

# 对自同步混沌密码的分割攻击方法

金晨辉, 杨 阳

(解放军信息工程大学电子技术学院, 河南郑州 450004)

**摘 要:** 本文分析了自同步混沌密码的信息泄漏规律, 并据此提出了对自同步混沌密码的分割攻击方法. 由于能够利用所有时刻的已知明文进行攻击, 因而自同步混沌密码比同步混沌密码的抗分割攻击能力更弱. 本文以 Hong Zhou 等提出的自同步混沌密码为例, 完成了对密钥规模为 64 比特且以混沌映射的 65 次迭代为自同步映射的自同步密码的分割攻击. 利用 1 万个已知明文, 在主频为 2.5GHz 的 Pentium-4 PC 上, 攻击的平均时间为 25 小时 22 分, 成功率为 0.86.

**关键词:** 混沌密码; 自同步密码; 分割攻击; 已知明文攻击

**中图分类号:** TN 918.1 **文献标识码:** A **文章编号:** 0372-2112 (2006) 07-1337-05

## A Divide-and-Conquer Attack on Self-Synchronous Chaotic Ciphers

JIN Chen-hui, YANG Yang

(Electronic Technology Institute, PLA Information Engineering University, Zhengzhou, Henan 450004 China)

**Abstract** In this paper the authors analysis the law of information leak of self-synchronous chaotic ciphers hereby present a divide-and-conquer attack to self-synchronous chaotic ciphers which is the first attack of this kind in this area. Because the output signals at each clock can be used in the attack, the ability of self-synchronous chaotic ciphers to resist the divide-and-conquer attack is weaker than that of synchronous chaotic ciphers. As an example we realized the divide-and-conquer attacks to the self-synchronous chaotic cipher proposed by Hong Zhou etc with 64 bits key and the self-synchronous chaotic map being the 65 iterations of a chaotic map. It took about 25 hours and 22 minutes to recover the key at a success rate 0.86 on a Pentium 4/2.5GHz personal computer with 10000 known plaintexts.

**Key words** chaotic cipher; self-synchronous chaotic cipher; divide-and-conquer attack; known plaintexts attack

## 1 引言

混沌密码是一类新型的密码算法, 在数据加密、图像加密和信息隐藏等领域的应用研究日益增多, 因此, 研究混沌密码的安全性及安全性分析方法具有重要的意义. 文献 [2] 提出了对混沌密码的多分辨率攻击方法, 该方法平均将文献 [3] 提出的混沌密码的密钥熵降低 2 比特. 但该方法所需的已知明文量和计算复杂度与密钥的分辨率 [3] 有关, 分辨率越小, 计算复杂性和数据复杂性越小. 由于密钥在随机情况下的分辨率很大, 因而该攻击方法平均所需的已知明文量和计算复杂度一般都很大. 文献 [4] 和文献 [5] 发现了迭代型混沌密码的输出序列初态的低位比特对前若干输出信号的影响不大这个信息泄漏, 并利用吻合度 [5] 的分布规律对该信息泄漏进行了定量刻画, 进而提出了对迭代型混沌密码的分割攻击方法, 从而利用不长的乱数序列大大降低了对迭代型混沌密码攻击的计算复杂性, 该分割攻击方法本质上也是一种差分攻击. 然而, 其他类型的混沌密码的信息泄漏有何特点, 以及如何利用其信息泄漏规律对它们进行分析, 还有待深入研究和解决. 本文

将研究对自同步混沌密码的攻击方法问题.

首先介绍自同步密码的模型. 设  $f_k(x)$  是具有  $t$  个二元输入变量和  $h$  个二元输出变量的多输出 Boolean 函数,  $\{m_i\}_{i=1}^{\infty}$ ,  $\{d_i\}_{i=1}^{\infty}$  和  $\{c_i\}_{i=1}^{\infty}$  分别是  $2^t$  元明文序列、 $2^h$  元乱数序列和  $2^h$  元密文序列, 则自同步密码的模型就是  $c_n = m_n \oplus f_k(c_{n-1}, c_{n-2}, \dots, c_{n-t})$ , 其中  $(c_{-t+1}, c_{-t+2}, \dots, c_0)$  是初始化向量,  $k$  是自同步密码的密钥. 此时称该自同步密码为  $t$  步  $h$  比特的自同步密码, 称  $f_k$  为自同步映射. 自同步密码的优点在于只要连续  $t$  个密文块正确,  $t$  步自同步密码就能对后续的密文脱密, 与前面接收的密文块有多少错误或丢失无关, 因而有重要的应用价值.

对于自同步密码, 由于  $f_k(c_{n-1}, c_{n-2}, \dots, c_{n-t}) = m_n \oplus c_n$ , 故在已知明文攻击的条件下, 利用连续  $N+t$  个已知的  $h$  比特密文块  $c_{e+t+1}, c_{e+t+2}, \dots, c_{e+N-1}, c_{e+N}$  和  $N$  个对应的明文块  $m_{e+1}, m_{e+2}, \dots, m_{e+N}$ , 就可得到自同步映射  $f_k$  的  $N$  个输入  $(c_{e+t+1}, c_{e+t+2}, \dots, c_{e+t+t})$  及对应的输出  $m_{e+i} \oplus c_{e+i}$ , 这里  $1 \leq i \leq N$ . 此时, 基于  $f_k$  的  $N$  个输入输出对, 就可对密钥  $k$  实施穷举攻击. 但是, 只要密钥量足够大, 就足以挫败穷举攻击. 要成功对自同步密码实施攻击, 还需发掘和利用自

同步映射自身的信息泄漏规律.

