

《密码学及应用》开卷考试宝典

依据中南大学计算机学院网络空间安全系 段桂华 主讲课件全文整理 | 共 7 章 | 左侧可搜索, Ctrl+F 也行, 打印时自动隐藏侧栏

★ 考前速查 (公式 · 模板 · 易错)

一页纸公式表

类别	公式	说明
约简	负数先加模数变正: $-200 \bmod 98 = 94$	密码学中任何整数模 m 的约简都是 非负数
同余	$x \equiv y \pmod m \iff m \mid (x-y)$	差能被 m 整除
乘法逆元	$mx \bmod n = 1$, 存在条件 $(n,m)=1$	用 Euclid 算法求; 满足 $ax+bn=1$ 的 a 即逆元
欧拉函数	$\varphi(p) = p-1$; $\varphi(p^e) = (p-1)p^{e-1}$; $\varphi(mn)=\varphi(m)\varphi(n)$ (m,n 互素)	$ Z_n^* = \varphi(n)$
欧拉定理	$(x,n)=1 \implies x^{\varphi(n)} \equiv 1 \pmod n$	用来 降指数
费马小定理	p 为素数 $\implies x^{p-1} \equiv 1 \pmod p$	欧拉定理的特例
阶	使 $x^t \equiv 1 \pmod n$ 成立的 最小 正整数 t	阶一定整除 $\varphi(n)$
原根	阶 = $\varphi(n)$ 的元素	$g^1 \dots g^{\varphi(n)}$ 组成 Z_n^*
移位密码	$y=(x+k)\%26, x=(y-k)\%26$	密钥空间 25
仿射密码	$y=(ax+b)\%26, x=(a^{-1}y - a^{-1}b)\%26$	密钥空间 $12 \times 26 - 1 = 311$
Vigenère	$y_i=(x_i+k_{i \% m})\%26$	密钥空间 26^m
OTP	同 Vigenère, 但密钥与明文等长且只用一次	密钥空间 26^n , 理论安全
完美快速洗牌	$f(x) = (2x) \bmod (2n+1)$	洗牌次数 = 2 模 $2n+1$ 的阶
RC4 密钥流	$i=(i+1)\%256; j=(j+S_j)\%256$; 交换 S_i, S_j ; $t=(S_i+S_j)\%256; k=S_t$	256 字节 S 盒
线性同余	$S_{n+1} = (a \cdot S_n + b) \bmod m$; 坏种子 $S_{\text{bad}} = -(a-1)^{-1}b \bmod p$	周期 = a 在 Z_p^* 中的阶
LFSR	$b_{n+1} = \sum c_i b_i$ (异或)	最大周期 $2^n - 1$ (本原多项式)
RSA	$n=pq; \varphi(n)=(p-1)(q-1); ed \equiv 1 \bmod \varphi(n); c=m^e \bmod n; m=c^d \bmod n$	安全性: 因子分解难题; $n \geq 1024$ bit
RSA 签名	$s = m^d \bmod n$; 验证 $s^e \bmod n == m$	私钥签名, 公钥验证
ElGamal	$e=g^d \bmod p$; 密文对 $(a=g^k, b=m \cdot e^k)$; 解密 $b \cdot a^{-d} \bmod p$	离散对数难题; 同一明文可不同密文
ECC	$E=dG; C_m=\{kG, P_m+kE\}$; 解密 $P_m+kE-dkG = P_m$	椭圆曲线点群离散对数难题
Knapsack	$y = \sum e_i x_i$; 解密 $t^{-1} \cdot y \equiv \sum d_i x_i \pmod m$	超递增背包 $\{2,3,6,13,27,52\}$, $m=105, t=31$
DSA 签名	$r=(g^k \bmod p) \bmod q; s=(k^{-1}(H(m)+dr)) \bmod q$	验证 $r == ((g^{u1}e^{u2}) \bmod p) \bmod q$
CRT	$x \equiv a_1 T_1 W_1 + \dots + a_n T_n W_n \pmod{M_n}$, $W_i = M_n / m_i, T_i = W_i^{-1} \bmod m_i$	"物不知数": $x \equiv 2(3), x \equiv 3(5), x \equiv 2(7) \rightarrow 23 \pmod{105}$
Shamir 门限	$f(x)=m+a_1x+\dots+a_{k-1}x^{k-1} \bmod p$; 份额 $D_i=f(i)$	恢复用拉格朗日插值, $m = f(0)$
Asmuth-Bloom	$a_i = x \bmod m_i$; 份额 $\{(a_i, m_i)\}$	用 CRT 恢复; $S_k < x < L_{k-1}$ 关系保证安全性
DH 密钥协商	共享密钥 $g^{xy} \bmod p$	公开 p 和原根 g ; 易受中间人攻击
ECB	$C_i=E_k(P_i); P_i=D_k(C_i)$	分组独立; 易受重放攻击
CBC	$C_i=E_k(P_i \oplus C_{i-1}); P_i=C_{i-1} \oplus D_k(C_i)$	IV 可明文传输
CFB	$C_i=P_i \oplus E_k(C_{i-1}); P_i=C_i \oplus E_k(C_{i-1})$	反馈在分组内与分组间

OFB	$C_i = P_i \oplus S_i, S_i = E_k(S_{i-1})$	反馈独立于明文密文
PCBC	$C_i = E_k(P_i \oplus C_{i-1} \oplus P_{i-1}); P_i = C_{i-1} \oplus P_{i-1} \oplus D_k(C_i)$	Kerberos4 用过, 5 改用 CBC

八大解题模板（照着抄卷面）

模板 1 求乘法逆元 $a^{-1} \bmod n$ (Euclid 算法)

第一步·辗转相除（左列竖排）：大数 - 商×小数 = 余数，一直除到余数为 1。

第二步·回代（右列）：从“1 = ...”出发，把中间余数逐层替换成原始两个数。

第三步·读系数：要“求逆”的那个数的系数就是答案；负数加模数变正。

示例 求 $101^{-1} \bmod 234$ ：

$$\begin{array}{rcl}
 234 & - & 2 \times 101 = 32 \\
 101 & - & 3 \times 32 = 5 \\
 32 & - & 6 \times 5 = 2 \\
 5 & - & 2 \times 2 = 1 \\
 2 & - & 2 \times 1 = 0
 \end{array}
 \quad
 \begin{array}{l}
 \text{回代:} \\
 1 = 5 - 2 \times 2 \\
 = 5 - 2(32 - 6 \times 5) = 13 \times 5 - 2 \times 32 \\
 = 13(101 - 3 \times 32) - 2 \times 32 = 13 \times 101 - 41 \times 32 \\
 = 13 \times 101 - 41(234 - 2 \times 101) \\
 = 95 \times 101 - 41 \times 234
 \end{array}$$

$$\text{答: } 101^{-1} \bmod 234 = 95 \quad \text{验算: } 101 \times 95 = 9595 = 41 \times 234 + 1 \checkmark$$

模板 2 计算 $\varphi(n)$

三步：①分解质因数 ②每个 p^e 单独算 $(p-1) \cdot p^{e-1}$ ③全部相乘。

$$\text{示例 } \varphi(18): 18 = 2 \times 3^2 \Rightarrow (2-1) \times 2^0 \times (3-1) \times 3^1 = 1 \times 1 \times 2 \times 3 = 6$$

$$\varphi(155): 155 = 5 \times 31 \Rightarrow (5-1) \times 5^0 \times (31-1) \times 31^0 = 4 \times 1 \times 30 \times 1 = 120$$

$$\varphi(10000): 10000 = 2^4 \times 5^4 \Rightarrow (2-1) \times 2^3 \times (5-1) \times 5^3 = 1 \times 8 \times 4 \times 125 = 4000$$

模板 3 用欧拉定理算大指数 $x^k \bmod n$

三步：①验证 $\gcd(x, n) = 1$ ②算 $\varphi(n)$ ③指数对 $\varphi(n)$ 取模，再算小指数。

$$\text{示例 } 3^{45} \bmod 10: \gcd(3, 10) = 1; \varphi(10) = 4; 45 \bmod 4 = 1 \Rightarrow 3^1 = 3$$

$$2^{100} \bmod 13: \varphi(13) = 12; 100 \bmod 12 = 4 \Rightarrow 2^4 = 16 \equiv 3$$

$$3^{200} \bmod 7: \varphi(7) = 6; 200 \bmod 6 = 2 \Rightarrow 3^2 = 9 \equiv 2$$

注意：指数取模后若余数为 0，结果就是 1。

模板 4 已知明文攻击求仿射密码密钥

题给： $E_{a,b}(7)=1$ 且 $E_{a,b}(12)=4$ ，求解密密钥。

$$\text{第 1 步·列方程相减消 } b: 7a+b \equiv 1, 12a+b \equiv 4 \Rightarrow 5a \equiv 3 \pmod{26}$$

$$\text{第 2 步·求 } a: 5^{-1} = 21 \text{ (Euclid)} \Rightarrow a = 21 \times 3 = 63 \equiv 11 \pmod{26}, \text{ 且 } \gcd(11, 26) = 1 \text{ 合法}$$

$$\text{第 3 步·求 } b: b = (1 - 7 \times 11) \bmod 26 = -76 \equiv 2$$

$$\text{第 4 步·转成解密密钥: } a^{-1} = 11^{-1} = 19; -a^{-1}b = -38 \equiv 14$$

答：加密密钥 (11, 2)；解密密钥 (19, 14)，即 $x = (19y+14)\%26$

$$\text{验算: } D(1) = 33\%26 = 7 \checkmark \quad D(4) = 90\%26 = 12 \checkmark$$

易错：题目问“解密密钥”就绝不能填成加密的 (11,2)。

模板 5 RSA 计算（给定小素数）

示例：p=3, q=5, e=3，加密 m=7。

$n = 3 \times 5 = 15$; $\varphi(n) = (3-1)(5-1) = 8$
 $d = e^{-1} \bmod \varphi(n) = 3^{-1} \bmod 8 = 3$ (因为 $3 \times 3 = 9 \equiv 1$)
公钥 $(e, n) = (3, 15)$; 私钥 $d = 3$
加密: $c = m^e \bmod n = 7^3 \bmod 15 = 343 \bmod 15 = 13$
解密: $m = c^d \bmod n = 13^3 \bmod 15 = 2197 \bmod 15 = 7 \checkmark$

模板 6 中国剩余定理 CRT

示例: $x \equiv 2 \pmod{3}$, $x \equiv 3 \pmod{5}$, $x \equiv 2 \pmod{7}$ 。
 $M = 3 \times 5 \times 7 = 105$; $W_1 = 105/3 = 35$, $W_2 = 21$, $W_3 = 15$
 $T_1 = 35^{-1} \bmod 3 = 2^{-1} \bmod 3 = 2$; $T_2 = 21^{-1} \bmod 5 = 1$; $T_3 = 15^{-1} \bmod 7 = 1$
 $x \equiv 2 \times 35 \times 2 + 3 \times 21 \times 1 + 2 \times 15 \times 1 = 140 + 63 + 30 = 233 \equiv 23 \pmod{105}$

模板 7 完美快速洗牌的循环分解

第 1 步: $2n$ 张牌 $\rightarrow$ 模数 $= 2n+1$; 公式 $f(x) = (2x) \bmod (2n+1)$
第 2 步: 从 1 开始迭代, 乘 2 后超过 $2n$ 就减 $(2n+1)$
第 3 步: 写回 1 停笔, 结果塞进括号
示例 (12 张牌, 模 13):

1 $\rightarrow$ 2 $\rightarrow$ 4 $\rightarrow$ 8 $\rightarrow$ 3 $\rightarrow$ 6 $\rightarrow$ 12 $\rightarrow$ 11 $\rightarrow$ 9 $\rightarrow$ 5 $\rightarrow$ 10 $\rightarrow$ 7 $\rightarrow$ 回到 1
答: (1 2 4 8 3 6 12 11 9 5 10 7), 12 阶循环, 故洗 12 次复原

注意: 中间乘 2 会出现超过 12 的数 (如 16、24), 那只是中间量, 取模后必然回到 1~12。

多循环时取各循环长度的最小公倍数 (lcm), 如课件 8 张牌: (1 2 4 8 7 5)(3 6) $\Rightarrow \text{lcm}(6, 2) = 6$ 次。

模板 8 Shamir 门限方案 (分割 / 恢复)

分割: ① 造 $k-1$ 次多项式 $f(x) = m + a_1x + \dots + a_{k-1}x^{k-1} \bmod p$ ② 算 $D_i = f(i)$, $i=1\dots n$ ③ 分发

恢复: 拿 k 个份额, 用拉格朗日插值还原 $f(x)$, 取 $f(0) = m$ 。

课件例 (3,5): $f(x) = 13 + 10x + 2x^2 \bmod 17$

$D_1=8$, $D_2=7$, $D_3=10$, $D_4=0$, $D_5=11$ (分给 5 人)

取 $D_1=8$ 、 $D_3=10$ 、 $D_5=11$ 恢复 $\Rightarrow$ 秘密 $m = f(0) = 13$

易错点清单

高频错

- 加密密钥 $\neq$ 解密密钥。仿射密码加密是 (a, b) , 解密是 $(a^{-1}, -a^{-1}b)$ 。RSA 加密用公钥 (e, n) , 签名用私钥 d 。
- 算出负数必须加模数变正。 $-2 \% 26 = 24$, $-13 = 87$, $-5 = 21$ 。密码学里约简结果一律非负。
- 求逆元前先验证互素。 $\text{gcd}(m, n) \neq 1$ 就写“无解”, 别硬算。
- 模运算没有除法, 只有乘逆元。 $5a \equiv 3$ 不能“两边除以 5”, 要乘 $5^{-1}=21$ 。
- 仿射密码 a 必须与 26 互素。否则多个明文挤到一个密文, 信息丢失、解不回来。
- 阶必须整除 $\varphi(n)$ 。说“3 的阶是 4 (模 7)”是错的, 因为 4 不整除 6。
- 判断原根看“半路会不会提前回到 1”。只验 $\varphi(n)$ 的素因子对应的指数 $g^{\varphi(n)/q_i}$, 只要有一个 $\equiv 1$ 就不是原根。
- Vigenère 的密钥会循环。相隔密钥长度倍数的位置用同一个密钥字母 $\Rightarrow$ 这是它的死穴 (周期性)。
- 算法公开、密钥保密。香农: “数据的安全基于密钥而不是密码算法。”别答反。
- 序列密码的密钥流绝不能重复使用。一旦重复, 攻击者可逐位推出整段明文。
- ECB 最大的问题是重放攻击 (分组独立、相同明文分组对应相同密文分组), 不是“速度慢”。
- CBC 的 IV 可以明文传输, 它的作用是“把每个消息唯一化”, 本身不需要保密。
- LFSR 最大周期是 2^n-1 不是 2^n , 因为全 0 状态是死循环。
- Shamir 三次传输的加解密顺序可交换; 匿名密钥分配协议是为了“减弱对 KDC 的依赖”。

15. 位承诺: hash 方案靠 r_1 防篡改、 r_2 防猜测; 对称密码方案靠 r 防篡改、 k 做验证。

第 1 章 密码学概述

1.1 信息安全概述

什么是信息安全 / 密码学

课件原文

信息安全 information security：指为数据处理系统而采取的技术的和管理的安全保护，保护计算机硬件、软件、数据不因偶然的或恶意的原因而遭到破坏、更改、显露。

密码学 cryptography：研究编制密码和破译密码的技术科学。

—— 编码学：研究密码变化的客观规律，应用于编制密码以保守通信秘密。

—— 破译学：应用于破译密码以获取通信情报。

五类安全服务（必背）

服务	含义	对应密码技术
认证（鉴别）服务	在网络交互过程中，对收发双方的身份及数据来源进行验证	加密、签名、哈希
访问控制服务	防止未授权用户非法访问资源，包括用户身份认证和用户权限确认	密钥、权限
数据保密性服务	防止数据在传输中被破解或泄露	加密
数据完整性服务	防止数据在传输过程中被篡改	加密、数字摘要
抗否认性服务	防止发送方与接收方在执行各自操作后，否认各自所做的操作	数字签名

八类安全机制

加密机制、数字签名机制、认证机制、数据完整性机制、访问控制机制、路由控制机制、业务流填充机制、公证机制。

记忆线索

把八类机制按"服务 → 机制"配对记：

安全服务	用到的机制
保密性服务	加密机制、业务流填充机制
认证鉴别服务	数字签名机制、认证机制
数据完整性服务	数据完整性机制、认证机制
访问控制服务	访问控制机制
抗否认性服务	公证机制、数字签名机制

三个"如何实现"的对应（课件用图说明）

- 保密性：Alice 张 → （加密）→ 公共网络 → （解密）→ Bob 李。中间有黑客攻击。**解决方法：加密。**
- 完整性与鉴别：Alice 计算 $z = h_k(m)$ 随消息一起发；Bob 收到后自己算 $y = h_k(m)$ ；若 $y \neq z$ ，说明 m 被篡改。**解决方法：密钥 + 哈希函数。**
- 抗否认性：Alice 说"我没发"、Bob 说"你发了"，需要仲裁者判"谁是正确的"。**解决方法：数字签名。**

1.2 密码算法发展

密码系统的组成（五要素）

明文、密文、加解密算法、密钥、密钥空间。

要素	说明
明文	要保护的原始消息（如 "Good luck..."）
密文	加密后看不懂的形式（如 "@**&%#\$..."）

加解密算法	明文 ↔ 密文的变换规则
密钥	算法中的可调参数，是保密的核心
密钥空间	密钥所有可能取值的集合； 密钥空间越大，穷举越难

三个阶段

阶段	特征
1949 年以前	手工阶段，密码学是 艺术 。明文隐写术（隐形墨水、字符格式变化、图像）
1949 ~ 1975	机械阶段，现代密码出现。1949 年 香农 提出"保密系统信息理论"，提出： 数据的安全基于密钥而不是密码算法
1976 年以后	计算机阶段，公钥密码出现。1976 年 Diffie & Hellman 提出不对称密钥密码；1977 年 RSA ；90 年代 ECC

密码算法进一步发展（时间线）

- 1977 年 DES 正式成为标准
- 80 年代出现 IDEA、RCx、CAST 等
- 90 年代 Rijndael、MARS、Twofish、Serpent、RC6 等
- 1989 年混沌密码思想
- 2001 年 Rijndael 成为 DES 的替代者 **AES**
- 2004 年美国 NIST 提出量子密码
- **2005 年**山东大学王小云教授成功破解 MD5、SHA-1
- 国产商用密码算法 **SM** 系列

理论安全 vs 实际安全

	理论安全（无条件安全）	实际安全（计算安全）
定义	攻击者无论截获多少密文，都无法得到足够的信息来唯一地决定明文	如果攻击者拥有无限资源，任何密码系统都可以被破译；但在 有限资源 内无法系统地破解，则称计算上安全
条件	加密密钥长度 ≥ 明文长度，密钥只用一次，用完即丢，即 一次一密密码本	破解代价 > 收益

四种基本攻击类型（必背）

攻击类型	攻击者手里有什么	难度
唯密文攻击	只有一些密文	最难
已知明文攻击	知道一些"明文 ↔ 密文"对	中
选择明文攻击	攻击者 可以选 明文密文对	最易
针对密钥的攻击	目标不是读消息，而是直接拿密钥（ 主要针对公钥密码系统 ）	—

另有两种具体手段：

- **穷举攻击**：尝试穷举所有可能的密钥。当**密钥空间较小**时很有效。
- **字典攻击**：利用一些常用的单词进行组合。（后面 7.1 节还会讲撞库、拖库）

密码算法的两个分类维度（重点）

核心结论

维度一：算法是否公开

类型	算法	密钥
古典密码	保密	保密
现代密码	公开	保密

香农：数据的安全基于密钥而不是密码算法。算法可以公开（全世界帮你查漏洞），只要密钥保密就安全。

维度二：加密密钥是否公开

	对称密码	公钥密码
密钥	加密密钥和解密密钥相同（或相互推导出）	公钥和私钥不同，且 很难从公钥推导出私钥

特点	运算速度快，密钥管理复杂	密钥管理简单，运算速度慢
代表	DES、AES、SM4、RC4	RSA、ECC、SM2、DH
用途	用于加密消息	用于协商密钥、数字签名
密钥数量 (n 人互通)	$n(n-1)/2$	n 个私钥 (公钥公开)

课件比喻

对称密码：把文件放入保险柜、设置密码锁上；开柜子要用**同一个密码**。

公钥密码：把文件发到 lisi@163.com 邮箱；**谁都能往邮箱发**（公钥公开），**只有本人有密码开邮箱**（私钥保密）。

古典密码的两大分类

古典密码

- 代替密码：将明文字符用另外的字符代替
 - 单字母代替 — 单表代替密码（移位、仿射、OTP）
 - 多表代替密码 (Vigenère)
 - 多字母代替 — (分组密码)
- 换位密码：将明文顺序打乱（列换位、矩阵换位）

1.3 古典密码算法

古代密码举例

例子	出处	方法
象形文字的修改	公元前 19 世纪，古埃及法老坟墓	代替
凯撒密码	古罗马	代替 (明文 ilovechina → 密文 knqxgejkpc)
斯巴达密码棒	古希腊	换位 (明文 ilovechina → 密文 iclhoivnea)
希腊密码	古希腊	代替
莫斯密码	—	代替 (最容易发出、最容易接收)

快速判别法

数一数密文和明文的字母是不是同一批：

- 是同一批（字母没变，只是顺序乱了）⇒ 换位
- 不是同一批（字母被换过了）⇒ 代替

例：computer → retupmoc，字母完全相同 ⇒ 换位（别看它长得陌生就以为是代替）。

① 数学基础（1.3 里嵌入的，第 2 章全靠它）

整除与约简

$d \mid n$ ：d 整除 n，即存在整数 k 使 $n = kd$ 。

性质： $a \mid b, b \mid c \Rightarrow a \mid c$ ； $d \mid x, d \mid y \Rightarrow d \mid (ax+by)$ 。

约简定义

n 模 m 的约简：n 除以 m 的余数 r， $0 \leq r < |m|$ ，记作 $r = n \% m$ 或 $n \bmod m$ ，m 称为模数。

在密码学中：任何整数模 m 的约简都是非负数。

课件例： $(24+3) = 27 \% 26 = 1$ ； $(1-3) = -2 \% 26 = 24$ 。

同余

定义： x 模 m 同余 $y \iff m \mid (x-y)$ ，记作 $x \equiv y \pmod{m}$ 。

例： $27 \equiv 6 \pmod{7}$ ； $29 \equiv 5 \pmod{12}$ 。

同余定理（6 条）

定理 1 (等价关系): 自反性、对称性、传递性。

定理 2: $ax \equiv ay \pmod{m}$, 若 $(a,m)=1$, 则 $x \equiv y \pmod{m}$ 。

(想"约掉"因数, 必须先验证它与模数互素!)

