Surge:一种新型、低资源、高效的轻量级分组密码算法

李 浪[1,2] 刘波涛[1]
(衡阳师范学院计算机科学与技术学院 湖南 衡阳 421002)[1]
(湖南大学信息科学与工程学院 长沙 410082)[2]

摘 要 目前,适合资源约束的轻量级密码算法已成为研究热点。提出一种低资源、高性能与高安全性的新轻量级分组密码算法 Surge。Surge 密码分组长度为 64 位,使用 64 位、80 位和 128 位 3 种密钥长度,且基于 SPN 结构。轮函数分为 5 个模块,密钥扩展模块采用无扩展方式;轮常数加模块采用 0 到 15 的数字组合成轮常数,构造高效且高度混淆的轮常数加变换;列混合模块利用易于硬件实现的(0,1,2,4)组合矩阵,从而可以在有限域 GF(2^4)上构造硬件实现友好型矩阵。将 Surge 算法在 FPGA 上进行了实现,实验结果表明,相对于目前 SPN 结构的轻量级密码算法,Surge 算法占用的面积资源更小,同时有着良好的加密性能;安全性实验证明了 Surge 可以有效抗差分与线性攻击、代数攻击。

关键词 轻量级分组密码算法,FPGA 实现,差分攻击,线性攻击,代数攻击
中图法分类号 TP309 **文献标识码** A **DOI** 10.11896/j.issn.1002-137X.2018.02.041

Surge:A New Low-resource and Efficient Lightweight Block Cipher

LI Lang[1,2] LIU Bo-tao[1]
(College of Computer Science and Technology,Hengyang Normal University,Hengyang,Hunan 421002,China)[1]
(College of Information Science and Engineering,Hunan University,Changsha 410082,China)[2]

Abstract Lightweight cryptography algorithm has become a hot research. The paper presented a new lightweight block cipher algorithm named Surge. Surge has low resource,high performance and high security. Block length of Surge cipher is 64 bits. Its variable key uses 64,80 or 128-bit length. Surge is based on the SPN structure. The round function is divided into 5 modules. Key expansion module is no expansion. Round-constants add module uses 0 to 15 to combine so that it can achieve efficient and highly confused round-constrants add operation. MixColumn module uses (0,1,2,4) to composite hardware-friendly matrix on the GF (2^4). Low resource and highly efficient of Surge is attained by this novel design. Surge is implemented and downloaded in FPGA. Experimental results show that it has smaller area resources and better cryptographic properties. The security expriment proves that surge can be against differential and linear attacks,algebraic attacks.

Keywords Lightweight block cipher,FPGA implementation,Differential attacks,Linear attacks,Algebraic attacks

1 引言

从 2006 年开始,学术界就陆续发表了一些有关轻量级分组密码算法设计的论文。一些有影响的轻量级分组密码算法有:Hong D 等[1]在密码硬件与嵌入式系统国际会议(CHES 2006)上提出的 HIGHT;BOGDANOV A 等[2]在 CHES 2007 上提出的 PRESENT(CHES 2007),由于其安全与高效的优点,PRESENT 成为了轻量级分组密码算法的一个标杆,但其占用资源仍然较多;CHENG H 等[3]在 EUROMICRO 2008 上提出的 PUFFIN;IZADI M 等[4]在中美高级网络技术研讨会(CANS 2009)上提出的 MIBS;Shibutani K[5]在 CHES2011 上提出的 Piccolo;Guo J[6]在 CHES2011 上提出的 LED;我国学者吴文玲和张蕾在密码学应用和网络安全国际会议(ACNS 2011)上提出的 LBlock[7]密码算法;龚征在 RFID 安全国际会议(RFIDSEC 2011)上提出的 KLEIN[8]密码算法等。

本文提出一种低资源、高性能与高安全性的轻量级分组密码算法,取名为 Surge。该密码算法保持了 AES 算法的设计原则,有着与 AES 相似的差分特征概率和最佳线性逼近优势。相对于目前已有的 SPN 结构轻量级密码,Surge 算法的实现面积更小,同时能保持相当高的性能。