现分析基于混沌映射或其多次复合构造的自同步映射  $f_k$  的信息泄漏规律. 此时, 由于  $f_k$  在绝大多数点  $k$  上连续, 故若记  $k^{(m)}$  是将小于 1 的实数  $k$  除高  $m$  比特外全部设置为 0 所得的新密钥, 则以  $k$  为变量的函数  $f_k$  的输入差  $|k^{(m)} - k| < 2^{-m}$ , 故由  $f_k$  在  $k$  点连续知, 输出差  $|f_k(x) - f_{k^{(m)}}(x)|$  的取值一般偏小, 因而在  $\{0, 1\}^h$  中不服从均匀分布, 从而造成密钥信息的泄漏. 此外, 在已知明文攻击条件下,  $f_k$  的输入和输出还是已知的, 已知的明文序列越长, 获得的  $f_k$  的输入-输出对就越多, 因而获得的密钥信息就越多, 攻击成功的概率就越大. 然而, 如何定量刻划这种信息泄漏, 以及如何收集和利用这种信息泄漏还原密钥, 还需给出具体的解决方法和解决手段. 本文将解决这个问题. 下面以 Hong Zhou 等提出的自同步混沌密码算法<sup>[1,7]</sup>为例, 分析自同步混沌密码的信息泄漏规律, 给出定量刻划、收集和利用这些信息泄漏还原密钥的方法, 给出对自同步混沌密码的分割攻击方法. 利用本文提出的方法, 当密钥规模为 64 比特并且以混沌映射的 65 次迭代作为自同步映射时, 在 Pentium-4 PC 上, 使用 1 万个连续的已知明文信号, 整个攻击的平均时间为 25 小时 22 分, 成功率为 0.86.

## 2 ZLY 自同步混沌密码算法及其信息泄漏规律

### 2.1 ZLY 自同步混沌密码算法<sup>[1,7]</sup>

文献[1]和文献[7]中提出的自同步混沌密码算法是利用一维分段线性混沌映射的多次迭代设计的.

一维分段线性混沌映射  $F_k: [0, 1) \rightarrow [0, 1)$  的定义<sup>[1,7]</sup>如下:

$$F_k(x) = \begin{cases} x/k & , 0 \leq x < k; \\ (x-k)/(0.5-k) & , k \leq x < 0.5; \\ 0 & , x = 0.5; \\ F_k(1-x) & , x \geq 0.5 \end{cases}$$

其中  $0 < k < 0.5$ . 此时文献[1]和文献[7]提出的自同步混沌密码算法(以下简称为 ZLY 自同步密码)的自同步映射  $f_k(x) = F_k^n(x)$  是函数  $F_k(x)$  的  $n$  次迭代, 这里  $n \geq 1$ .

设  $m_t \in [0, 1)$  和  $c_t$  分别是第  $t$  时刻的明文和密文, 则以  $k$  为密钥的 ZLY 自同步密码的加密算法为:

表 1 输出差值的分布规律

小区间序号	0	1	2	3	4	5	6	7	...	125	126	...	254	255
平均个数	20631	625	623	623	619	617	609	600	...	314	310	...	3	0

从实验结果可以看出, 输出差  $|F_k^{n(m)}(x) - F_k^n(x)|$  大都落入序号较小的小区间, 而且落入序号为 0 的小区间  $[0, 2^{-8})$  的概率最大, 说明输出差偏小, 从而证实了分析结论.

上述信息泄漏规律对于  $m$  的其他选择也同样存在. 采用同样的方法和样本数量, 我们对  $m$  的其他选择具体统计了输出差  $|F_k^{n(m)}(x) - F_k^n(x)|$  落入小区间  $[0, 2^{-8})$  的概率  $p_m$ . 如果输出差的分布是均匀的, 则概率应近似为  $2^{-8} \approx$

$$c_t = [m_t + F_k^n(c_{t-1})] \bmod 1$$

这里  $\bmod 1$  的含义是取实数的小数部分,  $c_0$  为初始化向量.

设  $x = \sum_{i=1}^{\infty} x_i / 2^i$  是一个非负小数且诸  $x_i \in \{0, 1\}$ , 则称  $\sum_{i=1}^m x_i / 2^i$  是  $x$  的  $m$  精度小数. 以下总将  $m$  维二元向量  $(x_1, x_2, \dots, x_m)$  与小数  $\sum_{i=1}^m x_i / 2^i$  之间相互转换, 且称  $(x_1, x_2, \dots, x_m)$  为  $x = \sum_{i=1}^{\infty} x_i / 2^i$  的高  $m$  位比特.

在具体实现时, ZLY 自同步混沌密码算法中的密钥  $k$ , 函数  $F_k(x)$  的输入和输出以及明密文都是  $L$  精度小数, 该算法约定迭代次数  $n > L$ . 显然, 当混沌映射的迭代次数  $n$  过大时, 将大大降低混沌密码的加密速度. 文献[2]和文献[6]得出结论,  $n = L + 1$  是 ZLY 密码算法的最好选择.

