非线性反馈移存器型序列密码的完全性通用算法

李俊志,关 杰
(信息工程大学,河南郑州 450000)

摘 要: 非线性反馈移存器型序列密码被使用于智能卡、射频识别标签(RFID)和无线传感器等硬件资源受限设备的信息加密中,其典型代表为Trivium算法、Grain v1算法和Mickey算法,然而现有的完全性算法在应用于此类序列密码时存在分析轮数较少及对依赖关系区分不清楚等问题.本文提出了一种考察此类序列密码完全性的通用算法,将算法内部状态表示成线性集合和非线性集合,将序列密码每轮更新转化为集合的运算,通过迭代计算可给出算法达到非线性完全性所需轮数的下界,克服了现有完全性算法的不足.应用此通用算法给出Trivium算法更优的1比特差分区分器并完成对Trivium-B算法的实时攻击.本方法可为此类序列密码的设计提供一定的理论依据.

关键词: 序列密码;非线性反馈移位寄存器;安全性指标;完全性;Trivium;区分攻击;分别征服攻击
中图分类号: TN918.1 **文献标识码:** A **文章编号:** 0372-2112 (2018)09-2075-06
电子学报URL: http://www.ejournal.org.cn **DOI:** 10.3969/j.issn.0372-2112.2018.09.005

Universal Algorithm of Full Diffusion of Stream Cipher Based on Nonlinear Feedback Shift Register

LI Jun-zhi, GUAN Jie
(*Information Engineering University, Zhengzhou, Henan* 450000, *China*)

Abstract: Stream ciphers based on nonlinear feedback shift register are used in information security of hardware restricted devices such as smart cards, radio frequency identification (RFID) and wireless sensor network. Typical examples of these ciphers are Trivium, Grain v1 and Mickey. Previous algorithms of full diffusion have disadvantages such as few analyzing rounds and indistinct dependent relationship. This paper proposes an algorithm for full diffusion of stream cipher based on nonlinear feedback shift register. The internal states of cipher are represented as linear sets and nonlinear sets. Then round functions of stream cipher are converted to operations of sets. So we can estimate the lower bound of rounds which a stream cipher needs to reach full nonlinear diffusion. Using this algorithm, this paper presents an improved one bit differential distinguisher of Trivium and a real-time attack on full rounds of Trivium-B. Moreover, this method can provide certain theory basis for the design of this kind of stream cipher.

Key words: stream cipher; nonlinear feedback shift register; security index of stream ciphers; full diffusion; Trivium; distinguishing attack; divide-and-conquer attack

1 引言

非线性反馈移位寄存器型序列密码是将非线性反馈移位寄存器作为初始乱源的序列密码.由于其硬件实现的资源开销少,此类序列密码在资源极端受限的环境中得到广泛使用.在欧洲eSTREAM工程[1]等推动下非线性反馈移存器受到了人们的持续关注,Trivium算法[2]、Grain v1算法[3]及Mickey算法[4]都是典型的非线性反馈移位寄存器型序列密码.

完全性是保证序列密码安全的重要指标之一,序列密码的完全性是指每个密钥流比特都包含所有密钥比特及IV比特的信息[5].目前研究完全性的方法有统

收稿日期:2017-08-29;修回日期:2017-12-07;责任编辑:覃怀银
基金项目:国家自然科学基金(No.61572516, No.61272488)

计测试法[6],表达式判断法[7]和信息包含法[8]等. 统计测试法的缺点在于只能以一定的概率否定完全性. 表达式判断法是将一个密码系统看成是输出比特关于明文(IV)及密钥的布尔函数[6],该方法的缺点在于分析的轮数较少. 信息包含法利用某一比特内部状态表达式中所包含的密钥和明文(IV)信息来判断算法是否达到完全,此方法的缺点在于考虑信息的包含时未区分输出比特与密钥和 IV 的线性依赖和非线性依赖关系.

为了能够改善现有完全性算法的不足,本文提出了一种判断非线性反馈移位寄存器型序列密码是否达到非线性完全性的通用算法. 该算法可以区分输出序列对于密钥和 IV 比特的线性依赖和非线性依赖,可以概率 1 否定完全性,能够从理论上给出算法非线性完全性轮数的下界.

