密码学期末复习

1 密码学原理

1.1 信息安全的基本属性

- **机密性**：信息不泄漏给非授权的用户、实体或过程。
- **完整性/认证性**：
  - 数据未经授权不能被改变，即信息在存储或传输过程中保持不被偶然或蓄意的删除、修改、伪造、乱序、重放、插入等操作所破坏；
  - 可证明信息由授权者发出。
- 机密性由加密实现，完整性可由消息认证码（对称密码学）、数字签名（公钥密码学）实现。
- 其他属性：
  - 可用性：保证信息和信息系统可被授权实体访问并按需求使用的特性，即当需要时应能存取所需的信息。
  - 不可否认性：无论发送方还是接收方都不能抵赖所进行的传输。
  - 安全性：在攻击下仍保持完整。
  - 正确性：在所有算法都无差错执行时能符合预期地工作。
  - 安全性的提高必然以牺牲效率/可用性为代价。

1.2 密码学基本原则

- **Kerckhoff原则**：一个密码系统的安全性不应依赖于其算法的保密性，而是应该建立在密钥的保密之上。即使密码系统的所有细节，包括明文的统计特性、加密体制（操作方式、处理方法、加/解密算法）、密钥空间等，都被公众所知，只要密钥本身是安全的，那么整个密码系统仍然被认为是安全的。
- **大密钥空间原则**：密钥空间的大小必须足够大，以确保密码系统的安全性。 密钥空间的大小应该远远超过攻击者的计算能力，以保证密码系统的强大安全性。该原则对于安全性来说必要但不充分。

1.3 现代密码学三要素

- **Syntax 语法**
- **Security Models 安全模型**
- Provable Security 可证明安全

1.4 现代密码学三原则

1. **公式化的、表述严格且精确的安全定义**。包括语法（Syntax）和安全模型。
   - 语法：阐明功能性（Functionalities）。
   - 安全模型包括：
     - **要保护的对象（security guarantee）**：密文不泄露任何有关明文的信息。（不可区分性：看到密文和不看到密文没有区别。）
     - **要防范的敌人（threat model）**：刻画攻击者获得信息的能力以及对信息的控制能力。

    - 例如：CPA/CCA区别是前者不能访问解密机。故人强度：CCA>CPA>KPA>COA。
    - 如果特定的敌手不能完成特定的攻破，则对给定任务的一个密码学方案是安全的。比如：确定性算法不是CPA安全的。
    - 攻击模型：①Probing Phase ②Challenge Phase ③Output Phase（能以显著的优势猜中则胜利）
2. **对精确假设的依赖**。当密码学构造方案的安全性依赖于某个未被证明的假设时，这种假设必须被精确地陈述，并且所假设的要尽可能少。
    - 假设定义的一般结构：如果对于所有多项式时间算法A，存在一个可忽略函数$negl(n)$满足$Pr[Experiment_{A,GenProblem}(n)=1]\leq negl(n)$（Experiment是一个精确定义的关于A和$GenProblem$的实验），则与算法$GenProblem$相关的问题$Problem$是困难的。
    - 例如：RSA的安全性依赖于大整数因子分解困难假设、离散对数求逆困难假设、Diffie-Hellman困难假设等。
3. **严格的安全性证明**。“若给定假设X是正确的，根据给定的定义，构造方案Y是正确的。”

2 对称密码学

2.1 对称加密方案一般结构

- Syntax：
    - $KeyGen(\lambda)\rightarrow k$ 概率性算法
    - $Enc(k,m)\rightarrow c$ 概率性算法/确定性算法
    - $Dec(k,c)\rightarrow m$
- Correctness：$Dec(k,Enc(k,m))=m$
- 安全模型：CCA安全，攻击者无法从密文中获得任何明文的信息。

2.2 对称加密方案实例

- （流密码）RC4加密：
    - 密钥空间：2^128
    - S盒、K盒长度：256
- （分组密码）DES和AES：
    - 密钥长度：DES：56；AES：128/192/256
- 分组密码应用模式：
    - ECB（电子密码本）：$Enc(k,m_i)\rightarrow c_i$；
    - CBC（分组链接，有一个随机的初始向量）：$Enc(k,IV\oplus m_0)\rightarrow c_0$，$Enc(k,c_{i-1}\oplus m_i)\rightarrow c_i$；
    - CTR（计数器模式）：$m_i\oplus Enc(k,IV+i)\rightarrow c_i$

