ch21-网络安全(2)

—— 非对称密码学

南京大学计算机系黄皓教授

2007年12月14日 星期五

内容

1. 公开密钥密码体制的基本概念

公开密钥算法的特点

- 加密功能
  - <K_U, K_R>是一对密钥;
  - $E(K_U, M) = Y$,
  - $D(K_R, Y)=M$,
  - $D(K_U, Y) \neq M$：加密者不必保密K_U，接收者可以将自己的公钥公布在目录服务器上。
- 签名功能
  - $S(K_R, M) = Z$,
  - $V(K_U, Z)=M$,
  - 寻找Y，使得$V(K_U, Y)=M$是计算不可能的。

公开密钥系统的的特性

- 加密和解密运算是计算上容易的问题。
- 密码分析应该属于NP完全问题。

单向陷门函数

- 对于每一个给定的k，f: x→f(k,x)是一一对应函数。
- 给定x和k，计算y=f(k,x)是容易的，反之，给定y和k，计算x是困难的问题。
- 存在陷门信息d(k)=k’及函数g(k’,y)，当y=f(k,x)时，x=g(d(k), y)。
- 陷门信息d(k)使得计算f(k,x)的逆变得容易起来。

2. RSA公开密钥算法

素数

- 素数：只能被1和它本身整除的自然数；否则为合数。
- 每个合数都可以唯一地分解出素数因子
  - $6 = 2 \cdot 3$
  - $999999 = 3 \cdot 3 \cdot 3 \cdot 7 \cdot 11 \cdot 13 \cdot 37$
  - $27641 = 131 \cdot 121$

  从2 开始试验每一个小于等于 $\sqrt{27641}$ 的素数。

素因子分解的速度

- 整数n的十进制位数因子分解的运算次数所需计算时间（每微秒一次）

50	1.4x1010	3.9小时
75	9.0x1012	104天
100	2.3x1015	74年
200	1.2x1023	3.8x109年
300	1.5x1029	4.0x1015年
500	1.3x1039	4.2x1025年

费马小定理

- 如果p是素数，a 与p互素，则

$$a^{p-1} = 1 \bmod p$$

如果1 ⩽i<j ⩽p-1, i • a mod p = j • a mod p 则

存在1 ⩽k⩽p-1，满足 k • a = 0 mod p ⇒ a | p， 得出矛盾。

{ a mod p, 2a mod p, …, (p-1)a mod p} 是{1,2, …, p-1}的一个排列。

(a mod p) (2a mod p) …[(p-1)a mod p] =(p-1)! mod p

a^{p-1} (p-1)!=(p-1)! mod p

a^{p-1} = 1 mod p

欧拉函数

- $\phi(n)$：小于n，但与n互素的正整数的个数。
  - p是素数，则$\phi(p)=p-1$。
  - p, q是素数，则$\phi(pq)=(p-1)(q-1)$。

欧拉定理

欧拉定理： 如果a与n是互素的，则$a^{\phi(n)} \equiv 1 \pmod{n}$

- 如果a与n是互素，则 $a \cdot b = a \cdot c \pmod{n}$ 推出：$b = c \pmod{n}$
- 设$S=\{x_1, x_2, \ldots\ldots, x_{\phi(n)}\}$是所有小于n且与n是互素的整数的集合。
- 因为$a \cdot x_i = a \cdot x_j \pmod{n}$，则有 $x_i = x_j \pmod{n}$，
- $\{a \cdot x_1 \pmod{n}, a \cdot x_2 \pmod{n}, \ldots\ldots, a \cdot x_{\phi(n)} \pmod{n}\} = S$

$$\prod_{i=1}^{\phi(n)} a \cdot x_i \pmod{n} = \prod_{i=1}^{\phi(n)} x_i \pmod{n}$$

$$a^{\phi(n)} \cdot \prod_{i=1}^{\phi(n)} x_i \pmod{n} = \prod_{i=1}^{\phi(n)} x_i \pmod{n}$$

$$a^{\phi(n)} \equiv 1 \pmod{n}$$

RSA 公开密钥密码算法

- M是明文，n是一个大数
  - 加密：$C = M^e \bmod n$
  - 解密：$M = C^d \bmod n = M^{ed} \bmod n$
- 条件：
  - 能找到e,d,n使得对所有的M，当M<n时，
    $M^{ed} = M \bmod n$
  - 对所有的M，计算M^e 和C^d 的容易的。
  - 给定e、n，推导 d 是困难的。