为简单起见, 本文以下均假定迭代次数  $n = L + 1$ .

### 2.2 自同步混沌密码算法的信息泄漏规律

对于 ZLY 自同步混沌密码, 在已知明文攻击的条件下, 函数  $F_k^n(c_{t-1}) = (c_t - m_t) \bmod 1$  的输入和输出都是已知的, 需要求解的是密钥  $k$ .

正如引言中分析的那样, 对任意固定的  $x$ , 以  $k$  为变量的函数  $F_k^n(x)$  都是间断连续函数, 根据间断连续函数的性质, 对绝大多数的  $k$ , 当变量  $k$  有微小变化时,  $F_k^n(x)$  的值也只有微小的变化. 特别地, 如果记  $k^{(m)}$  是  $k$  的  $m$  精度小数, 则  $|k - k^{(m)}| < 2^{-m}$ , 因而  $|F_{k^{(m)}}^n(x) - F_k^n(x)|$  的值将会偏小. 当迭代次数  $n = 1$  时, 可具体计算出  $|F_{k^{(m)}}^1(x) - F_k^1(x)|$  的平均值, 但当迭代次数  $n$  较大时, 要具体计算出  $|F_{k^{(m)}}^n(x) - F_k^n(x)|$  的平均值是非常困难的, 但我们可利用模拟实验方法得到它的分布规律. 具体方法如下:

将  $F_k^n(x)$  的输入、输出和密钥的精度都选定为 64 比特, 并选定迭代次数  $n = 65$ . 取  $m = 32$  然后从  $[0, 1)$  中随机选取 10 万个精度为 64 的小数  $x_1, x_2, \dots, x_{100000}$ , 再随机选择一个密钥  $k$ , 并计算输出差  $|F_k^n(x_i) - F_{k^{(m)}}^n(x_i)|$ . 最后将区间  $[0, 1)$  等分为 256 个小区间  $I_0, I_1, \dots, I_{255}$ , 并统计这 10 万个输出差落入各小区间的个数, 其中  $I_i = \left[ \frac{i}{256}, \frac{i+1}{256} \right)$ . 如果输出差的分布是均匀的, 则落入每个小区间的个数应在  $100000/256 \approx 390$  左右. 我们共对 10 万个随机选择的密钥进行了实验, 具体实验结果如表 1.

0.0039 但实验结果远大于此. 具体的实验结果如表 2.

表 2 落入区间  $[0, 2^{-8})$  的概率的统计规律

$m$	16	20	24	32	40	48	56
落入区间 $[0, 2^{-8})$ 的概率 $p_m$	0.0626	0.145	0.172	0.206	0.269	0.378	0.426

上述实验结果得到了  $k^{(m)}$  对应的输出差

$|F_k^{(m)}(x) - F_{k'}^{(m)}(x)|$  的分布规律. 对于与  $k^{(m)}$  不等的  $k'$ , 由于  $|k - k'|$  的值平均为  $0.5$  因而远大于差值  $|k^{(m)} - k| < 2^m$ , 故输出差  $|F_{k'}^{(m)}(x) - F_k^{(m)}(x)|$  一般不会严重偏小, 更不会出现以很大的概率落入小区间  $[0, 2^8)$  之中, 据此就可将正确的  $k^{(m)}$  与错误的  $k^{(m)}$  区分开来.

因此, 只要知道了  $F_k^{(m)}(x)$  的输入和输出就可通过对  $k^{(m)}$  的穷举, 筛选出正确的  $k^{(m)}$ . 由于  $k^{(m)}$  的变化量  $2^m$  远小于  $k$  的变化量, 因而这种分割攻击方法可大大降低密钥求解的计算复杂性.

### 3 对自同步混沌密码的分割攻击

首先给出攻击  $k^{(m)}$  的原理和具体算法.

由于在已知明文攻击条件下, 可以得到  $F_k^{(m)}(x)$  的  $N$  个输入  $c_1, c_2, \dots, c_N$  及对应的输出  $e_1, e_2, \dots, e_N$ , 其中  $e_i = (c_{i+1} - m_{i+1}) \bmod 1$ . 设  $k'$  是  $k^{(m)}$  的一个假设值, 则当  $k' = k^{(m)}$  时, 输出差  $|F_{k'}^{(m)}(c_i) - F_k^{(m)}(c_i)|$  落入  $[0, 2^8)$  的概率近似为  $p_m$ , 记

$$\xi_i(k') = \begin{cases} 1, & |F_{k'}^{(m)}(c_i) - e_i| \in [0, 2^8) \\ 0, & |F_{k'}^{(m)}(c_i) - e_i| \notin [0, 2^8) \end{cases}$$

