Trivium 流密码的基于自动推导的差分分析

丁 林,关 杰
(解放军信息工程大学,河南郑州 450000)

摘 要: Trivium 是欧洲 eSTREAM 工程评选出的 7 个最终胜出的流密码算法之一.本文提出了针对 Trivium 的基于自动推导的差分分析技术,利用该技术可以得到任意轮 Trivium 算法的差分传递链.将该技术应用于轮数为 288 的简化版 Trivium 算法,提出了一个有效的区分攻击,仅需 2^{26}个选择 IV,区分优势为 0.999665,攻击结果远优于已有的线性密码分析和多线性密码分析.将该技术应用于更多轮的 Trivium 算法和由 Turan 和 Kara 提出的修改 Trivium 算法,结果表明,初始化轮数低于 359 的 Trivium 算法不能抵抗差分分析,修改 Trivium 算法在抵抗差分分析方面优于原 Trivium 算法.

关键词: 密码分析; 差分分析; Trivium; 流密码
中图分类号: TN918.1 **文献标识码:** A **文章编号:** 0372-2112 (2014)08-1647-06
电子学报 URL: http://www.ejournal.org.cn **DOI:** 10.3969/j.issn.0372-2112.2014.08.030

Differential Cryptanalysis of Trivium Stream Cipher Based on Automatic Deduction

DING Lin, GUAN Jie
(*The PLA Information Engineering University, Zhengzhou, Henan* 450000, *China*)

Abstract: Trivium is a stream cipher and has successfully been chosen as one of seven finalists by European eSTREAM project. In this paper, a differential cryptanalysis of Trivium based on automatic deduction is presented. This new technique enables the attacker to obtain differential characteristics on arbitrary-round Trivium. The technique is applied to 288-round Trivium, which results in an efficient distinguishing attack. Our attack only requires 2^{26}chosen IVs with a distinguishing advantage of 0.999665. The result is much better than the existing single linear cryptanalysis and Linear Cryptanalysis with Multiple Approximations on 288-round Trivium. We also apply the technique to more-round Trivium and the modified Trivium proposed by Turan and Kara. The results show that Trivium reduced to no more than 359(out of 1152)initialization rounds is weak against differential cryptanalysis, and the modified Trivium is better against differential cryptanalysis than the original Trivium.

Key words: cryptanalysis; differential cryptanalysis; Trivium; stream cipher

1 引言

鉴于流密码的分析和设计在军事和外交保密通信中有重要价值,因而对其的研究一直是密码学者们关注的重点.为了发展安全快速的流密码,欧洲 ECRYPT(European Network of Excellence for Cryptology)在 2004 年启动了 eSTREAM 工程[1],开始征集可以广泛使用的流密码算法.截止到 2005 年 12 月共提交了 34 个流密码候选算法.2008 年 9 月有 7 个流密码算法最终胜出,其中有 3 个是面向硬件实现的,Trivium[2]流密码算法便是其中之一.

Trivium 流密码算法的密钥流生成器由 3 个非线性寄存器构成,其特点是将 3 个寄存器的非线性反馈交叉使用,从而达到 3 个寄存器相互控制的目的,该结构设计简单,易于硬件实现,现有的安全性分析表明,该算法的安全程度较高,是一个设计得很好的流密码算法.

Trivium 流密码算法一经发布,便吸引了很高的关注,对该算法的安全性分析结果也较多,主要有线性攻击[3~5]、代数攻击[6~10]、滑动攻击[11~13]和立方攻击[14~17]等;然而,仍没有一种攻击比穷举攻击有效,Trivium 至今仍然能够提供 80 比特安全性.作为欧洲

收稿日期:2012-11-28;修回日期:2013-06-14;责任编辑:蓝红杰
基金项目:国家自然科学基金(No.61202491,No.61272041,No.61272488);全军军事学研究生课题(No.2010JY0263-149);信息保障技术重点实验室开放基金(No.KJ-13-007);保密通信重点实验室基金(No.9140C110303140C11003)

eSTREAM 工程最终胜出的 7 个流密码算法之一,对 Trivium 流密码进行全面的安全性分析、考察其抵抗各种已知攻击方法的能力是十分必要的;然而,到目前为止,还没有对 Trivium 流密码进行差分分析的结果出现.针对这一现状,本文提出了针对 Trivium 的基于自动推导的差分分析技术,利用该技术对 Trivium 算法及由 Turan 和 Kara 提出的修改 Trivium 算法[3]进行了差分分析.