为了验证所提完全性通用算法的有效性,将其应用于 Trivium 系列算法[2,9],首次给出了 Trivium 系列算法的完全性质,据此对 398 轮 Trivium 算法进行了差分-区分攻击,攻击所需数据量为 32,成功率为 $1-2^{-16}$,优于丁林等人在文献[10]中利用基于自动推导的差分分析技术构造的 359 轮 Trivium 算法的 1 比特差分区分器. 此外,本文对满轮的 Trivium-B 算法进行了密钥分割攻击,攻击的时间复杂度为 $O(2^{27})$,可以完成对该算法的实时攻击.

2 完全性通用算法

为了描述非线性完全性通用算法,首先给出非线性完全性的数学定义.

令 $P_{k_{ij}}=\Pr(\Delta z_i=1\mid\Delta k_j=1)$(或 $P_{v_{ij}}=\Pr(\Delta z_i=1\mid\Delta v_j=1)$). 则完全性的数学定义如下:若 $P_{k_{ij}}=0$(或 $P_{v_{ij}}=0$),则称 z_i 不依赖于 $k_j(v_j)$;若 $P_{k_{ij}}=1$(或 $P_{v_{ij}}=1$),则称 z_i 线性依赖于 $k_j(v_j)$;若 $0<P_{k_{ij}}<1$(或 $0<P_{v_{ij}}<1$),则称 z_i 非线性依赖于 $k_j(v_j)$. 当输出非线性依赖于所有密钥和 IV 时,称输出达到非线性完全性. 因此要保证序列密码安全,输出密钥流至少达到非线性完全性.

完全性通用算法的基本思想是从密钥和 IV 填充完毕开始,将内部状态分成线性部分和非线性部分两个集合,形式地按照算法更新函数和密钥流生成函数进行运算,得到输出密钥流中包含初始密钥和 IV 的信息.

2.1 符号说明

k_i:密钥的第 i 比特;

v_i:IV 的第 i 比特;

s_j^i:密码算法的第 i 轮的第 j 比特内部状态;

$\mathrm{LF}(X)$:多项式 X 的线性部分,$\mathrm{LF}(X)=\{x\mid$变量 x 在多项式 X 的线性项中出现$\}\cup\{$多项式 X 的常数项$\}$;

$\mathrm{NF}(X)$:多项式 X 的非线性部分,$\mathrm{NF}(X)=\{x\mid$变量 x 在多项式 X 的非线性项中出现$\}$;

$\hat{\boldsymbol{Y}}$:多项式 Y 的形式化描述,即 $\hat{\boldsymbol{Y}}=(\mathrm{LF}(Y),\mathrm{NF}(Y))$;

$\mathrm{PF}(A)$:将集合 A 组成仿射多项式,即 $\mathrm{PF}(A)=\sum_{x\in A}x$;

$+$:多项式的加法,指的是二元域上的加法;

$\cdot$:多项式的乘法,指的是二元域上的乘法;

$\boxplus$:形式加法;

$\boxdot$:形式乘法;

$\odot$:集合乘法,运算过程如下:

$$A\odot B=\begin{cases}\varnothing, & \text{if } B=\varnothing;\\ A\cup B, & \text{if } B\neq\varnothing;\end{cases}$$

$\hat{f}(\hat{\boldsymbol{x}}_1,\cdots,\hat{\boldsymbol{x}}_n)$:函数 f 的形式化描述,将其中的 $+$ 和 $\cdot$ 换成 $\boxplus$ 和 $\boxdot$,并将 x_i 换成 $\hat{\boldsymbol{x}}_i$;

$\mathrm{LP}(\hat{\boldsymbol{Y}})$:取形式化描述 $\hat{\boldsymbol{Y}}=(A,B)$ 的线性部分,即 $\mathrm{LP}((A,B))=A$;

$\mathrm{NP}(\hat{\boldsymbol{Y}})$:取形式化描述 $\hat{\boldsymbol{Y}}=(A,B)$ 的非线性部分,即 $\mathrm{NP}((A,B))=B$.