2 Surge 密码算法的设计原理

Surge 密码算法采用 SPN 结构,分组长度为 64-bit,密钥长度设计为 64-bit,80-bit 和 128-bit 3 种,对应记为 Surge-64,Surge-80 和 Surge-128,迭代轮数 N_R 分别为 32,36 和 40。算法加密轮运算中包含常数加(AddConstants)、轮密钥加(Add-

到稿日期:2016-12-13 返修日期:2017-02-15 本文受国家自然科学基金(61572174),湖南省教育厅科研资助(15A029),衡阳师范学院产学研基金(16CXYZ01),湖南省科技计划项目(2016TP1020),湖南省自然科学基金资助项目(2015JJ4011,2017JJ4001)资助。

李 浪(1971—),男,博士,教授,CCF 会员,主要研究方向为嵌入式系统与信息安全,E-mail:lilang911@126.com(通信作者);刘波涛(1991—),男,硕士生,主要研究方向为嵌入式系统与信息安全。

RoundKey)、S 盒替换(SubCells)、行移位(ShiftRows)、列混合(MixColumns) 5 个模块。算法解密轮运算包含列混合逆变换(InvMixColumns)、行移位逆变换(InvShiftRows)、S 盒替换逆变换(InvSubCells)、轮密钥加变换(AddRoundKey)和常数加逆变换(InvAddConstants) 5 个模块。

Surge 加密运算的流程如图 1 所示，解密运算的流程如图 2 所示。

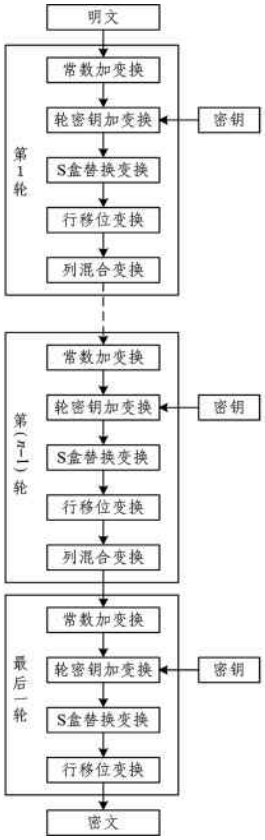

图 1　Surge 算法的加密过程

Fig. 1　Encryption process of Surge

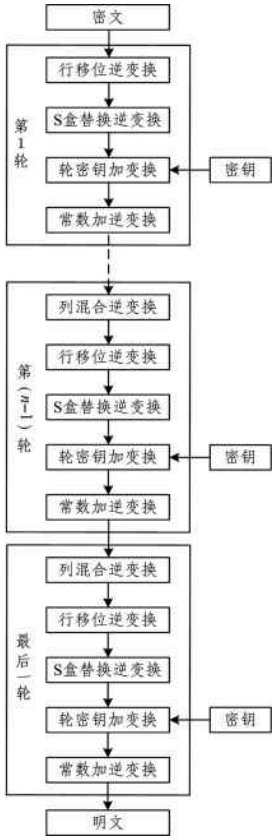

图 2　Surge 算法的解密过程

Fig. 2　Decrption process of Surge

3　Surge 加密的设计与实现

3.1　Surge 加密过程

Surge 加密的描述如算法 1 所示。

算法 1　Surge 加密

输入：Plaintext，Key

输出：Ciphertxet

1. State←Plaintext；
2. for i=1 to N_R-1 do
3. AddConstants(State)；
4. AddRoundKey(State，Key)；
5. SubCells(State)；
6. ShiftRows(State)；
7. MixColumns(State)；
8. end for
9. AddConstants(State)；
10. AddRoundKey(State，Key)；
11. SubCells(State)；
12. ShiftRows(State)；
13. Ciphertext←State；

3.2　Surge 加密的设计与实现

Surge 算法各个模块的运算单元为 4-bit，算法分组长度为 64-bit，分为 16 个单元，分别为 $state_0$，$state_1$，…，$state_{15}$。密钥为 64-bit 时，分为 16 个单元，分别为 key_0，key_1，…，key_{15}；密钥为 80-bit 时，分为 20 个单元，分别为 key_0，key_1，…，key_{19}；密钥为 128-bit 时，分为 32 个单元，分别为 key_0，key_1，…，key_{31}。

