文章编号:1001 - 9081(2006)12 - 2928 - 03

RSA 密码算法的一种新的快速软件实现方法

贺毅朝[1],张建勋[1],王彦祺[1],田俊峰[2]
(1. 石家庄经济学院 信息工程学院,河北 石家庄 050031;
2. 河北大学 数学与计算机学院,河北 保定 071002)
(heyichao@ sjzue. edu. cn)

摘 要:在介绍标准 RSA 密码系统的基础上,利用计算近似最短加法链算法给出了软件实现模幂运算的一种改进方法;基于求解孙子定理的混合基数计算算法(MRC)改进了 RSA 的解密方法;最后,结合快速有效的素数测试方法提出了一种能够快速软件实现 RSA 密码算法的新方法,并分析比较了各相关算法的计算效率。实验结果表明:利用该方法实现的 RSA 密码软件系统,可使加、解密运算速度平均提高 6 ~ 10 倍。

关键词:PKC 算法;RSA 算法;最短加法链;孙子定理;混合基数计算算法
中图分类号: TP309.7 **文献标识码**:A

New faster software implementation method of RSA algorithm

HE Yi-chao[1], ZHANG Jian-xun[1], WANG Yan-qi[1], TIAN Jun-feng[2]
(1. *Information Engineering School, Shijiazhuang University of Economics, Shijiazhuang Hebei* 050031, *China*;
2. *College of Computer and Mathematics, Hebei University, Baoding Hebei* 071002, *China*)

Abstract: Based on introducing standard RSA crytsystem, we advanced an improved method of the software implementation of module power operation using approximate algorithm of calculation shortest addition chains, and improved decryption method of RSA on the basement of Mixed-Radix Conversion (MRC) which is to solve Chinese Remainder Theorem (CRT). Finally, combined with rapid effective prime testing method, a new algorithm that can rapid software implementation RSA cryptsystem was proposed, and it has also been analyzed and compared to other related algorithms. Experimental results show that operation velocity of encryption and decryption operation can be increased 6 to 10 times on average by using the new method.

Key words: Public Key Cryptosystem (PKC) algorithm; RSA algorithm; shortest addition chains; Chinese remainder theorem; Mixed-Radix Conversion (MRC)

0 引言

公开密钥密码体制(Public Key Cryptosystem,PKC)[1]开辟了密码学研究的新方向,此后,人们基于背包问题、因子分解问题和离散对数问题等数学难题[2]提出了大量的 PKC 算法,例如著名的 RSA 算法[3]、Rabin 算法[4]、Chor-Rivest 算法、ElGamal 算法和 ECC 算法等[5]。其中,RSA 算法是一种基于因子分解的指数函数作为单向陷门函数[1]的 PKC 算法,是第一个完善并且简单实用的 PKC 算法。近年来,国内外学者对 RSA 密码算法提出了多种攻击方法[2,5],例如 Pollard p-1 方法、二次筛法、椭圆曲线算法和数域筛法等。实践证明,在当前的技术和方法下,密钥不小于 1 024 bit 的 RSA 算法仍然是安全的[2,6]。因此,尽管先后出现了很多新的 PKC 算法,但 RSA 仍然在不同应用领域占据了重要的位置。

为了提高 RSA 密码算法的软硬件实现速度,人们提出了许多可行的方法,例如旨在快速产生大素数的 Monte Carlo 概率算法[2],加速模幂运算的 m-ary 法、Yacobi 法、加法链法和向量加法链[5,6]法,以及利用孙子定理(CRT)改进 RSA 的解密运算等[9,11]。本文首先介绍了 RSA 密码算法、素数测试概率算法、模幂运算的加法链法和数论中著名的孙子定理,然后结合文献[8]提出的计算近似最短加法链算法,给出了模幂运算的改进实现方法;同时,基于混合基数计算算法(Mixed-Radix Conversion,MRC)[10]改进了文献[5]中利用 CRT 加速 RSA 的解密方法。在此基础上,提出了一种能够快速软件实现 RSA 密码算法的新方法。实验表明,该方法与基本 RSA 算法的实现相比较,使 RSA 密码系统的运算速度提高将近 6 ~ 10 倍,加快了密码系统的运行速度。