2.3 消息认证码（MAC）

- Syntax：
  - $KeyGen(\lambda) \to k$
  - $Mac(k, m) \to t$ （tag）
  - $Verify(k, (m, t)) \to 1/0$
- Correctness：$Verify(k, m, Mac(k, m)) = 1$
- 安全模型：没有攻击者能够以不可忽略的优势伪造有效的（消息，tag）对。

3 Hash

3.1 Hash函数一般结构

- Syntax：$H(x)$　　多项式时间
- 安全模型：
  - 抗碰撞性（强抗碰撞性）：寻求使得$H(x) = H(y)$的(x, y)在计算上不可行。
  - 目标抗碰撞性（抗第二原像、弱抗碰撞性）：对于任意给定的x，寻求y使$H(y) = H(x)$在计算上不可行。
  - 原像安全（单向性）：对于给定的哈希值，无法反向推导出原始数据。即从哈希输出无法倒推输入的原始数值，这是哈希函数安全性的基础。
- 一个强抗碰撞性的H必定是目标抗碰撞性的，一个目标抗碰撞性的H必定是单向性的。

3.2 Hash函数实例

- MD5
  - Output：128 bits
  - 已被破解。
- SHA-1（160 bits）/SHA-2（SHA-256/384/512）/SHA-3（SHA3-256/384/512）

3.3 HMAC

- $HMAC_k(m) = H(k \oplus opad || H((k \oplus ipad) || m))$
- 广泛应用于IPSec。

4 数论

4.1 素数分解

$$a = p_1^{\alpha_1} p_2^{\alpha_2} \dots p_t^{\alpha_t}$$

- a为正整数，p_i为素数，对a的素数分解唯一。
- 如果a很大，则对a做素数分解是困难的。

4.2 辗转相除法求最大公因数

$$gcd(a,b)=gcd(b,a \mod b),\ a>b$$

- 计算$gcd(a,b)$:
  - $a=k_1b+r_1$;
  - $b=k_2r_1+r_2$;
  - $r_1=k_3r_2+r_3$;
  - ...
  - $r_{n-1}=k_{n+1}\cdot r_n+0$;
  - $gcd(a,b)=r_n$。

4.3 拓展欧几里得算法求模逆元

$$a^{-1} \mod b$$

- 例：$a=911$，$b=999$。

  由辗转相除法得

$$\begin{gathered}999=1\times911+88\\911=10\times88+31\\88=2\times31+26\\31=1\times26+5\\26=5\times5+1\\5=5\times1+0\\\rightarrow gcd(999,911)=1\end{gathered}$$

  再追溯得到

$$\begin{aligned}1&=26-5\times5\\&=26-5\times(31-1\times26)=-5\times31+6\times26\\&=-5\times31+6\times(88-2\times31)=6\times88-17\times31\\&=6\times88-17\times(911-10\times88)=-17\times911+176\times88\\&=-17\times911+176\times(999-1\times911)=176\times999-193\times911\end{aligned}$$

  所以

$$\begin{gathered}gcd(911,999)=1=-193\times911+176\times999\\\therefore1=806\times911 \mod 999\\\therefore911^{-1} \mod 999=806\end{gathered}$$

4.4 欧拉函数ϕ

$$\phi(n)=|\{x|1\leq x\leq n且gcd(x,n)=1\}|$$

- 表示小于等于n且与n互素的正整数个数。
- 如果n是一个素数则$\phi(n)=n-1$。
- 如果m与n互素则$\phi(mn)=\phi(m)\phi(n)$。
- $\phi(p^e)=p^{e-1}(p-1)$，$e\geq1$且为整数。
- 如果$n=p_1^{e_1}p_2^{e_2}\dots p_k^{e_k}$，$p_i$为素数，$e_i$为正整数，则$\phi(n)=n\prod_{i=1}^{k}(1-\frac{1}{p_i})$。

- 如果a与n互素，则$a^{\phi(n)} = 1 \mod n$。
- $a^b \mod n = a^{b \mod \phi(n)} \mod n$，$gcd(a, n) = 1$。

4.5 快速幂取余

- $a^b \mod n = (a \mod n)^b \mod n$。
- 例：

$$
\begin{gathered}
11^{15} \mod 13 = 11^8 \times 11^4 \times 11^2 \times 11^1 \mod 13 \\
11^2 = 4 \mod 13 \\
11^4 = (11^2)^2 = 4^2 = 3 \mod 13 \\
11^8 = (11^4)^2 = 3^2 = 9 \mod 13 \\
\therefore 11^{15} = 9 \times 3 \times 4 \times 11 = 5 \mod 13
\end{gathered}
$$

