密码学实验综合实验二

18374480-黄翔

2021 年 6 月 18 日

1 实验目的

1. 实现 SM4 的故障注入攻击

2 实验环境

1. python 3.9
2. pwntools

3 实验内容

3.1 实验原理

DFA 差分故障攻击是一种侧信道攻击。这类攻击将故障注入到密码算法的某一轮中，并根据正确-错误的密文对获得差分值，从而进行差分攻击。1997 年。Biham 和 Shamir 首次将故障攻击应用于对称密码体制，首次提出“差分故障攻击”的概念，并成功攻击了 DES 算法，此后研究人员提出了各种不同的故障攻击方法，成功攻击了多种密码体制，比如 ECC 公钥体制，AES 算法，3DES 算法以及 RC4 算法等。

对 SM4 的整体攻击方法思想如下：

1 选择明文，获得该明文对应的正确密文

2 从最后一轮开始，对加密过程进行单字节故障诱导，获得错误密文，与正确密文差分分析，恢复出该轮子密钥部分字节信息，重复这一过程，直至完全恢复该轮子密钥

3 对倒数第二轮进行单字节故障注入，获得错误密文，利用步骤 2 中恢复的轮密钥，获得目前该轮输出中间值，结合差分分析，恢复出倒数第二轮密钥部分字节，重复至恢复该轮子密钥全部字节

4 同理恢复出倒数第三、四轮的子密钥

5 由最后四轮子密钥，经过逆密钥扩展恢复出加密密钥的值

SM4 一轮过程如下

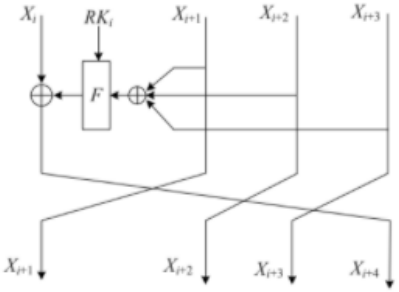


Fig. 1. The overall structure of SMS4

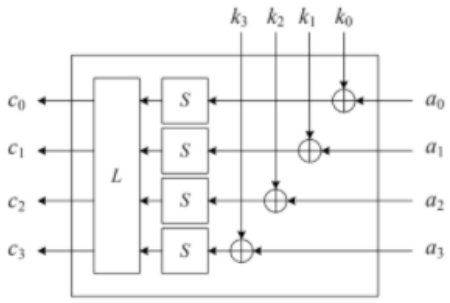


Fig. 2. The round function F of SMS4

$X_{i+4} = F(X_i, X_{i+1}, X_{i+2}, X_{i+3}, RK_i) = X_i \oplus T(X_{i+1} \oplus X_{i+2} \oplus X_{i+3} \oplus RK_i)$，T 为合成置换 $T : F_2^{32} \to F_2^{32}$，由非线性变换 τ 和线性变换 L 复合而成，即 $T(\cdot) = L(\tau(\cdot))$，其中 $B = \tau(A \oplus K) \Leftrightarrow (b_0, b_1, b_2, b_3) = (S(a_0 \oplus k_0), S(a_1 \oplus k_1), S(a_2 \oplus k_2), S(a_3 \oplus k_3))$, $C = L(B) = B \oplus (B <<< 2) \oplus (B <<< 10) \oplus (B <<< 18) \oplus (B <<< 24)$

为了方便讨论，我们定义一些概念，并给出一些命题

差分表 S 盒置换 $S : F_2^8 \to F_2^8$ 则

$$INs(\alpha, \beta) = \left\{x \in F_2^8 : S(x) \oplus S(x \oplus \alpha) = \beta\right\}$$

$$Ns(\alpha, \beta) = \sharp\left\{x \in F_2^8 : S(x) \oplus S(x \oplus \alpha) = \beta\right\}$$

命题 1 对于 SM4 的 S 盒，给定任意一个输入差分 $\alpha \neq 0$，存在 127 个可能的输出差分 β，其中只有一个满足 $Ns(\alpha, \beta) = 4$，其余 126 个则满足 $Ns(\alpha, \beta) = 2$