1 背景知识介绍

1.1 标准 RSA 密码算法

在 RSA 密码算法初始化阶段,首先随机生成两个不同的安全的强素数 p 和 q,计算 $N = pq$ 和 $\varphi(n) = (p-1)*(q-1)$,并将 p 与 q 保密。再随机选取正整数 e,$e < \varphi(N)$ 且 $GCD(e, \varphi(N)) = 1$,根据扩展 Euclid 算法[3]计算 $d = e^{-1} (\mathrm{mod}\ \varphi(N))$。于是 RSA 算法的公钥 $PK = (e,N)$,私钥 $SK = (d,p,q)$。

RSA 算法的加密计算公式为:$c = E_k(m) = m^e (\mathrm{mod}\ N)$,其中 $1 \leqslant m \leqslant N-1$ 为明文。解密计算公式为:$m = D_k(c) = c^d (\mathrm{mod}\ N)$,其中 $1 \leqslant c \leqslant N-1$ 为密文。

收稿日期:2006 - 06 - 20;**修订日期**:2006 - 08 - 28
基金项目:河北省自然科学基金资助项目(402400);河北省教育厅科研资助项目(2005338)
作者简介:贺毅朝(1969 -),男,河北晋州人,讲师,硕士,主要研究方向:智能算法、密码学和计算复杂性理论; 张建勋(1978 -),男,河北石家庄人,助教,硕士,主要研究方向:分布式计算与网络技术; 王彦祺(1948 -),男,吉林长春人,教授,主要研究方向:信息安全与算法理论;田俊峰(1965 -),男,河北保定人,教授,博士生导师,主要研究方向:并行处理、网络安全、网格计算.

RSA 密码算法是一种分组密码,当明文 m 的大小超过 N 时,应将 m 分成若干个 1 到 $N-1$ 之间的整数,即 $m = m_1 m_2 \cdots m_k$,其中 $m_i \in [1, N-1]$。相应地,密文 $c = c_1 c_2 \cdots c_k$ 且 $c_i = E_k(m_i)$,$i = 1,2,\cdots,k$。综上,标准 RSA 密码算法的伪代码为:

算法 1　Standard RSA Algorithm

1) Initialize:

Random_Prime (p, q);　　//要求 p, q 为强素数

$N \leftarrow p * q$; $\varphi(N) \leftarrow (p-1)(q-1)$;

Select $e(e < \varphi(N))$ satisfied $GCD(e, \varphi(N)) = 1$;

　　//利用扩展 Euclid 算法很容易实现

$d \leftarrow e^{-1} (\mathrm{mod}\ \varphi(N))$;

$PK \leftarrow (e, N)$, $SK \leftarrow (d, p, q)$

2) Encryption:

Input_Plaintext ($m_1 m_2 \cdots m_k$);

$c_i \leftarrow E_{pk}(m_i) = m_i^e (\mathrm{mod}\ N)$;

3) Decryption:

$m_i \leftarrow D_{sk}(c_i) = c_i^d (\mathrm{mod}\ N)$

1.2　强素数的随机生成方法

一般地,产生强素数的方法有两种:一种方法是利用确定性算法迭代生成满足条件的素数,如 Demytko 算法[5];另一种方法是先随机生成一个大奇数,然后再利用概率算法对其进行测试,通过测试确定其是否为素数,例如 Lehmann 算法和 Rabin-Miller 算法[5,6]。迭代算法生成的素数往往缺少随机性,存在潜在的缺陷,而概率算法不仅是多项式时间算法,而且实现简单、速度快,因此在实际应用中通常采用第二种方法。在文献[7]中,利用 Monte Carlo 型概率算法——Miller-Rabin 算法,结合素数测试优化策略以及运算数与 Visual C++ 的特性,基于递归算法设计技术提出了一种快速素数测试方法,以下简记 FastPrimalityTest(P)。利用此方法可快速测定大奇数 P 是否为强素数。

