PART 7

公钥密码学与RSA

公钥密码的基本概念

一个函数f：$A \to B$，若它满足：

1º 对所有$x \in A$，易于计算$f(x)$。

2º 对“几乎所有$x \in A$”，由$f(x)$求x“极为困难”，以至于实际上不可能做到。

则称f为一单向(One-way)函数。

定义中的“极为困难”是对现有的计算资源和算法而言。

陷门单向函数(Trapdoor one-way function)，是这样的单向函数：

- 在不知陷门信息下，由$f(x)$求x“极为困难”，；
- 当知道陷门信息后，由$f(x)$求x是易于实现的。

单向函数举例

给定一大素数p（比如，p在2^{1024}数量级），$p-1$含另一大素数因子。称$\log_2 p$为素数p的长度。

$\{1, 2, \ldots, p-1\}$关于mod p乘法构成了一乘群$\mathbf{Z}_p^*$，它是一个$p-1$阶循环群。该循环群的生成元一共有$\varphi(p-1)$个。

单向函数举例

设一个生成元为整数g，$1<g<p-1$，设一个整数x，$1<x<p-1$，设y满足$y=g^x \bmod p$。

已知x，g，p，求$y=g^x \bmod p$容易。

这是因为，采用折半相乘，只需要不超过$2\log_2 p$次的$\bmod p$乘法运算。

（实际上只需要不超过$2\log_2 x$次的$\bmod p$乘法运算。如

$$x=15=1111_2,$$

$$g^{15} \bmod p=(((g)^2 g)^2 g)^2 g \bmod p,$$

要用6次$\bmod p$乘法）

单向函数举例

若已知y，g，p，求x满足$y = g^x \bmod p$，称为**求解离散对数问题**。记为

$x=\log_g y \bmod p$。

求解离散对数问题的“最笨的方法”当然就是穷举，对每一个$x \in \{0, 1, 2, \ldots, p-1\}$检验是否$y=g^x \bmod p$。穷举求解法的运算次数约为$(p-1)/2$。