RSA 公开密钥密码算法（续）

- p、q是素数 **秘密地选择**
- n = p q， **公开n**
- 选择e: e与ϕ(n)是互素的; **公开选择**
- 计算d, 使得e$\cdot$ d=1 mod ϕ(n); **秘密地计算**

即：e $\cdot$ d=1 +s$\cdot\phi$(n);

根据欧拉定理的推广

对于任意的m, 0 < m < n，

$$m^{ed} = m^{s\,\phi(n)+1} \equiv m \pmod{n}$$

欧拉定理的推广

- 如果p, q是素数，n=pq, 则
  对于任意的m, 0< m < n，

$$m^{\phi(n)} \equiv 1 \pmod{n}$$

欧拉定理的推广

若n=pq是两个素数因子的乘积，X限制在加密与解密所要求的集合{0,1,…,n-1}之中，则对于任何明文X 有$X^{m\phi(n)+1} \equiv X \pmod{n}$

证明:

- X=0的情况显然是成立的。
- 下面要证明X>0的情况也是成立的:
- 若X不是与n=pq互素的，则X必须包括p或者q作为一个因子。设p为X的因子，对于某些正整数c，有关系式X=cp。
- 由于X限于集合{0,1,…,n-1}之中，且n=pq，从而可知X必定是与q互素的，否则X将包含q作为一个因子，在这种情况下X将越出n-1。
- 由欧拉定理有$X^{\phi(q)} \equiv 1 \pmod{q}$，其中$\phi(q)=q-1$，
- 但 $X^{m(p-1)\phi(q)} \equiv 1^{m(p-1)} \equiv 1 \pmod{q}$，对于任何整数m都成立，并且$(p-1)\phi(q)=(p-1)(q-1)=\phi(n)$，因此有 $X^{m\phi(n)} \equiv 1 \pmod{q}$，或对于某些整数t，有
  $$1= X^{m\phi(n)} +tq$$
- 用X=cp分别乘以等式的两端，得
- $$X= X^{m\phi(q)+1} +(tq)(cp)$$
- $$= X^{m\phi(q)+1} +tcn$$
- 因此 $X^{m\phi(q)+1} \equiv X \pmod{n}$
- 对于q为X的一个因子的情况，同理可证。

例

例:

- n=15, p=3, q=5, $\phi(n)=8$
- 生成密钥对：令e=3, 则d=3, (d e = 1 mod $\phi(n)$)
- 加密：设M=7， $C=7^e \bmod n = 7^3 \bmod 15 = 13$
- 解密： $M= C^d \bmod n = 13^3 \bmod 15 = 7$

问题

- 如何计算 $m^e \bmod n$
- 如何判定一个给定的整数是素数?
- 如何找到足够大的素数p和q？

如何计算 $m^e \bmod n$

- $e = \sum_{bi\neq 0} 2^i$
- $m^e = m^{\sum_{bi\neq 0} 2^i} \bmod n$

$$= \prod_{bi\neq 0} m^{2^i} \bmod n$$

$$= \prod_{bi\neq 0} (m^{2^i} \bmod n)$$

```
d:=1;
for i=k downto 0
    d:=d*d mod n
    if bi = 1 then
        d:=d*a mod n
return d
```

大素数生成— 素数的分布

- 存在无穷多个素数
  - 假设已知有k个素数$p_1, p_2, \ldots, p_k$;
  - 考虑 $n = p_1 \cdot p_2 \cdot \ldots \cdot p_k + 1$
    - 要么存在 $p \mid n$，这时$p \neq p_1$， …， $p \neq p_k$
    - 要么n也是素数

大素数生成— 素数的分布

- 设x>0，记π(x)为不大于x的素数的个数，则：

$$\lim_{x\to\infty}\frac{\pi(x)\ln x}{x}=1$$

$$\pi(x)\approx\frac{x}{\ln x}$$

长度为t位的数共有$2t-2t-1$个，所以一个长度为t位的随机数为素数的概率为

$$\left(\frac{2^t}{\ln 2^t}-\frac{2^{t-1}}{\ln 2^{t-1}}\right)/(2^t-2^{t-1})=\frac{t-2}{(t-1)t\ln 2}$$

