Section 2.4 Public Key Cryptosystem

本讲主要内容

- 公钥密码体制的简介
- 背包问题
- **RSA算法**
- **ElGamal算法**
- **ECC算法**
- **IBE算法**

回顾

- 保密通信进入计算机网络时代，传统密码体制逐渐暴露其固有的弱点，当时主要体现密钥管理。
- 1976年W. Diffie和Hellman 在《密码学的新方向》中首次提出了公钥密码算法的思想。
- 1978年后Rivest，Shamir和Adleman提出的RSA算法体现了公钥算法的思想，并具有实用性。
- 公钥密码体制是现代密码学的一个标志，到目前为止，是密码学史上最大也是唯一真正的革命。
- 公钥密码引起密码界高度关注，并得到迅速的发展，尤其在信息安全的应用中涉及公钥密码技术。

引言

在公钥密码体制以前的整个密码学发展史中，所有的密码算法，包括原始手工计算的、由机械设备实现的以及由计算机实现的，都是基于代换和换位这两个基本方法，建立在位(字符)方式的操作上。

而公钥密码体制则为密码学的发展提供了新的理论和技术思想，一方面公钥密码算法是建立在数学函数基础上的，而不是建立在位(字符)方式的操作上的；另一方面公钥密码算法是以非对称的形式使用两个密钥，两个密钥的使用对密钥分配、认证等都有着深刻的意义。可以说，公钥密码体制的出现在密码学发展史上是一次质的飞跃。

对称密码体制的缺陷

- 密钥分配问题

通信双方要进行加密通信，需要通过秘密的安全信道协商加密密钥，而这种安全信道在实际中很难实现。

- 密钥管理问题

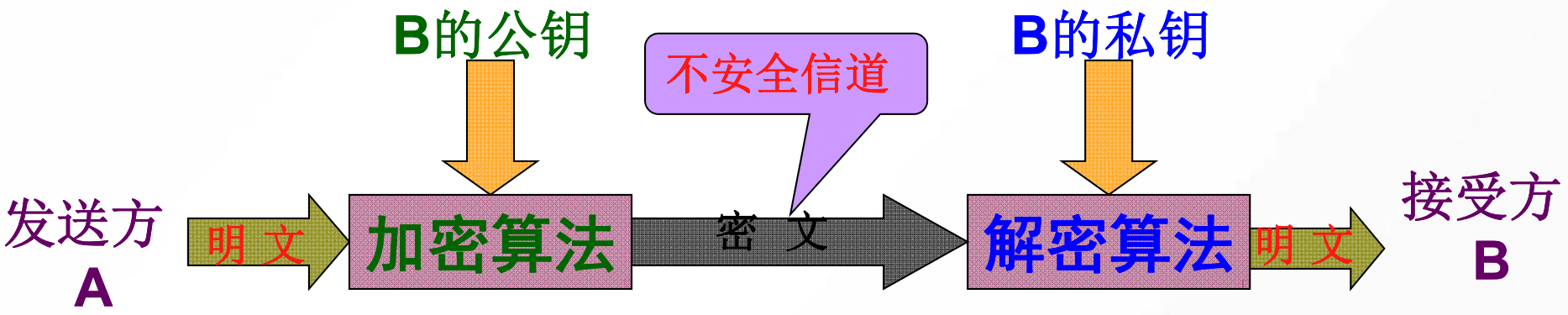
在有*n*个用户的通信网络中，每个用户要想和其它*n*-1个用户进行通信，必须使用*n*-1个密钥，而系统中的总密钥量将达到*n*(*n*-1)/2。当*n*较大时，这样大的密钥量，在产生、保存、传递、使用和销毁等各个环节中都会变得很复杂，存在着安全隐患。

- 数字签名问题

对称密码体制中通信双方拥有同样的密钥，所以接收方可以伪造签名，发送方也可以否认发送过某消息，难于解决陌生人之间的身份认证和交易信息认证的问题。

加解密模型

- 发送方A查找接受方B的公钥。
- A采用公钥加密算法以B的公钥作为加密密钥对明文加密。
- A通过不安全信道将密文发送给B。
- B收到密文后使用自己的私钥对密文解密还原出明文。



基本原理(陷门单向函数)

- 正向计算容易。即如果知道密钥 P_k 和消息**M**，容易计算 $C=f_{p_k}(M)$ 。
- 在不知道密钥 S_k 的情况下，反向计算不可行。即如果只知道加密后的消息**C**而不知道密钥 S_k ，则计算不可行 $M=f^{-1}(C)$。
- 在知道密钥 S_k 的情况下，反向计算容易。即如果同时知道加密消息**C**和密钥 S_k ，则计算 $M=f^{-1}{}_{S_k}(C)$ 是容易的。这里的密钥 S_k 相当于陷门，它和 P_k 配对使用。

公钥密码算法应满足的要求

- 接收方B产生密钥对（公钥PK_B和私钥SK_B）在计算上是容易的。
- 发送方A用接收方的公钥对消息m加密以产生密文c，即 $c = E_{pk_B}(m)$ 在计算上是容易的。
- 接收方B用自己的私钥对c解密，即$m = D_{sk_B}(c)$ 在计算上是容易的。
- 攻击方由B的公钥PK_B求私钥SK_B在计算上是不可行的。
- 攻击方由密文c和B的公钥PK_B恢复明文m或求私钥SK_B（选择明文攻击）在计算上是不可行的。

公钥密码算法思考的主要问题

- 私钥和公钥是如何生成的，它们之间有着怎样的关系；
- 安全的密钥长度是多少；
- 公钥密码体制的安全性依赖的数学难题是什么；
- 如何实现加密算法(公钥加密)，以及解密算法(私钥解密)，反之亦然（数字签名）；
- 现在这种公钥密码体制的安全分析。

常用算法

- 背包问题（NP问题）。

  背包算法

- 基于大整数素因子分解问题。

  RSA，Rabin等

- 基于有限域乘法群上的离散对数问题。

  Elgamal（DS）。

- 椭园曲线上的离散对数问题。

  ECC。

- 基于身份的密码体制

  IBE。

背包算法简介

- 由默克尔（Merkle）和赫尔曼（Hellman）提出的；
- 最早提出的公开密钥算法；
- 安全性源于背包问题，是一个NP问题；
- 大部分是不安全的，很少使用；

背包问题

已知一长度 b 的背包及长度分别为 $a_1, a_2, \cdots, a_n$ 的 n 个物品。假定这些物品的半径和背包相同，若从这 n 个物品中选出若干物品使得恰好装满这个背包。

公式化的描述为：给定一个正整数集$A=\{a_1,a_2,\ldots,a_n\}$，已知b是A的某子集中元素的和。问题是，找到一个*n*元的0、1向量$X=(x_1,x_2,\ldots,x_n)$使得：

这是一个NP问题。

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

其中a_1，a_2，…，a_n 和 b 都是正整数。

超递增序列

如果序列 a_1，a_2，…，a_n 满足条件：

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

则称该序列为超递增序列。

例如：{1，2，4，8，…，2^n }

{1，3，6，13，27，52，…}

而 {1，3，4，9，15，25} 不是。

如果序列为超递增序列，则背包问题则是P类问题。

MH背包密码的基本思想

利用实际上存在两类不同的背包问题，一类在线性时间内可解(超递增背包问题)，而另一类不能(普通背包问题)。易解的背包问题可以转化成难解的背包问题。公钥使用难解的背包问题，它可很容易地被用来加密明文但不能用来解密密文。私钥使用易解的背包问题，它给出一个解密的简单方法。不知道私钥的人要破解密文就不得不解一个难解的背包问题。

