1 传统密码不适合计算机网络时代

- 进入计算机网络时代, 传统密码逐渐暴露出其固有弱点: [1] 要求通信双方利用安全信道进行密钥 K 的预先通信, 在实际应用中是非常困难的.

  [2] 要求不同用户间约定不同的密钥, 若网络上有 n 个用户, 则需要

  $$C_n^2 = \frac{n(n-1)}{2}$$

  如果 $n = 1000$, 则 $C_{1000}^2 = 500000$. 这给密钥的管理和更换带来十分繁重的工作.

  [3] 每个用户必须记下与其他 n-1 个用户通信所用的密钥, 由于数量巨大, 只能记录在本子上或存储在计算机内存或外存上, 是极不安全的.

- 鉴于传统密码在密钥分配与管理上的困难等原因, DH 提出了公钥密码体制的思想.

2 公钥密码体制

- 公钥密码体制将加密密钥和解密密钥公开, 并将加密密钥公开, 解密密钥保密. 每个用户拥有两个密钥: 公开钥 (公钥) 和私秘钥 (私钥), 所有公开钥军备局路在类似电话本的密码本中.
- 公开密码思想: 每个用户有一个加密用的密钥 k_e, 还有一个解密用的密钥 k_d, 将 k_e 公开, k_d 保密, 且公开 k_e 不会影响 k_d 的安全. 可将个用户的 k_e 集中成公钥文件, 像电话本一样公开发给所有用户.
- 若 A 与 B 保密通信, 查公钥文件得到 B 的加密用公钥 $k_e^{(B)}$, 对信息加密如下:

  $$c = E_{k_e^{(B)}}(m)$$

  将 c 传给 B, B 用其私钥 $k_d^{(b)}$ 解密密文 c 得到明文 m:

  $$m = D_{k_d^{(B)}}(c)$$

- 公钥密码体制的安全性体现在: 即使得到公开钥和加密算法以及密文, 也无法求出明文和私秘钥
- 非对称密码

3 公钥密码体制原理

- E_B, D_B 满足以下三个条件:

  [1] D_B 是 E_B 的逆变换, 即 $\forall m \in M$明文空间, 均有:

  $$D_B(c) = D_B[E_B(m)] = m$$

  [2] 在已知 B 的公开钥和私秘钥的条件下, E_B 和 D_B 均是多项式时间的确定性算法.

  [3] 对 $\forall m \in M$, 找到算法 D_B^*, 使得 $D_B^*[E_B(m)] = m$ 是非常困难的

- 公钥密码体制无法提供无条件的安全性.

4 计算复杂性理论

- 计算复杂性理论是分析密码技术的计算要求和研究破译密码固有难度的基础.

4.1 算法复杂性

- 算法的计算复杂性使用该算法所需的计算时间 (T) 和存储空间 (S) 要求来度量的.
- 当给定一个特定问题的求解算法后, 执行这个算法的计算时间要求和存储空间要求也随之确定. 通常把计算时间和存储空间进行抽象, 以该实例所需输入数据的长度 n 的函数 $f(n)$ 来表示, 则 n 越大, 所需花费的时间代价和空间代价 $f(n)$ 也越大.
- $f(n)$ 通常表示为形如 $O[g(n)]$ 的" 量级" 函数, 并称为" 大 O" 表示法.
- $f(n) = O[g(n)]$ 表示存在满足下面要求的常数 c 和 n_0

  $$f(n) \leq c|g(n)|, 当n \geq n_0$$

4.2 问题复杂性和 NP 完全问题

1. P-问题

如果一个问题已经找到了一个多项式算法, 就称这个问题为一个 P-问题, 或者说属于 P 类.

2. 确定性或非确定性问题

   - 若一个算法中每一步计算中, 下一步是唯一的, 称该算法是确定性的.
   - 若一个算法, 在它的某一计算步骤必须从有限个可选项中选出一个作为下一步, 则称该算法是非确定性的.
   - 属于 P-类的算法都是确定性的, 且执行时间是多项式时间.

3. NP-问题

   - NP-问题即不确定性多项式类, 指用非确定性算法在多项式时间内可以解决的问题.

4. NP 完全问题

   - NP 完全问题 (或称 NPC 问题) 即不确定性多项式类, 指用非确定性算法不能在多项式时间内解决的问题
   - 设 x 是一个给定的问题, 如果对于 $\forall x' \in NP$, 均有 $x' \in x$, 则称 x 是困难问题, 且 $x \in NP$, 则 x 称为 NP 完全问题或 NPC 问题
   - NPC 问题是 NP 问题中最困难的问题.
   - 目前有数百个问题被证明是 NP 完全问题. NP 完全问题的发现为人们提供了一种判定一种问题是" 难问题" 的标准.
   - 结论: 在现代密码中, 一个密码系统的破译常常可以归结为求解某个数学问题, 数学问题的算法求解复杂性可通过计算复杂性理论来描述.