2.2 形式加法和形式乘法

下面定义通用算法中的形式加法与形式乘法:

多项式 f_1,f_2 的形式化描述为:

$$\hat{\boldsymbol{f}}_1=(\mathrm{LF}(f_1),\mathrm{NF}(f_1)),\hat{\boldsymbol{f}}_2=(\mathrm{LF}(f_2),\mathrm{NF}(f_2)).$$

形式加法:

$$\hat{\boldsymbol{f}}_1\boxplus\hat{\boldsymbol{f}}_2=(\mathrm{LF}[\mathrm{PF}(\mathrm{LF}(f_1))+\mathrm{PF}(\mathrm{LF}(f_2))],\ \mathrm{NF}(f_1)\cup\mathrm{NF}(f_2)) \tag{1}$$

形式乘法:

$$\hat{\boldsymbol{f}}_1\boxdot\hat{\boldsymbol{f}}_2=(\mathrm{LF}[\mathrm{PF}(\mathrm{LF}(f_1))\cdot\mathrm{PF}(\mathrm{LF}(f_2))],B) \tag{2}$$

其中 B 如下定义:

$$B=\mathrm{NF}[\mathrm{PF}(\mathrm{LF}(f_1))\cdot\mathrm{PF}(\mathrm{LF}(f_2))]\cup(\mathrm{LF}(f_1)\odot\mathrm{NF}(f_2))\cup(\mathrm{LF}(f_2)\odot\mathrm{NF}(f_1)). \tag{3}$$

经验证,形式加法和形式乘法满足交换律,结合律及乘法对加法的分配律,这些运算法则在证明定理 1 及完全性通用算法的具体实现过程中将会用到.

定理 1 给出了复合函数经过形式加法和形式乘法运算后的完全性描述,保证了形式运算后的结果与密码算法真实值相比不会损失信息.

定理 1 对多项式 $f(x_1,\cdots,x_n)$,各 x_i 是关于初始变量的表达式,则

$$\mathrm{LF}(f(x_1,\cdots,x_n))=\mathrm{LF}(\hat{f}(\hat{\boldsymbol{x}}_1,\cdots,\hat{\boldsymbol{x}}_n)),$$

且 $\mathrm{NF}(f(x_1,\cdots,x_n))\subseteq\mathrm{NF}(\hat{f}(\hat{\boldsymbol{x}}_1,\cdots,\hat{\boldsymbol{x}}_n))$.

证明 设 $f(x_1,\cdots,x_n)=L(x_{n_1},\cdots,x_{n_s})+N(x_{n_1},\cdots,$

$x_{n_i'}$),其中 $L(x_{n_1},\cdots,x_{n_s})$ 为 f 函数的线性表达式,$N(x_{n_1'},\cdots,x_{n_t'})$ 为 f 的非线性表达式,设 $x_1,\cdots,x_n$ 均是关于 $y_1,\cdots,y_m$ 的多项式,其线性部分别为 $\mathrm{LF}(x_i)$,其中,$1\le i\le n$,非线性部分别为 $\mathrm{NF}(x_i)$,其中,$1\le i\le n$,则 $\mathrm{LF}(f(x_1,\cdots,x_n))$ 只与 $\mathrm{LF}(x_i)$ 有关,故根据形式加法和形式乘法的定义有:

$$\mathrm{LF}(f(x_1,\cdots,x_n))=\mathrm{LF}(L[\mathrm{PF}(LF(x_{n_1})),\cdots,\mathrm{PF}(\mathrm{LF}(x_{n_s}))]\oplus N[\mathrm{PF}(\mathrm{LF}(x_{n_1'})),\cdots,\mathrm{PF}(\mathrm{LF}(x_{n_t'}))])$$

$$=\mathrm{LF}(\hat{L}(\hat{\boldsymbol{x}}_1,\cdots,\hat{\boldsymbol{x}}_n)\boxplus\hat{N}(\hat{\boldsymbol{x}}_1,\cdots,\hat{\boldsymbol{x}}_n))=\mathrm{LF}(\hat{f}(\hat{\boldsymbol{x}}_1,\cdots,\hat{\boldsymbol{x}}_n)).$$

