SERPENT 和 SAFER 密码算法的能量攻击

吴文玲,蒙 杨,冯登国,卿斯汉
(中国科学院软件研究所信息安全国家重点实验室,北京 100080;中国科学院信息安全技术工程研究中心,北京 100080)

摘 要: SERPENT 和 SAFER 是 AES 的两个候选算法,本文使用能量攻击方法对它们进行了深入分析,结果表明:对于 256、192 和 128 比特密钥的 SERPENT 算法,能量攻击平均需分别进行 2^{159}、2^{119}和 2^{79}次试验. 虽然所需的试验次数实际没法达到,但是此攻击方法大大地降低了 SERPENT 的密钥规模,并且发现对于能量攻击,SERPENT 有许多弱密钥. 经过深入分析和穷尽搜索可知:能量攻击可以获取 SAFER 的种子密钥. 文中还给出了两种抵抗能量攻击的 SERPENT 的改进密钥方案以及设计密钥方案时需注意的问题.

关键词: 密钥; 算法; 能量攻击
中图分类号: TP391 **文献标识码**: A **文章编号**: 0372-2112 (2001) 01-0090-03

Power Attack of SERPENT and SAFER Cipher Algorithm

WU Wen-ling, MENG yang, FENG Deng-guo, QING Si-han
(1. *State Key Laboratory of Information Security, Institute of Software, Chinese Academy of Sciences, Beijing* 100080, *China*;
2. *Engineering Research Center for Information Security Technology, Chinese Academy of Sciences, Beijing* 100080, *China*)

Abstract: SERPENT and SAFER are AES candidates, which are analyzed by power attack in this paper. It is shown that power attack needs 2^{159}、2^{119} and 2^{79} trials for 256、192 and 128 bits key- SERPENT respectively. Although the number of trials is too big to realize, it reduces greatly the size of key. SERPENT have many weak keys for power attack. By analyzing and computing, it is received that SAFER is broken to power attack. Finally, some suggestions and two improving key scheduling of SERPENT are given.

Key words: key; algorithm; power attack

1 引言

1998 年,Paul Kocher[1]针对密码算法的 smart 卡实现,提出了密码算法的能量攻击(power attack)方法,这种攻击方法在实际应用中易于实现而且非常有效. 我们知道,在密码的运算过程中,smart 卡所消耗的能量与所执行的指令和所处理的数据有关,例如,乘法运算比加法运算消耗的能量多,写"1"比写"0"消耗的能量多. 能量攻击方法正是利用了这一基本思想. Biham 和 Shamir 在文献[2]中给出了能量攻击的一种变体,此种变体的攻击者不需要知道加密算法的输入、输出以及加密算法在 smart 卡中的实现细节. 为了简单起见,假设 smart 卡中的协议按相同的顺序执行子协议,且执行每个子协议所需的时钟一样. 因此,可以把协议的不同次执行的能量消耗图排成列,然后比较单个指令所消耗的能量.

攻击可以分三步:第一步,攻击者寻找能量消耗图上与密钥编排算法有关的部分. 他可以通过下面两步完成此任务.

(1) 用同一个 smart 卡对不同的数据实验多次,然后在每个时钟比较所得的多个能量消耗图. 如果在某些时钟,能量消耗图的变化比较大,则放弃这些时钟,因为这些时钟和所处理的数据有关. 保留下的时钟所代表的运算和数据无关.

(2) 对若干 smart 卡重复 (1). 假定这些卡有不同的密钥,寻找它们和数据无关的共同部分(时钟). 对于这些时钟,如果不同卡的能量消耗图的变化比较小,则放弃. 保留下的时钟就是攻击者所需的能量消耗图上与密钥编排算法有关的部分.

第二步,攻击者获取轮子密钥的每个字节的汉明重量. 迭代分组密码在 8-比特 msart 卡中的软件实现通常是在每一轮计算子密钥,而不是提前把所有子密钥计算出来,这是因为 smart 卡的储存能力小的缘故. 子密钥被计算出来后,以字节储存在 RAM 中,在储存的过程中,读/写所消耗的能量和字节中"1"的个数有关;因此,可以推测每个子密钥字节的汉明重量. 详细的操作可参看文献[2].