S 盒差分攻击 $y = S(x \oplus k)$，其中 x 为输入，k 为轮密钥，y 为 S 盒输出。假设攻击者有一对 S 盒输入 (x, x^*)，以及输出的差分值 $\beta = y \oplus y^* = S(x \oplus k) \oplus S(x^* \oplus k)$，则有 k 的候选集为

$$x \oplus INs(x \oplus x^*, \beta) = \{x \oplus z : z \in INs(x \oplus x^*, \beta)\}$$

命题 2 令 $S(\cdot)$ 为 SM4 S 盒置换，(x, x^*, β) 为 F_2^8 的随机三元组，则有

(1) $Pr\{Ns(x \oplus x^*, \beta) > 0\} = 0.4912$，即 $S(x \oplus k) \oplus S(x^* \oplus k) = \beta$ 有解的概率为 0.4942

(2) 若 $Ns(x \oplus x^*, \beta) > 0$，则 $E(Ns(x \oplus x^*, \beta)) = 2.0236$，即若 $S(x \oplus k) \oplus S(x^* \oplus k) = \beta$ 有解，则解个数的期望为 2.0236

命题 3 线性变换 $L(\cdot)$ 的逆变换 L^{-1} 有：

$$\begin{aligned} L^{-1}(C) = C \oplus (C <<< 2) \oplus (C <<< 4) \oplus (C <<< 8) \oplus (C <<< 12) \oplus (C <<< 14) \\ \oplus (C <<< 16) \oplus (C <<< 18) \oplus (C <<< 22) \oplus (C <<< 24) \oplus (C <<< 30) \end{aligned}$$

对某一轮的具体攻击流程如下：轮函数为 $F(A, K) = L \circ \tau \circ \sigma_K(A)(\sigma_K(A)$ 为轮密钥加)，攻击者对轮输入 $(X_i, X_{i+1}, X_{i+2}, X_{i+3})$ 进行故障注入到同一字的不同字节（不注入到 X_i），则得到错误 F 函数输入 A^*，同时得到轮函数输出异或 ΔC, 得到 32bits 的三元组 $(A, A^*, \Delta C)$

1. 计算 S 盒输出异或 $\Delta B = L^{-1}(\Delta C)$
2. 对 $i = 0, 1, 2, 3$，计算
   (a) $a_i = (A)_i, a_i^* = (A^*)_i, \Delta b_i = (\Delta B)_i$
   (b) $\langle k_i \rangle = a_i \oplus INs(a_i \oplus a_i^*, \Delta b_i)$
3. 若对每个 $i \in \{0, 1, 2, 3\}, \langle k_i \rangle \neq \Phi$,则轮密钥候选集为 $\langle K \rangle = \langle k_0 \rangle || \langle k_1 \rangle || \langle k_2 \rangle || \langle k_3 \rangle$
4. 若轮密钥某个字节可能值 $\langle k_i \rangle = \Phi$，则攻击失败，重新注入

上述方法恢复出最后 4 轮的密钥后，即可通过逆向密钥扩展算法最终得到初始的加密密钥

复杂度分析 要恢复轮密钥的某一个字节，平均需要注入在同一字节位置的两个错误密文，因此，恢复出该轮密钥评价需要 8 个不同错误密文。因此要完全恢复 SM4 密钥，理论上只要 32 个错误密文