2 Trivium 流密码算法

Trivium 是以 80 比特密钥和 80 比特 IV 为输入的二元同步流密码算法,该算法产生的密钥流序列与明文序列异或产生密文序列.算法分为两部分:密钥流生成过程和初始化过程.Trivium 的内部状态由 3 个非线性寄存器构成,三个寄存器的长度分别为 93、84 和 111,其内部状态分别记为 $(s_1, s_2, \cdots, s_{93})$,$(s_{94}, s_{95}, \cdots, s_{177})$ 和 $(s_{178}, s_{179}, \cdots, s_{288})$.$\oplus$ 和 $\cdot$ 分别表示比特异或和比特与运算.密钥流生成算法描述如下:

For $i = 1$ to N do

$z_i \leftarrow s_{66} \oplus s_{93} \oplus s_{162} \oplus s_{177} \oplus s_{243} \oplus s_{288}$

$t_1 \leftarrow s_{66} \oplus s_{93} \oplus s_{91} \cdot s_{92} \oplus s_{171}$

$t_2 \leftarrow s_{162} \oplus s_{177} \oplus s_{175} \cdot s_{176} \oplus s_{264}$

$t_3 \leftarrow s_{243} \oplus s_{288} \oplus s_{286} \cdot s_{287} \oplus s_{69}$

$(s_1, s_2, \cdots, s_{93}) \leftarrow (t_3, s_1, \cdots, s_{92})$

$(s_{94}, s_{95}, \cdots, s_{177}) \leftarrow (t_1, s_{94}, \cdots, s_{176})$

$(s_{178}, s_{179}, \cdots, s_{288}) \leftarrow (t_2, s_{178}, \cdots, s_{287})$

End for

在初始化过程中,需要将 80 比特密钥和 80 比特 IV 加载到 Trivium 的 3 个寄存器中,随后动作 1152 次,在此期间不生成密钥流.完成初始化过程后,进入密钥流生成过程.初始化过程描述如下:

加载过程:

$(s_1, s_2, \cdots, s_{93}) \leftarrow (k_1, k_2, \cdots, k_{80}, 0, \cdots, 0)$

$(s_{94}, s_{95}, \cdots, s_{177}) \leftarrow (iv_1, iv_2, \cdots, iv_{80}, 0, 0, 0, 0)$

$(s_{178}, s_{179}, \cdots, s_{288}) \leftarrow (0, \cdots, 0, 1, 1, 1)$

空转过程:

For $i = 1$ to 1152 do

$t_1 \leftarrow s_{66} \oplus s_{93} \oplus s_{91} \cdot s_{92} \oplus s_{171}$

$t_2 \leftarrow s_{162} \oplus s_{177} \oplus s_{175} \cdot s_{176} \oplus s_{264}$

$t_3 \leftarrow s_{243} \oplus s_{288} \oplus s_{286} \cdot s_{287} \oplus s_{69}$

$(s_1, s_2, \cdots, s_{93}) \leftarrow (t_3, s_1, \cdots, s_{92})$

$(s_{94}, s_{95}, \cdots, s_{177}) \leftarrow (t_1, s_{94}, \cdots, s_{176})$

$(s_{178}, s_{179}, \cdots, s_{288}) \leftarrow (t_2, s_{178}, \cdots, s_{287})$

End for

3 Trivium 的基于自动推导的差分分析

由上述描述可以看出,Trivium 算法结构简单,这使得攻击者可以推导出 Trivium 算法的若干轮的差分传递过程.然而,由于该算法的初始化过程需要执行 1152 次状态刷新,手工推导差分传递链显然是比较困难的.本文提出了针对 Trivium 的基于自动推导的差分分析技术,其基本思想是考察 Trivium 算法的差分传递特征,依据这些特征设计差分传递的规则,按照设计出的规则使用差分值代替变元数值执行状态刷新过程,得到具体的差分传递链和相应的差分转移概率.上述过程可以通过计算机自动推导得到.