1.3　快速模幂运算的改进算法

所谓正整数 n 的长度为 r 的加法链是一个递增的整数序列 $\{a_0, a_1, \cdots, a_r\}$,它满足下面两个条件:(1) $1 = a_0 < a_1 < \cdots < a_r = n$;(2) $a_i = a_i + a_j$, $k \leqslant j < i$,对所有的 $i = 2,3,\cdots,r$。如果已知 n 的加法链为 $\{a_0, a_1, \cdots, a_r\}$,则 $x^n (\mathrm{mod}\ N)$ 的计算为 $x (\mathrm{mod}\ N)$, $x^{a_1} (\mathrm{mod}\ N)$, $x^{a_2} (\mathrm{mod}\ N)$, $\cdots$, $x^{a_r} (\mathrm{mod}\ N)$。例如计算 $x^{23} (\mathrm{mod}\ N)$ 时,由 23 的加法链为序列 $\{1,2,3,5,10,20,23\}$,只需计算 $x (\mathrm{mod}\ N)$, $x^2 (\mathrm{mod}\ N)$, $x^3 (\mathrm{mod}\ N)$, $x^5 (\mathrm{mod}\ N)$, $x^{10} (\mathrm{mod}\ N)$, $x^{20} (\mathrm{mod}\ N)$, $x^{23} (\mathrm{mod}\ N)$;若令 $T = x^{23} (\mathrm{mod}\ N)$,则只需迭代计算 $T_1 = x^2 (\mathrm{mod}\ N)$, $T_2 = T_1 * x (\mathrm{mod}\ N)$, $T_3 = T_1 * T_2 (\mathrm{mod}\ N)$, $T_4 = T_2 * T_3 (\mathrm{mod}\ N)$, $T_5 = T_{42} (\mathrm{mod}\ N)$, $T = T_2 * T_5 (\mathrm{mod}\ N)$,共 6 次模 N 乘法运算。但已经证明:计算正整数 n 的最短加法链问题是一个 NPC 问题[12],利用目前算法所求得加法链多为近似最短加法链。

1.4　孙子定理及其计算改进

孙子定理(又称为中国剩余定理,记为 CRT)[9,11]是计算数论中的基本定理之一,主要是用于刻画剩余系的结构和计算同余方程组 $x \equiv d_i (\mathrm{mod}\ p_i)$ 的解,其中 p_i 为素数,$1 \leqslant i \leqslant s$。

定理 1　孙子定理。设 $p_1, p_2, \cdots, p_s$ 是 s 个两两互素的正整数,$P = p_1 p_2 \cdots p_s$,对于任意整数 $d_i (0 \leqslant d_i < p_i, i = 1,2,\cdots,s)$,方程组 $x \equiv d_i (\mathrm{mod}\ p_i)$ 有惟一解 x,且 $x = d_1 P_1 Q_1 + d_2 P_2 Q_2 + \cdots + d_s P_s Q_s (\mathrm{mod}\ P)$,其中 $P_i \leftarrow P/p_i$, $Q_i \leftarrow P_{i-1} (\mathrm{mod}\ p_i)$ $(1 \leqslant i \leqslant s)$。

推论 1[9]　设 $p_1, p_2, \cdots, p_s$ 是 s 个两两互素的正整数,$P = p_1 p_2 \cdots p_s$,则同余式 $f(x) \equiv 0 (\mathrm{mod}\ P)$ 的解与同余方程组 $f(x) \equiv 0 (\mathrm{mod}\ p_i)$ $(1 \leqslant i \leqslant s)$ 的解相同。

定理 2　设 p 是一个素数,x 是一个满足 $x (\mathrm{mod}\ p) \neq 0$ 的整数,则 $x^{p-1} \equiv 1 (\mathrm{mod}\ p)$。

MRC 是 CRT 的一种改进算法。尽管在一般意义下,利用 MRC 计算同余方程组并没有较大的优势,但是将此方法用于 RSA 的解密运算时,却能够极大地提高运算速度。下面以定理的形式给出 MRC 的数学描述。