许多求解离散对数问题的算法比穷举快得多，比如Shanks算法，Pohlig-Hellman算法等。**最快求解法的运算次数约为数量级**

$$O(\exp(\sqrt{(\ln p)(\ln \ln p)}))$$

单向函数举例

这个计算量称为亚指数计算量。这是什么概念呢？我们知道p的长度是$\log_2 p$。看以下的不等式。

当$\log_2 p \approx 1024$时，亚指数计算量不小于2^{100}数量级。至少在在当前的计算水平之下是不能实现的。

$$\exp(\sqrt{(\ln p)(\ln\ln p)}) << \exp(\sqrt{(\ln p)(\ln p)})$$

$$= p = 2^{\log_2 p};$$

$$\exp(\sqrt{(\ln p)(\ln\ln p)}) >> \exp(\sqrt{(\ln\ln p)(\ln\ln p)})$$

$$= \ln p \sim \log_2 p.$$

公钥密码学

- 是密码学一次伟大的革命
- 使用两个密钥：公密钥、私密钥
- 加解密的非对称性
- 利用数论的方法
- 是对对称密码的重要补充

公钥密码学解决的基本问题

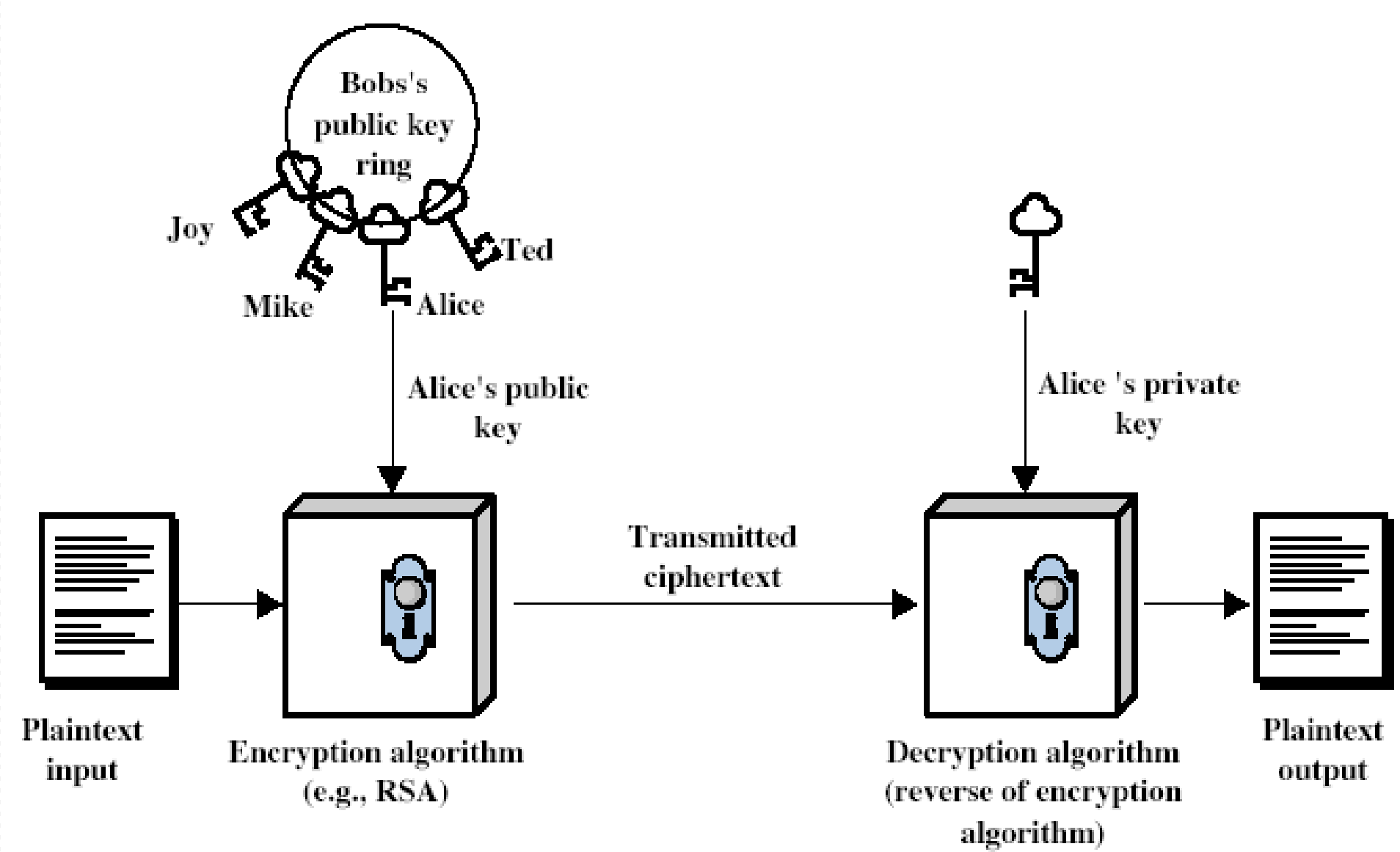
公钥密码学是为了解决传统密码中的密钥分配和数字签名而提出的

- 密钥分配（交换）
- 对称密码进行密钥分配（交换）的要求：
  - 已经共享一个密钥
  - 利用密钥分配中心
- 数字签名
  - 与传统手写的签名比较

公钥密码体制

- 重要特点
  - 仅根据密码算法和加密密钥来确定解密密钥在计算上不可行
  - 两个密钥中的任何一个都可用来加密，另一个用来解密。
- 六个组成部分：
  - 明文、密文；公钥、私钥；
  - 加密、解密算法

公钥密码体制


Bobs's public key ring
Joy
Ted
Mike
Alice
Alice's public key
Alice 's private key
Plaintext input
Encryption algorithm (e.g., RSA)
Transmitted ciphertext
Decryption algorithm (reverse of encryption algorithm)
Plaintext output

公钥密码体制的加密功能

- A向B发消息X，
- B的公钥为KUb，私钥为KRb
- 加密 $Y = E_{KUb}(X)$
- 解密 $X = D_{KRb}(Y)$

公钥密码体制的认证

- A向B发送消息X
- A的公钥为KUa，私钥为KRa
- “加密”： $Y = E_{KRa}(X)$ （数字签名）
- “解密”： $X = D_{KUa}(Y)$
- 思考：能保证消息的保密性吗？
- 请问：利用公钥密码体制，n个用户通信需要多少个密钥？

