密码学系列讲座

第 3 课：RSA、环签名、同态加密

lynndell 博士

新火科技 密码学专家 lynndell2010@gmail.com

目录

1. RSA 非对称密码算法

1.1 欧几里得辗转相除法

假设正整数$a \geq b$，求a和b的最大公因数$\gcd(a,b)$？

（1）如果 a mod b=0，则 gcd(a, b)=b；
（2）否则，令 r=a mod b， gcd(a, b)=gcd(b, r)，**递归**直至 r=0，满足情况（1）。
举例：
a=254, b=8;
254=8*h+6, h=31; gcd(254, 8) $r=6 \neq 0$
8=6*h+2,h=1; gcd(8,6) $r=2 \neq 0$
6=2*3+0; gcd(6,2), r=0, gcd(6, 2)=2.
因此，gcd(254, 8)=2.
多项式时间内可解的问题（简单问题、P 问题）
已知大合数 a，求小于 a 的大素数 b，使得 a 与 b 互素，即 gcd(a, b)=1。
该问题能够使用**欧几里得辗转相除法**快速求解。

随机选择一个大素数 b，如果$\gcd(a,b) \neq b$，则 gcd(a,b)=1。

因此，这样的大素数有很多，且很容易找。

1.2 因子分解困难问题

大数的因子分解问题，非常困难！
初始化：随机选择 2 个不同的大素数p,q，计算$n=p \cdot q$
公开n，求p,q？

举例 1：素数$p=5, q=7$；公开数据$n=35$。

已知$n=35$，求p,q？**答：**暴力搜索，令 p,q=2,3…

举例 2：保密数据p,q；公开数据n
公开 n=823912106125050294393 7=p*q
8239121061250502943937 = [redacted] * [redacted]

For $2 \leq i \leq \sqrt{n}$ **do**

For $2 \leq j \leq \sqrt{n}$ **do**

If 8239121061250502943937=$i * j$

Break;
Break;
Output i, j.

如果是 2048bit 的大数$n=2^{2048}-110+1$**，则计算机需要 10 年以上。**

但是，8 年抗日战争胜利以后，会公开 p 和 q。

因子分解 NP 问题：求解时，需要 for 循环，暴力搜索，需要指数时间；
但是，一旦已知解，则可以快速验证解的正确性；

1.3 费马小定理

若 p 为素数，a 是正整数且不能被 p 整除，互素，则

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

证明：正整数 a 与素数 p 互素，则集合$S=\{a,2a,...,(p-1)a\}$中均不可能模 p 同余，即 $ja\neq ka \bmod p$，其中 $j,k\in\{1,...,p-1\}$。因此，集合 S 的模 p 结果恰好就是集合 $T=\{1,2,3,...,p-1\}$中的元素。因此

$$a\times 2a\times ...\times (p-1)a\equiv 1\times 2\times ...\times (p-1)(\bmod p)$$
$$(p-1)!\times a^{p-1}\equiv (p-1)!(\bmod p)$$

由于$\gcd\left((p-1)!,p\right)=1$，所以两边能够消去$(p-1)!$，得到$a^{p-1}\equiv 1\bmod p$

举例 3：p=7，a=3，3 不能被 7 整除，则$3^{7-1}\equiv 1\bmod 7$；

公式推导：$3^6 \bmod 7=(3*3)*(3*3)*(3*3)\bmod 7=9*9*9\bmod 7=729\bmod 7=1$

举例 4：p=11，a=5，5 不能被 11 整除，则$5^{11-1}\equiv 1\bmod 11$；

公式推导：$5^{10}\bmod 11=9765625\bmod 11=1$

1.4 欧拉函数

小于 n 且与 n **互素**的正整数个数，记为$\varphi(n)$。习惯上$\varphi(1)=1$

结论：如果 p 是素数，则$\varphi(p)=p-1$；

推论：如果有两个素数 p,q 且 $p\neq q$，那么对于$n=p\cdot q$，有$\varphi(n)=\varphi(q)\cdot\varphi(p)$，则 $\varphi(n)=(p-1)\cdot(q-1)$；

举例 $\varphi(37)=36$，1,2,3,…,36 均与 37 互素。

举例 $\varphi(35)=\varphi(5)\cdot\varphi(7)=4*6=24$，与 35 互素的 24 个正整数分别为：

1,2,3, 4,6,8,9,11,12,13,16,17,18,19,22,23,24,26,27,29,31,32,33,34

费马小定理：若 p 为素数，a 是正整数且不能被 p 整除，则

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

欧拉定理：对任意互素的 a 和 n，有 $a^{\varphi(n)}\equiv 1 \bmod n$

欧拉定理中：n 可以是合数，a 是素数与 n 互素也可以。
因此，欧拉定理是费马小定理的一般化。

举例 5：n=7，a=3，互素，则 $3^{\varphi(7)} \bmod 7 = 3^6 \bmod 7 = 729 \bmod 7 = 1$

$$3^{\varphi(7)+1} \bmod 7 = 3^{6+1} \bmod 7 = 3^6 * 3^1 \bmod 7 = 1*3^1=3$$

$$3^{\varphi(7)+\varphi(7)+1} \bmod 7 = 3^{6+6+1} \bmod 7 = 729*729*3 \bmod 7 = 1*1*3=3$$

一般化 $3^{k\cdot\varphi(7)+1} \bmod 7=3^1 \bmod 7=3, k=1,2,3...$

再一般化 $3^{k\cdot\varphi(7)+r} \bmod 7 = 3^r \bmod 7, k=1,2,3...$

结论：**指数部分**超过 $\varphi(n)$，则是多余的，所以**指数部分能够模** $\varphi(n)$

举例 8：n=7，a=3，互素，计算 $3^{601} \bmod 7$ 、$3^{1025} \bmod 7$

$3^{601} \bmod 7 = 3^{(600+1) \bmod \varphi(7)} = 3^1 \bmod 7 \bmod 7 = 3$

$3^{1025} \bmod 7 = 3^{170*6+5} \bmod 7 = 3^{170*\varphi(7)+5} \bmod 7 = 3^5 \bmod 7 = 243 \bmod 7 = 5$

欧拉定理：对任意互素的 a 和 n，有 $a^{\varphi(n)}\equiv 1 \bmod n$

欧拉定理扩展：

$$a^{\varphi(n)}\equiv 1 \bmod n$$
$$a^{k\cdot\varphi(n)}\equiv 1 \bmod n$$
$$a^{k\cdot\varphi(n)+1}\equiv a \bmod n$$

1.5 模反元素存在性

如果大素数 a 和大合数 π 互素，则一定可以找到整数 b，使得 $ab\equiv 1 \bmod \pi$，则称 b 是 a 的模反元素。

证明：使用欧拉定理$a^{\varphi(\pi)} \equiv 1 \bmod \pi$，则$a^{1+\varphi(\pi)-1} \equiv 1 \bmod \pi$，则$a \times a^{\varphi(\pi)-1} \equiv 1 \bmod \pi$，则$b = a^{\varphi(\pi)-1}$。

符号替换：$a = sk, b = pk, \pi = \varphi(n)$。$\varphi(n)$为大合数，sk 为大素数，使用欧几里得辗转相除法，快速测试 sk 与大合数$\varphi(n)$**互素**。

模反元素存在性--等价描述：如果大素数 sk 和大合数$\varphi(n)$互素，那么一定可以找到整数 pk，使得$sk \cdot pk \equiv 1 \bmod \varphi(n)$，则称 pk 是 sk 的模反元素，则$pk = sk^{\varphi(\varphi(n))-1}$。

已知$n = p \cdot q$，$\varphi(n) = (p-1)(q-1)$，将合数$(p-1),(q-1)$因子分解，求$\varphi(\varphi(n))$简单。

因此，已知大素数 sk 和大合数$\varphi(n)$互素，可以快速求$pk = sk^{\varphi(\varphi(n))-1}$。

反之，已知pk和n，不知道因子分解$n = p \cdot q$，则无法快速求$\varphi(n)$，从而无法快速计算sk。

无法快速计算的理由：$pk = sk^{\varphi(\varphi(n))-1}$是一个等式，2 个未知数$\varphi(n)$和$sk$。

转化等价：$ab \equiv 1 \bmod \pi \Leftrightarrow pk \cdot sk = i * \varphi(n) + 1$

选择随机数m，计算

$$(m^{sk})^{pk} = (m^{pk})^{sk} = m^{pk \cdot sk} \bmod n = m^{i \cdot \varphi(n)+1} \bmod n = m$$

1.6 RSA 加密与签名

应用场景 1：RSA 公钥加密$(m^{pk})^{sk} = m^{pk \cdot sk} \bmod n = m^{i \cdot \varphi(n)+1} \bmod n = m$

已知：Alice 拥有公开数据是：pk 公钥；保密数据是：sk 私钥。
需求：Bob 需要将保密数据 m 发送给 Alice。
解决方案：

步骤 1：公钥对数据加密：Bob 加密$C \leftarrow m^{pk} \bmod n$，将$C$**广播**给 Alice

步骤 2：私钥对数据解密：接收方 Alice 解密$m \leftarrow (C)^{sk} \bmod n$，从$C$中计算出保密数据 m。

公式推导：$(C)^{sk} \bmod n = (m^{pk} \bmod n)^{sk} \bmod n = m^{pk \cdot sk} \bmod n = m^{i \cdot \varphi(n)+1} \bmod n = m$

应用场景 2：RSA 数字签名$(m^{sk})^{pk} = m^{pk \cdot sk} \bmod n = m^{i \cdot \varphi(n)+1} \bmod n = m$