观察 Trivium 算法的迭代过程可知,虽然三个非线性移位寄存器的更新函数的输入不同,但是三个函数的结构是相同的,即都具有五个变元,最高次数为 2 且只有一个最高次项.因此,可以将它们统一描述为如下函数:

$$f(x_1, x_2, x_3, x_4, x_5) = x_1 \oplus x_2 \oplus x_3 \cdot x_4 \oplus x_5$$

假设已知各变元的输入差分别为 $\Delta_1, \Delta_2, \Delta_3, \Delta_4, \Delta_5$,$f$ 函数的输出差记为 Δ.则可得如下关系式

$$\begin{aligned}\Delta &= f(x_1, x_2, x_3, x_4, x_5) \oplus f(x_1 \oplus \Delta_1, x_2 \oplus \Delta_2, x_3 \oplus \Delta_3, x_4 \oplus \Delta_4, x_5 \oplus \Delta_5)\\ &= \Delta_1 \oplus \Delta_2 \oplus \Delta_5 \oplus x_3 \cdot \Delta_4 \oplus x_4 \cdot \Delta_3 \oplus \Delta_3 \cdot \Delta_4\end{aligned}$$

Trivium 算法的差分传递特征:

(1)若 $\Delta_3 = 0, \Delta_4 = 0, \Delta = \Delta_1 \oplus \Delta_2 \oplus \Delta_5$

(2)若 $\Delta_3 = 0, \Delta_4 = 1, \Delta = \Delta_1 \oplus \Delta_2 \oplus \Delta_5 \oplus x_3$

(3)若 $\Delta_3 = 1, \Delta_4 = 0, \Delta = \Delta_1 \oplus \Delta_2 \oplus \Delta_5 \oplus x_4$

(4)若 $\Delta_3 = 1, \Delta_4 = 1, \Delta = \Delta_1 \oplus \Delta_2 \oplus \Delta_5 \oplus x_3 \oplus x_4 \oplus 1$

从上述特征可以看出,只有在第 1 种特征中,输出差仅由输入差决定,与变元无关.在自动推导差分传递链的过程中,我们需要用差分值代替变元数值,即需要将变元消掉以使得差分传递链能够传递下去.因此,我们设计出如下差分传递的规则:

(1)若 $\Delta_3 = \Delta_4 = 0$,则 $\Delta = \Delta_1 \oplus \Delta_2 \oplus \Delta_5$

(2)若 $\Delta_3 | \Delta_4 = 1$,则规定 $\Delta = 0$

其中"|"表示逻辑或运算.从上述两条规则可以看出,规则 A 成立的概率为 1;假设 x_3 和 x_4 独立且都服从均匀分布(在 Trivium 中,因 x_3 和 x_4 是不同的变元,这假设是成立的),规则 B 成立的概率为 0.5.依据这两条规则,攻击者便可以推导出任意轮 Trivium 算法的差分传递链.在推导过程中,依据这两条规则可对差分传递链的差分转移概率进行计算.

给定一个未知的 80 比特密钥,攻击者在 IV 的 80 比特中引入差分,不妨设其输入差分为 $(\Delta iv_1, \Delta iv_2, \cdots, \Delta iv_{80})$.在完成密钥 K 和 IV 的加载后,Trivium 的内部状态的差分可表示为:

$(\Delta s_1^1, \Delta s_2^1, \cdots, \Delta s_{93}^1) \leftarrow (0, 0, \cdots, 0, 0, \cdots, 0)$

$(\Delta s_{94}^1, \Delta s_{95}^1, \cdots, \Delta s_{177}^1) \leftarrow (\Delta iv_1, \Delta iv_2, \cdots, \Delta iv_{80}, 0, 0, 0, 0)$

$(\Delta s_{178}^1, \Delta s_{179}^1, \cdots, \Delta s_{288}^1) \leftarrow (0, \cdots, 0, 0, 0, 0)$

其中 Δs_j^i 表示变元 s_j 在第 i 个时刻的差分,$j=1,2,\cdots,288$.

