论 文

改进的多步法快速相关攻击及其应用

张斌*, 冯登国

中国科学院软件研究所信息安全国家重点实验室, 北京 100190

* 通信作者. E-mail: zhangbin@is.iscas.ac.cn

收稿日期: 2010-03-12; 接受日期: 2011-01-05

国家自然科学基金 (批准号: 60833008, 60603018) 资助项目

摘要 本文提出了两个新的流密码多步法快速相关攻击算法. 第一个算法适用于噪音不是很大, 而攻击者只有很少预计算资源的环境. 第二个算法适用于高噪音而只能获得有限密钥流的场合. 新算法均适用于任何形式的 LFSR, 并在所考虑情况下优于以前的结果. 作为应用, 本文分别给出一级蓝牙流密码 E0 和 LILI-128 新的密钥恢复攻击. 给定 2^{37} 比特密钥流和 2^{28} 字节存储空间, 针对一级蓝牙流密码 E0 的新攻击可以在 $2^{35.1}$ 次操作中完成. 给定 2^{24} 比特密钥流和 $2^{24.5}$ 字节存储, 针对 LILI-128 的新攻击复杂度为 $2^{70.6}$ 次操作.

关键词 流密码 快速相关攻击 线性反馈移位寄存器 (LFSR) 蓝牙流密码 E0 LILI-128

1 引言

相关攻击是针对基于 LFSR 流密码的一种经典攻击^[1~14]. 给定一个合适的相关概率, 这种攻击试图以尽可能低的复杂度来恢复相应 LFSR 的初始状态. 相关攻击的思想最初由 Siegenthaler^[13] 在 1984 年提出, 随后 Meier-Staffelbach^[9] 提出了一种有趣的方法来有效地利用这种相关, 但前提是 LFSR 的反馈多项式具有非常低的重量. 关于相关攻击的一个最近进展由文献 [14] 提出, 其思想是把快速相关攻击和分别征服策略相结合, 产生一个多步法攻击. 虽然有时需要更多的密钥流比特, 与之前的最好结果相比^[3,12], 这种攻击在效率方面获得了极大提升. 然而在实际密码分析中, 可获得的密钥流长度往往十分有限, 这要么是由于密码设计者的规定, 要么是由于相关概率随密钥流长度增加而减小. 这两种情况都不能用当前的快速相关攻击来有效处理.

本文提出两个新的多步法快速相关攻击算法, 并将第二个算法分别应用于一级蓝牙流密码 E0 (本文只关注 E0 初始化后的密钥流产生算法. 因此, 只与针对一级蓝牙流密码 E0 的有关攻击做比较, 而不与针对二级蓝牙流密码 E0 的攻击比较^[15,16]) 和 LILI-128. 第一个算法适用于噪音不太高, 而攻击者只有很少预计算资源的环境. 第二个算法适用于高噪音和有限密钥流的情况. 这两个算法都适用于任何形式的 LFSR, 在所考察的情境下, 都优于相应的已有算法. 本文对每个算法都给出透彻的理论分析, 并辅以一定数量的计算机实验. 如果拥有一些特别的硬件, 第二个算法可以大规模并行实现以进一步减低复杂度.

引用格式: 张斌, 冯登国. 改进的多步法快速相关攻击及其应用. 中国科学: 信息科学, 2012, 42: 469–482

图 1 快速相关攻击模型

Figure 1 Model of fast correlation attacks

作为应用, 本文分别给出一级蓝牙流密码 E0^[17] 和 LILI-128^[18] 新的密钥恢复攻击. 如果有 2^{37} 比特的密钥流和 2^{28} 字节的存储, 针对一级蓝牙流密码 E0 的新攻击只需要 $2^{35.1}$ 次操作和一个复杂度为 2^{49} 次操作的预计算. 与之前的攻击相比 [8,19~26], 这是目前针对一级蓝牙流密码 E0 的最好攻击. 给定 2^{24} 比特密钥流和 $2^{24.5}$ 字节存储, 针对 LILI-128 的新攻击需要 $2^{70.6}$ 次操作和一个复杂度为 $2^{36.9}$ 次操作的预计算. 这个结果与之前的最好攻击相比 [21,25,27~31] 完全具有可比性.

本文按如下结构组织. 首先给出有关模型和定义, 然后回顾与分析一下最初的多步法快速相关攻击. 接着, 本文提出两个新的多步法算法, 并给出模拟实验结果及与其他重要快速相关攻击算法的比较. 最后, 将第二个算法分别应用于一级蓝牙流密码 E0 和 LILI-128, 并给出相关总结.

2 模型、定义和回顾

图 1 给出了快速相关攻击的解码模型, 这里密钥流 z_i 看作 LFSR 序列 $\{a_i\}$ 的一个噪音版本, 即 $\{a_i\}$ 通过一个交叉概率为 $1-p=0.5-\varepsilon$ ($\varepsilon>0$) 的二元对称信道后的输出. 攻击目的是恢复 LFSR 的初始状态 $a_L=(a_0, \dots, a_{L-1})$. 本文使用下列符号.

- ◆ $\oplus$ 是比特级的异或;
- ◆ $\lceil x \rceil$ 是大于或等于 x 的最小整数;
- ◆ $\lfloor x \rfloor$ 是小于或等于 x 的最大整数;
- ◆ m 是攻击的总步数;
- ◆ $k^{(j)} (1 \leq j \leq m)$ 是初始状态 a_L 的第 j 段长度;
- ◆ $N_j (1 \leq j \leq m)$ 是第 j 步攻击使用的密钥流, 这里 $N_j \leq N_1$, 而 N_1 是总的可获得的密钥流长度.
- ◆ $t_j (1 \leq j \leq m)$ 是第 j 步攻击使用的校验等式重量, 这里 $t_j \leq t_1 (2 \leq j \leq m)$;
- ◆ $\Omega_j (1 \leq j \leq m)$ 是第 j 步攻击使用的校验等式数量;
- ◆ $s_j (1 \leq j \leq m)$ 是第 2 个算法第 j 步攻击中用到的子组数量;
- ◆ $\delta = \sum_{i=1}^{j-1} k^{(i)}$;
- ◆ $1-h(p, t_j) = 1 + q_j \cdot \log_2 q_j + (1-q_j) \cdot \log_2 (1-q_j)$, 其中 $q_j = 0.5 + 2^{t_j-1} \varepsilon^{t_j}$.

本文首先概述一下作者最初提出的多步法快速相关攻击, 若想细致了解, 请看文献 [14]. 在一个多步法快速相关攻击中, 长为 L 的目标 LFSR 之初始状态 a_L 被分成了几份 (每份 $k^{(i)}$ 比特长, 如 (1) 式所示). 攻击每一步的目标就是通过构造适当的统计区分器, 全部或部分地恢复当前分段.

$$\left(\underbrace{a_0, \dots, a_{k^{(1)}-1}}_{k^{(1)}}, \underbrace{a_{k^{(1)}}, \dots, a_{k^{(1)}+k^{(2)}-1}}_{k^{(2)}}, \underbrace{a_{k^{(1)}+k^{(2)}}, \dots, a_{L-1}}_{\dots} \right). \quad (1)$$

在预计算阶段, 通过平方根时间/存储折衷方法或一般性的匹配——排序算法^[3]来预计算几类校验等式, 方法的选择依赖于攻击所使用的校验等式的重量. 由于后续阶段解码所需要的校验等式数量要远远少于前面阶段所需要的数量, 整个的预计算渐近复杂度并不增加. 这意味着攻击第一阶段的解码最重要.