定理 3　设 $p_1, p_2, \cdots, p_s$ 是两两互素的正整数,对于任意整数 $d_i (0 \leqslant d_i \leqslant p_i - 1, i = 1,2,\cdots,s)$,同余式方程组 $x \equiv d_i (\mathrm{mod}\ p_i)$ 有惟一解 x,x 的计算如下:

(1) 计算 $B_{ji} = p_j^{-1} (\mathrm{mod}\ p_i) (1 \leqslant j < i \leqslant s)$;

(2) 计算 $v_1 = d_1, v_2 = (d_2 - v_1) B_{12} (\mathrm{mod}\ p_2), \cdots, v_s = (d_s - (v_1 + p_1(v_2 + p_2(v_3 + \cdots + p_{s-2} v_{s-1}) \cdots))) B_{1s} \cdots B_{(s-1)s} (\mathrm{mod}\ p_s)$;则解为 $x = v_s p_{s-1} \cdots p_2 p_1 + \cdots + v_3 p_2 p_1 + v_2 p_1 + v_1$。

对于上述定理,计算 $v_1, v_2, \cdots, v_s$ 的过程可以利用递推公式进行迭代计算。设 $a_{i1} = d_i (1 \leqslant i \leqslant s)$,则易得 $a_{i(j+1)} = (a_{ij} - a_{jj}) B_{ji} (\mathrm{mod}\ p_j)$,其中 $1 \leqslant j < i \leqslant s$。因为 $p_j B_{ji} \equiv 1 (\mathrm{mod}\ p_i)$, $(1 \leqslant j < i \leqslant s)$,于是:

$$
\begin{aligned}
a_{11} &= d_1 = v_1 \\
a_{21} &= d_2, \\
a_{22} &= (a_{21} - a_{11}) B_{12} (\mathrm{mod}\ p_2) \\
&= (d_2 - v_1) B_{12} (\mathrm{mod}\ p_2) = v_2; \cdots; \\
a_{s1} &= d_s, \\
a_{s2} &= (a_{s1} - a_{11}) B_{1s} (\mathrm{mod}\ p_s) \\
&= (d_s - v_1) B_{1s} (\mathrm{mod}\ p_s), \cdots, \\
a_{ss} &= (a_{s(s-1)} - a_{(s-1)(s-1)}) B_{(s-1)s} (\mathrm{mod} p_s) \\
&= (a_{s(s-1)} - v_{(s-1)}) B_{(s-1)s} (\mathrm{mod} p_s) \\
&= ((a_{s(s-2)} - a_{(s-2)(s-2)}) B_{(s-2)s} - v_{(s-1)}) B_{(s-1)s} (\mathrm{mod} p_s) \\
&= ((a_{s(s-2)} - v_{(s-2)(s-2)}) B_{(s-2)s} - \\
&\quad v_{(s-1)} p_{(s-2)} B_{(s-2)s}) B_{(s-1)s} (\mathrm{mod}\ p_s) \\
&= \cdots \\
&= (d_s - (v_1 + p_1(v_2 + p_2(v_3 + \cdots + \\
&\quad p_{s-2} v_{s-1}) \cdots))) B_{1s} B_{2s} \cdots B_{(s-1)s} (\mathrm{mod}\ p_s) \\
&= v_s
\end{aligned}
$$

这样,$x = a_{11} + a_{22} p_1 + a_{33} p_1 p_2 + \cdots + a_{ss} p_1 p_2 \cdots p_s$。

2　RSA 算法的快速实现方法

2.1　实现模幂运算的改进方法

在文献[8]中,基于构造对状态空间树进行剪枝的精细剪枝函数,提出了求解任意正整数 n 的最短(或近似最短)加法链算法,下面简记为 IterativeSearch(n)。此外,在文献[10]中给出了目前已知计算最短加法链长度 r 的一个最好上界 $l(n) = \lfloor \log_2 n \rfloor + b(n) + 1$, $b(n)$ 为 n 的二进制表示中 1 的个数。因此,利用上述结论,可将模幂运算的实现方法改进如下:

设 $len = 2\lfloor \log_2 n \rfloor + 1$,令 $A[len+1]$ 和 $X[len+1]$ 为两个整型数组,数组 A 用于保存所求 n 的最短(或近似最短)加法链,其中 $A[0]$ 存放加法链的实际长度,数组 A 用于保存模幂运算的中间结果。又设 $B[len][2]$ 为一辅助数组,保存模幂运算中各计算步骤进行模乘运算的对应项。则模幂运算 $x^n (\mathrm{mod}\ N)$ 的改进实现方法如下:

算法 2　FastModeExpon(X, A[len])

//A[len]←IterativeSearch(n);

1) For $i = A[0]$ downto 2 Do
2) For $j = A[0]$ downto 2 Do
3) If $(A[i-1]+A[j-1] = A[i])$ *then* $B[i][0] \leftarrow i-1$, $B[i][1] \leftarrow j-1$ goto 4
4) $X[1] \leftarrow x \pmod N$;
5) For $i = 2$ to $A[0]$ Do
6) $X[i] \leftarrow X[B[i][0]] * X[B[i][1]] \pmod N$;
7) Return $X[A[0]]$;

对于合法的使用者,由于 RSA 密码系统建立后 e 和 d 的值不再改变,因此只需分别利用算法 IterativeSearch(n)计算一次 e 和 d 的最短(或近似最短)加法即可。这样,无论是加密还是解密中的模幂运算,均直接利用算法 2 快速实现。

2.2 基于 MRC 的改进解密算法

由于拥有私钥 $SK = (d,N)$ 的合法解密者已知 $N = pq$(p 和 q 为不同强素数),利用推论 1,*RSA* 的解密计算 $m = D_{sk}(c) = c^d \pmod N$ 等价于计算同余式方程组[9]:

$$\begin{cases} x_1 \equiv c^d \pmod p \\ x_2 \equiv c^d \pmod q \end{cases} \tag{1}$$

令 $r = d \pmod{p-1}$,由于 d 和 c 往往不小于 p 或 q,因此利用定理 2 和同余式的性质知,必存在 k 满足 $d = k(p-1) + r$。于是 $x_1 \equiv c^d \pmod p \equiv c^{k(p-1)+r} \pmod p \equiv c^{k(p-1)} c^r \pmod p \equiv 1 * c^r \pmod p \equiv c^{d \pmod{(p-1)}} \pmod p$;同理 $x_2 \equiv c^d \pmod q \equiv c^{d \bmod (q-1)} \pmod q$。则方程组(1)可简化为:

$$\begin{cases} x_1 \equiv (c \pmod p)^w \pmod p \\ x_2 \equiv (c \pmod q)^u \pmod q \end{cases} \tag{2}$$

其中 $w = d \pmod{p-1}$,$u = d \pmod{q-1}$。对于(2)式,利用定理 3 即可得到基于 MRC 改进的 RSA 的解密算法,简记 MRC_Decryption(c),具体描述为:

算法 3 MRC_Decryption(c)

$T \leftarrow p^{-1} \pmod q$;
$w \leftarrow d \pmod{p-1}$, $u \leftarrow d \pmod{q-1}$;
$X_1 \leftarrow c^w \pmod p$, $X_2 \leftarrow c^u \pmod q$;
$x \leftarrow X_1 + [(X_2 - X_1) * T \pmod q] * p$;
Return (x)

2.3 RSA 的快速软件实现方法

综合快速素数测试算法 FastPrimalityTest(P)、近似最短加法链算法 IterativeSearch(n)、实现模幂运算的改进方法 FastModeExpon(X, n)和改进的 RSA 解密算法 MRC_Decryption(c),可以得到 RSA 密码算法的一种快速软件实现方法。

设 $e[k_1]$ 用于保存 e 所对应的近似最短加法链,设 $w[k_2]$ 和 $u[k_3]$ 分别保存算法 3 中 $w = d \pmod{p-1}$ 和 $u = d \pmod{q-1}$ 所对应的近似最短加法链;T, X_1, X_2, w, u 均为整型临时变量,则改进的 *RSA* 密码算法的快速软件方法为:

算法 4 Modified RSA Algorithm

1) Initialize:

FastPrimalityTest(p) and FastPrimalityTest(q);
$N \leftarrow p * q$; $\varphi(N) \leftarrow (p-1)(q-1)$;
Select $e(e < \varphi(N))$ satisfied $GCD(e,\varphi(N)) = 1$;
$d \leftarrow e^{-1} \pmod{\varphi(N)}$;
$e[k_1] \leftarrow IterativeSearch(e)$;
$T \leftarrow p^{-1} \pmod q$, $w \leftarrow d \pmod{p-1}$, $u \leftarrow d \pmod{q-1}$;
$w[k_2] \leftarrow$IterativeSearch(w) and $u[k_3] \leftarrow$IterativeSearch(u);
$PK \leftarrow (e[k_1], N)$, $SK \leftarrow (d[k_2], p, q)$

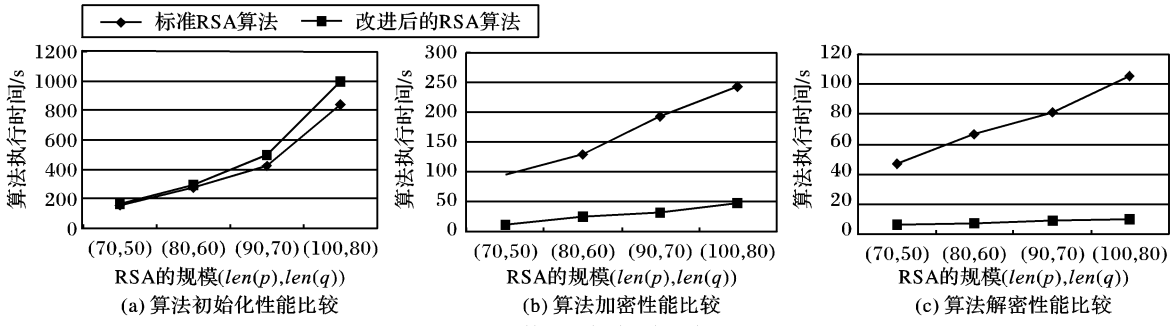
2) Encryption:

Input_Plaintext($m_1 m_2 \cdots m_k$);
$c_i \leftarrow E_{pk}(m_i)$ = FastModeExpon(m_i, $e[k_1]$)

3) Decryption:

$X_1 \leftarrow$FastModeExpon(c_i, $w[k_2]$);
$X_2 \leftarrow$FastModeExpon(c_i, $u[k_3]$);
$m_i \leftarrow X_1 + [(X_2 - X_1) * T \pmod q] * p$

算法 4 与算法 1 相比较,算法初始化阶段和解密计算步骤虽然增多,但由于算法 4 中模数由原来的 $N = pq$ 减小到最大不超过 $\max\{p,q\}$,使得其素数测试和模幂运算简化,加密和解密计算量减小;而且由于算法初始化阶段的各步骤均可预计算,从而能够有效地提高系统的运算速度。



(a) 算法初始化性能比较 (b) 算法加密性能比较 (c) 算法解密性能比较

图 1 算法的各种性能比较

3 实验结果分析

在 DELL Pentium(R)4-CPU1.69GHz 微型计算机上,利用 C++ 语言对算法 1 和算法 4 编程实现,其中的 $GCD(e, \varphi(N))$ 和 $e^{-1} \pmod{\varphi(N)}$ 的计算采用扩展 Euclid 算法实现,大数乘法与求余运算利用基本方法实现。在算法1的实现中,系统初始化阶段的大素数生成,采用直接由 Miller-Rabin 算法测试随机生成的大奇数;同时,利用常用的"平方—乘"方法实现模幂运算。在算法 4 中,大素数的生成采用文献[7]中的快速素数测试方法,模幂运算利用算法 2 实现。