需求：Alice 需要花费金额 m。
已知：Alice 拥有公开数据是：公钥 pk；保密数据是：私钥 sk。
解决方案：

步骤 1：Alice **签名：**$\sigma \leftarrow m^{sk} \bmod n$，公开$(m,\sigma,pk)$

步骤 2：**任可人**均可**校验**：$m == (\sigma)^{pk} \bmod n$

公式推导：$(\sigma)^{pk} \bmod n = (m^{sk} \bmod n)^{pk} \bmod n = m^{sk \cdot pk} \bmod n = m^{i \cdot \varphi(n)+1} \bmod n = m$

举例：选择 2 个素数 $p=17, q=11$，计算$n = pq = 17*11 = 187$，

计算欧拉函数$\varphi(n) = (p-1)(q-1) = 16*10 = 160$

选择一个随机数e，小于且与欧拉函数$\varphi(n) = 160$互素$\gcd(e,\varphi(n)) = 1$。假设选择$e = 7$满足条件。计算模反元素d，满足$de = 1 \bmod \varphi(n)$且小于$\varphi(n) = 160$，则元素$d = 23$。因为$23*7 = 161 \bmod (160) = 1$。

公钥为$(e,n) = (7,187)$；私钥为$(d,n) = (23,187)$。

对于任意一个消息$m = 88$加密

加密：$c = m^e \bmod n = 88^7 \bmod 187 = 11$

解密：$m = c^d \bmod n = 11^{23} \bmod 187 = 88$

计算过程：

$$
\begin{aligned}
&88 \bmod 187 = 88 \\
&88^2 \bmod 187 = 7744 \bmod 187 = 77 \\
&88^4 \bmod 187 = 77^2 \bmod 187 = 132 \\
&88^7 \bmod 187 = (88*77*132) \bmod 187 = 894432 \bmod 187 = 11
\end{aligned}
$$

反之，对该消息$m = 88$签名

签名：$\sigma = m^d \bmod n = 88^{23} \bmod 187$

验证：$m == \sigma^e$

RSA 算法改进版

$$
\begin{aligned}
&M = m \mid r \\
&\sigma \leftarrow Sig(SK, M) \\
&Valid / Invalid \leftarrow Ver(PK, M, \sigma)
\end{aligned}
$$

$RSA:(m,\sigma,pk)$
$RSA':(m,r,\sigma,pk)$ 安全性更高。

加密类似处理，也是附带随机数 r 。

1.7 正向计算与逆向计算

再看 RSA 公钥加密方案

加密：Bob 计算 $C \leftarrow m^{pk} \bmod n$：从 m 到 C 称为**正向计算（任何人均可计算）；**

解密： Alice 计算 $m \leftarrow (C)^{sk} \bmod n$：从 C 到 m 称为**逆向计算（拥有私钥才可以计算）。**

同理：AES_Enc 加密也称为**正向运算**；AES_Dec 解密也称为**逆向运算。**拥有 AES 对称密钥才可以进行正向运算和逆向运算。

2. 环签名

2.1 基于 RSA 的环签名

真实签名用户为用户 3，其他用户为 1,2,4,5.

用户 1 的公钥为 (e_1,n_1)，用户 2 的公钥为 (e_2,n_2)，

真实签名用户 3 的**公钥和私钥**分别为 $(e_3,n_3),(d_3,n_3)$，

用户 4 的公钥为 (e_4,n_4)，用户 5 的公钥为 (e_5,n_5)。

签名：用户 3 签名：对消息 m，计算哈希值作为对称密钥 $key = hash(m)$。**选择随机数 c_4，**

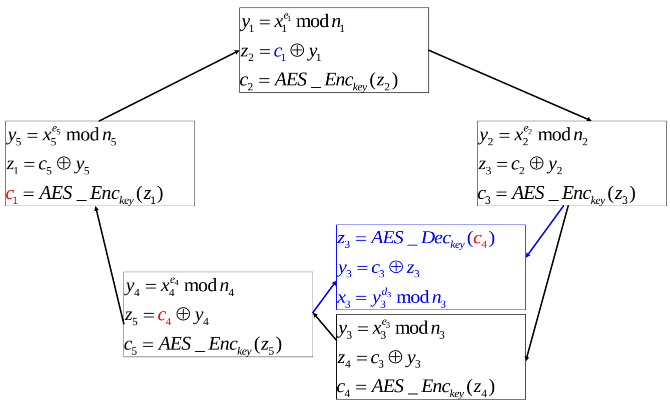
- **RSA 与 AES 正向计算：**选择随机数 x_4，使用用户 4 的公钥 (e_4,n_4) 进行 **RSA 加密** $y_4 = x_4^{e_4} \bmod n_4$；计算 $z_4 = c_4 \oplus y_4$，使用 AES 对称加密 $c_5 = AES\_Enc_{key}(z_4)$；
- **RSA 与 AES 正向计算：**选择随机数 x_5，使用用户 5 的公钥 (e_5,n_5) 进行 **RSA 加密** $y_5 = x_5^{e_5} \bmod n_5$；使计算 $z_1 = c_5 \oplus y_5$，使用 AES 对称加密 $c_1 = AES\_Enc_{key}(z_1)$；
- **RSA 与 AES 正向计算：**选择随机数 x_1，使用用户 1 的公钥 (e_1,n_1) 进行 **RSA 加密** $y_1 = x_1^{e_1} \bmod n_1$；使计算 $z_2 = c_1 \oplus y_1$，使用 AES 对称加密 $c_2 = AES\_Enc_{key}(z_2)$；

- **RSA 与 AES 正向计算：**选择随机数x_2，使用用户 2 的公钥(e_2,n_2)进行 **RSA 加密** $y_2=x_2^{e_2} \bmod n_2$；计算$z_3=c_2\oplus y_2$，使用 AES 对称加密$c_3=AES_Enc_{key}(z_3)$；

- **【RSA 与 AES 正向计算】：**选择随机数x_3，使用用户 3 的公钥(e_3,n_3)进行 **RSA 加密** $y_3=x_3^{e_3} \bmod n_3$；使计算$z_4=c_3\oplus y_3$，用 AES 对称加密$c_4'=AES_Enc_{key}(z_4)$；其中，$c_4=c_4'$概率$1/2^{256}$可忽略。

- **【RSA 与 AES 逆向计算】：AES 逆向计算**随机数$z_4=AES_Dec_{key}(c_4)$，**异或逆向计算**$y_3=c_3\oplus z_4$，用私钥(d_3,n_3)计算 **RSA 逆向计算（解密）**$x_3=y_3^{d_3} \bmod n_3$。要求：知道私钥(d_3,n_3)能够逆向计算。

则环签名为$\sigma=\{(e_1,n_1),(e_2,n_2),(e_3,n_3),(e_4,n_4),(e_5,n_5),c_1,x_1,x_2,x_3,x_4,x_5\}$



验证：对消息m，计算哈希值作为对称密钥$key=hash(m)$。

- **RSA 与 AES 正向计算：**使用公钥(e_1,n_1)对x_1进行 **RSA 加密** $y_1=x_1^{e_1} \bmod n_1$；基于c_1使用 AES 对称加密$c_2=AES_Enc_{key}(c_1\oplus y_1)$；

- **RSA 与 AES 正向计算：**使用公钥(e_2,n_2)对x_2进行 **RSA 加密** $y_2=x_2^{e_2} \bmod n_2$；基于c_2使用 AES 对称加密$c_3=AES_Enc_{key}(c_2\oplus y_2)$；

- **RSA 与 AES 正向计算：**使用公钥(e_3,n_3)对x_3进行 **RSA 加密** $y_3=x_3^{e_3} \bmod n_3$；基于c_3使用 AES 对称加密$c_4=AES_Enc_{key}(c_3\oplus y_3)$；分析：该正向计算是正确的。

- **RSA 与 AES 正向计算：**使用公钥(e_4,n_4)对x_4进行 **RSA 加密** $y_4=x_4^{e_4} \bmod n_4$；基于

c_3 使用 AES 对称加密 $c_5 = AES\_Enc_{key}(c_4 \oplus y_4)$；

- **RSA 与 AES 正向计算：** 使用公钥 (e_5, n_5) 对 x_5 进行 **RSA 加密** $y_5 = x_5^{e_5} \bmod n_5$；基于 c_5 使用 AES 对称加密 $c_1' = AES\_Enc_{key}(c_5 \oplus y_5)$；

校验： $c_1 = c_1'$。

分析：上述 5 个用户均知道自己的私钥，所以在关键步骤上均能够进行逆向计算。所以上述 5 个签名方均能够对某个消息生成正确的签名。环签名具有伪造性。因为签名使用了多个公钥，每个拥有对应私钥的用户均能生成正确的签名。
环签名具有伪造性，添加**唯一标识符的环签名，才是好的环签名。称为可链接的环签名。**

RSA2048 比特，效率较低，尽量不用。通常使用椭圆曲线离散对数。

2.2. 基于离散对数的环签名

2.2.1 预备知识

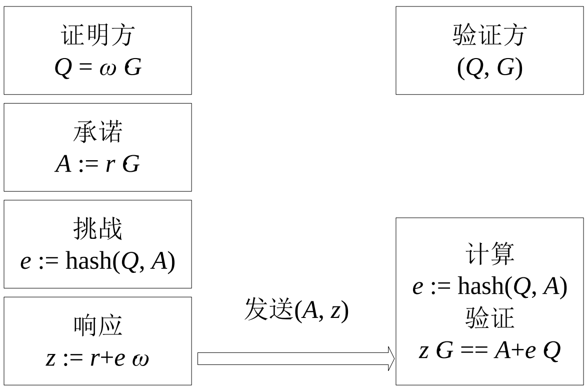
模式 1：非交互 Sigma 零知识证明
系统生成元为 G，秘密为 ω，与公开参数 H，满足离散对数关系 $H = \omega \cdot G$。
1：（**承诺**）**证明方** P 选择随机数 r，计算 $A = r \cdot G$；

