软件学报 ISSN 1000-9825, CODEN RUXUEW
Journal of Software,2012,23(1):171−176 [doi: 10.3724/SP.J.1001.2012.03983]

E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel/Fax: +86-10-62562563

流密码算法 Grain 的立方攻击*

宋海欣[1,2+], 范修斌[1], 武传坤[1], 冯登国[1]

[1](中国科学院 软件研究所 信息安全国家重点实验室,北京 100190)

[2](中国科学院 研究生院,北京 100049)

Cube Attack on Grain

SONG Hai-Xin[1,2+], FAN Xiu-Bin[1], WU Chuan-Kun[1], FENG Deng-Guo[1]

[1](State Key Laboratory of Information Security, Institute of Software, The Chinese Academy of Sciences, Beijing 100190, China)

[2](Graduate University, The Chinese Academy of Sciences, Beijing 100049, China)

+ Corresponding author: E-mail: songhaixin@is.iscas.ac.cn; shx_201@sina.com

Song HX, Fan XB, Wu CK, Feng DG. Cube attack on Grain. *Journal of Software*, 2012,23(1):171−176. http://www.jos.org.cn/1000-9825/3983.htm

Abstract: At EUROCRYPT 2009, Dinur and Shamir proposed a new type of algebraic attacks named cube attack. Grain is one of the 3 final hardware-oriented stream ciphers in the eSTREAM portfolio, which takes an 80-bit secret key and a 64-bit initial vector as input and produces its keystream after 160 rounds of initialization. Applying cube attack on Grain with 70 initialization rounds, the study finds that 15-bit secret key can be recovered and can find 4 linear equations on another 23 bits of the secret key. Moreover, 1-bit secret key can be recovered by applying cube attack on Grain with 75 initialization rounds.

Key words: eSTREAM project; stream cipher; Grain; cube attack; key recovery

摘 要: Dinur 和 Shamir 在 2009 年欧洲密码年会上提出了立方攻击的密码分析方法.Grain 算法是欧洲序列密码工程 eSTREAM 最终入选的 3 个面向硬件实现的流密码算法之一,该算法密钥长度为 80 比特,初始向量(initial vector,简称 IV)长度为 64 比特,算法分为初始化过程和密钥流产生过程,初始化过程空跑 160 拍.利用立方攻击方法对 Grain 算法进行了分析,在选择 IV 攻击条件下,若算法初始化过程空跑 70 拍,则可恢复 15 比特密钥,并找到了关于另外 23 比特密钥的 4 个线性表达式;若算法初始化过程空跑 75 拍,则可恢复 1 比特密钥.

关键词: eSTREAM 工程;流密码算法;Grain;立方攻击;密钥恢复

中图法分类号: TP309 **文献标识码**: A

eSTREAM 工程[1]始于 2004 年 11 月,面向社会征集流密码算法,征集活动于 2005 年 4 月结束,一共征集到 34 种流密码算法.经过 3 个阶段的评选,选出 4 种适合软件实现的算法及 4 种适合硬件实现的算法.由于面向硬件实现的 F-FCSR-H 算法被成功破译[2],最终剩下 7 种算法.

Grain 算法[3]是由瑞典的 Hell,Johansson 和瑞士的 Meier 共同设计的一种面向硬件实现的流密码算法.针对

* 基金项目: 国家自然科学基金(60833008, 60902024)
收稿时间: 2010-10-26; 定稿时间: 2010-12-31
CNKI 网络优先出版: 2011-05-12 11:47, http://www.cnki.net/kcms/detail/11.2560.TP.20110512.1147.002.html

该算法过滤函数所存在的弱点,文献[4]提出了密钥恢复攻击,文献[5]提出了区分攻击,针对该算法初始化过程所存在的弱点,文献[6]提出了滑动再同步攻击.为了解决该算法存在的安全性问题,设计者对 Grain 算法进行修改后又提交了 Grain v1 算法[7],算法密钥长度为 80 比特.此外,设计者还提交了 128 比特密钥版本的 Grain 算法 Grain-128[8].Grain v1 算法是 eSTREAM 工程最终入选的 7 种流密码算法之一.若无特别声明,本文中“Grain”均指 Grain v1 算法.