常数加(AddConstants)：常数加变换轮常数选取的原则为高位从 0，1，2，3 中选取，低位从 0 到 15 之间选取。轮常数组合的原则为：Surge-64 是当高位为 0 时，低位为 0 到 15 之间的奇数组合；高位为 1 时，低位为 0 到 15 之间的偶数组合；高位为 2 时，低位由 0 到 15 之间的偶数组合；高位为 3 时，低位为 0 到 15 之间的奇数组合。共有 32 个组合数，32 个组合常数随机固定一个排列，如表 1 所列，每一轮常数固定不变。Surge-80 常数组合原则为前 32 个组合数及排列顺序与 Surge-64 一致，后面依次还包括 0x36，0x30，0x34，0x32 4 个组合数，共 36 个组合数，如表 2 所列。Surge-128 常数组合的原则为前 36 个组合数及排列顺序与 Surge-80 一致，后面依次还包括 0x38，0x3c，0x3e，0x3a 4 个组合数，共 40 个组合数，如表 3 所列。其中，表 1－表 3 中的一个字节数据均为 16 进制。

表 1　Surge-64 算法常数加变换常数

Table 1　Transform constant of Surge-64 AddConstants

轮数	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
轮常数	22	35	07	20	0d	39	3d	1e	1a	2e	31	14	37	26	33	12
轮数	10	11	12	13	14	15	16	17	18	19	1a	1b	1c	1d	1e	1f
轮常数	2a	18	0f	24	05	1c	16	2c	3f	10	03	0b	09	01	28	3b

表 2　Surge-80 算法常数加变换常数

Table 2　Transform constant of Surge-80 AddConstants

轮数	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f	10	11	12	13	14	15	16	17	18	19	1a	1b	1c	1d	1e	1f	20	21	22	23
轮常数	22	35	07	20	0d	39	3d	1e	1a	2e	31	14	37	26	33	12	2a	18	0f	24	05	1c	16	2c	3f	10	03	0b	09	01	28	3b	36	30	34	32

表 3　Surge-128 算法常数加变换常数

Table 3　Transform constant of Surge-128 AddConstants

轮数	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f	10	11	12	13	14	15	16	17	18	19	1a	1b	1c	1d	1e	1f	20	21	22	23	24	25	26	27
轮常数	22	35	07	20	0d	39	3d	1e	1a	2e	31	14	37	26	33	12	2a	18	0f	24	05	1c	16	2c	3f	10	03	0b	09	01	28	3b	36	30	34	32	38	3c	3e	3a

常数加变换方法为 $state_0$，$state_8$ 与第 $i(0\leqslant i\leqslant N_R)$ 轮常数字节的高位进行异或，$state_4$，$state_{12}$ 与第 $i(0\leqslant i\leqslant N_R)$ 轮常数字节的低位进行异或。此方法每一轮运算中运算单元少且需要的寄存器资源少，但变换结果会扩散到整个数据中，是一个高效且高度混淆的常数加变换方法。运算关系如图 3 所示。

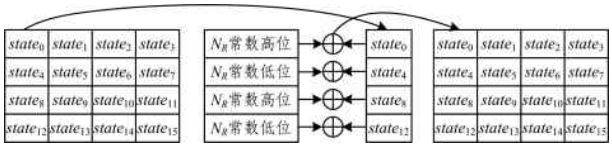

图 3 常数加变换运算

Fig. 3 AddConstants transform

轮密钥加(AddRoundKey)：将 64-bit 明文或每一轮中间值与第 $i(0\leqslant i\leqslant N_R)$ 轮轮密钥 64-bit 进行异或运算，64-bit 明文或每一轮中间值 $State(state_0\cdots state_{15})$、第 i 轮轮密钥 $k_0^i\cdots k_{15}^i$ 的运算关系如式(1)所示。