则  $T_{k'}^{(m)} = \sum_{i=1}^N \xi_i(k')$

的期望值为  $p_m N$ ; 当  $k' \neq k^{(m)}$  时, 由于  $|k - k'|$  一般较大, 故差值  $|F_{k'}^{(m)}(c_i) - F_k^{(m)}(c_i)|$  的分布一般不再偏小. 若将它按均匀分布近似, 则它落入  $[0, 2^8)$  的概率近似为  $2^{-8}$ , 故此时  $T_{k'} = \sum_{i=1}^N \xi_i(k')$  的期望值近似为  $2^{-8}N$ . 由于  $p_m N$  和  $2^{-8}N$  有很大的差异, 例如, 当  $m = 32, N = 10000$  时,

$$p_m N = 0.206 \times 10000 = 2060, \quad 2^{-8}N = 10000 \div 256 = 39$$

因此, 统计量  $T_{k'}$  最大的那个  $k'$  是正确密钥的概率最大, 据此就可将  $T_{k'}$  达到最大的那个  $k'$  判定为正确密钥. 这样就可利用统计方法给出求解  $k^{(m)}$  的算法, 这就是下面的基本算法:

Step 1 利用  $N + 1$  个已知明文, 计算出  $F_k^{(m)}(x)$  的  $N$  个输入  $c_1, c_2, \dots, c_N$  及对应的输出  $e_1, e_2, \dots, e_N$ ;

Step 2 对  $k^{(m)}$  的每个可能值  $k'$ , 对  $\forall i: 1 \leq i \leq N$ , 求出  $F_{k'}^{(m)}(c_i)$ ,

并构造统计量  $T_{k'} = \sum_{i=1}^N \xi_i(k')$ , 其中

$$\xi_i(k') = \begin{cases} 1, & |F_{k'}^{(m)}(c_i) - e_i| \in [0, 2^8) \\ 0, & |F_{k'}^{(m)}(c_i) - e_i| \notin [0, 2^8) \end{cases}$$

并将使  $T_{k'}$  达到最大的  $k'$  判断为  $k^{(m)}$  的第一候选值, 将使  $T_{k'}$  达到第  $t$  大的  $k'$  判断为  $k^{(m)}$  的第  $t$  候选值.

备注 如果基本算法只要求输出  $r$  个候选值, 则在基本算法中可采取下述方法选出这  $r$  个候选值:

预设  $H_0 = N + 1, H_1 = H_2 = \dots = H_r = 0, k_1 = k_2 = \dots = k_r = 0$ . 对  $k^{(m)}$  的每个可能值  $k'$ , 求出  $T_{k'}$ . 如果  $T_{k'} \leq H_r$ , 则检测下个可能的  $k'$ ; 否则, 如果  $H_1 < T_{k'} \leq H_r$ , 则在执行

$(H_1, H_2, \dots, H_r) \leftarrow (H_1, \dots, H_{r-1}, T_{k'}, H_r, H_{r+1}, \dots, H_{r-1})$

$(k_1, k_2, \dots, k_r) \leftarrow (k_1, \dots, k_{r-1}, k', k_i, k_{i+1}, \dots, k_{r-1})$

后检测下个可能的  $k'$ .

显然, 基本算法结束时保留的  $k_i$  就是第  $i$  候选值. 当  $r$  不大时, 由于与  $r$  个  $H_i$  比较及对当前的  $r$  个候选值重新设置的计算量远小于  $F_{k'}^{(m)}(c_i)$  的计算量, 因而适当增大  $r$  并不明显增加基本算法的计算复杂性.

前面我们通过比较  $T_{k^{(m)}}$  和  $T_{k'}$  的期望值的差异, 说明了基本算法的有效性. 显然,  $N$  越大, 这种差异越大, 因而基本算法越有效. 下面给出基本算法成功率的计算公式.

定理 1 设基本算法中的试验密钥为  $k'$ , 记  $p_{k'} = p(\xi_i(k') = 1)$ , 且诸  $\xi_i(k')$  相互独立, 则基本算法输出的第一候选值是  $k^{(m)}$  的概率为

$$\sum_{i=1}^N \{ C_N^i p_{k'}^i (1 - p_{k'}^{(m)})^{N-i} [ \prod_{k' \in \{0, 1\}^m \setminus k^{(m)}} \sum_{j=0}^{i-1} C_N^j p_{k'}^j (1 - p_{k'}^{(m)})^{N-j} ] \}$$

证明  $T_{k'} = i$  等价于在  $\xi_1(k'), \dots, \xi_N(k')$  中, 有  $i$  个为 1,  $N - i$  个为 0, 即  $T_{k'}$  服从二项分布, 故有

$$p(T_{k'} = i) = C_N^i p_{k'}^i (1 - p_{k'}^{(m)})^{N-i}$$

又因基本算法求出的第一候选值是  $k^{(m)}$  等价于  $T_{k^{(m)}} > \max_{k' \in \{0, 1\}^m \setminus k^{(m)}} T_{k'}$ , 且诸  $T_{k'}$  相互独立, 故由基本算法求出的第一候选值是  $k^{(m)}$  的概率为

$$\begin{aligned} p(T_{k^{(m)}} > \max_{k' \in \{0, 1\}^m \setminus k^{(m)}} T_{k'}) &= \sum_{i=0}^N p(T_{k^{(m)}} = i \text{ 且 } \forall k' \in \{0, 1\}^m \setminus k^{(m)}, \text{ 有 } T_{k'} < i) \\ &= \sum_{i=0}^N p(T_{k^{(m)}} = i) \prod_{k' \in \{0, 1\}^m \setminus k^{(m)}} p(T_{k'} < i) \\ &= \sum_{i=1}^N \{ C_N^i p_{k^{(m)}}^i (1 - p_{k^{(m)}}^{(m)})^{N-i} [ \prod_{k' \in \{0, 1\}^m \setminus k^{(m)}} \sum_{j=0}^{i-1} C_N^j p_{k'}^j (1 - p_{k'}^{(m)})^{N-j} ] \} \end{aligned}$$