将消息编码为背包问题的解。明文分组长度等于堆中物品个数，且明文位与b的值相对应，密文是计算得到的和值。

公私钥对的生成

公开的背包序列 $a_1, a_2, \cdots, a_n$ 是由超递增序列 $b_1, b_2, \cdots, b_n$ 进行以下变换得到的。

选取素数p和u，且 $p > \sum_{i=1}^{n} b_i$ ，$1 \le u \le p-1$ 。

利用已知**u**并根据**uv=1 (mod p)**，可求得**v**。

下面作变换称为墨科-赫尔曼(Merkle-Hellman)变换：

$a_k \equiv ub_k \ (\mathrm{mod}\ p)$ ， $k=1, 2, \cdots, n$

非超递增序列$\{a_i\}$和p作为公钥；

超递增序列$\{b_i\}$和v作为私钥；

加解密的实现过程

消息$m=m_1m_2\ldots m_n$ 是n位0，1 二进制串

加密过程：

$$c=a_1m_1+a_2m_2+\ \ldots+a_nm_n$$

c 是消息 m 的密文

解密过程：

利用超递增序列$\{b_i\}$求 vc 的解，可得消息m。

(这是因为 $vc=va_1m_1+va_2m_2+\ \ldots+v\,a_nm_n$

$$=b_1m_1+b_2m_2+\ \ldots+b_nm_n\ \ )$$

$uv\equiv1\ (\mathrm{mod}\ p)\quad a_k\equiv u\,b_k\ (\mathrm{mod}\ p)\quad b_k\equiv va_k\ (\mathrm{mod}\ p)\ (k=1,\ldots,n)$

举例说明（公私钥对生成）

{1,3,7,13,26,65,119,267}是超递增序列，是私有密钥。

1+3+7+13+26+65+119+267=501

取p=523 和u=467，可得v=28。

通过 uv=13076≡1 mod 523 可验证。然后，计算：

a_1=467*1≡467 mod 523　　a_2=467*3≡355 mod 523

a_3=467*7≡131 mod 523　　a_4=467*13≡318 mod 523

a_5=467*26≡113 mod 523　　a_6=467*65≡21 mod 523

a_7=467*119≡135 mod 523　　a_8=467*267≡215 mod 523

{467,355,131,318,113,21,135,215}是背包序列，是公开密钥。

举例说明(加解密)

设明文m=10101100

加密过程：

$a_1+a_3+a_5+a_6=467+131+113+21=732$（密文）

解密解密：

接受方收到密文732后，乘以v=28，再取模523，即得：

$732*28=20496=99 \mod 532$

解超递增序列背包问题：

$m_1+3m_2+7m_3+13m_4+26m_5+65m_6+119m_7+267m_8=99$

$m_1=m_3=m_5=m_6=1$， $m_2=m_4=m_7=m_8=0$

即得明文10101100

安全性分析

- 背包问题是NP问题，至今还没有有效的求解方法。若对 2^n 种所有可能进行穷举搜索， 当n很大时在实际上是不可能的。

  以 n=100 为例：$2^{100}=1.27*10^{30}$

  以每秒搜索10^7种方案的超高速电子计算机进行穷举，一年只能完成$3.2*10^{14}$次，所以共要：

  $1.27*10^{30} / 3.2*10^{14} = 4*10^{15}$年

- 事实上，墨科-赫尔曼变换将超递增序列的特性隐蔽性不够，已被证明大多数基于背包问题的公钥密码是不安全的，现很少在使用，但它的思想是值得借鉴的。关于对背包问题的攻击，感兴趣的同学请参见相关文献。

背包密码小结

- 背包密码算法是第一个公开密钥算法，但大多数不适合数字签名。
- 背包算法的安全性源于背包问题，它是一个NP问题(随着背包中物品数目的增多而计算量呈指数增长)。尽管这个算法是不安全的，但由于它证明了如何将NP问题用于公钥密码体制，是值得研究的。
- 自从MH方案被破译后，又有许多其他的背包变型被提出，但这些背包中的大多数都被用同样的密码分析方法攻破，少数则采用更高级的分析方法，虽然有极个别的背包变型还没有破解，但人们已不再信赖它们了。

RSA公钥密码的简介

在Diffie和Hellman提出公钥密码体制的设想后两年，先后有Merkle和Hellman提出了MH背包公钥密码，Rivest、Shamir、Adleman联合提出的简称为RSA公钥密码系统。RSA虽稍后于MH背包公钥系统，但它到目前为止应用最广的一种公钥密码。RSA的理论基础是数论的欧拉定理，它的安全性依赖于大整数的素因子分解的困难性。

欧拉函数

给定一个正整数 n ，用 $\varphi(n)$ 表示比 n 小但与 n 互为素数的正整数的个数，也称 $\varphi(n)$ 为欧拉函数。

$$\varphi(n) = r_1^{a_1-1}(r_1-1)r_2^{a_2-1}(r_2-1)\cdots r_n^{a_n-1}(r_n-1)$$

其中 $n = r_1^{a_1} r_2^{a_2} \cdots r_n^{a_n}$ 。

例如: $24 = 2^3 * 3$, $\varphi(24) = 2^{3-1}(2-1) * 3^{1-1}(3-1) = 8$

{1, 5, 7, 11, 13, 17, 19, 23}

$35 = 5 * 7$, $\varphi(35) = (5-1)(7-1) = 24$

{1, 2, 3, 4, 6, 8, 9, 11, 12, 13, 16, 17, 18, 19, 22, 23,

24, 26, 27, 29, 31, 32, 33, 34}

欧拉定理

若整数 a 和 n 互素，则$a^{\varphi(n)} \equiv 1 \pmod{n}$

证明：设$\phi(n)=k$。又设$r_1, r_2, \cdots, r_k$是小于n并与n互素的数，且由于a是与n互素的数，则$ar_1, ar_2, \cdots, ar_k$也和n互素且两两不同。

若：$ar_i \equiv ar_j \pmod{n}$

则根据数论定理，因a与n互素，所以存在ã满足：

$$\tilde{a}a \equiv 1 \pmod{n}$$

$$\tilde{a}ar_i \equiv \tilde{a}ar_j \pmod{n}$$

即：$r_i \equiv r_j \pmod{n}$ （与假定矛盾）

所以，$\{r_1, r_2, \cdots, r_k\} = \{ar_1, ar_2, \cdots, ar_k\}$

所以，等式$a^k r_1 r_2 \cdots r_k \pmod{n} \equiv r_1 r_2 \cdots r_k \pmod{n}$成立。

但$r_1, r_2, \cdots, r_k$和n互素，故 $a^k \equiv 1 \pmod{n}$

举例说明

$5^{\varphi(24)} \equiv 1 \bmod 24$

φ(24)=8。又**1,5,7,11,13,17,19,23**是小于24并与24互素的数.