例: $95 \equiv 25 \pmod{7}$, $(5,7)=1 \implies 19 \equiv 5 \pmod{7}$

定理 3: $x \equiv y \pmod{m}$, $a > 0 \implies ax \equiv ay \pmod{am}$ 。

例: $19 \equiv 5 \pmod{7}$, $a=4 \implies 76 \equiv 20 \pmod{28}$

定理 4: $x \equiv y \pmod{m}$, 若 $d|(x,y,m)$, 则 $x/d \equiv y/d \pmod{m/d}$ 。

例: $190 \equiv 50 \pmod{70}$, $d=10 \implies 19 \equiv 5 \pmod{7}$

定理 5: $x \equiv y \pmod{m}$, 如果 $d|m$, 则 $x \equiv y \pmod{d}$ 。

定理 6: $a \equiv b \pmod{m_i}$, $i=1\dots k \implies a \equiv b \pmod{[m_1, \dots, m_k]}$ 。

特别地: p, q 是两个不同素数, 若 $a \equiv b \pmod{p}$ 且 $a \equiv b \pmod{q}$, 则 $a \equiv b \pmod{pq}$ 。

例: $19 \equiv 7 \pmod{3}$, $19 \equiv 7 \pmod{4}$, $(3,4)=1 \implies 19 \equiv 7 \pmod{12}$

素数

正整数 p 是素数, 当且仅当它不能被任何整数 d 整除, 其中 $1 < d \leq p^{1/2}$ 。

算术基本定理: 每一个正整数都可以分解为素数的乘积, 而且这种分解是唯一的。

$n = p_1^{e_1} p_2^{e_2} \dots p_m^{e_m}$ 例: $3960 = 2^3 \times 3^2 \times 5 \times 11$; $400 = 2^4 \times 5^2$

重要性

"分解唯一"是 RSA 的安全根基 —— 想从 n 反推出 p 和 q , 就是要做唯一分解, 这是难题。

Euclid 算法

定理

整数 x 和 y 不同时为 0, 则其最大公因子 (x,y) 是以 $ax + by$ 表示的最小正整数。

作用: 找到整数 a 和 b , 表示 x 和 y 的最大公因子: $ax + by = (x,y)$ 。

例: $(3,5) = 2 \times 3 - 1 \times 5 = 1$; $(9,15) = 2 \times 9 - 1 \times 15 = 3$

【课件原例】求 $(100, 23)$

左列: 辗转相除

$$100 - 4 \times 23 = 8$$

$$23 - 2 \times 8 = 7$$

$$8 - 1 \times 7 = 1$$

$$7 - 7 \times 1 = 0$$

右列: 回代

$$1 = 8 - 1 \times 7$$

$$= 8 - 1 \times (23 - 2 \times 8)$$

$$= 3 \times 8 - 1 \times 23$$

$$= 3 \times (100 - 4 \times 23) - 1 \times 23$$

$$= 3 \times 100 - 13 \times 23$$

$$\text{所以: } 1 = 3 \times 100 - 13 \times 23$$

课件还给了矩阵表格写法 (把回代打包成矩阵连乘): 每个除法式 $A - q \times B = r$ 对应矩阵 $\begin{pmatrix} 0 & 1 \\ 1 & -q \end{pmatrix}$, 全部乘起来, 最后拿原数当系数。

乘法逆元

定义与定理

定义: m 模 n 的乘法逆 t (记作 $m^{-1} \% n$) 满足 $m \times t \% n = 1$ 。

存在条件: $(n, m) = 1$ 。不互素 = 无解。

定理: 满足 $ax + bn = 1$ 的整数 a 是一个 x 模 n 的乘法逆元。

举例: $2^{-1} \% 5 = 3$ (穷举法); $3^{-1} \% 10 = 7$; $15^{-1} \% 26 = 7$ (因 $7 \times 15 - 4 \times 26 = 1$)

课件例: 计算 $23^{-1} \% 100 \implies$ Euclid 得 $3 \times 100 - 13 \times 23 = 1 \implies 23^{-1} \% 100 = -13 = 87$ 。

课件练习: $21^{-1} \% 25 = 6$ (因 $-5 \times 25 + 6 \times 21 = 1$); $123^{-1} \% 421$ (课后练)。

② 代替密码

移位密码（凯撒密码）

公式

英文字母 a~z 用数字 0~25 表示。

$$\text{加密: } y = f(x) = (x + k) \% 26 \quad \text{解密: } x = f(y) = (y - k) \% 26$$

密钥空间: 25 ($k = 1 \sim 25$; $k=0$ 等于没加密)

课件例: $(24+3) \% 26 = 1$; $(1-3) \% 26 = 24$

仿射密码

公式

算法思想: 增加密钥参数, 扩大密钥空间。

$$\text{加密: } y = E_{a,b}(x) = (ax + b) \% 26 \quad \text{解密: } x = D_{a,b}(y) = (a^{-1}y - a^{-1}b) \% 26$$

密钥空间: 由于 a^{-1} 必须存在, a 必须与 26 互素 (共 12 个: 1,3,5,7,9,11,15,17,19,21,23,25), b 有 26 个, 故 $12 \times 26 = 312$ 。

课件例: $(a,b) = (9,3)$, $y = (9x+3)\%26$, 解密 $x = (3y+17)\%26$ 。

明文 meet me at midnight $\rightarrow$ 密文 hnns hn ds hxeqxfos。

为什么 a 要与 26 互素

若 $a = 2$, 加密 $y = (2x+b)\%26$: $x=0$ 与 $x=13$ 都会得到同一个 y (因为 $26 = 2 \times 13$)。

多个明文挤到同一密文 $\Rightarrow$ 信息丢失, 解不回来。

攻击方法

针对密钥的攻击 / 唯密文攻击 / 选择明文攻击 / 已知明文攻击

穷举攻击: 尝试可能的密钥 (最多 312 次)。

唯密文攻击: 字母频率攻击。

选择明文攻击: $E_{a,b}(0) = b$, $E_{a,b}(1) = a+b \Rightarrow b = E_{a,b}(0)$, $a = (E_{a,b}(1) - b) \% 26$

已知明文攻击:

$$a = (x-y)^{-1}(E_{a,b}(x) - E_{a,b}(y)) \% 26$$

$$b = (E_{a,b}(x) - ax) \% 26$$

Vigenère 密码

公式

算法原理: 将明文中的每个字母移动 k_i 位, k_i 为密钥, $0 \leq k_i \leq 25$ 。

密钥是序列 $k = (k_0, k_1, \dots, k_{m-1})$, m 为密钥长度; 明文 $x = (x_0, \dots, x_n)$ 分成长度为 m 的段。

$$\text{加密: } y_i = (x_i + k_{i \% m}) \% 26 \quad \text{解密: } x_i = (y_i - k_{i \% m}) \% 26$$

密钥特征: 加密密钥和解密密钥相同。

密钥空间: 与密钥长度 m 有关, 为 26^m 。

代替特征

单字母代替、多表代替 —— 即明文中**相同字母可能替换成不同的密文**。

原因: 每个位置的密钥字母不同 $\Rightarrow$ 移位量不同 $\Rightarrow$ 同一个明文字母变成不同密文。

有周期性的弱点

若两个字符出现的间隔是**密钥长度的倍数**, 则它们将以同样的方法加密 (因为位置相隔 m 时用的是同一个密钥字母)。

多轮加密：若一个明文使用密钥长度为 m 的 Vigenère 加密，得到的密文再用长度为 n 的密钥加密，其结果与用长度 $\text{lcm}(m,n)$ 的 Vigenère 加密一样。若 m 、 n 互素，则 $\text{lcm}(m,n) = m \times n$ 。

破译：Kasiski 与 Friedman 方法

总体思路：找到密钥长度 $m \rightarrow$ 按 m 分组 $\rightarrow$ 每组变成移位密码 $\rightarrow$ 频率攻击。

Kasiski 方法：找重复出现的片段，量它们的距离；距离的公约数就是 m 。

（密文重复 = 明文重复 + 距离是 m 的倍数，两个条件缺一不可）

Friedman 方法（重合指数）：不看重复片段，看字母分布。

文本类型	重合指数	说明
随机字符流	≈ 0.038	$= 1/26$ ，字母各占 $1/26$ ，分布最平
英文字符流	≈ 0.067	分布“扎堆”（e 最多）
Vigenère 密文	≈ 0.047	介于两者之间

怎么用来找 m ：猜一个 m ，按 m 分组，算各组重合指数的平均值。若接近 $0.067 \Rightarrow m$ 猜对了（每组都是英文字符流）；若接近 $0.038 \Rightarrow$ 猜错了。

OTP 密码（一次一密）

公式

算法原理：一次一密加密方式，明文和密钥一样长，密钥本身只使用一次。

加密解密与 Vigenère 相同： $y_i = (x_i + k_i) \% 26$ ， $x_i = (y_i - k_i) \% 26$

密码特点：密钥和明文等长，**克服了 Vigenère 易受重合指数攻击的缺点**。但密钥的产生、保存和传输非常重要。密钥空间与明文长度 n 有关，为 26^n 。

联系

OTP 就是 1.2 节讲的“一次一密密码本”的实现 —— **理论上唯一不可破**，但密钥管理极端困难（密钥要和明文一样长、只能用一次），所以现实中用不了。

③ 换位密码

共性

算法原理：将明文字符在消息内变换位置。

算法特点：**明文和密文一样长**（字母一个不多一个不少）。

举例：如将文本翻转，computersystems $\rightarrow$ SMETSYSRETUPMOC。

唯密文攻击可能得到多种不同的破译结果；如 live – evil – vile（同样是这 4 个字母，分不清原文）。

列换位

做法：按行填格子 $\rightarrow$ 按列读出。

课件例：明文 `canyouunderstand`（16 个字母），按 4 个字符一行排列：

```
第1行  c  a  n  y
第2行  o  u  u  n
第3行  d  e  r  s
第4行  t  a  n  d
```

按列读出：codt | auea | nurn | ynsd

密文：CODTAUEANURNYNSD

解密：反过来 —— 把密文按 4 个字符一列排回去，再按行读出，还原 `canyouunderstand`。

加关键词的版本：给每列贴标签（课件用的是 `type`，标为 t3 y4 p2 e1），按标签字母的字母序读列：

标签排序: $e(1) < p(2) < t(3) < y(4)$
读列顺序: 第4列(e) $\rightarrow$ 第3列(p) $\rightarrow$ 第1列(t) $\rightarrow$ 第2列(y)
= ynsd + nurn + codt + auea
密文: YNSDNURNCODTAUEA

矩阵换位

用"置换矩阵"指定每个位置搬到哪。

课件例: 加密置换矩阵 $f = (1\ 2\ 3\ 4 / 2\ 4\ 1\ 3)$, 即 $f(1)=2, f(2)=4, f(3)=1, f(4)=3$ 。

加密规则: 明文第 j 位的字符 $\rightarrow$ 密文第 $f(j)$ 位。

明文分组: c a n y | o u u n | d e r s | t a n d

第一组 c a n y:

c(第1位) $\rightarrow$ 第 $f(1)=2$ 位

a(第2位) $\rightarrow$ 第 $f(2)=4$ 位

n(第3位) $\rightarrow$ 第 $f(3)=1$ 位

y(第4位) $\rightarrow$ 第 $f(4)=3$ 位

密文组: n c y a

密文: NCYAUONURDSENTDA

解密: 用逆置换 $f^{-1} = (1\ 2\ 3\ 4 / 3\ 1\ 4\ 2)$ 。

为什么能还原: $f \circ f^{-1} = (1\ 2\ 3\ 4 / 1\ 2\ 3\ 4) =$ 恒等置换 (谁都待在原位)。

联系

这就是 1.4 节的置换群: 置换矩阵 $\leftrightarrow$ 置换, $\circ$ $\leftrightarrow$ 置换的复合, 恒等置换 $\leftrightarrow$ 单位元, $f^{-1} \leftrightarrow$ 逆置换。换位密码的本质就是置换。

CTF 中常用密码 (课件补充)

- Base64: 3 个 8 位字节转化为 4 个 6 位的字节, 之后在 6 位前面补两个 0, 形成 4 个 8 位字节。
- 猪圈密码 (Pigpen Cipher / 九宫格密码 / 共济会密码): 以格子为基础的简单替代式密码。
- 培根密码: 明文字母被 5 字符组成序列替换。
- jjencode / aaencode: 转换成特定符号 / 网络表情。

1.4 密码学数学知识

① 群

二元运算 (3 条)

设 S 为集合, 函数 $f: S \times S \rightarrow S$ 称为 S 上的二元运算 (用 $*$ 表示), 满足: 可计算性、单值性、封闭性。

$*$ 通常指加法 (+) 和乘法 ($\times$ 或 $\cdot$)。

结合律: $(a*b)*c = a*(b*c)$ 交换律: $a*b = b*a$

单位元: 加法为 0, 乘法为 1。

群的 4 条规矩

1. 封闭性: 集合内任两元素运算结果仍在集合内
2. 结合律: $(a*b)*c = a*(b*c)$
3. 有单位元: 存在 e 使 $a*e = a$
4. 每个元素有逆元: 对每个 a 存在 a^{-1} 使 $a*a^{-1} = e$

再加交换律就是交换群 (阿贝尔群)。

课件例子

$\langle \mathbb{Z}_6, + \rangle$ 是交换加群: $\mathbb{Z}_6 = \{0,1,2,3,4,5\}$, $\oplus$ 为模 6 加法。单位元 0; $1 \oplus 5=0$ 、 $2 \oplus 4=0$ 、 $3 \oplus 3=0 \implies 1$ 和 5、2 和 4 为逆元, 3 和 0 的逆元不变。

$\langle \mathbb{Z}_6^*, \times \rangle$ 是交换乘群: $\mathbb{Z}_6^* = \{1,5\}$, $\times$ 为模 6 乘法。单位元 1; $1 \times 1=1$ 、 $5 \times 5=1 \implies 1$ 和 5 的逆元分别是 1 和 5。

$\mathbb{Z}^* = \mathbb{Z} \setminus \{0\}$ 不是群: 满足乘法交换律、结合律, 有单位元 1, 但 **不是每个元素都有逆元** (如 2 的逆元 $1/2$ 不在整数里)。

② 剩余系

剩余类 (等价类)

定义: 整数模 m 剩余类 (等价类) 的集合, 记作 $\mathbb{Z}/m$ 或 $\mathbb{Z}_m$ 。

给定整数 x 和模数 m , x 模 m 等价类 $= \{y \in \mathbb{Z} : y \equiv x \pmod{m}\}$, 通常记为 $\bar{x}$, 又称 x 模 m 的同余类或剩余类。

举例 (模 12): $0 = 12 = -12 = 2400$; $1 = 13 = -11 = 2401$

定义: 模 m 的剩余类有 m 个: $0, 1, \dots, m-1$ 。

完全剩余系

定理 1

已知正整数 m , 则 m 个整数 $x_0, x_1, \dots, x_{m-1}$ 为模 m 的一个完全剩余系的充要条件是它们两两不同余。

课件给的 5 种形式 (以 $m=10$ 为例):

名称	内容
最小非负完全剩余系	0, 1, 2, 3, 4, 5, 6, 7, 8, 9
最小正完全剩余系	1, 2, 3, 4, 5, 6, 7, 8, 9, 10
最大非正完全剩余系	-9, -8, -7, -6, -5, -4, -3, -2, -1, 0
最大负完全剩余系	-10, -9, -8, -7, -6, -5, -4, -3, -2, -1
绝对值最小完全剩余系	-5, -4, -3, -2, -1, 0, 1, 2, 3, 4 或 -4, -3, -2, -1, 0, 1, 2, 3, 4, 5

定理 2 / 定理 3

定理 2: 已知整数 m, a, b , 满足 $(a,m)=1$ 。若 x 遍历模 m 的一个完全剩余系, 则 $ax+b$ 也遍历模 m 的一个完全剩余系。

例: $m=4, a=3, b=1, x \in \{0,1,2,3\} \implies ax+b \in \{1,0,3,2\}$

反例: $m=4, a=2, b=1, x \in \{0,1,2,3\} \implies ax+b \in \{1,3,1,3\}$ (不平移整圈)

定理 3: 设 m, n 是两个互素的正整数, 如果 x, y 分别遍历模 m, n 的完全剩余系, 则 $nx+my$ 遍历模 mn 的完全剩余系。

简化剩余系

定义

整数模 m 的简化剩余类 (等价类) 的集合, 记作 $\mathbb{Z}_m^*$ 。

即 $\mathbb{Z}_m$ 中与 m 互素的元素的集合。

举例: $\mathbb{Z}_{10}^* = \{1,3,7,9\}$; $\mathbb{Z}_{15}^* = \{1,2,4,7,8,11,13,14\}$; $\mathbb{Z}_{11}^* = \{1,2,3,4,5,6,7,8,9,10\}$

元素个数 = 欧拉函数 = $\varphi(m)$ 。

$\mathbb{Z}_n$ vs $\mathbb{Z}_n^*$ 的区别 (易混)

	$\mathbb{Z}_n$	$\mathbb{Z}_n^*$
含义	完全剩余系	简化剩余系 (只含与 n 互素的)
含 0 吗	含	不含
元素个数	n	$\varphi(n)$

配加法	群 ✓	—
配乘法	不是群 ✗	群 ✓
用途	加法运算	乘除运算 (RSA、DH 都靠它)