4.6 群论

- 群$(G, \circ)$
  - 封闭性：$a, b \in G$，则$a \circ b \in G$。
  - 结合律：$a, b, c \in G$，则$(a \circ b) \circ c = a \circ (b \circ c)$。
  - 单位元存在性：存在$e \in G$，对任何$a \in G$，都有$a \circ e = e \circ a = a$。
  - 逆元存在性：对每个$a \in G$，存在$a^{-1} \in G$，满足$a^{-1} \circ a = e$。
  - 满足交换律的群为可交换群。
  - 例：
    - $Z_n = \{0, 1, \ldots, n-1\}$是一个关于模n加法运算的群。
    - $Z_n^* = \{a \in Z_n | gcd(a, n) = 1\}$是一个关于模n乘法运算的群。
- 循环群：如果存在一个群元素$g \in G$，满足对每个$a \in G$，存在整数i使得$a = g^i$，则这个群是循环群。g被称为这个群的生成元。
  - 例：判断$\{1, 2, 3, 4; \times (\mod 5)\}$是否为循环群？
    - 设生成元$g = 2$；
    - $g^0 = 1, g^1 = 2, g^2 = 4, g^3 = 3 \mod 5, g^4 = 1 \mod 5$；
    - 所以这是一个乘法循环群，2是它的一个生成元。
    - 同理可得3也可以是它的生成元。4不是。
  - Z_n^*拥有至少一个生成元当且仅当$n = 2, 4, p^k, 2p^k$，p为奇素数，$k \geq 1$。
  - g^i的模n逆元是g^{n-1-i}。

4.7 密码学中的困难性假设

- **因子分解假设**：存在一个$GenModulus$，与之相关的因子分解问题是困难的。
  - 因子分解问题：给定N，找到素数p, q满足$N = pq$。

- 正式/精确定义：设$GenModulus$为多项式时间算法，输入1^n，输出(N,p,q)，$N=pq$，p,q为两个n比特素数。该算法失败的概率（以n表示）是可忽略的。定义实验$Factor_{A,GenModulus}(n)$：给定算法A和参数n：
  1. 运行$GenModulus(1^n)$得到(N,p,q)；
  2. A被给定N，输出p',q'>1；
  3. 如果$p'\cdot q'=N$则定义实验输出为1，否则为0。（除可忽略的概率外，如果实验输出为1，则$\{p',q'\}=\{p,q\}$。）
- 如果对于任意概率多项式时间算法A，存在一个可忽略函数$negl(n)$满足$Pr[Factor_{A,GenModulus}(n)=1]\leq negl(n)$，则称与$GenModulus$相关的因子分解问题是困难的。

- **RSA困难假设**：存在一个$GenRSA$，与之相关的RSA问题是困难的。
  - RSA问题：给定N，一个满足$gcd(e,\phi(N))=1$的整数e>1，和一个元素$y\in Z_N^*$，计算$y^{1/e} \mod N$，即找到x满足$x^e=y \mod N$。
    - 正式/精确定义：设$GenRSA$为概率多项式算法，输入1^n，输出模N（两个n比特素数的乘积）、满足$gcd(e,\phi(N))=1$的整数e>1和满足$ed=1 \mod \phi(N)$的整数d。该算法失败的概率（以n表示）是可忽略的。定义实验$RSA-inv_{A,GenRSA}(n)$：给定算法A和参数n：
      1. 运行$GenRSA(1^n)$得到(N,e,d)；
      2. 选择$y\in Z_N^*$；
      3. A被给定N,e,y，输出$x\in Z_N^*$；
      4. 如果$x^e=y \mod N$则定义实验输出为1，否则为0。
  - 如果对于任意概率多项式时间算法A，存在一个可忽略函数$negl(n)$满足$Pr[RSA-inv_{A,GenRSA}(n)=1]\leq negl(n)$，则称与$GenRSA$相关的RSA问题是困难的。
  - RSA困难假设比大素数分解困难假设要强。