1*5=5 mod 24≡5　　5*5=25 mod 24≡1　　7*5=35 mod 24≡11

11*5=55 mod 24≡7　13*5=65 mod 24≡17　17*5=85 mod 24≡13

19*5=95 mod 24≡23　　23*5=115 mod 24≡19

{1,5,7,11,13,17,19,23} ={5*1,5*5,5*7,5*11,5*13,5*17,5*19,5*23}

5*1*5*5*5*7*5*11*5*13*5*17*5*19*5*23 mod 24

= 1*5*7*11*13*17*19*23 mod 24

$5^8 \equiv 1 \bmod 24$　即：$5^{\varphi(24)} \equiv 1 \bmod 24$

应用举例

计算$2^{1000000} \bmod 77 \equiv ?$

显然，$\gcd(2,77)=1$，$\phi(77)=\phi(7*11)=6*10=60$。

根据欧拉定理，$2^{60} \bmod 77=1$。

由于，$1000000=16666*60+40$

所以，$2^{1000000} \bmod 77 = 2^{16666*60+40} \bmod 77$

$\equiv 2^{40} \bmod 77 \equiv 1024^4 \bmod 77 \equiv 23^4 \bmod 77$

$\equiv 67^2 \bmod 77 \equiv 10^2 \bmod 77 \equiv 23$。

RSA的密钥对生成算法

1. 选取两个大素数p和q，两个数长度接近且相差较大。

2. 计算$n=p*q$，$\phi(n)=(p-1)(q-1)$

3. 随机选取整数e，满足$\gcd(e,\phi(n))=1$

4. 计算d，满足$d*e \equiv 1 \pmod{\phi(n)}$。

注：p和q保密。

e和n为公钥，d为私钥。

RSA的加解密过程

- 加密算法：$c=E(m)\equiv m^e \pmod n$
- 解密算法：$m=D(c)\equiv c^d \pmod n$

证明：

$$c^d=(m^e)^d=m^{ed} = m^{k\phi(n)+1}$$

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

若m与n互素，根据欧拉定理，RSA加解密成立。

- 明文m的长度小于$\log_z n$ 位，即 $m<n$。

RSA的加解密证明过程（续）

若gcd(m, n)>1, 由于n=p*q, 所以gcd(m, n)必含p, q之一，设gcd(p, q)=p, 或m=s*p， 1≤s≤q, 由欧拉定理得：

$m^{\phi(q)}=1 \pmod{q}$.

因此，对于任何k，总有： $m^{k(q-1)}=1 \pmod{q}$

$m^{k(p-1)(q-1)}=(1)^{k(p-1)}=1 \pmod{q}$

即：$m^{k\phi(n)}=1 \pmod{q}$ 或 $1= m^{k\phi(n)}+h*q$

其中h是某个整数， 由假定m=s*p, 得：

$s*p= m^{k\phi(n)} *s*p+s*p*h*q$， $m= m^{k\phi(n)+1}+s*h*p*q$

$m=m^{k\phi(n)+1}+s*h*n$, 即得： $m^{k\phi(n)+1} \equiv m \pmod{n}$

RSA的加解密举例说明

p=43， q=59， n=p*q=43*59=2539　　ϕ(2539)=42*58=2436

取e=13,解方程 de ≡1 (mod 2436)　　得：d=937

若有明文：cyber greatwall

先将明文分块为：cy be rg re at wa ll

如利用英文字母表的顺序，即a为00，b为01，。。。，z为25，将明文数字化得：

0224　0104　1706　1704　0019　2200　1111

利用加密得到密文：

1692　0803　1359　1943　2299　1254　0724

RSA的计算技巧

- 避免中间结果是天文数字，利用模运算的性质：

  $[(a \bmod n)*(b \bmod n)] \bmod n \equiv (ab) \bmod n$

- 求m^e（mod n）可通过指数e的快速取模指数算法：

```
t ← 0；  c ← 1
for  i ← k downto 0
do  t ← 2×t
        c ← (c×c) mod n
if   b_i=1  then  t ← t+1
        c ← (c×m) mod n
return c
```

注：$e=(b_k b_{k-1} \ldots b_2 b_1)$

RSA算法的安全性

- RSA同态性质
- 选择密文攻击
- 针对n分解的攻击
- 侧信道攻击
- 短明文
- 针对RSA算法参数的攻击

同态性

由RSA体制的加密可知，对于一切m_1，$m_2 \in Z_n$，有$E_k(m_1m_2) \equiv E_k(m_1)E_k(m_2) \pmod{n}$，称这个性质为RSA的同态性质。根据这个性质，如果敌手知道秘文c_1和c_2的明文m_1和m_2，就知道$c_1c_2 \pmod{n}$所对应的明文是$m_1m_2 \pmod{n}$。

选择密文攻击

敌手对用户A和B的通信过程进行窃听，得到用户B的公开密钥e加密的消息c。为了解密c，敌手随机选择数r<n，然后用B的公钥e计算：

$$x \equiv r^e \pmod{n};\ y \equiv xc \pmod{n};\ t \equiv r^{-1} \pmod{n}$$

敌手让B用他的私钥d对y签名，即计算$u \equiv y^d \pmod{n}$，敌手收到u后计算：

$$tu \pmod{n} \equiv r^{-1}y^d \equiv r^{-1}x^d c^d \equiv r^{-1}rc^d \equiv m \pmod{n}$$

这表明，使用RSA公钥体制时，绝对不能用自己的私钥d对一个陌生人提供给你的随机消息进行签名。

针对n分解的攻击

- 试除法

完全尝试小于$n^{1/2}$的所有素数，直到因子找到。根据素数理论，尝试的次数上限为 $2\sqrt{n}/\log\sqrt{n}$ 。

- 因子分解分析法

p-1因子分解法、**p+1**因子分解法、二次筛因子分解法、椭圆曲线因子分解法、数域筛因子分解法等。

二次筛因子分解法

基于著名的费马因子分解：找出正整数 x 、y，使得 $x^2 \equiv y^2 \pmod{n}$，即存在整数 c 满足 $cn=x^2-y^2=(x-y)(x+y)$，并且满足 $x \neq \pm y \pmod{n}$，因此 n 是数 $x^2-y^2=(x-y)(x+y)$ 的因子，故 $\gcd(x+y, n)$ 或 $\gcd(x-y, n)$ 均为 n 的因子，由此便可将 n 分解。费马因子分解就是利用了二次筛法的原理，它基于等式 $cn=x^2-y^2$，令 $c=1$，则 $n+y^2=x^2$，为找出 x 、y 的值直接计算 $n+1^2$ 、$n+2^2$ 、…、$n+k^2$ 直至 $n+k^2$ 为完全平方数为止。如令 $n=295927$，**$295927+3^2=295936=544^2$**，因此可得因数分解 **$295927=544^2-3^2$** **=（544+3）（544-3）=547×541。**

用|p-q|/2步找到分解。

侧信道攻击

指攻击者利用从公钥密码设备中容易获得的信息（如电源消耗、运行时间和在特意操控下的输入/输出行为等）攻击秘密信息（如私钥和随机数等）。侧信道攻击的核心思想就是首先通过对加密软件或硬件运行时产生的各种泄露信息进行监测，这些泄露信息包括程序的运行时间、能量消耗、机器发出声音和硬件发出各种电磁辐射等，然后对监测得到的各种数据进行分析和推断，从而得出算法内部运行的情况，进而破解出这些秘密信息。

短明文

用RSA加密消息m为$c \equiv m^e \bmod n$(假设m少于30位)。虽然m很小，密文c可能和n差不多大小，但攻击者可以进行如下攻击，他建立两张表:

(1)$cx^{-e} \bmod n$ 对所有的$1 \leqq x \leqq 10^9$;

(2) $y^e \bmod n$ 对所有的$1 \leqq y \leqq 10^9$;

然后寻找第一张表和第二张表的匹配。如果找到了一个，那么就对某个x和y有$cx^{-e} \equiv y^e \bmod n$，得到$c \equiv (xy)^e \bmod n$，所以$m \equiv xy \bmod n$。

针对RSA算法参数的攻击

- 对素数p和q的选取的一些限制：
  - p和q的长度相差不能太大，大小相差比较大；
  - p-1和q-1都应有大的素数因子；
- 共模攻击
- 低指数攻击
- 不动点

难抵御试除法的攻击。

若不然，$p-q=k, q=p+k$。则

$$(p+q)^2-(p-q)^2=4pq=4n$$

$$(p+q)^2=4n+k^2$$

$$p+q=\sqrt{4n+k^2}$$

又 $p-q=k$

可联立解出p和q。

循环攻击

设攻击者截获密文c,可如下进行重复加密:

$$c^e \equiv (m^e)^e \equiv m^{e^2} \pmod{n}$$

$$c^{e^2} \equiv (m^e)^{e^2} \equiv m^{e^3} \pmod{n}$$

$$\cdots$$

$$c^{e^{t-1}} \equiv (m^e)^{e^{t-1}} \equiv m^{e^t} \pmod{n}$$

$$c^{e^t} \equiv (m^e)^{e^t} \equiv m^{e^{t+1}} \pmod{n}$$

若$m^{e^{t+1}} \equiv c \pmod{n}$,即$(m^{e^t})^e \equiv c \pmod{n}$,则有$m^{e^t} \equiv m \pmod{n}$,

即$c^{e^t-1} \equiv m \pmod{n}$,所以在上述重复加密的倒数第2步就已恢复出明文$m$,这种攻击只有在$t$较小时才是可行的.为抵抗这种攻击,$p,q$的选择应保证使$t$很大.