具有保密与认证的公钥体制

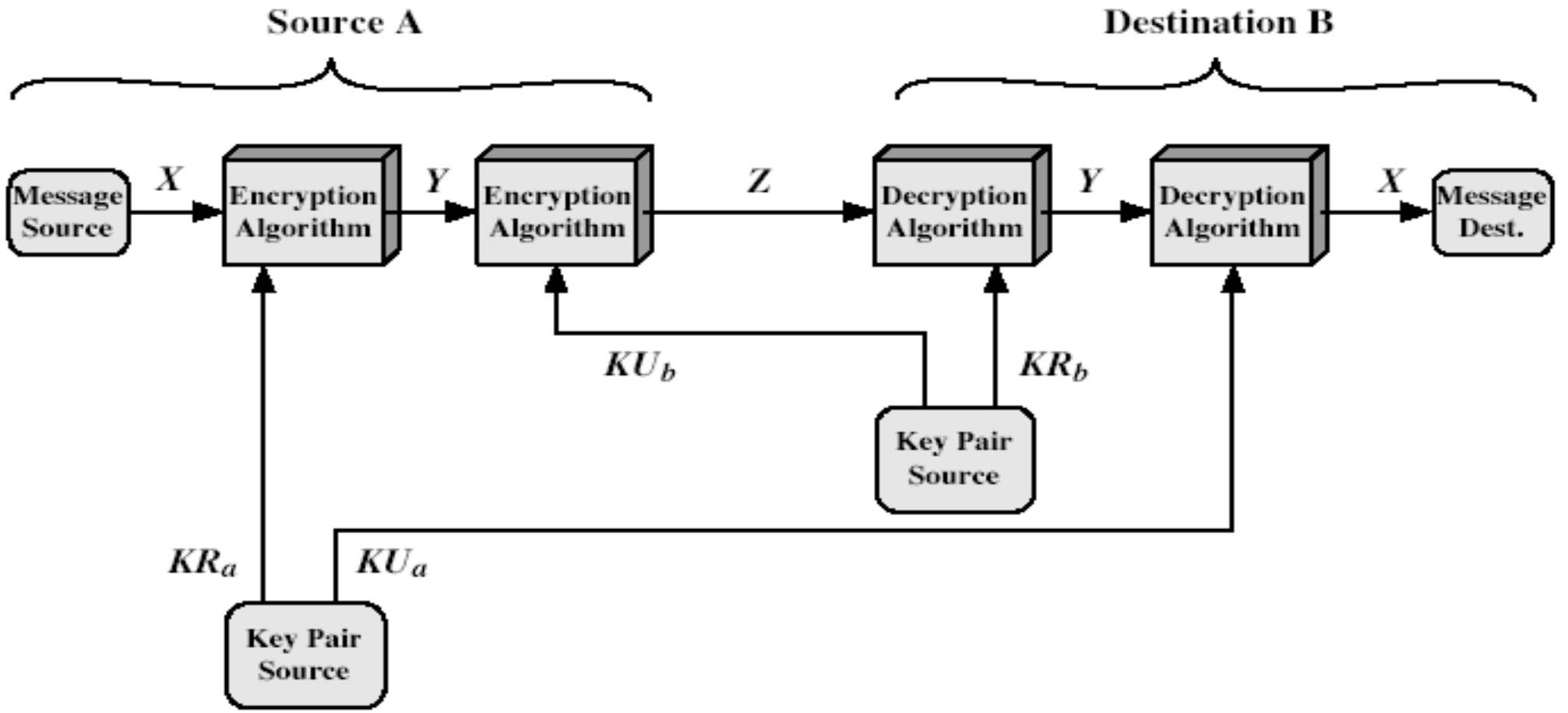


Figure 9.4 Public-Key Cryptosystem: Secrecy and Authentication

对称密码	公钥密码
一般要求： 1、加密解密用相同的密钥 2、收发双方必须共享密钥 安全性要求： 1、密钥必须保密 2、没有密钥，解密不可行 3、知道算法和若干密文不足以确定密钥	一般要求： 1、加密解密算法相同，但使用不同的密钥 2、发送方拥有加密或解密密钥，而接收方拥有另一个密钥 安全性要求： 1、两个密钥之一必须保密 2、无解密密钥，解密不可行 3、知道算法和其中一个密钥以及若干密文不能确定另一个密钥

关于公钥密码的几种误解

- 公钥密码比传统密码安全？
  - 事实上，任何加密方法的安全性依赖于密钥的长度和破译密文所需要的计算量。从抗密码分析的角度看，原则上不能说传统密码优于公钥密码，也不能说公钥密码优于传统密码。

关于公钥密码的几种误解

- 公钥密码是通用方法，所以传统密码已经过时？
  - 由于现有的公钥密码方法所需的计算量大，所以取缔传统密码似乎不太可能。

关于公钥密码的几种误解

- 公钥密码实现密钥分配非常简单？
  - 事实上，使用公钥密码也需要某种形式的协议，该协议通常包含一个中心代理，并且它所包含的处理过程既不必传统密码中的那些过程更简单，也不比之更有效。