- **离散对数假设**：存在一个$\mathcal{G}$，与之相关的离散对数问题是困难的。
  - 离散对数问题：给定随机元素$h\in G$作为输入，计算$log_g h$。
    - 正式/精确定义：设$\mathcal{G}$为多项式时间算法，输入1^n，输出一个循环群G，阶为q（$||q||=n$），生产元为g，群运算也是以n表示的多项式时间运算。该算法失败的概率（以n表示）是可忽略的。定义实验$DLog_{A,\mathcal{G}}(n)$：给定算法A和参数n：
      1. 运行$\mathcal{G}(1^n)$得到(G,q,g)；
      2. 选择$h\in G$，可以通过选择$x'\in Z_q$，并设置$h=g^{x'}$来完成；
      3. A被给定G,q,g,h，输出$x\in Z_q$；
      4. 如果$g^x=h$则定义实验输出为1，否则为0。（除可忽略的概率外，如果实验输出为1，则$x=x'$。）
  - 如果对于任意概率多项式时间算法A，存在一个可忽略函数$negl(n)$满足$Pr[DLog_{A,\mathcal{G}}(n)=1]\leq negl(n)$，则称与$\mathcal{G}$相关的离散对数问题是困难的。

- **Diffie-Hellman假设**：存在一个$\mathcal{G}$，与之相关的CDH/DDH问题是困难的。
  - 选定循环群G，以及一个生成元g。给定两个元素h_1,h_2，定义$DH_g(h_1,h_2)=g^{log_g h_1\cdot log_g h_2}$，也就是说，如果$h_1=g^x$且$h_2=g^y$，则$DH_g(h_1,h_2)=g^{xy}=h_1^y=h_2^x$。
  - 计算D-H问题（CDH）：当随机选定h_1,h_2时，计算$DH_g(h_1,h_2)$。
  - 判定D-H问题（DDH）：给定随机选择的h_1,h_2和候选答案h'，判断$h'=DH_g(h_1,h_2)$是否成立。

- 如果对于任意概率多项式时间算法A，存在一个可忽略函数$negl(n)$满足$|Pr[A(G,q,g,g^x,g^y,g^z)=1]-Pr[A(G,q,g,g^x,g^y,g^{xy})=1]|\leq negl(n)$，则称与$\mathcal{G}$相关的DDH问题是困难的。$(G,q,g)$是实验$\mathcal{G}(1^n)$的输出，随机选择$x,y,z\in Z_q$。
- 如果与某些$\mathcal{G}$相关的离散对数问题是简单的，则与之相关的CDH问题也是简单的；如果与某些$\mathcal{G}$相关的CDH问题是简单的，则与之相关的DDH问题也是简单的。这说明在难度上离散对数问题>CDH问题>DDH问题。

- $\mathcal{G}$：群生成算法$GenGroup()$：
  - 输入：安全参数1^n，参数l。
  - 选择一个n位的素数q；
  - 选择一个l位的素数p，满足$q|(p-1)$；
  - 选择一个$h\in Z_p^*$，$h\neq 1$；
  - $g=h^{\frac{p-1}{q}} \mod p$；
  - 输出：p,q,g。
  - q为循环群的阶数，g为生成元，运算为$\times \mod p$

5 公钥密码学

5.1 公钥加密一般结构

- Syntax：
  - $KeyGen(\lambda)\to(pk,sk)$
  - $Enc(pk,m)\to c$
  - $Dec(sk,c)\to m$
- Correctness：$Dec(sk,Enc(pk,m))=m$
- 安全模型：公钥密码算法不是CPA安全的，是CCA安全的。

5.2 公钥加密实例

- RSA加密算法
  - $KeyGen(1^\lambda)\to(pk,sk)$：
    1. Input：1^λ，指定密钥空间大小；
    2. 生成大素数 p、q，计算$N=pq$，$\phi(N)=(p-1)(q-1)$；
    3. 选择正整数e，满足$gcd(e,\phi(N))=1$且$e<\phi(N)$，公钥对即为$pk=(N,e)$；
    4. 计算$d=e^{-1} \mod \phi(N)$，私钥对即为$sk=(N,d)$；
    5. Output：(pk,sk)，公私钥对。
- $Enc(pk,m)\to c$：
  1. Input：$pk=(N,e)$，公钥对；$m\in Z_N^*$，为明文（Z_N^*为明文空间，表示从1到N之间与N互素的正整数集合）；
  2. 计算得到密文$c=m^e \mod N$；
  3. Output：c，密文。

- $Dec(sk, c) \to m$:
    1. Input：$sk = (N, d)$，私钥对；$c \in Z_N^*$，为密文（Z_N^*为密文空间）；
    2. 计算得到明文$m = c^d \mod N$；
    3. Output：m，明文。
- Correctness：

$$\begin{aligned} Dec(sk, Enc(pk, m)) &= Dec(sk, m^e \mod N) \\ &= (m^e \mod N)^d \mod N \\ &= m^{ed \mod \phi(N)} \mod N \\ &= m \mod N \end{aligned}$$