在实时处理阶段, 整个初始状态按一步一步的方式来恢复. 在每一阶段, 攻击者进行一个假设检验来排除当前初态分段的错误猜测, 并记录下通过检测的那些猜测值, 作为当前分段的候选状态. 解码的主要技巧是定义一个布尔函数, 使用快速 Walsh 谱变换来同时估值对应于各个猜测的校验等式^[14]. 若当前步不是最后一步, 则将通过检测的候选状态传递到下一步. 当所有步都处理完后, 遗留的未知比特可以通过一个小规模的穷搜索和相关检测步骤来恢复.

这样的攻击在效率方面提高很大. 同之前的最好结果相比^[3,12], 预计算与实时处理阶段的时间复杂度都大大减少. 这种攻击的主要瓶颈是在某些情况下, 文献[3]中的攻击需要更长一些的密钥流. 对一个不能得到超过一定数量密钥流比特的攻击者而言, 一个需要稍高些复杂度的预处理和实时处理阶段的攻击是可以接受的. 最初版本的多步法攻击在这些情况下并不能有效解码. 下面, 本文将发展一些新的多步法算法来适应不同的攻击条件, 同时仍将时间复杂度保持在最佳水平.

3 两个新的多步法攻击算法

本文首先提出一个轻量级的按比特解码的多步法算法. 有趣的是, 这样一个简单的算法在同样噪音下, 其性能仍然超过了以前基于 Gallager 迭代解码^[1], 基于卷积码解码^[6]和 turbo 码解码^[5]的各个算法. 这个算法的主要思想是通过相应的校验等式来顺序检查 L 个初始状态比特, 并记录下通过检查的比特来作为候选. 此时, 通过检查的比特很可能不具有连续的位置, 而且可能包含少量错误预测. 通过一个小规模的穷搜索可以来补偿这些错误预测.

3.1 预计算阶段

由于第一个分段的解码最重要, 首先来看第一步. 构造如下形式的校验等式:

$$a_{i_1} \oplus a_{i_2} \oplus \cdots \oplus a_{i_{t_1}} = a_i, \quad (2)$$

这里 $a_i (0 \leq i \leq L-1)$ 是某个初始状态的未知比特. 对于 N 比特密钥流, 如果 $t_1 \leq 3$, 则使用平方根时间/存储折衷方法来构造这些等式. 时间和空间复杂度分别是 $N^{\lceil \frac{t_1-1}{2} \rceil} \log_2 N$ 次操作与

$$N^{\lfloor \frac{t_1-1}{2} \rfloor} t_1 \lceil \log_2 N \rceil$$

比特. 如果 $t_1 \geq 4$, 则使用一般的匹配——排序算法^[6]来得到这些等式. 当 $t_1 = 4$ 时, 时间与存储复杂度分别是 $N^2 \log_2 N$ 次操作与 $4N \lceil \log_2 N \rceil$ 比特. 注意无需再重复计算 a_{i+t} 的校验等式, 只需平移(2)式来得到 $a_{i_1+t} \oplus a_{i_2+t} \oplus \cdots \oplus a_{i_{t_1}+t} = a_{i+t}$. 还需注意每个校验等式都会通过下标平移而导出更多的校验等式. 因此, 解码中间部分分段的比特会导出更有效的攻击. 这里并未使用这条性质, 因为这样做会在解码时需要更多的密钥流比特.

3.2 实时处理阶段

一旦获得密钥流, 则有

$$z_{i_1} \oplus z_{i_2} \oplus \cdots \oplus z_{i_{t_1}} \oplus a'_i = (a_i \oplus a'_i) \oplus \bigoplus_{l=1}^{t_1} e_{i_l}, \quad (3)$$

这里 a'_i 是 a_i 的猜测值, 而 $z_{i_j} = a_{i_j} \oplus e_{i_j}$, 其中 $e_{i_j} (1 \leq j \leq t_1)$ 是噪音随机变量. 当 a'_i 猜测正确时, 有 $z_{i_1} \oplus z_{i_2} \oplus \cdots \oplus z_{i_{t_1}} \oplus a'_i = \bigoplus_{l=1}^{t_1} e_{i_l}$ 成立; 否则有 $z_{i_1} \oplus z_{i_2} \oplus \cdots \oplus z_{i_{t_1}} \oplus a'_i = 1 \oplus \bigoplus_{l=1}^{t_1} e_{i_l}$.

令 $\Delta(i_1, \dots, i_{t_1}) = z_{i_1} \oplus z_{i_2} \oplus \cdots \oplus z_{i_{t_1}} \oplus a'_i$, 若有 $\Omega_1 = \binom{n}{t_1}/2^L$ 这样的校验等式, 当 a'_i 猜测正确时, 和式 $\sum_{i=1}^{\Omega_1} (\Delta(i_1, \dots, i_{t_1}) \oplus 1)$ 应该符合二项分布 $(\Omega_1, q^{(1)} = 0.5 + 2^{t_1-1}\varepsilon^{t_1})$; 否则这个和式符合分布 $(\Omega_1, q'^{(1)} = 0.5 - 2^{t_1-1}\varepsilon^{t_1})$. 因此对 a_i 的一个猜测, 估值 Ω_1 个这样的校验等式, 并记录下 Δ 为 0 的次数将揭示出 a_i 的真值. 令 a_i 取导致超过一半 Δ 为 0 的猜测值, 上述步骤将用去 $\Omega_1 t_1$ 次操作和 $\Omega_1 t_1 \lceil \log_2 N \rceil$ 比特存储.

a_i 的正确猜测通过检测的概率是

$$P_1^{(1)} = \sum_{i=T_1}^{\Omega_1} \binom{\Omega_1}{i} (q^{(1)})^i (1 - q^{(1)})^{\Omega_1 - i},$$

而错误猜测通过检测的概率是

$$P_2^{(1)} = \sum_{i=T_1}^{\Omega_1} \binom{\Omega_1}{i} (q'^{(1)})^i (1 - q'^{(1)})^{\Omega_1 - i},$$

这里 T_1 是判决门限, 将在后面决定. 采用正态分布来逼近二项分布, 记 $\mu = \Omega_1 \cdot q^{(1)}$ 和 $\sigma = \sqrt{\Omega_1 q^{(1)}(1 - q^{(1)})}$ 分别为 $P_1^{(1)}$ 中的期望与方差; $\mu' = \Omega_1 \cdot q'^{(1)}$ 和 $\sigma' = \sqrt{\Omega_1 q'^{(1)}(1 - q'^{(1)})}$ 分别为 $P_2^{(1)}$ 中的期望和方差. 下面给出算法的完整描述.

参数: $m, t_1, \dots, t_m, k^{(1)}, \dots, k^{(m)}, p$

输入: 密钥流 $\{z_i\}_{i=0}^{N-1}$

for $i = 0, 1, \dots, L-1$ **do**

 令 a'_i 随机取 0 或 1, 且 $A = 0$

for a_i 的 Ω_1 个校验等式 **do**

 计算 $\Delta(i_1, \dots, i_{t_1}) = z_{i_1} \oplus z_{i_2} \oplus \cdots \oplus z_{i_{t_1}} \oplus a'_i$

if $\Delta(i_1, \dots, i_{t_1}) = 0$ **then** $A = A + 1$ **end if**

end for

if $A > T^{(1)}$ **then**

 令 $a_i = a'_i$, 且将其存储至数组 U

else if $A < \Omega_1 - T^{(1)}$ **then**

 令 $a_i = a'_i \oplus 1$, 并将其存储至数组 U

end if

end for

for $j = 2, \dots, m$ **do**

 采用最初的多步法快速相关攻击来恢复剩下的 $L - |U|$ 比特

end for

输出: 初始状态 a_L , 或一小列个候选状态

表 1 第一个算法的理论与模拟结果

Table 1 Experimental and theoretical results of the first algorithm^{a)}

Attack	Noise	Keystream	Time	Memory	Pre-computation
Ref. [1]	0.3	9770	2^{34}	$2^{14.2}$	2^{39}
Ours	0.3	4770	$2^{21.8}$	$2^{19.3}$	2^{28}
Ref. [1]	0.44	400000	2^{43}	$2^{19.1}$	2^{45}
Ours	0.44	200000	$2^{33.2}$	2^{32}	$2^{39.4}$

a) The LFSR polynomial is $1 + x + x^3 + x^5 + x^9 + x^{11} + x^{12} + x^{17} + x^{19} + x^{21} + x^{25} + x^{27} + x^{29} + x^{32} + x^{33} + x^{38} + x^{40}$.