$$state_j\rightarrow state_j\oplus k_j^i(0\leqslant j\leqslant 15) \tag{1}$$

其中，轮密钥的产生规则为：Surge 算法分为 3 种密钥长度，即 64-bit，80-bit，128-bit。密钥长为 64-bit 时，每一轮的轮密钥就是 64-bit 原始密钥，轮密钥组合子项如式(2)所示。密钥长为 80-bit，当 i 为奇数次轮运算时，轮密钥为原始密钥前 64-bit；当 i 为偶数次轮运算时，轮密钥为原始密钥后 64-bit，轮密钥组合子项如式(3)与式(4)所示。密钥长为 128-bit，当 i 为奇数次轮运算时，轮密钥为原始密钥前 64-bit；当 i 为偶数次轮运算时，轮密钥为原始密钥后 64-bit，轮密钥组合子项如式(5)与式(6)所示。

64-bit 密钥组合子项密钥为：

$$Key_i=\begin{bmatrix} key_0 & key_1 & key_2 & key_3\\ key_4 & key_5 & key_6 & key_7\\ key_8 & key_9 & key_{10} & key_{11}\\ key_{12} & key_{13} & key_{14} & key_{15}\end{bmatrix} \tag{2}$$

80-bit 密钥 $Key_i=key_0\cdots key_{19}(1\leqslant i\leqslant N_R)$ 组合子项，当 i 为奇数轮运算密钥时：

$$Key_i=\begin{bmatrix} key_0 & key_1 & key_2 & key_3\\ key_4 & key_5 & key_6 & key_7\\ key_8 & key_9 & key_{10} & key_{11}\\ key_{12} & key_{13} & key_{14} & key_{15}\end{bmatrix} \tag{3}$$

当 i 为偶数轮运算密钥时：

$$Key_i=\begin{bmatrix} key_4 & key_5 & key_6 & key_7\\ key_8 & key_9 & key_{10} & key_{11}\\ key_{12} & key_{13} & key_{14} & key_{15}\\ key_{16} & key_{17} & key_{18} & key_{19}\end{bmatrix} \tag{4}$$

128-bit 密钥 $Key_i=key_0\cdots key_{31}(1\leqslant i\leqslant N_R)$ 组合子项，当 i 为奇数轮运算密钥时：

$$Key_i=\begin{bmatrix} key_0 & key_1 & key_2 & key_3\\ key_4 & key_5 & key_6 & key_7\\ key_8 & key_9 & key_{10} & key_{11}\\ key_{12} & key_{13} & key_{14} & key_{15}\end{bmatrix} \tag{5}$$

当 i 为偶数轮运算密钥时：

$$Key_i=\begin{bmatrix} key_{16} & key_{17} & key_{18} & key_{19}\\ key_{20} & key_{21} & key_{22} & key_{23}\\ key_{24} & key_{25} & key_{26} & key_{27}\\ key_{28} & key_{29} & key_{30} & key_{31}\end{bmatrix} \tag{6}$$

S 盒变换(SubCells)：S 盒变换是 Surge 算法的唯一非线性组件。

行移位变换(ShiftRows)：对于 16 个单元组成的 4×4 矩阵，矩阵每一行左循环不同的单元移量，第 0 行单元移量循环左移 3 个单行移位变换的运算关系如图 4 所示。

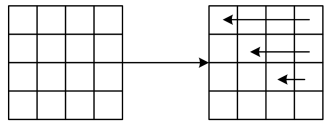

图 4 行移位变换运算关系图

Fig. 4 Transform relation of ShiftRows

列混合变换(MixColumns)：采用硬件实现友好型变换矩阵 M，矩阵 M 是由简单元素 0，1，2，4 组合成矩阵 A，根据有限域 $GF(2^4)$ 运算中的 4 次方构造出来的，构造公式如式(7)所示，其中的数据以 16 进制表示。

$$(A)^4=\begin{pmatrix}4&1&2&2\\1&0&0&0\\0&1&0&0\\0&0&1&0\end{pmatrix}^4=\begin{pmatrix}5&2&b&f\\e&8&c&4\\2&6&a&8\\4&1&2&2\end{pmatrix}=M \tag{7}$$