按照以上我们设计的差分传递的规则,Trivium 算法的差分传递过程可以描述为如下的自动推导算法,其中 $R(1\leqslant R\leqslant 1152)$表示 Trivium 算法的初始化轮数.

自动推导算法

Set $counter\leftarrow 0$.

For $i=1$ to R do

If $\Delta s_{91}^i=\Delta s_{92}^i=0$, $\Delta t_1^i\leftarrow\Delta s_{66}^i\oplus\Delta s_{93}^i\oplus\Delta s_{171}^i$;

Else $\Delta t_1^i\leftarrow 0$ and $counter\leftarrow counter+1$.

If $\Delta s_{175}^i=\Delta s_{176}^i=0$, $\Delta t_2^i\leftarrow\Delta s_{162}^i\oplus\Delta s_{177}^i\oplus\Delta s_{264}^i$;

Else $\Delta t_2^i\leftarrow 0$ and $counter\leftarrow counter+1$.

If $\Delta s_{286}^i=\Delta s_{287}^i=0$, $\Delta t_3^i\leftarrow\Delta s_{243}^i\oplus\Delta s_{288}^i\oplus\Delta s_{69}^i$;

Else $\Delta t_3^i\leftarrow 0$ and $counter\leftarrow counter+1$.

$(\Delta s_1^{i+1},\Delta s_2^{i+1},\cdots,\Delta s_{93}^{i+1})\leftarrow(\Delta t_3^i,\Delta s_1^i,\cdots,\Delta s_{92}^i)$;

$(\Delta s_{94}^{i+1},\Delta s_{95}^{i+1},\cdots,\Delta s_{177}^{i+1})\leftarrow(\Delta t_1^i,\Delta s_{94}^i,\cdots,\Delta s_{176}^i)$;

$(\Delta s_{178}^{i+1},\Delta s_{179}^{i+1},\cdots,\Delta s_{288}^{i+1})\leftarrow(\Delta t_2^i,\Delta s_{178}^i,\cdots,\Delta s_{287}^i)$.

End for

Set $m\leftarrow counter$.

通过如上的自动推导算法可知,在给定输入差的情况下,攻击者可以得到任意轮 Trivium 算法的差分传递链.由于规则 A 成立的概率为 1 且规则 B 成立的概率为 0.5,因此,m 的数值就对应着差分传递链的差分转移概率,即差分转移概率为:

$$(0.5)^m=2^{-m}$$

4　Trivium 的简化版本的差分分析

针对初始化轮数为 288 的简化版 Trivium 流密码算法,文献[3~5]先后对其进行了线性密码分析和多线性密码分析.需要指出的是,因在以上三个攻击中,攻击者都需要将 Trivium 算法 80 比特密钥中的 10 比特固定为 0.因此,他们的区分攻击只有在攻击者能够选择密钥的基础上才能进行,这样的攻击条件是比较苛刻的,也是很难达到的.在本节中,我们将运用上节给出的自动推导算法对这一简化版本进行差分分析,得到具体的差分传递链,在此基础上对该简化版本进行区分攻击.

首先我们要选择输入差分,为保证攻击是在选择 IV 的条件下进行,我们只在 IV 比特上引入输入差,由于 IV 有 80 比特,因此非零输入差分就有 $2^{80}-1$ 种,显然这样的量是很难穷尽的.事实上,我们经过大量的实验发现,为了得到具有较高差分转移概率的差分传递链,应使得输入差分的重量尽可能地小,即差分值为 1 的尽可能地少.当输入差分的重量为 1,即差分只出现在某一个 IV 比特时得到的差分转移概率是较大的.因此,我们只需要穷尽这 80 种情况即可.经过计算机的模拟实验,发现当差分出现在倒数第二个 IV 比特即 iv_{79} 上时,得到的差分转移概率是最大的,为 2^{-22},即 $\max\{p\mid w(\Delta IV)=1\}=2^{-22}$,其中 p 表示差分转移概率,$w(\Delta IV)$表示 IV 的 80 个比特中差分值为 1 的个数.