例如：一个长度为256位的随机数为素数的概率为

$$\frac{254}{255\times 256\times\ln 2}\approx 0.011$$

基于费马小定理的素性检测

I. 随机选取一个奇数n
II. 随机选取一个整数a, a<n
III. $a^{n-1} \neq 1 \bmod n$，则n不是素数，转I
IV. 如果多次通过检测就接受n是素数，否则转II。

其它素性检测算法

- Solovay-Strassen算法
- Miller-Rabin算法
- Mesenne算法
- 基于椭圆曲线的素性检测

- 裴定一，祝跃飞，算法数论，科学出版社，2002年9月第一版。
- 周玉洁，冯登国，公开密钥密码算法及其快速实现，国防工业出版社，2002年9月。

对RSA的攻击方法

1. 强力攻击（穷举法）：尝试所有可能的私有密钥
2. 因子分解方法
3. 时间性攻击：取决于解密算法的运算时间

Pollard p-1 算法（分解因子算法）

```
a=2;
for  j=2 to B   do a = a^j;
d = gcd( a-1, n)
if d>1 and d<n then
        return d
else
        return false;
```

- 设p是n的一个素因子，假定p-1的每一个素因子q都有q$\leqslant$B。
- (p-1) | B!
- for 循环结束的时候， $a = 2^{B!} \bmod n$
- 由于 p | n， 所以 $a = 2^{B!} \bmod p$
- 由Fermat定理，2p-1=1 mod p
- 由于(p-1)|B!，a = 1 mod p,
- 所以p | (a-1)
- p | gcd(a-1, n)

- 对于n/d 继续。

Douglas R. Stinson，密码学原理与实践，电子工业出版社，2003年2月。

其它因子分解算法

- 二次筛法(quadric sieve)

$$O(e^{(1+o(1))\sqrt{\ln n \ln\ln n}})$$

- 椭圆曲线分解算法(elliptic curve factoring)

$$O(e^{(1+o(1))\sqrt{2\ln n \ln\ln n}})$$

- 数域筛法(number field sieve)

$$O(e^{(1.92+o(1))(\ln n)^{1/3}(\ln\ln n)^{2/3}})$$

因子分解的一些事例

- 1985年用二次筛法分解了69位的十进制数2^{251}-1.
- 1989年Lenstra、Manasse利用二次筛法再大量的工作站上并行运算，分解了一个106位的十进制数
- 1994年Atkins等利用二次筛法分解了一个RSA-129的十进制数。
- RSA-130在1996年被利用数域筛法分解。
- RSA-140在1999年2月被利用数域筛法分解。
- RSA-155在1999年8月被利用数域筛法分解。
- 155位的十进制数大约十512位，因此512位的RSA不应该被认为是安全的。
- 推测：768位的模式将在2010年被分解，1024位的模数将在2018年被分解。

其它对RSA的攻击

- RSA参数选择的攻击
- 选择密文攻击
- 对RSA公共模数的攻击
- 对RSA低加密指数的攻击
- 对RSA低解密密指数的攻击
- 对RSA的加密和签名的攻击

- 周玉洁，冯登国，公开密钥密码算法及其快速实现，国防工业出版社，2002年9月。

RSA 参数的选择

- 模数 n 的选择
  - p, q 的差 | p-q | 必须很大。
    - $n=p\bullet q = [(p+q)/2]^2 - [(p-q)/2]^2$
    - 如果p-q很小，则$n \approx [(p+q)/2]^2$
    - 逐个检查大于$n^{1/2}$的t, 看是否有t^2-n是一个平方数s^2。如果存在则$n=t^2-s^2 = (t+s)(t-s)$
  - p-1与 q-1 的最大公因子应很小
  - p, q 必须为强素数
- **赖溪松，韩亮，张真诚，计算机密码学及其应用，国防工业出版社，2001年7月。**

选择密文攻击

情况：Eve 在Alice的通信过程中进行窃听，设法成功选取了一个用Alice公钥加密的密文c。Eve想读出消息。

- e: 加密密钥，d: 解密密钥。
- $c=m^e \bmod n$, $m=c^d \bmod n$
- 已知c, 没有d， 求m，即求 $m = c^d \bmod n$
- 随机选取一个数r，r<n。
- $x= r^e \bmod n$ $r=x^d \bmod n$

- $y= x\, c \bmod n$ 选择的密文
- $t \cdot r = 1 \bmod n$
- $u=y^d \bmod n$:   请Alice对y签名解密服务
- $t\, u \bmod n$