     [1] 计算复杂性理论为破译密码的计算复杂都提供了实际的度量方法 [2] 计算复杂性理论中的一些典型的数学问题给人们提供了设计实用安全的高强度密码系统的基础 (例如背包公开钥密码系统, RSA 公开钥密码系统)

5 数论

5.1 欧几里得算法

- 欧几里得算法即" 辗转相除法". 利用该算法可以迅速地找出给定的两个整数 a 和 b 地最大公因数 (a,b), 并判断 a 和 b 是否互素.
- 当 $(a,b)=1$ 时, 利用欧几里得算法可以找到两个常数 u 和 v, 使得

$$ua+vb=1$$

可得 $ua\equiv 1(\mathrm{mod}b), vb\equiv 1(\mathrm{mod}a)$ 于是 u 是 a 模 b 的乘法逆, v 是 b 模 a 的乘法逆.

- 在 IDEA 的密钥变换过程中, 多次用到求 $16bit$ 密钥 Z 在乘法 ($\odot$) 下的逆元 Z^{-1}

6 RSA 公钥密码系统

- 1977 年, RSA 联合提出一种欧拉定理的公钥密码系统, 简称为 RSA 公钥密码系统, 他的安全性是基于大数因子分解这一困难问题.

6.1 RSA 加密算法

一、RSA 体制用户 i 的公开加密变换 E_i 与保密的解密变换 D_i 的生成.

[1] 随机选取两个 100 个位 (指十进制) 以上的素数 p_i 和 q_i.

[2] 计算 $n_i=p_iq_i, \Phi(n_i)=(p_i-1)(q_i-1)$

[3] 随机选取整数 e_i, 满足 $(e_i,\Phi(n_i))=1$

[4] 利用欧几里得算法计算 d_i 满足 $e_id_i=1[\mathrm{mod}\Phi(n_i)]$.

[5] 公布 n_i,e_i 作为 E_i, 记为 $E_i=<n_i,e_i>$. 保密 $p_i,q_i,d_i,\Phi(n_i)$ 作为 D_i, 记为 $D_i=<p_i,q_i,d_i,\Phi(n_i)>$.

加密算法: $c=E_i(m)=m^{e_i}(\mathrm{mod}n_i)$ 解密算法: $m=D_i(c)=c^{d_i}(\mathrm{mod}\ n_i)$

7 RSA 体制的加解密过程

- RSA 体制为分组密码体制, 第一步需将明文数字化, 用户 i 的明文字母表和密文字母表均取 $Z_{n_i} = \{0, 1, \cdots, n_{i-1}\}$.

  1. 加密过程

     – 设用户 j 欲将明文加密后传给用户 i, 用户 j 实施下列各步:

       [1] 在公开钥数据库中查得用户 i 的公开钥 $E_i =< n_i, e_i >$

       [2] 将 m 分组为 $m = m_1 m_2 \cdots m_r, m_a \in Z_{n_i}, a = 1, 2, \cdots, r.$

       [3] 对每一个分组作加密变换, 即对 $a = 1, 2 \cdots, r$ 作

$$c_a = E_i(m_a) \equiv m_a^{e_i} (\mathrm{mod} n_i)$$

       [4] 将密文 $c = c_1 c_2 \cdots c_r$ 传给用户 i.

  2. 解密过程

     – 设用户 i 收到密文 $c = c_1 c_2 \cdots c_r$ 后

       [1] 对每一分组密文做解密变换, 即对 $a = 1, 2, \cdots, r$, 作

$$m_a = D_i(c_a) = c_a^{d_i} (\mathrm{mod} n_i)$$

       [2] 合并分组, 得到 $m = m_1 m_2 \cdots m_r$.

7.1 模求幂运算

- 模求幂运算即进行形式为 $x^c (\mathrm{mod} n)$ 的函数计算
- 求 $x^c (mod n)$ 的计算可通过对指数 c 的二进制化来表示.

  方法一:

  假设 $x^c (\mathrm{mod} n)$ 中的指数 c 以二进制形式表示为:

$$c = (c_l c_{l-1} \cdots c_1 c_0)_2 = \sum_{i=0}^{l} c_i 2^i = \sum_{c_i \neq 0} 2^i; c_i = 0; 0 \leq i \leq l$$

  则

$$x^c (\mathrm{mod} n) \equiv x^{\sum\limits_{c_i \neq 0} 2^i} (\mathrm{mod} n) = \prod_{c_i \neq 0} x^{2^i} (\mathrm{mod} n)$$

例如: 计算 $16^7(\mathrm{mod}4731)$

解: 因为 $7=(111)_2$

$$i=0,16^{2^0}=16^1\equiv 16(\mathrm{mod}4731)$$
$$i=1,16^{2^1}=16^2\equiv 256(\mathrm{mod}4731)$$
$$i=2,16^{2^2}=16^4\equiv 65536(\mathrm{mod}4731)=4033(\mathrm{mod}4731)$$

所以

$$\begin{aligned}16^7(\mathrm{mod}4731)&\equiv(16\times 256\times 4033)(\mathrm{mod}4731)\\&=16519168(\mathrm{mod}4731)\\&=3427(\mathrm{mod}4731)\end{aligned}$$

方法二:

假设 $x^c(\mathrm{mod}n)$ 中的指数 c 以二进制形式表示为:

$$c=\sum_{i=0}^{l-1}c_i2^i$$

其中 $c_i=0,1,0\le i\le l-1$

计算 $z=x^c(\mathrm{mod}n)$ 的算法如下:

[1] z=1,

[2] for i=l-1, down to 0 do

[3] $z=z^2(\mathrm{mod}n)$,

[4] if $c_i=1$ 则 $z=z^2\times x(\mathrm{mod}n)$

[5] if $c_i=0$ 则 $z=z^2(\mathrm{mod}n)$

例如: 计算 $1520^{13}(\mathrm{mod}2537)$

解: 此例中 $x=1520,c=13,n=2537$, 因为

$13=(1101)_2=1\times 2^3+1\times 2^2+0\times 2^1+1\times 2^0$

$i=3,c_3=1,z=z^2\times x=1^2\times 1520(\mathrm{mod}2537)$

$i=2,c_2=1,z=z^2\times x=1520^2\times 1520(\mathrm{mod}2537)\equiv 1649(\mathrm{mod}2537)$

$i=1,c_1=0,z=z^2=1649^2(\mathrm{mod}2537)=2074(\mathrm{mod}2537)$

$i=0,c_0=1,z=z^2\times x=2074^2\times 1520(\mathrm{mod}2537)=1285(\mathrm{mod}2537)$

所以, $1520^{13}(\mathrm{mod}2537)\equiv 1285(\mathrm{mod}2537)$

7.2 RSA 加密举例

[1] 假设用户 i 随机选择两个十进制大素数为 $p_i = 43, q_=59$

[2] 计算 $n_i = p_i q_i = 43 \times 59 = 2537$

$\Phi(n_i) = (p_i - 1) \times (q_i - 1) = 42 \times 58 = 2436$

[3] 随机选取整数 $e_i = 13$, 满足 $(e_i, \Phi(n_i)) = (13, 2436) = 1$

[4] 利用欧几里得算法计算 d_i, 满足 $e_i d_i = 1 \bmod \Phi(n_i) = 1(\mathrm{mod}2436)$

辗转相除, 得:

$$\begin{aligned} 1 &= 3 - 2 = 3 - (5 - 3) = 2 \times 3 - 5 \\ &= 2 \times (13 - 2 \times 5) - 5 = 2 \times 135 \times 5 \\ &= 2 \times 13 - 5 \times (2436 - 187 \times 13) \\ &= -5 \times 2436 + 937 \times 13 \end{aligned}$$

$$937 \times 13 \equiv 1(\mathrm{mod}2436)$$

$$d_i = 937$$

pu bl ic ke ye nc ry pt io ns

表 1: 利用下表将明文数字化

明文	a	b	c	d	e	f	g	h	i	j	k	l	m
数字	0	1	2	3	4	5	6	7	8	9	10	11	12
明文	n	o	p	q	r	s	t	u	v	w	x	y	z
数字	13	14	15	16	17	18	19	20	21	22	23	24	25

对明文块数字化得结果如下: 1520 0111 0802 1004 2404 1302 1724 1519 0814 1318

[3] 加密变换

- $1520^{13}(\mathrm{mod}2537) \equiv 1285(\mathrm{mod}2537)$

$$13 = (1101)_2 = 1 \times 2^3 + 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0$$

$$i = 3, c_3 = 1, z = z^2 \times x = 1^2 \times 1520 = 1520(\mathrm{mod}2537)$$

$$i = 2, c_2 = 1, z = z^2 \times x = 1520^2 \times 1520 = 1649(\mathrm{mod}2537)$$

$$i = 1, c_1 = 0, z = z^2 = 1649^2 = 2074(\mathrm{mod}2537)$$

$$i = 0, c_0 = 1, z = z^2 \times x = 2074^2 \times 1520 = 1285(\mathrm{mod}2537)$$

- $111^{13}(\mathrm{mod}2537) \equiv 1648(\mathrm{mod}2537)$
- $802^{13}(\mathrm{mod}2537) \equiv 1410(\mathrm{mod}2537)$

  ……

- $1519^{13}(\mathrm{mod}2537) \equiv 2132(\mathrm{mod}2537)$
- $814^{13}(\mathrm{mod}2537) \equiv 1751(\mathrm{mod}2537)$
- $1418^{13}(\mathrm{mod}2537) \equiv 1289(\mathrm{mod}2537)$

8 RSA 的安全性

- 若 $n = pq$ 被因数分解, 则 RSA 便被攻破

  若 p,q 已知, 则 $\Phi(n) = (p-1)(q-1)$ 可以算出, 解密密钥 d 可用欧几里得算法求出.

- RSA 的安全性依赖于因数分解的困难性, 时间复杂性为 $e^{\sqrt{\ln n \cdot \ln(\ln n)}}$
- 基于对 RSA 系统安全性的考虑, 在设计 RSA 系统时, p,q 应满足:

  [1] p,q 要足够大, 一般因该在 $10^{100} \sim 10^{125}$ 之间, 这样才可以基本保证不会再有效时间内被密码分析人员破译出参数.