目前,针对 Grain 算法最好的密码分析结果如下:2008 年,文献[9]基于滑动再同步攻击的思想,对 Grain 算法的初始化过程进行了分析,比密钥穷举攻击快了 1 倍,时间复杂度由 2^{80} 降为 2^{79};2009 年,文献[10]指出 Grain 算法在 2^{144} 个密钥-IV 空间中存在 2^{64} 个弱密钥,导致初始化过程完成后算法线性反馈移位寄存器 LFSR 的状态为全 0,在这种条件下,恢复密钥将比穷举攻击更加有效.

在 2009 年欧洲密码年会上,Dinur 和 Shamir 提出了一种新型的代数攻击方法,称为立方攻击(cube attack)[11].它是一种密钥恢复攻击[12],只要有一个输出比特可以表示为密钥和初始向量(或明文)的低次数多变量多项式,它就可以用来攻击任何密码系统.在文献[11]中,作者对 eSTREAM 工程最终入选的另外一种适合硬件实现的流密码算法 Trivium[13]进行了立方攻击,Trivium 算法初始化过程空跑 1 152 拍,若初始化过程空跑 735 拍后即输出乱数,则在选择 IV 攻击条件下,他们找到了 52 个关于密钥的线性表达式,使恢复全部 80 比特密钥的时间复杂度降为 2^{30}.

Grain 算法与 Trivium 算法相比,非线性次数较高.目前,公开文献上尚未有关于 Grain 算法立方攻击方面的分析结果.本文利用立方攻击密码分析方法,对减少初始化空跑拍数(由 160 拍减少至 70 拍和 75 拍)的 Grain 算法进行了分析.对初始化空跑拍数减少到 70 拍的 Grain 算法,我们可以恢复 15 比特密钥,并找到了关于另外 23 比特密钥的 4 个线性表达式;对初始化空跑拍数减少到 75 拍的 Grain 算法,我们可以恢复 1 比特密钥.

本文第 1 节介绍 Grain 算法.第 2 节对立方攻击进行简要介绍.第 3 节介绍我们对 Grain 算法进行立方攻击分析的结果.第 4 节对本文进行总结.

1 Grain 算法简介

1.1 算法描述

Grain 算法是由瑞典的 Hell,Johansson 和瑞士的 Meier 共同设计的一种流密码算法,算法分为密钥流产生过程和初始化过程,密钥长度为 80 比特,初始向量 IV(initial vector)长度为 64 比特,适用于对硬件资源(如门电路数、能量消耗、内存)限制很大的环境[14],每个时钟周期产生 1 比特乱数.Grain 算法由非线性反馈移位寄存器(NFSR)、线性反馈移位寄存器(LFSR)和输出函数 $h(x)$等部分组成,算法结构如图 1 所示.

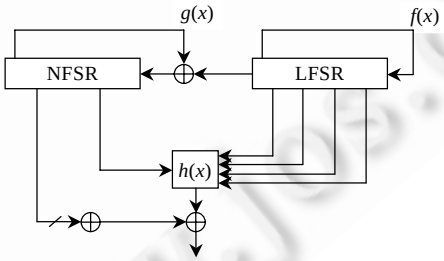


Fig.1 Schematic view of Grain

图 1 Grain 算法框图

1.1.1 密钥流(keystream)产生过程

1. 线性反馈移位寄存器(LFSR)

LFSR 为 80 级的线性反馈移位寄存器,反馈多项式为

$$f(x)=1+x^{18}+x^{29}+x^{42}+x^{57}+x^{67}+x^{80}.$$

LFSR 从右向左运动,每个时钟周期运动 1 拍,状态位从左至右按比特记为 $s_t,s_{t+1},s_{t+2},\ldots,s_{t+79}$,状态位的更新可表示如下:

$$s_{t+80}=s_{t+62}\oplus s_{t+51}\oplus s_{t+38}\oplus s_{t+23}\oplus s_{t+13}\oplus s_t.$$