循环攻击(续)

设m在模n下阶为k,由$m^{e^t} \equiv m(\mathrm{mod}\, n)$得$m^{e^t-1} \equiv 1(\mathrm{mod}\, n)$,所以$k \mid (e^t-1)$,即$e^t \equiv 1(\mathrm{mod}\, k)$,$t$取为满足上式的最小值(为$e$在模$k$下的阶).又当$e$与$k$互素时$t \mid \phi(k)$.为使$t$大,$k$就应大且$\phi(k)$应有大的素因子.又由$k \mid \phi(n)$,所以, 为使$k$大,$p-1$和$q-1$都应有大的素因子.

共模攻击

- 不同用户不可共享模n。

假如用户A和B共享模数为n，他们的公钥分别为e_1,e_2，攻击者对同一消息m加密可得：$c_1 \equiv m^{e_1} \bmod n$ 和 $c_2 \equiv m^{e_2} \bmod n$ 。攻击者截获密文c_1和c_2后，利用欧几里德(Euclid)算法计算得到r和s，满足 $r*e_1+s*e_2=1 \bmod n$ 。然后，攻击者将计算:

$$c_1^{\ r} * c_2^{\ s} = (m^{e_1})^r (m^{e_2})^s \bmod n = m^{re_1+se_2} \bmod n = m$$

- 不同用户选用的素因子(p或q)不能相同。

著名的素数定理：不超过x的素数的个数大约为x/lnx。在长度为512位的自然数中，有超过10^{151}个素数，且素数的概率大约为1/ lnx≈1/177。

低指数攻击

- 为了增强加密的有效性，希望选择较小的加密指数e（公钥）。如果相同的消息要送给多个实体，就不应该使用小的加密指数。例如：

$$\begin{aligned} c_1 &\equiv m^3 (\bmod n_1) \\ c_2 &\equiv m^3 (\bmod n_2) \\ c_3 &\equiv m^3 (\bmod n_3) \end{aligned} \Longrightarrow \begin{aligned} m^3 &\equiv c_1 (\bmod n_1) \\ m^3 &\equiv c_2 (\bmod n_2) \\ m^3 &\equiv c_3 (\bmod n_3) \end{aligned}$$

- 由中国剩余定理可求出$m^3 (\bmod n_1 n_2 n_3)$。由于$m^3 < n_1 n_2 n_3$，可直接由开立方根得到m。

注：M.Wiener提出一种攻击，可以成功地计算出小于$(1/3)n^{1/4}$的私钥d。

不动点

满足条件$m^e \equiv m \pmod{n}$的m为不动点。显然，不动点对于RSA的安全有一定的威胁，因此，应当减少这样的m。容易证明，RSA体制下的不动点的个数为：

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

因此，为了减少不动点个数，必须使p-1和q-1的因子尽可能少。

如果a是素数，那么素数p=2a+1称为安全素数，当p和q都为安全素数时，不动点的个数最多为四个。

RSA公钥密码的小结

- 第一个实用的公开密钥算法。
- 目前使用最多的一种公钥密码算法。
- RSA的理论基础是数论的欧拉定理。
- RSA的安全性依赖于大数的素因子分解的困难性。
- 密码分析者既不能证明也不能否定RSA的安全性。
- 既能用于加密也能用于数字签名。
- 目前密钥长度1024位是安全的。

离散对数问题

- 设p是素数，g是p的本原元，即$g^0, g^1, g^2, \ldots, g^{p-2}$在 mod p 下产生1到p-1的所有值，所以对任意$y \in [1,\ldots,p-1]$ 有唯一的$x \in [0,\ldots,p-2]$使得 $y \equiv g^x \bmod p$ ，称x为模p下以g为底y的离散对数，即为$x \equiv \log_g^y (\bmod p-1)$ 。
- 当g, p, x已知时，求y比较容易，但如果已知g, p, y，求x则非常困难。

公私钥对生成

- 选择一素数 p ，1024位。
- g是GF(p)的一个本原元.
- 选择随机数 $x \in [0, p-2]$，计算 $y \equiv g^x \bmod p$
- 私有密钥为 x，公开密钥为 y。g 和 p 可由一组用户共享。

注： 已知g, x, p, 计算y是容易的

已知y, g, p, 计算x是困难的

加解密过程

设通信双方为A和B，A和B各选一个秘密整数x_A和x_B，x_A，x_B <p-1。

A计算：$y_A \equiv g^{x_A} \pmod p$

B计算：$y_B \equiv g^{x_B} \pmod p$

y_A和y_B公开。

A要对m加密给B，0≤m≤p-1，A可在[0，p-2]上任选随机数k，计算：$K \equiv y_B^{k} \pmod p$

加解密过程(续)

加密算法：

$$c_1 \equiv g^k \pmod{p}$$

$$c_2 \equiv Km \pmod{p}$$

$$C=(c_1, c_2)$$

解密算法：

先计算：$K \equiv (y_B)^k \pmod{p} \equiv (g^k)^{x_B} \pmod{p}$

$$\equiv (c_1)^{x_B} \pmod{p}$$

$$m \equiv c_2/K \pmod{p}$$

举例说明

假设 p=2579，本原元g=2，私钥x=765，则公钥

$y=2^{765} \bmod 2579=949$，消息m=1299。

A加密过程：选择随机数k=853，并计算$c_1=2^{853} \bmod 2579=435$和$c_2=1299*949^{853} \bmod 2579=2396$。