$=t\, y^d \bmod n$

$= t\, x^d c^d \bmod n$

$= c^d \bmod n$

$= m$

不能天真地对消息签名。

公共模数攻击

- m是明文消息，两个加密密钥是e1、e2，n是公共的模数，两个密文是

  $c1 = m^{e1} \bmod n$

  $c2 = m^{e2} \bmod n$
- 密码分析者知道n、e1、e2、c1、c2
- 如果e1、e2互素(一般情况下会如此)，则存在r、s满足 $r \cdot e1+s \cdot e2=1$ 。
- 不妨假定$r<0$, $c1 \cdot c1' = 1 \bmod n$,则

  $$(c1')^{-r} \cdot c2^{s} = m \bmod n$$
- 不要在一组用户之间共享n。

3. 离散对数的基本概念

模运算及其性质

- $[a \pmod n) + b \pmod n)]\bmod n = (a+b) \bmod n$
- $[a \pmod n) - b \pmod n)]\bmod n = (a-b) \bmod n$
- $[a \pmod n) \cdot b \pmod n)]\bmod n = (a\,b) \bmod n$

- 如果a 与n互素，则存在b使得

$$a \cdot b = 1 \bmod n$$

- 小于n且与n互素的非0整数全体Z_n^*,按照模n乘法构成群。

求最大公因子的欧几里德算法

- 输入：整数a>b⩾0
- 输出：gcd(a,b)
  - If b=0 return a;
  - return (gcd(b, a mod b))
- 对于a>b >0 成立:

  $a = b \cdot q + r \qquad 0 \leqslant r < b$

  $r = a - b \cdot q$

  $gcd(a, b) = gcd(b, r)$

$a = b \cdot q_1 + r_1 \qquad gcd(a, b) = gcd(b, r1)$

$b = r_1 \cdot q_2 + r_2 \qquad = gcd(r_1, r_2)$

$r_1 = r_2 \cdot q_3 + r_3 \qquad = gcd(r_2, r_3)$

$\cdots\cdots\cdots \qquad \cdots\cdots\cdots$

$r_{k-3} = r_{k-2} \cdot q_{k-1} + r_{k-1} \qquad = gcd(r_{k-2}, r_{k-1})$

$r_{k-2} = r_{k-1} \cdot q_{k-2} + r_k \qquad = gcd(r_{k-1}, r_k)$

$r_k=0 \longrightarrow r_{k-1} \mid r_{k-2}$

$r_{k-2}=r_{k-1} \cdot q_{k-2}+r_k \longrightarrow \gcd(r_{k-2}, r_{k-1})= r_{k-1}$

$r_{k-3}=r_{k-2} \cdot q_{k-1}+r_{k-1} \longrightarrow \gcd(r_{k-3}, r_{k-2})= \gcd(r_{k-2}, r_{k-1})= r_{k-1}$

$\cdots\cdots\cdots$

$r_1=r_2 \cdot q_3+r_3 \longrightarrow \gcd(r_1, r_2)= \gcd(r_2, r_3)= \cdots = r_{k-1}$

$b=r_1 \cdot q_2+r_2 \longrightarrow \gcd(b,r_1)=\gcd(r_1, r_2)= \cdots = r_{k-1}$

$a= b \cdot q_1 + r_1 \qquad \gcd(a,b)=\gcd(b,r_1)=\cdots = r_{k-1}$

$r_k=1$ $\qquad$ $\gcd(r_{k-1}, r_k)=\gcd(r_{k-1}, 1)=1$

$r_{k-2}=r_{k-1}\bullet q_{k-2}+r_k \longrightarrow \gcd(r_{k-2}, r_{k-1})=\gcd(r_{k-1}, r_k)=1$

$r_{k-3}=r_{k-2}\bullet q_{k-1}+r_{k-1} \longrightarrow \gcd(r_{k-3}, r_{k-2})=\gcd(r_{k-2}, r_{k-1})=1$

• • • • • • • • • • • •

$r_1=r_2\bullet q_3+r_3 \longrightarrow \gcd(r_1, r_2)=\gcd(r_2, r_3)=\bullet\bullet\bullet=1$

$b=r_1\bullet q_2+r_2 \longrightarrow \gcd(b,r_1)=\gcd(r_1, r_2)=\bullet\bullet\bullet=1$

