F_1和F_2中的运算，控制密钥

对每轮S盒的生成进行控制;P置换的数据是通过梅森旋转算法生成高伪随机数据再进行筛选产生;列混淆对组合后128位数据采用伽罗瓦域下的乘法,加大扩散程度;F轮函数采用两个不同的函数F_1和F_2。算法的内部结构相比固定密码结构在消耗资源差别不大的情况下,大幅度提高了算法本身的安全性,能够在一定程度上增加对线性攻击、差分攻击等攻击的防御系数。SCS算法具有灵活性高、可扩展性强、低资源消耗和高随机性的特点,相比其他基于Feistel结构的轻量级算法安全和加密性能更优越。

[0070] 该模式基于广义Feistel网络结构,明文长度为64位,根据密钥长度96位和192位,迭代轮数分为20轮和32轮。SCS算法包括三个部分:拆分组合部分、运算部分和控制部分。拆分部分,将明文与密钥进行拆分组合;运算部分,轮函数运算包括两种运算模式,每个模式包括五个基本运算模块:轮密钥加、S盒更新、P_1置换、S盒替换、P_2置换,轮函数运算结束后还有一个扩位变换的处理,将迭代结果结合部分密钥位扩展至128位,然后再通过行移位和列混淆变换运算后得到密文;控制部分,由高位至低位,从第0位计算,对于96位密钥,将第65位到第79位和第80位到第95位作为SCS算法的初始控制密钥wk和vk,对于192位密钥,将第160位到第175位和第176位到第191位作为SCS算法的初始控制密钥wk和vk,控制密钥(W、V)是由初始控制密钥(wk、vk)和每轮的运算结果(M_0、M_2)决定的,S盒在轮函数F_1中称为S_1盒,在轮函数F_2中称为S_2盒,S_1盒与S_2盒是由每轮的控制密钥进行更新和生成,而控制密钥和轮密钥的更新又与每轮的运算结果有关,这样一来不仅每轮的运算结果是随机的,控制密钥与轮密钥也变得随机化,由此使得加/解密过程进一步随机化,这是一种新的加密方式,能够有效提高密码算法的安全性。

[0071] 利用本发明所述的SCS算法的广义Feistel结构设计可以节约很多硬件实现面积资源的开销,而且性能效率方面也比算法之间进行重构设计好很多。相比目前一些轻量级分组密码算法只是简单将轮函数中的置换模块进行顺序替换、使用单一S盒参与运算和变换、采用固定密钥参与轮函数中变换等方式去变换,SCS算法有如下优点:一是因其结构高度对称,加解密实现的资源相对较少;二是轮函数中S盒的设计采用的是每轮通过控制密钥控制其生成和更新,使每轮运算中的S盒不固定,控制密钥和轮密钥的更新与每轮的运算结果有关,这样一来每轮的运算结果是随机的,非线性变换程度大大提高,这是一种新的加密方式,安全性也进一步提升;三是不同轮函数里使用的P置换不同,完成20轮/32轮迭代后,把密文结合部分密钥位扩展为128位的密文进行行移位和列混淆变换,两种扩散方式结合密钥增加了扩散效果,提高了安全性。这种模式沿用Feistel结构实现资源少的优点,解密过程与加密相似,无需构造逆S盒,并且安全性也足以应对线性攻击和差分攻击等现有的一些攻击和分析手段。SCS密码算法是将现有算法暴露的一些缺点做了综合考虑和设计,从而在节约大量软硬件资源的基础上使得密码算法更有灵活性、可扩展性、随机性和安全性。

附图说明

[0072] 图1为本发明所述方法的加密过程示意图;

[0073] 图2为本发明所述轮密钥与控制密钥更新过程示意图;

[0074] 图3为本发明所述方法的解密过程示意图;

[0075] 图4为本发明所述方法的S_1盒更新过程示意图;

[0076] 图5为本发明所述方法的轮函数F_1流程示意图;

具体实施方式

[0077] 下面将结合附图和实例对本发明做进一步的说明。

[0078] 一种新型高安全的轻量级SCS分组密码实现方法，SCS算法明文长度为64位，密钥长度分为96位和192位两种，分别进行20轮和32轮函数迭代。SCS算法基于广义Feistel网络结构，轮函数F包括F_1、F_2两种轮函数，如图1所示。

[0079] F_1轮函数包含：轮密钥加(AddRoundKey)、S_1盒更新(SubUpdata1)、P_1置换(Permutation1)、S_1盒替换(SubCells1)、P_2置换(Permutation2)五个模块。

[0080] F_2轮函数包含：轮密钥加(AddRoundKey)、S_2盒更新(SubUpdata2)、P_2置换(Permutation2)、S_2盒替换(SubCells2)、P_1置换(Permutation1)五个模块。

[0081] 本发明所述的算法输入的密钥，经过高伪随机P_1置换，再划分成初始控制密钥(wk、vk)和初始轮密钥(Lkey、Rkey)。SCS算法经过密钥划分，通过更新控制密钥和轮密钥实现对S盒的更新，进而控制加/解密运算。

[0082] 在本发明所述的SCS密码算法中，先将明文与置换后的密钥后64位进行异或操作。再将每一轮值都分为4个单元，每个模块运算单元为16位，分别表示为M_0($state_0$～$state_{15}$)、M_1($state_{16}$～$state_{31}$)、M_2($state_{32}$～$state_{47}$)、M_3($state_{48}$～$state_{65}$)。将经高伪随机P_1置换变换的密钥划分为6组，分别为wk，vk，Lkey，Rkey，H_1，H_2，其中wk和vk是初始控制密钥划分，各为16位，对于96位密钥长度，wk(key_{64}～key_{79})，vk(key_{80}～key_{95})；对于192位密钥长度，wk(key_{160}～key_{175})，vk(key_{176}～key_{191})。初始轮密钥Lkey与Rkey，各为16位，其中F_1轮函数的轮密钥为Lkey(key_{32}～key_{47})，F_2轮函数的轮密钥为Rkey(key_{48}～key_{63})。在完成迭代20轮/32轮后，对于96位密钥长度，取密钥部分H_1(key_0～key_{31})和H_2(key_{31}～key_0)，与经过迭代的64位数据组合成128位；对于192位密钥长度，取密钥部分H_1(key_{96}～key_{127})和H_2(key_{128}～key_{159})，H_1与H_2各为32位。表1为密钥划分分组。轮密钥有两个作用部分，一个是参与F轮函数中轮密钥加模块的运算，另一个是在待加密数据进入S盒前先与轮密钥做相应的运算，得到结果再进入S盒进行替换，如图5所示；

[0083] 表1密钥划分分组

[0084]