2. 非线性反馈移位寄存器(NFSR)

NFSR 为 80 级的非线性反馈移位寄存器,反馈多项式为

$$\begin{aligned}g(x)=1+x^{18}+x^{20}+x^{28}+x^{35}+x^{43}+x^{47}+x^{52}+x^{59}+x^{66}+x^{71}+x^{80}+x^{17}x^{20}+x^{43}x^{47}+\\ x^{65}x^{71}+x^{20}x^{28}x^{35}+x^{47}x^{52}x^{59}+x^{17}x^{35}x^{52}x^{71}+x^{20}x^{28}x^{43}x^{47}+\\ x^{17}x^{20}x^{59}x^{65}+x^{17}x^{20}x^{28}x^{35}x^{43}+x^{47}x^{52}x^{59}x^{65}x^{71}+x^{28}x^{35}x^{43}x^{47}x^{52}x^{59}.\end{aligned}$$

NFSR 从右向左运动,每个时钟周期运动 1 拍,状态位从左至右按比特记为 $b_t,b_{t+1},b_{t+2},\ldots,b_{t+79}$,LFSR 的状态位 s_t 参与 NFSR 状态位的更新,NFSR 状态位的更新可表示如下:

$$\begin{aligned}b_{t+80}=s_t+b_{t+62}+b_{t+60}+b_{t+52}+b_{t+45}+b_{t+37}+b_{t+33}+b_{t+28}+b_{t+21}+b_{t+14}+b_{t+9}+b_t+b_{t+63}b_{t+60}+\\ b_{t+37}b_{t+33}+b_{t+15}b_{t+9}+b_{t+60}b_{t+52}b_{t+45}+b_{t+33}b_{t+28}b_{t+21}+b_{t+63}b_{t+45}b_{t+28}b_{t+9}+b_{t+60}b_{t+52}b_{t+37}b_{t+33}+\\ b_{t+63}b_{t+60}b_{t+21}b_{t+15}+b_{t+63}b_{t+60}b_{t+52}b_{t+45}b_{t+37}+b_{t+33}b_{t+28}b_{t+21}b_{t+15}b_{t+9}+b_{t+52}b_{t+45}b_{t+37}b_{t+33}b_{t+28}b_{t+21}\end{aligned}$$

3. 过滤函数

过滤函数为 5 入 1 出函数,表达式为

$$h(x)=x_1\oplus x_4\oplus x_0x_3\oplus x_2x_3\oplus x_3x_4\oplus x_0x_1x_2\oplus x_0x_2x_3\oplus x_0x_2x_4\oplus x_1x_2x_4\oplus x_2x_3x_4,$$

其中,$x_0=s_{t+3},x_1=s_{t+25},x_2=s_{t+46},x_3=s_{t+64},x_4=s_{t+63}$.将过滤函数的输出记为 h.

4. 密钥流产生

从 NFSR 取 7 个比特 $b_{t+1},b_{t+2},b_{t+4},b_{t+10},b_{t+31},b_{t+43},b_{t+56}$ 及过滤函数输出的 1 个比特 h 共计 8 个比特做模 2 加运算,得到 1 比特的密钥流,记为 ks,可表示如下:

$$ks=b_{t+1}\oplus b_{t+2}\oplus b_{t+4}\oplus b_{t+10}\oplus b_{t+31}\oplus b_{t+43}\oplus b_{t+56}\oplus h.$$

1.1.2 初始化过程

记 80 比特密钥为 $k_0,k_1,k_2,\ldots,k_{79}$,记 64 比特 IV 为 $v_0,v_1,v_2,\ldots,v_{63}$.首先,将密钥载入 NFSR,即 $b_{t+i}=k_i(0\leqslant i\leqslant 79)$,将初始向量 IV 作为前 64 比特状态载入 LFSR,LFSR 后 16 位用 1 填充,即 $s_{t+i}=v_i(0\leqslant i\leqslant 63),s_{t+i}=1(64\leqslant i\leqslant 79)$.然后,密钥流 ks 与移位寄存器 NFSR 及 LFSR 的反馈进行模 2 加运算,运行“密钥流产生过程”160 拍,完成初始化过程.初始化过程如图 2 所示.