- EI-Gamal加密算法
    - $KeyGen(1^\lambda) \to (pk, sk)$：
        1. Input：1^λ，指定密钥空间大小；
        2. $GenGroup(1^\lambda) \to (G, p, g)$；
        3. 随机选择一个$x \in Z_p$，计算$h = g^x$；
        4. $pk = (G, p, g, h)$，$sk = (G, p, g, x)$；
        5. Output：(pk, sk)，公私钥对。
    - $Enc(pk, m) \to c$：
        1. Input：$pk = (G, p, g, h)$，公钥对；$m \in G$，为明文；
        2. 选择一个$y \in Z_p$，计算得到密文$c = (g^y, mh^y) \mod p$；
        3. Output：c，密文。
    - $Dec(sk, c) \to m$：
        1. Input：$sk = (G, p, g, x)$，私钥对；$c = (c_1, c_2)$，为密文；
        2. 计算得到明文$m = c_2(c_1^x)^{-1} \mod p$；
        3. Output：m，明文。
    - Correctness：

$$Dec(sk, Enc(pk, m)) = \ldots = m \mod p$$

    - EI-Gamal不是CCA安全的。

5.3 数字签名

- Syntax：
    - $KeyGen(\lambda) \to (pk, sk)$
    - $Sign(sk, m) \to \sigma$
    - $Verify(pk, m, \sigma) \to 1/0$
- Correctness：$Verify(pk, m, Sign(sk, m)) = 1$
- 安全模型：
    - EUF（不可伪造性）：没有攻击者能够针对一个“新”消息以不可忽略的优势伪造有效的（消息，签名）对。

    - 强不可伪造性：没有攻击者能够产生一个“新”的有效的（消息，签名）对。
- 不可否认性：签名者无法否认签过名的消息。
- MAC比数字签名快。
- RSA签名：私钥签名，公钥验证。

5.4 公私钥密码学特点对比

对称加密	公钥加密
通信双方互相信任	通信双方不需要互相信任
双方共享同一个密钥	公钥和私钥
攻击方式：暴力破解	解数学题（比如因式分解，离散对数问题）
更快	更慢
密钥空间更小	密钥空间更大
例：DES、AES	RSA、EIGamal

6 混合加密

- KEM 密钥封装机制
    - Syntax：
        - $KeyGen(1^n) \to (pk, sk)$
        - $Encaps(pk) \to (c, k)$
        - $Decaps(sk, c) \to k$
    - 安全模型：已知pk，无法伪造合法的(c, k)对。
- 混合加密
    - $KeyGen(1^\lambda) \to (pk, sk)$：
    - $Enc(pk, m) \to (c, c')$：
        1. Input：公钥pk，明文m；
        2. KEM封装：$Encpas(pk) \to (c, k)$；
        3. 对称加密：$Enc'(k, m) \to c'$；
        4. Output：(c, c')，密文。
    - $Dec(sk, (c, c')) \to m$：
        1. Input：私钥sk；密文(c, c')；
        2. KEM解封：$Decaps(sk, c) \to k$；
        3. 对称解密：$Dec'(k, c') \to m$；
        4. Output：m，明文。
    - 混合加密的本质是公钥加密。

7 PKI

- 公开密钥基础建设（PKI）是一套通过公钥密码算法原理与技术提供安全服务的具有通用性的安全基础设施，是能够为电子商务提供一套安全基础平台的技术规范。它通过数字证书管理公钥，通过证书颁发机构（CA）把用户的公钥与其他标识信息捆绑在一起，实现互联网上的用户身份验证。PKI的基础技术包括加密、数字签名、数据完整性机制、数字信封、双重数字签名等。
- 数字证书的基本架构是对称密码学，即利用一对密钥实施加密和解密。私钥主要用于签名和解密，由用户自定义，只有用户自己知道；公钥用于签名验证和加密，可被多个用户共享。