2：（**挑战**）**证明方** P 计算随机数 $e = Hash(H, A)$；

3：（**响应**）**证明方** P 计算 $z = r + e \cdot \omega$，发送**承诺**和**响应** A, z （数据量大）；

4：（**验证**）**验证方** V 计算 $e = Hash(H, A)$，校验 $z \cdot G == A + e \cdot H$ 。

公式推导过程： $z \cdot G = (r + e \cdot \omega) \cdot G = A + e \cdot H$



模式 2：非交互 Sigma 零知识证明

系统生成元为 G，秘密为 ω_1，与公开参数 H_1，满足离散对数关系 $H_1 = \omega_1 \cdot G$。

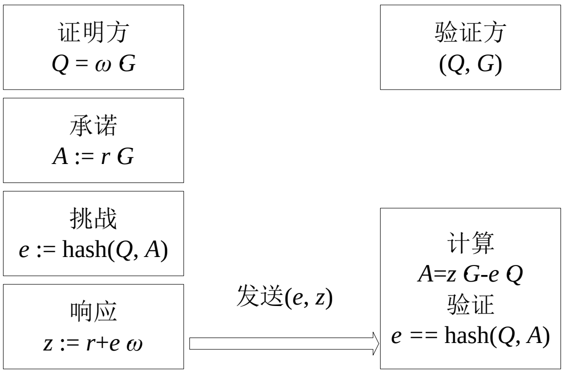
1：（**承诺**）**证明方** P 选择随机数 r_1，计算 $A_1 = r_1 \cdot G$；

2：（**挑战**）**证明方** P 计算随机数 $e_1 = Hash(H_1, A_1)$；

3：（**响应**）**证明方** P 计算 $z_1 = r_1 + e_1 \cdot \omega_1$，发送**挑战**和**响应** (e_1, z_1)（**数据量少**）；

4：（**验证**）**验证方** V 重新计算**承诺** $A_1 = z_1 \cdot G - e_1 \cdot H$，校验 $e_1 == Hash(H_1, A_1)$。

公式推导过程：$z_1 \cdot G - e_1 \cdot H_1 = (r_1 + e_1 \cdot \omega_1) \cdot G - e_1 \cdot H_1 = A_1$



Pedersen 承诺与打开承诺的扩展

系统生成元为 G 与公开参数 H_2，证明方**不知道离散对数** ω_2，**离散对数关系** $H_2 = \omega_2 \cdot G$。

1：（**承诺**）**证明方** P 选择 2 个随机数 r_2, s_2，计算 **Pedersen** 承诺 $A_2 = r_2 \cdot G + s_2 \cdot H_2$；

2：（**挑战**）**证明方** P 计算挑战 $e_2 = Hash(H_2, A_2)$；

3：（**响应**）**证明方** P，没有响应，而是发送**打开承诺和挑战** (r_2, s_2, e_2)

4：(**验证**)**验证方** V 重新计算 Pedersen 承诺 $A_2 = r_2 \cdot G + s_2 \cdot H_2$，校验 $e_2 == Hash(H_2, A_2)$。

验证方认可：Pedersen 承诺是正确打开的。

分析：该过程与 Sigma 协议高度相似，承诺是一个椭圆曲线点、挑战是一个随机数、验证也在计算一个椭圆曲线点、校验等式一样。**唯一区别：响应部分发的数据量更大。**

如果证明方发送 (e_1, z_1) 和 (r_2, s_2, e_2)，则验证方很可能不知道自己在验证 Sigma 零知识证明与承诺协议。

去掉这点差异，模式 2 非交互式 Sigma 零知识证明与 Pedersen 承诺完美耦合，就是环签名。

2.2.2 环签名协议

密钥生成： n 个用户，每个用户记为 i，$i = 1, \ldots, n$。假设 n=5，需要签名的用户排名第 3。

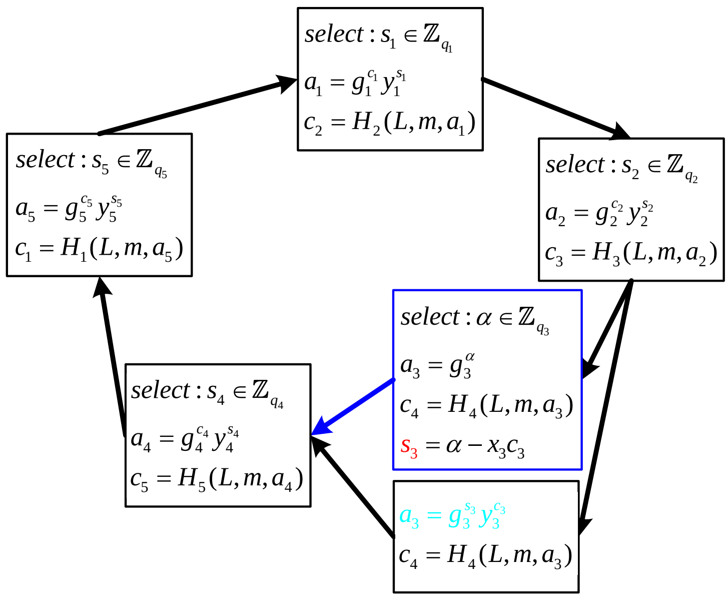
群 $\langle g_i \rangle$ 的生成元为 g_i，阶为素数 q_i。这 5 个用户的私钥为 x_i，公钥为 y_i，其中 $y_i = g_i^{x_i} \bmod p_i$。集合 $L=\{y_i, p_i, q_i, g_i\}$。n+1 个哈希函数分别为 $H:\{0,1\}^* \to \{0,1\}^l, H_i:\{0,1\}^* \to \mathbb{Z}_{q_i}$。

签名：用户 3 如下计算

1. 选择随机数 $s_1 \in \mathbb{Z}_{q_1}$，计算 **Pedersen 承诺** $a_1 = g_1^{c_1} y_1^{s_1}$，计算**挑战** $c_2 = H_2(L, m, a_1)$；
2. 选择随机数 $s_2 \in \mathbb{Z}_{q_2}$，计算 **Pedersen 承诺** $a_2 = g_2^{c_2} y_2^{s_2}$，计算**挑战** $c_3 = H_3(L, m, a_2)$；
3. 选择随机数 $\alpha \in \mathbb{Z}_{q_3}$，计算 **Sigma 协议承诺** $a_3 = g_3^{\alpha}$，计算**挑战** $c_4 = H_4(L, m, a_3)$；
4. 选择随机数 $s_4 \in \mathbb{Z}_{q_4}$，计算 **Pedersen 承诺** $a_4 = g_4^{c_4} y_4^{s_4}$，计算**挑战** $c_5 = H_5(L, m, a_4)$；
5. 选择随机数 $s_5 \in \mathbb{Z}_{q_5}$，计算 **Pedersen 承诺** $a_5 = g_5^{c_5} y_5^{s_5}$，计算**挑战** $c_1 = H_1(L, m, a_5)$；

响应：

1. 第 1 步的 Pedersen 打开承诺为 c_1, s_1，
2. 第 2 步的 Pedersen 打开承诺为 s_2
3. 第 3 步 **Sigma 协议中的响应** $s_3 = \alpha - x_3 c_3$；
4. 第 4 步的 Pedersen 打开承诺为 s_4；
5. 第 5 步的 Pedersen 打开承诺为 s_5；



环签名为 $(c_1, s_1, s_2, s_3, s_4, s_5)$。注意：n 越大，签名越长。

验证：基于环签名$(c_1,s_1,s_2,s_3,s_4,s_5)$，进行以下计算

1. 重新计算 **Pedersen 承诺** $a_1=g_1^{s_1}y_1^{c_1}$，计算**挑战** $c_2=H_2(L,m,a_1)$
2. 重新计算 **Pedersen 承诺** $a_2=g_2^{s_2}y_2^{c_2}$，计算**挑战** $c_3=H_3(L,m,a_2)$
3. 重新计算 **Pedersen 承诺(Sigma 承诺)** $a_3=g_3^{s_3}y_3^{c_3}=g_3^{\alpha-x_3c_3}y_3^{c_3}=g_3^{\alpha}$，计算**挑战** $c_4=H_4(L,m,a_3)$
4. 重新计算 **Pedersen 承诺** $a_4=g_4^{s_4}y_4^{c_4}$，计算**挑战** $c_5=H_5(L,m,a_4)$
5. 重新计算 **Pedersen 承诺** $a_5=g_5^{s_5}y_5^{c_5}$，计算**挑战** $c_1'=H_1(L,m,a_5)$

   重新计算承诺的形式一样，没有泄露用户公钥信息。

   校验 $c_1==c_1'$。

分析：公式等式 3 是标准的 Sigma 检测，其他是 Pedersen 承诺与打开承诺。
用户 3 能够产生上述签名，其余 4 个用户知道自己的私钥，也能产生上述环签名。因此，上述环签名具有伪造性。**很不好!**

门罗币添加**密钥镜像/唯一标识符/防双花标识**，能够去掉环签名的伪造性，让每个用户生成的环签名与一个**唯一标识符**相对应。

2.3 门罗币环签名

初始化：椭圆曲线群为$\mathbb{G}$，生成元为G，阶为n。椭圆曲线点的横坐标和纵坐标的取值空间为F_q，基域为$\mathbb{F}_q$。哈希函数$H_s:\{0,1\}^*\to\mathbb{F}_q, H_p:\mathbb{G}\to\mathbb{G}$