第三步,攻击者深入讨论密钥方案的每一步,利用"轮子密钥每个字节的汉明重量"这一信息,析取种子密钥或与种子密钥有关的信息. 文献[2]认为第一步和第二步在实际中易于实现. 因此,本文在攻击密码体制时,总假定已知轮子密钥的每个字节的汉明重量.

收稿日期:1999-08-23;修回日期:2000-04-18
基金项目:973 项目基金(No. G1999035802);国家自然科学基金跨学科重点项目基金(No. 19931010)

2 SERPENT的密钥方案

首先,把种子密钥 K 填充为 256 比特,并表示为 8 个 32 比特的字 $w_{-8}, \cdots, w_{-1}$. 其次,利用下列递归式:$w_i = (w_{i-8} \oplus w_{i-5} \oplus w_{i-3} \oplus w_{i-1} \oplus \phi \oplus i) \lll 11$ 构作 132 个 32 比特的字 $w_0, \cdots, w_{131}$. 其中 ϕ 是黄金分割率 $(\sqrt{5}+1)/2$ 的小数部分,或用十六进制表示 0x9e3779b9.

最后,利用 S_j 盒构作子密钥:$K_0 = S_3(w_0, w_1, w_2, w_3)$, $K_1 = S_2(w_4, w_5, w_6, w_7)$, ……, $K_i = S_j(w_{4i}, w_{4i+1}, w_{4i+2}, w_{4i+3})$, $3 \equiv (i+j) \bmod 8$, ……, $K_{32} = S_3(w_{128}, w_{129}, w_{130}, w_{131})$

其中 S_j 是由 32 个盒子 s_j (s_j 参见文献[3])并置而成.

3 SERPENT的密钥方案的攻击

令 $X, Y \in GF(2)^4$, $S_j(X) = Y$, $S_j: GF(2)^4 \to GF(2)^4$ 是置换,已知 $W_H(Y)$,则满足 $W_H(S_j(X)) = W_H(Y)$ 的 X 的个数 $N(X)$ 如下表 1 所示.

表 1

$W_H(Y)$	0	1	2	3	4
$N(X)$	1	4	6	4	1

因此,对于 (X_1, X_2), $(Y_1, Y_2) \in GF(2)^4 \times GF(2)^4$ 已知 $W_H(Y_1, Y_2)$,则满足 $W_H(S_j(X_1, X_2)) = W_H(Y_1, Y_2)$ 的 (X_1, X_2) 的个数 $N(X_1, X_2)$ 如表 2 所示.

表 2

$W_H(Y_1, Y_2)$	0	1	2	3	4	5	6	7	8
$N(X_1, X_2)$	1	8	28	56	70	56	28	8	1

在 SERPENT 的密钥方案中,

$K_i = (k_{4i}, k_{4i+1}, k_{4i+2}, k_{4i+3}) = S_j(w_{4i}, w_{4i+1}, w_{4i+2}, w_{4i+3})$

即 $k_{4i+t} = S_j(w_{4i+t})\ 0 \le t \le 3$,

$$k_{4i+t} = (k^1_{4i+t}, k^2_{4i+t}, k^3_{4i+t}, k^4_{4i+t}) = S_j(w^1_{4i+t}, w^2_{4i+t}, w^3_{4i+t}, w^4_{4i+t}) \begin{cases} k^1_{4i+t} = S_j(w^1_{4i+t}) \\ k^2_{4i+t} = S_j(w^2_{4i+t}) \\ k^3_{4i+t} = S_j(w^3_{4i+t}) \\ k^4_{4i+t} = S_j(w^4_{4i+t}) \end{cases}$$