考察 $\mathrm{NF}(f(x_1,\cdots,x_n))$,设 $y_j\in\mathrm{NF}(f(x_1,\cdots,x_n))$,

若 $y_j\notin\bigcup_{1\le i\le n}\mathrm{NF}(x_i)$,由于 $y_j\in\mathrm{NF}(f(x_1,\cdots,x_n))$,必然存在 $k_1,k_2\in\{n_1',\cdots,n_t'\}$,使 $y_j\in\mathrm{LF}(x_{k_1})$ 且 $y_j\in\mathrm{NF}(\mathrm{PF}(\mathrm{LF}(x_{k_1}))\cdot\mathrm{PF}(\mathrm{LF}(x_{k_2})))\subseteq\mathrm{NF}(\hat{\boldsymbol{x}}_{k_1}\boxdot\hat{\boldsymbol{x}}_{k_2})\subseteq\mathrm{NF}(\hat{f}(\hat{\boldsymbol{x}}_1,\cdots,\hat{\boldsymbol{x}}_n))$.

否则,存在 $k_3\in\{1,\cdots,n\}$,使 $y_j\in\mathrm{NF}(x_{k_3})$,由形式加法和形式乘法的定义,不管 $k_3\in\{n_1,\cdots,n_s\}$ 或 $k_3\in\{n_1',\cdots,n_t'\}$,都有 $y_j\in\mathrm{NF}(x_{k_3})\subseteq\mathrm{NF}(\hat{f}(\hat{\boldsymbol{x}}_1,\cdots,\hat{\boldsymbol{x}}_n))$.

因此 $\mathrm{NF}(f(x_1,\cdots,x_n))\subseteq\mathrm{NF}(\hat{f}(\hat{\boldsymbol{x}}_1,\cdots,\hat{\boldsymbol{x}}_n))$.

证毕.

利用形式加法和形式乘法计算出来的达到完全性所需圈数是在假设运算过程的变量抵消不影响完全性时得出的,故此计算结果是达到完全性所需圈数的下界,并且当复合函数运算过程的变量抵消不影响完全性时,它就是所需的真实圈数.

利用形式加法和形式乘法可以将原来基于内部状态比特的更新方程转化成现在基于内部状态线性和非线性部分的更新方程,并且保持所包含初始密钥及 IV 信息的完整性. 这样克服了表达式判断法因计算量大而只能研究算法较少轮数的缺点,可以研究任意轮数算法的完全性,还能够区分输出序列对于密钥和 IV 比特的线性依赖和非线性依赖,而且可以概率 1 否定完全性和非线性完全性.

2.3　通用算法描述

输出密钥流(内部状态)非线性完全性所需轮数的定义:

定义 1　若小于 t 时刻时,输出密钥流(内部状态)关于初始密钥及 IV 的表达式非线性部分没有包含所有初始密钥及 IV 的信息,而第 t 时刻输出密钥流(内部状态)关于初始密钥及 IV 的表达式非线性部分包含了所有密钥及 IV 的信息,则称 t 为密钥流(内部状态)达到非线性完全性所需轮数,简称非线性完全性轮数.

根据定义 1 给出基于非线性反馈移位寄存器序列密码非线性完全性轮数下界判断通用算法,即算法 1.

算法 1　基于非线性反馈移位寄存器序列密码非线性完全性轮数下界判断通用算法

输入:算法初始化轮数 M

输出:内部状态达到非线性完全性的轮数 r_s 及输出密钥流达到非线性完全性的轮数 r_z

令 $\mathrm{flag}_s=0$,$\mathrm{flag}_z=0$;

密钥及 IV 填充:

将密钥及 IV 的填充过程转化为每个内部状态比特的线性部分和非线性部分,对内部状态比特进行初始化填充.

