第33卷 第12期 计 算 机 工 程 2007年6月
Vol.33 No.12 Computer Engineering June 2007
·安全技术· 文章编号：1000—3428(2007)12—0191—02 文献标识码：A 中图分类号：TN918

针对RSA快速实现算法的计时攻击

张 鹏，陈开颜，赵 强
(军械工程学院计算机工程系，石家庄 050003)

摘 要：给出了一种改进的计时攻击方法。针对采用Montgomery模指数运算和中国剩余定理的RSA快速实现算法，通过分析在Montgomery模指数运算中额外约简发生的概率，得到RSA输入参数与运行时间之间的关系，并通过选择密文输入，计时分析按位获取RSA的秘密因子，最终破解了RSA的因子分解。

关键词：RSA；计时攻击；Montgomery约简；中国剩余定理；额外约简

Timing Attack on Fast Implementation Algorithm of RSA

ZHANG Peng, CHEN Kaiyan, ZHAO Qiang
(Dept. of Computer Engineering, Ordnance Engineering College, Shijiazhuang 050003)

【Abstract】 This paper presents an improved method for timing attack. It can attack the fast implementation of RSA, including Montgomery exponentiation algorithm and Chinese remainder theorem. It gets the relationship between the input parameters and execution time of RSA by analyzing the probability of extra reduction during Montgomery exponentiation operation, and gets the secret factors bit by bit by timing analyzing with the chosen input ciphertexts, and exposes the factorization of RSA.

【Key words】 RSA; Timing attack; Montgomery reduction; Chinese remainder theorem; Extra reduction

RSA是当前使用最为广泛的公钥密码算法。它的安全性在于大整数因子分解问题的困难性。对于密钥强健、使用正确的RSA来说，用传统的数学分析方法和穷举攻击方法进行破解十分困难。近年来又诞生了一类新的解密技术，称为旁路密码分析(side channel cryptanalysis)[1]，又称旁路攻击(side channel attacks，SCA)。计时攻击作为一种很重要的旁路攻击方法由Kocher[2]提出，它通过分析密码设备执行密码操作所花费的时间，推导出加密系统在计算中涉及到的秘密参量。它对于普通的RSA实现来说是可行的[3]，但是由于RSA在实际应用中常采用Montgomery模指数算法，并结合中国剩余定理(CRT)进行快速解密，因此对时间的分析非常困难，使得传统的计时攻击基本失效。本文对这些快速算法进行分析，提出了一种改进的计时攻击方法，可以有效地对RSA的快速实现进行攻击。

1 RSA的快速实现算法

计时攻击的目标是最终获取RSA的私钥。由于私钥操作只存在于RSA的解密过程中，因此攻击的对象是RSA的解密操作。

1.1 RSA的解密操作

RSA算法解密的核心是一个模指数运算：$M=C^d \bmod N$，其中，$N=pq$是RSA的模数；p、q是素因子；d是解密私钥；C是要被解密的密文。通常使用CRT加速执行该模指数运算。

算法1 使用中国剩余定理的RSA解密

输入 c，d，p，q

输出 $c^d \bmod n$，其中，$n=pq$

(1)$m_p \leftarrow m \bmod p$，$m_q \leftarrow m \bmod q$，$d_p \leftarrow d \bmod (p-1)$，$d_q \leftarrow d \bmod (q-1)$；

(2)$x_p \leftarrow m_p^{d_p} \bmod p$，$x_q \leftarrow m_q^{d_q} \bmod q$；

(3)$m \leftarrow q(q^{-1} \bmod p)\, x_p + p(p^{-1} \bmod q)\, x_q \bmod n$；

(4)返回(m)。

1.2 模指数运算

算法1中，由于q^{-1}和p^{-1}都可以预计算得到，因此算法的时间主要在于步骤(2)中的2个模指数运算。它们所采用的算法是相同的。下面分析$x^k \bmod m$的计算过程。

1.2.1 从左至右的二元指数运算(重复平方乘算法)