8 身份认证与密钥交换

- 重放攻击：指攻击者发送一个目的主机已接收过的包，来达到欺骗系统的目的。
- 前向安全：即使长期使用的主密钥被泄露，也不会导致过去会话密钥的泄露。这种属性能够保护过去进行的通讯不受密码或密钥在未来暴露的威胁。
- Diffie-Hellman密钥交换协议：
  - 公钥：$(\mathbf{G}, p, g)$，p是模。
  - 私钥：Alice选择a、Bob选择b。
  - Alice发送g^a给Bob，Bob发送g^b给Alice；
  - Alice计算$(g^b)^a$，Bob计算$(g^a)^b$；
  - 使用$K = g^{ab}$作为对称密钥。
  - 无法抵御中间人攻击。
  - 因为每次会话Alice和Bob都会选择新的私钥，生成新的K，所以可以实现前向安全。
- 具有**完美前向安全（PFS）**的基于公钥密码学的密钥交换协议：
  - $A \xrightarrow{\text{"I'm Alice"}, R_A} B$
  - $A \xleftarrow{R_B, [\{R_A, g^b\}_{Alice}]_{Bob}} B$
  - $A \xrightarrow{[\{R_B, g^a\}_{Bob}]_{Alice}} B$
  - 使用$K = g^{ab}$作为对称密钥。

  注：[]表示用私钥签名，{}表示用公钥加密。
  - R可以让对方知道这是对上一条信息的回复，加密提供了机密性，签名提供了完整性。

9 SSL

- SSL协议（简化版）：
  - $A \xrightarrow{R_A} B$
  - $A \xleftarrow{Cert_B, R_B} B$
  - $A \xrightarrow{\{S\}_B, Enc(K, h(msgs||K))} B$
  - $A \xleftarrow{h(msgs||K)} B$
  - A随机选择一个S。
  - Enc表示对称加密。

  - A与B使用$K = h(S, R_A, R_B)$作为会话密钥加密http请求。
- SSL连接：
  - $A \xrightarrow{R_A} B$
  - $A \xleftarrow{R_B, h(msgs||K)} B$
  - $A \xrightarrow{h(msgs||K)} B$
  - A与B使用$K = h(S, R_A, R_B)$作为连接密钥加密数据。
  - 假设SSL会话存在，则S已经为A和B 所知。S是会话密钥，K是连接密钥。
  - SSL连接中没有公钥加密相关操作，安全性依赖于S。

10 比特币与区块链

- 一个区块链是以密码学技术为基础，以去中心化或多中心化的方式，
  - 对大量数据进行组织和维护的一个数据库；
  - 对大量数据进行组织和维护的一套协议；
  - 对数据进行存储和管理，从而在不依赖可信中心机构的情况下提供“可信数据”的一种技术。
  - 可信数据：不可伪造，不可篡改，不可否认，可溯源。
- 如何实现去中心化：每个用户在本地数据库保存一份所有交易记录的副本，以**区块的哈希链**的方式保存。
- 数字签名在比特币中的使用：
  - 对每一笔交易签名。例如：$\sigma = Sign(sk_{Alice}, Transaction\_ID||\text{"Alice pay Bob \$10"})$
  - 实现了去中心化、高安全性、可追溯。
- **Stealth Address 秘密地址技术**
  - 步骤 1：Bob生成并分享他的秘密地址
    - 拥有公钥$(\mathbf{G}, p, G)$，G为椭圆曲线乘法（ECDSA）中的G点；
    - 生成两对公私钥$(A, a), (B, b)$，$A = aG$，$B = bG$，私钥$(a, b) \in Z_p$；A为scan public key，a为scan secret key，B为spend public key，b为spend secret key；
    - stealth address = (A, B)；
    - Bob公开stealth address。
  - 步骤 2：Alice计算出临时地址并发送资金
    - Alice随机选择$r \in Z_p$；
    - 计算出$R = rG$，临时地址$P = H(rA)G + B$，也称为stealth address的公钥（commitment public key）；
    - Alice将资金发送至P，同时在区块链上公开R。
  - 步骤 3：Bob检索资金
    - Bob利用a扫描区块链找到Alice公开的R（怎么扫的不知道，没查到）；
    - 计算出stealth address的私钥$sk = H(aR) + b$；
    - 使用私钥与R计算
      $P = sk \cdot G = (H(aR) + b)G = H(aR)G + bG = GH(arG)G + B = H(raG)G + B = H(rA) + B$
      ；
    - Bob从P拿到Alice发送的资金。

  - 每次交易的r重新选择，所以每次交易的临时地址都不一样。
  - 安全性：由私钥b的保密性（椭圆曲线除法是离散对数问题）来保证，知道b的人才能花掉这笔资金，就算攻击者知道了a，他也无法花掉这笔资金。
- 挖矿：谁能最快地计算出一个新的交易记录块的工作量。