星号的作用：筛出“有乘法逆元的那部分”，让乘法也构成群（能做除法）。

③ 欧拉函数

定义与公式

定义：对正整数 n ，与其互素的**小于等于 n 的正整数的个数**，表示为 $\varphi(n)$ 。

定理 1：若正整数 m, n 满足 $(m, n)=1$ ，则 $\varphi(mn) = \varphi(m)\varphi(n)$

定理 2： $n = p_1^{e_1} p_2^{e_2} \dots p_m^{e_m}$ ，则

$$\varphi(n) = (p_1 - 1)p_1^{e_1 - 1} (p_2 - 1)p_2^{e_2 - 1} \dots (p_m - 1)p_m^{e_m - 1}$$

特例： p 为素数时， $\varphi(p) = p - 1$ 。另 $\varphi(1) = 1$ 。

课件例： $\varphi(35) = \varphi(5 \times 7) = 4 \times 6 = 24$

$\varphi(10000) = \varphi(2^4 \times 5^4) = (2-1) \times 2^3 \times (5-1) \times 5^3 = 1 \times 8 \times 4 \times 125 = 4000$

手算三步

① 分解质因数 ② 每个 p^e 单独算 $(p-1) \cdot p^{e-1}$ ③ 全部相乘

例： $\varphi(18)$ ： $18 = 2 \times 3^2 \Rightarrow 1 \times 1 \times 2 \times 3 = 6$

④ 欧拉定理与费马小定理

公式

欧拉定理：对 $n > 1$ ，如果 $(x, n) = 1$ ，则有 $x^{\varphi(n)} \equiv 1 \pmod{n}$

费马小定理：对任意的整数 x 和素数 p ，有 $x^{p-1} \equiv 1 \pmod{p}$

怎么理解（“转圈”）

因为 x 与 n 互素 $\Rightarrow$ 一直乘 x 不会跑出 Z_n^* ；而 Z_n^* 一共只有 $\varphi(n)$ 个元素 $\Rightarrow$ 转 $\varphi(n)$ 步一定回到 1。

例 ($n=9, x=2$)： $2 \rightarrow 4 \rightarrow 8 \rightarrow 7 \rightarrow 5 \rightarrow 1$ ，走 6 步回到 1，而 $\varphi(9)=6 \checkmark$

用途：降指数

课件例：求 3^{303} 展开后的最后三位数字。

解： $3^{303} \equiv 3^3 \equiv 27 \pmod{1000}$ ，因为 $\text{ord}_{1000}(3) = 100$ ，而 $303 \equiv 3 \pmod{100}$ 。

课件还要求：求 8^{1001} 展开后的最后四位数字。

费马小定理是欧拉定理的特例： p 为素数时 $\varphi(p) = p-1$ 。

⑤ 阶

定义

幂的定义：设 $\langle G, * \rangle$ 是群， $g \in G$ ， $n \in \mathbb{Z}$ ，则 g 的 n 次幂为 $g^n = g^{n-1} * g$ ($n > 0$)、 $(g^{-1})^{-n}$ ($n < 0$)、 e ($n=0$)。

阶的定义：对群 $\langle G, * \rangle$ ， $g \in G$ ，使得等式 $g^t = e$ 成立的**最小正整数 t** 称为 g 的阶（指数），记做 $|g| = t$ 。 g 称为 t 阶元。若不存在这样的整数 t ，则 g 称为无限阶元。

性质：群中任何元素 g 与其逆元 g^{-1} 具有相同的阶。

课件例：分析 $\langle Z_{10}^*, \times \rangle$

$Z_{10}^* = \{1, 3, 7, 9\}$ ， $\varphi(10) = 4$

x	自乘过程	阶
1	1	1
3	3 → 9 → 7 → 1	4
7	7 → 9 → 3 → 1	4
9	9 → 1	2

1、4、4、2 全部整除 $\varphi(10)=4$ ✓ 原根是 3 和 7 (阶为 4)。

⑥ 原根 (生成元、本原元)

定义

对正整数 n , 如果 g 是模 n 的原根, 那么对于任意的 $x \in \mathbb{Z}_n^\times$, 都存在 d , 使得 $g^d \equiv x \pmod{n}$ 。

即 $g^1, g^2, \dots, g^{\varphi(n)}$ 组成模 n 的简化剩余系。

$\mathbb{Z}_n^\times$ 中的阶: 对任意 $x \in \mathbb{Z}_n^\times$, 即 $(g, n)=1$, 存在正整数 t 满足 $g^t \equiv 1 \pmod{n}$, 则满足条件的最小正整数 t 是 g 模 n 的指数或阶, 记为 $\text{ord}_n(g)$ 。

循环群: 设 $\langle G, * \rangle$ 是群, 若存在元素 $g \in G$, 使得 $G = \{g^k \mid k \in \mathbb{Z}\}$, 称 G 为循环群, 记 $G = \langle g \rangle$, g 称为生成元。

性质 (重要)

模 n 的原根 g 具有最大的阶 $\varphi(n)$ 。

- 若 g 是模 n 的原根, $\text{ord}_n(g) = \varphi(n)$
- 若 g 不是模 n 原根, 则 $\text{ord}_n(g) \mid \varphi(n)$

应用: 对整数 $a, n > 1$, $(a, n)=1$, 则 $a^d \equiv a^k \pmod{n}$ 的充要条件是 $d \equiv k \pmod{\text{ord}_n(a)}$ 。

判定定理

定理 1: 模 n 原根 g 存在的充要条件是 $n = 2, 4, p^e, 2p^e$ (p 为奇素数, $e \geq 1$)。

举例: 6 有原根 5; 9 有原根 2。

定理 2: 设 $n > 1$, $\varphi(n) = q_1 \times q_2 \times \dots \times q_k$ (q_i 为素因子), 则 g 是模 n 的一个原根的充要条件是

$$g^{\varphi(n)/q_i} \not\equiv 1 \pmod{n}, \quad i = 1, \dots, k$$

举例: 对 $n=11$, 测试 $g=2, g=3$ 。

手算示范

验证 3 是否模 41 的原根: 41 为素数, $\varphi(41) = 40 = 2^3 \times 5$, 素因子为 2 和 5。

只需验 $3^{40/2} = 3^{20}$ 与 $3^{40/5} = 3^8$:

$$3^2 = 9; 3^4 = 81 \equiv -1 \pmod{41}; 3^8 = (3^4)^2 \equiv 1 \pmod{41}$$

因为 $3^8 \equiv 1$, 不满足条件 $\Rightarrow$ 3 不是模 41 的原根 (3 的阶为 8, 而 $8 \nmid 40$)。

⑦ 快速指数算法

算法思想

计算 x^e 时, 将 e 表示为一个二进制整数 $e = e_0 + e_1 \cdot 2 + e_2 \cdot 2^2 + \dots + e_n \cdot 2^n$, $e_i \in \{0, 1\}$ 。

$$x^e = x^{e_0} \cdot (x^2)^{e_1} \cdot (x^4)^{e_2} \dots (x^{2^n})^{e_n}$$

若 $e_i=0$ 则忽略该项; 若 $e_i=1$ 则包含该项。

举例: $x^5 = x^{(101)} = x^1 \cdot (x^2)^0 \cdot (x^4)^1$; $x^{13} = x^{(1101)} = x^1 \cdot (x^2)^0 \cdot (x^4)^1 \cdot (x^8)^1$

三元组表格法 (考试用这个)

设置三元组 (X, E, Y) , 初始值为 $(X, E, Y) = (x, e, 1)$

- 若 E 是奇数: $Y = X \cdot Y$, $E = E - 1$
- 若 E 为偶数: $X = X \cdot X$, $E = E / 2$
- 当 $E = 0$ 时, Y 的值即为 x^e

举例: 计算 x^5

X	E	Y	说明
x	5	1	初始
x	4	x	$E=5$ 奇: $Y = x \cdot 1 = x$, $E = 5 - 1 = 4$
x^2	2	x	$E=4$ 偶: $X = x \cdot x$, $E = 4 / 2$
x^4	1	x	$E=2$ 偶: $X = x^2 \cdot x^2$, $E = 2 / 2$
x^4	0	x^5	$E=1$ 奇: $Y = x^4 \cdot x = x^5$, $E = 0 \Rightarrow$ 停

口诀: 遇到奇数 $\rightarrow$ 乘到 Y 里、然后减 1; 遇到偶数 $\rightarrow X$ 自己平方、然后除 2。

课件练习: ① 计算 $2^{1000} \bmod 89$ ② 计算 $2^{57} \% 59$

课件里的实际演算 ($3^{80} \bmod 1000 = 601$, $3^{100} \bmod 1000 = 1$):

$3^{80} \bmod 1000$ 3 80 1 9 40 1 81 20 1 561 10 1 721 5 1 721 4 721 841 2 721 281 1 721 $\Rightarrow 281 \times 721 = 601$	$3^{100} \bmod 1000$ 3 100 1 9 50 1 81 25 1 561 (即 81) 12 81 721 6 81 841 3 81 281 2 121 281 1 121 $\Rightarrow 281 \times 121 = 1$
--	--

⑧ 置换群

定义

定义 1: 设 $S = \{1, 2, \dots, n\}$, S 上的任何**双射映射函数** $\delta: S \rightarrow S$ 称为 S 上的 n 元置换, 记为 $\delta = (1 \ 2 \ \dots \ n / \delta(1) \ \delta(2) \ \dots \ \delta(n))$ 。

定义 2: 设 $S = \{1, 2, \dots, n\}$, S 上的 $n!$ 个置换构成集合 S_n , 则称 S_n 与置换的复合运算 $\circ$ 构成的群 $\langle S_n, \circ \rangle$ 为 S 上的 n 元对称群; $\langle S_n, \circ \rangle$ 的任意子群称为 S 上的 n 元置换群。

元素个数为 $n!$; $g \circ h$ 表示**先应用 h 再应用 g** 。

性质

$\langle S_n, \circ \rangle$ 是群: 满足结合律, 有单位元 e , 对任意置换 g 存在逆置换 g^{-1} 。

平凡置换 e : 没有移动任何元素的置换, 即对所有 i 有 $e(i) = i$ 。

不可交换性: 课件例 $g = (1 \ 2 \ 3 / 2 \ 3 \ 1)$ 、 $h = (1 \ 2 \ 3 / 3 \ 2 \ 1)$, 则 $g \circ h \neq h \circ g$ 。

应用: 换位密码、现代对称密码中的置换等等。

不相交循环

k 循环: 在一个 n 元置换 g 中, 有 k 个元素满足 $f(i_1)=i_2, f(i_2)=i_3, \dots, f(i_k)=i_1$; 对于任意 $j \notin \{i_1, \dots, i_k\}$, $f(j)=j$, 则称 g 为 k 循环, 记作 $(i_1 \ i_2 \ \dots \ i_k)$ 。

例: $(1 \ 2 \ 3 / 2 \ 1 \ 3) = (1 \ 2)$; $(1 \ 2 \ 3 / 2 \ 3 \ 1) = (1 \ 2 \ 3) = (2 \ 3 \ 1) = (3 \ 1 \ 2)$

注意: 一个 k 循环有 k 种表示法。比较 $(1 \ 2 \ 3)$ 与 $(1 \ 3 \ 2)$ 是不同的。

循环置换的阶: 若置换 g 是一个 k 循环, 则 $g^k = e$, g 的阶为 k 。

不相交循环: 若循环 g 和 h 的元素列表不相交, 则 $gh = hg$ 。

置换的表示：任何置换都可以表示为不相交循环的乘积，不考虑乘积次序时该表达式是**唯一的**。

置换 g 的阶 k = 不相交循环分解中各循环长度的最小公倍数 (lcm)

课件例： $g = (1\ 2\ 3\ 4\ 5\ 6\ 7\ 4\ 3\ 2\ 5\ 7\ 6\ 1) = (1\ 4\ 5\ 7)(2\ 3)(6)$ ，循环长度 4、2、1 $\Rightarrow$ 阶 = $\text{lcm}(4,2,1) = 4$ 。

⑨ 洗牌

三种洗牌

$2n$ 张牌记作 $1, 2, \dots, n, n+1, \dots, 2n-1, 2n$

- 不好的洗牌（两半交错）： $n+1, 1, n+2, 2, \dots, 2n-1, n-1, 2n, n$
- 完美洗牌： $1, n+1, 2, n+2, \dots, n-1, 2n-1, n, 2n$
- 完美快速洗牌（本章重点）： $f(x) = (2x) \bmod (2n+1)$

定理

若 e 为 2 模 $2n+1$ 的阶，即 e 是满足 $2^e \equiv 1 \pmod{2n+1}$ 的最小正整数。那么对一副有 $2n$ 张的牌经过 e 次洗牌后，所有的牌都第一次返回到它们的起始位置。

课件例（8 张牌）： $1\ 2\ 3\ 4\ 5\ 6\ 7\ 8 \rightarrow 2\ 4\ 6\ 8\ 1\ 3\ 5\ 7 = (1\ 2\ 4\ 8\ 7\ 5)(3\ 6)$ ，循环长度 6 和 2 $\Rightarrow$ $\text{lcm} = 6$ 次洗牌。

课件例（14 张牌）： $1..14 \rightarrow 2\ 4\ 6\ 8\ 10\ 12\ 14\ 1\ 3\ 5\ 7\ 9\ 11\ 13 = (1\ 2\ 4\ 8)(3\ 6\ 12\ 9)(5\ 10)(7\ 14\ 13\ 11) \Rightarrow \text{lcm}(4,4,2,4) = 4$ 次洗牌。

课件例（12 张牌）：模数 13， $1 \rightarrow 2 \rightarrow 4 \rightarrow 8 \rightarrow 3 \rightarrow 6 \rightarrow 12 \rightarrow 11 \rightarrow 9 \rightarrow 5 \rightarrow 10 \rightarrow 7 \rightarrow 1$ ，一个 12 阶循环 $\Rightarrow$ 12 次洗牌。

课件还提到：“如果一副 50 张的牌洗得好，重复洗 8 次后所有的牌将返回初始位置。”

第 2 章 现代密码算法

2.1 对称密码算法

课件原文

对称密码算法的特点

加密 —— 将文件放入保险柜，设置密码锁上；解密 —— 用密码打开保险柜，取出文件。

密钥 —— 加密密钥和解密密钥**相同或相容**；分组密码 —— 将明文分组进行加密；性能 —— **运算速度快，密钥管理复杂**。

常用算法：DES、IDEA、AES、RC4、RC6、SM...

2.1.1 DES 算法

背景

1973 年 5 月 15 日 NBS 开始公开征集标准加密算法 DES (Data Encryption Standard)，并公布了设计要求：

- 算法必须提供高度的安全性
- 算法必须有详细的说明，并易于理解
- **算法的安全性取决于密钥，不依赖于算法**
- 算法适用于所有用户
- 算法适用于不同应用场合
- 算法必须高效、经济
- 算法必须能被证实有效
- 算法必须是可出口的

1974 年 8 月 27 日第二次征集，IBM 提交了算法 LUCIFER；1975 年 3 月公开发表；**1977 年 1 月**由美国国家标准局 NBS 颁布为数据加密标准 DES；1983 年 ISO 同意将 DES 作为国际标准，称之为 DEA-1；1998 年美国不再将 DES 作为联邦加密标准，但其结构和部件一直作为分组密码设计的标准参照物。

DES 核心参数

- 分组加密算法、对称密码
- **64 位分组**，密钥长度为 64 位（实际为 56 位，去掉 8 位奇偶校验位）
- 三个步骤：**初始置换 IP**、**16 轮迭代的乘积变换**、**逆初始置换 IP^{-1}**
- **基本思想：混淆与扩散**
 - **混淆**：复杂化密文与明文、密钥之间的统计特性
 - **扩散**：将每一位明文的影响尽可能多地分散到多个输出密文中，以隐藏明文的统计特性
- **IP 和 IP^{-1}** 的作用主要是打乱输入的 ASCII 码字划分关系，并将明文校验码变成 IP 输出的一个字节

乘积变换（核心）

将经过 IP 置换后的数据分成 **32 位的左右两组**，在迭代过程中彼此左右交换位置。每次迭代时只对右边的 32 位进行一系列的加密变换，然后把左边的 32 位与右边得到的 32 位逐位进行异或操作，作为下一轮迭代时左边的段。

$$\text{迭代公式: } L_i = R_{i-1}, R_i = L_{i-1} \oplus f(R_{i-1}, k_i)$$

$\oplus$ ：按位异或运算符，即按位作模 2 相加。运算规则：1 $\oplus$ 0=1，0 $\oplus$ 1=1，0 $\oplus$ 0=0，1 $\oplus$ 1=0

f 的功能：将 32 比特的数据经过**选择扩展运算 E**、**密钥加密运算**、**选择压缩运算 S**、**置换运算 P** 转换为 32 比特的输出。

S 盒代替运算

将密钥加密运算后的 **48 比特**数据从左至右分成 8 组，每组为 **6 比特**，并行送入 8 个 S 盒后转换成 **32 比特**输出。

每个 S 盒的输入为 6 比特，输出为 4 比特。（8 盒 $\times$ 4 位 = 32 位）

子密钥生成

64 比特初始密钥 k 经过换位函数 **PC-1**，将位置号为 8,16,24,32,40,48,56 和 64 的 8 位奇偶位去掉并换位；换位后的数据分为 2 组，经过循环左移位 LS_i 和换位函数 **PC-2** 变换后得到每次迭代加密用的子密钥 k_i 。

安全性分析

- **穷举攻击**：进行 $2^{56} \approx 10^{17}$ 次穷举
 - **线性密码分析**（Mitsuru Matsui 提出）：采用近似值逼近分组密码的操作，约 2^{43} 次
 - **差分密码分析**（Biham 和 Shamir 提出）：通过分析特定明文差分对密文差分的影响来获得可能密钥，约 2^{47} 次
- DES 的安全性完全依赖于密钥，而 56 bit 的密钥太短 $\Rightarrow$ 出现三重 DES、分组反馈链接式 DES、密码反馈模式 DES 等改进算法。

课件给了 IP 与 IP^{-1} 的置换表：

IP = 1 2 3 4 ... 63 64
 40 8 48 16 ... 57 19

IP^{-1} = 1 2 3 4 ... 63 64
 58 50 42 34 ... 15 7

2.1.2 IDEA 算法

来历与参数

来学嘉和 James Massey 于 1990 年公布，1992 年更名为 IDEA（国际数据加密算法，International Data Encryption Algorithm）。

- 分组长度为 64 位，分 4 个 16 位子分组
- 密钥长度 128 位
- 使用异或、模 2^{16} 加、模 $2^{16}+1$ 乘三个混合运算，在 16 位子分组上进行
- 共进行 8 轮操作，每轮与 6 个 16 位子密钥异或、相加、相乘；在每轮之间，第 2 个和第 3 个子分组交换
- 最终由一个输出变换
- 强化了抗差分分析能力，应用于 PGP 中
- 软件实现 IDEA 比 DES 快两倍
- 乘加结构（MA）保证了扩散的特点

子密钥生成：128 位的 k 分组： $k_1, k_2, \dots, k_8$ ；然后进行左环移 25 位。

DES vs IDEA 对比

	DES	IDEA
颁布	1977 年，数据加密标准	1990 年，国际数据加密标准
分组长度	64 位	64 位
密钥长度	64 位（实际 56 位）	128 位
子分组	32 位	16 位
迭代轮数	16 轮	8 轮
子密钥数	16 个	52 个
运算	S-Box、 $\oplus$	$\oplus$ 、+、 $\times$

2.1.3 AES 算法

来历与参数

发布：1997 年 NIST 征集，**2002 年正式公布**。由 Vincent Rijmen 和 Joan Daemen 设计，原名 Rijndael 算法。

特点：安全，易实现，算法灵活与简便。

- **128 位分组**，密钥长度 l 为 128 / 192 / 256 位
- **4x4 字节矩阵**， $r = (l/32 + 6)$ 轮迭代， $r+1$ 个子密钥

密钥长度	4×4 / 128	4×6 / 192	4×8 / 256
分组长度	4×4 / 128	4×4 / 128	4×4 / 128
轮数	10	12	14
轮密钥长	4×4 / 128	4×4 / 128	4×4 / 128
密钥总数	11	13	15