对 L 个未知初态比特, 约有 $R = \lceil L \cdot P_1^{(1)} + L \cdot P_2^{(1)} \rceil$ 个比特将通过检测, 这些比特正确恢复的概率为 $\frac{P_1^{(1)}}{P_1^{(1)} + P_2^{(1)}}$. 在这个算法中, 选择门限 $T^{(1)}$, 以使只有少量比特恢复错误, 亦即令 $\frac{P_2^{(1)}}{P_1^{(1)} + P_2^{(1)}}$ 很小. 这样就部分地恢复了 R 个初态比特, 剩下的 $L - R$ 比特可以用最初的多步法快速相关攻击恢复^[14].

仍然采用上面的记号, 关于这个算法的时间/存储/数据复杂度有下列定理.

定理 1 如果取门限的合适值, 使上述攻击的成功概率接近于 1 (超过 99%), 且设 $V = \lceil R \cdot \frac{P_2^{(1)}}{P_1^{(1)} + P_2^{(1)}} \rceil$ 是所恢复的 R 比特初始状态中, 错误恢复的期望比特数, 则上述算法的时间复杂度为 $L\Omega_1 t_1 + \binom{R}{V} \cdot (\sum_{i=2}^m (\Omega_i t_i + 2^{k^{(i)}} k^{(i)}) + 2^{L-R-\sum_{i=2}^m k^{(i)}} \cdot (\frac{1}{\epsilon_2}))$ 次操作 (大部分是比特级的操作), 存储复杂度为 $\Omega_1 t_1 \lceil \log_2 N \rceil$ 比特, 而数据复杂度为 $\sqrt[4]{\frac{2^{L+1} \cdot L \cdot t_1!}{1-h(p, t_1)}}$ 比特.

证明 时间复杂度就是上述算法的一个直接估算, 具体如下. $\sum_{i=2}^m (\Omega_i t_i + 2^{k^{(i)}} k^{(i)})$ 是第 i ($i \geq 2$) 步的解码复杂度, 而 $2^{L-R-\sum_{i=2}^m k^{(i)}} \cdot (\frac{1}{\epsilon_2})$ 为相关检测步骤的复杂度. 存储复杂度证明来源于下述事实: 在一个多步法攻击中, 第 $i+1$ ($i \geq 1$) 步用到的校验等式个数要远远少于第 i 步所用到的校验等式个数, 因此, 整个攻击的存储消耗由第一步的存储复杂度决定. 对于数据复杂度, 根据 Shannon 的信道编码定理^[32], 有 $\frac{\binom{N}{t_1}}{2^L} = \frac{2 \cdot L}{1-h(p, t_1)}$ 式成立, 这里 $N = N_1$ 是攻击所需要的密钥流长度, t_1 是第一步所用校验等式的重量. 2 乘以 L , 将使攻击的成功概率非常接近于 1. 通过式 $\binom{N}{t_1} \approx N^{t_1}/t_1!$ 可以导出 N .

虽然上面的一些分析是直观性的, 但实验结果确实肯定如下事实, 即在精心选择的参数下, 这些分析是有效的. 本文进行了一定数量的计算机实验来验证这个算法的性能. 在精心选择的攻击参数下, 实验结果与理论分析符合得很好. 表 1 给出了上述算法的模拟结果及其与之前按位解码算法^[1]的比较. 由于使用重量超过 3 的校验等式的 Gallager 迭代解码^[1]在性能上优于基于卷积码^[6]和基于 turbo 码^[5]的解码算法, 本文只与文献^[1]中的结果做比较.

为便于比较, 与其他许多文献中一样, 本文选择一个标准的 LFSR 来进行实验与理论分析. 表 1 表明本文的攻击可以在只消耗大约一半密钥流的条件下显著地降低时间复杂度. 这里大部分操作均为异或及逻辑运算, 而存储复杂度以比特为单位. 注意到虽然本文攻击的存储复杂度有所增加, 但它们仍然在实际许可的范围内. 本文在一台 Pentium 4 个人电脑上用 C 语言实现了关于 40 比特 LFSR 在噪音为 $1-p=0.3$ 时的攻击. 预计算阶段大约需要数十秒, 而解码阶段只需数秒就输出结果. 关于 $1-p=0.44$ 的结果只是理论分析.

在噪音为 0.3 时, 共有 $\binom{4700}{5}/2^{40} \approx 17346$ (期望值) 个重量为 5 的校验等式. 此时, 每个校验等式的相关概率为 $0.5 + 2^4 \times (0.2)^5 = 0.50512$, 而 $P_1^{(1)} = 0.826055$, $P_2^{(1)} = 2^{-4.66897}$. 一共有 $40 \times (0.826055 + 2^{-4.66897}) \approx 35$ 个初态比特通过检测, 其中约有 $35 \cdot \frac{2^{-4.66897}}{0.826055 + 2^{-4.66897}} \approx 2$ 比特是错误预测. 本文用一个时间复杂度为 $2^{9.21675}$ 的穷搜来补偿这些错误比特. 在噪音为 0.44 时, 共有 $\binom{200000}{4}/2^{40} \approx 60631162$ (期望值) 个重量为 4 的校验等式. 此时, 每个校验等式的相关概率为 $0.5 +$

$2^3 \times (0.06)^4 = 0.500104$, 而 $P_1^{(1)} = 0.895286$, $P_2^{(1)} = 2^{-5.40365}$. 一共有 $40 \times (0.895286 + 2^{-5.40365}) \approx 37$ 个初态比特通过检测, 其中约有 $37 \cdot \frac{2^{-5.40365}}{0.895286 + 2^{-5.40365}} \approx 1$ 比特是错误预测. 本文用一个时间复杂度为 $2^{5.20945}$ 的穷搜来补偿这个错误比特.

下面, 本文引入一个新的基于块的解码算法, 这一算法同样用到了多步法的思想. 当噪音很高, 并且可获得的密钥流很短时, 这个新算法提供了时间\存储\预计算之间的新的可能折中. 新算法的基本思想是利用几个不同种类的校验等式, 并结合替换密钥流和有效估值校验等式的技巧. 本文之前, 这样一个组合还没在公开文献中得到过研究.

1) 预计算阶段.