利用能量攻击的第一步,检测出 $W_H(k^1_{4i+t})$、$W_H(k^2_{4i+t})$、$W_H(k^3_{4i+t})$ 和 W^4_{4i+t}. 满足 $W_H(S_j(w^1_{4i+t})) = W_H(k^1_{4i+t})$ 的 w^1_{4i+t} 的个数记为 $N(w^1_{4i+t})$;满足 $W_H(S_j(w^2_{4i+t})) = W_H(k^2_{4i+t})$ 的 w^2_{4i+t} 的个数记为 $N(w^2_{4i+t})$;满足 $W_H(S_j(w^3_{4i+t})) = W_H(k^3_{4i+t})$ 的 w^3_{4i+t} 的个数记为 $N(w^3_{4i+t})$;满足 $W_H(S_j(w^4_{4i+t})) = W_H(w^4_{4i+t})$ 的 w^4_{4i+t} 的个数记为 $N(w^4_{4i+t})$;则 w_{4i+t} 的候选值有 $N(w_{4i+t}) = N(w^1_{4i+t}) \times N(w^2_{4i+t}) \times N(w^3_{4i+t}) \times N(w^4_{4i+t})$ 个. 因此,在已知轮子密钥的每个字节的汉明重量的情况下,对于每个 $w_l (0 \le l \le 131)$ 都有 $N(w_l)$ 个候选值. 因为 $w_0, \cdots, w_{131}$ 是由 256 比特种子密钥 $w_{-8}, \cdots, w_{-1}$ 用递归方式生成,因此,只需知道 $w_0, \cdots, w_7$ 就可以计算 $w_8, w_9, \cdots, w_{131}$,反之,也可以推出 $w_{-8}, \cdots, w_{-1}$. 用下述步骤滤出 $(w_0, \cdots, w_7)$.

第一步,选定 $(w_0, \cdots, w_7)$ 的一候选值,共有 $N(w_0) \times \cdots \times N(w_7)$ 个. 第二步,用递归式计算 $w_l (8 \le l \le 131)$,并判别 $W_H(S_j(w_l))$ 是否和检测到的子密钥的每个字节的汉明重量相符,如果不相符,丢掉 $(w_0, \cdots, w_7)$ 的此候选值,返回第一步;如果相符,继续计算 w_{l+1}. 第三步,最后所剩的 $(w_0, \cdots, w_7)$ 应该是唯一的.

上述攻击的复杂性依赖于 $N(w_0) \times \cdots \times N(w_7)$. 对于 32 比特字 $w_i = (w^1_i, w^2_i, w^3_i, w^4_i)$,因为 $N(w^1_i)$、$N(w^2_i)$、$N(w^3_i)$ 及 $N(w^4_i)$ 的平均值都为 32,所以 $N(w_i)$ 的平均值为 2^{20}. 因此,对于 256 比特的密钥平均需进行 2^{159} 次试验;对于 192 比特的密钥平均需进行 2^{119} 次试验;对于 128 比特的密钥平均需进行 2^{79} 次试验. 虽然所需的试验次数实际没法达到,但是此攻击大大地降低了算法的安全性. 对于上述攻击,SERPENT 有弱密钥. 我们以 128 比特密钥长度为例来说明.

令种子密钥 $K = (w_{-8}, w_{-7}, w_{-6}, w_{-5})$,且设 $w_{-4} = w_{-3} = w_{-2} = w_{-1} = 0$;任取 w_{-8} 和 w_{-7},令 $w_{-5} = w_{-8} \oplus \phi$, $w_{-6} = \phi \oplus 2 \oplus (w_{-7} \oplus \phi \oplus 1) \lll 11$,

则这样的种子密钥 K 所构作的 $w_0 = w_2 = 0$,

$w_1 = (w_{-7} \oplus \phi \oplus 1) \lll 11$, $w_3 = (w_{-8} \oplus 3) \lll 11$;因此,攻击所需的试验次数平均为 2^{39},而这样的密钥有 2^{64} 个. 如果选取的 w_{-8} 和 w_{-7} 使得 w_1 和 w_3 满足:

$k_1 = (k^1_1, k^2_1, k^3_1, k^4_1) = S_3(w_1)$, $k_3 = (k^1_3, k^2_3, k^3_3, k^4_3) = S_3(w_3)$,

$$0 \le W_H(k^i_t) \le 2, 6 \le W_H(k^i_t) \le 8, 1 \le i \le 4 \quad t = 1, 3 \qquad (1)$$

每个 k^i_t 有 6 种取值;由表 2 知:当 $k^i_t = 0、8$ 时,对应的 w_t 的第 i 个字节有 2 种可能;当 $k^i_t = 1、7$ 时,对应的 w_t 的第 i 个字节有 2×8 种可能;当 $k^i_t = 2、6$ 时,对应的 w_t 的第 i 个字节有 2×28 种可能. 因此,满足条件 (1) 的 w_1 和 w_3 分别有 $(2 \times 37)^4$ 个,k_1 和 k_3 的字节汉明重量分布分别有 6^4 种可能,每一种可能值对应的 w_1 和 w_3 的候选值的个数平均为 $(37/3)^4$;因此,满足条件 (1) 的密钥的试验次数平均不超过 2^{29} 次,这样的密钥个数为 $(2 \times 37)^8 > 2^{48}$.

如果把条件 (1) 改为:

$$0 \le W_H(k^i_t) \le 1, \quad 7 \le W_H(k^i_t) \le 8 \qquad (2)$$

每个 k^i_t 有 4 种取值;由表 2 知:当 $k^i_t = 0、8$ 时,对应的 w_t 的第 i 个字节有 2 种可能;当 $k^i_t = 1、7$ 时,对应的 w_t 的第 i 个字节有 2×8 种可能. 因此,满足条件 (2) 的 w_1 和 w_3 分别有 $(2 \times 9)^4$ 个,k_1 和 k_2 的字节汉明重量分布分别有 4^4 种可能,每一种可能值对应的 w_1 和 w_3 的候选值的个数平均为 $(9/2)^4$;则满足条件 (2) 的密钥的试验次数平均为 $(9/2)^4 \times (9/2)^4$ 次,这样的密钥的个数为 $18^8 > 2^{33}$.

针对上述攻击,我们给出 Serpent 密钥方案的改进形式:首先,把种子密钥 K 填充为 256 比特,并表示为 8 个 32 比特的字 $w_{-8}, \cdots, w_{-1}$. 其次,利用下列递归式:$w_i = (w_{i-8} \oplus w_{i-5} \oplus w_{i-3} \oplus w_{i-1} \oplus \phi \oplus i) \lll 11$ 构作 264 个 32 比特的字 $w_0, \cdots, w_{131}, \cdots, w_{263}$. 其中 ϕ 是黄金分割率 $(\sqrt{5}+1)/2$ 的小数部分,或用十六进制表示为 0x9e3779b9. 再其次,利用 S 盒构做 K^*_i $(0 \le j \le 65)$:

$K^*_0 = S_3(w_0, w_1, w_2, w_3)$, ……, $K^*_i = S_j(w_{4i}, w_{4i+1}, w_{4i+2}, w_{4i+3})$, $3 \equiv (i+j) \bmod 8$, ……, $K^*_{65} = S_2(w_{260}, w_{261}, w_{262}, w_{263})$

最后,令子密钥

$$K_i = K_i^* \oplus K_{i+33}^*, (0 \le i \le 32).$$

因为从 $K_i(0 \le i \le 32)$ 的每个字节的汉明重量得不到 $K_i^*(0 \le i \le 65)$ 的每个字节的汉明重量,因此前面的攻击不能实行.但是此方案的运算量比原密钥方案的运算量大.