3.2 算法流程图

3.3 调用关系图

3.4 算法伪代码

Algorithm 1 sm4_dfa 单轮单字节差分攻击

Input: 错误密文分组 $falseCyphers$，正确密文分组 $trueCyphers$，轮数 $round$，字节位置 $target$

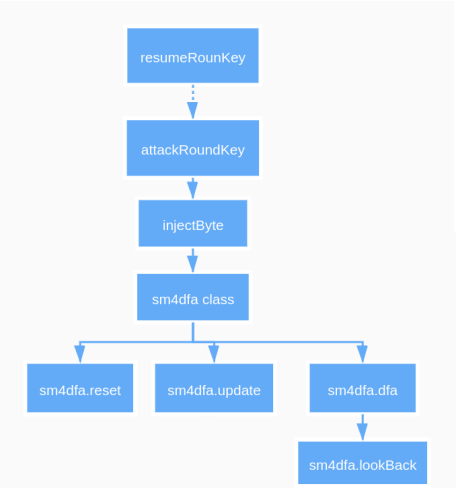


图 1: 函数调用

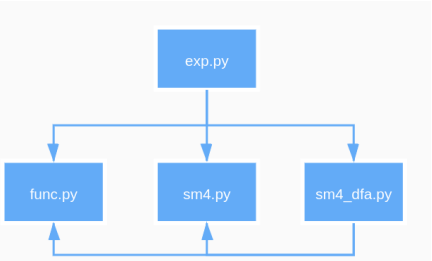


图 2: 程序调用

Output: 轮密钥单字节候选集 $\langle k_i \rangle$

$\langle k_i \rangle = \Phi$
for $k \leftarrow 0$ to 255 **do**
  for a, b in zip$(trueCyphers, falseCyphers)$ **do**
    $trueCypher \leftarrow R^{-1}(a)$
    $falseCypher \leftarrow R^{-1}(b)$
    if $round < 32$ **then**
      $trueCypher \leftarrow lookBack(trueCypher, roundKey)$
      $falseCypher \leftarrow lookBack(falseCypher, roundKey)$
    end if
    $tx1, tx2, tx3, tx4 \leftarrow trueCypher$
    $fx1, fx2, fx3, fx4 \leftarrow falseCypher$
    $outdiff \leftarrow L^{-1}(tx4 \oplus fx4)$
    $indiff \leftarrow (tx1 \oplus fx1) \oplus (tx2 \oplus fx2) \oplus (tx3 \oplus fx3)$
    $Sa \leftarrow outdiff[target]$
    $Sb \leftarrow S(fx1 \oplus fx2 \oplus fx3 \oplus k)$
    $Sc \leftarrow S(fx1 \oplus fx2 \oplus fx3 \oplus k \oplus indiff)$
    if $Sa == Sb \oplus Sc$ **then**
      if not $k \in \langle k_i \rangle$ **then**

19: $\langle k_i \rangle$.append(k)

20: **end if**

21: **end if**

22: **end for**

23: **end for**

24: **return** $\langle k_i \rangle$

Algorithm 2 sm4_dfa

Output: SM4 加密密钥 Key

1: $roundKey \leftarrow [\quad]$

2: **for** $i \leftarrow 32$ to 29 **do**

3: $rK_i \leftarrow [\quad]$

4: **for** $j \leftarrow 0$ to 4 **do**

5: $byteKey \leftarrow injectByte(target = j, round = i)$

6: rK_i.append(byteKey)

7: **end for**

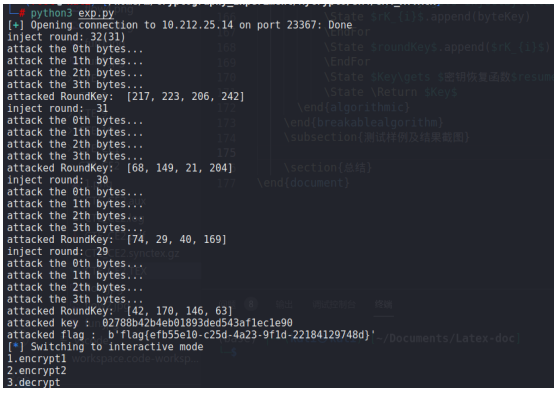
8: $roundKey$.append(rK_i)

9: **end for**

10: $Key \leftarrow$ 密钥恢复函数 $resumeKey(roundKey)$

11: **return** Key

3.5 测试样例及结果截图



4 总结

本次实验实现了对 SM4 的故障注入攻击，其中对 L 线性变换的逆变换推导运用到了有限域的知识，SM4 的故障注入攻击本身流程较为清晰。理论上只需要 32 个错误密文就可以恢复出 SM4 的 128bit 密钥，但由于实际故障诱导过程中故障发生的字节位置不可能完全平均，所以实际所需的错误密文数会略大于 32。

从实验中可以看出，差分故障攻击（DFA）是一种强大的侧信道攻击方法，其原理是以不可预测的环境条件诱导加密实现产生错误，来剖析其内部过程以及中间变量，最终恢复出需要的密钥。实际上，除了 DFA 攻击方式，常见侧信道攻击还有差分能量攻击 DPA，cache 攻击，TEMPEST 攻击等。