关于公钥密码的几种误解

- 为什么要提出公钥密码体制？
  - 是为了解决传统密码中最困难的两个问题而提出的——密钥分配问题以及数字签名问题。

RSA算法

- 由MIT的 Rivest, Shamir & Adleman 在1977提出
- 最著名的且被广泛应用的公钥加密体制
- 明文、密文是0到n-1之间的整数，通常n的大小为1024位或309位十进制数

RSA算法描述

- 加密： $C=M^e \bmod N$, where $0 \le M < N$
- 解密： $M=C^d \bmod N$
- 公钥为（e，N）， 私钥为（d，N）
- 必须满足以下条件：
  - $M^{ed} = M \bmod N$
  - 计算M^e和C^d是比较容易的
  - 由e和n确定d是不可行的

RSA密钥产生过程

- 随机选择两个大素数 **p, q**
- 计算 **N=p·q**
  - 注意 **ø(N)=(p-1)(q-1)**
- 选择 **e使得1<e<ø(N),且gcd(e,ø(N))=1**
- 解下列方程求出 **d**
  - **e·d=1 mod ø(N)** 且 **0≤d≤N**
- 公布公钥: **KU={e,N}**
- 保存私钥: **KR={d,p,q}**

RSA的使用

- **发送方要加密明文M:**
  - **获得接收方的公钥 KU={e,N}**
  - **计算: $C=M^e \bmod N$, where $0 \le M < N$**
- **接收方解密密文C:**
  - **使用自己的私钥 KR={d,N}**
  - **计算: $M=C^d \bmod N$**
- **注意：M必须比N小**

为什么RSA可以加解密

- 因为 Euler 定理的一个推论:
  - $M^{k\phi(n)+1} = M \bmod N$
- RSA 中:
  - $N=p \cdot q$
  - $\phi(N)=(p-1)(q-1)$
  - 选择 e & d 使得$ed=1 \bmod \phi(N)$
  - 因此 存在k使得$e.d=1+k \cdot \phi(N)$
- 因此

$$C^d = (M^e)^d = M^{1+k.\phi(N)} = M \bmod N$$

RSA Example

1. 选择两个素数: p=17 & q=11

2. 计算 n = pq =17×11=187

3. 计算 ø(n)=(p－1)(q-1)=16×10=160

4. 选择e 使其gcd(e,160)=1，且e<160；这里选择 e=7

5. 确定 d: 使得de=1 mod 160 且d < 160

因为 23×7=161= 1×160+1 故取 d=23

6. 所得的公钥 KU={7,187}

7. 私钥 KR={23,187}

给定 M = 88

- 加密:

$$C = 88^{7} \bmod 187$$

$$=(88^{4} \bmod 187)(88^{2} \bmod 187)(88^{1} \bmod 187)$$

$$=11$$

- 解密:

$$M = 11^{23} \bmod 187 = 88$$

RSA密钥生成

- 必须做
  - 确定两个大素数： p，q
  - 选择e或者d，并计算d或者e
- 素数测试是重要的算法
- 由e求d要使用到扩展Euclid算法

RSA的安全性

三种攻击 RSA的方法：

- 强力穷举密钥
- 数学攻击 ：实质上是对两个素数乘积的分解
- 时间攻击：依赖解密算法的运行时间

因子分解问题

- 三种数学攻击方法
  - 分解 N=p·q, 因此可计算出ø(N)，从而确定d
  - 直接确定ø(N)，然后找到d
  - 直接确定d
- 由N确定ø(N)等价于因子分解

使用RSA几个注意点

- 计算能力的不断增强和因子分解算法的不断改进，给大密钥的使用造成威胁。
- 因此我们在选择RSA的密钥大小时应谨慎小心。在现阶段，密钥大小取在1024到2048位是合适的。

使用RSA几个注意点

- 除了要指定n的大小外，研究者还提出了其他一些限制条件，为了防止可以很容易地分解n，RSA算法的发明者建议p和q还应满足下列条件

1. p和q的长度应仅相差几位。这样对1024位的密钥而言，p和q都应约在10^{75}到10^{100}之间。
2. (p-1)和(q-1)都应有一个大的素因子。
3. gcd(p-1, q-1)应该比较小