另一种改进方案是用种子密钥 $w_{-8}, \cdots, w_{-1}$ 利用下列递归式: $w_i = (w_{i-8} \oplus w_{i-5} \oplus w_{i-3} \oplus w_{i-1} \oplus \phi \oplus i) \ll 11$ 构作 264 个 32 比特的字 $w_0, \cdots, w_{131}, \cdots, w_{263}$. 其中 ϕ 是黄金分割率 $(\sqrt{5}+1)/2$ 的小数部分,或用十六进制表示为 0x9e3779b9. 其次,令 $v_i = w_i \oplus w_{i+132} (0 \le i \le 131)$. 最后,令子密钥

$$K_i = S_j(v_{4i}, v_{4i+1}, v_{4i+2}, v_{4i+3})$$
$$3 \equiv (i+j) \bmod 8, (0 \le j \le 32)$$

用上述攻击方法可以获得 $v_i(0 \le i \le 131)$,但是得不到 $w_0, \cdots, w_{131}, \cdots, w_{263}$,因此,此方案对能量攻击是安全的且它的计算量和原方案相比不是很大.

4 SAFER 的密钥方案

限于篇幅,我们仅讨论密钥长度为 128 比特的情况:

给定 32 个 128 比特的常量 $B_2, B_3, \cdots, B_{17}$; K 作为第一轮子密钥 K_1 并放入 17 个字节的寄存器的前 16 个,寄存器的最后一个位置放入 K 的 16 字节的模 2 加.然后寄存器的每个字节循环左移 3 比特,将 16 字节的常量 K_2 分别与寄存器的第 2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17 字节逐字节模 256 加,输出子密钥 K_2. 寄存器的每个字节再循环左移 3 比特,将 16 字节的常量 B_3 分别与寄存器的第 3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、1 字节逐字节模 256 加,输出子密钥 K_3. 如此下去,直到将 16 字节的常量 B_{17} 分别与寄存器的第 17、1、2、3、4、5、6、7、8、9、10、11、12、13、14、15 字节逐字节模 256 加,输出子密钥 K_{17}.

5 SAFER 的密钥方案的攻击

用 $K_i^j(1 \le i \le 17, 1 \le j \le 16)$ 表示子密钥 K_i 的第 j 个字节,用能量攻击的第一步和第二步获取每个 $W_H(K_i^j)$. 令 $K_1^j = (x_{j1}, x_{j2}, \cdots, x_{j8})$,对于任意给定的 j,可以构作以 $x_{j1}, x_{j2}, \cdots, x_{j8}$ 为未知变量的方程组.例如对于 $j=1$,有如下方程组:

$$\begin{cases} x_{11} + x_{12} + \cdots + x_{18} = W_H(K_1^1) = y_1 \\ |(x_{11}, \cdots, x_{18}) \ll 6| + {}_{256} B_3^{16} = W_H(K_3^{16}) = y_2 \\ |(x_{11}, \cdots, x_{18}) \ll 1| + {}_{256} B_4^{15} = W_H(K_4^{15}) = y_3 \\ |(x_{11}, \cdots, x_{18}) \ll 4| + {}_{256} B_5^{14} = W_H(K_5^{14}) = y_4 \\ |(x_{11}, \cdots, x_{18}) \ll 7| + {}_{256} B_6^{13} = W_H(K_6^{13}) = y_5 \\ |(x_{11}, \cdots, x_{18}) \ll 2| + {}_{256} B_7^{12} = W_H(K_7^{12}) = y_6 \\ |(x_{11}, \cdots, x_{18}) \ll 5| + {}_{256} B_8^{11} = W_H(K_8^{11}) = y_7 \\ |(x_{11}, \cdots, x_{18})| + {}_{256} B_9^{10} = W_H(K_9^{10}) = y_8 \\ |(x_{11}, \cdots, x_{18}) \ll 3| + {}_{256} B_{10}^{9} = W_H(K_{10}^{9}) = y_9 \\ |(x_{11}, \cdots, x_{18}) \ll 6| + {}_{256} B_{11}^{8} = W_H(K_{11}^{8}) = y_{10} \\ |(x_{11}, \cdots, x_{18}) \ll 1| + {}_{256} B_{12}^{7} = W_H(K_{12}^{7}) = y_{11} \\ |(x_{11}, \cdots, x_{18}) \ll 4| + {}_{256} B_{13}^{6} = W_H(K_{13}^{6}) = y_{12} \\ |(x_{11}, \cdots, x_{18}) \ll 7| + {}_{256} B_{14}^{5} = W_H(K_{14}^{5}) = y_{13} \\ |(x_{11}, \cdots, x_{18}) \ll 2| + {}_{256} B_{15}^{4} = W_H(K_{15}^{4}) = y_{14} \\ |(x_{11}, \cdots, x_{18}) \ll 5| + {}_{256} B_{16}^{3} = W_H(K_{16}^{3}) = y_{15} \\ |(x_{11}, \cdots, x_{18})| + {}_{256} B_{17}^{2} = W_H(K_{17}^{2}) = y_{16} \end{cases}$$