初始化:

For $j=1$ to M

If 每个内部状态的非线性部分均包含所有密钥及 IV 且 $\mathrm{flag}_s=0$

$r_s=j-1$;

$\mathrm{flag}_s=1$;

计算更新比特的线性部分和非线性部分,并按照初始化算法对内部状态比特进行移位和更新.

计算输出比特的线性部分和非线性部分.

If 输出比特的非线性部分均包含所有密钥及 IV 且 $\mathrm{flag}_z=0$

$r_z=j$;

$\mathrm{flag}_z=1$;

If $\mathrm{flag}_s=0$

始化后内部状态不完全;

If $\mathrm{flag}_z=0$

初始化后密钥流不完全;

由定理 1 可知如下结论:

定理 2　如果密码算法的第 i 轮的第 j 个内部状态 s_j^i 包含初始密钥 k_t(或 v_t)的信息,则算法 1 得到的 s_j^i 也一定包含 k_t(或 v_t)的信息,其中 $1\le i\le M$,$1\le j\le N$,$1\le t\le n$.

证明:由定理 1 可得,算法 1 得到的内部状态 s_j^i 的线性部分组成的仿射函数表达式与密码算法实际内部状态 s_j^i 的线性部分的表达式相同,s_j^i 的非线性部分包含算法实际内部状态 s_j^i 表达式的非线性部分,定理得证.

证毕.

推论 1　算法 1 得到的密钥流(内部状态)非线性完全性轮数 r 是真实算法非线性完全性轮数 R 的下界.

证明:假设算法输出密钥流在第 R 轮达到非线性完全,即密钥流的非线性部分包含了所有的密钥及 IV 的信息,则由定理 2,此时由算法 1 得到的密钥流输出的非线性部分也一定包含了所有的密钥及 IV 的信息. 因此由算法 1 得到的算法非线性完全性轮数 $r\le R$.

证毕.

推论 1 的逆否命题为:如果在某一轮,算法 1 得到密钥流(内部状态)还未达到非线性完全,则密码算法也一定未达到非线性完全. 这个性质便于密码分析者构造区分攻击或密钥分割攻击.

将具体的密码算法代入到算法 1 中,可得到密码算法密钥流(内部状态)非线性完全性轮数的下界,它可以为算法初始化设计提供一定的理论依据.

3 Trivium 算法及其修改版本的完全性研究

3.1 Trivium 算法完全性分析

Trivium[2] 的密钥和 IV 长度均为 80 比特. 该算法初始化算法和密钥流生成算法两部分的内部状态更新方式相同,初始化算法共 1152 步,不输出密钥流,而后进入密钥流生成算法,每个时钟周期输出密钥流并更新寄存器. 算法的伪代码描述如下:

算法 2 Trivium 算法

$(s_1,s_2,\cdots,s_{93})\leftarrow(k_1,k_2,\cdots,k_{80},0,\cdots,0)$
$(s_{94},s_{95},\cdots,s_{177})\leftarrow(v_1,v_2,\cdots,v_{80},0,0,0,0)$
$(s_{178},s_{179},\cdots,s_{288})\leftarrow(0,\cdots,0,1,1,1)$
For $i=1$ to N do
  $z_i\leftarrow s_{66}+s_{93}+s_{162}+s_{177}+s_{243}+s_{288}$
  $t_1\leftarrow s_{66}+s_{93}+s_{91}\cdot s_{92}+s_{171}$
  $t_2\leftarrow s_{162}+s_{177}+s_{175}\cdot s_{176}+s_{264}$
  $t_3\leftarrow s_{243}+s_{288}+s_{286}\cdot s_{287}+s_{69}$
  $(s_1,s_2,\cdots,s_{93})\leftarrow(t_3,s_1,\cdots,s_{92})$
  $(s_{94},s_{95},\cdots,s_{177})\leftarrow(t_1,s_{94},\cdots,s_{176})$
  $(s_{178},s_{179},\cdots,s_{288})\leftarrow(t_2,s_{178},\cdots,s_{287})$

将 Trivium 算法结构及参数代入到第 2 节的通用算法中,可得 Trivium 算法的完全性通用算法.