密钥生成：私钥$x\in[1,n-1]$，计算公钥$P=x\cdot G$，计算**密钥镜像** $I=x\cdot H_p(P)$。

签名：使用 n 个 UTXO，假如 n=5。其中 4 个 UTXO 是其他用户的，用户需要花费的真实 UTXO 是第 3 个。每个 UTXO 对应的公钥为$P_i, i=1,2,3,4,5$。

这 5 个 UTXO 记为消息m，

1. 选择随机数q_1,w_1，计算 2 个 **Pedersen 承诺** $L_1=q_1G+w_1P_1, R_1=q_1H_p(P_1)+w_1I$；
2. 选择随机数q_2,w_2，计算 2 个 **Pedersen 承诺** $L_2=q_2G+w_2P_2, R_2=q_2H_p(P_2)+w_2I$；
3. 选择随机数q_3，计算 2 个 **Sigma 协议的承诺** $L_3=q_3G, R_3=q_3H_p(P_3)$；

4. 选择随机数 q_4, w_4，计算 2 个 **Pedersen 承诺** $L_4 = q_4 G + w_4 P_4, R_4 = q_4 H_p(P_4) + w_4 I$；

5. 选择随机数 q_5, w_4，计算 2 个 **Pedersen 承诺** $L_5 = q_5 G + w_5 P_5, R_5 = q_5 H_p(P_5) + w_5 I$；

计算**挑战** $c = H_s(m, L_1, ..., L_5, R_1, ..., R_5)$

计算**响应**：

1. 第 4 步的 Pedersen 打开承诺 w_4, q_4；

2. 第 5 步的 Pedersen 打开承诺 w_5, q_5；

3. 第 1 步的 Pedersen 打开承诺 w_1, q_1；

4. 第 2 步的 Pedersen 打开承诺 w_2, q_2；

5. 第 3 步 Sigma 协议中的响应 $\tilde{w}_3 = c - (w_1 + w_2 + w_4 + w_5), \tilde{q}_3 = q_3 - \tilde{w}_3 \cdot x$；

令环签名为 $\sigma = \{I, w_1, w_2, \tilde{w}_3, w_4, w_5, q_1, q_2, \tilde{q}_3, q_4, q_5\}$。注意：n 越大，签名越长。

验证：验证方基于环签名 $\sigma = \{I, w_1, w_2, \tilde{w}_3, w_4, w_5, q_1, q_2, \tilde{q}_3, q_4, q_5\}$，计算

1. 重新计算 **Pedersen 承诺** $L_1 = q_1 G + w_1 P_1, R_1 = q_1 H_p(P_1) + w_1 I$，

2. 重新计算 **Pedersen 承诺** $L_2 = q_2 G + w_2 P_2, R_2 = q_2 H_p(P_2) + w_2 I$，

3. 重新计算 **Pedersen 承诺(Sigma 协议的承诺)** $L_3' = \tilde{q}_3 G + \tilde{w}_3 P_3, R_3' = \tilde{q}_3 H_p(P_3) + \tilde{w}_3 I$，

4. 重新计算 **Pedersen 承诺** $L_4 = q_4 G + w_4 P_4, R_4 = q_4 H_p(P_4) + w_4 I$，

5. 重新计算 **Pedersen 承诺** $L_5 = q_5 G + w_5 P_5, R_5 = q_5 H_p(P_5) + w_5 I$，

计算承诺的形式一样，没有泄露用户信息。

公式推导：
$$L_3' = \tilde{q}_3 G + \tilde{w}_3 P_3 = (q_3 - \tilde{w}_3 \cdot x) G + \tilde{w}_3 P_3 = q_3 G = L_3$$
$$R_3' = \tilde{q}_3 H_p(P_3) + \tilde{w}_3 I = (q_3 - \tilde{w}_3 \cdot x) H_p(P_3) + \tilde{w}_3 I = q_3 H_p(P_3) = R_3$$

分析密钥镜像使用 5 次，其中第 3 次必须使用密钥镜像对应的私钥 x，否则验证会失败。

校验：$w_1 + w_2 + \tilde{w}_3 + w_4 + w_5 == H_s(m, L_1, ..., L_5, R_1, ..., R_5)$

链接：如果私钥 x 使用第 2 次，则**密钥镜像** $I = x \cdot H_p(P_3)$ 一定出现第 2 次，则双重花费。

分析：

（1）每次支付仅使用一个 UTXO，而不能批量使用 UTXO，应该使用**门限环签名**，每次支付使用多个密钥镜像，即使用多个私钥计算响应，则能够实现多个 UTXO 批量支付，节约存储 gas。

（2）**密钥镜像**唯一对应私钥，使用某个密钥镜像，就必须要知道对应的私钥 x 。如果不知道，则不能计算出正确的响应 r_3 。则不能花费该 UTXO。

（3）**密钥镜像是唯一标识符**，用于防止双重花费攻击。已知密钥镜像，无法在多项式时间内计算公钥 P_3 。密钥镜像是唯一标识符，UTXO 和公钥 Pi 可以被任意用户使用多次。

（4）可追踪的环签名：拥有追踪密钥的管理员能够追踪签名方的真实身份。

3. Paillier 同态加密及其应用

3.1 预备知识

3.1.1 费马小定理

若 p 为素数，a 是正整数且不能被 p 整除，互素，则

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

举例 1： p=7，a=3，则 $3^{7-1} \equiv 1 \bmod 7$ ； $3^6 \bmod 7 = 9*9*9 \bmod 7 = 729 \bmod 7 = 1$

举例 2： p=11，a=5，则 $5^{11-1} \equiv 1 \bmod 11$ ； $5^{10} \bmod 11 = 9765625 \bmod 11 = 1$

证明： 正整数 a 与素数 p 互素，则集合 $S = \{a, 2a, ..., (p-1)a\}$ 中均不可能模 p 同余，即 $ja \neq ka \bmod p$ ，其中 $j, k \in \{1, ..., p-1\}$ 。因此，集合 S 的模 p 结果恰好就是集合 $T = \{1, 2, 3, ..., p-1\}$ 中的元素。因此，以下等式成立

$$a \times 2a \times ... \times (p-1)a \equiv 1 \times 2 \times ... \times (p-1) (\bmod p)$$
$$(p-1)! \times a^{p-1} \equiv (p-1)! (\bmod p)$$

由于 $\gcd\left((p-1)!, p\right) = 1$，所以两边能够消去 $(p-1)!$，得到 $a^{p-1} \equiv 1 \bmod p$ 。

3.1.2 欧拉函数

小于 n 且与 n 互素的正整数个数称为欧拉函数 $\varphi(n)$ 。习惯上 $\varphi(1) = 1$

- 如果 n 是素数，则 $\varphi(n) = n - 1$
- 如果有两个素数 p, q 且 $p \neq q$ ，那么对于 $n = p \cdot q$ ，有 $\varphi(n) = \varphi(q) \cdot \varphi(p)$

  举例： $\varphi(37) = 36$ ， $\varphi(35) = \varphi(5) \cdot \varphi(7) = 4*6 = 24$

- **素数幂 p^r，欧拉函数为** $\varphi(p^r)=p^{r-1}(p-1)$

举例 $\varphi(37)=36$，1,2,3,…,36 均与 37 互素。

举例 $\varphi(35)=\varphi(5)\cdot\varphi(7)=4*6=24$，小于 35 且与 35 互素的 24 个正整数分别为：

1,2,3,4,6,8,9,11,12,13,16,17,18,19,22,23,24,26,27,29,31,32,33,34

举例 $p^r=2^3$，$\varphi(2^3)=2^{3-1}(2-1)=4$，小于 8 且与 8 互素的 4 个正整数为：1,3,5,7。

欧拉定理：对任意互素的 a 和 n，有 $a^{\varphi(n)}\equiv 1 \bmod n$

因此，欧拉定理是费马小定理的一般化。
欧拉定理中：n 可以是合数，a 是素数与 n 互素也可以。

举例 5：n=7，a=3，互素，则 $3^{\varphi(7)} \bmod 7=3^6 \bmod 7=729 \bmod 7=1$

$$3^{\varphi(7)+1} \bmod 7=3^{6+1} \bmod 7=3^6*3^1 \bmod 7=1*3^1=3$$

$$3^{\varphi(7)+\varphi(7)+1} \bmod 7=3^{6+6+1} \bmod 7=729*729*3 \bmod 7=1*1*3=3$$

一般化 $3^{k\cdot\varphi(7)+1} \bmod 7=3^1 \bmod 7=3, k=1,2,3...$

再一般化 $3^{k\cdot\varphi(7)+r} \bmod 7=3^r \bmod 7, k=1,2,3...$

结论：指数部分超过 $\varphi(n)$，则是多余的，所以指数部分能够**模** $\varphi(n)$

举例 8：n=7，a=3，互素，计算 $3^{601} \bmod 7$ 、$3^{1025} \bmod 7$

$3^{601} \bmod 7=3^{100*\varphi(7)+1} \bmod 7=3^{601 \bmod \varphi(7)+1} \bmod 7=3^1 \bmod 7=3$

$3^{1025} \bmod 7=3^{170*6+5} \bmod 7=3^{170*\varphi(7)+5} \bmod 7=3^5 \bmod 7=243 \bmod 7=5$

欧拉定理扩展：

$$a^{\varphi(n)}\equiv 1 \bmod n$$
$$a^{k\cdot\varphi(n)+1}=a \bmod n$$

剩余类（同余类）

设模为 n，根据**余数**可将所有的整数分为 n 类，分别为 $[0],[1],...,[n-1]$。所有与整数 a 模 n 同余的整数构成的集合叫做模 n 的一个**剩余类**，记作 $[a]$。a 称为**剩余类** $[a]$ 的一个代表元。