  [2] 如果 $p > q$, 则差值 $p - q$不宜太小, 最好是与 p,q 的位数接近

  如果 p 和 q 的数值相当接近, 则 $\frac{1}{2}(p+q) \approx \sqrt{n}$, 并且 $\frac{1}{2}(p-q)$ 是一个相当小的数, 因此等式 $(\frac{p+q}{2})^2 - n = (\frac{p-q}{2})^2$ 的右端是一个相当小的数, 可以利用 Fermat 因数分解将 n 分解因数.

  [3] p-1 和 q-1 的最大公约数 d=((p-1), (q-1)) 应尽量小. 否则, 将有 d^2 个整数 b, 使得 n 对 b 是伪素数, 这就增大了将 n 因数分解的可能性.

  [4] p-1 和 q-1 都应该至少含有一个大的素因子, p+1 与 q+1 也应该至少含有一个大的素因子, 否则就可能利用 p-1 和 p+1 方法求出 n 的真因数.

- 对 RSA 是否存在一种不依赖于分解 n 破译方法? 虽未发现, 但没有严格的讨论. 值得注意的是存在这样一种例子: $m = m_{k-1}^{e}(\mathrm{mod}n)$. 例如: 取 $p = 17, q = 11$, 则 $n = pq = 187$, 取 $e = 7$, 明文 $m = 123$. 可证明

经过 RSA 连续加密变换可得到 $m_4 = m$, 即连续 4 次进行 RSA 变换后又恢复到原文.

$$m_0 = m = 123, 123^7 \equiv 183(\mathrm{mod}187),$$
$$m_0 = m = 183, 183^7 \equiv 72(\mathrm{mod}187),$$
$$m_0 = m = 72, 72^7 \equiv 30(\mathrm{mod}187),$$
$$m_0 = m = 30, 30^7 \equiv 123(\mathrm{mod}187),$$
$$m_0 = m = 123, 123^7 \equiv 183(\mathrm{mod}187).$$

- 这是 RSA 公钥密码体制存在的一种潜在弱点.

8.1 RSA 公钥密码体制的弱点

- 对 RSA 公钥密码体制的攻击, 一般欲从分解模 n 而获得解密密钥很难奏效, 但这并不意味者密码分析者不能通过其他途径破译该体制
- 美国学者 Simmons 首先提出了破译方案的一种途径, 其描述如下:
  - RSA 公钥密码体制利用公开钥 $[e, n]$ 对明文进行加密
  $$m^e \equiv c(\mathrm{mod}n)$$
  - 密码分析者可能从不保密的信道上截获到密文 c, 但因为不知道私秘钥 d, 故不能按下式译成明文 m
  $$c^d \equiv m(\mathrm{mod}n)$$
  - 此时可对密文 c 利用公开钥 e 依次进行如下变换
  $$c^e \equiv c_1(\mathrm{mod}n)$$
  $$c_1^e \equiv c_2(\mathrm{mod}n)$$
  $$\cdots\cdots$$
  $$c_{\beta-1}^e \equiv c_\beta(\mathrm{mod}n)$$
  $$c_\beta^e \equiv c_{\beta+1} \equiv c(\mathrm{mod}n)$$
  - 即当上述模求幂运算依次迭代 $\beta + 1$ 后, 其运算结果恰好等于密文 c. 即可以知道第 β 步的计算结果 c_β 就是明文 m. 这种破译方法的理论根据是基于下述定理.

定理 8.1 设 n 为素数或不同素数之积, 如果 $(e, \Phi(n)) = 1$, 则存在正整数 h

$$a^{e^h} \equiv a(\mathrm{mod} n)$$

例如设 RSA 公钥密码体制的一组参数为 $p = 383, q = 563, n = p \times q = 215629, e = 49, d = 56957$. 加密、解密变换式为

$$m^{49} \equiv c(\mathrm{mod} 215629)$$
$$c^{56957 \equiv m(\ \mathrm{mod}\ 215629)}$$

设明文信息为 $m = 123456$, 由于

$$(123456)^{49} = 1603(\mathrm{mod} 215629)$$

密码分析者利用公钥 $< 49, 215629 >$ 和密文 $c = 1603$, 按公式 (3) 运算得到

$$c^e =\equiv c_1(\mathrm{mod} n), c_1 = 180661$$
$$c_1^e =\equiv c_2(\mathrm{mod} n), c_2 = 109265$$
$$c_2^e =\equiv c_3(\mathrm{mod} n), c_3 = 131172$$
$$c_3^e =\equiv c_4(\mathrm{mod} n), c_4 = 98178$$
$$c_4^e =\equiv c_5(\mathrm{mod} n), c_5 = 56372$$
$$c_5^e =\equiv c_6(\mathrm{mod} n), c_6 = 63846$$
$$c_6^e =\equiv c_7(\mathrm{mod} n), c_7 = 146799$$
$$c_7^e =\equiv c_8(\mathrm{mod} n), c_8 = 85978$$
$$c_8^e =\equiv c_9(\mathrm{mod} n), c_9 = 123456$$
$$c_9^e =\equiv c_10(\mathrm{mod} n), c_10 = 1603$$

仅仅经过 10 步变换就得到 $c_{10} = c$, 即 $c_9 = m = 123456$ 这是因为 $49^{10} \equiv 1(\mathrm{mod} 215629)$

- 在此例中周期等于 10a
- 为使的上述破译方法不能成为实际, 必须合理选择 RSA 公钥密码体制的有关参数, 以保证 β 足够大.
- Rivest 等人提出, 当 $(p-1)$ 和 $(q-1)$ 不具有小素因数时, 可以迫使周期 β 足够大, 使得 RSA 体制在实际上仍是不可破译的. 具体在选择素数 p 和 q 时可按强素数的定义选取.