4 Surge 解密的设计与实现

4.1 Surge 解密过程

Surge 解密算法的描述如算法 2 所示。

算法 2 Surge 解密

输入：Ciphertxet，Key

输出：Plaintext

```
1. State←Ciphertxet;
2.    InvShiftRows(State);
3.    InvSubCells(State);
4.    AddRoundKey(State,Key);
5.    InvAddConstants(State);
6.    for i=2 to NR do
7.       InvMixColumns(State);
8.       InvShiftRows(State);
9.       InvSubCells(State);
10.      AddRoundKey(State,Key);
11.      InvAddConstants(State);
12.   end for
13. Plaintext←State;
```

4.2 Surge 解密算法的设计与实现

Surge 使用了加密运算变换中 4 种逆变换与轮密钥加变换，其中轮密钥加逆变换为其自身；以加密运算相反的顺序对密文进行解密，解密过程与加密过程使用的密钥相同。

常数加逆变换(InvAddConstants)：每一轮常数固定不变，Surge-64，Surge-80 与 Surge-128 解密运算是加密运算的反序。

S 盒逆变换(InvSubCells)：Surge 算法解密采用 PRESENT算法加密过程的 S 盒。

行移位逆变换(InvShiftRows)：对于 16 个单元组成的 4×4 矩阵，每一行右循环不同的单元移量，第 0 行循环右移 3 个单元，第 1 行循环右移 2 个单元，第 2 行循环右移 1 个单

元，第 3 行单元移量保持不变，则行移位逆变换的运算关系如图 5 所示。

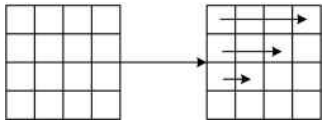

图 5　行移位逆变换运算关系图

Fig. 5　Transform relation of Invsubcells

列混合逆变换（InvMixColumns）：列混合矩阵为逆矩阵 M^{-1}，矩阵 M^{-1} 是由矩阵 A^{-1} 在有限域 GF(2^4) 运算中的 4 次方构造出的，构造公式如式(8)所示，其中的数据以 16 进制表示。

$$(A^{-1})^4=\begin{pmatrix}0&1&0&0\\0&0&1&0\\0&0&0&1\\9&2&9&1\end{pmatrix}^4=\begin{pmatrix}9&2&9&1\\9&b&b&8\\4&a&f&3\\8&2&2&c\end{pmatrix}=M^{-1}\quad(8)$$

列混合逆变换运算是 State 中元素 4×4 矩阵与列混合变换矩阵 M^{-1} 在有限域 GF(2^4) 上的乘法变换，变换公式如式(9)所示，其中的数据以 16 进制表示。

$$State=\begin{pmatrix}9&2&9&1\\9&b&b&8\\4&a&f&3\\8&2&2&c\end{pmatrix}\times\begin{pmatrix}state_0&state_1&state_2&state_3\\state_4&state_5&state_6&state_7\\state_8&state_9&state_{10}&state_{11}\\state_{12}&state_{13}&state_{14}&state_{15}\end{pmatrix}\quad(9)$$

Surge 解密时第一轮没有列混合逆变换。

5　Surge 算法的安全性分析

5.1　差分攻击与线性攻击

为了计算 Surge 算法抗差分与线性攻击，给出其差分特性与线性逼近的活动盒数量；对于 Surge 算法，算法的差分特性与线性逼近活动盒数量是由算法扩散层 P 的分支数决定的。下面分析算法扩散层 P 的分支数。

设 a_m，a_{m_1}，a_{m_2}，a_n 为 4 个正整数，且 $a_m\geqslant a_{m_1}\geqslant a_{m_2}$；$S_i$：GF(2)$^{a_m}\to$GF(2)$^{a_{m_1}}$，$T_i$：GF(2)$^{a_{m_1}}\to$GF(2)$^{a_{m_2}}$，其中 $i=1,\cdots,a_n$。