	96位密钥划分分组	192位密钥划分分组
0—31（位）	H_1	（保留命名分组）
31—0（位）	H_2	（保留命名分组）
32—47（位）	$Lkey$	
48—63（位）	$Rkey$	
64—79（位）	wk	（保留命名分组）
80—95（位）	vk	（保留命名分组）
96—127（位）	空	H_1
128—159（位）		H_2
160—175（位）		wk
176—191（位）		vk

[0085] SCS算法的加密流程如图1所示。SCS密码算法加密描述如下算法1所示。

[0086] SCS分组密码算法加密伪代码描述：

[0087] 算法1：SCS算法加密过程，根据密钥长度96位或192位，N_R为20轮或32轮；

[0088] 输入：$M_{(64)}$，K；

[0089] 输出：$C_{(128)}$；

1: Permutation1 (K)

2: $K \rightarrow (wk, vk, Lkey, Rkey, H_1, H_2)$

3: $M(64) \leftarrow M(64) \oplus K$

4: $M(64) \rightarrow M_0(16) \,||\, M_1(16) \,||\, M_2(16) \,||\, M_3(16)$

5: for i=1 to NR do the following

[0090]

6:	Wschedule(wk, $M_{0(16)}$)	Vschedule(vk, $M_{2(16)}$)
7:	K1schedule (Lkey, wk, $M_{0(16)}$)	K2schedule (Rkey, vk, $M_{2(16)}$)
8:	$L_1 = M_{0(16)}$	$L_2 = M_{2(16)}$
9:	$M_{0(16)} \leftarrow M_{0(16)} \oplus K_{1(16)}$	$M_{0(16)} \leftarrow M_{0(16)} \oplus M_{2(16)}$
10:	SubUpdata$_1$（$W_{(16)}$）	SubUpdata$_2$（$V_{(16)}$）
11:	Permutation$_1$（$M_{0(16)}$）	Permutation$_2$（$M_{2(16)}$）

12: $SubCells_1$（$M_{0(16)}$） $SubCells_2$（$M_{2(16)}$）

13: $Permutation_2$（$M_{0(16)}$） $Permutation_1$（$M_{2(16)}$）

14: $M_{0(16)} \leftarrow M_{0(16)} \oplus M_{1(16)}$ $M_{2(16)} \leftarrow M_{2(16)} \oplus M_{3(16)}$

15: $M_{1(16)} = L_1$ $M_{3(16)} = L_2$

16: end for

[0091] 17: $M_{0(16)} \,||\, M_{1(16)} \,||\, M_{2(16)} \,||\, M_{3(16)} \rightarrow M_{1(16)} \,||\, M_{0(16)} \,||\, M_{3(16)} \,||\, M_{2(16)}$

18: $SplCom(M_1, M_0, H_1, M_3, M_2, H_2) \rightarrow C_{(128)}$;

19: $ShiftRows(C_{(128)})$

20: $MixColumns(C_{(128)})$

21:Return $C_{(128)}$

[0092] 以下对密钥进行置换、密钥划分和两部分轮函数中各个模块进行详细描述。

[0093] 轮函数F_1和F_2分别采用P_1-S_1-P_2和P_2-S_2-P_1这两种不同方式分别进行变换运，在密钥进行置换所用的高伪随机P置换表为P_1置换表，P_1置换表为按位变换，每次置换16位数据，对于96位密钥，按由高位至低位，将96位分成6组，每组16位，分别进行P_1置换；对于192位密钥，按由高位至低位，将192位分成12组，每组16位，分别进行P_1置换。P_1置换表数据如表5所示。

[0094] 对于密钥的划分，所划分出的16位初始轮密钥(Lkey、Rkey)与16位初始控制密钥(wk、vk)，需要进行各自的更新才能进入本轮的轮函数中，轮密钥与控制密钥更新过程如图2所示。划分密钥及轮密钥与控制密钥的更新具体如下：

[0095] 密钥划分分组如表1。划分的轮密钥，其中(key_{32}～key_{47})，记为Lkey；(key_{48}～key_{63})，记为Rkey，关系如下公式(1)(2)：

[0096] $Lkey_i = key_{32+i}$ $(0 \leqslant i \leqslant 15)$ (1)

[0097] $Rkey_i = key_{48+i}$ $(0 \leqslant i \leqslant 15)$ (2)

[0098] 若96位密钥长度，划分的控制密钥，其中(key_{64}～key_{79})，记为wk；(key_{80}～key_{95})，记为vk，关系如下公式(3)(4)：

[0099] $wk_i = key_{(64+i)}$ $(0 \leqslant i \leqslant 15)$ (3)

[0100] $vk_i = key_{(80+i)}$ $(0 \leqslant i \leqslant 15)$ (4)

[0101] 若192位密钥长度，划分的控制密钥，其中(key_{160}～key_{175})，记为wk；(key_{176}～key_{191})，记为vk，关系如下公式(5)(6)：

[0102] $wk_i = key_{(160+i)}$ $(0 \leqslant i \leqslant 15)$ (5)

[0103] $vk_i = key_{(176+i)}$ $(0 \leqslant i \leqslant 15)$ (6)

[0104] 轮密钥更新(K1schedule、K2schedule)：所划分的初始轮密钥在进入F轮函数之前需要进行密钥更新，将初始轮密钥Lkey与F_2轮函数的M_2异或，再与初始控制密钥wk异或，作为当前轮函数F_1的轮密钥K_1；将初始轮密钥Rkey与F_1轮函数的右明文M_0异或，再与初始控制密钥vk异或，作为当前轮函数F_2的轮密钥K_2。如图2所示。运算关系如下公式(7)(8)：

[0105] $$K_{1i} = Lkey_i \oplus M_2 \oplus wk_i \quad (0 \leq i \leq 15) \qquad (7)$$

[0106] $$K_{2i} = Rkey_i \oplus M_0 \oplus vk_i \quad (0 \le i \le 15) \qquad (8)$$

[0107] 控制密钥更新(Wschedule、Vschedule):将初始控制密钥wk与F_1轮函数的右明文M_0异或,得到的结果W作为当前轮函数F_1的控制密钥;将初始控制密钥vk与F_2轮函数的右明文M_2异或,得到的结果V作为当前轮函数F_2的控制密钥。如图2所示。运算关系如下公式(9)(10):

[0108] $$W_i = wk_i \oplus M_{0i} \quad (0 \le i \le 15) \qquad (9)$$

[0109] $$V_i = vk_i \oplus M_{2i} \quad (0 \le i \le 15) \qquad (10)$$

[0110] F_1轮函数加密运算模块描述:F_1轮函数将64位明文前半部分32位分为左右等长两半,左半16位记为M_0,右半16位记为M_1,轮函数F_1的详细设计结构如图5所示:

[0111] 轮密钥加(AddRoundKey):将16位的M_0值与轮密钥K_1值进行异或运算,运算关系如下公式(11):

[0112] $$M_{0i} = M_{0i} \oplus K_{1i} \quad (0 \le i \le 15) \qquad (11)$$

[0113] S_1盒更新(SubUpdata1):轮函数F_1中的S盒称为S_1盒,将控制密钥W的十进制数作为随机数的种子,先生成16个伪随机数,进行相互异或,得到一个运算结果dex,将该结果再作为随机数种子,生成16个0到15之间的伪随机数,保存在数组d[i](i<16)中,作为变换所需的数据,对初始S盒作用,初始S盒如公式(12)所示。分两次生成随机数目的是保证所生成的d[i]尽可能随机,不存在相关关系。在Feistel网络结构中,解密不需要额外写逆S_1盒,加解密相似,模块高度复用。每一轮S_1盒更新,通过控制密钥生成的16个数据d[i](i<16),对初始S盒进行更新。更新具体步骤:i从0开始,用下标i与下标所在位置的值d[i]比较,若不相等,则交换初始S盒中i所在位置的值与d[i]所在位置的值,直到完成所有交换。例如,对于16个伪随机数,从下标为0开始,d[0]与0对比,如果d[0]不等于0,则交换S_1[0]与S_1[d[0]]的值;接着到下标1,d[1]与1对比,如果d[1]不等于1,则交换S_1[1]与S_1[d[1]]的值,以此类推,直到变换完所有的数。如图4所示。通过控制密钥对初始S盒的更新进行生成新的S_1盒,从而使加密过程S_1盒随机化,不再是单一固定的一个S盒;

[0114] S_1={0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F} (12)

[0115] 密钥长度为96位时,经过两轮生成随机数运算生成的16个伪随机数,这里只列出前6轮如表2,以明文0000-0000-0000-0000,密钥0000-0000-0000-0000-0000-0000为例,生成对应的前6轮S_1盒如表3。

[0116] 表2经两轮生成随机数运算生成的16个伪随机数前6轮

[0117]

轮数	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1	13	11	1	14	8	7	3	4	3	6	10	5	15	2	8	10
2	7	15	13	13	15	3	12	6	13	8	4	10	12	10	0	10
3	0	7	12	2	11	13	3	3	13	11	10	7	7	2	1	14
4	11	4	10	10	5	14	5	14	6	4	6	9	6	14	11	4
5	5	15	4	6	12	8	14	15	2	5	6	6	3	3	11	13
6	0	10	8	10	6	13	4	6	9	9	3	6	9	15	0	3

[0118] 表3 96位密钥对应生成的前6轮S_1盒

[0119]

轮数	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1	d	2	0	4	5	1	9	8	3	e	c	7	f	b	6	a
2	e	f	d	5	a	2	0	c	9	3	4	1	6	b	7	8
3	0	e	8	1	b	d	c	2	5	4	a	6	9	3	f	7
4	b	4	a	2	f	6	c	1	e	0	8	d	3	7	5	9
5	5	f	0	d	c	9	b	1	4	8	e	3	6	7	a	2
6	e	a	8	5	4	d	b	6	9	c	1	7	2	f	0	3

[0120] 密钥长度为192位时，以明文0000-0000-0000-0000，密钥0000-0000-0000-0000-0000-0000-0000-0000-0000-0000-0000-0000为例，只列出生成的前6轮S_1盒如表4。

[0121] 表4 192位密钥生成的前6轮S_1盒

[0122]

轮数	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1	5	1	2	8	6	a	e	9	b	7	c	3	4	0	d	f
2	6	2	c	1	7	0	3	5	e	f	d	a	4	9	8	b
3	8	a	4	7	6	c	e	1	b	0	d	3	2	f	9	5
4	1	5	c	7	a	e	9	4	2	3	6	d	0	f	8	b
5	7	d	5	3	1	f	6	2	b	4	a	e	8	0	c	9
6	8	2	d	b	6	7	f	4	1	5	9	e	a	2	c	3

[0123] P_1置换(Permutation$_1$)与P_2置换(Permutation$_2$)：P_1、P_2置换变换按照表5、表6所示位置规则，将每一比特位的位置进行交换。由表5、表6位置规则得知，将进行P置换的16位数据每一比特位P(i)移动变换到i所表示的位置。表5、表6的数据采用梅森旋转算法随机产生，梅森旋转算法是一种随机函数算法，随机函数产生的数具有高强度的伪随机性，且算法所产生的数不重复，通过限制其随机数生成的范围来获取大量数据，最后通过手动筛选的方法选取两组混淆性较高的两组数据作为P_1、P_2置换表，轮函数F_1、F_2使用的P_1、P_2置换表相同。

[0124] 表5P_1置换表数据

[0125]

i	0	1	2	3	4	5	6	7
P_1[i]	4	10	2	12	6	0	9	3
i	8	9	10	11	12	13	14	15
P_1[i]	11	5	15	8	1	13	7	14

[0126] 表6P_2置换表数据

[0127]

i	0	1	2	3	4	5	6	7
P_2[i]	9	3	11	5	13	4	15	1
i	8	9	10	11	12	13	14	15
P_2[i]	10	2	8	14	7	0	6	12

[0128] S_1盒替换(SubCells$_1$):在广义Feistel网络结构中,S_1盒替换是最重要的部分之一,不同于传统的算法直接将待加密数据直接进入S盒中,而是将进行S_1盒变换的16位待加密数据分为4组,记为$state^0$、$state^1$、$state^2$、$state^3$,每组4位;轮密钥K_1的16位数据分成4组,每组4位,分别为K_1^0、K_1^1、K_1^2、K_1^3,分别与4组待加密数据进行十六进制加法运算,并对16取模,所得结果进入S_1盒进行替换,如图5所示。通过轮密钥与待加密数据运算,大大增加算法的混淆程度。

[0129] P_1、S_1、P_2模块之间的相互关系,根据图5,轮函数F_1详细设计结构所示。

[0130] F_2轮函数加密运算模块描述:F_2轮函数将64位明文后半部分32位分为左右等长两半,左半16位记为M_2,右半16位记为M_3:

[0131] 轮密钥加(AddRoundKey):将16位的M_2值与轮密钥K_2值进行异或运算,运算关系如下公式(13):

[0132] $$M_{2i} = M_{2i} \oplus K_{2i} \quad (0 \le i \le 15) \qquad (13)$$

[0133] S_2盒更新(SubUpdata$_2$):S_2盒的更新与S_1盒更新相同,将控制密钥V作为随机数的种子,经过两轮生成随机数运算,生成16个伪随机数保存在d[i]中(i<16),i从0开始,若下标i与下标所在位置的值d[i]不相等,则交换初始化S盒中i所在位置的值与d[i]所在位置的值,直到完成所有交换,生成新的S_2盒,通过控制密钥对S_2盒的更新进行控制,从而使加密过程S_2盒随机化,不再是单一固定的S盒;