9 大素数的产生和因数分解

9.1 大素数的产生

- 构造 RSA 公钥密码体制时, 选取大素数 p 和 q 是非常关键的.
- 大素数的产生及测试使密码学领域中的一个重要课题.
- 产生大素数的方法可以分为两大类:

  1. 确定性素数产生法

     [1] 基于 P 定理的确定性素数产生方法—该方法需要已知 $n-1$ 的部分素因子.

     [2] 基于 Lucas 定理的确定性素数产生方法—该方法需要已知 $n-1$ 的全部素因子

     – 优点: 该方法产生的一定是素数
     – 缺点: 产生的素数带有一定的限制. 即如果算法设计不当, 构造的素数容易产生规律性, 使密码分析人员较容易地追踪素数的变化, 直接猜测到 RSA 系统所使用的素数.

  2. 概率性素数产生方法

     – 算法较多, 是当今产生大素数的主要算法
     – 在实际应用中的方法是: 先产生大的随机数, 然后利用 S-S 或者 M-R 算法进行素数测试
     – 缺点: 该方法产生的不一定是素数, 即产生的数是伪素数, 虽然是合数的可能性很小, 但这种可能性依然存在.
     – 优点: 产生的伪素数的速度较快, 构造的伪素数无规律性
     在构造 RSA 体制中的大素数时, 可将上述两种方法结合起来:
     [1] 首先利用概率性素数产生方法产生伪素数;
     [2] 然后再利用确定性素数产生方法对所产生的伪素数进行检验

9.1.1 因数分解

- 因数分解 $n = p \times q$ 是对 RSA 公钥密码体制进行有效攻击的方法, 但因数分解比大素数的产生更为困难.

- 随着对 RSA 公钥密码体制研究的深入, 大整数因数分解
- 因数分解方面的重大事件如下:

  [1] 1983 年, DHS 利用二次筛选法成功分解了 69 为十进制数

  [2] 1989 年, LM 利用二次筛选法通过把计算分配给数百台工作站实现了 106 位十进制的因数分解

  [3] 1990 年, LM 和 P 利用数域筛选法动用 700 台工作站, 分解了 155 位十进制数的一个特例
- 为保证 RSA 公钥密码体制的安全性, n 应大于 200 位十进制数
- 常用的因数分解法有:

  [1] Fermat 因数分解法

  [2] 连分数因数分解发

  [3] 圆锥曲线分解整数

  [4] P-1 方法

10 背包公钥密码系统

在 DH 提出公钥密码设想两年后, MH 提出了一种基于组合数学中背包问题的公钥密码系统, 称为 MH 背包公钥密码系统

10.1 背包问题

- 背包问题是著名的数学难题, 至今尚无有效的求解方法: 因为对 2^n 中可能进行穷举搜索, 在目前条件下, 不可能在实际允许的时间内完成
- 背包问题属于 NP 完全问题
- 背包问题的描述如下

10.1.1 背包公钥密码

- 取 $a = (a_1, a_2, \cdots, a_n)$ 作为加密密钥公开, 其中 $a_1, a_2, \cdots, a_n$ 为整数
- 明文 $m = m_1 m_2 \cdots m_n$, 是 n 位 0,1 字符串

- 利用公钥加密如下: $c = a_1m_1 + a_2m_2 + \cdots a_nm_n$
- 从密文求明文等价于背包问题

  例如: 若 $a = (28, 32, 11, 8, 71, 51, 43), m = 0100101$

  则 $c = 32 + 71 + 43 = 146$

  即得到密文 $c = 146$

  第三者无法从密文推知明文 $m_1m_2 \cdots m_7$.

10.1.2 超递增序列

- 并非所有的背包问题都没有有效的求解算法
- 若序列 $a = 1_1a_2 \cdots a_n$ 满足条件:

$$a_i > \sum_{j=1}^{i-1} a_j, i = 1, 2, 3, \cdots, n$$

  则称这样的序列为超递增序列

- 一般来说, 超递增序列的解:

  对满足不等式 $a_i > \sum\limits_{j=1}^{i-1}, i = 1, 2, 3 \cdots, n$ 的方程

$$a_1x_1 + a_2x_2 + \cdots + a_nx_n = b$$

  其解为

$$\begin{cases} 0, 若b \leq \sum\limits_{j=1}^{n-1} a_j \\ 1, 若b > \sum\limits_{j-1}^{n-1} a_j \end{cases} \quad i = n, n-1, \cdots, 2$$