在新算法的第 j 步, 构造如下的校验等式.

$$a_{i_1} \oplus a_{i_2} \oplus \cdots \oplus a_{i_{t_j}} \oplus \bigoplus_{i=0}^{\delta-1} y'_i a_i = \bigoplus_{i=\delta}^{\delta+k(j)-1} x_i^{(j)} a_i \oplus \bigoplus_{i=\delta+k(j)}^{L-1} h_i^{(j)} a_i, \quad (4)$$

在第 1 步, $\delta = 0$, 而 $\bigoplus_{i=0}^{\delta-1} y'_i a_i$ 并不存在. 在 (4) 式的右侧包含一项 $\bigoplus_{i=\delta+k(j)}^{L-1} h_i^{(j)} a_i$, 原因是攻击者无需知道 $(a_{\delta+k(j)}, \dots, a_{L-1})$ 的真实值. 当恢复 $(a_\delta, \dots, a_{\delta+k(j)-1})$ 时, 攻击者可以随意给 $(a_{\delta+k(j)}, \dots, a_{L-1})$ 赋值. 原因是攻击者只需要一个偏差, 而无需在乎这个偏差的符号. (4) 式大大拓展了预计算阶段可以利用的校验等式类别和数量. 对 N 比特的密钥流而言, 攻击者可以利用平方根的时间\存储折中来构造校验等式, 其时间与空间复杂度分别是 $N^{\lceil (t_j-1)/2 \rceil} \log_2 N$ 次操作和 $N^{\lfloor (t_j-1)/2 \rfloor} (t_j \lceil \log_2 N \rceil + L)$ 比特.

2) 实时处理阶段.

由上述校验等式, 可得如下的区分器:

$$z_{i_1} \oplus \cdots \oplus z_{i_{t_j}} \oplus \bigoplus_{i=0}^{\delta-1} y'_i a_i \oplus \bigoplus_{i=\delta}^{\delta+k(j)-1} x_i^{(j)} a'_i \oplus \bigoplus_{i=\delta+k(j)}^{L-1} h_i^{(j)} a''_i = \bigoplus_{i=\delta}^{\delta+k(j)-1} x_i^{(j)} (a_i \oplus a'_i) \oplus \bigoplus_{l=1}^{t_j} e_{i_l}, \quad (5)$$

其中 a'_i 是 a_i 的猜测值, $(a''_{\delta+k(j)}, \dots, a''_{L-1})$ 是对某个固定 $(h_{\delta+k(j)}^{(j)}, \dots, h_{L-1}^{(j)})$ 而言, $(a_{\delta+k(j)}, \dots, a_{L-1})$ 的赋值. 由于在当前步可以利用几类不同的, 分别对应 $(h_{\delta+k(j)}^{(j)}, \dots, h_{L-1}^{(j)})$ 值的校验等式, 攻击者需要将对应于每个 $(h_{\delta+k(j)}^{(j)}, \dots, h_{L-1}^{(j)})$ 固定值的所有偏差之平方和相加. 下面是这个算法的完整描述.

参数: $m, t_1, \dots, t_m, k^{(1)}, \dots, k^{(m)}, p$

输入: 密钥流 $\{z_i\}_{i=0}^{N-1}$

for $i = 1, \dots, m$ do

对 ω 的所有 $2^{k^{(i)}}$ 种取值, 令 $B_\omega = 0$

for 每个由 $(h_{\delta+k(j)}^{(i)}, \dots, h_{L-1}^{(i)})$ 确定的校验等式类 do

令 $(a''_{\delta+k(j)}, \dots, a''_{L-1})$ 任意取值

for 这一类中的所有校验等式 do

$$\begin{aligned} & \text{定义 } h^{(i)}(x_\delta^{(i)}, \dots, x_{\delta+k(i)-1}^{(i)}, h_{\delta+k(i)}^{(i)}, \dots, h_{L-1}^{(i)}) = \\ & \sum_{(x_\delta^{(i)}, \dots, x_{\delta+k(i)-1}^{(i)})} (-1)^{z_{j_1} \oplus \cdots \oplus z_{j_{t_j}} \oplus \bigoplus_{l=0}^{\delta-1} y'_l a_l \oplus \bigoplus_{l=\delta+k(i)}^{L-1} h_l^{(i)} a''_l} \end{aligned}$$

对 $2^{k^{(i)}}$ 种 ω 的可能取值, 使用 FWT 来计算 $H_{h_{L-k(j)-\delta}}(\omega)$,

这里 $H_{h_{L-k(j)-\delta}}(\omega)$ 是 $h^{(i)}(x_\delta^{(i)}, \dots, x_{\delta+k(i)-1}^{(i)}, h_{\delta+k(i)}^{(i)}, \dots, h_{L-1}^{(i)})$ 的 Walsh 谱变换.

end for

对 $2^{k^{(i)}}$ 种 ω 的可能取值, 令 $B_\omega = B_\omega + (H_{h_{L-k(j)-\delta}}(\omega))^2 / 4$.

end for

寻找满足 $B_\omega \geq T^{(i)}$ 的 B_ω , 并接受相应的 ω 进入下一步, 即第 $i+1$ 步.

end for

if $\sum_{i=1}^m k^{(i)} < L$ then

穷搜剩下的 $L - \sum_{i=1}^m k^{(i)}$ 比特, 并利用相关检测来恢复这些比特.

end if

输出: 初态 a_L , 或者一个候选者的小型列表.

这里 $T^{(i)}$ 是由整个攻击的成功概率决定的门限值. $B_\omega = B_\omega + (H_{h_{L-k(j)-\delta}}(\omega))^2/4$ 累积了每个猜测的偏差平方和. 上述描述只是用来说明整个算法的结构, 实际编程时需要做出相应的变动.

下面, 本文给出一个该算法的理论证明. 首先注意对于每个固定 $(h_{\delta+k(j)}^{(i)}, \dots, h_{L-1}^{(i)})$, 校验等式的期望值为 $\Omega(h_{L-k(j)-\delta}) = \binom{N_j}{t_j}/2^{L-\sum_{i=1}^j k^{(i)}}$. 这样由 (5) 式, 当 $(a_\delta, \dots, a_{\delta+k(j)-1})$ 猜测正确时, 在由标定的 $h_{L-k(j)-\delta}$ 校验式类中, 关于下面等式成立次数 $u(h_{L-k(j)-\delta})$, 应当存在一个距离 $\frac{1}{2} \cdot \Omega(h_{L-k(j)-\delta})$ 的偏差 $\Omega(h_{L-k(j)-\delta}) 2^{t_j-1} \varepsilon_{t_j}: z_{i_1} \oplus \dots \oplus z_{i_{t_j}} \oplus \bigoplus_{i=0}^{\delta-1} y'_i a_i \oplus \bigoplus_{i=\delta}^{\delta+k(j)-1} x_i^{(j)} a'_i \oplus \bigoplus_{i=\delta+k(j)}^{L-1} h_i^{(j)} a''_i = 0$. 当猜测错误时, 这样一个偏差应不存在. 这是该算法的基础.

因此, 累积的偏差平方和就是

$$\begin{aligned} & \sum_{h_{L-k(j)-\delta}} \left(u(h_{L-k(j)-\delta}) - \frac{\Omega(h_{L-k(j)-\delta})}{2} \right)^2 \\ &= \sum_{h_{L-k(j)-\delta}} \left(\frac{|H_{h_{L-k(j)-\delta}}(\omega)| + \Omega(h_{L-k(j)-\delta})}{2} - \frac{\Omega(h_{L-k(j)-\delta})}{2} \right)^2 \\ &= \sum_{h_{L-k(j)-\delta}} \frac{(H_{h_{L-k(j)-\delta}}(\omega))^2}{4}. \end{aligned}$$

这正是上述算法中计算的 B_ω . 该算法的判断条件可以改写为

$$B_\omega \geq T^{(j)} \Leftrightarrow \sum_{h_{L-k(j)-\delta}} \left(u(h_{L-k(j)-\delta}) - \frac{\Omega(h_{L-k(j)-\delta})}{2} \right)^2 \geq T^{(j)}. \quad (6)$$