为了验证算法的有效性,本文求得 Trivium 算法初始化第 240 轮的内部状态的表达式,并利用表达式判断法求出内部状态的完全性,与本文方法得到的第 240 轮的各状态完全性相对照,发现两者吻合,验证了完全性通用算法的有效性. 同时,此实验也说明了变量抵消一般不影响完全性通用算法的计算结论. 在考虑变量抵消时,算法 1 的计算结果往往与实际情况吻合或非常接近,所给出的完全性下界实用性强. 而当初始化轮数再增大若干轮时,由于计算能力的限制,将很难获得内部状态的表达式,利用表达式判断法来研究完全性也变得不可行,这反映了表达式判断法的局限性,而本文提出的完全性通用算法依然有效.

关于 Trivium 算法的完全性有以下结论:

性质 1 Trivium 算法密钥流非线性完全性的下界为 399.

398 轮的简化版 Trivium 算法输出密钥流的第 1 比特的非线性部分中没有包含 v_1 的信息,但是线性部分却出现了 v_1. 根据定理 2,真实的初始化为 398 轮的简化版 Trivium 算法输出的第 1 比特的非线性部分中也一定没有出现 v_1,而其线性部分中一定出现 v_1. 因此在密钥及 IV 的其它值不变的情况下改变 v_1 的值,算法输出密钥流的第 1 比特一定改变. 可以利用此性质构造差分区分器.

算法 3 对 398 轮的 Trivium 算法区分攻击

输入:选择 IV 对数 N
输出:判断结果
步骤 1:选取 N 个(IV,IV′)对,每个(IV,IV′)对满足如下条件
$$\Delta v_1=1,\text{其它 }\Delta v_k=0,\text{其中 }2\le k\le 80.$$
步骤 2:对于一个固定的密钥 K,将满足上述差分条件的(IV,IV′)对分别加载到初始化为 398 轮的 Trivium 算法中,检测算法输出密钥流第 1 比特的差分是否为 1.
步骤 3:若这 N 对 IV 输出密钥流第 1 比特的差分均为 1,判断此为 Trivium 算法输出,否则判断此为随机数.

此攻击方案使用 N 对选择 IV 下产生的首个密钥流比特,计算量为 $2N$ 次简化版 Trivium 算法,成功率为 $1-2^{-N}$. 当 N 取 16 时,经过 32 次简化版 Trivium 算法,区分攻击成功率为 $1-2^{-16}\approx 0.99998\approx 99.998\%$.

与现有的差分区分攻击进行对比. 文献[10]构造了 359 轮的 Trivium 算法的 1 比特差分区分器,区分攻击成功率 77.9%,数据复杂度为 $O(2^{66})$. 本节对 398 轮 Trivium 算法进行了差分-区分攻击,攻击所需数据量为 32,成功率为 $1-2^{-16}$,在攻击轮数、复杂度和成功率方面都有很大改进.

3.2 Trivium-B 算法完全性分析

文献[9]中作者给出了 Trivium 算法的 4 个修改版本,为了使内部状态的表达式的次数增长得更快,作者设计了 Trivium-B 算法,其密钥及 IV 填充过程及初始化轮数与 Trivium 相同,算法的伪代码描述如下:

算法 4 Trivium-B 算法

$(s_1,s_2,\cdots,s_{93})\leftarrow(k_1,k_2,\cdots,k_{80},0,\cdots,0)$
$(s_{94},s_{95},\cdots,s_{177})\leftarrow(v_1,v_2,\cdots,v_{80},0,0,0,0)$
$(s_{178},s_{179},\cdots,s_{288})\leftarrow(0,\cdots,0,1,1,1)$
For $i=1$ to N do
  $z_i\leftarrow s_{36}+s_{93}+s_{126}+s_{177}+s_{213}+s_{288}$
  $t_1\leftarrow s_{36}+s_{93}+s_{66}\cdot s_{162}+s_{147}$
  $t_2\leftarrow s_{126}+s_{177}+s_{162}\cdot s_{243}+s_{246}$
  $t_3\leftarrow s_{213}+s_{288}+s_{243}\cdot s_{66}+s_{63}$
  $(s_1,s_2,\cdots,s_{93})\leftarrow(t_3,s_1,\cdots,s_{92})$
  $(s_{94},s_{95},\cdots,s_{177})\leftarrow(t_1,s_{94},\cdots,s_{176})$
  $(s_{178},s_{179},\cdots,s_{288})\leftarrow(t_2,s_{178},\cdots,s_{287})$