[0134] S_2盒替换(SubCells$_1$):S_2盒与S_1盒的替换方式相同,数据不同,S_2盒是通过控制密钥V,作为随机数种子,经两轮运算生成16个伪随机数,控制S_2盒的更新,从而使每一轮有不同的S_2盒。将进行S_2盒变换的16位待加密数据分为4组,记为$state^0$、$state^1$、$state^2$、$state^3$,每组4位;轮密钥K_2的16位数据分成4组,每组4位,分别为K_2^0、K_2^1、K_2^2、K_2^3,分别与4组待加密数据进行十六进制加法运算,并对16取模,所得结果进入S_2盒进行替换。

[0135] 组合(SplCom):将M_0,M_1,M_2,M_3,H_1,H_2'按一定顺序组合,获得128位数据,H_1,H_2取值如表1所示,其中96位长度的密钥,H_2为H_1的逆排列。组合顺序如下公式(14):

[0136] $C_{128}=M_1||M_0||H_1||M_3||M_2||H_2$ (14)

[0137] 行移位变换(ShiftRows):通过与部分密钥位组合成的128位加密数据,先分成16组,每组8位,即每组一个字节,形成一个4×4的矩阵,进行行移位变换,具体方法,第一行不变,第二行循环左移1个字节,第三行循环左移2个字节,第四行循环左移3个字节。

[0138] 列混淆变换(MixColumns):可以通过改变公式(15)来改变生成的列变换的值,公式(16)可以通过改变GF(2^8)域上的多项式的系数,确定列混淆矩阵;

[0139] $S'(x)=C(x)\cdot S(x) \bmod (x^4+1)$ (15)

[0140] $C(x)=\{03\}\cdot x^3+\{02\}\cdot x^2+\{01\}\cdot x+\{02\}$ (16)

[0141] GF(2^8)域上的多项式:4个字节构成的向量可以表示为系数在GF(2^8)域上的次数小于4的多项式。规定多项式的乘法运算必须要取模$M(x)=x^4+1$,这样使得次数小于4的多项式的乘积仍然是一个次数小于4的多项式,将多项式的模乘运算记为⊗,设如公式(17)(18)(19):

[0142] $a(x)=a_3x_3+a_2x_2+a_1x+a_0$ (17)

[0143] $b(x)=b_3x_3+b_2x_2+b_1x+b_0$ (18)

[0144] $$c(x) = a(x) \otimes b(x) = c_3x_3 + c_2x_2 + c_1x + c_0 \quad (19)$$

[0145] 由于$x^j \bmod (x^4+1) = x^{j \bmod 4}$,所以如公式(20):

[0146] $$c_0 = a_0b_0 \oplus a_3b_1 \oplus a_2b_2 \oplus a_1b_3$$

[0147] $$c_1 = a_1b_0 \oplus a_0b_1 \oplus a_3b_2 \oplus a_2b_3$$

[0148] $$c_2 = a_2b_0 \oplus a_1b_1 \oplus a_0b_2 \oplus a_3b_3$$

[0149] $$c_3 = a_3b_0 \oplus a_2b_1 \oplus a_1b_2 \oplus a_0b_3 \quad (20)$$

[0150] 可将上述计算表示为(21):

[0151] $$\begin{bmatrix} c_0 \\ c_1 \\ c_2 \\ c_3 \end{bmatrix} = \begin{bmatrix} a_0 & a_3 & a_2 & a_1 \\ a_1 & a_0 & a_3 & a_2 \\ a_2 & a_1 & a_0 & a_3 \\ a_3 & a_2 & a_1 & a_0 \end{bmatrix} \begin{bmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \end{bmatrix} \quad (21)$$

[0152] M(x)不是GF(2^8)上的不可约多项式,因此非零多项式的这种乘法不是群运算。对于多项式b(x),这种乘法运算只限于乘以一个固有的有逆元的多项式如(22)所示:

[0153] $a(x) = a_3x_3 + a_2x_2 + a_1x + a_0$ (22)

[0154] 系数在GF(2^8)上的多项式$a_3x_3+a_2x_2+a_1x+a_0$是模x^4+1可逆的,当且仅当矩阵如下在GF(2^8)上可逆如式(23)所示:

[0155] $$\begin{bmatrix} a_0 & a_3 & a_2 & a_1 \\ a_1 & a_0 & a_3 & a_2 \\ a_2 & a_1 & a_0 & a_3 \\ a_3 & a_2 & a_1 & a_0 \end{bmatrix} \quad (23)$$

[0156] 根据公式(23)所示,使得M(x)矩阵模x^4+1可逆,SCS算法所使用的M(x)矩阵如公式(24)所示:

[0157] $$M(x) = \begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} \quad (24)$$

[0158] SCS算法的解密流程如图3所示。SCS密码算法解密描述如下算法2所示。

[0159] SCS分组密码算法解密伪代码描述:

[0160] 算法2:SCS算法解密过程,根据密钥长度96位或192位,N_R为20轮或32轮;

[0161] 输入:$C_{(128)}$,K;

[0162] 输出:$M_{(64)}$;

1: $Permutation_1$ (K)

2: $K \rightarrow (wk, vk, Lkey, Rkey, H_1, H_2)$

3: $InvMixColumns(C_{(128)})$

4: $InvShiftRows(C_{(128)})$

5: $InvSplCom(C_{(128)}) \rightarrow (C_0, C_1, C_2, C_3)$;

6: for i=1 to NR do the following

7:	$Wschedule(wk, C_{0(16)})$	$Vschedule(vk, C_{2(16)})$
8:	$K1schedule(Lkey, wk, C_{0(16)})$	$K2schedule(Rkey, vk, C_{2(16)})$
9:	$L_1 = C_{0(16)}$	$L_2 = C_{2(16)}$
[0163] 10:	$C_{0(16)} \leftarrow C_{0(16)} \oplus K_{1(16)}$	$C_{0(16)} \leftarrow C_{0(16)} \oplus C_{2(16)}$
11:	$SubUpdata_1(W_{(16)})$	$SubUpdata_2(V_{(16)})$
12:	$Permutation_1(C_{0(16)})$	$Permutation_2(C_{2(16)})$
13:	$SubCells_1(C_{0(16)})$	$SubCells_2(C_{2(16)})$
14:	$Permutation_2(C_{0(16)})$	$Permutation_1(C_{2(16)})$
15:	$C_{0(16)} \leftarrow C_{0(16)} \oplus C_{1(16)}$	$C_{2(16)} \leftarrow C_{2(16)} \oplus C_{3(16)}$
16:	$C_{1(16)} = L_1$	$C_{3(16)} = L_2$