明显地, 当 $(a_\delta, \dots, a_{\delta+k(j)-1})$ 猜测正确时, $|u(h_{L-k(j)-\delta})|$ 将服从二项分布 $(\Omega(h_{L-k(j)-\delta}), q_j)$; 否则它将服从另一个二项分布 $(\Omega(h_{L-k(j)-\delta}), \frac{1}{2})$. 若用正态分布去逼近二项分布, 则

$$\sum_{h_{L-k(j)-\delta}} \left(u(h_{L-k(j)-\delta}) - \frac{\Omega(h_{L-k(j)-\delta})}{2} \right)^2 \geq T^{(j)}$$

等价于

$$\frac{\Omega(h_{L-k(j)-\delta}) s_j}{4q_j(1-q_j)} \geq \sum_{h_{L-k(j)-\delta}} \frac{\left(u(h_{L-k(j)-\delta}) - \frac{\Omega(h_{L-k(j)-\delta})}{2} \right)^2}{\Omega(h_{L-k(j)-\delta}) q_j(1-q_j)} \geq \frac{T^{(j)}}{\Omega(h_{L-k(j)-\delta}) q_j(1-q_j)}. \quad (7)$$

当 $(a_\delta, \dots, a_{\delta+k(j)-1})$ 猜测正确时, 等价于

$$\Omega(h_{L-k(j)-\delta}) s_j \geq \sum_{h_{L-k(j)-\delta}} \frac{\left(u(h_{L-k(j)-\delta}) - \frac{\Omega(h_{L-k(j)-\delta})}{2} \right)^2}{\left(\frac{1}{2} \sqrt{\Omega(h_{L-k(j)-\delta})} \right)^2} \geq \frac{4T^{(j)}}{\Omega(h_{L-k(j)-\delta})}; \quad (8)$$

当 $(a_\delta, \dots, a_{\delta+k(j)-1})$ 猜测错误时, (7) 式可改写为

$$\begin{aligned} \frac{\Omega(h_{L-k(j)-\delta}) s_j}{4q_j(1-q_j)} &\geq \sum_{h_{L-k(j)-\delta}} \frac{(u(h_{L-k(j)-\delta}) - \Omega(h_{L-k(j)-\delta})q_j + \Omega(h_{L-k(j)-\delta}) \cdot 2^{t_j-1}\varepsilon^{t_j})^2}{\Omega(h_{L-k(j)-\delta})q_j(1-q_j)} \\ &= \sum_{h_{L-k(j)-\delta}} \left(\frac{u(h_{L-k(j)-\delta}) - \Omega(h_{L-k(j)-\delta})q_j}{\sqrt{\Omega(h_{L-k(j)-\delta})q_j(1-q_j)}} + \frac{\Omega(h_{L-k(j)-\delta})2^{t_j-1}\varepsilon^{t_j}}{\sqrt{\Omega(h_{L-k(j)-\delta})q_j(1-q_j)}} \right)^2 \\ &\geq \frac{T^{(j)}}{\Omega(h_{L-k(j)-\delta})q_j(1-q_j)}. \end{aligned} \quad (9)$$

注意当 $(a_\delta, \dots, a_{\delta+k(j)-1})$ 猜测正确时, $\frac{u(h_{L-k(j)-\delta}) - \Omega(h_{L-k(j)-\delta})q_j}{\sqrt{\Omega(h_{L-k(j)-\delta})q_j(1-q_j)}}$ 服从标准正态分布 $N(0, 1)$. 当 $(a_\delta, \dots, a_{\delta+k(j)-1})$ 猜测错误时, 变量 $\frac{u(h_{L-k(j)-\delta}) - \Omega(h_{L-k(j)-\delta})q_j/2}{\frac{1}{2}\sqrt{\Omega(h_{L-k(j)-\delta})q_j(1-q_j)}}$ 也服从标准正态分布 $N(0, 1)$. (8) 式意味着当 $(a_\delta, \dots, a_{\delta+k(j)-1})$ 猜测错误时, $\sum_{h_{L-k(j)-\delta}} \frac{(u(h_{L-k(j)-\delta}) - \Omega(h_{L-k(j)-\delta})q_j/2)^2}{(\frac{1}{2}\sqrt{\Omega(h_{L-k(j)-\delta})q_j(1-q_j)})^2}$ 服从中心 Chi-square 分布, 而 (9) 式则意味着当 $(a_\delta, \dots, a_{\delta+k(j)-1})$ 正确猜测时, $\sum_{h_{L-k(j)-\delta}} \frac{(u(h_{L-k(j)-\delta}) - \Omega(h_{L-k(j)-\delta})q_j)^2}{\Omega(h_{L-k(j)-\delta})q_j(1-q_j)}$ 将服从非中心 Chi-square 分布. 这是本文用来淘汰错误猜测的标准.

具体来说, 令 $\Gamma(y) = \int_0^{+\infty} e^{-x} x^{y-1} dx$ 表示 Gamma 函数. 由于存在 s_j 个不同的 $h_{L-k(j)-\delta}$, 一个中心 Chi-square 分布的概率密度函数为 $\phi_1(x) = \frac{x^{\frac{s_j-2}{2}} e^{-\frac{x}{2}}}{2^{\frac{s_j}{2}} \Gamma(\frac{s_j}{2})}$ 对 $x > 0$; 而一个非中心 Chi-square 分布的概率密度函数为 $\phi_2(x) = \frac{e^{-(x+\gamma^2)/2}}{2^{\frac{s_j}{2}}} \sum_{j=0}^{\infty} \frac{x^{j-1+\frac{s_j}{2}} \gamma^{2j}}{\Gamma(j+\frac{s_j}{2}) 2^{2j} j!}$ 对 $x > 0$, 这里 $\gamma^2 = \sum_{h_{L-k(j)-\delta}} (\frac{\sqrt{\Omega(h_{L-k(j)-\delta})q_j(1-q_j)} 2^{t_j-1}\varepsilon^{t_j}}{\sqrt{q_j(1-q_j)}})^2$. 这样, 正确的 $(a_\delta, \dots, a_{\delta+k(j)-1})$ 猜测可以导致 $B(a_\delta, \dots, a_{\delta+k(j)-1}) \geq T^{(j)}$ 的概率为

$$P_{\text{right}} = \int_{\frac{T^{(j)}}{\Omega(h_{L-k(j)-\delta})q_j(1-q_j)}}^{\frac{\Omega(h_{L-k(j)-\delta})s_j}{4q_j(1-q_j)} + 0.5} \phi_2(x) dx, \quad (10)$$

而一个错误猜测 $(a_\delta, \dots, a_{\delta+k(j)-1})$ 会通过检测的概率为

$$P_{\text{wrong}} = \int_{\frac{4T^{(j)}}{\Omega(h_{L-k(j)-\delta})}}^{\Omega(h_{L-k(j)-\delta})s_j + 0.5} \phi_1(x) dx. \quad (11)$$

在该算法中, 可以通过仔细选择 $T^{(j)}$ 来控制上述概率. 典型的选择是使得 $P_{\text{wrong}} < 2^{-k^{(j)}}$ 而同时 P_{right} 保持一定的数值, 亦即, 没有一个错误猜测可以通过检测, 而正确的猜测可以一定的概率通过该检测. 作为一个总结, 本文有:

定理 2 设上述算法第 1 步的成功概率为 P_{right} , 而 $P_{\text{wrong}} < 2^{-k^{(1)}}$, 则其时间复杂度为

$$\sum_{i=1}^m \sum_{h_{L-\sum_{j=1}^i k^{(j)}}} (2^{k^{(i)}} k^{(i)} + \Omega(h_{L-\sum_{j=1}^i k^{(j)}})(t_i + k^{(i)}))$$

次操作, 其存储复杂度为至多

$$\sum_{i=1}^m \left(2^{k^{(i)}} + \sum_{h_{L-\sum_{j=1}^i k^{(j)}}} (t_i \lceil \log_2 N_i \rceil + L) \Omega(h_{L-\sum_{j=1}^i k^{(j)}}) \right)$$