发送密文（435，2396）。

B解密过程：消息$m=2396*(435^{765})^{-1} \bmod 2579$

=1299。

常用分析方法

- 穷举法和列表法
- 小步大步（**Baby-step Giant-step**）算法
- 指数积分法（**Index Calculus**）

穷举法和列表法

穷举法

计算g，g^2，g^3 … ，直到发现$y=g^x$。

列表法

预先计算所有可能的值g^i，并对有序对(i,g^i)以第二项排序列表，然后给定y，我们对存储的列表实施一个二分查找，直到找到i使得$y=g^i$。

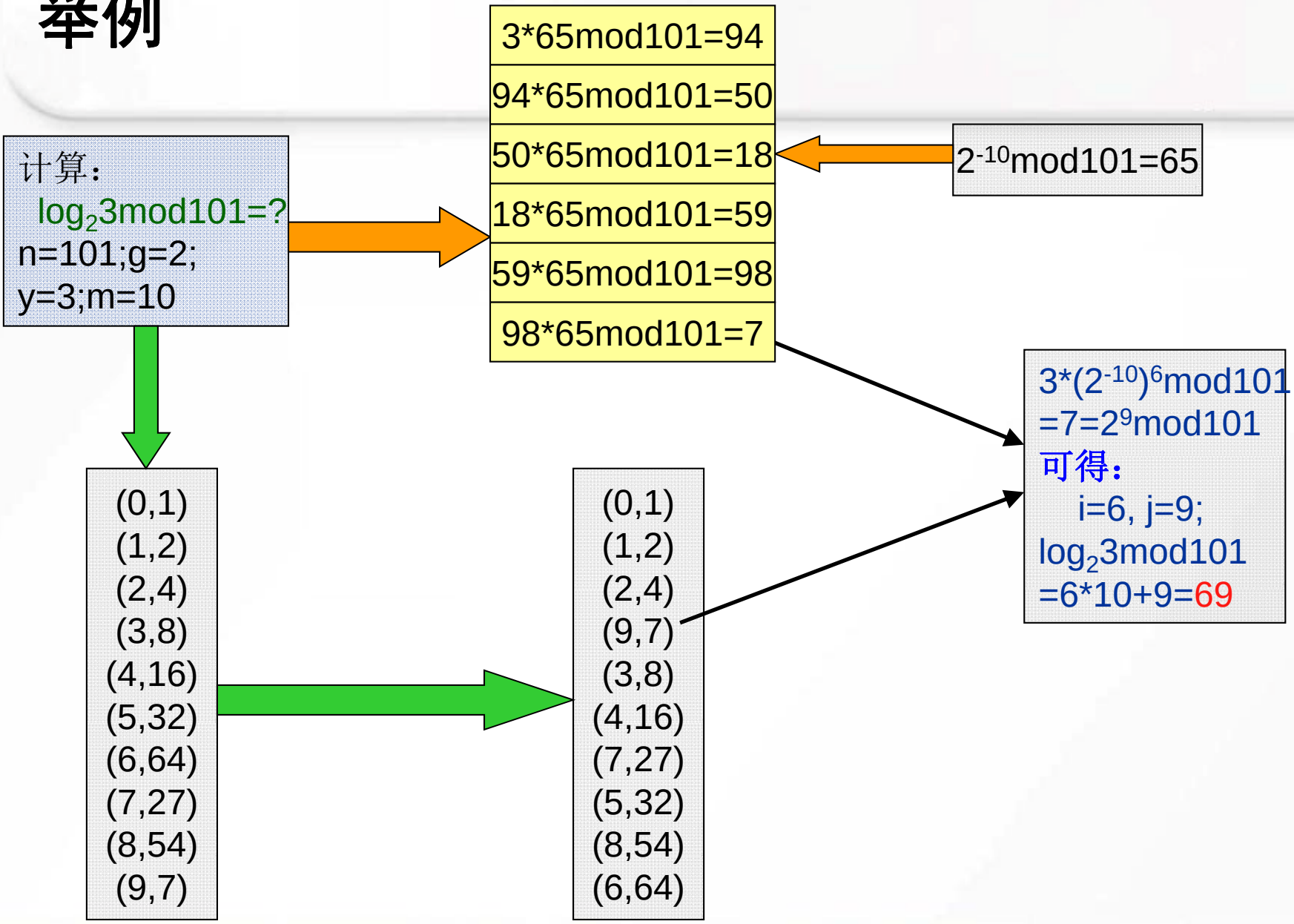
小步大步算法

设G是以g为生成元的循环群，若求解离散对数问题$y \equiv g^x \bmod p$，相当于已知g和y求x。令n为群G的阶，即n=#G，并令m= $\lfloor \sqrt{n} \rfloor$，即m为恰好不超过的整数。当$x=\log_g y$时，可以记$x=m \cdot i+j$，其中$0 \leqslant i<m$，$0 \leqslant j<m$。这样首先对$0 \leqslant j<m$，计算m个g^j，并把它们放在某个有序的查询表中。然后连续计算如下m个值：y，$y \cdot g^{-m}$，$y \cdot g^{-2m}$，$\cdots$，$y \cdot g^{-mi}$，$\cdots$

在第i步，比较$y \cdot g^{-mi}$和g^j，如果相等，则$\log_g y = mi+j$。

注：关于可查询表g^j的构造，最初需要$\sqrt{n}$步。$y \cdot g^{-mi}$和g^j的比较共有$O(\sqrt{n})$次，每一次应该必须在不超过$O(\sqrt{n})$的时间内完成。否则总共就需要O(n)步，这就与穷举法没有区别了。因此查询表g^j的编排必须保证能非常快速地判断一个元素$y \cdot g^{-mi}$是否在表中。比如，在简单的情况，这个判断应只需要$\log_2 m$次比较即可以完成。

举例


3*65mod101=94
94*65mod101=50
50*65mod101=18
18*65mod101=59
59*65mod101=98
98*65mod101=7
2-10mod101=65
计算：
log23mod101=?
n=101;g=2;
y=3;m=10
(0,1)
(1,2)
(2,4)
(3,8)
(4,16)
(5,32)
(6,64)
(7,27)
(8,54)
(9,7)
(0,1)
(1,2)
(2,4)
(9,7)
(3,8)
(4,16)
(7,27)
(5,32)
(8,54)
(6,64)
3*(2-10)6mod101
=7=29mod101
可得：
i=6, j=9;
log23mod101
=6*10+9=69

指数积分法

- 选取因数基底S：如同筛法选取小素数集基底一样，

$$S=\{p_1,p_2,\cdots,p_m\}$$

- 建构同余方程组：对若干随机整数 $k(0\le k\le p)$，计算g^k。

尝试将g^k写成S中的元素幂次的乘积，即 $g^k=\prod_i p_i^{e_i} \bmod p$ ，

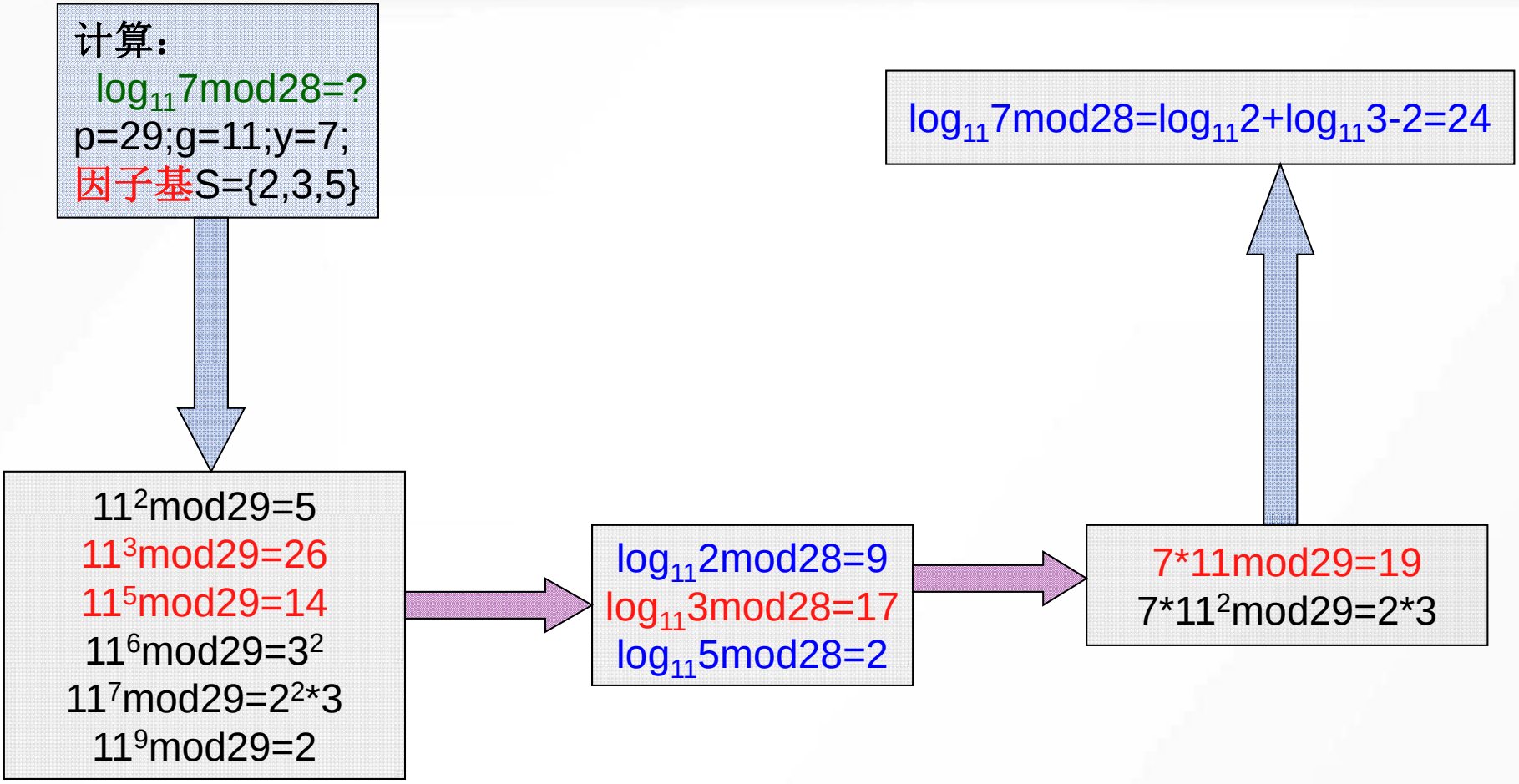
式子两边取离散对数，得 $k=\sum_i e_i \log_g(p_i) \bmod (p-1)$ ，