17: end for

18: $C_{1(16)} || C_{0(16)} || C_{3(16)} || C_{2(16)} \rightarrow C_{0(16)} || C_{1(16)} || C_{2(16)} || C_{3(16)}$

19: $M_{(64)} \leftarrow C_{(64)} \oplus K$

[0164] 20:Return $M_{(64)}$

[0165] 本发明所述的SCS算法解密过程中广义Feistel网络结构使用与加密过程相同的模块，只需要将部分模块的顺序稍微变换一下，即可完成解密操作；轮函数外增加了一个行移位和列混淆变换的逆变换，相对于加密函数的各个组件顺序，将明文与密钥的异或调整到末尾位置，逆行移位和逆列混淆变换调整到顺序首位置，其他不变，解密过程与加密过程使用相同的初始轮密钥与初始控制密钥。

[0166] 逆行移位变换(InvShiftRow)：将128位待解密数据，先分成16组，每组8位，即每组一个字节，形成一个4×4的矩阵，进行逆行移位变换，具体方法，第一行不变，第二行循环右移1个字节，第三行循环右移2个字节，第四行循环右移3个字节。

[0167] 逆列混淆变换(InvMixColumns)：逆列混淆变换的处理方法与列混淆变换类似，矩阵变换公式如公式(25)：

[0168]
$$M^{-1}(x) = \begin{bmatrix} 0e & 0b & 0d & 09 \\ 09 & 0e & 0b & 0d \\ 0d & 09 & 0e & 0b \\ 0b & 0d & 09 & 0e \end{bmatrix} \qquad (25)$$

[0169] SCS算法测试数据如表7和表8所示：

[0170] 表7 SCS-96测试向量

[0171]

Plaintext	Key	Ciphertext
0000-0000-0000-0000	0000-0000-0000-0000-0000-0000	2DAA-A73D-2AFC-0B37-7497-1611-444A-2A60
0000-0000-0000-0000	FFFF-FFFF-FFFF-FFFF-FFFF-FFFF	E3EA-A53C-FD9E-D5A1-1839-9A05-CF63-FF8F
FFFF-FFFF-FFFF-FFFF	0000-0000-0000-0000-0000-0000	A973-D819-0430-2B44-4980-498F-DB92-55A0
FFFF-FFFF-FFFF-FFFF	FFFF-FFFF-FFFF-FFFF-FFFF-FFFF	7CDA-ADE2-48F8-43D5-3C36-C00D-572F-9500
0123-4567-89AB-CDEF	0123-4567-89AB-CDEF-0123-4567	0C06-5B5B-ED2A-3848-C66D-21B3-3EA4-579B

[0172] 表8 SCS-192测试向量

[0173]

Plaintext	Key	Ciphertext

[0174]

0000-0000-0000-0000	0000-0000-0000-0000-0000-0000-0000-0000-0000-0000-0000-0000	9DDF-2DEF-4DD7-77AE-01C7-611E-A538-6090
0000-0000-0000-0000	FFFF-FFFF-FFFF-FFFF-FFFF-FFFF-FFFF-FFFF-FFFF-FFFF-FFFF-FFFF	556A-43E9-8848-AA12-A1DB-4D58-E569-3E66
FFFF-FFFF-FFFF-FFFF	0000-0000-0000-0000-0000-0000-0000-0000-0000-0000-0000-0000	B923-E235-EBF4-5B43-A9E1-3145-98E8-A67C
FFFF-FFFF-FFFF-FFFF	FFFF-FFFF-FFFF-FFFF-FFFF-FFFF-FFFF-FFFF-FFFF-FFFF-FFFF-FFFF	E8B7-B409-BA10-B338-366C-F538-847D-32D3
0123-4567-89AB-CDEF	0123-4567-89AB-CDEF-0123-4567-0123-4567-89AB-CDEF-0123-4567	6824-FE82-A095-B24A-3C6A-44C9-C686-AD2B

[0175] 本发明所述的SCS-96密码算法在ModelSim SE 6.1f Evaluation上进行仿真;在Synopsys Design Compiler Version B-2008.09进行综合,其中综合工艺库为SMIC 0.18μm CMOS,在综合实验中,面积资源用等效门数GE来衡量。

[0176] SCS-96算法各组件硬件实现资源具体描述为:64位的明文保存在寄存器中需要344GE,密钥96位密钥保存在存放128位组合数据的寄存器中需要为688GE。密钥与明文的64位异或运算,需要64位异或单元,因此需要172GE。对F轮函数中的轮密钥加操作以及其他异或运算,均为16位异或操作,16位异或单元需要43GE。一个S盒替换模块占28GE,S盒的实现共需要28*(4+4)＝224GE。P置换模块与行移位模块,采用连线方式实现,硬件实现不需要消耗资源。列混淆模块,将部分乘法运算转换为异或与移位运算,可以减少实现资源,从而只需要消耗资源为40GE。算法实现中,控制逻辑单元以及计数器共需要40GE。SCS算法硬件实现仅需要1551GE。表9是SCS算法ASIC资源面积列表。

[0177] 表9SCS-96面积资源列表

[0178]

算法模块	GE
明文寄存器	344
密钥寄存器	688
64位异或单元	172
16位异或单元	43
S盒替换层	224
P置换层/行移位	0
列混淆层	40
控制逻辑单元与计数器	40
总和	1551

[0179] 满足不同用户多层次的安全性需求，采用两种密钥长度，96位长度的密钥更适合资源受限的环境，而192位长度的密钥主要用于更考虑安全因素的环境。算法采用结构高度对称的Feistel结构，通过将密钥划分成不同功能密钥参与轮函数的运算和控制S盒的生成等，同时轮函数中采用P置换且数据由大量不重复高伪随机数据筛选产生，在轮函数迭代完最后一轮后再将数据扩为128位进行一次行移位和列混淆变换从而进一步提高扩散性等。综上使得算法具有灵活性高、可扩展性强、低资源消耗和高随机性的特点，相比其他基于Feistel结构的轻量级算法安全和加密性能更加优越。

[0180] 表10为各轻量级分组密码算法ASIC硬件实现，通过表10的数据对比表明，SCS相比其他分组密码占用面积资源更小，适合于资源受限的环境。

[0181] 表10各分组密码算法ASIC实现

[0182]

算法	结构	分组长度(bits)	密钥长度(bits)	资源面积(GE)
PRESNET-80	SPN	64	80	1570
Twine-128	Feistel	64	128	1866
Piccolo-128	GFN	64	128	1938
CLEFIA	GFN	64	128	2488
mCRYPTON	SPN	64	128	2500
Cube Cipher	SPN	64	128	2536
PUFFIN	SPN	64	128	2577
SCS-96	GFN	64	96	1551
SCS-192	GFN	64	192	2583