表 2 实验结果与比较

Table 2 Experimentnal results and comparison with [7]

	Algorithm in Ref. [7]	Our algrithm	Algorithm in Ref. [7]	Our algrithm	Algorithm in Ref. [7]	Our algrithm	Algorithm in Ref. [7]	Our algrithm	Algorithm in Ref. [7]	Our algrithm
L	40	40	40	40	60	60	60	60	60	60
p	0.64	0.64	0.55	0.55	0.57	0.57	0.68	0.68	0.6	0.6
t	2	3	2	3	2	3	3	5	2	5
N	$4 \cdot 10^5$	$5 \cdot 10^3$	$4 \cdot 10^5$	10^4	$4 \cdot 10^7$	$2 \cdot 10^5$	$1.5 \cdot 10^5$	$8 \cdot 10^3$	$4 \cdot 10^7$	$9 \cdot 10^3$
Time $\approx$	3 min	5 s	3 min	25 s	106 min	6 min	4.6 min	20 s	13 min	1 min

表 3 本文第 2 个算法的实验结果与理论预测

Table 3 Experimentnal results and theoretical predictions of the second algorithm

p	Attack	L	N	t_1	$k^{(1)}$	Time	Memory	Pre-computation
0.531	Ref. [3]	40	80000	3	1	2^{31}	$2^{34.1}$	2^{37}
0.531	Ref. [14]	40	2^{22}	2	20	2^{24}	$2^{32.8}$	2^{27}
0.531	Ours	40	2^{20}	2	23	2^{28}	$2^{28.5}$	2^{21}
0.531	Ref. [3]	89	2^{28}	3	1	2^{44}	$2^{35.81}$	2^{61}
0.531	Ref. [14]	89	2^{32}	3	26	2^{32}	$2^{41.12}$	2^{37}
0.531	Ours	89	2^{30}	3	26	$2^{38.2}$	2^{38}	2^{56}
0.51	Ref. [3]	40	80000	3	1	2^{40}	$2^{40.54}$	2^{37}
0.51	Ref. [14]	40	2^{24}	2	22	2^{29}	$2^{35.13}$	2^{29}
0.51	Ours	40	2^{22}	2	20	$2^{32.9}$	$2^{34.7}$	2^{23}

比特, 而数据复杂度为 N 比特, 由第 1 步中的式 (10), (11) 和 $\Omega(h_{L-k^{(1)}})$ 决定.

证明 对于时间复杂度, 由于上述共计算法使用 FWT 来同时计算 $H_{h_{L-k^{(j)}-\delta}}(\omega)$ 的 $2^{k^{(j)}}$ 个 Walsh 谱值, 因此, 算法第 j 步中的替换和估值步骤的时间复杂度为

$$\sum_{h_{L-\sum_{i=1}^j k^{(i)}}} \left(2^{k^{(j)}} k^{(j)} + \Omega \left(h_{L-\sum_{i=1}^j k^{(i)}} \right) (t_j + k^{(j)}) \right)$$

次操作, 这其中 $\Omega(h_{L-\sum_{i=1}^j k^{(i)}}) (t_j + k^{(j)})$ 次操作用来准备函数 $h^{(i)}(x_\delta^{(i)}, \dots, \dots, x_{\delta+k^{(i)}-1}^{(i)}, h_{\delta+k^{(i)}}^{(i)}, \dots, h_{L-1}^{(i)})$, 而 $2^{k^{(j)}} k^{(j)}$ 次操作用来进行 FWT. 相应地, 第 j 步中的存储消耗为 $2^{k^{(j)}} + \sum_{h_{L-\sum_{i=1}^j k^{(i)}}} (t_j \lceil \log_2 N_j \rceil + L) \Omega(h_{L-\sum_{i=1}^j k^{(i)}})$ 比特, 这主要是由于本文使用下述直接的方法来存储一个如下形式的校验等式 $z_{i_1} \oplus \dots \oplus z_{i_{t_j}} \oplus \oplus_{i=0}^{\delta-1} y'_i a_i \oplus \oplus_{i=\delta}^{\delta+k^{(j)}-1} x_i^{(j)} a'_i \oplus \oplus_{i=\delta+k^{(j)}}^{L-1} h_i^{(j)} a''_i, 1) t_j \cdot \lceil \log_2 N_j \rceil$ 比特用来表示 t_j 个整数 $i_1, i_2, \dots, i_{t_j}$; 2) $k^{(j)}$ 比特用来表示系数向量 $(x_\delta^{(j)}, \dots, x_{\delta+k^{(j)}-1}^{(j)})$, 而 $L - \sum_{i=1}^j k^{(i)}$ 比特用来表示向量 $h_{L-\sum_{i=1}^j k^{(i)}}$.

本文进行了一定的计算机实验来检查上述算法的实际性能. 总体来说, 该算法的模拟结果与理论分析较为符合. 下面首先给出该算法在与文献 [7] 中算法相同条件下的实验结果. 表 2 只列出了一些典型例子, 其他情况这里由于篇幅限制略去.

文献 [7] 中的实验在一台安装 Solaris 的 Sun Ultra-80 上进行, 而本文的实验平台是一台安装 Windows XP 的 Pentium 4 PC. 表 2 说明本文算法能够成功地削减时间与数据复杂度.

表 4 本文第 2 个算法的理论估计, 及与另两篇文献中算法的比较 (文献 [14] 中对 $L = 61$ 和 $p = 0.501$ 情况下的解码分析不正确, 这里根据其中的公式列出了正确结果)

Table 4 Theoretical estimations of the second algorithm and comparisons with [14] (The result in [14] for the case of $L = 61$ and $p = 0.501$ is not correct, here we list the correct result obtained according to the formulas in [14])

p	Attack	L	N	t_1	$k^{(1)}$	Time	Memory	Pre-computation
0.501	Ref. [14]	61	2^{36}	2	22	2^{43}	$2^{49.71}$	2^{42}
0.501	Ours	61	2^{38}	2	24	$2^{46.7}$	$2^{49.1}$	2^{38}
0.531	Ref. [14]	103	2^{36}	3	29	2^{34}	$2^{38.51}$	$2^{41.17}$
0.531	Ours	103	2^{30}	4	22	$2^{43.2}$	$2^{46.2}$	2^{60}

图 2 蓝牙流密码 E0

Figure 2 Bluetooth stream cipher E0

在公开文献中, 还有其他的快速相关攻击算法发表, 但它们均不如文献 [3,14] 中的算法有效. 因此, 本文取第 2 个算法来与文献 [3] 和 [14] 中的算法进行比较. 表 3 和 4 表明本文的算法与之相比毫不逊色. 这里只列出导致较为平均的复杂度估计的参数选择, 亦即, 时间、存储、数据和预计算复杂度的尽量平衡. 其他的参数选择也是可以的, 但它们不具有这种较为平均的性质. 可以看到, 本文算法很灵活, 并且与其他算法相比, 可比性很强.

4 分析实例

4.1 一级蓝牙流密码 E0 的攻击

BluetoothTM[17] 定义了一个流密码 E0, 用来在蓝牙网络内实现点到点的加密. 本文只研究一级蓝牙 E0 的攻击, 即只研究初始化之后的密钥流产生算法. 从图 2 可以看出, E0 使用 4 个长度分别为 25, 31, 33 和 39 比特的 LFSR, 其每个反馈多项式均为本源多项式, 且具有 5 个非零抽头. 在时刻 t , 密钥流比特 z_t 按如下方式产生.