计算$x^k \bmod m$的最简单的算法是重复平方乘算法[4]。

算法2 从左至右的重复平方乘算法

输入 x，正整数$k=(k_i k_{i-1} \ldots k_1 k_0)_2$

输出 $x^k \bmod m$

(1)$A \leftarrow 1$；

(2)对于t从i递减到0，执行：

1) $A \leftarrow A \cdot A \bmod m$；

2)若$k_t=1$，则$A \leftarrow A \cdot x \bmod m$；

(3)返回(A)。

1.2.2 Montgomery乘法

算法2中步骤(2)进行以m为模的约简，因为平方也是2数相乘，所以它们可以采用相同的算法。最简单的方法是通过多精度除法然后返回余数，但这样做代价很大。1985年，Montgomery发明了一种方法执行乘法取模操作，该方法被称为Montgomery约简[5]。

基金项目：国家自然科学基金资助项目(60571037)；军械工程学院科学研究基金资助项目(YJJXM0630)

作者简介：张 鹏(1976－)，男，硕士生，主研方向：信息安全；陈开颜，副教授、博士生；赵 强，教授、博导

收稿日期：2006-08-15 **E-mail：**zhangp210@163.com

设m是正整数，R和T是整数，满足$R>m$，gcd(m,R)=1，$0 \le T < mR$。这里gcd表示求最大公因子。Montgomery用一种方法计算$TR^{-1} \mod m$。$TR^{-1} \mod m$就称为T模m关于R的Montgomery约简。对于使用CRT加速的RSA算法，R典型选择2^w(这里w是因子m的二进制位数)，显然$R>m$，gcd(m,R)=1均满足。

对2个整数的乘积进行以m为模的约简，为了使用Montgomery约简算法，须先将乘积化为Montgomery形式，即乘以R^{-1}。结合了Montgomery约简的Montgomery乘法算法见算法3。算法3的输出与$R \mod m$相乘可得$xy \mod m$。

算法3 Montgomery 乘法

输入 整数$m=(m_{n-1}...m_1m_0)_2$，$x=(x_{n-1}...x_1x_0)_2$，$y=(y_{n-1}...y_1y_0)_2$，满足$0 \le x,y < m$，$R=2^n$，gcd(m,2)=1，$m'=-m^{-1} \mod 2$

输出 $xyR^{-1} \mod m$

(1)$A \leftarrow 0$(记为$A=(a_na_{n-1}...a_1a_0)_2$)；

(2)对于i从0到(n-1)，执行：

1)$u_i \leftarrow (a_0 + x_iy_0)m' \mod 2$；

2)$A \leftarrow (A + x_iy + u_im)/2$；

(3)若$A \ge m$，则$A \leftarrow A - m$；

(4)返回(A)。

将算法3简记为Mont(x, y)=$xyR^{-1} \mod m$，其步骤(3)须检查输出的A是否大于等于m，如果是，从输出中减去m以确保输出A的范围是[0,m]。这个额外的步骤称为一个额外约简。正是额外约简的存在导致对于不同的输入，算法3耗费的时间不同。

1.2.3 Montgomery 模指数运算

综合运用算法2和算法3可以得到计算$x^k \mod m$的Montgomery模指数运算方法。

算法4 Montgomery 模指数运算

输入 整数$m=(m_{n-1}...m_1m_0)_2$，$R=2^n$，$m'=-m^{-1} \mod 2$，$k=(k_i\ k_{i-1}...k_1k_0)_2$，$k_i=1$，整数$x$，$1 \le x < m$

输出 $x^k \mod m$

(1)$\tilde{x} \leftarrow$Mont(x,$R^2 \mod m$)，$A \leftarrow R \mod m$ ($R^2 \mod m$和$R \mod m$可以预先计算)；

(2)对于t从i递减到0，执行：

1)$A \leftarrow$Mont(A, A)；

2)若$k_t=1$，则$A \leftarrow$Mont(A, $\tilde{x}$)；

(3)$A \leftarrow$Mont(A, 1)；

(4)返回(A)。

在算法4中，循环的执行序列仅与指数k相关。而对于的RSA解密运算来说，解密指数(私钥)是固定的，因此，算法4的运算时间仍然取决于Mont函数的运算时间。

2 计时攻击

2.1 攻击原理

如前面所述，在Montgomery乘法中，最后需要对额外约简操作进行判断。对于不同的输入，是否进行额外约简操作是不相同的，这将导致算法运行时间有小的差别。由于算法4多次调用Mont函数，因此算法4的运行时间差可能足以被精确的时间测量检测到。Schindler指出[6]：在一个模指数运算$x^k \mod m$期间，发生一个额外约简的概率与x同m之间的近似度对应成比例。发生一个额外约简的概率是