基本算法只是给出了求解  $k^{(m)}$  的方法, 要完整求出密钥  $k$ , 还需将密钥分成若干个密钥块, 并结合分割攻击方法逐块求出各密钥块, 从而达到完整求出密钥  $k$  的目的. 换句话说, 我们可根据表 2 选择适当的  $m_1, m_2, \dots, m_{t-1}$ , 使下式成立  $0 = m_0 < m_1 < m_2 < \dots < m_t = 64$

然后再通过对  $k^{(m_1)}$  的穷举, 利用基本算法找出正确的  $k^{(m_1)}$ , 接着依次对  $i = 2, 3, \dots, t$  在  $k^{(m_{i-1})}$  已知的条件下, 通过对  $k^{(m_i)}$  的其他未知比特的穷举, 利用基本算法找出正确的  $k^{(m_i)}$ , 从而实现了对密钥  $k$  的分割攻击.

分割点  $m_1, m_2, \dots, m_{t-1}$  的选择结果对攻击算法的成功率和计算复杂性都有很大的影响. 它们的选择首先应能将正确的  $k^{(m_i)}$  与其他  $2^{m_i - m_{i-1}} - 1$  个错误密钥区分开, 即要求诸  $p_{m_i}$  与  $2^{-8}$  有一定的差别, 这就要求  $m_i - m_{i-1}$  不能太小; 其次, 由于当已知的明文量为  $N$  时, 在  $k^{(m_{i-1})}$  已知条件下需要穷举的  $k^{(m_i)}$  的数量为  $2^{m_i - m_{i-1}}N$ , 所以  $m_i - m_{i-1}$  也不能过大. 这就是  $m_1, m_2, \dots, m_{t-1}$  的基本选择原则.

下面给出对具有 64 比特密钥的 ZLY 自同步混沌密码的两个分割攻击算法, 其中算法 1 是单候选密钥攻击方案, 即在利用基本算法求解诸  $k^{(m_i)}$  时, 都只保留一个候选密钥; 算法 2 是多候选密钥攻击方案, 即在利用基本算法

求解诸  $k^{(m_i)}$  时, 都保留多个候选密钥。

在具体的攻击算法中, 我们将密钥分割成 4 块, 并选择  $m_1 = 20, m_2 = 32, m_3 = 48, m_4 = 64$  同时假设  $N$  为 1 万, 此时  $n = 65, k^{(64)}$  就是正确密钥。具体算法如下:

算法 1 (单候选密钥攻击方案):

Step 1 取  $N = 10000, m_1 = 20, m_2 = 32, m_3 = 48, m_4 = 64, m_0 = 0$  初始化  $i = 1$

Step 2 利用  $N + 1$  个已知明文, 计算出  $F_k^n(x)$  的  $N$  个输入  $c_1, c_2, \dots, c_N$  及对应的输出  $e_1, e_2, \dots, e_N$ ;

Step 3 攻击  $k^{(m_i)}$ 。当  $i = 1$  时, 穷举  $k^{(m_1)}$  的每个可能值, 当  $2 \leq i \leq 4$  将  $k^{(m_i)}$  的最高  $m_{i-1}$  比特块设置为  $k^{(m_{i-1})}$  的候选值, 并穷举  $k^{(m_i)}$  的低  $m_i - m_{i-1}$  比特块的每个可能值。设  $k'$  是如此得到的  $k^{(m_i)}$  的一个试验密钥。当  $1 \leq i \leq 3$  时, 对  $\forall j, 1 \leq j \leq N$ , 求出  $F_{k'}^{n_j}(c_j)$ , 并构造出统计量  $T_{k'} = \sum_{j=1}^N \xi_j(k')$ , 其中

$$\xi_j(k') = \begin{cases} 1 & |F_{k'}^{n_j}(c_j) - e_j| < 2^{-8} \\ 0 & |F_{k'}^{n_j}(c_j) - e_j| \geq 2^{-8} \end{cases}$$

并将使  $T_{k'}$  达到最大的  $k'$  判断为  $k^{(m_i)}$  的候选值, 再将  $i$  增 1; 当  $i = 4$  时执行 Step 4, 否则返回执行 Step 3。

Step 4 若以  $k'$  为密钥对明文序列的加密结果与正确的密文序列一致, 则输出  $k^{(m_4)}$ , 若不一致, 则在所有可能的  $k^{(m_4)}$  都检验完毕时, 报告算法 1 失败。在未全部检验完毕时, 返回 Step 3 检验下个可能的  $k^{(m_4)}$ 。