通过 2^8 次计算可知,任取 $X, X^* \in F_2^8, X \ne X^*$,由上方程组所得的两个 16 维向量 $(y_1, \cdots, y_{16})$ 不相等.因此,一旦给定 $(y_1, \cdots, y_{16})$ 的值,则可以唯一确定 $(x_{11}, \cdots, x_{18})$;即只要能获取子密钥的每个字节的汉明重量,则能唯一确定种子密钥的第一字节.类似地,对 $K_i(2 \le j \le 16)$ 分别构作相应的方程组并做计算,结果表明,只要能获取子密钥的每个字节的汉明重量,则能唯一确定种子密钥.

SAFER 的密钥方案中的常量 $B_i(2 \le i \le 33)$ 是为了"随机化"轮子密钥,但是它们使得方程组(3)在给定 $(y_1, \cdots, y_{16})$ 的值时,有唯一确定的解.这是能量攻击成功的原因;如果能找到一组 $B_i(2 \le i \le 33)$,使得方程组(3)在给定 $(y_1, \cdots, y_{16})$ 的值时,解不唯一,则攻击不成功.但是如果攻击者知道一些明密文对,则可以对候选的密钥进行检测,而候选的密钥个数小于 $C_8^{W_H(K_1^1)} \times C_8^{W_H(K_1^2)} \times \cdots \times C_8^{W_H(K_1^{16})}$.

6 结束语

针对能量攻击,我们认为密钥方案的设计应注意以下几点:(1)避免种子密钥直接用作子密钥;这是因为如果种子密钥直接用作子密钥,通过能量攻击的第一、二步就能获取种子密钥每个字节的汉明重量,从而大大地降低了算法的密钥规模.(2)避免使用没有非线性成分的密钥方案;例如 SAFER 和 DES 的密钥方案.因为没有非线性成分的密钥方案往往可以从子密钥字节的汉明重量获取种子密钥的信息.(3)充分混合使用线性运算(例如 $\oplus$ 和 +)和非线性置换;例如 Serpent 的密钥方案,上述攻击能成功的原因有二,一是知道预密钥 $w_0, \cdots, w_{131}$ 的连续 8 个就可恢复种子密钥;二是从预密钥到子密钥仅经过一层非线性运算,没有用线性运算做进一步的重量"隐蔽".

参考文献:

[1] PC Kocher. Differential Power Analysis [DB/OL]. http://www.Cryptography.com/dpa/

[2] Eli Biham, Adi Shamir. Power Analysis of the Key Scheduling of the AES Candidates [DB/OL]. http://www.cs.technion.ac.il/~biham/

[3] SERPENT, SAFER [DB/OL]. http://www.nist.gov/aes.

作者简介:

吴文玲 1966 年出生,1987 年和 1990 年分别在西北大学获学士和硕士学位,1997 年在西安电子科技大学获博士学位,1997 年至 1999 年在中国科学院软件研究所做博士后,现在中国科学院信息安全技术工程研究中心从事信息安全理论和技术研究工作.

蒙 杨 1972 年出生,1993 年和 1996 年分别在兰州大学获学士和硕士学位,现在中国科学院信息安全技术工程研究中心攻读博士学位.