具体的模拟结果描述如下:

输入差分为:

$(\Delta s_1^1,\Delta s_2^1,\cdots,\Delta s_{93}^1)\leftarrow(0,0,\cdots,0,0,\cdots,0)$

$(\Delta s_{94}^1,\Delta s_{95}^1,\cdots,\Delta s_{177}^1)\leftarrow(0,0,\cdots,0,1,0,0,0,0,0)$

$(\Delta s_{178}^1,\Delta s_{179}^1,\cdots,\Delta s_{288}^1)\leftarrow(0,\cdots,0,0,0,0)$

输出差分参见表 1.其中表中的第 i 行第 j 列的数值指内部状态差分 Δs_{24i+j}^{288}.差分转移概率为 2^{-22}.

表 1　288 轮 Trivium 的输出差分

j / i	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
0	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1	1	1
4	1	0	0	0	0	0	1	1	1	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0
5	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
6	0	0	0	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1	0	0
7	0	0	0	1	1	1	1	0	0	0	0	0	1	0	0	0	0	0	1	1	1	1	0	0
8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
9	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
10	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
11	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0

经过 288 轮的初始化过程后,简化版本的 Trivium 算法开始输出密钥流.记完成初始化过程的 288 比特内部状态为 $S^{288}=(s_1^{288},\cdots,s_{288}^{288})$.考察 Trivium 算法的输出函数可知,在完成初始化过程后的 66 个时刻内输出的密钥流可以由 S^{288}线性表出.由表 1 可知

$$\Delta s_{66}^{288}=\Delta s_{93}^{288}=\Delta s_{162}^{288}=\Delta s_{177}^{288}=\Delta s_{243}^{288}=\Delta s_{288}^{288}=0$$

因此可得

$$\Delta z_1=\Delta s_{66}^{288}\oplus\Delta s_{93}^{288}\oplus\Delta s_{162}^{288}\oplus\Delta s_{177}^{288}\oplus\Delta s_{243}^{288}\oplus\Delta s_{288}^{288}=0$$

类似地,可以得到如下关系式

$$\Delta Z=\Delta z_1\parallel\cdots\parallel\Delta z_{66}=(1C800824024807E2)_4(01)_2 \tag{1}$$

其中$()_4$记为 16 进制表示,$()_2$记为 2 进制表示.

将关系式(1)记为事件 L,若上述差分传递链成立,则输出的前 66 比特密钥流以概率 1 满足关系式(1),即

$$\Pr(L)=2^{-22}$$

根据这一结论可以对该简化版本进行区分攻击.在选择 IV 的攻击模型下,给定一个未知的密钥 K,任意选择一个 80 比特 IV,将 IV 中的 1 比特 iv_{79}取反并保持其余的 79 比特不变就得到了 IV^*.将(K,IV)和(K,IV^*)分别加载到简化版的 Trivium 算法中,经过 288 轮的初始化过程,生成 66 比特密钥流,分别记为 Z 和 Z^*,检测关系式(1)是否成立.若攻击者随机选择 M 个(K,IV)和相应的(K,IV^*)进行检测,由于 $\Pr(L)=2^{-22}$成立,则 M 次检测中事件 L 至少发生一次的概率为

$$P_1=1-(1-2^{-22})^M$$

如果两条序列相互独立时,事件 L 发生的概率为 2^{-66},M 次检测中事件 L 至少发生一次的概率为

$$P_2=1-(1-2^{-66})^M$$

已知 $\lim\limits_{n\to+\infty}(1-\frac{1}{n})^n=e^{-1}$,其中 e 为自然数.则可得

$$P_2\approx 1-e^{-\frac{M}{2^{66}}}$$

为了使得区分攻击具有较高的区分优势,M 的选择应使得 P_1 十分接近于 1 且远远大于 P_2.表 2 中给出了 M 的取值与相应得到的 P_1,P_2 的取值.