首先, 将 4 个 LFSR 的输出比特 ($1 \leq i \leq 4$) 按整数相加, 亦即, 令 $y_t = \sum_{i=1}^4 x_t^i$. 接着如下更新 4 比特的记忆, 由 $(c_{t-1}^1, c_{t-1}^0, c_t^1, c_t^0)$ 到 $(c_t^1, c_t^0, c_{t+1}^1, c_{t+1}^0)$: $s_{t+1} = (s_{t+1}^1, s_{t+1}^0) = \left\lfloor \frac{y_t + 2 \cdot c_t^1 + c_t^0}{2} \right\rfloor$, $c_{t+1}^1 = s_{t+1}^1 \oplus c_t^1 \oplus c_{t-1}^0$ 和 $c_{t+1}^0 = s_{t+1}^0 \oplus c_t^0 \oplus c_{t-1}^1 \oplus c_{t-1}^0$. 最后, 产生 $z_t = \oplus_{i=1}^4 x_t^i \oplus c_t^0$.

表 5 本文攻击与已知攻击的复杂度比较

Table 5 Complexities comparisons of our attack with other attacks

Attack	Keystream	Time	Memory	Pre-computation
Ref. [19]	$2^{23.07}$	$2^{67.58}$	$2^{46.14}$	—
Ref. [20, 25]	$2^{23.4}$	2^{49}	2^{37}	2^{28}
Ref. [8]	2^{39}	2^{39}	2^{27}	2^{37}
Ref. [22]	2^{34}	2^{63}	2^{34}	2^{54}
Ref. [24]	2^7	2^{81}	2^{77}	—
Ref. [23]	2^7	2^{81}	2^{42}	—
Ref. [26]	2^7	2^{87}	2^{23}	—
Ours	2^{37}	$2^{35.1}$	2^{32}	2^{49}

图 3 LILI-128

Figure 3 LILI-128

文献 [33] 中列出了 E0 的各种各样的相关性质, 但之前的解码算法均无法充分利用这些相关性来有效解码, 下面本文使用第 2 个算法来建立一个新的针对一级蓝牙流密码的密钥恢复攻击.

设 $W \cdot X^f = \bigoplus_{i=1}^4 \bigoplus_{j=0}^{f-1} \omega_{ij} x_j^i$ 和 $V \cdot Z^f = \bigoplus_{i=0}^{f-1} v_i z_i$ 分别表示由一个矩阵 W 和一个向量 V 定义的线性相关. 对于每个长为 f 的密钥流段落 $Z^f = (z_0, \dots, z_{f-1})$ 和较小的 f , 相关系数 $\Delta(W \cdot X^f, 0|Z^f)$ 可以使用 FWT 来计算. 根据文献 [33], 若 $f = 4$, 则存在 96 个 W s 满足上述条件相关公式, 且对每个 Z^4 值, 相关值为 $0.5 + (1/16)/2 = 0.53125$. 本文攻击的复杂度列在了表 5 中, 其中还包括了与其他攻击的比较.

如果攻击者具有 2^{37} 比特密钥流, 则选择 $t_1 = 3$, $s_1 = 8$ 和 $k^{(1)} = 26$, 亦即, 首先在第 1 步恢复长度为 25 比特的 LFSR 和其他 1 比特, 这里共有 $(\binom{2^{37}-3}{3} \times 96) / 2^{128-26} \approx 2^{26.09}$ 个重量为 3 的校验等式. 若攻击者选择 $T^{(1)} = 2^{30.4594}$, 则有 $P_1^{(1)} = 0.886882$ 和 $P_2^{(1)} = 2^{-29.25}$. 当只有一个处理器可供利用时, 时间复杂度为 $20 \times (2^{26.09} + 2^{26} \times 26) \approx 2^{35.1}$ 次操作. 在实际中, 可以完成 2^{49} 次操作的预计算来构造校验等式, 比如利用一个工作站或者一个服务器 (比如, Dell Power Edge 7250). 这非常值得去做, 因为预计算只进行一次.

第 1 步完成后, 剩下的 102 比特能够通过上面的多步法快速相关攻击恢复, 或者通过一个新的多步法攻击, 但利用基于密钥流和 25 比特 LFSR 输入的相关性 [33]. 这两种情况都具有低得多的复杂度, 因此, 总的攻击复杂仍然是 $2^{35.1}$ 次操作.

4.2 LILI-128 的攻击

LILI-128 是一个欧洲 NESSIE 计划的候选算法. 它包含两个子系统, 如图 3 所示.

表 6 本文攻击与其他已知攻击的比较

Table 6 Comparison of our attack with other best known attacks

Attack	Ref. [21, 25]	Ref. [29]	Ref. [28]	Ref. [30]	Ref. [27]	Ref. [31]	Ours
Keystream	2^{60}	2^{46}	2^{30}	2^7	2^{47}	$2^{29.0875}$	2^{24}
Time	2^{40}	2^{48}	2^{71}	$2^{99.1}$	2^{53}	2^{64}	$2^{70.6}$

在钟控子系统中, LFSR c 长度为 39 比特, 且具有本源反馈多项式. 布尔函数 f_c 用来产生一个钟控值 $c_k = f_c(y_1, y_2) = 2y_1 + y_2 + 1$, 这里 y_1 和 y_2 均取自 LFSR c 的第 12 和第 20 级. 然后将数据产生子系统内的长为 89 比特的 LFSR d 向前运行 c_k 步, 并将布尔函数 f_d 作用于其状态来 (用到 10 个变量) 产生密钥流比特 z_k . 请看文献 [18]. 虽然这个流密码已经被发现安全性较弱, 但近年仍有连续的密码分析结果发表, 甚至见于著名的美密会和 IEEE IT 杂志的常规论文.

注意到函数 f_d 有 240 个偏差为 0.53125 的线性逼近, 若攻击者获得 $N = 2^{24}$ 比特的密钥流, 则可构造一个如下的多步法快速相关攻击. 首先穷搜 LFSR c 的所有可能初态, 并导出每个钟控值 c_k . 这样可以获得具体的采样值, 并导出一个包含 89 个变量和 $2^{24} \times 240$ 个概率等式的方程组. 使用本文的第 2 个算法解这一概率线性系统, 可以导出 LFSR d 的 89 比特状态. 具体来说, 令 $k^{(1)} = 21$, $t_1 = 3$ 和 $s_1 = 40$, 则共有 $\binom{2^{24} \times 240}{3} / 2^{89-21}$ 个重量为 5 的校验等式. 使用本文的第 2 个算法, 可以超过 86% 的成功概率, 恢复出 21 比特的信息. 时间复杂度为 $40(2^{21} \times 21 + 36863523) = 2^{31.6}$ 次操作, 而存储复杂度为 $2^{24.5}$ 字节. 预计算的复杂度为 $2^{36.9}$ 次操作. 总的时间复杂度为 $2^{39+31.6} = 2^{70.6}$ 次操作. 表 6 给出了本文攻击与其他已知攻击的比较.

5 结束语

本文提出了两个新的进行多步法快速相关攻击的算法, 并通过理论分析和计算机实验验证了它们的有效性. 新算法适用于任何类型的 LFSR, 其中第 2 个算法给出了更多更有效的关于复杂度的折中. 作为应用, 相应的对于一级蓝牙流密码 E0 和 LILI-128 的结果表明, 快速相关攻击比之前预想的要具有威胁. 笔者相信本文的新方法适用于基于 LFSR 的流密码的安全性分析, 特别是其他方法, 比如代数攻击无法奏效时.