记差分为 $\Delta\chi_i\in$GF(2)^{a_m}，$\Delta\delta_i,\Delta\lambda_i\in$GF(2)$^{a_{m_1}}$，其中 $i=1,\cdots,a_n$；线性为 $\Gamma\chi_i\in$GF(2)$^{a_{m_1}}$，$\Gamma\delta_i,\Gamma\lambda_i\in$GF(2)$^{a_{m_2}}$，其中 $i=1,\cdots,a_n$。

以 $wt(\Delta\chi)$ 表示 $\Delta\chi$ 的包重量，即 $wt(\Delta\chi)=\#\{i\mid\Delta\chi_i\neq0\}$；$wt(\Gamma\chi)$ 表示 $\Gamma\chi$ 的包重量。P：$(GF(2)^{a_{m_1}})^{a_n}\to(GF(2)^{a_{m_2}})^{a_n}$ 为线性变换，且其扩散性达到最佳，及其分支数[9] $\min\limits_{\Gamma\lambda\neq0}\{wt(\Gamma\delta)+wt(\Gamma\lambda)\}=\min\limits_{\Gamma\delta\neq0}\{wt(\Delta\delta)+wt(\Delta\lambda)\}=a_n+1$。设 $a_n\times a_n$ 的矩阵 M 对应于扩散变换 P，则 P 的分支数[10] 为 a_n+1。

定理 1　任何四轮 Surge 算法的差分特征最少有 25 个活动盒。

证明：Surge 算法扩散层 P 的列混合矩阵为 4×4 矩阵变换，则分支数为 5。Surge 算法运算单元的活动半个字节称为一个活动盒（半个字节二进制数不全为 0 时）。Surge 算法每四轮差分特征如下（其中密钥长度为 Surge-64，Surge-80 和 Surge-128，轮数 N_R 分别为 32，36 和 40）。

Surge 算法某轮输入差分为 1 个活动盒，经算法 P 扩散层扩散，四轮运算活动盒变为 1→4→16→4，从而 Surge 算法四轮运算输出差分特征活动盒为 25。

Surge 算法某轮输入差分为 2 个活动盒，在算法 P 扩散层的扩散作用下，情况 1：四轮运算活动盒变为 2→3→16→4；情况 2：四轮运算活动盒变为 2→8→12→3；从而 Surge 算法四轮运算输出差分特征的活动盒为 25。

Surge 算法某轮输入差分为 3 个活动盒，在算法 P 扩散层的扩散作用下，情况 1：四轮运算活动盒变为 3→2→16→4；情况 2：四轮运算活动盒变为 3→7→12→3；情况 3：四轮运算活动盒变为 3→12→8→2。从而 Surge 算法四轮运算输出差分特征的活动盒为 25。

Surge 算法某轮输入差分为 4 个活动盒，经算法 P 扩散层扩散，情况 1：四轮运算活动盒变为 4→1→16→4；情况 2：四轮运算活动盒变为 4→6→12→3；情况 3：四轮运算活动盒变为 4→11→8→2；情况 4：四轮运算活动盒变为 4→16→4→1。从而 Surge 算法四轮运算输出差分特征的活动盒为 25。

不失一般性，如果某轮输入更多活动盒，经过四轮变换直接产生的最小数目活动盒将不低于 25。因此，Surge 算法的任何四轮变换差分特征至少有 25 个活动盒。

由于 Surge 算法 P 扩散层保证了差分分支数与线性分支数为 5，因此四轮变换的线性活动盒数最小为差分的活动盒数。证毕。

定理 2　任何四轮 Surge 算法的线性逼近最少有 25 个活动盒。

证明同上。

一个密码算法能否抗差分与线性攻击，主要表现在差分特征与线性逼近的活动盒数量上。上述分析给出的差分特征与线性逼近活动盒是最小值，根据每轮确定的差分特征与线性特征的活动盒数量来估算 Surge 算法的抗差分与线性攻击能力。Surge 算法 S 盒最大输出差分分配与线性近似为 2^{-2}；Surge 算法 S 盒最大输出差分为 $4\div16=2^{-2}$；最大线性近似达到 $(2\times4\div16-1)^2=2^{-2}$。