四个核心步骤

128 比特的明文分组被分成 **4×4 的字节矩阵**（称为状态 State）。

- 字节替换 SubBytes**：非线性置换，独立作用于状态中的每一个字节。（逆运算 InvSubBytes 用逆 S 盒）
- 行移位 ShiftRows**：字节循环移位运算，第 1 行~第 4 行字节分别向左移动 0~3 列。（逆运算 InvShiftRows 向右移动 0~3 列）
- 列混合 MixColumns**：由一个线性变换对状态的每一列进行变换。（逆运算 InvMixColumns）

$$\begin{bmatrix} 0E & 0B & 0D & 09 \end{bmatrix} \begin{bmatrix} a0 \end{bmatrix} \quad \begin{bmatrix} b0 \end{bmatrix} \quad \begin{bmatrix} 1 & 0 & 0 & 0 \end{bmatrix} \\ \begin{bmatrix} 09 & 0E & 0B & 0D \end{bmatrix} \begin{bmatrix} a1 \end{bmatrix} = \begin{bmatrix} b1 \end{bmatrix} \quad \begin{bmatrix} 0 & 1 & 0 & 0 \end{bmatrix} \\ \begin{bmatrix} 0D & 09 & 0E & 0B \end{bmatrix} \begin{bmatrix} a2 \end{bmatrix} \quad \begin{bmatrix} b2 \end{bmatrix} \quad \begin{bmatrix} 0 & 0 & 1 & 0 \end{bmatrix} \\ \begin{bmatrix} 0B & 0D & 09 & 0E \end{bmatrix} \begin{bmatrix} a3 \end{bmatrix} \quad \begin{bmatrix} b3 \end{bmatrix} \quad \begin{bmatrix} 0 & 0 & 0 & 1 \end{bmatrix}$$

- 轮密钥加 AddRoundKey**：与相应的密钥进行异或。

算法步骤小结：字节替换（非线性置换 S 盒，混淆）→ 行移位（字节矩阵按行移位 0~3 列）→ 列混合（线性矩阵模乘运算，扩散）→ 轮密钥加（混入密钥信息）。

AES 中的多项式运算 ($GF(2^8)$)

不可约多项式：设 $f(x)$ 是整环 R 上的非常数多项式，如果除了 1 和 $f(x)$ 外没有其他因式，则 $f(x)$ 叫做不可约多项式，否则称为可约多项式或者合式。

举例 1：多项式 x^2+1 在 $Z(x)$ 中不可约，在有限域 $F_2[x]$ （通常记做 $GF(2^n)$ ）中可约，因为在 $F_2[x]$ 中 $x^2+1 = (x+1)^2$ 。

举例 2： $GF(2^2)$ 中不可约多项式有 $x, x+1$ ； $GF(2^3)$ 中二次不可约多项式有 x^2+x+1 ； $GF(2^4)$ 中三次不可约多项式有 x^3+x+1, x^3+x^2+1 。

AES 中以字节为单位进行模 $x^8+x^4+x^3+x+1$ 运算。

AES 多项式运算规则

- $GF(2^9)$ 中的不可约多项式 $x^8+x^4+x^3+x+1$ （二进制 100011011）
- 分析： $GF(2^9)$ 中次数 ≤ 4 的不可约多项式有 $x, x+1, x^2+x+1, x^3+x+1, x^3+x^2+1, x^4+x+1, x^4+x^3+1$ ，都不能整除 $x^8+x^4+x^3+x+1$ （即它不可约）
- 有限域 $F_2[x]$ 中，加法和减法相当于按位异或。
例： $x^3+x^2+x - (x^3+1) = x^2+x+1 \iff 1110 \oplus 1001 = 0111$
- 有限域 $F_2[x]$ 中，乘 2 相当于按位左移 1 位。
例： $(x^3+1) \cdot x = x^4+x \iff (1001) \times (10) = 10010$

课件算例

$02 \times d4 + 03 \times bf + 01 \times 5d + 01 \times 30 \rightarrow 04$

$$d4 = 11010100 \rightarrow f(x) = x^7+x^6+x^4+x^2$$

$$\begin{aligned} 02 \times d4 &= x \cdot f(x) = x^8+x^7+x^5+x^3 \pmod{x^8+x^4+x^3+x+1} \\ &= x^8+x^7+x^5+x^3 - (x^8+x^4+x^3+x+1) \\ &= x^7+x^5+x^4+x+1 = 10110011 \rightarrow b3 \quad \text{【类比：} 15 \pmod{11} \text{】} \end{aligned}$$

$$\begin{aligned} 04 \times d4 &= x^2 \cdot f(x) = x^9+x^8+x^6+x^4 \pmod{x^8+x^4+x^3+x+1} \\ &= x^9+x^8+x^6+x^4 - x \cdot (x^8+x^4+x^3+x+1) \end{aligned}$$

$$= x^8 + x^6 + x^5 + x^2 + x - (x^8 + x^4 + x^3 + x + 1)$$

$$= x^6 + x^5 + x^4 + x^3 + x^2 + 1 = 01111101 \rightarrow 7d \quad \text{【类比: } 15 \bmod 7 \text{】}$$

AES 参考代码

```
Rijndael(State, CipherKey){
    KeyExpansion(CipherKey, W[1,...,Nb*(Nr+1)]);
    AddRoundKey(State, W[0,...,Nb-1]);
    for (i=1; i<Nr; i++) {
        SubBytes(State);
        ShiftRows(State);
        MixColumns(State);
        AddRoundKey(State, W[i*Nb,...,(i+1)*Nb-1]);
    } // 中间 Nr-1 轮
    SubBytes(State); ShiftRows(State); // 末轮
    AddRoundKey(State, W[Nr*Nb,...,(Nr+1)*Nb-1]);
}
```

对称密码算法小结

优缺点

优点：效率高，算法简单，系统开销小，适合加密大量数据。

缺点：需要以安全方式进行密钥交换，**密钥管理复杂**。

算法	分组长度	密钥长度	迭代次数	数学运算
DES	64	56 (原 64)	16	S-Box、 $\oplus$
IDEA	64	128	8	$\oplus$ 、+、 $\times$
AES	128	128	10,12,14	S-Box、 $\oplus$ 、 $\times$
BlowFish*	—	最大 448	16	S-Box、 $\oplus$ 、+
RC6*	—	最大 2048	<255	$\oplus$ 、+、-、 $\times$

2.2 序列密码算法

分组密码 vs 序列密码

- **分组密码：**将明文分组进行加密
- **序列密码：**将明文按字节或位进行加密

	序列密码	分组密码
代表	RC4、LFSR	DES、AES
特点	简单快速	扩散性好

序列密码算法原理：密钥流 $\oplus$ 明文 = 密文。

密钥流的生成方式：RC4 $\rightarrow$ 字节流；同余发生器 $\rightarrow$ 数据流；移位寄存器 $\rightarrow$ 位流。

为什么序列密码的密钥流不能重复使用

原始明文：P1 P2 P3 P4	新明文：P1 P' P2 P3 P4
原始密钥：k1 k2 k3 k4	新密钥：k1 k2 k3 k4 k5
原始密文：C1 C2 C3 C4	新密文：C1 C2' C3' C4' C5'

由于黑客知道 P'，他可以根据：

$k_2 = P' \oplus C_2'$ 得到 k_2 ， $P_2 = C_2 \oplus k_2$ ； $k_3 = P_2 \oplus C_3'$ 得到 k_3 ， $P_3 = C_3 \oplus k_3$

以此类推，从而确定整个明文。

2.2.1 RC4 算法

背景与特点

发布：Ron Rivest 在 1987 年设计（也是 RSA 中的 R：Ron Rivest、Adi Shamir、Len Adleman）。

- 简单，速度快，10 倍于 DES 的速度
- 面向字节的流密码，密钥独立于明文
- 应用广泛，很多 HTTPS 网站使用 RC4
- 有一个 256 字节的 S 盒： $S_0, S_1, \dots, S_{255}$ ； S_i 的值是 0 到 255 之间不重复的值
- 密钥长度可变

S 盒初始化

$S_0=0, S_1=1, \dots, S_{255}=255$

然后用初始密钥 key 填充另一个 256 字节的数组 K，
不断重复密钥直至填充整个数组

$j = 0$

对于 $i = 0$ 至 255

$j = (j + S_i + K_i) \bmod 256$

交换 S_i 和 S_j

最后得到用密钥初始化的 S 盒

密钥流生成（核心）

1 $i = j = 0$

2 $i = (i + 1) \bmod 256$

$j = (j + S_i) \bmod 256$

交换 S_i 和 S_j

$t = (S_i + S_j) \bmod 256$

$k = S_t$

k 即为密钥流，为产生的随机字节

加密：明文 $\oplus$ 密钥流 $\rightarrow$ 密文；密文 $\oplus$ 密钥流 $\rightarrow$ 明文。

课件作业：编程实现 RC4 算法密钥流的生成；查阅资料了解 RC4 可能的替代算法。

2.2.2 线性同余发生器

公式

$$s_{n+1} = (a \cdot s_n + b) \bmod m$$

其中 $0 < a < m$ ， $0 \leq b < m$ ，初值 s_0 称为种子。

a、b、m 的取值是产生高质量随机数的关键。

课件算例

取 $a=2, b=7, m=17$ ，则 $s_{n+1} = (2s_n + 7) \bmod 17$ ，取种子 $s_0=1$ ：

$$s_1 = (2 \times 1 + 7) \% 17 = 9$$

$$s_2 = (2 \times 9 + 7) \% 17 = 25 \% 17 = 8$$

$$s_3 = (2 \times 8 + 7) \% 17 = 23 \% 17 = 6$$

$$\begin{aligned}s_4 &= (2 \times 6 + 7) \% 17 = 19 \% 17 = 2 \\s_5 &= (2 \times 2 + 7) \% 17 = 11 \\s_6 &= (2 \times 11 + 7) \% 17 = 29 \% 17 = 12 \\s_7 &= (2 \times 12 + 7) \% 17 = 31 \% 17 = 14 \\s_8 &= (2 \times 14 + 7) \% 17 = 35 \% 17 = 1 \leftarrow \text{回到 } 1\end{aligned}$$

序列: 1, 9, 8, 6, 2, 11, 12, 14, 1, 9, 8, 6, 2, 11, 12, 14, 1, 9, ...

另一组 ($a=7, b=1$, 种子 $s_0=14$):

1, 8, 6, 9, 13, 7, 16, 11, 10, 3, 5, 2, 15, 4, 12, 0, 1, 8, ...

周期

线性同余发生器 $s_{n+1} = a \cdot s_n \pmod{p}$ 的周期为 a 在乘法群 Z_p^* 中的阶 l 。

- p 为素数时, $l \mid p-1$; 当 a 为模 p 的本原根时, $l = p-1$
- 种子 s_0 不等于坏种子

$$\text{坏种子: } s_{\text{bad}} \equiv -(a-1)^{-1} \cdot b \pmod{p}$$

为什么是坏种子: 若 $s = (as+b) \pmod{p}$, 则 $s(1-a) = b \implies s = b/(1-a) = -(a-1)^{-1}b$ 。种子落在此处, 序列永远卡死, 周期退化为 1。

课件举例 1: 求 $s_{n+1} = 7s_n + 2 \pmod{13}$ 的坏种子 s_{bad} 。

$a=7, b=2, p=13 \implies s_{\text{bad}} = -(6)^{-1} \times 2 \pmod{13}$ 。 $6^{-1} = 11$ (因 $13 - 2 \times 6 = 1 \implies 6^{-1} = -2 = 11$) $\implies s_{\text{bad}} = -11 \times 2 = -22 \equiv 4 \pmod{13}$ 。

验算: $(7 \times 4 + 2) = 30, 30 \pmod{13} = 4 \checkmark$ 果然卡死。

设计结论

结论: 线性同余发生器 $s_{n+1} = a \cdot s_n + b \pmod{m}$ 的选择是 m 为大素数, a 为模 m 的本原根, s_0 不为坏种子, 以保证最大周期。

算法不具备随机性。

课件作业: 计算同余发生器 $x_{n+1} = 6x_n + 9 \pmod{11}$ 的坏种子。 ($a=6, b=9, p=11: 5^{-1} \pmod{11} = 9 \implies s_{\text{bad}} = -9 \times 9 = -81 \equiv 7 \pmod{11}$; 验算 $6 \times 7 + 9 = 51, 51 \pmod{11} = 7 \checkmark$)

2.2.3 LFSR 线性反馈移位寄存器

组成与公式

基于移位寄存器的序列密码被广泛应用于军事密码学中。由两部分组成: 移位寄存器和反馈函数。

最简单的是 LFSR。

$$b_{n+1} = c_n b_n + c_{n-1} b_{n-1} + \dots + c_2 b_2 + c_1 b_1, \quad c_i \in \{0, 1\}$$

参与异或的位置称为**抽头**, 一串 c_i 称为**抽头序列**。

课件例: 4 位 LFSR, 抽头第 1、4 位

$b_{n+1} = b_n + b_{n-3}$, 初始值 1111, 求输出序列。

寄存器状态 (每行左移、右侧补入新 bit):

1111 $\rightarrow$ 0111 $\rightarrow$ 1011 $\rightarrow$ 0101 $\rightarrow$ 1010 $\rightarrow$ 1101 $\rightarrow$ 0110 $\rightarrow$...

输出序列: 111101011001000...

取种子 $(b_4 b_3 b_2 b_1) = (0011)$, 则输出序列为: 1100100011110101100...

周期

n 位 LFSR 中，具有特殊抽头序列的才有最大周期 2^n-1 ，在该类抽头序列加上常数 1 形成的多项式必须是模 2 的本原多项式。

为什么是 2^n-1 而不是 2^n ：全 0 状态是死循环（0000 → 新 bit = $0 \oplus 0 = 0 \rightarrow 0000$ ），所以能用的状态少 1 个。

课件举例 1：取种子 (0011)，分析抽头序列 (4,1,0) 和 (4,2,1,0) 的周期。

x^4+x+1
1100100011110101100...
⇒ 周期 15 (2^4-1)

x^4+x^2+x+1
有因子 $x^3+x^2+1, x+1$
110001110001...
⇒ 可分解，周期只有 6

课件举例 2：一个 32 位 LFSR 的抽头序列 (32,7,5,3,2,1,0) 和 (32,7,6,2,0) 具有最大周期 $2^{32}-1$ 。

三种密钥流生成方式对比

	RC4	线性同余发生器	LFSR
产出	字节流	数据流	位流
随机性来源	256 字节 S 盒洗牌	线性递推公式	移位 + 异或反馈
内部状态	256 字节排列	1 个数	n 个 bit
最大周期	极长	$p-1$ (a 为本原根)	2^n-1 (本原多项式)
"钥匙"	初始密钥	选 a	选抽头序列
抗分析	非线性，较难	纯线性，易解	纯线性，易解

2.3 数字摘要算法

散列函数 (hash / 哈希 / 杂凑函数)

是一个公开的函数，它将任意长的信息映射成一个固定长度的信息（数字摘要 / 散列值 / 数字指纹）。

性质	含义	别称
公开性	算法公开	—
单向性	由 $x \rightarrow h(x)$ 易，但 $h(x) \rightarrow x$ 难	—
扩散性	输入预映射值单个位的改变，将引起输出散列值中一半的位发生改变	雪崩效应
抗碰撞性	当 $x \neq y$ 时，有 $h(x) \neq h(y)$ (即 $h(x)=h(y)$ 的概率很小)	无碰撞性

作用

- 保证消息的完整性
- 用在数字签名中，提高签名效率

消息鉴别码 MAC：与密钥有关的单向散列函数称为消息鉴别码。

2.3.1 MD5 算法

概述

消息摘要 (Message Digest) 或杂凑函数，由 Ronald Rivest 1992 年开发。

输入消息 x，输出为 128 位的消息摘要值。

算法包括 5 个步骤：

- 附加填充比特：在消息 x 后填充串 1000...，使得消息的长度满足 $|x| \equiv 448 \bmod 512$ ；填充比特总是有效，填充数为 1~512
- 附加长度：将原始消息的长度（64 比特）附加到填充后的消息后面，填充后消息的总长为 512 比特的倍数；设消息最终分成 n 个 512 的消息块 $V_1 \sim V_n$ ，每个消息块 V 分成 16 个 32 比特的子块 $X[0] \sim X[15]$
- 初始化或填充 MD 缓冲区：4 个 32 比特的寄存器 A, B, C, D。对 V_1 操作时初始值：
 - $a = A = 0x01234567$

- $b = B = 0x89abcdef$
- $c = C = 0xfedcba98$
- $d = D = 0x76543210$

否则为前一消息块的迭代输出

4. 按每块 16×32 (512 比特) 对数据进行处理: 主循环有 4 轮, 每轮进行 16 次操作。

轮数	$X[k]$ 的 k 值
1	0, 1, 2, 3, ..., 15
2	1, 6, 11, 0, ..., 12
3	5, 8, 11, 14, ..., 2
4	0, 7, 14, 5, ..., 9

说明: $X[k]$ 为消息块 X 的第 k 个子分组 (0~15); $T[i]$ 为 $2^{32} \times \text{abs}(\sin(i))$ 的整数部分, i 为弧度。

4 轮操作: $b = b + (a + (g(b, c, d) + X[k] + T[i]) \lll s)$, g 在每轮中分别为 F 、 G 、 H 、 I 逻辑函数。

第 1 轮: $b = b + (a + (F(b, c, d) + X[k] + T[i]) \lll s)$

5. 输出: $a = (A+a) \bmod 2^{32}$; $b = (B+b) \bmod 2^{32}$; $c = (C+c) \bmod 2^{32}$; $d = (D+d) \bmod 2^{32}$

2.3.2 SHA 算法

概述

SHA (Secure Hash Algorithm) 由 NIST 开发并于 1993 年采纳为标准, 其基础是 MD4, 因此其设计与 MD5 类似。

算法	输入值	输出值 (位)
MD5	任意	128
SHA1	2^{64}	160
SHA224	2^{64}	224
SHA256	2^{64}	256
SHA384	2^{128}	384
SHA512	2^{128}	512

SHA 算法流程 (5 步): 附加填充比特 → 附加长度 → 初始化缓冲区 → 80 次迭代 → 输出。

以分组 512 比特为单位对消息进行 4 轮压缩处理, 每轮处理过程对缓冲区进行 20 次迭代运算。

32 比特寄存器初始值:

$A = 0x67452301$ $B = 0xefcdab89$ $C = 0x98badcfe$ $D = 0x10325476$ $E = 0xc3d2e1f0$

SHA 流程分析 (课件原例)

step1 附加填充比特: 在消息 x 后填充串 1000..., 使得 $|x| \equiv 448 \bmod 512$ 。

例: $x = \text{"abc"} \Rightarrow$ 填充后: 01100001 01100010 01100011 1000...0000 (a 、 b 、 c 后补 423 个 0)

step2 附加长度: 将原始消息长度附加到填充后消息后面的 64 位中。

0110000101100010011000111000...0000 000...11000 (64 位)

step3 初始化或填充缓冲区: $A=0x67452301$ $B=0xefcdab89$ $C=0x98badcfe$ $D=0x10325476$ $E=0xc3d2e1f0$

step4 80 次迭代:

$$\begin{aligned}
 f_t = f(t, B, C, D) = & \begin{aligned} & (B \wedge C) \vee (B \wedge D) & 0 \leq t < 20 \\ & B \oplus C \oplus D & 20 \leq t < 40 \\ & (B \wedge C) \vee (B \wedge D) \vee (C \wedge D) & 40 \leq t < 60 \\ & B \oplus C \oplus D & 60 \leq t < 80 \end{aligned} \\
 X[t] & \quad 0 \leq t < 16 \\
 W_t = & \begin{aligned} & (W_{t-16} \oplus W_{t-14} \oplus W_{t-8} \oplus W_{t-3}) \lll 1 & 16 \leq t < 80 \\ & 2^{30} \times \sqrt{2} = 0x5a827999 & 0 \leq t < 20 \end{aligned}
 \end{aligned}$$

$$\begin{aligned} K_t = 2^{30 \times \sqrt{3}} &= 0x6ed9eba1 & 20 \leq t < 40 \\ 2^{30 \times \sqrt{5}} &= 0x8f1bbcdc & 30 \leq t < 60 \\ 2^{30 \times \sqrt{10}} &= 0xca62c1d6 & 60 \leq t < 80 \end{aligned}$$