参考文献

- 1 Canteaut A, Trabbia M. Improved fast correlation attacks using parity-check equations of weight 4 and 5. In: Advances in Cryptology-EUROCRYPT'2000. Berlin: Springer-Verlag, 2000. 573-588
- 2 Chepyzhov V V, Johansson T, Smeets B. A simple algorithm for fast correlation attacks on stream ciphers. In: Fast Software Encryption-FSE'2000. Berlin: Springer-Verlag, 2000. 181-195
- 3 Chose P, Joux A, Mitton M. Fast correlation attacks: An algorithmic point of view. In: EUROCRYPT'2002. Berlin: Springer-Verlag, 2002. 209-221
- 4 Golic J D, Hawkes P. Vectorial approach to fast correlation attacks. Des Cod Cryptogr, 2005, 35: 5-19
- 5 Johansson T, Jonsson F. Fast correlation attacks based on turbo code techniques. In: Advances in Cryptology-CRYPTO'99. Berlin: Springer-Verlag, 1999. 181-197
- 6 Johansson T, Jonsson F. Improved fast correlation attacks on stream ciphers via convolutional codes. In: Advances in Cryptology-EUROCRYPT'99. Berlin: Springer-Verlag, 1999. 347-362

- 7 Johansson T, Jonsson F. Fast correlation attacks through reconstruction of linear polynomials. In: CRYPTO'2000. Berlin: Springer-Verlag, 2000. 300–315
- 8 Lu Y, Vaudenay S. Faster correlation attack on Bluetooth keystream generator E0. In: CRYPTO'2004. Berlin: Springer-Verlag, 2004. 407–425
- 9 Meier W, Staffelbach O. Fast correlation attacks on certain stream ciphers. J Cryptol, 1989, 1: 159–176
- 10 Menezes A J, Van Oorschot P C, Vanstone S A. Handbook of Applied Cryptography. Boca Raton FL: CRC Press, 1996
- 11 Mihaljevic M, Fossorier M P C, Imai H. A low-complexity and high-performance algorithm for fast correlation attack. In: Fast Software Encryption-FSE'2000. Berlin: Springer-Verlag, 2000. 196–212
- 12 Mihaljevic M, Fossorier M P C, Imai H. Fast correlation attack algorithm with listing decoding and an application. In: Fast Software Encryption-FSE'2001. Berlin: Springer-Verlag, 2001. 208–222
- 13 Siegenthaler T. Decrypting a class of stream ciphers using ciphertext only. IEEE Trans Comput, 1985, 34: 81–85
- 14 Zhang B, Feng D. Multi-pass fast correlation attack on stream ciphers. In: Selected Areas in Cryptography-SAC'2006. Berlin: Springer-Verlag, 2007. 234–248
- 15 Lu Y, Vaudenay S. Cryptanalysis of Bluetooth keystream generator two-level E0. In: ASIACRYPT'2004. Berlin: Springer-Verlag, 2004. 483–499
- 16 Lu Y, Meier W, Vaudenay S. The conditional correlation attack: A practical attack on Bluetooth encryption. In: Advances in Cryptology-CRYPTO'2005. Berlin: Springer-Verlag, 2005. 97–117
- 17 Bluetooth T M. Bluetooth Specification. Version 1.2. [November 2003]. <http://www.bluetooth.org>
- 18 Clark A, Dawson E, Fuller J, et al. The LILI-128 keystream generator. In: Selected Areas in Cryptography-SAC'2000. Berlin: Springer-Verlag, 2000
- 19 Armknecht F, Krause M. Algebraic attacks on combiners with memory. In: Advances in Cryptology-CRYPTO'2003, Berlin: Springer-Verlag, 2003. 162–175
- 20 Courtois N T. Fast algebraic attacks on stream ciphers with linear feedback. In: Advances in Cryptology-CRYPTO'2003. Berlin: Springer-Verlag, 2003. 176–194
- 21 Courtois N T, Meier W. Algebraic attacks on stream ciphers with linear feedback. In: EUROCRYPT'2003. Berlin: Springer-Verlag, 2003. 345–359
- 22 Ekdahl P. On LFSR based stream ciphers (analysis and design). PhD thesis. Lund: Lund University, 2003
- 23 Krause M, Stegemann D. Reducing the space complexity of BDD-based attacks on keystream generators. In: Fast Software Encryption-FSE'2006. Berlin: Springer-Verlag, 2006. 163–178
- 24 Krause M. BDD-based cryptanalysis of keystream generators. In: Advances in Cryptology-EUROCRYPT'2002. Berlin: Springer-Verlag, 2002. 222–237
- 25 Hawkes P, Rose G. Rewriting variables: The complexity of fast algebraic attacks on stream ciphers. In: Advances in Cryptology-CRYPTO'2004. Berlin: Springer-Verlag, 2004. 390–406
- 26 Shaked Y, Wool A. Cryptanalysis of the Bluetooth E0 cipher using OBDD's. In: Information Security Conference-ISC'2006. Berlin: Springer-Verlag, 2006. 187–202
- 27 Englund H, Johansson T. A new simple technique to attack filter generators and related ciphers. In: Selected Areas in Cryptography-SAC'2004. Berlin: Springer-Verlag, 2005. 39–53
- 28 Johansson T, Jonsson F. A fast correlation attack on LILI-128. Inf Process Lett, 2002, 81: 127–132
- 29 Saarinen M J O. A time-memory tradeoff attack against LILI-128. In: Fast Software Encryption-FSE'2002. Berlin: Springer-Verlag, 2002. 231–236
- 30 Tsunoo Y, Saito T, Shigeri M, et al. Shorter bit sequence is enough to break stream cipher LILI-128. IEEE Trans Inf Theory, 2005, 51: 4312–4319
- 31 Molland H, Helleseht T. An improved correlation attack against irregular clocked and filtered keystream generators. In: Advances in Cryptology-CRYPTO'2004. Berlin: Springer-Verlag, 2004. 373–389
- 32 Shannon C E. A mathematical theory of communication. Bell Syst Tech, 1948, 27
- 33 Golic J D. Linear cryptanalysis of Bluetooth stream cipher. In: Advances in Cryptology-EUROCRYPT'2002. Berlin: Springer-Verlag, 2002. 238–255

Improved multi-pass fast correlation attacks with applications

ZHANG Bin* & FENG DengGuo

State Key Laboratory of Information Security, Institute of Software, Chinese Academy of Sciences, Beijing 100190, China

*E-mail: zhangbin@is.iscas.ac.cn

Abstract In this paper we propose two new algorithms for multi-pass fast correlation attacks on stream ciphers. The first algorithm aims at fast symbol-wise decoding in the circumstances that the noise is not very high and we have little resource for pre-computation. The second algorithm deals with the practical decoding problem in the high noise and limited keystream cases. The new algorithms are applicable to arbitrary form LFSR and compare favorably to the previously known algorithms in the scenarios under consideration. As applications, we demonstrate new key recovery attacks on one-level Bluetooth E0 and LILI-128, respectively. Given 2^{37} -bit keystream and 2^{28} -byte memory, our attack against one-level E0 needs $2^{35.1}$ operations. Given 2^{24} -bit keystream and $2^{24.5}$ -byte memory, our attack on LILI-128 has time complexity $2^{70.6}$ operations.

Keywords stream cipher, fast correlation attacks, linear feedback shift register (LFSR), Bluetooth E0, LILI-128

ZHANG Bin was born in 1976. He received the Ph.D. degree in 2005 from the Graduate University of Chinese Academy of Sciences, Beijing. Currently, he is an Associate Professor at Institute of Software, Chinese Academy of Sciences. His research interests include the design and analysis of stream ciphers.

FENG DengGuo was born in 1965. He is a Professor and doctoral supervisor at State Key Laboratory of Information Security, Institute of Software, Chinese Academy of Sciences. His research areas are information security and network security. He is the fellow and Chair of the Chinese Association for cryptology research.