表 2 M 的取值与相应得到的 P_1,P_2 的取值

M	2^{22}	2^{23}	2^{24}	2^{25}
P_1	0.632121	0.864665	0.981684	0.999665
P_2	$1-e^{-2^{-44}}$	$1-e^{-2^{-43}}$	$1-e^{-2^{-42}}$	$1-e^{-2^{-41}}$

由表 2 可以看出,当利用 2^{26}个选择 IV 对简化版本的 Trivium 进行区分攻击时,P_1 十分接近于 1 且远远大于 P_2,将事件 L 发生作为判据,可以保证事件 L 以接近 1 的概率发生.当事件 L 发生时,攻击者判断该输出序列为简化版本的 Trivium 的密钥流;当事件 L 不发生时,攻击者判断该输出序列为随机序列.有可能发生的误判有两种,分别为:

误判 1:事件 L 发生,而输出序列为随机序列;

误判 2:事件 L 不发生,而输出序列为简化版本的 Trivium 的密钥流.

误判 1 发生意味着在输出序列为随机的情况下事件 L 发生,误判 2 发生意味着在输出序列为密钥流的情况下事件 L 不发生.故易知误判 1 发生的概率为 $\alpha=1-e^{-2^{-41}}$,而误判 2 发生的概率为 $\beta=1-0.999665=0.000335$.根据文献[18]的结论易知本文提出的区分攻击的区分优势为

$$Adv=1-(\alpha+\beta)=e^{-2^{-41}}-0.000335\approx 0.999665$$

因此,我们提出的针对 288 轮 Trivium 的区分攻击所需的数据量为 2^{26}个选择 IV,区分优势为 0.999665.目前,针对 288 轮 Trivium 的分析结果为线性密码分析[3]和多线性密码分析[4-5].表 3 给出了与已有攻击结果的比较.表 3 表明,与已有的攻击相比,本文的攻击能够在更实际可行的攻击条件下,达到更高的区分优势,并且大大降低了攻击所需的数据量.

表 3 与同类攻击结果的比较

攻击	攻击条件	数据量	区分优势
Turan 等[3],线性分析	选择密钥	2^{62}	0.977
贾艳艳等[4],多线性密码分析	选择密钥	2^{61}	0.977
孙文龙等[5],多线性密码分析	选择密钥	2^{44}	0.977
本文,差分分析	选择 IV	2^{26}	0.999665

为了验证以上攻击的正确性,我们在普通的 PC 机上进行了实验,首先随机选择 2^{26}个 IV,运用初始化轮数为 288 的 Trivium 算法分别产生 66 比特密钥流,检验其输出密钥流差分是否满足等式(1),最后输出满足该等式的个数,实验结果为 10.理论上,事件 L 发生次数的期望为 $2^{25}\times 2^{-22}=8$,与以上的实验结果相近,这表明我们攻击的正确性.

5 进一步分析

5.1 更多轮 Trivium 算法的差分分析

Trivium 算法的初始化算法共有 1152 轮,因而有必要对更多轮的 Trivium 算法进行差分分析,以考察该算法抵抗差分分析的能力.运用以上提出的自动推导算法,我们对更多轮 Trivium 算法进行了分析,分析结果如表 4.

由表 4 可以看出,随着轮数的增加,差分转移概率逐渐减小.当初始化轮数为 359 时,差分转移概率为 2^{-61},攻击者需要 2^{66}的选择 IV 便能以 0.77880089 的区分优势攻击成功.由于数据量和区分优势间存在折中关系,当初始化轮数超过 359 时,攻击者需要考虑这一

关系进行选择.因此,本文的攻击方法对初始化轮数不大于 359 的简化版 Trivium 算法是有效的.表 4 同时表明,虽然 Trivium 算法结构简单且只采用 2 次的非线性函数,由于非线性反馈交叉机制的使用和初始化轮数较高,仍然具有很好的抵抗差分分析的能力.由于 Trivium 初始化轮数较高,增加了寻找差分传递链的难度,使得至今仍然没有对 Trivium 流密码算法进行差分分析的研究成果出现.