$$\frac{x \bmod m}{2R} \qquad (1)$$

这里对R的定义与算法3相同。

对于使用CRT的RSA来说，式(1)中的模数m对应于RSA的因子p或q。因此，如果x从下方接近于p或q，在模指数运算期间的额外约简的数目将显著增长。当x精确地为p或q的倍数时，额外约简的数目急剧下降，导致运算耗时减小。换言之，如果2个输入x_1、x_2比较接近的时候，则$x_1<q$ ($x_1<p$)时解密要比$x_2>q$ ($x_2>p$)时解密慢。图1显示了这个关系，图中断点发生在p或者q的倍数位置。可以根据RSA解密操作的运行时间猜测得到因子p或是q。

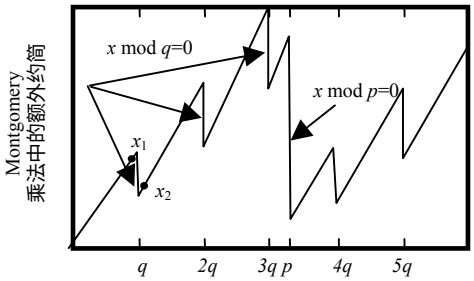

图1 Montgomery 乘法中的额外约简的数目

2.2 改进的计时攻击方法

下面给出通过计时攻击破解RSA模块的因子分解的方法。令$N=pq$，不妨设$q<p$，N为1 024(或其它)位二进制数，攻击目标是获取q。记$q=(q_{\omega-1}q_{\omega-2}...q_0)_2$，通过建立$q$的逐渐接近的猜测数进行攻击，即从最高位开始，每次获取1比特位数值，从而逐步改进猜测数。

对于q的猜测数记为g。根据通常的因子选择规则，$2^{511}<g<2^{512}$，因此$\omega=512$，且$q_{511}=1$。

假设已找到$q_{\omega-1}$~q_{i+1}位的值，之后可用算法5获取q_i位的值。在算法5中，带下标的变量均表示该变量的二进制展开式的一个比特位。

算法5 寻找RSA因子q的二进制展开式第i位

输入 q的高ω-1~i+1位比特值$q_{\omega-1}q_{\omega-2}...q_{i+1}$，$N$，$R$，判别阈值THR

输出 q的第i位值q_i

(1)$g_{\omega-1}g_{\omega-2}...g_{i+1} \leftarrow q_{\omega-1}q_{\omega-2}...q_{i+1}$，$g_ig_{i-1}...g_0 \leftarrow 0$，$h \leftarrow g$；

(2)$h_i \leftarrow 1$；

(3)$x \leftarrow gR^{-1} \mod N$，$x' \leftarrow hR^{-1} \mod N$；

(4)$t \leftarrow$RsaTime(x)，$t' \leftarrow$RsaTime(x')；

(5)$\triangle \leftarrow |t - t'|$；

(6)若$\triangle \ge$THR，则$q_i \leftarrow 0$；否则$q_i \leftarrow 1$；

(7)返回(q_i)。

算法5中，RsaTime( )是对使用算法4进行运算的计时函数。步骤(3)确保进入算法4循环体的输入值$\tilde{x}$等于g或h。可以看到，如果$q_i=1$，那么$g<h<q$，由2.1节可知，$\triangle$将很“小”；如果$q_i=0$，那么$g<q<h$，$\triangle$将很“大”。判别$\triangle$值的大小是通过阈值常数THR，该常数可以通过对同一个RSA解密模块输入已知的密文和解密指数进行多次试验预先确定的。

从q的第2位开始反复执行算法5，就可以得到完整的q。一旦获取了因子q，可由简单算法得到RSA的私钥。

如果考虑算法5中g和h的差值相对于q不应太大(由图1可看出，差值太大会导致2个输入点分隔太远，精度下降)，那么对于q的高几位比特值可采用穷举，剩下的位通过反复执行算法5获取。另外，为了减小噪声的影响，步骤(4)测量t (t')时可以用相同的x (x')进行多次测量然后取均值。

(下转第204页)