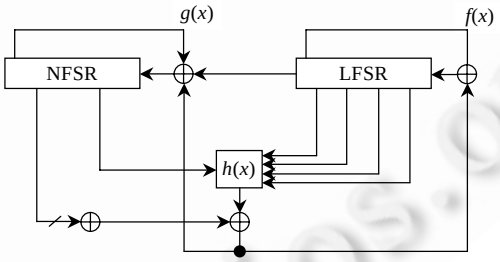


Fig.2 Initialization of Grain

图 2 Grain 算法初始化过程

2 立方攻击简介

立方攻击[11]是一种新型的代数攻击方法,旨在寻找密码算法的低次方程以恢复密钥或进行区分攻击.它吸收了饱和攻击[15]及高阶差分分析[16]的思想,该攻击方法主要基于下述定理:

定理 1[11]. 设 $f(x_1,x_2,\ldots,x_n)$为含有 n 个变量的布尔函数,I 为集合 $S=\{x_1,x_2,\ldots,x_n\}$的真子集:

$$f(x_1,x_2,\ldots,x_n)=p(I)\cdot(P(x_j|j\in S-I))\oplus R(x_j|j\in S),$$

其中: $p(I)=\left(\prod_{i\in I}x_i\right)$;$P(\cdot)$和 $R(\cdot)$均为代数标准型(*ANF*)表示的布尔函数,$P(\cdot)$函数中的变量均取自集合 I 的补集

S–I,$R(\cdot)$函数中的每一项均不含 I 中的全部变量.那么, $\oplus_{\{x_i|i\in I\}} f(x_1,x_2,...,x_n)=P(\cdot)$.

举例如下:设

$$f(x_1,x_2,x_3,x_4,x_5)=x_1x_2x_3\oplus x_1x_2x_4\oplus x_2x_4x_5\oplus x_1x_2\oplus x_2\oplus x_3x_5\oplus x_5\oplus 1,$$

它可以表示为

$$f(x_1,x_2,x_3,x_4,x_5)=x_1x_2(x_3\oplus x_4\oplus 1)\oplus(x_2x_4x_5\oplus x_3x_5\oplus x_2\oplus x_5\oplus 1).$$

这里,

$$I=\{x_1,x_2\},\ p(I)=x_1x_2,$$

$$P(\cdot)=x_3\oplus x_4\oplus 1,\ R(\cdot)=x_2x_4x_5\oplus x_3x_5\oplus x_2\oplus x_5\oplus 1,$$

$$\begin{aligned}\oplus_{\{x_i|i\in I\}} f(x_1,x_2,x_3,x_4,x_5)&=f(0,0,x_3,x_4,x_5)\oplus f(0,1,x_3,x_4,x_5)\oplus f(1,0,x_3,x_4,x_5)\oplus f(1,1,x_3,x_4,x_5)\\&=x_3\oplus x_4\oplus 1\\&=P(\cdot).\end{aligned}$$

在流密码算法中,初始向量 IV 为公开变量,密钥为未知变量.若集合 I 中的变量均为公开变量,$P(\cdot)$为非常量的线性表达式,我们就得到了关于未知变量(密钥)的一个线性方程 $P(\cdot)=\oplus_{\{x_i|i\in I\}} f(x_1,x_2,...,x_n)$, 并称 $p(I)$为极大项(maxterm),称 $P(\cdot)$为超级多项式(superpoly).