将 Trivium-B 算法的结构代入算法 1 中,得到 Trivium-B 的非线性完全性算法. 由该算法发现了 Trivium-B 算法的一个严重安全缺陷. 无论 Trivium-B 算法初始化的轮数选取为多少,其输出密钥流均不完全,且有以下

结论成立.

性质 2　对于 Trivium-B 算法有:

(1)z_{3t-2}只与 k_{3i}和 v_{3i}有关,其中 $t\geq 1, 1\leq i\leq 26$;

(2)z_{3t-1}只与 k_{3i-1}和 v_{3i-1}有关,其中 $t\geq 1, 1\leq i\leq 27$;

(3)z_{3t}只与 k_{3i-2}和 v_{3i-2}有关,其中 $t\geq 1, 1\leq i\leq 27$.

据此可以对 Trivium-B 算法进行密钥分割攻击. 攻击的条件是已知 IV 及输出密钥流的前 108 比特

$$\boldsymbol{Z}=(z_1,z_2,\cdots,z_{108}) \tag{4}$$

攻击的准备工作是,先将密钥 K 分成三部分

$$\boldsymbol{K}^{(0)}=(k_1,k_4,k_7,\cdots,k_{79}) \tag{5}$$

$$\boldsymbol{K}^{(1)}=(k_2,k_5,k_8,\cdots,k_{77},k_{80}) \tag{6}$$

$$\boldsymbol{K}^{(2)}=(k_3,k_6,\cdots,k_{78}) \tag{7}$$

相应地,IV 也分成三部分

$$\boldsymbol{V}^{(0)}=(v_1,v_4,v_7,\cdots,v_{79}) \tag{8}$$

$$\boldsymbol{V}^{(1)}=(v_2,v_5,v_8,\cdots,v_{77},v_{80}) \tag{9}$$

$$\boldsymbol{V}^{(2)}=(v_3,v_6,\cdots,v_{78}) \tag{10}$$

同样,由此密钥 K 和 IV 产生的密钥流也可分为三部分

$$\boldsymbol{Z}^{(0)}=(z_3,z_6,\cdots,z_{105},z_{108}) \tag{11}$$

$$\boldsymbol{Z}^{(1)}=(z_2,z_4,z_8,\cdots z_{107}) \tag{12}$$

$$\boldsymbol{Z}^{(2)}=(z_1,z_4,z_7,\cdots,z_{106}) \tag{13}$$

攻击的具体过程如下.

算法 5　针对 **Trivium-B** 算法的密钥分割攻击

输入:IV 及密钥流的前 108 比特 $\boldsymbol{Z}$

输出:密钥 K

步骤 1:令 $\text{flag}_0=\text{flag}_1=\text{flag}_2=0$;$\boldsymbol{K}^0=\boldsymbol{K}^1=\boldsymbol{K}^2=\boldsymbol{0}$;

步骤 2:根据 $\boldsymbol{K}^{(m)}$的取值和已知的 $\boldsymbol{V}^{(m)}$计算 $\boldsymbol{Z}'(m)$;

步骤 3:For $m=0$ to 2

　If $\text{flag}_m=0$

　　If $\boldsymbol{Z}^{(m)}=\boldsymbol{Z}'^{(m)}$

　　令此时的 $\boldsymbol{K}^{(m)}$为候选密钥;

　　令 $\text{flag}_m=1$;

　Else

　　重新选取 $\boldsymbol{K}^{(m)}$;

　If $\text{flag}_0=\text{flag}_1=\text{flag}_2=1$

　　将得到的 K 作为正确密钥输出;