小于 n 且与 n 互素的正整数个数称为**欧拉函数** $\varphi(n)$，且欧拉定理 $a^{\varphi(n)}\equiv 1 \bmod n$ 成立。

3.1.3Carmichael 函数

定义 Carmichael 函数：函数 $\lambda=\lambda(n)$，对于一个**剩余类** $a\in Z_n^*, a^{\lambda(n)}\equiv 1 \bmod n$ 均成立，且 $\lambda(n)$ 是满足该等式的**最小数**。

Carmichael 定理：

$$
\lambda(p^\alpha)=\begin{cases}\varphi(p^\alpha), if\,(p^r=2,3^r,4,5^r,7^r,11^r,\dots)\\ \dfrac{1}{2}\varphi(p^\alpha), if\,(p^r=8,16,32,64,\dots)\end{cases}
$$

$$
\lambda(p_1^{\alpha_1}\dots p_k^{\alpha_k})=lcm\left(\lambda(p_1^{\alpha_1}),\dots,\lambda(p_k^{\alpha_k})\right)
$$

$$
if\,(a\,|\,b), then\left(\lambda(a)\,|\,\lambda(b)\right)
$$

$$
\lambda\left(lcm(a,b)\right)=lcm\left(\lambda(a),\lambda(b)\right)
$$

举例：令 $n=360$，寻找**最小数** $\lambda=\lambda(360)$，使得任意与 360 互素的剩余类 a 满足等式 $a\in Z_{360}^*, a^{\lambda(360)}\equiv 1 \bmod 360$。

$a^{\lambda(360)}\equiv 1 \bmod 360$ 等价于 $360\,|\,a^{\lambda(360)}-1$

$$
360=2^3\times 3^2\times 5
$$

$$
\lambda(360)=lcm\left(\lambda(2^3),\lambda(3^2),\lambda(5)\right)=lcm(2,6,4)=12
$$

定理证明：https://brilliant.org/wiki/carmichaels-lambda-function/?subtopic=modular-arithmetic&chapter=eulers-theorem

PROOF

Let p be an odd prime. An element of order $\phi(p^\alpha)$ in $(\mathbb{Z}/p^\alpha\mathbb{Z})^*$ is called a primitive root. The wiki on primitive roots contains the full classification of integers n for which there is a primitive root mod n; in particular, there is a primitive root g mod n when n is an odd prime power. Since the smallest positive integer power of g that is congruent to 1 is $g^{\phi(p^\alpha)}$, this shows that $\lambda(p^\alpha)\ge\phi(p^\alpha)$. Since $\lambda(p^\alpha)\le\phi(p^\alpha)$ from the discussion in the previous section, this shows that they are equal.

When $p=2$, the primitive roots wiki shows that $\lambda(2^\alpha)\big|2^{\alpha-2}$ for $\alpha\ge 3$, and an easy induction shows that $5^{2^{\alpha-3}}\equiv 1+2^{\alpha-1}\pmod{2^\alpha}$, so the order of 5 does not divide $2^{\alpha-3}$, but it is a power of 2, so it is $2^{\alpha-2}$. This shows that $\lambda(2^\alpha)=2^{\alpha-2}=\frac{1}{2}\phi(2^\alpha)$.

The last statement is a straightforward application of the Chinese remainder theorem. In particular, if $n=p_1^{\alpha_1}\cdots p_k^{\alpha_k}$, then for any choice of primitive roots g_i $\bmod p_i^{\alpha_i}$ $\big($and $g_i=5$ if $p_i^{\alpha_i}$ is a power of 2 greater than 4$\big)$, there is a unique element g mod n that is congruent to each of the g_i mod $p_i^{\alpha_i}$, and it is easy to show that the order of g equals the LCM of the $\lambda(p_i^{\alpha_i})$. On the other hand, a similar Chinese remainder theorem argument shows that any element raised to that LCM must be 1 mod n since it is 1 modulo all the prime powers. $\square$

设 p,q 为长度相等的大素数，计算 $n=pq$，欧拉函数 $\varphi(n)=(p-1)(q-1)$，Carmichael 函数 $\lambda(n)=lcm(p-1)(q-1)$。欧拉函数性质 $\varphi(n^2)=n\varphi(n)$。

关键结论：对于 $n=pq$，$\lambda=\lambda(n)=lcm(p-1,q-1)$。对于 $r\in Z_{n^2}^*$，以下两个等式成立

$$
(1): r^\lambda=1 \bmod n
$$

$$
(2): r^{n\lambda}=1 \bmod n^2
$$

（1）证明：

$$\lambda = \lambda(n) = lcm(p-1, q-1)$$
$$p-1 | \lambda, q-1 | \lambda$$
$$\lambda = k_1(p-1) = k_2(q-1)$$

根据**费马小定理**

- $r^{\lambda} = r^{k_1(p-1)} = (r^{(p-1)})^{k_1} \equiv 1 \bmod p$，所以$r^{\lambda} - 1 \equiv 0 \bmod p$
- $r^{\lambda} = r^{k_2(q-1)} = (r^{(q-1)})^{k_2} = 1 \bmod q$，所以$g^{\lambda} - 1 \equiv 0 \bmod q$

所以

$$r^{\lambda} - 1 \equiv 0 \bmod pq$$
$$r^{\lambda} - 1 \equiv 0 \bmod n$$

所以：$r^{\lambda} \equiv 1 \bmod n$。等式 1 成立。

（2）证明：

$n = p \cdot q, n^2 = p^2 \cdot q^2$，根据 Carmichael 定理

$$\lambda(n^2) = lcm\left(\lambda(q^2), \lambda(p^2)\right) = lcm\left(\varphi(q^2), \varphi(p^2)\right) = lcm\left(q \cdot \varphi(q), p \cdot \varphi(p)\right)$$
$$= lcm\left(q(q-1), p(p-1)\right) = pq\left(lcm(p-1, q-1)\right) = n\lambda(n)$$

$$r^{\lambda} \equiv r^{\lambda(n)} \equiv 1 \bmod n$$
$$r^{\lambda(n^2)} \equiv 1 \bmod n^2$$
$$r^{n\lambda(n)} \equiv 1 \bmod n^2$$

所以：关键等式 $r^{n\lambda} \equiv 1 \bmod n^2$。所以等式 2 成立。

二项式泰勒展开

$$(1+n)^x = \sum_{k=0}^{x} C_x^k n^k = 1 + nx + \frac{x(x-1)}{2!} n^2 + \ldots + \frac{x(x-1) \cdot \ldots \cdot (x-n+1)}{n!} n^n + \ldots$$

二项式泰勒展开模(n^2)

关键等式 $(1+n)^x \bmod n^2 = 1 + nx \bmod n^2$

3.2Paillier 同态加密

密钥生成：生成两个长度相同的大素数 p, q 且 $p \neq q$，满足 $\gcd\left(pq, (p-1)(q-1)\right) = 1$。该性质确保这两个素数长度相同；计算 $n = pq$，最小公倍数 $\lambda = lcm(p-1, q-1)$；分式除法函数 $L(y) = (y-1)/n$；选择正整数 $g = 1 + n \in Z_{n^2}^*$，使得 $\mu = \left(L(g^{\lambda} \bmod n^2)\right)^{-1} \bmod n$ 存

在。**公钥为 n，私钥为 p,q。**

如果 p,q 长度相等，则密钥生成步骤能够化简为：$g=n+1,\lambda=\varphi(n),\mu=\varphi(n)^{-1}\bmod n$，其中 $\varphi(n)=(p-1)(q-1)$；如果 p,q 足够大，则 μ 的计算时间较长，推荐使用该方法。

加密：消息 $m\in Z_n$，选择随机数 $r\in Z_n^*$，计算密文 $c:=g^m\cdot r^n\bmod n^2$。

解密：输入密文 $c\in Z_{n^2}$，如下计算解密 $m:=L(c^\lambda\bmod n^2)\cdot\mu\bmod n$。注意解密使用 $\lambda=lcm(p-1,q-1)$ 等价于使用私钥 p,q。

公式推导：

$$
\begin{aligned}
&c=g^m\cdot r^n\bmod n^2\\
&c^\lambda\bmod n^2=g^{\lambda m}\cdot r^{\lambda n}\bmod n^2=g^{\lambda m}\cdot 1\bmod n^2=(1+n)^{m\lambda}\bmod n^2=1+nm\lambda\bmod n^2\\
&g^\lambda\bmod n^2=(1+n)^\lambda\bmod n^2=1+n\lambda\bmod n^2\\
&L(c^\lambda\bmod n^2)=\frac{c^\lambda\bmod n^2-1}{n}=m\lambda\bmod n^2\\
&L(g^\lambda\bmod n^2)=\frac{g^\lambda\bmod n^2-1}{n}=\lambda\bmod n^2\\
&m=\frac{L(c^\lambda\bmod n^2)}{L(g^\lambda\bmod n^2)}\bmod n
\end{aligned}
$$

同态性：给定两个密文 $c_1,c_2\in Z_{n^2}$，其中 $c_1=Enc_{pk}(m_1),c_2=Enc_{pk}(m_2)$

- 定义**密文之间的加法运算** $\oplus$

  $$c_1\oplus c_2=c_1c_2\bmod n^2=(g^{m_1}\cdot r_1^n\bmod n^2)(g^{m_2}\cdot r_2^n\bmod n^2)=g^{m_1+m_2}\cdot(r_1r_2)^n\bmod n^2$$