通过上述 Surge 算法抗差分与线性攻击的分析，Surge 算法在抗差分与线性攻击方面拥有绝对优势，从而证明了 Surge 算法能抗差分与线性攻击。

5.2　代数攻击

抗代数攻击能力是评论密码算法安全性的另一个重要指标。密码代数攻击归结于建立和求解有限域 GF(q) 上系数任意选取的非线性布尔方程组，而建立分组密码的 S 盒超定、稀疏代数方程组一直是代数攻击研究的难点。

构造代数方程组：

定义 1　设 f 是一个 n 元布尔函数，则称 $f(x)=\bigoplus\limits_{b\in\{0,1\}^n}c_bx^{(b)}$ 为 f 的代数正规型表示，并称 $\sum\limits_{b\in\{0,1\}^n}c_b$ 为 f 的项数。其中，$\forall x=(x_0,x_1,\cdots,x_{n-1})$，$b=(b_0,b_1,\cdots,b_{n-1})\in\{0,1\}^n$，有 $x^{(b)}=\prod\limits_{i=0}^{n-1}x_i^{b_i}$，且 $x_i^{b_i}$ 是 x_i 的 b_i 次方。

定义 2　设 S：$\{0,1\}^n\to\{0,1\}^m$，$f_1,f_2,\cdots,f_N$：$\{0,1\}^{n+m}\to\{0,1\}$。如果 $\forall z=(x,y)\in\{0,1\}^n\times\{0,1\}^m$，当 $S(x)=y$ 时，

对 $1\leqslant i\leqslant N$ 都有 $f_i(z)=0$，则称方程组 $f_i(z)=0, 1\leqslant i\leqslant N$ 为 S 的一个必要方程组。如果 $\forall z=(x,y)\in\{0,1\}^n\times\{0,1\}^m$，$S(x)=y$ 等价于对 $1\leqslant i\leqslant N$ 都有 $f_i(z)=0$，则称该方程组为 S 的等效方程组。

设 $S:\{0,1\}^n\rightarrow\{0,1\}^m$，则 $S_i(x)=y_i(1\leqslant i\leqslant m)$ 就是 S 的一个等效方程组，故等效方程组是存在的。显然，有 S 的必要方程组 $f_i(z)=0(1\leqslant i\leqslant N)$ 是 S 的一个等效方程组的充要条件是该方程组的解数为 2^n。

如上所述，构造 Surge 算法 S 盒的单项式个数均小于或等于 6 的无冗余等效方程组。

半字节的 S 盒可以由 GF(2)上 21 个二次方程描述。任何半字节四位 S 盒由至少 21 个这样的方程描述。整个 Surge 密码算法就可以通过 $E=n\times 21$ 个二次方程构造，二次方程由 $V=n\times 8$ 个变量描述，其中 n 是在 Surge 加密算法和密钥调度中使用的 S 盒数量。在 Surge 算法中：$n=32\times 16$，因此整个系统由 10752 个二次方程和 4096 个变量组成，这是一个典型的超定多变元高次方程组，即方程数多于变量数。在求解密钥时，增加明/密文的个数将会增大方程个数与密钥变量个数的差值，有利于缩短 CryptoMiniSat 求解代数方程的运行时间。但是，增加明/密文的个数相当于变相增加了中间变量的个数，每轮将会增加大约 200 个代数方程，从而增加了 SAT 语句数，使 CryptoMiniSat 的求解时间变长。求解一个这样的超定多变量二次方程组问题是 NP Hard 的，复杂度是关于 n 的指数。目前，在合理的时间和复杂度内代数攻击不能获取 Surge 的全部密钥，因此，Surge 密码算法可抗代数攻击。

6 Surge 密码算法的硬件实现及性能分析

本节对最小密钥长的典型轻量级分组密码算法进行了 FPGA 实现。表 4 列出了各轻量级密码算法的 FPGA 硬件实现面积与性能实验数据。通过表 4 中的数据对比表明，Surge-64算法是目前 SPN 结构轻量级密码算法中面积占用最小的，同时能保持相当高的加密周期、频率、吞吐率性能。