立方攻击为选择 IV 攻击,攻击者把密码算法看作一个黑盒子,它是关于 n 个未知变量和 m 个公开变量的未知多项式,输出 1 个比特.对密码算法的立方攻击分为两个阶段:预处理阶段和密钥恢复阶段.在预处理阶段,攻击者可以改变公开变量及未知变量的值,并可以模拟算法的执行,目的是找到尽量多的关于未知变量的线性方程(超级多项式),预处理过程只进行 1 次.在密钥恢复阶段,攻击者只改变公开变量的值,通过在预处理阶段找到的超级多项式来恢复某些密钥比特.对分组密码来讲,在选择明文攻击的条件下,明文为公开变量,密钥为未知变量;对序列密码来讲,在选择 IV 攻击的条件下,IV 为公开变量,密钥为未知变量.

3 对 Grain 算法的立方攻击

在文献[11]中,Dinur 和 Shamir 提出了立方攻击的分析方法,并对减少初始化空跑拍数的 Trivum 算法进行了密码分析.若算法非线性次数不高,立方攻击将十分有效.Grain 算法与 Trivium 算法相比,非线性次数较高,Grain 算法非线性反馈移存器的反馈多项式的非线性次数为 6,过滤函数的非线性次数为 3;而 Trivium 算法非线性反馈移存器的反馈多项式的非线性次数为 2,过滤函数是线性的.我们编程对初始化空跑拍数减少到 70 拍和 75 拍的 Grain 算法进行了立方攻击.在攻击的预处理阶段,主要问题是如何找到极大项 $p(I)$和超级多项式 P,并确定 P 是关于哪些密钥比特的线性表达式,下面分别进行介绍.

3.1 寻找极大项

我们通过线性测试的方法寻找极大项,随机选择 $x,y\in\{0,1\}^n$,验证 $P[0]\oplus P[x]\oplus P[y]=P[x\oplus y]$是否成立.具体步骤如下:

(1) 随机选择公开变量的一个子集记为 I,将其他公开变量设置为 0;

(2) 随机选择两个密钥变量值 $x,y\in\{0,1\}^n$,验证 $P[0]\oplus P[x]\oplus P[y]=P[x\oplus y]$是否成立.其中,函数 $P[x]$的计算方法为:密钥变量取值为 x,I 之外的公开变量取值为 0,跑遍 I 所有可能的取值,计算各函数值 $f(\cdot)$,将得到的结果模 2 加即为 $P[x]=\oplus_I f(\cdot)$;

(3) 若不成立,转步骤(1);若成立,转步骤(2)继续测试,直到连续通过 N 次测试,输出结果.

说明:关于测试次数 N 的选择,若 N 选择过大,则效率比较低;若 N 选择过小,则会有大量的非线性多项式连续通过 N 次测试.在实际编程过程中,我们采取了折衷的策略.若将测试次数 N 设为 100 仍有大量的非线性多项式被误以为是线性的,经过不断实验,我们最终将测试次数 N 设为 200.虽然仍有很少量的非线性多项式被误以为是线性的,但效率较高.下一步,我们对检测通过的每一个多项式再进行连续 10 000 次测试,若均通过,则认为该多项式是线性的,否则淘汰该多项式.

3.2 确定超级多项式

若 $p(I)$为极大项,确定 P 是关于哪些密钥比特的线性表达式的方法如下:

(1) 首先确定自由项,即表达式中有无⊕1:将 I 之外的所有变量(未知变量和公开变量)置为 0,跑遍 I,对结果求和,若为 0,则无⊕1;若为 1,则有⊕1;

(2) 确定 P 中变量 x_j 的系数是 0 还是 1:将 I 之外的所有变量(未知的和公开的)置为 0,跑遍 I,对结果求和得到 a;将 x_j 置为 1,将 I 之外的所有其他变量(未知的和公开的)置为 0,跑遍 I,对结果求和得到 b;若 $a=b$,则系数为 0;否则,系数为 1.

简言之,在一个线性表达式中,将所有变量置为 0 即得到自由项,若变量取值变反,运算结果也变反,则表达式中有该变量.

3.3 实验结果及分析

我们按照上述方法进行编程,在个人 PC 机上运行了两天,找到的极大项及超级多项式见表 1.程序又持续运行了几个星期,找到一些新的极大项,但没有找到新的超级多项式.