  因此，$c_1\oplus c_2=Enc_{pk}(m_1+m_2\bmod n)$。

- 给定 $a\in Z_n,c=Enc_{pk}(m)$ 定义**随机数 a 与密文 c 的乘法运算** $\otimes$

  $$a\otimes c=c^a\bmod n^2=g^{am}\cdot(r^a)^n\bmod n^2=Enc_{pk}(a\cdot m\bmod n)$$

  因此，$a\otimes c=Enc_{pk}(a\cdot m\bmod n)$

3.3Paillier 算法的 2 个优化点

3.3.1 预计算优化

3.3.1.1预备知识：勒让德符号与雅克比符号

如果某个非零元素可以开平方根，这样的元素称为模 n 二次剩余，否则叫模 n 二次非剩余。

例如$x^2 = a \bmod n$，则a是模 n 的二次剩余；$y^2 \neq a \bmod n$，则a是模 n 的二次非剩余；

性质：二次剩余的逆元仍然是二次剩余，二次非剩余的逆元也仍然是二次非剩余；而且每个二次剩余都有两个根并，且他们的和为 0。

说明：x也是二次剩余，即$\overline{x}^2 = x \bmod n$；$y$是二次非剩余，即$\overline{y}^2 \neq y \bmod n$；

一个数 a 是否是二次剩余的概率为 50%

勒让德符号：$\left(\frac{a}{p}\right) = \pm 1 = a^{\frac{p-1}{2}} \bmod p$

特殊情况：如果$a \equiv 0 \bmod p$，则勒让德符号$\left(\frac{a}{p}\right) = 0$

- 如果$a \neq 0 \bmod p$，且存在整数 x，满足$x^2 = a \bmod p$，则勒让德符号$\left(\frac{a}{p}\right) = 1$，则 a 是模 p 的二次剩余；
- 如果$a \neq 0 \bmod p$，且不存在整数 x，满足$x^2 = a \bmod p$，则勒让德符号$\left(\frac{a}{p}\right) = -1$，则 a 是模 p 的二次非剩余；

说明：勒让德符号与二次剩余是**充要条件**。

雅克比符号是勒让德符号的**累积**，整数 a 和整数 b，其中$b = p_1 p_2 \ldots p_r$。如果$\left(\frac{a}{b}\right) = 1$，则

雅克比符号定义为

$$\left(\frac{a}{b}\right) = \left(\frac{a}{p_1}\right)\left(\frac{a}{p_2}\right)\cdots\left(\frac{a}{p_r}\right)$$

其中$\left(\frac{a}{p_i}\right)$是勒让德符号。当 b 是奇素数时，雅可比符号就是勒让德符号。

（1）**如果雅克比符号**$\left(\frac{w}{N}\right) = 1$，则勒让德符号$\left(\frac{w}{p}\right) = \left(\frac{w}{q}\right) = 1$，即$w$是$p,q$的二次剩余；

或勒让德符号$\left(\frac{w}{p}\right) = \left(\frac{w}{q}\right) = -1$，即$w$是$p,q$的二次非剩余。

（2）**如果雅克比符号**$\left(\frac{w}{N}\right) = -1$，则勒让德符号$\left(\frac{w}{p}\right) = -1, \left(\frac{w}{q}\right) = 1$或$\left(\frac{w}{p}\right) = 1, \left(\frac{w}{q}\right) = -1$，$w$是$p,q$其中一个的二次剩余，另一个的二次非剩余。**该情况下一定存在一个二次剩余。**

因此，存在整数x，满足$x^2 = a \bmod p$或$x^2 = a \bmod q$。**等价于**存在整数x，满足

$x^2 = a \bmod N$。

3.3.1.2预计算优化原理

【参考文献】

1. Damgård I, Jurik M, Nielsen J B. A generalization of Paillier's public-key system with applications to electronic voting[J]. International Journal of Information Security, 2010, 9: 371-385.
2. Goldreich O, Rosen V. On the Security of Modular Exponentiation with Application to the Construction of Pseudorandom Generators[J]. Journal of Cryptology, 2003, 16(2).

密钥生成：

对大素数 p,q **额外要求** $p = q = 3 \bmod 4$，且 $\gcd(p-1,q-1)=2$，确保 p,q 不等。反之如果 $p=q$，则 $\gcd(p-1,q-1)=p-1=q-1\neq 2$。这样的素数称为 **Blum 整数**。

Blum 整数生成方法： 随机选择两个长度为 k-bit 的大数 k_1,k_2，计算 $p=4k_1+3, q=4k_2+3$；（a）检测 p,q 是否为素数；如果是，则计算 $p-1=2(2k_1+1), q-1=2(2k_2+1)$，（b）检测 $\gcd(p-1,q-1)=2$。

注释：

用户端： 寻找 Blum **整数** p,q 通常需要 5 到 6 秒（有点慢)。但是，密钥生成仅运行一次，但是有利于加密算法提速（加密算法执行多次)。

服务器： 可以预计算 p,q。

预计算： 选择随机数 $x\in\mathbb{Z}_n^*$，计算 $h=-x^2 \bmod n$，雅克比符号为 -1 确保存在二次剩余；（1）在小空间 $[0,...,(p-1)(q-1)/2]$ 选择随机数 a'，计算 $h^{a'} \bmod n$；（2）在小空间 $[0,...,2^{\lceil k/2\rceil}]$ 选择随机数 a，计算 $h^a \bmod n$。根据**参考文献 2** 的定理 3.2：假设因子分解是困难的，则三元组 $(n,h,h^{a'} \bmod n)$ 与三元组 $(n,h,h^a \bmod n)$ 计算不可区分。

预计算 $f=h^n \bmod n$

- **Paillier 私钥不变，还是 p,q；**
- **Paillier 公钥为 (n,f)，增加一个 f。**

Paillier 加密优化： 对于 Paillier 加密：$c=g^m\cdot r^n \bmod n^2$，

原来： 需要选择随机数 $r\in\mathbb{Z}_n^*$，计算 $r^n \bmod n^2$，

修改为： 基于预计算 f，在小空间随机数 $a\in\mathbb{Z}_{2^{\lceil n/2\rceil}}$，计算 $f^a \bmod n^2$。（随机数的长度减半，计算更快；如果因子分解是困难的，则计算不可区分，所以安全性一样）

$f^a \bmod n^2 = h^{na} \bmod n^2 = (h^a)^n \bmod n^2 = (-x^{2a})^n \bmod n^2$ 与 $r^n \bmod n^2$ 分布不可区分。

因此，Paillier 加密**优化为**：

$c = g^m \cdot r^n \bmod n^2 = g^m \cdot f^a \bmod n^2 = g^m \bmod n^2 \cdot f^a \bmod n^2$ （有 2 个模指数运算）

3.3.2 模指数优化

3.3.2.1预备知识：中国剩余定理

大空间$\mathbb{Z}_n$：1 维，有 100 个点$\mathbb{Z}_n$，分别为 $f_1,\dots,f_{100}$。

2 个小空间 p,q 取值范围是 10，组合为一个平面$\mathbb{Z}_p \times \mathbb{Z}_q$ 也能表达 100 个点。

所以大空间与两个小空间同构$\mathbb{Z}_n \simeq \mathbb{Z}_p \times \mathbb{Z}_q$

大空间**映射**到 2 个小空间（映射是模运算，速度快）、2 个小空间**并行快速计算**、2 个小空间**聚合映射回**大空间快（线性组合）

$$\begin{array}{ccc} \mathbb{Z}_n & \rightarrow & \mathbb{Z}_n \\ \downarrow & & \uparrow \\ \left\{\begin{matrix} \mathbb{Z}_p \\ \mathbb{Z}_q \end{matrix}\right\} & \rightarrow & \left\{\begin{matrix} \mathbb{Z}_p \\ \mathbb{Z}_q \end{matrix}\right\} \end{array}$$

模指数运算路径图

快速傅里叶变换也是类似绕圈：蝶形变换复用中间状态的多项式值，减小计算复杂度，

- **拉格朗日插值法**$O(n^2)$：多项式值表达计算多项式系数表达，计算复杂度$O(n^2)$；
- **蝶形变换**$O(n\log n)$：多项式的值复用、**中间状态复用**，计算复杂度$O(n\log n)$；

中国剩余定理的使用条件：知道n的因子分解为 p,q，才能构造子空间$\mathbb{Z}_p \times \mathbb{Z}_q$。如果不知道$n$的因子分解，则不能构造出同构子空间$\mathbb{Z}_p \times \mathbb{Z}_q$，则不能使用该定理。

- **ECDSA 多签协议中的一个子协议：份额转换协议 MtA：Alice 使用自己的 Paillier 公钥加密份额，发送给 Bob；Bob 进行 Paillier 同态计算后发送给 Alice；Alice 使用自己的 Paillier 私钥解密。因此，Alice 可以使用中国剩余定理加速 Paillier 加密算法。**
- **其他情况：对于 Paillier 解密算法：接收方肯定知道私钥 p,q，肯定可以使用中国剩余定理加速。**

费马小定理：若 p 为素数，a 是正整数且不能被 p 整除，互素，则$a^{p-1} \equiv 1 \bmod p$

欧拉定理：对任意互素的 a 和 n，有$a^{\varphi(n)} \equiv 1 \bmod n$

模反元素存在：如果两个正整数 a 和 n 互素，则一定可以找到整数 b，使得 ab-1 被 n 整除或 $ab \equiv 1 \bmod n$，则称 b 是 a 的模反元素。