S_k : 循环左移 k 位

step5 输出: 5 个寄存器 A,B,C,D,E 分别加上初始化值。

$A = A + 0x67452301$ $B = B + 0xefcdab89$ $C = C + 0x98badcfe$ $D = D + 0x10325476$ $E = E + 0xc3d2e1f0$

2.3.3 小结

SHA 算法流程一句话

512 位分组, 160 位输出, 5 个 32 位寄存器, 4 轮 80 次迭代, 每轮使用 1 个逻辑函数和 1 个加法常数。

安全性现状

王小云教授于 2004 年破译 MD5 和 SHA-1, 目前普遍采用 SHA256。

实际运用中数字摘要通常与其他密码技术混合使用, 伪造有意义的电子签名需要更尖端的技术。

2.4 公钥密码算法

概述

要点

- 公钥密码学概念 —— 1976 年 Diffie 和 Hellman 提出
- 特点: 加密密钥公开, 密钥管理简单
- 代表性算法: RSA、ElGamal、ECC 等
- 安全性: 基于数学难题, 运算速度较慢

算法	依赖的数学难题
Knapsack	背包问题
RSA	因子分解难题
ElGamal	离散对数难题
ECC	椭圆曲线点群上的离散对数难题

所需数学知识: 欧拉定理、离散对数、同余方程、中国剩余定理、二次符号、素数生成……

2.4.1 数学知识 (公钥密码专用)

离散对数

定义

对正整数 n , 若 $g^t \equiv x \pmod{n}$, 则 t 称为 x 的以 g 为底模 n 的离散对数, 记为 $\log_g x \pmod{\text{ord}_n(g)}$ 。

举例: $3^2 \equiv 9 \pmod{13} \implies \log_3 9 \equiv 2 \pmod{3}$; 因为 $3^1=3, 3^2=9, 3^3=1$ 。

离散对数问题 DLP: 给定素数 p 、 Z_p^* 的生成元 g 以及整数 $x \in Z_p^*$, 求解 t 满足 $g^t \equiv x \pmod{n}$ 。

- 易: $g, t, n \rightarrow x$
- 难: $g, x, n \rightarrow t$ (当 n 很大时, 是一个难题)

应用: 很多密码技术的安全性建立在离散对数难题上, 如 ElGamal、ECC 等。

算法 1: Baby-step Giant-step

p 为素数, 在 Z_p^* 中计算 $\log_b a$, m 是不超过 $\sqrt{p-1}$ 的最大正整数, 步骤:

1. 计算 $(0, b^{0 \% p}), (1, b^{1 \% p}), \dots, (m-1, b^{m-1 \% p})$
2. 按照第二分量的大小排序

3. 计算 $c = b^{-m} \bmod p$; 初始化 $x = a$;
 对 $0 \leq i < m$, 若 x 是表 $b^0, b^1, \dots, b^{m-1}$ 中的 b^j , 则 $\log_b a = mi + j$;
 否则, $x = x \cdot c \bmod p$, $i = i+1$

原理: $b^{mi+j} = a \implies b^j = a \cdot (b^{-m})^i$

举例: 在 $\mathbb{Z}_{227}^*$ 中计算 $\log_2 11$

算法 2: Pollard's Rho 方法

定义函数 F :

$$F(x, m, n) = \begin{cases} (a \cdot x, m+1, n) & \text{若 } x_i \in s_1 \\ (x^2, 2m, 2n) & \text{若 } x_i \in s_2 \\ (b \cdot x, m, n+1) & \text{若 } x_i \in s_3 \end{cases}$$

步骤:

- 选取子集 s_1, s_2, s_3
- 初始化 $(x, m_x, n_x, y, m_y, n_y) = (1, 0, 0, 1, 0, 0)$
- 重复步骤: $(x, m_x, n_x) = F(x, m_x, n_x)$; $(y, m_y, n_y) = F(y, m_y, n_y)$
 直到 $x = y$, 则 $\log_b a = (m_y - m_x)^{-1} \cdot (n_x - n_y) \bmod \text{ord}_p(b)$

(由 $a^{m_x} b^{n_x} = a^{m_y} b^{n_y}$ 推出)

举例: 在 $\mathbb{Z}_{59}^*$ 中计算 $\log_4 9$

一次同余式

定义与定理

定义: 设整数 $m > 0$, $f(x)$ 为多项式且 $f(x) = a_n x^n + \dots + a_1 x + a_0$, a_i 为整数, 则 $f(x) \equiv 0 \pmod{m}$ 叫做模 m 同余式; 若 $a_n \neq 0$, 则 n 叫做 $f(x)$ 的次数。

定理: 设 $m \mid a$, $d = (a, m)$, 则模 m 的一次同余式 $ax \equiv y \pmod{m}$ 有解的充要条件是 $d \mid y$; 解的个数是 d 。

课件举例

- $6x \equiv 2 \pmod{9}$: 因为 $(6, 9) = 3$ 不整除 $2 \implies$ **无解**
- $3x \equiv 2 \pmod{5}$: 因为 $(3, 5) = 1 \mid 2 \implies$ 有解, 解为 $x \equiv 4 \pmod{5}$
- $3x \equiv 3 \pmod{6}$: 因为 $(3, 6) = 3 \mid 3 \implies$ 有解, 解为 $x \equiv 1, 3, 5 \pmod{6}$
- $19x \equiv 38 \pmod{57}$: 因为 $(19, 57) = 19 \mid 38 \implies$ **有 19 个解**, $x \equiv 2 \pmod{3} \pmod{57}$

中国剩余定理 CRT

定理与公式

又称**孙子定理**: 最早见于《孙子算经》中的"物不知数"问题: 今有物不知其数, 三三数之剩二, 五五数之剩三, 七七数之剩二。问物几何?

已知 $m_1, m_2, \dots, m_n$ 两两互素, 则同余方程组

$$\begin{aligned} x &\equiv a_1 \pmod{m_1} \\ x &\equiv a_2 \pmod{m_2} \\ &\vdots \\ x &\equiv a_n \pmod{m_n} \end{aligned}$$

令 $M_n = m_1 \times m_2 \times \dots \times m_n$, $W_i = M_n / m_i$, $T_i = W_i^{-1} \bmod m_i$

$$x \equiv a_1 T_1 W_1 + \dots + a_n T_n W_n \pmod{M_n}$$

课件例: $x \equiv 2(3), x \equiv 3(5), x \equiv 2(7)$

$$\begin{aligned} x &\equiv 2 \times 35 \times (35^{-1} \bmod 3) + 3 \times 21 \times (21^{-1} \bmod 5) + 2 \times 15 \times (15^{-1} \bmod 7) \\ &\equiv 2 \times 35 \times 2 + 3 \times 21 \times 1 + 2 \times 15 \times 1 \pmod{105} \equiv 23 \pmod{105} \end{aligned}$$

两式情形： $x \equiv a_1(m_1), x \equiv a_2(m_2)$ 有解

$$x \equiv a_1 \cdot m_2 \cdot (m_2^{-1} \bmod m_1) + a_2 \cdot m_1 \cdot (m_1^{-1} \bmod m_2)$$

课件例题

(1) $x \equiv 2 \pmod{11}, x \equiv 7 \pmod{13}$

$$\begin{aligned} x &\equiv 2 \times 13 \times (13^{-1} \bmod 11) + 7 \times 11 \times (11^{-1} \bmod 13) \pmod{143} \\ &\equiv 2 \times 13 \times (-5) + 7 \times 11 \times 6 \pmod{143} \equiv 46 \pmod{143} \end{aligned}$$

(2) $x \equiv 2(11), x \equiv 7(13), x \equiv 3(7)$

应用：简化复杂运算——例如计算 $47^{317} \bmod 713$ 。

解：因为 $713 = 23 \times 31$ ，令 $x = 47^{317}$ ，则计算 $x \bmod 713$ 等价于求解同余方程：

$$x \equiv 47^{317} \pmod{23} \equiv 13^{17} \pmod{23} \equiv 6 \pmod{23}$$

$$x \equiv 47^{317} \pmod{31} \equiv 8^{17} \pmod{31} \equiv 2 \pmod{31}$$

$$\Rightarrow x \equiv 374 \pmod{713}$$

二次同余式与平方剩余

定义

二次同余式：一般形式 $ay^2 + by + c \equiv 0 \pmod{n}$ ，其中 n 为正整数且 $a \neq 0$ ，可简化为：

$$(2ay+b)^2 \equiv b^2 - 4ac \pmod{n}, \text{ 等价于 } x^2 \equiv d \pmod{n}, (d, n) = 1$$

平方剩余：整数 $n > 0, (a, n) = 1$ ，若 $x^2 \equiv a \pmod{n}$ 有解，则 a 称为模 n 的**平方剩余**（二次剩余）；否则 a 称为模 n 的**平方非剩余**。

举例：1, 2, 4 是模 7 的平方剩余；-1, 1, 2, 4, 8, 9, 13, 15 是模 17 的平方剩余。

欧拉判别准则

设 p 为奇素数， $(a, p) = 1$ ，则

- $a^{(p-1)/2} \equiv 1 \pmod{p}$ 时， a 是模 p 的平方剩余
- $a^{(p-1)/2} \equiv -1 \pmod{p}$ 时， a 是模 p 的平方非剩余

举例：14 是模 101 的平方剩余吗？

求模 n 的二次根

定理：设 p 是满足 $p \equiv 3 \pmod{4}$ 的素数，如果同余式 $x^2 \equiv a \pmod{p}$ 有解，则其解是

$$x \equiv a^{(p+1)/4} \pmod{p}$$

证明：由于 y 是模 p 的平方数，设 $y = x^2$ ，则有 $(y^{(p+1)/4})^2 = y^{(p+1)/2} = x^{p+1} = x^p \cdot x = x \cdot x = y$

举例：求 6 模 19 的二次根

$$a^{(p-1)/2} \bmod p = 6^9 \bmod 19 = 1$$

$$x \equiv \pm a^{(p+1)/4} \bmod p = \pm 6^5 \bmod 19 = \pm 5 \bmod 19 \checkmark$$

模是合数的情形

模是合数 $n = p_1 p_2 \dots p_k$ ：拆成多个方程，用中国剩余定理合并。

课件的例子：解 $x^2 \equiv -1 \pmod{13 \times 17 \times 29}$

$$x^2 \equiv -1 \pmod{13} \Rightarrow x \equiv \pm 5 \pmod{13}$$

$$x^2 \equiv -1 \pmod{17} \Rightarrow x \equiv \pm 4 \pmod{17}$$

$$x^2 \equiv -1 \pmod{29} \Rightarrow x \equiv \pm 12 \pmod{29}$$

$\Rightarrow$ 8 个解 $(2 \times 2 \times 2)$

模是合数 $n = p^k$ ：用亨泽尔定理。假设 f 是一个多项式， p 为素数，如果 $x = x_1 \bmod p$ 是同余式 $f(x) \equiv 0 \pmod{p}$ 的一个解，且 $(f'(x_1), p) = 1$ ，则同余式 $f(x) \equiv 0 \pmod{p^k}$ 有解 $x = x_k \bmod p^k$ ，其中 x_k 由下面关系式递归得到：

$$x_i \equiv x_{i-1} - f(x_{i-1})f'(x_1)^{-1} \pmod{p^i}, i = 2, \dots, k$$

二次符号（勒让德符号 / 雅可比符号）

定义

$$\begin{aligned}(a/p) &= 0 && \text{若 } p \mid a \\ &= 1 && \text{若 } a \text{ 是模 } p \text{ 的平方剩余} \\ &= -1 && \text{若 } a \text{ 是模 } p \text{ 的平方非剩余}\end{aligned}$$

$$\begin{aligned}(1/7) &= (2/7) = (4/7) = 1 && \text{平方剩余} \\ (3/7) &= (5/7) = (6/7) = -1 && \text{平方非剩余}\end{aligned}$$

说明：如果素数 p 改为正奇数 n ，即为 **Jacobi 符号**。

计算定理与性质

定理：素数 $p > 2$ ，对于任意整数 a 有 $(a/p) \equiv a^{(p-1)/2} \pmod{p}$

性质：素数 $p > 2$ ，对于整数 a 和 b 有

$$\begin{aligned}(1/p) &= (a^2/p) = 1 && (-1/p) = (-1)^{(p-1)/2} && (2/p) = (-1)^{(p^2-1)/8} \\ (ab/p) &= (a/p)(b/p) && (a+p/p) &= (a/p) \\ \text{若 } a &\equiv b \pmod{p}, \text{ 则 } (a/p) &= (b/p)\end{aligned}$$

二次互反定理

如果 p, q 是不同的奇素数，那么

$$(q/p) = (-1)^{(p-1)(q-1)/4} (p/q)$$

课件例题：判断 $x^2 \equiv 19 \pmod{101}$ 是否有解

$$\begin{aligned}(19/101) &= (-1)^{(19-1)(101-1)/4} \times (101/19) \\ &= (6/19) \\ &= (2/19) \times (3/19)\end{aligned}$$

$$\begin{aligned}(2/19) &= (-1)^{(19^2-1)/8} = -1 \\ (3/19) &= (-1)^{(3-1)(19-1)/4} \times (19/3) = -(1/3) = -1\end{aligned}$$

所以 $(19/101) = (2/19) \times (3/19) = 1$ ，方程有解。

课件课内练习：判断 $x^2 \equiv 137 \pmod{227}$ 是否有解？（227 为素数）

分析讨论：判断 $x^2 \equiv 2 \pmod{3599}$ 是否有解？ $3599 = 59 \times 61$ 。

因 $3599 = 59 \times 61$ ，方程等价于 $x^2 \equiv 2 \pmod{59}$ 与 $x^2 \equiv 2 \pmod{61}$ 。

$(2/59) = (-1)^{(59^2-1)/8} = -1$ 无解； $(2/61) = (-1)^{(61^2-1)/8} = -1$ 无解

$\Rightarrow$ 方程 $x^2 \equiv 2 \pmod{3599}$ 无解。

素数生成

要点

素数的产生：采用素性检测法对随机产生的数进行检测。没有发现产生素数的模型或有效公式。

例如：中国古代数学家提出，若 $n \mid 2^n - 2$ ，则 n 为素数。（ $n=3, 5, 7$ 成立）

但是当 $n = 341 = 11 \times 31$ 时不成立（ $2^{340} = (3 \times 341 + 1)^{34} \equiv 1 \pmod{341}$ ）。

素数有无穷个：设 $p_1, \dots, p_n$ 为所有素数，根据 $q = p_1 p_2 \dots p_n + 1$ 可知必然存在一个除 $p_1 \sim p_n$ 之外的新素数。

数值越大，个数越少：[1,100] 25 个；[1000,1100] 16 个；[100000,100100] 6 个；[10000000,10000100] 2 个。

筛法求素数：1914 年 D.N.Lehmer 编制出 10^7 以内的素数表。

特殊形式的素数

梅森 (Mersenne) 素数：形如 $M_n = 2^n - 1$ 的素数 (n 为素数)。

举例： $n = 2(3), 3(7), 5(31), 7(127), 13(8191), 17(131071), 19, 31, 61, 89, 107, 127, 521, 607, 1279, 2203, 2281, 3217, 4253, 4423, 9689, 9941, 11213, 21701, \dots$

但 $M_{11} = 2047 = 23 \times 89$ 不是素数。

费马 (Fermat) 素数：形如 $F_n = 2^{2^n} + 1$ 的素数 (n 为自然数)。

举例： $n = 1(5), 2(17), 3(257), 4(65537), \dots$

但 $F_5 = 4294967297 = 641 \times 6700417$ 不是素数。

素数的判定

两大判别法

威尔逊 (Wilson) 判别法： n 是素数的充要条件是 $(n-1)! \equiv -1 \pmod{n}$

费马 (Fermat) 判别法：如果 n 为素数， $(a, n) = 1$ ，那么 $a^{n-1} \equiv 1 \pmod{n}$

费马伪素数：设 n 是奇数， $(n, a) = 1$ ，如果 $a^{n-1} \equiv 1 \pmod{n}$ ，则整数 n 对于基数 a 是费马伪素数。

举例：整数 $n = 63$ 对于基数 $a = 8$ 是伪素数；整数 $n = 341$ 对于基数 $a = 2$ 是伪素数。

卡迈克尔 (Carmichael) 数

定义：合数 n 称为 Carmichael 数，如果对所有 $a > 0$ 、 $(a, n) = 1$ ，有 $a^{n-1} \equiv 1 \pmod{n}$ 。

举例：561($3 \times 11 \times 17$)、1105($5 \times 13 \times 17$)、1729($7 \times 13 \times 19$)、2465($5 \times 17 \times 29$)、2821($7 \times 13 \times 31$)、6601($7 \times 23 \times 41$)、8911($7 \times 19 \times 67$)、10585($5 \times 29 \times 73$)

分析：如果 $(a, 561) = 1$ ，则 $(a, 3) = (a, 11) = (a, 17) = 1$

$a^2 \equiv 1 \pmod{3}$ 、 $a^{10} \equiv 1 \pmod{11}$ 、 $a^{16} \equiv 1 \pmod{17} \implies a^{560} \equiv 1 \pmod{561}$

意义：这类数能骗过费马判别法，所以需要更强的素性检测。

素性检测四种算法

① Fermat 算法

原理：由费马判别法可知，若 n 不是素数，则 $a^{(n-1)} \pmod{n}$ 的值不恒为 1。

描述：(1) 随机选 $2 \leq a \leq n-2$ ；(2) 计算 $r \equiv a^{n-1} \pmod{n}$ 。 $r \neq 1 \implies n$ 为合数，终止； $r = 1 \implies n$ 大于 50% 概率为素数，继续。

② Lehmann 算法

原理：由欧拉判别准则可知，若 n 不是素数，则 $a^{(n-1)/2} \pmod{n}$ 对于不同的 a 就是随机的。

描述：(1) 随机选 $a < n$ ；(2) 计算 $a^{(n-1)/2} \pmod{n}$ 。 $\neq \pm 1 \implies n$ 为合数，终止； $= \pm 1 \implies n$ 以 50% 概率为素数，继续；(3) 对 a 选 t 次重复，若结果等于 ± 1 并不恒为 1，则 n 以 $(1 - 1/2^t)$ 概率为素数。

③ Solovay-Strassen 算法

原理：由欧拉判别准则和二次符号可知，若 n 是素数，则 $(a/n) \equiv a^{(n-1)/2} \pmod{n}$ 。

描述：(1) 随机选 $a < n$ ；(2) 计算 $j = a^{(n-1)/2}$ 和 $J = (a/n)$ 。 $j \neq J \implies n$ 为合数，终止； $j = J \implies n$ 以 50% 概率为素数，继续。

④ Miller-Rabin 算法

原理：若 n 是素数，则 Z_n^* 中只有 1 的两个平方根，即 1 和 $n-1$ 。

描述：奇整数 n 因式分解为 $n-1 = 2^r \cdot m$ ， m 为奇数。若 $a^m \equiv \pm 1 \pmod{n}$ ，或对某些 $0 < s < r$ 满足 $a^{2^s \cdot m} \equiv -1 \pmod{n}$ ，则 n 是基数为 a 的强伪素数。

Miller-Rabin 详细步骤

1. 因式分解 $n-1 = 2^r \cdot m$ ， m 为奇数

2. ① 随机选 $1 < a < n-1$ ；② 设 $s = 0$ ，计算 $z = a^m \pmod{n}$ ；③ 若 $z = \pm 1$ ，停止， n 以 $\geq 3/4$ 概率是素数

3. ④ 否则继续: $s = s+1$, 计算 $z = z^2 \bmod n$ 。若 $s < r$ 且 $z = -1$, 停止, n 以 $\geq 3/4$ 概率为素数; 若 $z = 1$, n 为合数, 终止测试
4. ⑤ 若 $s = r$, 且没有一个 z 等于 -1 , 则 n 为合数
5. 重复 k 次, n 以 $1 - (1/4)^k$ 概率为素数

知识点巩固 (课件原题, 第 2 章数学部分)

1. 用穷举法计算以 3 为底模 25 的 4 的离散对数
2. 利用 Baby-step 算法在 Z_{29}^* 中求 $\log_2 7$
3. 求一次同余式 $8x \equiv 4 \pmod{12}$ 的所有解
4. 求 2 模 47 的平方根
5. 求一个整数 x , 满足同余方程 $2x \equiv 1 \pmod{3}$, $3x \equiv 2 \pmod{5}$
6. 判断 $5x^2 \equiv 3 \pmod{77}$ 是否有解, 有则求出所有解
7. 利用二次符号判断 2 是否为模 97 的一个平方
8. 判断方程 $x^2 \equiv 83 \pmod{107}$ 是否有解
9. 判定 3 是否为 2009 的一个平方

2.4.2 RSA 算法

背景

发布: 1977 年发明, 1978 年正式公布。Ron Rivest、Adi Shamir、Len Adleman, 2002 年图灵奖。

应用: 最广泛, 既可加密, 又可签名。

数学基础: Euler (欧拉) 定理、因子分解难题。

加解密过程

【密钥生成】

选择大素数 p 和 q , 计算 $n = pq$ 以及 $\varphi(n) = (p-1)(q-1)$
 选择随机数 $e < \varphi(n)$, $\gcd(e, \varphi(n)) = 1$, 计算 $d = e^{-1} \bmod \varphi(n)$
 公钥: (e, n) 私钥: d 说明: $p, q, \varphi(n)$ 注意保密

【加密】加密消息 m : $c = m^e \bmod n$

【解密】解密密文 c : $m = c^d \bmod n$

课件算例: $p=3, q=5 \Rightarrow n=15, \varphi(n)=8$; $e=3 \Rightarrow d=3$ ($3 \times 3 = 9 \equiv 1 \pmod{8}$); $m=7$

加密: $c = 7^3 \bmod 15 = 13$; 解密: $m = 13^3 \bmod 15 = 7 \checkmark$

可解性分析 (为什么能用私钥还原)

由 **欧拉定理**: 如果 $\gcd(x, n) = 1$, 则有 $x^{\varphi(n)} \equiv 1 \pmod{n}$

$$\begin{aligned} c &= m^e & ed &= 1 + k\varphi(n) \\ c^d \bmod n &\rightarrow m^{(ed)} \bmod n &= m^{(1+k\varphi(n))} \bmod n \\ & &= m \times m^{(k\varphi(n))} \bmod n \\ & &= m \times 1 = m & (m^{\varphi(n)} \equiv 1 \pmod{n}) \end{aligned}$$

思考: 若 m 和 n 不互素?