Table 1 Cube attack results on Grain

表 1 对 Grain 算法的立方攻击结果

极大项集合 I	超级多项式	空跑拍数
{28,2,46,48,42,14,57}	$1+k_{63}$	70
{31,17,42,32,2,48,47}	$1+k_{64}$	70
{2,44,20,5,48,49,37,55}	$1+k_{66}$	70
{24,19,41,18,39,2,50,48,16}	$1+k_{67}$	70
{28,48,42,10,51,14,61,2,39}	$1+k_{68}$	70
{25,17,12,54,13,14,52,48,2}	$1+k_{69}$	70
{5,48,42,28,2,44,36,53}	$1+k_{70}$	70
{24,25,10,6,9,54,48,2,28,17}	$1+k_{71}$	70
{55,48,37,35,7,31,2}	k_{72}	70
{2,12,9,15,56,48,31}	$1+k_{73}$	70
{32,2,12,1,35,48,31}	k_{74}	70
{56,37,26,48,29,31,58,2}	k_{75}	70
{17,48,59,63,2,6,7,38}	k_{76}	70
{48,35,61,32,37,2,26}	$1+k_{78}$	70
{48,13,60,11,1,2,7,28,32}	k_{79}	70
{63,13,59,48,10,2,0,36}	$k_2+k_3+k_5+k_{11}+k_{32}+k_{44}+k_{57}$	70
{9,10,34,29,35,19,7}	$k_3+k_4+k_6+k_{12}+k_{33}+k_{45}+k_{58}$	70
{48,37,31,2,18,44,20}	$k_4+k_5+k_7+k_{13}+k_{34}+k_{46}+k_{59}$	70
{12,40,9,41,31,57,60,37}	$k_5+k_6+k_8+k_{14}+k_{35}+k_{47}+k_{60}$	70
{62,15,25,40,12,46,10,34}	k_{63}	75

从表 1 可以看出:对初始化空跑拍数减少到 70 拍的 Grain 算法,在选择 IV 攻击的条件下,我们可以直接恢复 15 比特密钥,并找到了关于另外 23 比特密钥的 4 个线性表达式;对初始化空跑拍数减少到 75 拍的 Grain 算法,我们可以直接恢复 1 比特密钥.

表 1 中,极大项中公开变量的个数最少为 7,最大为 10;再随着初始化空跑拍数的增加,算法非线性次数增长很快,极大项中公开变量的个数也增加很快.对初始化空跑拍数减少到 80 拍的 Grain 算法,我们也编程进行了测试,极大项中公开变量的个数设置在 32 以下进行随机测试,在个人 PC 机上连续运行了几个月,仍未找到超级多项式.我们认为,初始化空跑拍数为 160 拍的 Grain 算法是可以抵抗立方攻击的.

4 结束语

本文应用立方攻击方法对减少初始化空跑拍数的 Grain 算法进行了密码分析,可以直接恢复部分密钥比特,从而极大地降低了密钥穷举量.如何改进和加速立方攻击算法是我们下一步研究工作的重点.

致谢 在此,我们向对本文的工作给予支持和建议的同行,尤其是冯登国教授领导的讨论班上的同学和老师表示衷心的感谢.

References:

[1] eSTREAM-ECRYPT stream cipher project. http://www.ecrypt.eu.org/stream/

[2] Hell M, Johansson T. Breaking the F-FCSR-H stream cipher in real time. In: Pieprzyk J, ed. Proc. of the ASIACRYPT 2008. LNCS 5350, Heidelberg: Springer-Verlag, 2008. 557–569. [doi: 10.1007/978-3-540-89255-7_34]

[3] Hell M, Johansson T, Meier W. Grain—A stream cipher for constrained environments. eSTREAM-ECRYPT Stream Cipher Project Report, 2005/010, 2005. http://www.ecrypt.eu.org/stream/ciphers/grain/grain.pdf [doi: 10.1504/IJWMC.2007.013798]