  例如: 设 $a = (2, 4, 7, 15, 30)$, 求解 $2x_1 + 4x_2 + 7x_3 + 15x_4 + 30x_5 = 39$

  解: 因为 $2 + 4 + 7 + 15 = 28 < 39$, 所以

$$x_5 = 1, 2x_1 + 4x_2 + 7x_3 + 15x_4 = 39 - 30 = 9$$

  因为 $2 + 4 + 7 = 13 > 9$, 所以

$$x_4 = 0, 2x_1 + 4x_2 + 7x_3 = 9 - 0 = 9$$

因为 $2+4=6<9$, 所以

$$x_3 = 1, 2x_1 + 4x_2 = 9 - 7 = 2$$

因为 $2=2$, 所以

$$x_2 = 0, 2x_1 = 2, x_1 = 1$$

- 不能将超递增序列本身作为公开密钥
- 一般的做法是将超递增序列转换为复杂序列, 然后作为公开钥公开, 第三者不知道变换及其逆变换, 故不能在多项式时间内破译出明文.

11 超递增序列的变换

设 $(a_1, a_2, \cdots, a_n)$ 是超递增序列, 取整数 w 和 m, 其中

$$m > 2a_n, (w, m) = 1, w\overline{w} \equiv 1(\mathrm{mod}m)$$

$\overline{w}$ 为模 m 的逆. 做变换 $b_i \equiv wa_i(\mathrm{mod}m), i = 1, 2\cdots, n$, 可得到一个序列 $(b_1, b_2, \cdots, b_n)$. 一般该序列不再是超递增序列, 将 $(b_1, b_2, \cdots, b_n)$ 公开.

对 $\sum_{i=1}^{n} b_i x_i = b$ 两边同时乘上 $\overline{w}$, 得到

$$\sum_{i=1}^{n} \overline{w} b_i x_i \equiv \sum_{i=1}^{n} \equiv \overline{w} b \equiv b_0(\mathrm{mod}m)$$

因为

$$b_0 < m, \sum_{i=1}^{n} a_i < m$$

所以

$$\sum_{i=1}^{n} a_i x_i = b_0$$

容易解密

11.1 背包密码体制加密和解密举例

例如假设 $a = (2, 5, 9, 21, 45, 103, 215, 450, 946)$ 是一个秘密的超递增序列, 取 $m' = 2003, w = 1289$

1. 计算公开钥由 $b_i \equiv wa_i(\mathrm{mod}m')$, 得到

$$b_1 = 1289 \times 2 \equiv 575(\mathrm{mod}2003)$$
$$b_2 = 1289 \times 5 \equiv 436(\mathrm{mod}2003)$$
$$b_3 = 1289 \times 9 \equiv 1586(\mathrm{mod}2003)$$
$$b_4 = 1289 \times 21 \equiv 1030(\mathrm{mod}2003)$$
$$b_5 = 1289 \times 45 \equiv 1921(\mathrm{mod}2003)$$
$$b_6 = 1289 \times 103 \equiv 569(\mathrm{mod}2003)$$
$$b_7 = 1289 \times 215 \equiv 721(\mathrm{mod}2003)$$
$$b_8 = 1289 \times 450 \equiv 1183(\mathrm{mod}2003)$$
$$b_9 = 1289 \times 946 \equiv 1570(\mathrm{mod}2003)$$

即将 $(575, 436, 1586, 1030, 1921, 569, 721, 1183, 1570)$ 公开作为加密钥.

2. 用户 A 对明文 $m = 101100111$ 加密, 得 b

$$b = b_1m_1 + b_2m_2 + b_3m_3 + b_4m_4 + m_5m_5 + b_6m_6 + b_7m_7 + b_8m_8 + b_9m_9$$
$$= 575 + 1586 + 1030 + 721 + 1183 + 1570$$
$$= 6665$$

3. 用户 B 收到密文 b 后, 需要进行以下工作才能回复明文 m:

[1] 利用欧几里得算法求解 w^{-1}

由 $w^{-1}w \equiv 1(\mathrm{mod}m')$ 及 $m' = 2003, w = 1289$ 得到

$$w^{-1}1289 \equiv 1(\mathrm{mod}2003)$$
$$w^{-1} \equiv 317(\mathrm{mod}2003)$$

[2] 计算 b_0, 即 c 由 $w^{-1}b \equiv b_0(\mathrm{mod}m')$

得到 $b_0 \equiv 317 \times 6665 \equiv 1643(\mathrm{mod}2003)$

[3] 恢复原来的超递增序列得到

由 $a_i \equiv w^{-1}b_i(\mathrm{mod}m')$, 得到

$$a_1 \equiv w^{-1}b_1 \equiv 317 \times 575 \equiv 2(\mathrm{mod}2003)$$
$$a_2 \equiv w^{-1}b_2 \equiv 317 \times 436 \equiv 5(\mathrm{mod}2003)$$
$$a_3 \equiv w^{-1}b_3 \equiv 317 \times 1586 \equiv 9(\mathrm{mod}2003)$$
$$a_4 \equiv w^{-1}b_4 \equiv 317 \times 1030 \equiv 21(\mathrm{mod}2003)$$
$$a_5 \equiv w^{-1}b_5 \equiv 317 \times 1921 \equiv 45(\mathrm{mod}2003)$$
$$a_6 \equiv w^{-1}b_6 \equiv 317 \times 569 \equiv 103(\mathrm{mod}2003)$$
$$a_7 \equiv w^{-1}b_7 \equiv 317 \times 721 \equiv 215(\mathrm{mod}2003)$$
$$a_8 \equiv w^{-1}b_8 \equiv 317 \times 1183 \equiv 450(\mathrm{mod}2003)$$
$$a_9 \equiv w^{-1}b_9 \equiv 317 \times 1570 \equiv 946(\mathrm{mod}2003)$$