安全性分析

为什么由 RSA 公钥很难得到私钥?

公钥 e 与私钥 d 的关系: $d = e^{-1} \bmod \varphi(n)$, $n = pq$, $\varphi(n) = (p-1)(q-1)$

已知 e 和 n , 要得到 d , 需知道 $\varphi(n)$; 要得到 $\varphi(n)$, 则需要知道 n 的因子分解。当 n 很大时, 这是一个难题。

从安全性考虑, 要求 n 至少 1024 比特。

针对因子分解的攻击

① Pollard's Rho 算法

给定整数 n , 设初始值 $x = 2, y = x^2 + 1$:

1. 计算 $g = (x - y, n)$
2. 若 $1 < g < n$, 则停止, g 为 n 的一个因子
3. 若 $g = 1, x^2 + 1 \rightarrow x, (y^2 + 1)^2 + 1 \% n \rightarrow y$, 返回 (1)
4. 若 $g = n$, 失败, 需要对算法重新初始化

课件例 ($n=1133$): (1) 选 $x=2, y=5$; (2) $g = \gcd(3, 1133) = 1$; (3) $x = 5, y = (5^2 + 1)^2 + 1 \% 1133 = 677$; (4) $g = \gcd(677 - 5, 1133) = 1$; (5) $x = 26, y = (667^2 + 1)^2 + 1 \% 1133 = 884$; (6) $g = \gcd(884 - 26, 1133) = \gcd(858, 1133) = 11 \Rightarrow n$ 的一个真因子为 **11**。

② Pollard's P-1 算法

选择随机整数 $b, 1 < b < n$:

1. 计算 $g = \gcd(b, n)$, 若 $g > 1$, 找到真因子, 停止; 否则令 $p_i = \{2, 3, 5, 7, \dots\}$
2. 令 $i = 0$, 按 $q = p_i, r = \text{floor}(\log_q n)$, 计算 $b^{q^r} \bmod n, g = \gcd(b^{q^r} - 1, n)$

课件例 ($n=9991$): (1) $b=3, g = \gcd(3, 9991) = 1$; (2) $q=2, r = \text{floor}(\log_2 9991) = 13, 3^{2^{13}} \bmod 9991 = 229, g = \gcd(229, 9991) = 1$; (3) $q=3, r = \text{floor}(\log_3 9991) = 8, 3^{3^8} \bmod 9991 = 3202, g = \gcd(3202, 9991) = 97 \Rightarrow n = 9991$ 的一个因子为 97。

针对协议的攻击

① 密钥使用不当 (骗取签名)

攻击者 Eve 选 $r < n$, 计算 $t = r^e \bmod n, x = r^e \bmod n, y = x \cdot c \bmod n$
计算 $t \cdot u = r^{-1} \cdot y^d = r^{-1} \cdot (x \cdot c)^d = r^{-1} \cdot r \cdot m \bmod n = m$

攻击者获得 n, e_1, e_2, c_1, c_2

② 共用模数 n

$$c_1 = m^{e_1} \bmod n$$

$$c_2 = m^{e_2} \bmod n$$

若 e_1 和 e_2 互素, 有 $r \cdot e_1 + s \cdot e_2 = 1$, 则可根据

$$(c_1)^r \times (c_2)^s = m^{(r \cdot e_1 + s \cdot e_2)} = m \bmod n$$

2.4.3 ElGamal 算法

过程

1985 年发表, 既可用于数字签名又用于加密。其安全性依赖于离散对数难题。

【密钥生成】

选取素数 $p > 10^{150}$, 一个模 p 的原根 g , 以及随机整数 $d (1 < d < p)$

计算 $e = g^d \bmod p$

则公钥为 (e, g, p) , 私钥为 d

【加密 m 】选择随机数 k , 得到密文对 (a, b) :

$$a = g^k \bmod p$$

$$b = m \cdot e^k \bmod p$$

【解密 (a, b) 】

$$b \cdot a^{-d} \bmod p = m \cdot e^k \cdot (g^k)^{-d} \bmod p = m$$

课件算例: $p=11, g=2$, 私钥 $d=4 \Rightarrow$ 公钥 $e = 2^4 \bmod 11 = 5$; 加密消息 $m=3$, 选 $k=4$:

$a = 2^4 \bmod 11 = 5$; $b = 3 \cdot 5^4 \bmod 11 = 5$

解密 (5,5): $5 \cdot 5^{-4} \bmod 11 = 3 \checkmark$

特点: 通过选择不同的 k , 即使是相同的加密密钥, 可将相同明文加密成不同的密文对。

2.4.4 ECC 算法

基本概念

1985 年 N.Koblitz 和 V.Miller 分别独立提出了椭圆曲线密码体制 (ECC), 其安全性依赖于定义在椭圆曲线点群上的离散对数问题 (ECDLP) 的难解性。

已知有限域 $GF(p)$ ($p = q^n, q > 3$) 上的椭圆曲线群:

$$E_p(a,b): y^2 = x^3 + ax + b \pmod{p}, a, b \in GF(p), 4a^3 + 27b^2 \neq 0$$

性质 (设 $P, Q \in E_p(a,b)$):

- $P + O = P, P + Q = Q + P$ (O 为无穷远点)
- 若 $P + Q = O$, 则 $Q = -P$ 为 P 的加法逆元
- 设 $P = (x_1, y_1), Q = (x_2, y_2), P \neq -Q$, 则 $P+Q = (x_3, y_3)$:
 - $x_3 = \lambda^2 - x_1 - x_2 \pmod{p}$
 - $y_3 = \lambda(x_1 - x_3) - y_1 \pmod{p}$
 - $\lambda = (y_2 - y_1)/(x_2 - x_1)$ 当 $P \neq Q$; $\lambda = (3x_1^2 + a)/(2y_1)$ 当 $P = Q$
- 点 Q 的倍数: 在 Q 点做一条切线, 与椭圆曲线相交于点 S , 则 $2Q = Q+Q = -S$
- 在 $E_p(a,b)$ 中有 $P, Q, Q = kP, k < p$, 则由 k 和 P 算 Q 易, 由 P 和 Q 算 k 难

ECC 算法过程

选取 $E_p(a,b)$ 和生成元 G , 公开。将明文消息 m 通过编码嵌入到曲线上的点 P_m 。

【密钥生成】Bob 选取私钥 d , 公钥为 $E = dG$

【加解密】Alice 将消息 P_m 发送给 Bob

Alice (随机选 k) $\longrightarrow C_m = \{kG, P_m + kE\} \longrightarrow$ Bob

Bob 计算: $P_m + kE - dkG = P_m$

课件算例

选 $E: y^2 = x^3 + x + 6 \pmod{11}$, 生成元 $G = (2, 7)$

首先计算 $2G$:

因为 $\lambda = (3x_1^2 + 1)/(2y_1) = (3 \times 2^2 + 1)/(2 \times 7) \pmod{11} = 8$

于是 $x_3 = \lambda^2 - x_1 - x_2 \pmod{11} = 5$

$y_3 = \lambda(x_1 - x_3) - y_1 \pmod{11} = 2$

所以 $2G = (5, 2)$

G 的各倍数 (G 的阶为 13):

$G=(2,7), 2G=(5,2), 3G=(8,3), 4G=(10,2), 5G=(3,6), 6G=(7,9),$

$7G=(7,2), 8G=(3,5), 9G=(10,9), 10G=(8,8), 11G=(5,9), 12G=(2,4)$

【加解密】

1) 设 Bob 私钥 $d=7$, 则公钥 $E = 7G = (7, 2)$

2) Alice 加密消息 $P_m=(10,9)$, 选随机数 $k=3$,

计算 $C_m = \{kG, P_m + kE\} = \{(8,3), (10,2)\}$

3) Bob 用私钥 d 解密: $P_m = (10,2) - 7(8,3) = (10,9)$

ECC 与 RSA 的比较

特点：安全性高，密钥量小，灵活性好。

同等安全条件下的密钥量	RSA	ECC
—	512	106
—	768	132
—	1024	160
—	2048	211
—	21000	600

结论：达到同等安全强度，ECC 的密钥短得多。

2.4.5 Knapsack 算法

背景

由 Ralph Merkle 和 Martin Hellman 开发，只能用于加密，安全性基于子集和问题，已被证明不安全。

背包问题描述： $y = d_1x_1 + d_2x_2 + \dots + d_nx_n$ ，明文分组长度为 n ，消息为 $x_1 \sim x_n$ ，密文为 y 。

课件例：明文 1 1 0 1 0 1；背包 62 93 81 88 102 37；密文 $y = 62 + 93 + 88 + 37 = 280$

解密： $280 = 62x_1 + 93x_2 + 81x_3 + 88x_4 + 102x_5 + 37x_6$ （需 2^6 次测试）

奥妙

背包问题有两种：**普通背包难解；超递增背包易解。**

超递增背包： $d_1 + d_2 + \dots + d_i < d_{i+1}$

于是存在一个较简单的方法：

- 如果 $y < d_n$ ，则 d_n **不可能** 出现在大小为 y 的背包中
- 如果 $y \geq d_n$ ，则 d_n **必须** 出现在大小为 y 的背包中

课件例： $280 = 62x_1 + 93x_2 + 81x_3 + 88x_4 + 102x_5 + 37x_6$ ，转换为 $70 = 2x_1 + 3x_2 + 6x_3 + 13x_4 + 27x_5 + 52x_6 \pmod{105}$

存在的问题：授权者能解密，非授权者亦可。

解决：将超增长背包转换为普通背包。

Knapsack 算法过程

【密钥生成】

选择一个超递增 $(d_1, \dots, d_n)$ ，整数 $m > d_1 + \dots + d_n$ ， $(d_i, m) = 1$ 以及 t 满足 $(t, m) = 1$

计算： $e_i \equiv t \times d_i \pmod{m}$

则公钥为背包序列 $(e_1, e_2, \dots, e_n)$ ，私钥为 $(d_1, \dots, d_n)$ 或 t

课件数值： $\{2, 3, 6, 13, 27, 52\}$ $m=105$, $t=31$

公钥 $\{62, 93, 81, 88, 102, 37\}$

【加密】 n 位的消息 $x=(x_1, \dots, x_n)$ ，得到密文：

$y = e_1x_1 + e_2x_2 + \dots + e_nx_n$

例： $110101 \Rightarrow y = 62 + 93 + 88 + 37 = 280$

【解密】对密文 y ：

$t^{-1} \times y \equiv t^{-1} \times (e_1x_1 + \dots + e_nx_n) \equiv d_1x_1 + \dots + d_nx_n \pmod{m}$

例： $31^{-1} \times 280 = 70 = 2x_1 + 3x_2 + 6x_3 + 13x_4 + 27x_5 + 52x_6$

2.4.6 小结：公钥密码算法比较

算法	公钥 e 和私钥 d 关系	加密 m	解密 c	安全性
----	-------------------	--------	--------	-----

RSA	$d = e^{-1} \bmod \varphi(n)$	$c = m^e \bmod n$	$m = c^d \bmod n$	因子分解难题
ElGamal	$e = g^d \bmod p$	$a = g^k \bmod p$ $b = m \cdot e^k \bmod p$	$b \cdot a^{-d} \bmod p$	离散对数难题
ECC	$E = dG$	$C_m = \{kG, P_m + kE\}$	$P_m + kE - dkG$	椭圆曲线上的离散对数难题

相同：都是公钥密码算法，**密钥管理简单，运算速度慢**

区别：相同加密密钥下，**ElGamal 和 ECC 可将相同明文加密成不同的密文对**（RSA 不行）

公钥 / 私钥对的使用

- 公钥 / 私钥对相互对应，**公钥公开，私钥保密**
- 公钥加密的消息只能用**私钥解密**；反之私钥加密（签名）的消息只能用对应的公钥验证
- 用于加密：任何人用 Bob 的公钥 k_{Bp} 对消息 m 加密，只有 Bob 才能用私钥 k_{Bv} 解密
- 用于数字签名：Alice 用私钥 k_{Av} "加密"消息 m ，其他人可以用 Alice 的公钥 k_{Ap} 验证

攻击者**没有私钥** $\rightarrow$ 无法解密密文、无法伪造签名或篡改消息。

2.5 数字签名算法

相关背景

1991 年 NIST 提出 DSA 用于数字签名标准中；美国（2000 年）和中国（2005 年）通过法律颁布数字签名和传统签名具有同等法律效力。

但 RSA 算法的供应商反对：DSA 不能用于加密或者密钥分配；由 NSA 研制，可能有陷门；速度比 RSA 慢；RSA 是事实上的标准。

1994 年该标准被颁布。NIST 1998 年 12 月规定 DSA 或 RSA 签名方案都可用于数字签名，2000 年又颁布**椭圆曲线数字签名算法 ECDSA** 也可用于签名。

签名的性质：可信，防伪造，防篡改，防抵赖，不可重用。

2.5.1 RSA 签名算法

过程

【密钥生成】

选择两个大素数 p 和 q ，计算 $n = pq$ 以及 $\varphi(n) = (p-1)(q-1)$

选择随机数 $1 < e < \varphi(n)$ ， $\gcd(e, \varphi(n)) = 1$ ，计算 $d = e^{-1} \bmod \varphi(n)$

公开公钥 (e, n) ，保密私钥 (d, p, q)

【签名过程】Alice 对消息 m 签名，得到 $s = m^d \bmod n$ ，发送 $(m \parallel s)$

【验证过程】Bob 验证签名 s ： $s^e \bmod n$ 是否等于 m

攻击 I —— 利用指数运算的特点（骗取签名）

场景：Alice 伪造消息 m' （仰慕之情）获取 Bob 对消息 m （借据 50 万）的签名。

Alice 选 k ，计算 $y = k^e \bmod n$ ， $m' = y \cdot m \bmod n$

Bob（用私钥 d 对 m' 签名）： $s(m') = (m')^d = (y \cdot m)^d = y^d \cdot m^d = k \cdot m^d \bmod n$

Alice（用 k 对 $s(m')$ 处理）：

$$s(m') \times k^{-1} = k \cdot m^d \times k^{-1} = m^d \bmod n = s(m)$$

思考：这其实是"盲签名"的思想 —— 让签名者在不知道内容的情况下签名。

攻击 II —— 针对密钥使用不当（获取密文）

场景：Carol 想知道 Alice 发给 Bob 的秘密消息。

Carol 选 $r < n$, 计算 $x = r^e \bmod n$, $y = x \cdot c \bmod n$

Bob (签名): $u = y^d \bmod n$

Carol 计算: $r^{-1}u = r^{-1}y^d = r^{-1}(x \cdot c)^d = r^{-1} \cdot r \cdot m \bmod n = m \checkmark$

防范: 加密密钥对 $\neq$ 签名密钥对。

2.5.2 DSA 签名算法

ElGamal 签名算法 (DSA 的前身)

签名密钥: 签名私钥为 d , 验证公钥为 $e = g^d \bmod p$

选择 k ($\gcd(k, p-1) = 1$), 计算签名对 (r, s) :

$$r = g^k \bmod p$$

$$m = (dr + ks) \bmod p-1$$

验证: 若 $e^r \cdot r^s \bmod p = g^m \bmod p$, 则签名有效

数字签名算法 DSA (Digital Signature Algorithm): 是 ElGamal 和 Schnorr 签名方案的变形; 1994 年被 NIST 采纳作为数字签名标准 DSS, 使用 SHA 作为散列函数。

DSA 算法描述

【密钥参数】

p : L 位长的素数, $512 \leq L \leq 1024$, $64 \mid L$

q : 160 位的素数且 $q \mid p-1$

g : $g = h^{((p-1)/q)} \bmod p$, $1 < h < p-1$, $g > 1$

用户私钥: d , $0 < d < q$ 用户公钥: e , $e = g^d \bmod p$

【签名】选 $k < p$, 计算 m 的签名 (r, s) :

Alice: $r = (g^k \bmod p) \bmod q$

$$s = (k^{-1}(H(m) + dr)) \bmod q$$

发送 (r, s) , $H(m)$

【验证】

Bob: $w = s^{-1} \bmod q$

$$u1 = (H(m) \times w) \bmod q$$

$$u2 = (r \times w) \bmod q$$

$$\text{验证: } r == ((g^{u1} \cdot e^{u2}) \bmod p) \bmod q$$

DSA 安全性

出于安全考虑, DSA 至少要 2048 位。

若 Eve 获得同一个 k 签名的两个消息, 则可恢复 d :

$$s1 = (k^{-1}(H(m1) + dr)) \quad s1/s2 = (H(m1)+dr)/(H(m2)+dr)$$

$$s2 = (k^{-1}(H(m2) + dr))$$

$$s1 \cdot (H(m2)+dr) = s2 \cdot (H(m1)+dr) \implies \text{可解出 } d$$

结论: k 的随机产生很关键。

Schnorr 算法描述

【密钥参数】

选取素数 p 和 q , 且 $q \mid p-1$, $a \neq 1$ 且 $a^q \equiv 1 \pmod{p}$

用户私钥: d , $0 < d < q$

用户公钥: $e, e = a^{(-d)} \bmod p$

【签名】选 $k < q$, 计算 m 的签名 (r, s) :

Alice: $x = a^k \bmod p, r = H(m, x), s = (k + dr) \bmod q$
发送 $(r, s), m$

【验证】

Bob: $x' = a^s \cdot e^r \bmod p$
验证: $r == H(m, x')$

2.5.3 离散对数签名方案

通用框架

DSA 是离散对数签名方案的一种。

选择大素数 p 和 $q, q \mid p-1$
选择 g 满足: $1 < g < p$ 且 $g^q = 1 \bmod p$
选私钥 $d < p$, 则公钥 $e = g^d \bmod p$