证明：使用**欧拉定理** $a^{\varphi(n)} \equiv 1 \bmod n$，则 $a^{1+\varphi(n)-1} \equiv 1 \bmod n$，则 $a \times a^{\varphi(n)-1} \equiv 1 \bmod n$，则 $b = a^{\varphi(n)-1}$。

中国剩余定理/孙子定理，作用：n 个小空间聚合映射回大空间（线性组合）

一个大代数空间 $\mathbb{Z}_n$ 可以被分解为 2 个小代数空间 $\mathbb{Z}_p \times \mathbb{Z}_q$，且大空间与 2 个小空间有一一映射。具体而言，$n = p \cdot q$,且 p 与 q 互素，存在同构 $\mathbb{Z}_n \simeq \mathbb{Z}_p \times \mathbb{Z}_q$，使得在 $\mathbb{Z}_n$ 空间中的运算**等价转化**为 $(\mathbb{Z}_p, \mathbb{Z}_q)$ 空间的**并行**运算。

线性同余方程组 S 表达如下

$$S: \begin{cases} x \equiv a_1 \bmod m_1 \\ x \equiv a_2 \bmod m_2 \\ \dots \\ x \equiv a_n \bmod m_n \end{cases}$$

已知 $m_1,...,m_n$ 两两互素，$a_1,...,a_n$ 为任意整数，求 x？

中国剩余定理 CRT：

假设 $m_1,...,m_n$ 两两互素，对于任意整数 $a_1,...,a_n$，线性同余方程 S 有解，且能够计算解的通用表达。

假设 $M = \prod_{i=1}^{n} m_i$，且 $M_i = M / m_i = \prod_{j=1, j \neq i}^{n} m_i, j = 1,...,n$。所以，$M_i$ 与 m_i 互素。（条件 1）

由于**模反元素存在性**，所以存在 t_i 满足 $M_i \cdot t_i \equiv 1 \bmod m_i$。（条件 2）

线性同余方程 S 的**通解**为 $x = kM + \sum_{i=1}^{n} a_i t_i M_i, k \in \mathbb{Z}$，

在模 M 意义下**通解**唯一 $x = \sum_{i=1}^{n} a_i t_i M_i \bmod M$

证明： 根据条件 1：$a_i t_i M_i \equiv 0 \bmod m_j$；

根据条件 2：$a_i t_i M_i \bmod m_i \equiv a_i \cdot 1 \bmod m_i \equiv a_i \bmod m_i$；

（1）令 $x = \sum_{i=1}^{n} a_i t_i M_i$，则

$$x \bmod m_i = a_i t_i M_i + \sum_{j \neq i}^{n} a_j t_j M_j \bmod m_i = a_i t_i M_i \bmod m_i + 0 = a_i \bmod m_i$$ 。

$i = 1, \ldots, n$

说明 x 是方程的一个解。

（2）假设方程有 2 个解 x_1, x_2，则

$$x_1 \equiv a_i \bmod m_i, i = 1, \ldots, n$$
$$x_2 \equiv a_i \bmod m_i, i = 1, \ldots, n$$
$$x_1 - x_2 \equiv 0 \bmod m_i, i = 1, \ldots, n$$

且 $m_1, \ldots, m_n$ 两两互素且 $M = \prod_{i=1}^{n} m_i$，则 $x_1 - x_2 \equiv 0 \bmod M$。因此，任意两个解 x_1, x_2 相差 M 的整数倍。

因此，对于一个解是 $x = \sum_{i=1}^{n} a_i t_i M_i$，则其他解是 $x = kM + \sum_{i=1}^{n} a_i t_i M_i, k \in \mathbb{Z}$。

3.3.2.2预备知识：裴蜀定理/贝祖定理

法国数学家艾蒂安·裴蜀

对任何整数 a、b,且 gcd(a,b)=d，关于未知数 x 和 y 的**线性不定方程**（称为**裴蜀等式**）：

若 a,b 是整数,且 gcd(a,b)=d，则

（1）对于任意整数 x, y, 则 ax+by 一定是 d 的倍数；

（2）特别地，一定存在整数 x, y，使 ax+by=d 成立；

（3）**重要推论：a,b 互素的充分必要条件是存在整数 x,y 使 ax+by=1.**

证明：（1）因为 gcd(a,b)=d，所以 $d \mid a, d \mid b$，所以任意正整数 x, y， $d \mid ax + by$ 成立。

（2.1）对任意整数 x,y，设 s 是 $ax + by$ 的最小正整数（a,b 的线性组合），商 $q = \lfloor a / s \rfloor$，则

$$余数\ r = a \bmod s = a - qs = a - q(ax + by) = a(1 - qx) + b(-qy)$$

余数 r 也是 a,b 的线性组合，且余数 r 的范围为 $0 \le r < s$。

s 为 $ax + by$ 的最小正整数，则余数 r=0。因此 $s \mid a$。同理有 $s \mid b$。因此 s 是 a,b 的公约数。

且 gcd(a,b)=d，所以 $s \le d$。

（**2.2**）因为 gcd(a,b)=d，所以 $d \mid a, d \mid b$。且 s 是 a,b 的线性组合，所以 $d \mid s$。由于 s 是最小正整数即 $s > 0$，则 $d \le s$。

所以 $d = s$。

（**3**）如果 a, b 互素，$\gcd(a, b) = 1 = d = s$，且 s 是 $ax + by$ 的最小正整数，则 $ax + by = 1$。

3.2.2.3模指数优化原理

当 $n = p \cdot q$ 且 p, q 互素，计算模指数 $a^b \bmod n$，有以下 2 个计算方法

- **慢方法：**直接计算 $a^b \bmod n$。在大空间 $\mathbb{Z}_n$ 范围内慢慢计算，计算复杂度高。
- **快方法：**映射到小空间，**并行**快速计算，然后中国剩余定理 $\mathbb{Z}_n \simeq \mathbb{Z}_p \times \mathbb{Z}_q$ **映射返回**。

步骤 1：大空间映射到 2 个小空间，在 2 个小空间并行模指数计算

（1）$a^b \bmod n$ **映射** $a' = a \bmod p$，根据欧拉定理 $b' = b \bmod \varphi(p)$；

在小空间 $\mathbb{Z}_p$ 上进行模指数运算 $x' = a'^{b'}$。

（2）计算 $a^b \bmod n$ **映射** $a'' = a \bmod q$，根据欧拉定理 $b'' = b \bmod \varphi(q)$；

在小空间 $\mathbb{Z}_q$ 上进行模指数运算 $x'' = a''^{b''}$。

步骤 2：2 个小空间聚合映射回大空间

中国剩余定理解的通项公式 $x = \sum_{i=1}^{n} a_i t_i M_i \bmod M$ （聚合映射回大空间）

$$x = x' \cdot q^{-1}(\bmod p) \cdot q + x'' \cdot p^{-1}(\bmod q) \cdot p$$

裴蜀定理： p 与 q 互素，则 $q^{-1}(\bmod p) \cdot q + p^{-1}(\bmod q) \cdot p = 1$

带入裴蜀定理进一步优化：

$$\begin{aligned} x &= x' \cdot q^{-1}(\bmod p) \cdot q + x'' \cdot p^{-1}(\bmod q) \cdot p \\ &= x' \cdot (1 - p^{-1}(\bmod q) \cdot p) + x'' \cdot p^{-1}(\bmod q) \cdot p \\ &= x' + (x'' - x') p^{-1}(\bmod q) \cdot p \end{aligned}$$

3.2.2.4Paillier加密模指数优化

Paillier 加密算法 $c = g^m \cdot r^n \bmod n^2 = g^m \cdot f^a \bmod n^2 = g^m \bmod n^2 \cdot f^a \bmod n^2$ **是模** n^2，所以构造中国剩余定理 $\mathbb{Z}_{n^2} \simeq \mathbb{Z}_{p^2} \times \mathbb{Z}_{q^2}$。（**注意使用条件：**发送方知道 p, q 时，才可以使用。）

步骤 1：大空间映射到 2 个小空间，在 2 个小空间并行模指数计算

- $g^m \bmod n^2, f^a \bmod n^2$ 映射到模 p^2 小空间

$$g' = g \bmod p^2, m' = m \bmod \varphi(p^2), f' = f \bmod p^2, a' = a \bmod \varphi(p^2)$$

然后在**小空间并行**模 p^2 指数运算 $(g')^{m'} \bmod p^2, (f')^{a'} \bmod p^2$

- $g^m \bmod n^2, f^a \bmod n^2$ 映射到模 q^2 小空间

$$g'' = g \bmod q^2, m'' = m \bmod \varphi(q^2), f'' = f \bmod q^2, a'' = a \bmod \varphi(q^2)$$

然后在**小空间并行**模 q^2 指数运算 $(g'')^{m''} \bmod q^2, (f'')^{a''} \bmod q^2$。

步骤 2：2 个小空间聚合映射回大空间

使用**中国剩余定理**聚合映射回大空间，并使用**裴蜀定理**优化。

3.2.2.5Paillier解密模指数优化

对于 Paillier 解密算法，解密方肯定知道 p,q，所以对于解密算法 $m := \dfrac{L(c^{\lambda} \bmod n^2)}{L(g^{\lambda} \bmod n^2)} \bmod n$

- 分母：$g^{\lambda} \bmod n^2$ 可以**预计算为** $\mu = \left(L(g^{\lambda} \bmod n^2)\right)^{-1} \bmod n$
- 分子：$c^{\lambda} \bmod n^2$ 可以使用**中国剩余定理加速**。

4. 素性测试

素数很重要！那么一个 2048bit 的大整数是否为素数？如何判断？

n 是奇数，则 $n-1$ 是偶数。将偶数 $n-1$ 进行 k 次除以 2，直至结果为奇数 q。因此，任意 $n \geq 3$ 的**奇整数**可以表达为 $n-1=2^k q$，其中 $k>0$，q 是奇数。