$a=b\bullet q_1+r_1$ $\qquad$ $\gcd(a,b)=\gcd(b,r_1)=\bullet\bullet\bullet=1$

$a= b \bullet q_1 + r_1$	$r_1 = a - b \bullet q_1 = s_1 a + t_1 b$
$b = r_1 \bullet q_2 + r_2$	$r_2 = b - r_1 \bullet q_2 = b - (s_1 a + t_1 b)\, q_2 = s_2 a + t_2 b$
$r_1 = r_2 \bullet q_3 + r_3$	$r_3 = r_1 - r_2 \bullet q_3 = s_3 a + t_3 b$
$\bullet\bullet\bullet\bullet\bullet\bullet\bullet\bullet\bullet\bullet\bullet\bullet$	$\bullet\bullet\bullet\bullet\bullet\bullet\bullet\bullet\bullet\bullet\bullet\bullet$
$r_{k-3} = r_{k-2} \bullet q_{k-1} + r_{k-1}$	$r_{k-1} = r_{k-3} - r_{k-2} \bullet q_{k-1} = s_{k-1} a + t_{k-1} b$
$r_{k-2} = r_{k-1} \bullet q_{k-2} + r_k$	$r_k = r_{k-2} - r_{k-1} \bullet q_{k-2} = s_k a + t_k b$
$r_k = 1$	$1 = r_k = \mathbf{gcd(a,b) = s \cdot a + t \cdot b}$

如果a，b互素，则存在s满足：

$$\mathbf{s \bullet a = 1 \mod b}$$

s是a的模n的逆： $\mathbf{s = a^{-1} \mod n}$

- Z_n 表示为按照 mod n 的加法构成交换群。
- 循环群：设G是群，如果存在一个元素a∈G，对与任意一个元素b，都存在一个整数i，使得b=a^i，则G称为循环群，元素a称为G的一个生成元；G也称为由a生成的群，记为G=<a>。
- Z_n是循环群，1 是生成元。
- Z_n中与n互素的元素a (gcd(a, n) =1)的全体按照乘法构成群，记为 Z_n^*
- Z_n^* 按照加法与乘法构成域 。
- 如果p是素数，Z_n^* 记为F_p。
- 域的乘法生成元称为本原根。
- 如果p是素数，则F_p 必有本原根，也就是F_p按照乘法构成循环群。

- 如果F是一个域，域F上的多项式全体F[x]构成环。
- 设f(x)也是域F上的多项式，F[x]中所有模f(x)的余式(记为$F[x]_f$)构成一个环。
- 设f(x)也是域F上的不可约多项式，F[x]中所有模f(x)的余式$F[x]_f$构成一个域。
- 设f(x)也是域F上的n次不可约多项式，F[x]中所有模f(x)的余式$F[x]_f$构成一个域，记为 F_{p^n} 。
- 任何一个有限域F，必存在素数p、整数n，使得F与F_{p^n}同构。
- F_{p^n} 的乘法群是一个循环群。

- 设p=2
  - 寻找上的8次不可约多项式f(x).
  - 8位二进制串，按照$F_2[x] / f(x)$上的运算构成域F_{2^8}= GF(2^8) 。
- GF(2^8)上加法定义为按位异或;
- GF(2^8)上定义乘法如下:
- $(a_7a_6a_5a_4a_3a_2a_1a_0)\ (b_7b_6b_5b_4b_3b_2b_1b_0) = (c_7c_6c_5c_4c_3c_2c_1c_0)$
  - $a(x)= a_7x^7+a_6x^6+a_5x^5+a_4x^4+a_3x^3+a_2x^2+a_1x+a_0$
  - $b(x)= b_7x^7+b_6x^6+b_5x^5+b_4x^4+b_3x^3+b_2x^2+b_1x+b_0$
  - $c(x) = c_7x^7+c_6x^6+c_5x^5+c_4x^4+c_3x^3+c_2x^2+c_1x+c_0$
  - $m(x)=x^8+x^4+x^3+x+1$
  - $c(x) = a(x)\ b(x) \mod f(x)$
  - f(x)是不可约多项式,
- GF(2^8)按照这样的定义的加法和乘法构成域。
- 如果选择$m(x) = x^8+1$, GF(2^8)按照这样的定义的加法和乘法构成环。

循环群上的离散对数

- 群⟨G, ·⟩的一个n阶元素$\alpha \in G$和元素$\beta \in \langle \alpha \rangle$
- **找一个唯一的整数s使得**