[0183] 一种轻量级分组密码SCS的实现装置，包括：

[0184] 初始化单元，利用高伪随机P_1置换对密钥进行置换得到置换后的密钥，并从置换后的密钥中提取初始轮密钥、初始控制密钥和组合数据H_1、H_2；

[0185] 数据拆分单元，利用初始化单元中置换后的密钥的低64位对64位明文进行异或操作，得到第一中间结果数据，并将第一中间结果数据从高位至低位按16位一组分成4组，得到M_0、M_1、M_2、M_3；

[0186] 轮函数迭代单元，采用上述的方法将第一中间结果数据的高32位和低32位按照

Feistel结构分别进行r轮F_1轮函数和F_2轮函数运算；

[0187] F_1轮函数模块：将M_0作为参与F_1轮函数运算的输入数据，进行F_1轮函数运算，将得到的结果与M_1进行异或，将得到的异或运算结果作为下一轮参与F_1轮函数运算的输入数据M_0，同时将前一轮参与F_1轮函数运算的输入数据M_0作为下一轮的M_1；

[0188] F_2轮函数：将M_2作为参与F_2轮函数运算的输入数据，进行F_2轮函数运算，将得到的结果与M_3进行异或，将得到的异或运算结果作为下一轮参与F_2轮函数运算的输入数据M_2，同时将前一轮参与F_2轮函数运算的输入数据的M_2作为下一轮的M_3；

[0189] 其中，所述F_1轮函数依次包括利用F_1轮函数运算的输入数据对每一轮的轮密钥进行异或操作、S_1盒更新、P_1置换、S_1盒替换以及P_2置换操作；

[0190] 所述F_2轮函数依次包括利用F_2轮函数运算的输入数据对每一轮的轮密钥进行异或操作、S_2盒更新、P_2置换、S_2盒替换以及P_1置换操作；

[0191] 所述S_1盒更新和S_2盒更新采用每一轮的控制密钥更新；

[0192] 每一轮的轮密钥和控制密钥依据每一轮的轮函数运算输入数据分别对初始轮密钥和初始控制密钥进行更新获得；

[0193] 合并单元：将经过r轮轮函数运算得到的M_1、M_0、M_3、M_2作为第二中间结果数据，把组合数据H_1、H_2分别放在第二中间结果数据的高32位和低32位的后面，得到第三中间结果数据；

[0194] 所述32位组合数据H_1和H_2从密钥中选取；

[0195] 行列操作单元：将合并单元输出第三中间结果数据依次进行行移位和列混淆操作，得到明文的加密结果。

[0196] 以上结合具体实施例对本发明进行了详细的说明，这些并非构成对发明的限制。在不脱离本发明原理的情况下，本领域的技术人员还可以做出许多变形和改进，这些也应属于本发明的保护范围。

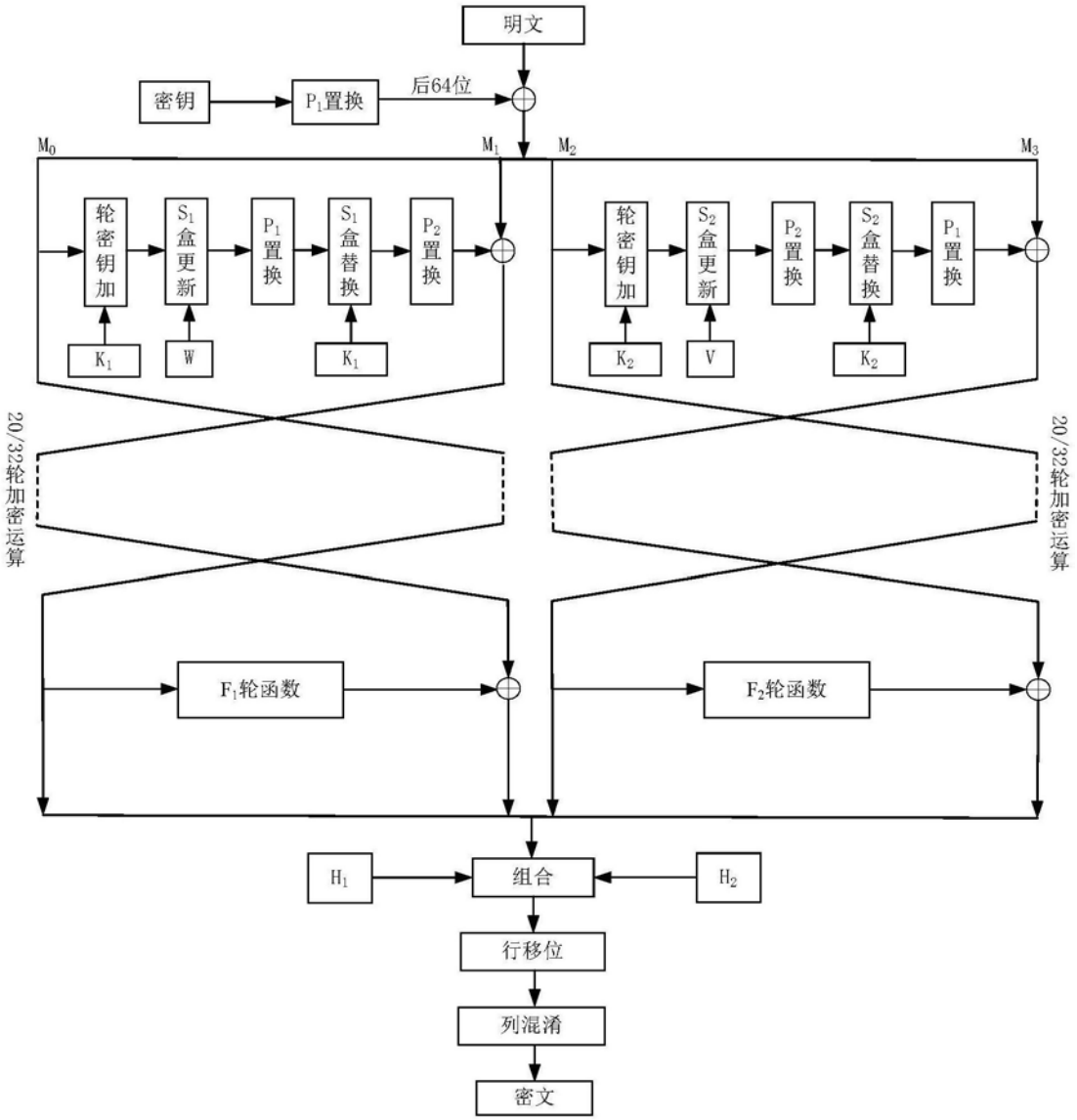

明文
密钥
P1置换
后64位
M0
M1
M2
M3
轮密钥加
S1盒更新
P1置换
S1盒替换
P2置换
轮密钥加
S2盒更新
P2置换
S2盒替换
P1置换
K1
W
K1
K2
V
K2
20/32轮加密运算
20/32轮加密运算
F1轮函数
F2轮函数
H1
组合
H2
行移位
列混淆
密文