[4] Berbain C, Gilbert H, Maximov A. Cryptanalysis of Grain. In: Robshaw MJB, ed. Proc. of the FSE 2006. LNCS 4047, Heidelberg: Springer-Verlag, 2006. 15–29. [doi: 10.1007/11799313_2]

[5] Khazaei S, Hassanzadeh M, Kiaei M. Distinguishing attack on Grain. eSTREAM-ECRYPT Stream Cipher Project Report, 2005/071, 2005. http://www.ecrypt.eu.org/stream/papersdir/071.pdf

[6] Kücük Ö. Slide resynchronization attack on the initialization of Grain 1.0. eSTREAM-ECRYPT Stream Cipher Project Report, 2006/044, 2006. http://www.ecrypt.eu.org/stream/papersdir/2006/044.ps

[7] Hell M, Johansson T, Maximov A, Meier W. The Grain family of stream ciphers. In: Robshaw M, Billet O, eds. Proc. of the New Stream Cipher Designs. LNCS 4986, Heidelberg: Springer-Verlag, 2008. 179–190. [doi: 10.1007/978-3-540-68351-3_14]

[8] Hell M, Johansson T, Meier W. A stream cipher proposal: Grain-128. eSTREAM-ECRYPT Stream Cipher Project. 2006. http://www.ecrypt.eu.org/stream/grainp3.html [doi: 10.1109/ISIT.2006.261549]

[9] De Cannière C, Kücük Ö, Preneel B. Analysis of Grain's initialization algorithm. In: Vaudenay S, ed. Proc. of the AFRICACRYPT 2008. LNCS 5023, Heidelberg: Springer-Verlag, 2008. 276–289.

[10] Zhang HN, Wang XY. Cryptanalysis of stream cipher Grain family. Cryptology ePrint Archive Report, 2009/109, 2009. http://eprint.iacr.org/2009/109/

[11] Dinur I, Shamir A. Cube attacks on tweakable black box polynomials. In: Joux A, ed. Proc. of the EUROCRYPT 2009. LNCS 5479, Heidelberg: Springer-Verlag, 2009. 278–299. [doi: 10.1007/978-3-642-01001-9_16]

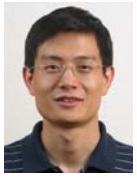
[12] Aumasson JP, Dinur I, Meier W, Shamir A. Cube testers and key recovery attacks on reduced-round MD6 and Trivium. In: Dunkelman O, ed. Proc. of the FSE 2009. LNCS 5665, Heidelberg: Springer-Verlag, 2009. 1–22. [doi: 10.1007/978-3-642-03317-9_1]

[13] De Cannière C, Preneel B. TRIVIUM—A stream cipher construction inspired by block cipher design principles. eSTREAM-ECRYPT Stream Cipher Project Report, 2005/030, 2005. http://www.ecrypt.eu.org/stream/ciphers/trivium/trivium.pdf

[14] Good T, Benaissa M. Hardware performance of eStream phase-III stream cipher candidates. In: Proc. of the SASC 2008. 2008. http://www.ecrypt.eu.org/stvl/sasc2008/

[15] Lucks S. The saturation attack—A bait for Twofish. In: Matsui M, ed. Proc. of the FSE 2001. LNCS 2355, Heidelberg: Springer-Verlag, 2002. 1–15. [doi: 10.1007/3-540-45473-X_1]

[16] Knudsen LR. Truncated and higher order differentials. In: Preneel B, ed. Proc. of the FSE'94. LNCS 1008, Heidelberg: Springer-Verlag, 1995. 196–211. [doi: 10.1007/3-540-60590-8_16]



宋海欣(1976－),男,山东泗水人,博士生,主要研究领域为流密码.

武传坤(1964－),男,博士,研究员,博士生导师,CCF 高级会员,主要研究领域为密码学,信息安全.

范修斌(1966－),男,博士,教授,博士生导师,主要研究领域为密码学,信息安全.

冯登国(1965－),男,博士,研究员,博士生导师,CCF 高级会员,主要研究领域为密码学,信息安全.