  $\alpha^s = \beta$

- **离散对数的蛮力算法**
  - **给定循环群G、β、α；**
  - 分别对i=1，…，|G|, 检测$\alpha^i = \beta$；

- 蛮力法最多需要|G|次群运算。

离散对数算法— Shanks算法

- 设循环群G的阶为n。
- m是正整数，满足 n>m>1。
- 对于任意的t, $0 \leqslant t<n$, $h=g^t$,我们有
- $t=qm+r$, $0 \leqslant q \leqslant [n/m]$, $0 \leqslant r<m$
- $h=g^t=g^{qm+r}=g^{qm}g^r = (g^m)^q g^r$
- $h \cdot g^{-r}= (g^q)^m$

- 计算

  $L= \{ (g^m)^q$， $q=0,1,\ldots,[n/m] \}$
- 然后计算

  $h \cdot g^{-r}$， $r=0,1,\ldots,m-1$
  - 找出哪一个出现在L表中，
  - 记录相应的q, r,
- $t=q \cdot m + r$ 就是需要的离散对数。
- 算法复杂度：O([n/m]+m)

Shanks算法是第一个达到求解离散对数一般算法的复杂度下界的确定性一般算法。

Diffie-Hellman问题

- 给定一个素数p，mod p的一个素根a,已知**a^x (mod p) , a^y (mod p)** 求**a^{xy} (mod p)** 。
  - **求离散对数问题：由p， a^x， a求x。**
  - **计算a^y (mod p)^x= a^{xy} (mod p)。**
- Diffie-Hellman密钥交换
- Diffie-Hellman问题的难度不超过离散对数的难度。

Diffie-Hellman密钥交换

Alice　　　　　　　　　　　　　　　　　　　　　　　　　　Bob

x　　　　　　　　$X = g^x \bmod n$
$$\longrightarrow$$

$$\longleftarrow$$
　　　　　　　　$Y = g^y \bmod n$　　　　　　　　　　　y

$(g^y)^x \bmod n$　　　　　　　$=$　　　　　　　$(g^x)^y \bmod n$

窃听者从X，Y无法计算x, y, 因而无法计算 $g^{xy} \bmod n$

4. ElGamal密码算法

ElGamal 加密算法

- 生成密钥对
  - 生成一个大的随机素数p和整数mod p的乘法群Z_p^*的生成元α;
  - 选取一个随机整数s (1≤x≤p-2),计算 $\beta = \alpha^s \pmod p$;
  - 公钥 = (p, α, β), 私钥 = s。
- 加密明文信息m
  - 选取一个随机整数k(1≤k≤p-2),
  - 计算$X=\alpha^k \pmod p$,$Y=m(\beta)^k \pmod p$
  - (X, Y)就是密文
- 解密
  - 计算X^{p-1-s}, $X^{p-1-s} \bmod p = X^{-s} \bmod p = \alpha^{-sk} \bmod p$
  - 计算$Y \cdot X^{-s} \pmod p$恢复m, $Y \cdot X^{-s} \pmod p = m(\alpha^s)^k \cdot \alpha^{-sk} \bmod p = m$
- 加密者用$(\alpha^s)^k$将明文隐藏起来，因为解密者知道s，所以可以从密文的一部分$X=\alpha^k$恢复$(\alpha^s)^k$，因而可以解密。

ElGamal 签名算法

- 生成密钥对
  - 生成一个大的随机素数p和整数mod p的乘法群Z_p^*的生成元α;
  - 选取一个随机整数s $(1\leqslant s \leqslant p-2)$, 计算 $\beta = \alpha^s \pmod p$;
  - 公钥(p, α, β), 私钥s。
- 对信息m签名
  - 选取一个随机整数 k $(1\leqslant k\leqslant p-2)$，计算$X=\alpha^k \bmod p$
  - 从方程 $m = (s \cdot X + k \cdot Y) \bmod (p-1)$ 中求解Y;
  - 签名为(X,Y);
- 验证签名
  - 验证等式: $\beta^X \cdot X^Y = \alpha^m \bmod p$
  - $(\alpha^s)^X \cdot (\alpha^k)^Y = \alpha^{s\cdot X + k\cdot Y} \bmod p$
    $= \alpha^{m + t\cdot(p-1)} \bmod p$
    $= \alpha^m \cdot \alpha^{t\cdot(p-1)} \bmod p$
    $= \alpha^m \bmod p$