图1

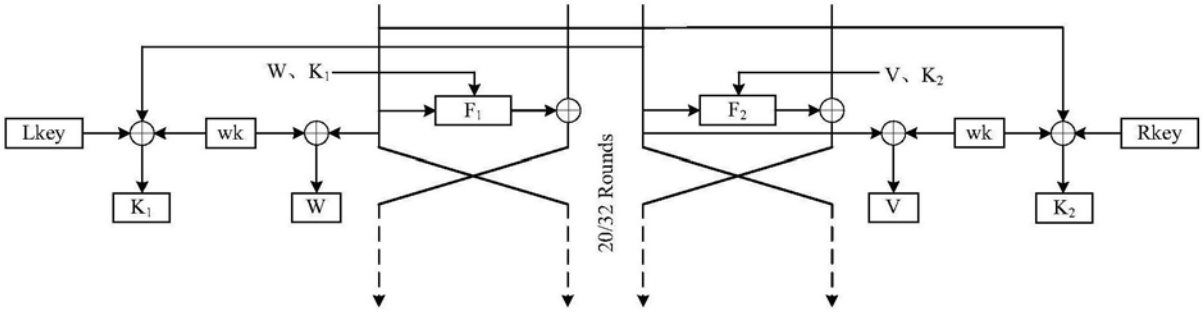

W、K1
V、K2
Lkey
wk
F1
F2
wk
Rkey
K1
W
20/32 Rounds
V
K2

图2

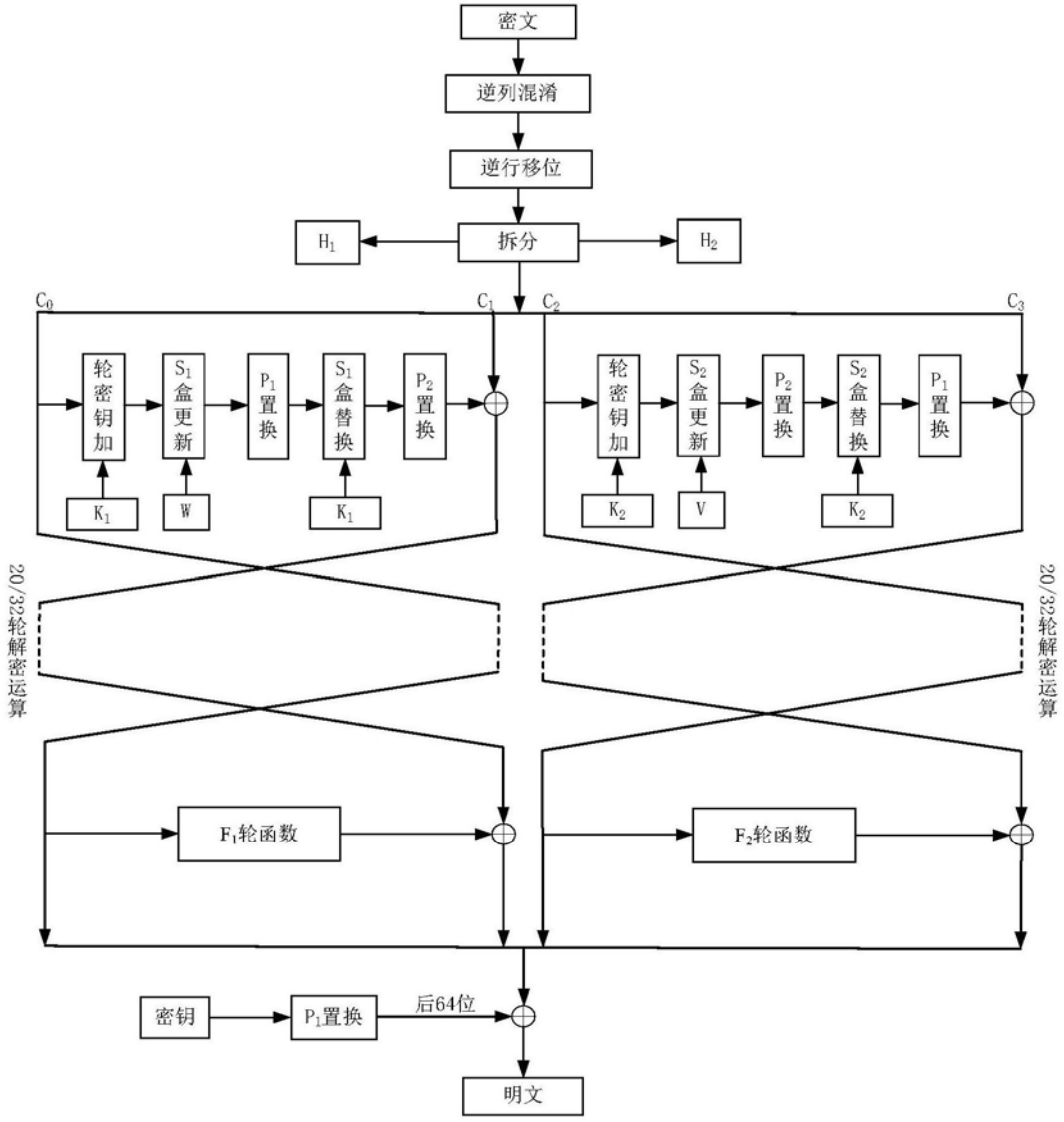

密文
逆列混淆
逆行移位
拆分
H1
H2
C0
C1
C2
C3
轮密钥加
S1盒更新
P1置换
S1盒替换
P2置换
K1
W
K1
轮密钥加
S2盒更新
P2置换
S2盒替换
P1置换
K2
V
K2
20/32轮解密运算
20/32轮解密运算
F1轮函数
F2轮函数
密钥
P1置换
后64位
明文