RSA 密码系统的规模由 $(len(p), len(q))$ 表示,其中 $len(p)$ 表示素数 p 的位数,$len(q)$ 表示素数 q 的位数,当:

p = 6 706 820 460 936 504 732 888 953 833 234 682 047 870 309 910 435 116 442 888 764 737 892 699

q = 59 053 795 389 775 275 694 674 628 180 318 134 151 674 809 968 241

时,RSA 密码系统的规模为(70,50)。公钥 $PK = (e, N)$ 和私钥 $SK = (d, p, q)$ 随机选取,其中 e 固定为 45 位的随机十进制奇数;明文采用随机输入的十进制整数序列,并且序列长度为 1000。对不同规模的 RSA 密码系统,分别独立运行 50 次,从系统的初始化(图 1(a))、加密运算(图 1(b))和解密运算(图 1(c))三方面统计 RSA 密码系统所耗费的时间。

(下转第 2934 页)

从图2可以看出,采用本文算法部署的疫苗存储库,总体上能够使疫苗分发时间达到最短。同时也可以发现,疫苗存储库节点数目并不是越多越好。更重要的是疫苗存储库节点的分布,同样的数目但分布不同,所花的时间相差很大。

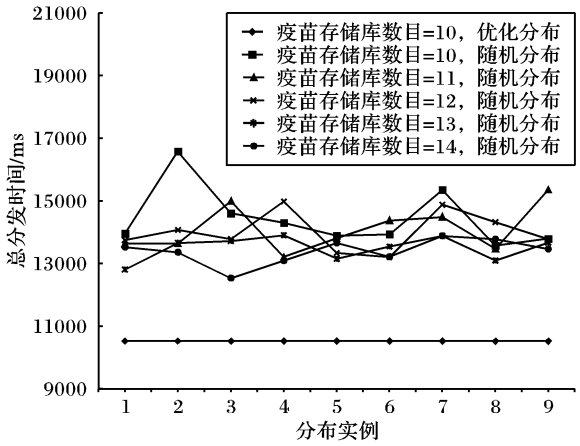


图2 各种疫苗存储库部署的分发时间比较

4 结语

网络蠕虫防御系统的研究是目前网络安全领域的热点。只有在足够短的时间内遏制蠕虫的蔓延,蠕虫防御才能起到真正的作用。本文提出了一个面向 Agent 的蠕虫防御系统 AOSWD,各个 Agent 之间相互独立又相互协作。由于采用分布冗余机制,增强了整个系统的健壮性。通过对疫苗分发策略和疫苗存储库的部署策略的研究,提高了系统响应时间。

今后的工作包括对系统的部署策略以及网络蠕虫的传播路径预测和边界防御策略展开更加深入的研究,进一步缩短系统的响应时间,更有效地遏制网络蠕虫的传播。

参考文献:

[1] CHEN ZS, GAO LX, KWIAT K. Modeling the spread of active worms[A]. Twenty-Second Annual Joint Conference of the IEEE Computer and Communications Societies[C]. IEEE, 2003, Vol 3: 1890 - 1900.

[2] NICOL D. Models of Active Worm Defense[A]. Proceedings of the Measurement, Modeling and Analysis of the Internet (IMA Workshop '04)[C]. Urbana-Champaign, Illinois, 2004.

[3] WONG C, WANG C, SONG D, *et al*. Dynamic Quarantine of Internet Worms[A]. Proceedings of the International Conference on Dependable Systems and Networks (DSN-2004)[C]. Florence, Italy, 2004.

[4] NOJIRI D, ROWE J, LEVITT K. Cooperative Response Strategies for Large Scale Attack Mitigation[A]. Proceedings of the 3rd DARPA Information Survivability Conference and Exposition[C]. 2003.

[5] OUDOT L. Fighting worms with honeypots: Honeyd vs Msblast. exe 2003[EB/OL]. http://ists. insecure. org/lists/honeypots/2003/Jul-Sep/0071. html, 2006.

[6] TOYOZUMI H, KARA A. Predators: Good Mobile Code Combat against Computer Viruses[A]. New Security Paradigms Workshop[C]. Virginia Beach, Virginia, 2002.