表 4　更多轮 Trivium 算法的差分分析

初始化轮数	输入差分位置	差分转移概率	数据量	区分优势
310	iv_1	2^{-36}	2^{41}	0.99999989
340	iv_{79}或 iv_{80}	2^{-49}	2^{54}	0.99999989
359	iv_{79}	2^{-61}	2^{66}	0.77880089
360	iv_{79}	2^{-62}	2^{67}	0.36787889
500	iv_{79}或 iv_{80}	2^{-178}	…	…
800	iv_1	2^{-492}	…	…
1152	iv_1	2^{-897}	…	…

5.2　修改 Trivium 算法的差分分析

为了得到更好的混乱特征,Turan 和 Kara 对 Trivium 的初始化算法进行修改,得到了修改 Trivium 算法[3].修改 Trivium 算法的初始化过程描述为以下形式:

$$(s_1, s_2, \cdots, s_{93}) \leftarrow (iv_1, \cdots, iv_{13}, iv_{14} \oplus k_1, \cdots, iv_{80} \oplus k_{67}, k_{68}, \cdots, k_{80})$$

$$(s_{94}, s_{95}, \cdots, s_{177}) \leftarrow (iv_1 \oplus k_1, \cdots, iv_{80} \oplus k_{80}, 0, 0, 0, 0)$$

$$(s_{178}, s_{179}, \cdots, s_{288}) \leftarrow (k_1, \cdots, k_{13}, k_{14} \oplus iv_1, \cdots, k_{80} \oplus iv_{67}, iv_{68}, \cdots, iv_{80}, 0, \cdots, 0, 1, 1, 1)$$

密钥流产生方式与原 Trivium 算法相同,初始化轮数也是 1152.

由于修改 Trivium 算法的密钥流产生方式与原 Trivium 算法相同,因而自动推导算法可以直接应用到该算法上,无需进行修改.表 5 给出了分析结果,并与原 Trivium 算法进行了对比.

表 5　Trivium 及其修改算法的差分分析

初始化轮数	Trivium 算法		修改 Trivium 算法	
	输入差分位置	差分转移概率	输入差分位置	差分转移概率
288	iv_{79}	2^{-22}	iv_{79}	2^{-65}
359	iv_{79}	2^{-61}	iv_{79}	2^{-134}
500	iv_{79}或 iv_{80}	2^{-178}	iv_{79}	2^{-272}
800	iv_1	2^{-492}	iv_{79}	2^{-574}
1152	iv_1	2^{-897}	iv_{79}	2^{-922}

从表 5 可以看出,相同的初始化轮数,修改 Trivium 算法的差分转移概率要低于原 Trivium 算法的差分转移概率.这一结果表明,由于修改 Trivium 算法在加载 IV 和 K 的过程中将 IV 重复使用了 3 次并将 IV 和密钥 K 的信息混合在一起,使得其扩散速率更快,能够更好地抵抗差分分析.

6　结束语

作为 eSTREAM 工程评选出的 7 个最终胜出的流密码算法之一,对 Trivium 进行全面的安全性分析是十分必要的.到目前为止,还没有对 Trivium 流密码进行差分分析的结果出现.针对这一现状,本文提出了针对 Trivium 的基于自动推导的差分分析技术,利用该技术可以得到任意轮 Trivium 算法的差分传递链.将该技术应用于轮数为 288 的简化版 Trivium 算法,提出了一个有效的区分攻击,仅需 2^{26} 个选择 IV,区分优势为 0.999665,攻击结果远优于已有的线性密码分析和多线性密码分析.将该技术应用于更多轮的 Trivium 算法和由 Turan 和 Kara 提出的修改 Trivium 算法.值得说明的是,本文提出的基于自动推导的差分分析思想具有一定的普适性,可以考虑将其应用于其它的流密码算法,这有待于进一步研究,并期望本文的分析结果对相关领域的研究者有所启发.

参考文献

[1] ECRYPT. eSTREAM: ECRYPT Stream Cipher Project, IST-2002-507932[EB/OL]. http://www.ecrypt.eu.org/stream, 2005-04-12.

[2] Cannière C D, Preneel B. Trivium[A]. New Stream Cipher Designs[C]. Germany: Springer-Verlag, 2008.244 – 246.

[3] Turan M S, Kara O. Linear Approximations for 2-round Trivium[EB/OL]. http://www.ecrypt.eu.org/stream/papersdir/2007/008.pdf, 2007-12-28.