定理 2 算法 1 的平均计算复杂性为计算自同步映射  $F_k^n(x)$  共  $1.3 \times 2^{33}$  次。

证明 算法 1 的平均计算复杂性是攻击  $k^{(m_1)}, k^{(m_2)}, k^{(m_3)}, k^{(m_4)}$  的平均计算复杂性之和。现将计算复杂性考虑为计算自同步映射  $F_k^n(x)$  的次数。则当  $1 \leq i \leq 3$  时, 攻击  $k^{(m_i)}$  时的最大计算复杂性均为  $2^{m_i - m_{i-1}} N$ , 因而攻击  $k^{(m_1)}, k^{(m_2)}, k^{(m_3)}$  的最大计算复杂性分别为  $1.22 \times 2^{23}, 1.22 \times 2^{25}, 1.22 \times 2^{29}$ 。在攻击  $k^{(m_4)}$  时, 假定  $k^{(m_i)}$  的分布是均匀的, 则平均搜索  $2^{m_4 - m_3 - 1}$  个试验密钥后找到正确密钥。对于一个试验密钥, 在比较其产生的  $t$  个乱数后才能将其判断为错误密钥的概率近似为  $2^{-t}$ , 因而  $t$  的期望  $E(t) = 2$ , 故攻击  $k^{(m_4)}$  计算复杂性平均为计算自同步映射  $2^{m_4 - m_3 - 1} E(t) = 2^{m_4 - m_3} = 2^{16}$ , 因此算法 2 的平均计算复杂性为计算自同步映射  $1.22 \times 2^{33} + 1.22 \times 2^{25} + 1.22 \times 2^{29} + 2^{16} \approx 1.3 \times 2^{33}$  次。

由定理 2 可知, 算法 1 的计算量很小, 且在 PC 机上可以完成。我们利用 40 余台主频为 2.5GHz 的 Pentium 4 PC 机对算法 1 做了 100 例攻击实验, 每例实验的平均运行时间为 17 小时 16 分, 求出的 64 比特密钥的正确率为 0.76。

由于增大  $N$  将增大诸  $k^{(m_i)}$  的统计量  $T_{k^{(m_i)}}$  与错误密钥  $k'$  的统计量  $T_{k'}$  之间的差异, 故可提高算法的成功率。实验表明, 当将  $N$  增加到 20000 时, 成功率可上升到 0.85。

在利用基本算法攻击  $k^{(m)}$  时, 如果试验密钥  $k'$  的高  $m - \varepsilon$  位与正确密钥的高  $m - \varepsilon$  位相同, 则  $|k - k'| < 2^{m - \varepsilon}$ , 故当  $\varepsilon$  很小时,  $|k - k'|$  也很小, 因而输出差  $|F_{k'}^n(x) - F_k^n(x)|$  也会偏小。这说明与正确  $k^{(m)}$  仅在低几位不同的那些假设值对应的输出差落入小区间  $[0, 2^{-8}]$  的

概率也很大。表 2 也能看出这个规律。尽管该概率小于  $|F_{k^{(m)}}^n(x) - F_k^n(x)|$  落入小区间  $[0, 2^{-8}]$  的概率, 但却能以不小的概率使某个错误假设值对应的统计量达到最大。这也是混沌映射的信息泄漏规律的一个特点。

针对这个问题, 我们可在攻击诸  $k^{(m_i)}$  时多保留几个候选值, 从而提高这些候选值包含  $k^{(m_i)}$  的概率。同时, 在对候选值进行检测时, 先检测正确率大的, 再检测正确率小的, 从而降低算法的平均计算复杂性。这就是下面给出的多候选密钥方案。在该攻击方案中, 诸  $k^{(m_i)}$  的候选量  $M_i$  均设置为 3 其他参数不变。

算法 2 (多候选密钥攻击方案):

Step 1 取  $N = 10000, m_1 = 20, m_2 = 32, m_3 = 48, m_4 = 64, m_0 = 0$  令  $M_1 = M_2 = M_3 = 3$  令  $t_1, t_2, t_3$  的初始值均为 1 再初始化  $i = 1$

Step 2 利用  $N + 1$  个已知明文, 计算出  $F_k^n(x)$  的  $N$  个输入  $c_1, c_2, \dots, c_N$  及对应的输出  $e_1, e_2, \dots, e_N$ ;

Step 3 攻击  $k^{(m_i)}$ 。当  $i = 1$  时, 穷举  $k^{(m_1)}$  的每个可能值, 当  $2 \leq i \leq 4$  将  $k^{(m_i)}$  的最高  $m_{i-1}$  比特块设置为  $k^{(m_{i-1})}$  的第  $t_{i-1}$  候选密钥, 并穷举  $k^{(m_i)}$  的低  $m_i - m_{i-1}$  比特块的每个可能值。设  $k'$  是如此得到的  $k^{(m_i)}$  的一个试验密钥。当  $1 \leq i \leq 3$  时, 对  $\forall j, 1 \leq j \leq N$ , 求出  $F_{k'}^{n_j}(c_j)$ , 并构造出统计量  $T_{k'} = \sum_{j=1}^N \xi_j(k')$ , 其中

$$\xi_j(k') = \begin{cases} 1, & |F_{k'}^{n_j}(c_j) - e_j| < 2^{-8} \\ 0, & |F_{k'}^{n_j}(c_j) - e_j| \geq 2^{-8} \end{cases}$$