[4] 有超递增序列 a 求解下列方程组

$$2x_1+5x_2+9x_3+21x_4+45x_5+103x_6+215x_7+450x_8+946x_9 = 1643$$

因为 $2+5+9+21+45+103+215+450 = 850 < 1643$, 所以

$$x_9 = 1, 2x_1+5x_2+9x_3+21x_4+45x_5+103x_6+215x_7+450x_8 = 1643-946 = 697$$

因为 $2+5+9+21+45+103+215 = 400 < 697$, 所以

$$x_8 = 1, 2x_1+5x_2+9x_3+21x_4+45x_5+103x_6+215x_7 = 697-450 = 247$$

因为 $2+5+9+21+45+103 = 185 < 247$, 所以

$$x_7 = 1, 2x_1+5x_2+9x_3+21x_4+45x_5+103x_6 = 247-215 = 32$$

因为 $2+5+9+21+45 = 82 > 32$, 所以

$$x_6 = 0, 2x_1+5x_2+9x_3+21x_4+45x_5 = 32$$

因为 $2+5+9+21 = 37 > 32$, 所以

$$x_5 = 0, 2x_1+5x_2+9x_3+21x_4 = 32$$

因为 $2+5+9 = 16 < 32$, 所以

$$x_4 = 1, 2x_1+5x_2+9x_3 = 32-21 = 11$$

因为 $2+5=7<11$, 所以

$$x_3=1, 2x_1+5x_2=11-9=2$$

因为 $2=2$, 所以

$$x_2=0, 2x_1=2, x_1=1$$

[5] 明文为 $m=101100111$

11.2 背包密码系统工作原理

[1] 每个用户选择一个长为 n 的超递增序列 $(a_1, a_2, \cdots, a_n)$.

[2] 每个用户选用整数 m 和 w, 其中 $m>2a_n, (w, m)=1$

[3] 做变换 $b_i \equiv wa_i(\mathrm{mod}m), i=1, 2, \cdots, n$.

[4] 将 $(b_1, b_2, b_3, \cdots, b_n)$ 公开.

[5] 欲传送明文 m, 先将明文 m 的字母转换成 0,1 符号串, 并将所得到的符号串分裂成长为 n 的块. 如果最末块长不足 n, 可用短块加密技术处理.

[6] 对一个明文块 $m_1, m_2, m_3, \cdots, m_n$, 给出对应的密文

$$c=b_1m_1+b_2m_2\cdots b_nm_n$$

可见若不知道 m 和 w, 则解密问题将导致一个背包问题.

11.3 Rabin 公钥密码

对 RSA 公钥密码体制的攻击, 若采用分解 n 的方法, 其难度不超过大数分; 若从 $\Phi(n)$ 入手进行攻击, 其难度与分解 n 相当.

- Rabin 提出了一种对 RSA 的修正方案, 其特点是:

  [1] 该方法不是一一对应的单向陷门函数为基础的, 因此, 对应于同一段密文, 有两个以上可能的明文, 这种不确定性, 增强了密码分析的困难.

  [2] 破译 Rabin 密文等价于对大整数 n 的分解.

- RSA 密码体制的加密密钥 e 的选择需满足 $1<e<\Phi(n)$, 且 $[e, \Phi(n)]=1$.

- Rabin 密码体制的加密算法是:

$$c \equiv m^2 (\mathrm{mod} n)$$

更一般的是

$c \equiv m(m+b)(\mathrm{mod} n)$, b 为 $0 \leq b < n$ 的整数.

- 其中, m 是明文对应的数据, c 是密文对应的数据, m, c 都属于 Z_n.
  - 与 RSA 密码体制一样, Rabin 密码体制也是分组密码体制
  - 式中: $n = pq$, p, q 为奇素数
  - 在 Rabin 密码体制中为了容易实现, 一般取 $p \equiv q \equiv 3(\mathrm{mod} 4)$, 其余对 Rabin 密码体制的要求与 RSA 密码体制一样
- 解密算法

已知 c 求 m, 导致解:

$$m^2 \equiv c(\mathrm{mod} n)$$

又

$$n = pq$$

亦即

$$\begin{cases} x^2 \equiv c(\mathrm{mod} p) \\ x^2 \equiv c(\mathrm{mod} q) \end{cases}$$