　Else

返回步骤 2 计算 $\boldsymbol{Z}'$;

此攻击需要的数据量为 108 比特密钥流,攻击的时间复杂度为 $O(2^{27})$,攻击的成功率为$\left(\frac{1}{1+2^{27}/2^{36}}\right)^2\otimes\frac{1}{1+2^{26}/2^{36}}>1-2\times 2^{-9}-2^{-10}>0.995=99.5\%$. 本文在个人电脑上进行了实验(计算环境为 Intel Core i5 - 3470 CPU 3.2GHz(4 CPUs),4GB 内存),随机选取初始密钥及 IV 进行了 1000 组实验,平均 45s 内可以求出所有初始密钥.

4　小结

完全性作为序列密码安全性的重要指标之一,对其进行深入研究是十分必要的,而现有的完全性算法存在着一些不足,针对这样的现状,本文提出了基于非线性反馈移位寄存器型序列密码完全性通用算法,可给出算法非线性完全性轮数的下界,为初始化算法的轮数选择及算法结构的设置提供有效指导. 本文将通用算法应用于 Trivium 系列算法中,首次给出了它们的完全性质并给出了新的分析结果. 此外若对非线性完全性通用算法中的非线性部分的信息扩散进行更加细致的研究,例如具体给出非线性表达式中的二次项,将有望进一步提高通用算法的有效性.

参考文献

[1] Robshaw M. The eSTREAM project[A]. Robshaw M. New Stream Cipher Designs:The eSTREAM Finalists[C]. Berlin,Heidelberg:Springer Berlin Heidelberg,2008. 1 - 6.

[2] De Cannière C. Trivium:a stream cipher construction inspired by block cipher design principles[A]. Katsikas S K. Information Security:9th International Conference[C]. Berlin,Heidelberg:Springer Berlin Heidelberg,2006. 171 - 186.

[3] Hell M,Johansson T,Meier W. Grain:a stream cipher for constrained environments[J]. International Journal of Wireless and Mobile Computing,2007,2(1):86 - 93.

[4] Babbage S,Dodd M. The MICKEY stream ciphers[A]. Robshaw M. New Stream Cipher Designs:The eSTREAM Finalists[C]. Berlin,Heidelberg:Springer Berlin Heidelberg,2008. 191 - 209.

[5] 金晨辉,郑浩然,张少武,等. 密码学[M]. 北京:高等教育出版社,2009. 166 - 167.

[6] Kölbl S,Leander G,Tiessen T. Observations on the SIMON block cipher family[A]. Gennaro R. CRYPTO 2015:35th Annual Cryptology Conference[C]. Berlin,Heidelberg:Springer Berlin Heidelberg,2015. 161 - 185.

[7] Shannon C E. Communication theory of secrecy systems[J]. Bell System Technical Journal,1949,28(4):656 - 715.

[8] Isobe T,Shibutani K. All subkeys recovery attack on block ciphers:extending meet-in-the-middle approach[A]. Knudsen L R. Selected Areas in Cryptography:19th International Conference[C]. Berlin,Heidelberg:Springer Berlin Heidelberg,2012. 202 - 221.

[9] Afzal M,Masood A. Modifications in the design of Trivium to increase its security level[J]. Proceedings of the Pakistan Academy of Sciences,2010,47(1):51 - 63.

[10] 丁林,关杰. Trivium 流密码的基于自动推导的差分分析[J]. 电子学报,2014,42(8):1647 – 1652.
DING Lin, GUAN Jie. Differential cryptanalysis of Trivium stream cipher based on automatic deduction[J]. Acta Electronica Sinica, 2014, 42(8): 1647 – 1652. (in Chinese)

作者简介

李俊志 男,1990 年生于河南新乡. 现为信息工程大学博士研究生. 主要研究方向为序列密码.
E-mail:lijunzhi1998@163.com

关 杰(通信作者) 女,1974 年生于河南郑州. 现为信息工程大学教授、博士生导师. 主要研究方向为密码算法设计与分析.
E-mail:guanjie007@163.com