若试验密钥  $k_1, \dots, k_{M_i}$  对任意的  $k' \in \{k_1, \dots, k_{M_i}\}$ , 都有  $T_{k_1} \geq T_{k_2} \geq \dots \geq T_{M_i} \geq T_{k'}$  成立, 则将  $k_1, \dots, k_{M_i}$  分别记为  $k^{(m_i)}$  的第 1 第 2, ..., 第  $M_i$  候选密钥, 并将  $i$  增 1; 若  $i = 4$  则执行 Step 4, 否则返回执行 Step 3。

Step 4 若以  $k'$  为密钥对明文序列的加密结果与正确的密文序列全部一致, 则输出  $k^{(m_4)}$ , 若不全部一致, 则当  $k^{(m_4)}$  的所有可能值来检验完毕时, 检验下一个可能值, 否则将  $t_3$  增 1 并在  $t_3 \leq M_3$  时返回执行 Step 3。如果  $t_3 = M_3 + 1$  令  $t_3 = 1, i = 3$  同时将  $t_2$  增 1 并在  $t_2 \leq M_2$  时返回 Step 3。如果  $t_2 = M_2 + 1$ , 令  $t_2 = t_3 = 1, i = 2$  同时将  $t_1$  增 1 并在  $t_1 \leq M_1$  时返回 Step 3。如果  $t_1 = M_1 + 1$  则宣告算法 2 失败, 算法结束。

备注 Step 4 是对  $(k^{(m_1)}, k^{(m_2)}, k^{(m_3)})$  的 3 个候选值检测的过程。由于  $t_1, t_2, t_3$  的初始值均为 1, 故算法 2 首先检测是  $k^{(m_1)}, k^{(m_2)}$  和  $k^{(m_3)}$  的第一候选值。只有在它们都不正确的条件下, 才检测  $k^{(m_3)}$  的其他候选值, 之后对  $k^{(m_2)}$  的其他各候选值, 生成  $k^{(m_3)}$  对应的 3 个候选值, 并通过对  $k^{(m_3)}$  的候选值的检测, 完成对  $k^{(m_2)}$  的各候选值的检测; 最后对  $k^{(m_1)}$  的其他各候选值, 生成  $k^{(m_2)}$  相应的 3 个候选值及与之对应的  $k^{(m_3)}$  的 9 个候选值, 并通过对  $k^{(m_3)}$  的候选值的检测, 完成对  $k^{(m_1)}$  的各候选值的检测。在这个过程中, 一旦找到了正确值, 就终止了算法。因此, 当各第一候选值都是正确值时, 算法 2 的执行过程与算法 1 一致, 故算法 2 的成功率一定大于算法 1 的成功率, 且算法 2 可在算法 1 失败条件下从其他最可能的候选密钥中找出正确密钥。

定理 3 算法 2 的平均计算复杂性的上界是计算  $1.9 \times 2^{33}$  次自同步映射  $F_k^n(x)$ , 平均计算复杂性约为计算  $1.3 \times 2^{33} + 1.22 \times 2^{29}$  次自同步映射  $F_k^n(x)$ 。其中  $E$  是  $(k^{(m_1)}, k^{(m_3)})$  在算法 2 中准备被检测时已检测过的  $(k^{(m_1)}, k^{(m_2)})$  的个数的期望。

证明 由定理 2 的证明知, 当  $(k^{(m_1)}, k^{(m_2)}, k^{(m_3)})$  的

候选值是正确值时,完成该候选值检测的计算复杂性平均为  $2^{16}$ ,当该候选值错误时,完成该候选值检测的计算复杂性平均为  $2^{17}$ .当  $(k^{(m_1)}, k^{(m_2)}, k^{(m_3)})$  的第  $3^2a+3b+c+1$  候选值  $(0 \leq a, b, c \leq 2)$  正确时,检测到该候选密钥时,已经完成了  $3^2a+3b+c$  个错误的候选值的检测,完成这些候选值的检测的计算量平均为  $(9a+3b+c)2^{17}$ .在产生这些候选值时,共执行了 1 次产生  $k^{(m_1)}$  的工作,由于此时正在对第  $a+1$  个  $k^{(m_1)}$  检测,因而共执行了  $a+1$  次产生  $k^{(m_1)}$  的工作,由于已检测了  $3a+b$  个  $k^{(m_2)}$ ,因而共执行了  $3a+b+1$  次产生  $k^{(m_2)}$  的工作,故在  $(k^{(m_1)}, k^{(m_2)}, k^{(m_3)})$  的当前候选值正确时,算法 2 的计算量平均为

$$2^{m_1-m_0}N + 2^{m_2-m_1}(a+1)N + 2^{m_3-m_2}N(3a+b+1) + (9a+3b+c)2^{17} + 2^{16} \approx 1.3 \times 2^{33} + 1.22 \times 2^{25}a + 1.22 \times 2^{29}(3a+b) + (9a+3b+c)2^{17} \approx 1.3 \times 2^{33} + 1.22 \times 2^{29}(3a+b)$$

故算法 2 在  $a=b=c=2$  时平均计算复杂性达到最大值  $1.3 \times 2^{33} + 1.22 \times 2^{29} \times 8 \approx 1.9 \times 2^{33}$ .再记  $q_i$  是算法 2 第  $i$

次检测的  $(k^{(m_1)}, k^{(m_2)})$  的候选值是正确值的概率,则

$$E+1 = \sum_{i=1}^9 q_i i \text{ 且算法 2 的平均计算复杂性为 } \sum_{i=1}^9 q_i [1.3 \times 2^{33} + 1.22 \times 2^{29}(i-1)] \approx 1.3 \times 2^{33} + 1.22 \times 2^{29} \left[ \sum_{i=1}^9 q_i i \right] = 1.3 \times 2^{33} + 1.22 \times 2^{29} E$$

由定理 3 可知,算法 2 的计算复杂性最多是算法 1 的 1.5 倍,因而计算复杂性不大.