图3

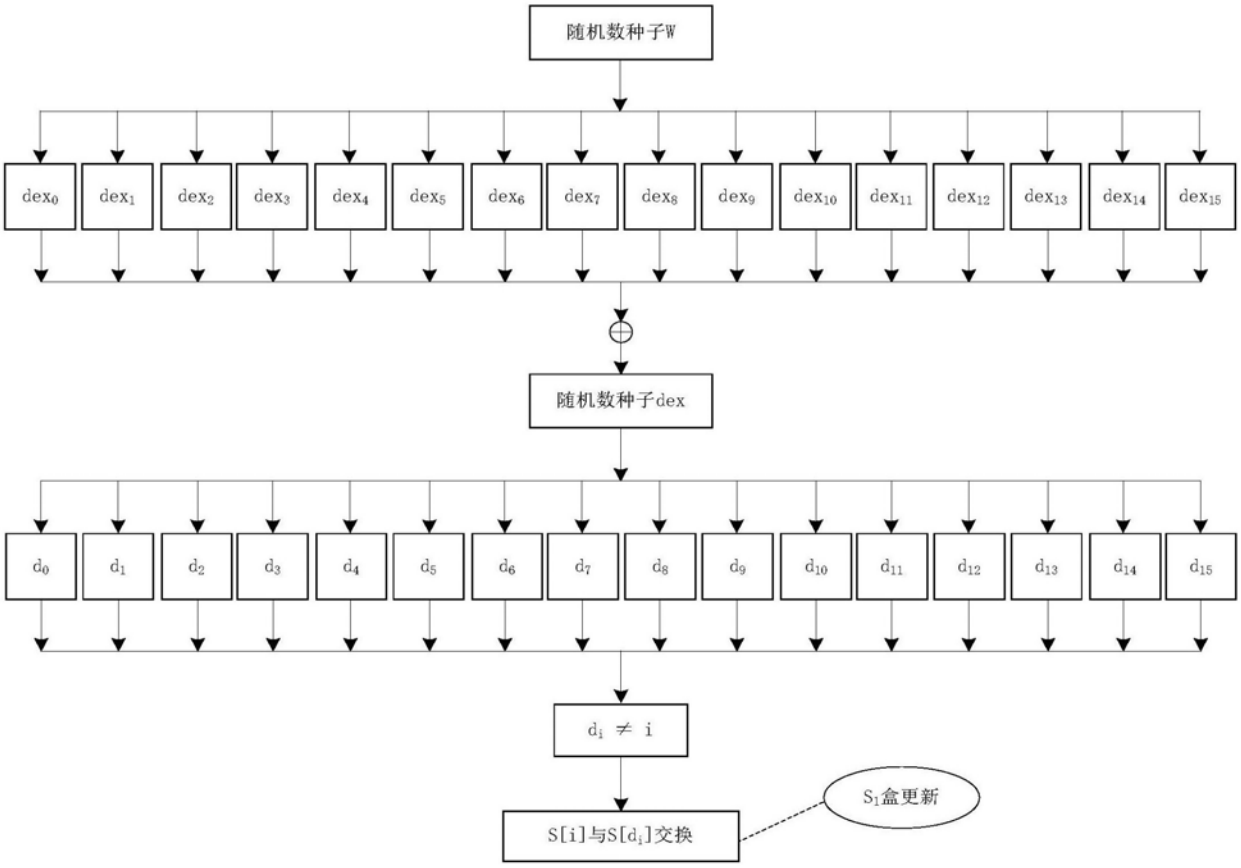

随机数种子W
dex0 dex1 dex2 dex3 dex4 dex5 dex6 dex7 dex8 dex9 dex10 dex11 dex12 dex13 dex14 dex15
随机数种子dex
d0 d1 d2 d3 d4 d5 d6 d7 d8 d9 d10 d11 d12 d13 d14 d15
di ≠ i
S[i]与S[di]交换
S1盒更新

图4

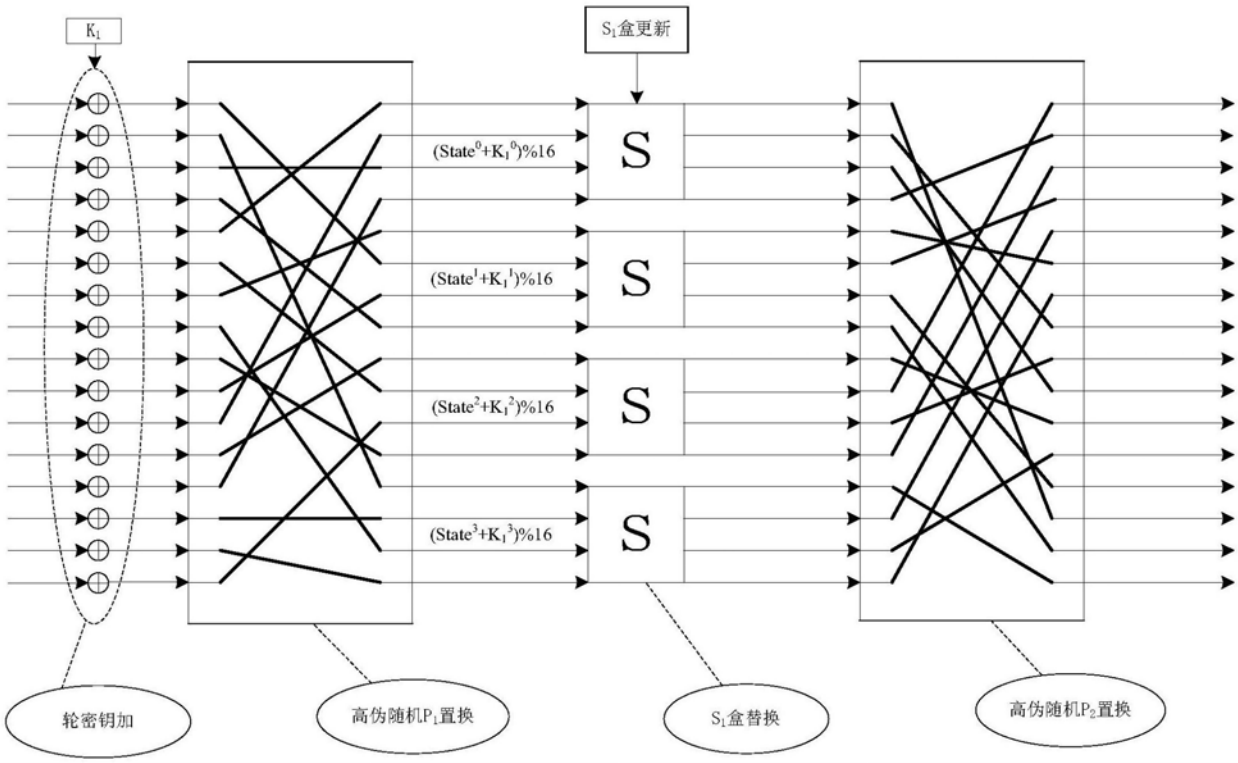

K1
S1盒更新
(State0+K1 0)%16
(State1+K1 1)%16
(State2+K1 2)%16
(State3+K1 3)%16
S
S
S
S
轮密钥加
高伪随机P1置换
S1盒替换
高伪随机P2置换

图5