【签名与验证】

选择随机数 $k < p, r = g^k \bmod p, r' = r \bmod q$
签名等式: $a \cdot k = b + c \cdot d \pmod{q}$
验证等式: $r^a \bmod p$ 是否等于 $g^b \cdot e^c \bmod p$

a, b, c 的值是 r', m, s 的组合, 产生多达 13000 种变型。

2.5.4 小结

算法	公钥 e 和私钥 d 关系	签名 m	验证 s
RSA	$d = e^{-1} \bmod \varphi(n)$	$s = m^d \bmod n$	$s^e \bmod n == m$
DSA	$e = g^d \bmod p$	$r = g^k, s = k^{-1}(H(m) + dr)$	$w = s^{-1}; r == g^{H(m)w} e^{rw}$

重要区分

保密通信的密钥对不同于数字签名的密钥对。

RSA 算法公钥 k_p 公开, 私钥 k_v 保密, 运算为 $x^y \bmod n$:

- 保密通信: 用公钥 k_p 加密 $m: E_{k_p}(m) \rightarrow c$; 用私钥 k_v 解密 $c: D_{k_v}(c) \rightarrow m$
- 数字签名: 用私钥 k_v 签名 $m: S_{k_v}(m) \rightarrow s$; 用公钥 k_p 验证 $s: V_{k_p}(s) == m$

2.6 国密算法系列

背景

由国家密码局发布, 主要有 SM 系列。算法公开的有 SM2、SM3、SM4, 密钥长度和分组长度均为 128 位。

国密算法	对应国际算法	特点
SM1	AES	对称密码 (SM7), 芯片调用
SM2	ECC256	公钥密码, 基于 ECC
SM3	SHA256	数字摘要算法
SM4	DES	对合运算, 非平衡 Feistel 网络
SM9	ECC256	基于对的身份标识密码算法

分组密码算法 SM4

- 2006 年公布, 无限局域网产品使用的 SM4 算法

- 分组长度和密钥长度为 128 位，32 轮，子块 32 位
- 轮函数： $X_{i+4} = F(X_i, X_{i+1}, X_{i+2}, X_{i+3}, rk_i) = X_i \oplus T(X_{i+1} \oplus X_{i+2} \oplus X_{i+3} \oplus rk_i)$
- 非线性置换：4 个平行的 8 位 S 盒
- 线性置换： $C = L(B) = B \oplus (B \ll 2) \oplus (B \ll 10) \oplus (B \ll 18) \oplus (B \ll 24)$

公钥密码算法 SM2

2010 年公布，基于椭圆曲线的 SM2 算法。

曲线方程： $y^2 = x^3 + ax + b \pmod{p}$

杂凑算法 SM3

2010 年公布，在 SHA-256 基础上改进的 SM3 算法。

分组长度 512 位，摘要长度 256 位，64 次迭代。

【初始化或填充缓冲区】

A=0x7380166f B=0x4914b2b9 C=0x172442d7 D=0xda8a0600
E=0xa96f30bc F=0x163138aa G=0xe38dee4d H=0xb0fb0e4e

【64 次迭代】

0x79cc4519 $0 \leq t < 16$
Kt =
0x7a879d8a $16 \leq t < 64$

$A \oplus B \oplus C$ $0 \leq t < 16$
ft = f(t, A, B, C) =
(A ∧ B) ∨ (A ∧ C) ∨ (B ∧ C) $16 \leq t < 64$

$A \oplus B \oplus C$ $0 \leq t < 16$
gt = g(t, A, B, C) =
(A ∧ B) ∨ (A ∧ C) $16 \leq t < 64$

X[t] $0 \leq t < 16$
Wt =
Q(W_{t-16} ⊕ W_{t-9} ⊕ (W_{t-3} ≪ 15)) ⊕ (W_{t-13} ≪ 7) ⊕ W_{t-6} $16 \leq t < 68$

W't = Wt ⊕ W_{t+4} $0 \leq t < 64$ Q(x) = x ⊕ (x ≪ 15) ⊕ (x ≪ 23)
P(x) = x ⊕ (x ≪ 9) ⊕ (x ≪ 17)

Vi+1 = ABCDEFG ⊕ Vi
Vn = ABCDEFG

第3章 基本的密码协议

3.1 密码协议概述

什么是密码协议

密码协议又称为安全协议，由密码算法设计而成，在网络环境中提供各种安全服务。

常用“零件”：对称密码算法（DES、IDEA、AES、SM4；RC4）、公钥密码算法（ECC、RSA、SM2）、数字摘要算法（SHA、MD5、SM3）、数字签名算法（RSA、DSA、SM2）。

场景题（课件原题）

Alice 想租房，在某租房网上看到 Bob 的房源，想通过网络完成租房协议签订，设计一个安全高效的通信方案。对应关系：

需求	对应协议
(1) 协议内容保密	保密通信协议
(2) 能防止欺骗	密钥协商协议
(3) 保证签名的性质	身份鉴别协议
(4) 考虑公平性问题	数字签名协议
(5) 不需可信第三方	—
(6) 要求性能最优	秘密共享协议

密码协议分类

按仲裁者分：

- 基于对称密码 $\Rightarrow$ 仲裁协议
- 基于公钥密码 $\Rightarrow$ 裁决协议
- 区别：裁决协议中仲裁者不直接参与协议执行过程
- 仲裁者：公正的第三方，如公证人、法院、KDC 等

按应用分：保密通信协议、密钥协商协议、身份鉴别协议、数字签名协议、秘密共享协议、盲签名协议、安全计算协议、匿名通信协议……

协议的安全性

可以对密码算法和协议的密码技术进行攻击，也可以对协议本身进行攻击。

- 被动攻击：窃听协议的一部分或全部，目的是获取消息，**不影响协议的执行**
- 主动攻击：改变协议以对自己有利，插入、删除和修改消息，破坏通信等等
- 骗子：协议中的一方，不遵守协议或者破坏协议

安全性举例：共享模数 n 的隐患

公钥密码算法本身较安全，若在协议中使用不当将会有安全隐患。

场景：在同一组用户之间共享 RSA 算法的公共模数 n 。

攻击者获得 Carol 王发给 Alice 张和 Bob 李的同一个消息 m 的密文 c_1 和 c_2 ；攻击者很容易知道 n, e_1, e_2 。

$$c_1 = m^{e_1} \bmod n$$

$$c_2 = m^{e_2} \bmod n$$

若 e_1 和 e_2 互素，有 $r \cdot e_1 + s \cdot e_2 = 1$ ，则可以根据

$$(c_1)^r \times (c_2)^s = m^{(r \cdot e_1 + s \cdot e_2)} = m \bmod n$$

3.2 保密通信协议

协议流程（4步）

1. 通信双方协商密码算法和密钥
2. 发送者将明文 m 加密成密文 c
3. 发送者发送密文消息 c
4. 接收者对密文 c 解密获得明文 m

四种设计方案对比

方案	做法	特点
设计 1 古典密码	用简单的古典密码算法（代替密码 / 换位密码）	简单，适用于普通用户； 需要设计和保密算法
设计 2 对称密码	引入 TTP，用户 Alice、Bob 和 TTP 共享密钥 k_A 和 k_B ；消息 m 经 TTP 解密再加密	加解密速度快，引入 TTP 简化密钥管理，但 TTP 易形成瓶颈，安全性依赖于 TTP ，可扩展性差
设计 3 公钥密码	Alice 用 Bob 的公钥 k_{Bp} 加密消息： $E_{Bp}(m)$	密钥管理简单，但加解密速度慢， 安全性依赖于公钥的可靠
设计 4 混合密码	用公钥密码保护和分发 会话密钥 ，会话密钥采用对称密码对通信消息加密，通信结束后销毁会话密钥（即 数字信封 ）	速度快、密钥管理简单 ，安全性依赖于公钥的可靠

交换公钥中的中间人攻击

当 Alice 和 Bob 互传公钥时，黑客截获，用自己的公钥取代，从而能用私钥解密双方传送的消息。

Alice	黑客	Bob
想给 Bob 发	提供 kMp 给 Bob	以为收到了"Alice"的公钥（实为黑客 kMp）
		Bob 用 kMp 加密 → 黑客用 kMv 解密得 m
黑客再用 Bob 的真实公钥 kBp 加密转发给 Bob		

如何避免中间人攻击? $\Rightarrow$ 数字证书 (见 7.3)

数字信封（混合密码）

明文 m — (对称加密 Ek) → Ek(m) —————→ 解密 → m
共享密钥 k — (公钥加密 EBp) → EBp(k) —————→ 解密 → k

↑ Bob 公钥 kBp ↑ Bob 私钥 kBv

3.3 密钥协商协议

流程

通信目的：产生临时会话密钥，实现保密通信。

协议思想：使用密码技术，保证密钥传送的安全性（公钥密码 / 对称密码）。

发送过程：将密钥 k 加密成密文 c ；接收过程：将密文 c 解密后获得密钥 k 。

协议	做法	特点
基于公钥密码	Alice 用 Bob 的公钥加密共享密钥 k : $E_{Bp}(k)$	裁决协议；安全性取决于 Bob 公钥的安全，可能受到中间人攻击
基于对称密码	Alice 和 Bob 与 KDC 有共享密钥 k_A 、 k_B ，借助 KDC 实现密钥协商	仲裁协议；依赖于 KDC 的安全； KDC 参与每一次交换，易形成瓶颈 ，可扩展性差
DH 协议	不需要事先分配公钥或者对称密钥	安全性基于离散对数难题，易受中间人攻击
EKE 协议	用共享密钥 p 加密随机产生的公开密钥 k_{Ap} ，建立会话密钥 k	使用共享密钥和公钥密码，安全性较好

$$\hat{e}(R, PQ) = \hat{e}(R, P) \hat{e}(R, Q)$$

$$\hat{e}(aP, bQ) = \hat{e}(abP, Q) = \hat{e}(P, abQ) = \hat{e}(P, Q)^{ab}$$

$$\hat{e}(P^a, Q^b) = \hat{e}(P^{ab}, Q) = \hat{e}(P, Q^{ab}) = \hat{e}(P, Q)^{ab}$$

三个性质:

- 非退化性: 存在 $P, Q \in G_1$, 使得 $\hat{e}(P, Q) \neq 1$
- 对称性: 存在 $P, Q \in G_1$, 使得 $\hat{e}(P, Q) = \hat{e}(Q, P)$
- 可计算性: 对于任意的 $P, Q \in G_1$, 存在一个高效的算法计算 $\hat{e}(P, Q)$

两方密钥协商

方式一 (PKG 分配私钥): PKG 选择 s , 公开 G, sG ; 给用户分配私钥。

Alice: 公钥 QA , 私钥 sQA , 随机选 r , 计算 $rG \rightarrow$ Bob
 Bob: 公钥 QB , 私钥 sQB
 Alice 计算 $\hat{e}(rsG, QB)$; Bob 计算 $\hat{e}(rG, sQB)$ 两者都等于 $\hat{e}(G, QB)^{rs}$

方式二 (用户自己分配私钥): PKG 公开 G 。

Alice: 公钥 aG , 私钥 a Bob: 公钥 bG , 私钥 b
 交换公钥
 Alice 计算 $\hat{e}(bG, G)^a$; Bob 计算 $\hat{e}(aG, G)^b$ 两者都等于 $\hat{e}(G, G)^{ab}$

三方密钥协商 (一句话协议)

PKG 公开 G ; 网络用户选择私钥分别为 a, b, c , 计算对应的公钥 $P_A = aG, P_B = bG, P_C = cG$ 公开。

$$k = \hat{e}(P_B, P_C)^a = \hat{e}(P_A, P_C)^b = \hat{e}(P_A, P_B)^c = \hat{e}(G, G)^{abc}$$

3.4 数字签名协议

协议流程

协议思想: 使用密码技术, 保证签名性质 (**密文防止篡改, 密钥代表身份**)。

- 签名过程: 将 m 用密钥 k 加密作为消息的签名 $s = E_k(m)$
- 验证过程: 将签名 s 用密钥 k 解密获得 $m' = D_k(s)$ 与 m 比较

四种协议对比 (必背)

数字签名协议	协议特点
① 基于对称密码	需要可信第三方, 小规模场景; 不能保证签名的性质 (协议 1 的缺点); 协议 1+ 需要仲裁人参与, 扩展性差
② 基于公钥密码	利用 hash 函数提高效率 , 扩展性好; 缺点是运算速度慢、效率低
③ 带加密的数字签名	只有指定的授权人才能验证签名; 保密
④ 签名者参与验证	签名需要在线验证; 可防止复制和重用

协议 1 / 1+: 基于对称密码

协议 1

场景: Alice 向 Bob 借 50 万。共享密钥 k 。

缺点: **不能保证签名的性质** (因为 Alice 和 Bob 都知道 k , Bob 也能伪造 Alice 的签名)。

改进: 引入可信第三方 **TTP** 参与协议 (借据由 TTP 中转, 双方各自用共享密钥签名)。

协议 1+ 符号描述

设定: TTP (Trent) 与 Alice 共享密钥 k_A , 与 Bob 共享密钥 k_B 。

	k_A	k_A	
Alice:	$m1 \xrightarrow{EA(m1)}$	$c1 \xrightarrow{DA(c1)}$	$m1 \quad (① \ ②)$
	k_B	k_B	
Trent:	$m2 = S(A) \parallel m1 \parallel c1$		$(③)$
	k_B	k_B	
Bob:	$m2 \xrightarrow{DB(c2)}$	$c2 \xrightarrow{EB(m2)}$	$m2 \quad (⑤ \ ④)$

签名性质分析：

- 签名可信：由 Trent 提供 $S(A)$ 证明
- 签名不可伪造：只有 Alice 知道 k_A
- 签名不可重用、签名文件不能改变： m_2 包含了文件消息 m_1 和 c_1 ， m_1 的改变将使得其与 c_1 不匹配
- 签名不可抵赖：由 $S(A)$ 和 c_1 证明

缺点：需要仲裁人参与，扩展性差，适用于小规模环境。

协议 2 / 2+：基于公钥密码 / hash + 公钥

协议 2 符号描述

用 Alice 的私钥 k_{Av} ，公钥 k_{Ap} 。

Alice	$\xrightarrow{EA_v(m)}$	c	$\xrightarrow{DA_p(c)}$	m	Bob
	K_{Av}		K_{Ap}		

签名性质分析：

1. 签名可信、不可伪造：由 Alice 的私钥保证
2. 签名不可重用：签名是文件的函数
3. 签名文件不能改变：由 Alice 的私钥保证
4. 签名不可抵赖：只有 Alice 拥有私钥

缺点：运算速度慢，效率低。

协议 2+ (sha256RSA)

- ① 签名方： $m \xrightarrow{H(m)} m1$ (sha256 摘要)
- ② 用私钥签名： $S_{Av}(m1) \rightarrow s$
- ③ 发送 $m \parallel s$
- ④ 验证方： $s \xrightarrow{V_{Ap}(s)} m2$ 用签名方公钥验证
 再算 $H(m)$ 与 $m2$ 比较，根据 $m1 = m2$ 判断

为什么是 sha256RSA：先用 hash 把任意长消息压成固定短摘要，再用 RSA 对摘要签名，效率大幅提高。

协议 3：带加密的数字签名（加密 + 签名）

思路

交叉使用“保密通信协议”和“数字签名协议”：

保密通信：	$m \xrightarrow{EB_p(m)} c \xrightarrow{DB_v(c)} m$	接收方公钥 / 接收方私钥
数字签名：	$m \xrightarrow{SA_v(m)} m \parallel s \xrightarrow{VA_p(s)} m$	签名方私钥 / 签名方公钥

带加密的数字签名：

签名方	$\xrightarrow{SA_v(m)} s \xrightarrow{EB_p(s')} c$	
	$m \parallel s \xrightarrow{\hspace{10em}} m \parallel s$	

验证方 $\leftarrow \text{DBv}(c) \rightarrow s' \xrightarrow{\text{VAp}(s)} m$
签名密钥为 Ap/Av 加密密钥为 Bp/Bv

优化思路（课件作业）：利用混合密码协议和哈希+公钥的数字签名协议进行优化。

协议 4：签名者参与验证

DSA 与 RSA 签名算法对比

算法	公钥 e 和私钥 d 关系	签名 m	验证 s
DSA	$e = g^d \bmod p$	$r = g^k; s = k^{-1}(H(m) + dr)$	$w = s^{-1}; r == g^{H(m)w} e^{rw}$
RSA	$d = e^{-1} \bmod \varphi(n)$	$s = m^d \bmod n$	$s^e \bmod n == m$

签名者参与验证的协议（David Chaum）

密钥关系：私钥 d ，公钥 $e = g^d \bmod p$

签名过程：签名方对消息 m 签名得到 $s = m^d \bmod p$

验证过程：

验证方选 a, b

- ① $x = s^a \cdot e^b \xrightarrow{\text{签名方}}$ 签名方
签名方计算 $z = d^{-1} \bmod p-1$
- ② $\leftarrow y = x^z \bmod p$
- ③ 验证： $y = m^a \cdot g^b \bmod p$

协议应用

应用 1：多个参与者的数字签名（RSA 推广）

场景：网络中 Alice 和 Bob 同时签名或者加密消息 m ，让接收者验证或者解密。

RSA 推广：模数 $n = pq$ ， $k_1 \times k_2 \times \dots \times k_m \equiv 1 \bmod \varphi(n)$ （类比 $ed \equiv 1 \bmod \varphi(n)$ ）。

用其中若干个 k_i 加密的消息可以用剩下的 k_j 解密。

举例：Alice 有 k_1 ，Bob 有 k_2, k_3 公开

Alice: $m' = m^{k_1} \bmod n \rightarrow \text{Bob}$

可以恢复 m : $(m')^{(k_2 k_3)} \bmod n$

可以添加自己的签名: $m'' = (m')^{k_2} = m^{(k_1 k_2)} \bmod n$

应用于多人参与的消息认证或秘密共享。

应用 2：阈值信道

场景：Alice 和 Bob 在监控下通过传送签名消息 m 来传送秘密 k 。监控者和 Bob 都能验证 m ；监控者无法发现 k 。

基于 **ElGamal 签名算法**，Alice 私钥为 d ，公钥 $e = g^d \bmod p$ ， $\gcd(s, p-1) = 1$ ：

Alice 计算： $r = g^k \bmod p$ ， $m = (dr + sk) \bmod p-1$

对消息 m 的签名为 r 和 s

监控者 验证： $e^r \cdot r^s = g^m \bmod p$

Bob 验证： $(g^d)^r \cdot r^s = g^m \bmod p$

计算： $k = s^{-1}(m - dr) \bmod p-1$

聚合签名（课件补充）

基于聚合签名与加密交易的全匿名区块链，使用 **Bilinear Pairing**（双线性对）：

$$e(\Pi g_i^{x_i}, h_i) = e(g, \Pi h_i^{x_i})$$

3.5 身份鉴别协议

基本概念

身份鉴别：确认实体是它所声明的，而非假冒。（课件用“天王盖地虎，宝塔镇河妖”、“长江长江，我是黄河”作引子）

单向 hash 链

哈希函数性质：公开性（算法公开）、扩散性（雪崩效应）、抗碰撞性（ $x \neq y$ 时 $h(x) = h(y)$ 的概率很小）、单向性（ $x \rightarrow h(x)$ 易， $h(x) \rightarrow x$ 难）。

单向 hash 链： $r \rightarrow x_0 \rightarrow x_1 \rightarrow x_2 \rightarrow \dots \rightarrow x_{n-1} \rightarrow x_n$

$x_{i+1} = h(x_i)$ ，则 $x_i \rightarrow x_{i+1}$ 易， $x_{i+1} \rightarrow x_i$ 难。

PAP 协议与三个安全问题

PAP 协议 (Password Authentication Protocol)

最简单的口令鉴别：Alice 直接发送“用户名 + 口令”。

Alice — 用户名，口令 —→ 公共网络 —→ 服务器
↑ 黑客可以：网络窃听 / 攻击数据库

安全问题	问题描述	防范策略
① 攻击数据库	黑客侵入服务器数据库，窃取用户口令	使用 hash 函数，保存口令的 hash 值（利用单向性）
② 网络窃听	黑客在公共网络中窃听用户口令	采用加密算法，将口令加密
③ 重放攻击	黑客截获密文伪造身份登录	动态口令、SKEY 协议

SKEY 协议（基于哈希链）

协议流程

【准备阶段】

银行根据 r 和 $x_{i+1} = h(x_i)$ 计算 $x_0 \sim x_{100}$
银行保存 x_{100} ，将 $x_0 \sim x_{99}$ 给 Alice