我们利用 40 余台主频为 2 GHz 的 Pentium 4 PC 机对算法 2 做了 100 例攻击实验,每例实验的平均运行时间为 25 小时 22 分,成功率为 0.86.实验表明其平均计算复杂性是单候选密钥攻击方案的 1.45 倍.这与定理 3 的结论是吻合的.在实验中,我们遇到了多例正确密钥的排序非常偏后的情形,从而增加了算法平均的运行时间.显然,其平均运行时间和成功率都优于  $N=2$  万时的单候选密钥攻击方案,因而证实了多候选密钥攻击方案的性能优于单候选密钥攻击方案.

由于当  $k'$  与  $k$  仅最低位不同时,  $k'$  与  $k$  的差异最小,因此,只要自同步映射  $f_k$  能使  $|f_k(x) - f_k(x)|$  的分布与均匀分布不可区分,就能对抗本文提出的分割攻击方法.这可以作为检验自同步混沌密码能否对抗本文提出的分割攻击方法的检测指标.

#### 4 结论

本文分析了自同步混沌密码的信息泄漏规律,并据此提出了对自同步混沌密码的分割攻击方法.作为例子,本文实现了对具有 64 比特密钥的 ZLY 自同步混沌密码算法的分割攻击.自同步混沌密码的信息泄漏的特点与迭代型混沌密码的信息泄漏特点<sup>[5]</sup>有所不同.随着迭代次数的增加,迭代型混沌密码初态的低位比特对乱数的影响将越来越大,这就导致分割攻击方法<sup>[5]</sup>在攻击初态(密钥的一部分)的高位比特时,只能利用前面有限个乱数信号提供

的信息.自同步映射一般是混沌映射的多次复合,结构比较复杂,信息泄漏相对较少,但因自同步映射  $f_k$  的每个输入——输出对都具有相同的信息泄漏,而且可利用所有时刻的已知明文造出  $f_k$  的大量输入——输出对,从而可利用统计分析方法,积少成多,还原出密钥.本文的结论还说明,只要自同步映射的具有一定联系的密钥产生的输出不是独立的,就有将其彻底攻破的可能.因此,确保自同步映射的不同密钥产生的输出的相互独立性,是保证自同步混沌密码安全的一个必要条件.

#### 参考文献:

- [1] Hong Zhou, Xie-Ting Lin, Jun Yu. Secure communication via one-dimensional chaotic inverse systems[J]. In Proc IEEE Int Symposium Circuits and Systems 97, 1997, 2: 9-12.
- [2] 李树钧等.一类混沌流密码的分析[J].电子与信息学报, 2003, 25(4): 473-478.  
Shujun Li et al. Cryptanalysis of a class of chaotic stream ciphers[J]. Journal of Electronics & Information Technology, 2003, 25(4): 473-478 (in Chinese).
- [3] 周红, 俞军, 凌雯.混沌前馈型流密码的设计[J].电子学报, 1998, 26(1): 98-101.  
Hong Zhou, Jun Yu, Xie-Ting Lin. Theoretical design of chaotic feed forward stream cipher[J]. Acta Electronica Sinica, 1998, 26(1): 98-101 (in Chinese).
- [4] 金晨辉.一个基于混沌的分组密码算法的分析[J].中国工程科学, 2001, 3(6): 75-80.  
Chenhui Jin. The analysis of a block cipher algorithm based on chaos[J]. China Engineering Science, 2001, 3(6): 75-80 (in Chinese).
- [5] 金晨辉, 高海英.对两个基于混沌的序列密码算法的分析[J].电子学报, 2004, 34(7): 1066-1070.  
Chenhui Jin, Haiying Gao. Analysis of two stream ciphers based on chaos[J]. Acta Electronica Sinica, 2004, 34(7): 1066-1070 (in Chinese).
- [6] 李树钧.数字化混沌密码的分析与设计, 西安交通大学博士论文, 2003.  
Shujun Li. Analyses and New Designs of Digital Chaotic Ciphers[D]. PhD thesis, School of Electronic and Information Engineering, Xi'an Jiaotong University, Xi'an, China, 2003.
- [7] Hong Zhou, Xie-Ting Lin. Problems with the chaotic inverse system encryption approach. IEEE Trans CAS-I, 1997, 44(3): 268-271.

#### 作者简介:

金晨辉 男, 1965 年出生于河南扶沟县, 解放军信息工程大学电子技术学院教授, 博士研究生导师, 主要研究方向是密码学和信息安全. E-mail: jinchenhui@126.com

杨 阳 女, 1980 年出生于辽宁锦州, 解放军信息工程大学电子技术学院助教, 主要研究方向是密码学.