解 $\log_g(p_i)(p_i\in s)$。

- 计算：随机取整数r，计算值 $yg^r \bmod p$，使得其值可表示为S中元素幂次的乘积，即取离散对数 $yg^r=\prod_i p_i^{d_i} \bmod p$

可得 $x=\log_g(y)=-r+\sum_i d_i \log_g(p_i) \bmod (p-1)$

举例


计算：
$\log_{11}7 \bmod 28 = ?$
p=29;g=11;y=7;
因子基S={2,3,5}
$11^2 \bmod 29 = 5$
$11^3 \bmod 29 = 26$
$11^5 \bmod 29 = 14$
$11^6 \bmod 29 = 3^2$
$11^7 \bmod 29 = 2^2 * 3$
$11^9 \bmod 29 = 2$
$\log_{11}2 \bmod 28 = 9$
$\log_{11}3 \bmod 28 = 17$
$\log_{11}5 \bmod 28 = 2$
$7 * 11 \bmod 29 = 19$
$7 * 11^2 \bmod 29 = 2 * 3$
$\log_{11}7 \bmod 28 = \log_{11}2 + \log_{11}3 - 2 = 24$

离散对数算法的小结

- 基于有限域的离散对数公钥密码又称ElGamal（厄格玛尔）算法。
- ElGamal算法的安全性依赖于计算有限域上的离散对数。
- ElGamal算法的离散对数等同RSA的大数分解问题。
- ElGamal算法既可用于数字签名又可用于加密，但更多地应用在数字签名中。
- ElGamal算法未申请专利。

椭圆曲线密码体制

- 椭圆曲线在代数学和几何学上已广泛研究了150多年之久，有丰富而深厚的理论积累。
- 1985年，Koblitz和Miller提出椭圆曲线密码体制（Elliptic Curve Cryptosystem，简称ECC）
- 椭圆曲线并不是椭圆，之所以称为椭圆曲线是因为它们是用三次方程来表示的，它的一般形式：

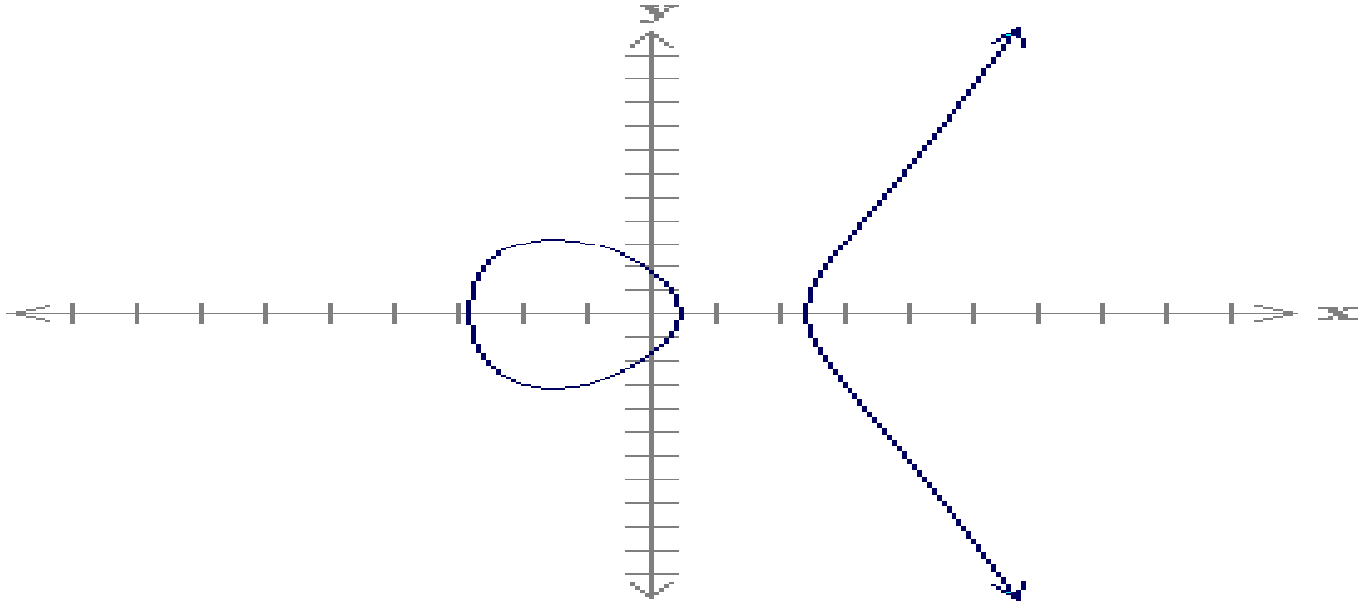
$$y^2 + axy + by = x^3 + cx^2 + dx + e$$

其中a,b,c,d和e是满足某些条件的实数。

- 大多数的椭圆曲线密码系统是在模p或F_2下运算 。
- 椭圆曲线已经逐渐被采用，很可能是一个重要的发展方向。

椭圆曲线的定义

在实数系中，椭圆曲线可定义成所有满足方程式**E:**$y^2=x^3+ax+b$的点**(x,y)**所构成的集合。若方程式没有重复的因式或$4a^3+27b^2\neq 0$，则**E:**$y^2=x^3+ax+b$能成为群**(group)**。例如，椭圆曲线**E:**$y^2=x^3-7x+3$的图形如下所示。