素数 p 有两个性质：

性质 1： 若 p 为素数，a 是小于 p 的正整数，则 $a^2 \bmod p = 1$ **当且仅当** $a \bmod p = 1$ 或 $a \bmod p = -1 = p-1$。

分析：如果 $a \bmod p = 1$ 或 $a \bmod p = -1 = p-1$，则 $(a \bmod p)(a \bmod p) = a^2 \bmod p = 1$

反之，如果 $a^2 \bmod p = 1$，则 $a^2 \bmod p = (a \bmod p)(a \bmod p) = 1$，则 $a \bmod p = 1$ 或 $a \bmod p = -1 = p-1$。

性质 2： 若 p 为大于 2 的素数，则 $p-1=2^k q$，a 是小于 $p-1$ 的正整数，则以下 2 个条件之一成立：（a）$a^q \equiv 1 \bmod p$；（2）整数集合 $\{a^q, a^{2q}, ..., a^{2^{k-1}q}\}$ 中存在某个数，假如是第 i 个数 $a^{2^i q}$，**满足** $a^{2^i q} \equiv -1 \bmod p$

分析： 根据费马小定理，p 为素数，a 是小于 p 的正整数，则 $a^{p-1} \equiv 1 \bmod p$。则 $a^{p-1} = a^{2^k q} = 1 \bmod p$。因此，**扩展整数集合** $\{a^q, a^{2q}, ..., a^{2^{k-1}q}, a^{2^k q}\}$ 中最后一个为 1.

则 $(a^{2^{k-1}q})^2 \bmod p = a^{2^k q} \bmod p = 1$。根据性质 1，则 $a^{2^{k-1}q} \bmod p = \pm 1$。

- 如果性质（a） $a^q \equiv 1 \bmod p$ 成立，则整数集合 $\{a^q, a^{2q}, ..., a^{2^{k-1}q}\}$ 每个元素均为 $a^{2^i q} \equiv 1 \bmod p$，性质（2）某一项为 $a^{2^i q} \equiv -1 \bmod p$ 不成立。
- 如果性质（a） $a^q \equiv 1 \bmod p$ 不成立，则 $a^{2^{k-1}q} \bmod p = -1$ 性质（b）成立。

因此，如果 n 是素数，则扩展整数集合 $\{a^q, a^{2q}, ..., a^{2^{k-1}q}, a^{2^k q}\}$，要么 $a^q \equiv 1 \bmod p$ 要么 $a^{2^i q} \equiv -1 \bmod p$。**反之，不一定成立（是充分条件，不是必要条件）。**

例如 n=2047=23*89，则 n-1=2*1023，$2^{1023} \bmod 2047 = 1$ 满足条件，但是 2047 不是素数。

素性测试算法：

输入整数 n，计算 $n-1 = 2^k q$；

选择随机数 $a \in (1, n-1)$；

如果 $a^q \bmod n = 1$，则返回"可能是素数"；

循环 k 次：如果 $a^{2^i q} \equiv -1 \bmod n, i = 0, ..., k-1$，则返回"可能是素数"；

返回"一定是合数"。

由于合数比素数更多，根据统计，

- 返回"一定是合数"的概率为 $3/4$，
- 返回"可能是素数"的概率小于 $1/4$。

如果重复测试 t 次，则返回"可能是素数"的概率小于 $1/4^t$，呈指数降低。因此，重复 t 次均返回"可能是素数"，则可以认为该整数是素数。

评估：这是一个概率算法，不是确定性算法。存在确定性算法，但效率太低。这个概率算法效率高。

5. 同态加密汇总

5.1Paillier 同态加密

密钥生成：生成两个长度相同的大素数 p, q，满足 $\gcd\left(pq, (p-1)(q-1)\right) = 1$，该性质确保这两个素数长度相同；计算 $n = pq$，最小公倍数 $\lambda = lcm(p-1, q-1)$；分式除法函数 $L(y) = (y-1)/n$；选择正整数 $g = 1+n \in Z_{n^2}^*$，使得 $\mu = \left(L(g^\lambda \bmod n^2)\right)^{-1} \bmod n$ 存在。

公钥为n，私钥为p,q。

如果p,q长度相等，则密钥生成步骤能够化简为：$g=n+1,\lambda=\varphi(n),\mu=\varphi(n)^{-1}\bmod n$，

其中$\varphi(n)=(p-1)(q-1)$；如果p,q足够大，则μ的计算时间较长，推荐使用该化简方法。

加密：消息$m\in Z_n$，选择随机数$r\in Z_n^*$，计算密文$c:=g^m\cdot r^n \bmod n^2$。

解密：输入密文$c\in Z_{n^2}$，如下计算解密$m:=L(c^\lambda \bmod n^2)\cdot\mu \bmod n$。

同态性：给定两个密文$c_1,c_2\in Z_{n^2}$，其中$c_1=Enc_{pk}(m_1),c_2=Enc_{pk}(m_2)$

- 定义**密文之间的加法运算**$\oplus$

  $$c_1\oplus c_2=c_1c_2 \bmod n^2=(g^{m_1}\cdot r_1^n \bmod n^2)(g^{m_2}\cdot r_2^n \bmod n^2)=g^{m_1+m_2}\cdot(r_1r_2)^n \bmod n^2$$

  因此，$c_1\oplus c_2=c_1c_2 \bmod n^2=Enc_{pk}(m_1+m_2 \bmod n)$。

- 给定$a\in Z_n,c=Enc_{pk}(m)$定义**明文与密文的乘法运算**$\otimes$：

$$a\otimes c=c^a \bmod n^2=g^{am}\cdot(r^a)^n \bmod n^2=Enc_{pk}(a\cdot m \bmod n)$$

5.2ElGamal 同态加密

初始化：素数群G的阶为q，生成元为 g。

密钥生成：私钥$sk=\alpha$，公钥$PK=h$，满足离散对数关系$h=g^\alpha$。

加密：消息为$m\in\mathbb{Z}_q$，选择随机数$r\in\mathbb{Z}_q$，计算$C_1=g^r,C_2=g^m\cdot h^r$。

令密文为$C=(C_1,C_2)$。

解密：输入私钥$sk=\alpha$和密文(C_1,C_2)，计算$g^m=C_2/C_1^\alpha$

对g^m暴力搜索获得明文消息m，需要**指数计算复杂度**，如果 m 是 32bit 就很慢了。

门罗币：32bit 金额
ECDSA 协议中的份额转换协议，保密份额 32bit，经不住暴力搜索。
msg 太大，自己无法解密保密份额。msg 太小，禁不住对方暴力搜索。所以 ECDSA 的协议通常使用 Paillier 同态加密。

同态性：对于两个密文C,C'，定义**加法同态**为$C\oplus C'$

$$C\oplus C'=C\cdot C'=\left(g^{r+r'},g^{m+m'}\cdot h^{r+r'}\right)$$

对于一个密文C与随机数x，定义**乘法同态**为$C\otimes x$

$$C\otimes x:=\left(g^{xr},g^{xm}\cdot h^{xr}\right)$$

5.3 特殊的素数类群

大群生成元为g，阶为n；
子群生成元为h，阶很小$\tilde{n}$。

$$C_1:=g^r$$
$$C_2:=g^r\cdot h^m$$

子群上的离散对数**相对不困难**，所以用于替换素数群或椭圆曲线群上的 ElGamal 同态加密。MPC 通常使用 Paillier 同态加密用于**份额转换协议**。ElGamal 加密通常用于别的目的。
举例：

大群$\{g,g^2,g^3,...,g^{90}=h,g^{91},...,g^{99}\}$大群上的离散对数是困难的。

小群$\{h,h^1,...,h^{10}\}$小群上的离散对数相对不困难。

大群用于数据随机化；小群用于数据同态计算。

6. 同态加密解决百万富翁问题

百万富翁问题：比谁的 Token 更多。本质上是两个数比较大小，两方隐私计算。

	Alice	Bob
1	私钥和公钥为(sk,PK)。 使用公钥PK对其财富值v_0进行加密 $C_0=Enc_{PK}(v_0)$ **发送**密文C_0和公钥PK	
2		**接收**密文$C_0=Enc_{PK}(v_0)$和公钥PK （1）选择 2 个随机数b_0,b_1。使用公钥PK对随机数b_0进行加密$C'=Enc_{PK}(b_0)$。对密文C_0进行**同态运算**

<table>
<tr><td></td><td></td><td>$$\begin{aligned}C_1 &= C_0^{b_1}\cdot C'\\ &= Enc_{PK}(b_1v_0)\cdot Enc_{PK}(r_0)\\ &= Enc_{PK}(b_1v_0+b_0)\end{aligned}$$
（2）使用使用公钥 PK 对其财富值 v_1 的计算掩盖值 $b_1v_1+b_0$，然后计算加密
$$C_2 = Enc_{PK}(b_1v_1+b_0)$$
发送 2 个密文 C_1, C_2</td></tr>
<tr><td>3</td><td>接收 2 个密文 C_1, C_2
（1）使用私钥 sk 从密文 C_1 中解密获得 Alice 财富的掩盖值 $b_1v_0+b_0$；
（2）从密文 C_2 中解密获得 Bob 财富的掩盖值 $b_1v_1+b_0$；
因此，对 Alice 财富的掩盖值 $b_1v_0+b_0$ 与 Bob 财富的掩盖值 $b_1v_1+b_0$ 比较大小，而不泄露财富值。
将比较结果告诉 Bob</td><td></td></tr>
</table>