另外，已经证明，若e<n, 且d<$n^{1/4}$, 则d很容易被确定

计时攻击

- 由密码分析家P.C.Kocher于1996年提出。被安全专家称为“有创意的攻击”
- 通过观察系统处理特定函数所花费的时间来寻找密文的信息。
- 定时攻击法的基本思想

要计算$f=a^b \bmod n$，其中b用二进制表示为

$$b=(b_k\cdots b_1 b_0)_2$$

即为k比特长度。

计时攻击

计算程序如下（c为临时变量）：

```
c ← 0, f ←1
for  i ← k down to 0 do
{
      c ←2 × c
      d ←(d×d ) mod n
      if  b_i=1 do  // 判断指数的某个比特
                  {
                   c ←c+1,
                   d ←(d×a)  mod n;  //进行模运算
                  }
                  return   d;
}
```

基本思想

- 计算程序
- 分析上述算法，如果指数的当前比特为1时，就要运算额外的模运算$(d \times a) \bmod n$，当然此时算法速度减慢。对于某些a和d，该模运算是非常慢的，而攻击者很容易掌握这些值。从而，攻击者通过观察系统处理上述函数所花费的时间来判断，当很慢时指数比特就是1，不然就是0，用上述原理可以得到整个指数。
- 由于上述攻击思想完全区别于以往的攻击方式，所以称其为“有创意的攻击”。

- 计时攻击不仅可以攻击RSA而且可以攻击其他的公钥密码系统
- 计时攻击的完全不可预知性以及它仅依赖于明文，所以它具有很大的威胁

其他方法

- 迭代攻击、选择明文攻击、公共模攻击、低加密指数攻击等。

- 消息隐匿问题

1. 对某些消息出现不动点问题。例如，对于明文x，使用RSA加密，可能出现$x^e=x \bmod n$，此时消息也就泄露了，称该现象为明文在RSA加密下的不动点。对RSA算法总有一些不动点，如$x=0,1,(n-1)$。

2. 一般来说，对RSA加密算法有：

$$[1+\gcd(e-1,\ p-1)]\times[1+\gcd(e-1,\ q-1)]$$

个不动点；

由于$e-1$，$p-1$，$q-1$都是偶数，所以不动点至少有

$$3\times3=9$$个

由于n非常大，所以一般情况下不动点可以忽略不计，一般系统可以不考虑。

如何避免计时攻击

尽管计时攻击会造成严重的威胁，但是有一些简单可行的解决办法，包括

1. 不变的幂运算时间

保证所有的幂运算在返回结果前执行的时间都相同，这种方法简单，但会降低算法的性能。

2. 随机延时

通过在求幂算法中加入随机延时来迷惑计时攻击者，可提高性能。

3. 隐蔽

在执行幂运算之前先将密文乘上一个随机数，这一过程可使攻击者不知道计算机正在处理的是密文的哪些位，这样可防止攻击者一位一位的进行分析，而这种分析正是计时攻击的本质所在。

- RSA数字签名应用较广，其在美国申请了专利但到2000年底保护期已经届满。
- 在标准化工作方面：在法国、澳大利亚等都先后采用了RSA并使之标准化；ISA9796也采用RSA算法。由于其专利保护过期，RSA算法在信息安全领域的应用将愈加广泛。


海南大学
Hainan university

对RSA的攻击－共模攻击

- 每一用户有相同的模数n
- 设用户的公开密钥分别为e_1,e_2,且e_1,e_2互素，明文消息为m,密文为

$$c_1 \equiv m^{e_1} \bmod n$$

$$c_2 \equiv m^{e_2} \bmod n$$

- 因为（e_1,e_2)=1,用欧几里德算法可求$re_1+se_2=1$假定r为负数，从而可知由Euclidean算法可计算
- $(c_1^{-1})^{-r} \cdot c_2^{s} = m \bmod n$

对RSA的攻击－低指数攻击

令网中三用户的加密钥e均选3，而有不同的模n_1，n_2，n_3，若有一用户将消息x传给三个用户的密文分别为

$y_1 = x^3 \bmod n_1 \quad x < n_1$

$y_2 = x^3 \bmod n_2 \quad x < n_2$

$y_3 = x^3 \bmod n_3 \quad x < n_3$

一般选n_1，n_2，n_3互素(否则，可求出公因子，而降低安全性)，利用中国余定理，可从y_1，y_2，y_3求出 $y = x^3 \bmod (n_1\ n_2\ n_3)$。 由$x < n_1$，$x < n_2$，$x < n_3$，可得$x^3 < n_1 \cdot n_2, \cdot n_3$，故有

$$\sqrt[3]{y} = x$$