有限群上的椭圆曲线

形式： $y^2=x^3+ax+b(\mathrm{mod}\ p)$

其中：p 是一个素数

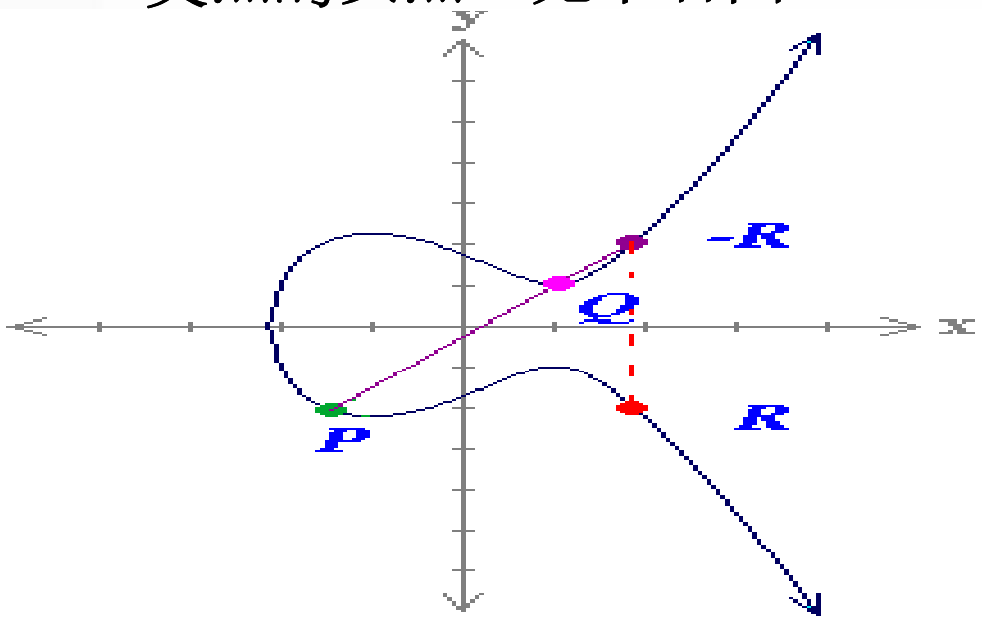
a和b都是小于p的非负整数，且满足：

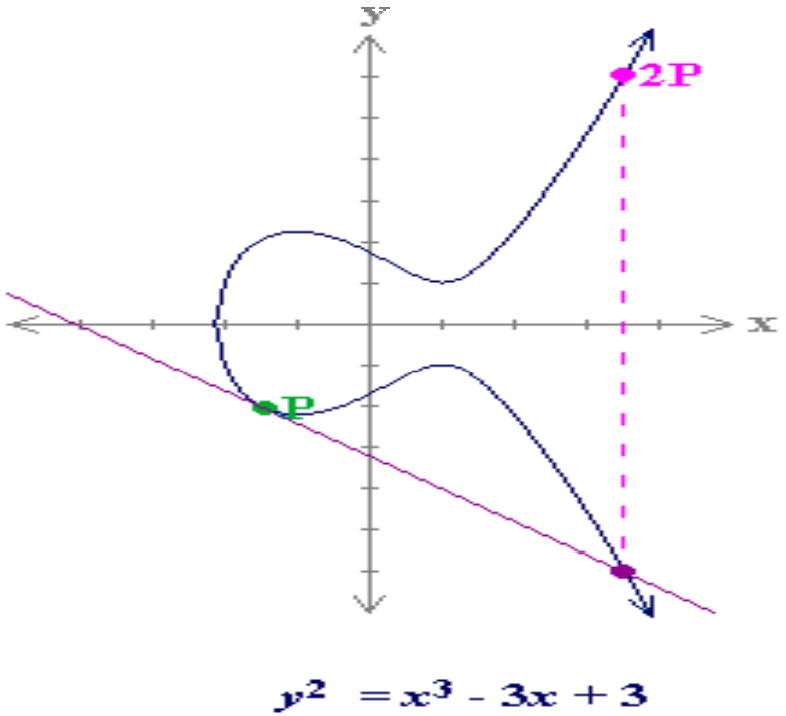
$$4a^3+27b^2(\mathrm{mod}\ p)\neq 0$$

椭圆曲线有一个特殊的点，记为*O*，它并不在椭圆曲线*E*上，此点称为无限远的点(the point at infinity)。Ep(a,b)为在模*p*之下椭圆曲线*E*上所有的点所构成的集合（包括*O*）。点P=（x，y）对*X*座标轴的对称点为-P=（x，-y），而称-P为点P的负点。若nP=0且*n*为最小的正整数，则*n*为椭圆曲线*E*上点P的秩。除了无限远的点*O*之外，椭圆曲线*E*上任何可以生成所有点都可视为是*E*的生成数(generator)，但并不是所有在*E*上的点都可视为生成数。

椭圆曲线的相加运算

- 相异点相加：假设*P*和*Q*是椭圆曲线上两个相异的点且P≠Q。若P+Q=R，则点*R*是经过*P*和*Q*两点的直线与椭圆曲线唯一交点的负点，见下左图。
- 双倍的点：令P+P=2P，则2*P*是经过*P*的切线与椭圆曲线唯一交点的负点。见下右图。





椭圆曲线在模p下的运算规则

- 加法规则：

  (i) 对所有点**P**∈Ep，则P+0=0+P=P，P+（−P）=0

  (ii) 令**P=(x_1,y_1)** ∈Ep 和**Q=(x_2,y_2)** ∈Ep，且**P**≠−Q，则**P+Q=R =(x_3,y_3)** ∈Ep，其中：

$$x_3 = \lambda^2 - x_1 - x_2$$

$$y_3 = \lambda(x_1 - x_3) - y_1$$

$$\lambda = \begin{cases} \dfrac{y_2 - y_1}{x_2 - x_1} & \text{if } P \neq Q \\ \dfrac{3x_1^2 + a}{2y_1} & \text{if } P = Q \end{cases}$$

椭圆曲线在模*p*下的运算规则(续)

(iii) 如果**s**和**t**为整数，则对所有的点**P** ∈Ep而言，

$$(s+t)P=sP+tP$$

- 乘法规则：

(i)如果**k**为整数，则对所有的点**P** ∈Ep而言，

$$kP=P+P+\ldots+P$$

(ii)如果**s**和**t**为整数，则对所有的点**P** ∈Ep而言，

$$s(tP)=(st)P$$

椭圆群的构造

- **Ep(a,b)**表示模**p**下的椭圆曲线上的整数点，再加上O。
- **Ep(a,b)**的生成过程

  (i)对**x=0,1,…,p-1**,计算$x^3+ax+b(\text{mod } p)$

  (ii)对于上一步骤得到的每一结果确定它是否有一个模**p**的平方根，如果没有，则**Ep(a,b)**中没有具有与该结果相应**x**坐标的点。如果有，就有两个平方根**y**和**p-y**，从而点**(x,y)**和**(x,p-y)**都是**Ep(a,b)**的点（如果**y=0**,只有**(x,0)**一个点）。
- 如取**p=23,a=b=1**,有$4*1^3+27*1^2$（**mod23**）**=8**$\neq 0$，则$y^2=x^3+x+1$是椭圆曲线。$E_{23}(1,1)$是一个模23的椭圆群。

椭圆群的构造(续)

例如：x=7.

$y^2=7^3+7+1 \bmod 23=343 \bmod 23=6 \bmod 23 =121 \bmod 23$

$y=11 \bmod 23$ $y= -11 \bmod 23 =12 \bmod 23$

(0，1)	(5，4)	(9，16)	(17，3)
(0，22)	(5，19)	(11，3)	(17，20)
(1，7)	(6，4)	(11，20)	（18，3）
(1，16)	(6，19)	(12，4)	(18 , 20)
(3，10)	(7，11)	(12，19)	(19 , 5)
(3，13)	(7，12)	(13，7)	(19, 18)
(4，0)	(9，7)	(13，16)	

椭圆群运算举例

- P=(3,10), Q=(9,7), 求： R=(x_3,y_3)=P+Q=?