表 4 各轻量级密码算法的 FPGA 实验数据

Table 4 FPGA experimental data of lightweight cryptographic algorithms

算法	结构	分组长度/bits	密钥长度/bits	资源面积/Slices	时钟周期/ns	时钟频率/MHz	吞吐率/Mbps
Led	SPN	64	64	10222	9.960	100.402	194.719
PRESNET[19]	SPN	64	80	10265	9.750	102.564	198.912
MIBS	Feistel	64	64	10218	9.654	103.584	194.982
EPCBC	SPN	96	96	10569	9.971	100.291	283.175
LBlock	Feistel	64	80	10258	9.986	100.140	188.499
KLEIN	SPN	64	64	10277	9.777	102.281	467.570
Twine	Feistel	64	80	10353	9.690	103.199	173.809
Surge-64	SPN	64	64	9985	9.734	102.733	199.240
Surge-80	SPN	64	80	10074	9.710	102.987	178.140
Surge-128	SPN	64	128	10169	9.741	102.659	160.248

结束语 本文提出了一种低资源占用、高性能与高安全的 Surge 轻量级分组密码算法，该密码算法采用 SPN 结构，加、解密时使用相同的密钥。为了适应不同的应用环境，Surge 密码使用了 64-bit，80-bit 和 128-bit 3 种密钥长度，同时，Surge 也适合软、硬件实现。Surge 密码算法加、解密运算结构清晰，扩散混淆特性好，相对于目前已知的 SPN 结构轻量级密码算法，其面积占用最小，同时加密周期、工作频率、吞吐率保持了相当高的性能。

参 考 文 献

[1] HONG D, SUNG J, HONG S, et al. HIGHT: a new block cipher suitable for low-resource device[C]// Proceedings of the 2006 International Workshop on Cryptographic Hardware and Embedded Systems. Yokohama, Japan, 2006: 46-59.

[2] BOGDANOV A, KNUDSEN L R, LEANDER G, et al. PRESENT: an ultra-lightweight block cipher[C]// Proceedings of the 2007 International Workshop on Cryptographic Hardware and Embedded Systems. Vienna, Austria, 2007: 450-466.

[3] CHENG H, HEYS H, WANG C. PUFFIN: a novel compact block cipher targeted to embedded digital systems[C]// Proceedings of the 11th EUROMICRO Conference on Digital System Design Architectures, Methods and Tools. Parma, Italy, 2008: 383-390.

[4] IZADI M, SADEGHIYAN B, SADEGHIAN S S, et al. MIBS: a new lightweight block cipher[C]// Proceeding of The 8th International Conference on Cryptology and Network Security. Kanazawa, Ishikawa, Japan, 2009: 334-348.

[5] SHIBUTANI K, ISOBE T, HIWATARI H, et al. Piccolo: An ultra-lightweight block cipher[C]// Proceedings of the 2011 International Workshop on Cryptographic Hardware and Embedded Systems. Nara, Japan, 2011: 342-357.

[6] GUO J, PEYRIN T, POSCHMANN A, et al. The LED block cipher[C]// Proceedings of the 2011 International Workshop on Cryptographic Hardware and Embedded Systems. Nara, Japan, 2011: 326-341.

[7] WU W L, ZHANG L. LBlock: a lightweight block cipher[C]// Proceedings of the 9th International Conference on Applied Cryptography and Network Security. Nerja, Spain, 2011: 327-344.

[8] ZHENG G, NIKOVA S, LAW Y W. KLEIN: A New Family of Lightweight Block Ciphers[C]// Proceedings of the 7th Workshop on RFID Security and Privacy. Amherst, MA, USA, 2011: 1-18.

[9] 吴文玲，冯登国. 分组密码的设计与分析[M]. 北京：清华大学，2009.

[10] HONG S, LEE S, LIM J, et al. Provable security against differential and linear cryptanalysis for the substitution permutation network[J]. ETRI Journal, 2001, 23(4): 158-167.