[7] NICOL D. Models of Active Worm Defense[A]. Proceedings of the Measurement, Modeling and Analysis of the Internet (IMA Workshop '04)[C]. Urbana-Champaign, Illinois, 2004.

[8] GOU XT, JIN WD. Multi-agent System for Security Auditing and Worm Containment in Metropolitan Area Autonomous Decentralized Systems[A]. ISADS 2005[C]. 2005. 201 - 207.

[9] YU M. Analyzing the performance of Internet worm attack approaches[A]. Proceedings of 13th International Conference on Computer Communications and Networks[C]. 2004. 501 - 506.

[10] HWANG FK, RICHARDS DS. The Steiner Tree Problems[J]. IEEE/ACM Transactions on Networking, 1999, 22: 55 - 89.

[11] ZHOU BD. Steiner Tree Optinization in Multicast Routing[D]. M. S. Thesis, University of Guelph, 2002.

[12] 卢开澄, 卢华明. 图论及其应用[M]. 北京, 清华大学出版社, 1995.

[13] CHRISTOFIDES N. Graph Theory: An Algorithmic Approach[M]. London: Academic Press, 1975. 79 - 120.

[14] MEHLHORN K. Data Structures and Algorithms 1: Sorting And Searching[M], Heidelberg: Springer-Verlag Berlin, 1984.

[15] Aglet workbench by IBM Japan research group[EB/OL]. http://www-trl. ibm. co. jp/aglets/, 2006.

(上接第 2930 页)

由图1(a)不难看出:随着 RSA 规模增大,改进后的 RSA 算法(算法4)的初始化耗费时间比标准 RSA 算法(算法1)要高,性能损失约 13%,但由于密码系统的初始化只有一次,一旦密码系统建立,在其后的应用中仅涉及到加密运算和解密运算,因此密码系统的加密和解密速度才是关键。比较图1(b),(c)中两种算法实现的加密和解密耗时情况容易发现:利用算法4实现的密码系统的加密性能约是算法1加密性能的6倍,而解密性能大约是算法1的10倍。因此,加入各种安全限制与要求,可以利用上述方法开发快速、实用的 RSA 密码系统,以满足计算机网络和电子商务等领域中的应用。

参考文献:

[1] DIFFIE W, HELLMAN ME. New Direction in Cryptography[J]. IEEE Transactions on Information Theory, 1976, IT - 22(6): 644 - 654.

[2] STINSON DR. 密码学原理与实践[M]. 冯登国译. 北京: 电子工业出版社, 2003.

[3] RIVEST RL, SHAMIR A, ADLEMAN LM. A Method for Obtaining Digital Signature and Public-Key Cryptosystems[J]. Communications of the ACM, 1978, 21(2): 120 - 126.

[4] RABIN MO. Digitalized Signatures and Public-Key Functions as Intractable as Factorization[M]. MIT Lab, For Computer Science, Cambridge, Mass, 1977.

[5] 冯登国, 裴定一. 密码学导引[M]. 北京: 科学出版社, 2001.

[6] STALLINGS W. 密码编码学与网络安全——原理与实践[M]. 刘玉珍, 等译. 北京: 电子工业出版社, 2004.

[7] 贺毅朝, 刘坤起. Rabin 密码算法的快速实现研究[J]. 计算机应用研究, 2006, 23(9): 51 - 53.

[8] 王晓东. 最短加法链算法[J]. 小型微型计算机系统, 2001, 10(20): 1250 - 1253.

[9] 柯召, 孙琦. 数论讲义[M]. 第2版. 北京: 高等教育出版社, 2003.

[10] KNUTH DE. The Art of Computer Programming: Seminumerical Algorithms, Volume 2[M]. 3rd edition. Addtion- Wesley, 2003.

[11] ROSE KH. Elementary Number Theory and Its Application[M]. Addison-Wesley, 1984.

[12] DOWNEY P, LEONY B, SETHI R. Computing Sequence with Addition Chains[J]. SIAM Journal of Computing, 1981, 10(3): 638 - 646.