  $\lambda=(7-10)/(9-3)=(-3)/6=(-1)/2=11 \mod 23$

  $x_3=11^2-3-9=109=17 \mod 23$

  $y_3=11*(3-(-6))-10=89=20 \mod 23$

  R=(17,20)

- 计算2P

  $\lambda=(3(3^2)+1)/(2*10)=5/20=1/4=6 \mod 23$

  $x_3=6^2-3-3=30=7 \mod 23$

  $y_3=6*(3-7)-10=-34=12 \mod 23$

  2P=(7,12)

椭圆曲线密码的公私钥对

密钥对生成:

- 选择一个椭圆曲线$E: y^2=x^3+ax+b \pmod p$，构造一个椭圆群Ep(a，b)。
- 在Ep(a,b)中挑选生成元点$G=(x_1, y_1)$，G应使得满足nG＝O的最小的n是一个非常大的素数.
- 选择一个小于n的整数n_A作为其私钥，然后产生其公钥P_A＝$n_A G$;

注：公开的信息：(E，G，n，P_A)

|Ep| 表示椭圆群Ep(a，b)的元素个数，n是|Ep| 的素因子。

$$p+1-2\sqrt{p} \le |Ep| \le p+1+2\sqrt{p}$$

椭圆曲线密码的加密

加密算法:

(1)将m编码成一个数m<p, 在椭圆群Ep(a，b)中选择一点 $P_t=(x_t,y_t)$

(2)在区间[1，n-1]内选取一个随机数k,计算点$(x_1,y_1)= kG$ 。

(3)依据接受方的公钥P_B计算点 $(x_2,y_2)= kP_B$.

(4)计算密文$C=m\cdot x_t + y_t$.

(5)传送加密数据{kG，P_t+kP_B，C}给接受方。

椭圆曲线密码的解密

解密算法:

(1)接受方接受加密数据{kG，P_t+kP_B，C}。

(2)接受方使用自己的私钥n_B作如下计算：

$$P_t+kP_B - n_B(kG)= P_t+k(n_B G) - n_B(kG)=P_t$$

(3)计算$m=(C - y_t)/x_t$，得明文m。

椭圆曲线密码的离散对数问题

- 椭圆曲线密码的离散对数问题是指已知群中的G和P，求方程P=kG中k值的问题。

如对基于F_{23}的椭圆群$y^2=x^3+9x+17$，求P=(4,5)对于G=(16,5)的离散对数，最直接的方法就是计算的G倍数，直到找到P。

G=(16,5) 2G=(20,20) 3G=(14,14) 4G=(19,20)
5G=(13,10) 6G=(7,3) 7G=(8,7) 8G=(12,17) 9G=(4,5)

因此，**P**关于**G**的离散对数是**9**，对于大素数构成的群**E**，这样计算离散对数是不现实的，事实上现在也没有更好的算法来解离散对数问题。

ECC 的小结

- 安全性能更高（160位等同RSA的1024位）

  在保持同等安全的条件下所需的密钥长度(单位为比特)

RSA/DSA	512	768	1024	2048	21000
ECC	106	132	160	211	600

- 计算量小，处理速度快
- 存储空间占用小
- 带宽要求低
- 应用前景非常好，特别在移动通信、无线设备上的应用。

引言

- 公钥密码学的出现极大地促进了网络信息安全理论与技术的发展，从其诞生之日起，就一直是持续的研究热点。

  较好地解决了密钥传递和不可否认的问题。

- 如何保护用户公钥的真实性是公钥密码系统中的一个重要问题。

公钥证书较好地解决了公钥的真实性问题，使得PKI能够为网络用户提供较好的安全服务。但公钥证书的管理和维护需要付出巨大的计算、通信和存储代价。

基于身份加密（IBE）概述

最初是在1984年，由密码学权威Shamir提出了基于身份加密（IBE，Identity－based Encryption）方案的概念，其初衷是简化电子邮件系统中的证书管理。当Alice要给地址是“Bob@abc.com”的Bob发邮件时，她只需用公钥“Bob@abc.com”加密消息即可，而无需通过其他途径获取Bob的公钥证书；Bob收到加密消息之后向一个可信第三方证实自己的身份，可信第三方将Bob的私钥送给Bob，他用该私钥解密消息。在IBE中，公钥可以是任意的关于用户身份的字符串；用户向可信第三方认证自己的身份并获得私钥，该过程可以在收到加密消息之前也可在之后完成；

基于身份的公钥体制的基本思想

基于身份的密码系统中，用户的公钥是一些公开的可以唯一确定用户身份的信息，一般这些信息称为用户的身份(ID)。在实际应用中，用户的身份可以是姓名、电话号码、身份证号码、IP 地址、电子邮件地址等作为公钥。用户的私钥通过一个被称作私钥生成器PKG(Private Key Generator)的可信任第三方进行计算得到。可信第三方可以像PKI中的CA一样为多个用户服务。

在这个系统中，用户的公钥是一些公开的身份信息，其他用户不需要在数据库中查找用户的公钥，也不需要对公钥的真实性进行检验。

实现基于身份的密码系统的前提条件

- 当种子密钥s (PKG的秘密信息---主密钥) 已知时，能容易地计算出任何可能的公钥所对应的私钥。

- 用有种子密钥s产生的一些公私钥对计算种子密钥s是困难问题。

基于身份的加密方案 (举例)

对任意$a, b \in Z$，任意$P, Q \in G_1$，
$e(aP, bQ) = e(P, Q)^{ab}$。

- 初始化

输入安全参数k，生成两个阶为素数q的群G_1, G_2，一个双线性映射$e: G_1 X G_1 \rightarrow G_2$，以及$G_1$的一个生成元P；随机选取$s \in Z_q^*$，计算$P_{pub}=sP$，选择安全的Hash函数$H_1$和$H_2$。

- 公私钥对生成

输入$ID \in \{1,0\}^*$，计算$Q_{ID}= H_1(ID)$，输出私钥$d_{ID}=sQ_{ID}$，其中s是PKG的主密钥。

基于身份的加密方案 (举例)续

- 加密过程

以公钥ID对消息M加密。随机选取$r \in Z_q^*$，计算

$Q_{ID}= H_1(ID)$，$g_{ID}=e(Q_{ID},P_{pub}) \in G_2^*$，

输出密文 $C=(rP,M \oplus H_2(g_{ID}^r))$。

- 解密过程

输入用公钥ID加密的密文C=(U,V)，计算并输出

$M=V \oplus H_2(e(d_{ID},U))$。

注：$U=rP$，$V=M \oplus H_2(g_{ID}^r)$

加解密算法的正确性

$$
\begin{aligned}
V \oplus \mathrm{H}_2(e(d_{ID}, U)) &= m \oplus \mathrm{H}_2((g_{ID})^r) \oplus \mathrm{H}_2(e(sQ_{ID}, rP)) \\
&= m \oplus \mathrm{H}_2((g_{ID})^r) \oplus \mathrm{H}_2(e(Q_{ID}, P)^{rs}) \\
&= m \oplus \mathrm{H}_2((g_{ID})^r) \oplus \mathrm{H}_2(e(Q_{ID}, sP)^{r}) \\
&= m \oplus \mathrm{H}_2((g_{ID})^r) \oplus \mathrm{H}_2((g_{ID})^r) \\
&= m
\end{aligned}
$$

IBE的小结

优点：

- 基于身份的加密方案天生就是密钥托管的；
- 不需要对证书进行管理。

不足：

- 身份确认本来就是一件复杂的事情，尤其用户数量很大。也就是说，IBE适合应用于用户群小的场合。
- 可信第三方如何安全地将用户的私钥送到用户的手中。

公钥密码的优点(与对称密码相比)

- 密钥分发简单。
- 需秘密保存的密钥量减少。
- 可以满足互不认识的人之间私人谈话的保密性要求。
- 可以实现数字签名和认证的功能。

公钥密码的不足(与对称密码相比)

- 公钥密码算法比对称密码算法慢。
- 公钥密码算法提供更多的信息对算法进行攻击，如公钥密码算法对选择明文攻击是脆弱的，尤其明文集比较小时。
- 有数据扩展。
- 公钥密码算法一般是建立在对一个特定的数学难题求解上，往往这种困难性只是一种设想。

本讲的主要内容

- 公钥密码体制的简介
- 背包问题
- **RSA算法**
- **ElGamal算法**
- **ECC算法**
- **IBE算法**

谢　谢！