[4] 贾艳艳,胡予濮,杨文峰,高军涛.2 轮 Trivium 的多线性密码分析[J].电子与信息学报,2011,33(1):223 – 227.
Jia Y Y, Hu Y P, Yang W F, Gao J T. Linear cryptanalysis of 2-round trivium with multiple approximations[J]. Journal of Electronics & Information Technology, 2011, 33(1): 223 – 227. (in Chinese)

[5] 孙文龙,关杰,刘建东.针对简化版 Trivium 算法的线性分析[J].计算机学报,2012,35(9):1890 – 1896.
Sun W L, Guan J, Liu J D. Linearcryptanalysis of simplified trivium[J]. Chinese Journal of Computers, 2012, 35(9): 1890 – 1896. (in Chinese)

[6] Maximov A, Biryukov A. Two trivial attacks on trivium[A]. Selected Areas in Cryptography[C]. Germany: Springer-Verlag, 2007.36 – 55.

[7] Wong K K, Gregory V B. Improved algebraic cryptanalysis of QUAD, Bivium and Trivium via graph partitioning on equation

systems[A]. The 15th Australasian Conference on Information Security and Privacy[C]. Germany: Springer-Verlag, 2010. 19 – 36.

[8] 李昕,林东岱.对 Bivium 流密码的变元猜测代数攻击[J].电子学报,2011,39(8):1727 – 1732.
Li X, Lin D D. Guessingspecific variables in algebraic attacks on Bivium[J]. Acta Electronica Sinica, 2011, 39(8): 1727 – 1732. (in Chinese)

[9] Huang Z Y, Lin D D. Attacking Bivium and Trivium with the characteristic set method [A]. Progress in Cryptology-AFRICACRYPT 2011[C]. Germany: Springer-Verlag, 2011. 77 – 91.

[10] Schilling T E, Raddum H. Analysis of Trivium using compressed right hand side equations[A]. The 14th International Conference on Information Security and Cryptology[C]. Germany: Springer-Verlag, 2011. 18 – 32.

[11] Priemuth-Schmid D, Biryukov A. slid pairs insalsa 20 and Trivium[A]. INDOCRYPT 2008[C]. Germany: Springer-Verlag, 2008. 1 – 14.

[12] 关杰,丁林.修改 Trivium 流密码算法的滑动攻击[J].上海交通大学学报(自然版),2012,46(6):926 – 930.
Guan J, Ding L. Slide attack on modified Trivium stream cipher[J]. Journal of Shanghai Jiaotong University (Science), 2012, 46(6): 926 – 930. (in Chinese)

[13] Zeng W, Qi W F. Finding slid pairs in Trivium with minisat [J]. Science China Information Sciences, 2012, 55(9): 1 – 8.

[14] Dinur I, Shamir A. cube attacks on tweakable black box polynomials[A]. EUROCRYPT 2009[C]. Germany: Springer-Verlag, 2009. 278 – 299.

[15] Mroczkowski P, Szmidt J. The cube attack on stream cipher Trivium and quadraticity tests[J]. Fundamenta Informaticae, 2012, 114(3 – 4): 309 – 318.

[16] Stankovski P. Greedy distinguishers and nonrandomness detectors[A]. INDOCRYPT 2010[C]. Germany: Springer-Verlag, 2010. 210 – 226.

[17] Pierre-Alain F, Thomas V. Improving key recovery to 784 and 799 rounds of Trivium using optimized cube attacks [EB/OL]. http://www.di.ens.fr/~fouque/pub/fse13a.pdf, 2013-6-14.

[18] Baignères T, Junod P, Vaudenay S. How far can we go beyond linear cryptanalysis? [A]. ASIACRYPT 2004[C]. Germany: Springer-Verlag, 2004. 432 – 450.

作者简介

丁 林 男,1987 年生于河南信阳.解放军信息工程大学博士生.研究方向为流密码分析.
E-mail: dinglin_cipher@163.com

关 杰 女,1974 年生于河南郑州.解放军信息工程大学教授、硕士生导师,研究方向为密码学与信息安全.
E-mail: guanjie007@163.com