由于

$$(\frac{a}{p}) = (\frac{a}{q}) = 1$$

所以对每一个都将有两个解, 即

$$x \equiv m(\mathrm{mod} p), x \equiv -m(\mathrm{mod} p)$$
$$x \equiv m(\mathrm{mod} q), x \equiv -m(\mathrm{mod} q)$$

经过组合可得到 4 个同余方程组

$$\begin{cases} x \equiv m(\mathrm{mod} p) \\ x \equiv m(\mathrm{mod} q) \end{cases}, \begin{cases} x \equiv m(\mathrm{mod} p) \\ x \equiv -m(\mathrm{mod} q) \end{cases}$$
$$\begin{cases} x \equiv -m(\mathrm{mod} p) \\ x \equiv m(\mathrm{mod} q) \end{cases}, \begin{cases} x \equiv -m(\mathrm{mod} p) \\ x \equiv -m(\mathrm{mod} q) \end{cases}$$

- 即 Rabin 密码的已知密文对应的明文不唯一, 有 4 种可能的结果, 其中一个有确切意义才是所求的明文.
- 可以用某种方式在明文后面不加记为数作为识别码, 用它们从四个可能的明文中确立正确的明文.
- 下面的定理说明了对 Rabin 密码攻击的困难程度等价于大数 n 的分解.

定理 11.1 *求解方程* $x^2 \equiv a(\mathrm{mod}n)$ *与分解* n *是等价的.*

12 数字签名

- RSA 公钥密码体制另有一个重要作用—用作数字签名, 其重要性不亚于公钥密码本身.
- 一般的密码体制能保证信息不被非法窃取, 但不能防治发信方抵赖, 也不能防止收信方作假.

  例如因为

  $$c \equiv m^e(\mathrm{mod}n)$$

  – 其中 e 和 n 是公开的, 容易引起收、发双方的争议和纠纷, 并且很难判断
  – 同时也为密码分析者利用模求幂运算的特殊周期性破译 RSA 提供了可能
  – 若采用隐藏加密指数的方法可以有效的解决上述问题
- 如何隐藏加密指数, 隐藏了加密指数, 合法的接受者如何解密

  [1] 用户 A 用一把只有自己能锁也能开的” 数码锁 a” 把保密信息 P(或密钥 K) 锁在箱子内, 经普通信道传递给 B

  [2] 用户 B 无法解密”a”, 而是用一把只有自己能锁也能开的” 数码锁 b” 加锁, 此时, 保密信息 P 同时收到数码锁 a 和 b 的保护.

  [3] 将箱子经过普通信道传回给发方 A

  [4] 用户 A 收到” 箱子” 后, 用自己的密钥开” 数码锁 a”, 此时, ” 箱子” 还保留收方 B 的” 数码锁 b”

[5] 将此箱子经过普通信道传回给发送方 A

[6] 用户 B 收到" 箱子" 后, 用自己的密钥开" 数码锁 b" 此时, 收方 B 即刻获得保密信息 P

- 显然, 当" 箱子" 经过普通信道在 A 和 B 之间传递时, 均有" 数码锁" 的加密保护. 这种工作方式保密性比较高, 而且杜绝了发、收双方出现就封的可能性.

12.1 用户 A 和 B 设计工作密钥参数的准则

用户 A 和 B 根据 RSA 公钥密码体制设计工作密钥参数

- 若用户 A 和 B 分别选定了公开钥 $< e_n, n_a >$ 和 $< e_b, n_b >$, 私秘钥 $< d_a, p_a, q_a >$ 和 $< d_b, p_b, q_b >$
- 假设用户 A 为发送方, 需要将密钥 K 传送给接收方 B. 发送方 A 可以:

  [1] 选择两个随机大素数 p_s 和 q_s, 且令 $p_s q_s = n_s$.

  [2] 随机选择一对加密、解密指数 e_s 和 d_s, 这时的工作密钥参数为 $< p_s, q_s, e_s, d_s, n_s >$

12.2 发送方 A 和接收方 B 之间的信息变换和传递过程

[1] 发送方 A 利用 RSA 公钥密码体制的工作模式, 将随机大素数 $< p_s, q_s >$ 传送给接收方 B, 即发送方 A 利用接收方 B 的公开钥 $< e_b, n_b >$, 对 p_s 和 q_s 进行加密运算, 获得相应的密文 $< p_{sc}, q_{sc} >$, 并经过普通信道发送给接收方 B

[2] 接收方 B 利用自己的解密密钥 $< d_b, n_b >$ 对密文 $< p_{sc}, q_{sc} >$ 进行解密运算, 恢复出 p_s 和 q_s, 并得到 $n_s = p_s q_s$

[3] 根据 $[d_r, \Phi(n_s)] = 1$, 产生接收方为该次保密通信的选定的随机解密指数 d_r, 根据 $e_r \cdot d_r \equiv 1[\mathrm{mod}\Phi(n_s)]$ 产生双方对应的随机加密指数 e_r.

[4] 发送方 A 向接收方 B 传送该次保密通信的随机密 K

- A 利用自己的私秘钥 $< d_a, n_a >$ 对 K 进行签名变换, 得到 K 的签名变换 K_s