$x_0 = h(r)$
 $x_1 = h(x_0)$
:
 $x_{100} = h(x_{99})$

【登录（第 1 次）】

Alice — z_s, x_{99} —→ 银行
银行计算 $h(x_{99}) \stackrel{?}{=} x_{100}$ ，成功则 x_{99} 代替 x_{100} 保存

【下次使用】

Alice 用 x_{98} ；银行计算 $h(x_{98}) \stackrel{?}{=} x_{99}$...依此类推

思路：Alice 手里持有一整条链 $x_0 \sim x_{99}$ ，倒着用（从 x_{99} 开始往 x_0 走）；服务器只保存一个最新的散列值。

优点：

- 由于每个数只用一次，因此可以抵抗重放攻击
- 服务器只保存一个最新散列值，节约存储开销

基于挑战应答的身份鉴别协议

使用对称密码机制

前提：拥有共享密钥 k 。 思想：根据密钥 k 确认身份。

- ① Bob $\rightarrow$ RA $\rightarrow$ Alice
- ② Alice 选随机数 RB , 计算 $Hk(RA, RB, B) \rightarrow$ Bob
- ③ Bob 计算并比较 $RB, Hk(RA, RB, B)$
- ④ Bob $\rightarrow Hk(RB, A) \rightarrow$ Alice
- ⑤ Alice 计算并比较 $Hk(RB, A)$

使用公钥密码机制

前提：用户保存自己的私钥 k_v , 公开公钥 k_p 。

思想：拥有与公钥 k_p 对应的私钥 k_v 的人即可确认身份。

Bob $\rightarrow$ 请求 $\rightarrow$ Alice
Bob $\rightarrow$ 随机数 $r \rightarrow$ Alice
Alice $\rightarrow zs, EAv(r) \rightarrow$ Bob (对 r 加密)
Bob 在数据库中查找 Alice 的公钥 k_{Ap} 验证

3.6 秘密共享协议

应用背景

- **密钥托管**：必须在加密系统中加入能够保证法律执行部门可方便获得明文的密钥恢复机制，否则将不允许推广使用，并于 1993 年颁布了 EES 标准。
矛盾：政府主张托管一把密钥用于反恐（国家安全）；民间团体质疑后门大开侵犯隐私（公民隐私）。
- **核按钮手提箱**：总统、国防部长和参谋总长中有两个被按下按钮，俄军战略部门就会收到核打击代码。
- **航天测控网**横跨全球，密钥若集中存放一旦被敌方获取将危及航天器安全 $\Rightarrow$ 门限分割 + 托管备案 ((3,5 门限方案) + (2,5 门限方案))。

核心概念（易混，务必分清）

- **秘密共享**：秘密分割成 n 份，只有当足够多的 k ($k \leq n$) 个成员达成一致时才能共享该秘密
- **秘密分割**：秘密分割成 n 份，当 n 份秘密份额全部收齐后才能恢复该秘密
- **门限方案**：秘密分割成 n 份； k (或 $\geq k$) 份秘密份额便可恢复；任何 $k-1$ (或 $< k$) 份秘密份额不能恢复出秘密

秘密分割协议设计

简单分割 vs 加密分割

简单分割：将秘密简单从物理上分割。

分割前的明文 11011920161008610000
分割后的密文 1101 1920 1610 0861 0000

安全隐患：秘密份额中直接包含明文，安全性差。

加密分割：将秘密通过加密变换成若干密文。

选择 4 个随机数 $k_1 \sim k_4$, 计算 $c = m \oplus k_1 \oplus k_2 \oplus k_3 \oplus k_4$

分割前的明文 m
分割后的密文 $c \quad k_1 \quad k_2 \quad k_3 \quad k_4$

$$c \oplus k_1 \oplus k_2 \oplus k_3 \oplus k_4 = m \oplus k_1 \oplus k_2 \oplus k_3 \oplus k_4 \oplus k_1 \oplus k_2 \oplus k_3 \oplus k_4 = m$$

特点：明文被隐藏，安全性好；但必须全部份额收齐才能恢复（不是门限方案）。

Shamir 门限方案

协议思想与流程

协议思想：(k, n) 门限方案，秘密分割成 n 份， $\geq k$ ($k < n$) 份可恢复。

【秘密分割】

1. 构建 $k-1$ 次多项式 $f(x) = m + a_1x + a_2x^2 + \dots + a_{k-1}x^{(k-1)} \mod p$
2. 计算 $D_i = f(i)$, $i = 1, \dots, n$
3. 将 $D_1, D_2, \dots, D_n$ 分给 n 个人保管

【秘密恢复】（拉格朗日插值）

1. 获取 k 个秘密份额 $D_{j1}, D_{j2}, \dots, D_{jk}$
2. 计算 $f(x) = \sum_i (D_{ji} \times \prod_{s \neq i} (x - j_s) / (j_i - j_s)) \mod p$
3. 恢复的秘密 $m = f(0)$

课件例：(3,5) 门限方案

1. 构建 2 次多项式为： $f(x) = 13 + 10x + 2x^2 \mod 17$
2. 计算 5 个秘密份额：
 - $D_1 = f(1) = 13+10+2 = 25 \mod 17 = 8$
 - $D_2 = f(2) = 13+20+8 = 41 \mod 17 = 7$
 - $D_3 = f(3) = 13+30+18 = 61 \mod 17 = 10$
 - $D_4 = f(4) = 13+40+32 = 85 \mod 17 = 0$
 - $D_5 = f(5) = 13+50+50 = 113 \mod 17 = 11$
3. 将 $D_1 \sim D_5$ 分给 5 个人保管

【恢复】取 3 个份额 $D_1=8, D_3=10, D_5=11$ ：

设 $f(x) = m + a_1x + a_2x^2 \mod 17$

$$8 = m + a_1 + a_2 \quad (x=1)$$

$$10 = m + 3a_1 + 9a_2 \quad (x=3)$$

$$11 = m + 5a_1 + 25a_2 \quad (x=5)$$

求解方程得到秘密 $m = 13$ ← 也等于 $f(0) = 13$

关键点：秘密藏在常数项 $m = f(0)$ ； n 个人各拿一个点；凑够 k 个点就能唯一确定 $k-1$ 次多项式。

Asmuth-Bloom 门限方案

方案描述

选取 n 个严格递增的 $m_1, m_2, \dots, m_n$, 满足 $\gcd(m_i, m_j) = 1$

计算 $M_t = t$ 个 m_i 的乘积

$$S_k = m_1 \times m_2 \times \dots \times m_{k-1} \times m_k$$

$$L_{k-1} = m_n \times m_{n-1} \times \dots \times m_{n-(k-1)+1}$$

要求 $S_k \geq (N+1)L_{k-1}$, N 为一个大的整数

对于任意的秘密 x ($L_{k-1} < x < S_k$)

计算： $a_i = x \% m_i$

则 $\{(a_1, m_1), (a_2, m_2), \dots, (a_n, m_n)\}$ 是一个 (k, n) 门限方案

课件数值： $m_1=9, m_2=11, m_3=13, m_4=14$

取 $k=3, n=4$: $S_3 = 9 \times 11 \times 13 = 1287$; $L_2 = 14 \times 13 = 182$; 满足 $182 < 500 < 1287$ 。

则消息 $x = 500$ 的 (3,4) 门限方案为: $\{(5,9) (5,11) (6,13) (10,14)\}$

$$\begin{aligned}a_1 &= 500 \bmod 9 = 5 \\a_2 &= 500 \bmod 11 = 5 \\a_3 &= 500 \bmod 13 = 6 \\a_4 &= 500 \bmod 14 = 10\end{aligned}$$

为什么有效 (可解性分析)

假设已知 t 个份额 (a_i, m_i) , $1 \leq t \leq n$, 则可构建 t 项的同余方程组, 用中国剩余定理求解:

$$x \equiv a_1 T_1 W_1 + \dots + a_t T_t W_t \pmod{M_t}, \quad M_t = m_1 m_2 \dots m_t$$

根据前面已知的 S_k 为 k 个最小 m_i 的乘积, 而 L_{k-1} 为 $k-1$ 个最大 m_i 的乘积, 则有:

$$M_{t < k} \leq L_{k-1} < x < S_k \leq M_{t \geq k}$$

故可以知道当 $t \geq k$ 时可以获得 x , 反之不能。

课件算例

【已知 $(5,9)(5,11)(6,13)$ 】— 3 份, 够 $k=3$

$$\begin{aligned}x &\equiv 5 \pmod{9} \\x &\equiv 5 \pmod{11} \\x &\equiv 6 \pmod{13}\end{aligned}$$

$$\begin{aligned}x &= 5 \times 11 \times 13 \times (11 \times 13)^{-1} \bmod 9 \\&\quad + 5 \times 9 \times 13 \times (9 \times 13)^{-1} \bmod 11 \\&\quad + 6 \times 9 \times 11 \times (9 \times 11)^{-1} \bmod 13 \quad (\bmod 1287) \\&= 5 \times 143 \times (-1) + 5 \times 117 \times 8 + 6 \times 99 \times 5 \bmod 1287 \\&= 500 \bmod 1287\end{aligned}$$

【已知 $(5,9)(6,13)$ 】— 只有 2 份, 不够 $k=3$

$$\begin{aligned}x &\equiv 5 \pmod{9} \\x &\equiv 6 \pmod{13} \\x &= 5 \times 13 \times (13^{-1} \bmod 9) + 6 \times 9 \times (9^{-1} \bmod 13) \bmod 117 \\&= 617 \bmod 117 = 32 \bmod 117\end{aligned}$$

所以 x 可能的值为: 266, 383, 500, 617, 734, ... ← 无法唯一确定!

结论: 凑够 k 份能唯一确定; 不足 k 份只能缩小范围, 不能恢复。

DH 密钥托管协议设计

流程

Alice 选择 5 个比 $p-1$ 小的整数 $S_1 \sim S_5$
Alice 的私钥为: $S = (S_1 + S_2 + S_3 + S_4 + S_5)$
公钥为: $t = g^S \bmod p$
计算 $t_i = g^{(S_i)} \bmod p$

Alice — S_1, t_1 → 1 号托管人
 : i 号托管人

Alice — t → Bob 对 t_i 签名, 保存 S_i
Bob 验证: $t = t_1 \times t_2 \times t_3 \times t_4 \times t_5 \bmod p$
成立则认可该公钥

特点: 公钥的托管, 可靠性验证。

小结：四种秘密共享方法对比

协议	特点
简单分割	秘密份额中 包含明文 ，安全性差
加密分割	明文被隐藏，安全性好
门限方案	部分秘密份额可恢复出明文
DH 方案	公钥的托管，可靠性验证

秘密共享的作用：保证重要信息和秘密数据的安全。

课件作业：已知 $m_1=3, m_2=5, m_3=7, m_4=8$ ，求消息 $x=60$ 的 (3,4) 门限方案的秘密份额。

第 4 章 中级密码协议

4.1 三次传输协议

应用背景（两个引子）

1. 不协商密钥能否进行保密通信？（对应三次传输协议）
2. 网络虚拟世界如何保证公平性？（对应智力扑克、不经意传输）

Shamir 三次传输协议

协议描述

协议目的：允许 Alice 和 Bob 在不知道对方密钥的情况下通信，不能防止中间人攻击。

协议描述：Alice 和 Bob 协商一个加密算法 E。算法可公开，满足交换律，如 RSA、异或。

A 方加密 B 方选择 A 方解密 B 方解密
原始明文 m $\xrightarrow{E_kA(m)}$ 密文 $\xrightarrow{E_kB(E_kA(m))}$ $\xrightarrow{E_kB(m)}$ $\rightarrow$ 获得明文 m

Alice: 选 k_A 对 m 加密: $E_{k_A}(m) \rightarrow$
Bob: 选 k_B 对信息加密: $E_{k_B}(E_{k_A}(m)) \rightarrow$
Alice: 用 k_A 对信息解密: $E_{k_B}(m) \rightarrow$
Bob: 用 k_B 对信息解密: $m \checkmark$

三次传输协议的特点：**重重加密，依次解密；加解密的运算顺序可交换。**

思考题：能否用异或运算？分析安全性？

（异或的问题： $E_{k_A}(m) = m \oplus k_A$ ，则 $E_{k_B}(E_{k_A}(m)) = m \oplus k_A \oplus k_B$ ；Alice 用 k_A 解密得到 $m \oplus k_B$ ，再传给 Bob，Bob 用 k_B 解密得到 m 。看上去可行，但攻击者若截获全部三段，可推出 $k_A \oplus k_B$ 等信息，安全性差。）

匿名密钥分配协议

协议思想

在很多协议中密钥的安全性很重要，由可信的 TTP 产生和分配。**如何避免 TTP 知道用户选择的密钥？**

协议思想：TTP 产生足够长的密钥序列，用自己的公钥加密传到网上。Alice 挑出其中一个用自己的公钥再加密。

Alice 密钥 A_p/A_v , TTP 密钥 K_p/K_v

TTP: $E_{K_p}(k_i)$ 加密后发布一批 $\rightarrow$ 网上
Alice: 选其中一个，用 A_p 再加密
 $E_{A_p}(E_{K_p}(k_a))$
TTP: 用 K_v 解密得到 $E_{A_p}(k_a)$ ，但看不到 k_a
Alice: 用 A_v 解密得到 $k_a \checkmark$

协议特点：保证密钥的保密性，减弱对 KDC 安全性的依赖。

智力扑克协议

场景与思想

Alice 和 Bob 通过网络玩五张牌游戏，如何保证**保密性和公平性**。

思想：Alice 发牌，每人发五张牌。Alice 产生 52 个消息 $m_1 \sim m_{52}$ ，用自己的公钥 A_p 加密，发送给 Bob。

- ① Alice 产生 $m_1 \sim m_{52}$
 $c_1 \sim c_{52} = E_{A_p}(m_i) \rightarrow$ Bob
- ② Bob 随机选五张: $E_{B_p}(E_{A_p}(m_j)) \rightarrow$ Alice
- ③ Alice 用 A_v 解密 $\rightarrow E_{B_p}(m_j) \rightarrow$ Bob 用 B_v 解密获得自己的牌

④ Bob 另随机选五张: $E_{Ap}(mk) \rightarrow$ Alice 用 A_v 解密获得自己的牌

协议特点: 公钥密码算法, 避免网络游戏的作弊。

关键: Bob 选牌时是“盲选” (Alice 不知道他选了哪几张); Alice 拿到自己牌的同时无法知道 Bob 的牌。

不经意传输协议 (Oblivious Transfer)

协议目的

Alice 将两个秘密中的一个传给 Bob, 但不知 Bob 收到其中的哪一个。

变体: 1/2-out-of-1、1-out-of-2、1-out-of-n、m-out-of-n

Alice 产生 $kp1/kv1$ 和 $kp2/kv2$

① $kp1, kp2 \rightarrow$ Bob

Bob 选择对称密钥 k

② $\leftarrow E_{pi}(k) \rightarrow$ (Bob 用其中一对公钥加密)

③ Alice 解密: $D_{vj}(E_{pi}(k)) = k \quad (i=j)$

$D_{vj}(E_{pi}(k)) = k' \quad (i \neq j)$

④ $E_k(m1), E_{k'}(m2) \rightarrow$ Bob

⑤ Bob 用 k 解密得到消息的一部分

思考: 是否可用 $\oplus$ 实现?

应用: 隐私保护的安全计算。

基于 ElGamal 的实例

Alice $\rightarrow$ Bob 两个秘密中的 1 个。用户共享 p, g, c_0 ($c_0 = 1$)

ElGamal 密码机制: 加密 选 k , $\{a=g^k, b=m \cdot e^k\}$; 解密 $b \cdot a^{(-d)} \rightarrow m$

【准备阶段】

Bob 选择 $i \in [0, 1]$ 和 d , 计算:

$e_i = g^d \bmod p$

$e_{\{1-i\}} = c_0 \cdot (g^d)^{-1} \bmod p$

Alice 验证: $e_0 \cdot e_1 \equiv c_0 \pmod{p}$, $c_0 \in [2, p-2]$

【传输阶段】

Alice 选择 $k_0, k_1 \in [2, p-2]$, 计算并发送 $\{a_0, b_0, a_1, b_1\}$:

$\{a_0 = g^{k_0}, b_0 = m_0 \cdot e_0^{k_0}\}$

$\{a_1 = g^{k_1}, b_1 = m_1 \cdot e_1^{k_1}\}$

【Bob 解密】

$i=0$ 时: $b_0 \cdot a_0^{(-d)} \rightarrow m_0$

$i=1$ 时: $b_1 \cdot a_1^{(-d)} \rightarrow m_1$

(Bob 只能算出他选择的那一个)

4.1 小结

协议	协议特点
Shamir 三次传输协议	不需协商密钥, 三次传输实现保密通信; 不能防止中间人攻击
匿名密钥分配协议	保证密钥的保密性, 减弱对 KDC 安全性的依赖
智能扑克协议	公钥密码算法, 避免网络游戏的作弊
不经意传输协议	公钥 + 对称密码算法, 实现消息保密传输, 常用于公平性和安全计算中

课件思考题：① 匿名密钥分配协议中能否采用对称密码算法？② 分析 Shamir 三次传输协议中的加密算法如果采用异或运算导致的安全隐患。

4.2 安全多方计算协议

应用背景

解决互不信任的参与方之间保护隐私的协同计算问题。

生活中的例子：

- 数据挖掘（“购买了该商品的用户还购买了……”、“根据浏览猜我喜欢……”）⇒ 隐私泄露
- 基于位置的服务（导航、兴趣点查询）⇒ 隐私保护、可追溯性
- 离线支付 ⇒ 同态加密

核心问题：如何在数据保密的同时实现数据的运算？

同态加密：具有同态性质的加密函数满足

$$D(E(a) \otimes E(b)) = a \otimes b$$

类型	算法
乘法同态	RSA 算法、ElGamal 算法
加法同态	ECC 算法、Paillier 算法
全同态	Gentry 算法

应用场景举例：钱包余额 ≥ 支付金额（在不知道具体金额的前提下完成判断）

设计思想

核心思想（一句话）

将需要计算的数据盲化；数据处理完后去掉盲化因子。

盲化的手段：加上或乘以随机数、哈希或加密。

目标：在计算的同时保证数据的保密性。

协议应用 1：平均薪资的保密计算

场景与流程

某部门 Alice、Bob 和 Carol 想计算出三人的平均薪资，但不能违反公司规定泄露薪资。

Alice 拥有 s_1 , Bob 拥有 s_2 , Carol 拥有 s_3

Bob 随机选 r , 把 $s_2 + r$ 传给 Alice
Alice 加上自己的: $s_1 + s_2 + r$, 传给 Carol
Carol 加上自己的: $s_1 + s_2 + s_3 + r$, 公布

最后: $((s_1 + s_2 + s_3 + r) - r) / 3$ ← 去掉盲化因子 r 即得平均薪资

协议特点：采用盲化因子，数据需要保密传输，实现在保密个人薪资的前提下计算平均工资。

协议应用 2：离散对数的保密计算

场景与流程

Alice 想让 KDC（密钥分配中心，拥有更强的计算能力）帮忙找回私钥 x ，但是不想让 KDC 知道 x 。公钥 y 与私钥 x 之间的关系为： $g^x \equiv y \pmod{p}$ 。

Alice 选 $r < p$, 把公钥 y 盲化成 y' :
 $y' \equiv y \cdot g^r \pmod{p} \longrightarrow$ KDC
KDC 计算出 x' 满足 $g^{x'} \equiv y' \pmod{p} \longrightarrow$ Alice

Alice 根据 $x \equiv (x' - r) \pmod{p-1}$ 得到 x
← 去掉盲化因子得到 x

KDC 只能得到 x' ，保护了用户的隐私

可解性分析（用费马小定理）

根据 $g^x \equiv y \pmod{p}$, $g^{(x')} \equiv y' \pmod{p}$, $y' \equiv y \cdot g^r \pmod{p}$
 $\Rightarrow g^{(x')} \equiv y \cdot g^r \equiv g^x \cdot g^r \equiv g^{(x+r)} \pmod{p}$
 $\Rightarrow g^{(x'-x-r)} \equiv 1 \pmod{p}$ 【费马小定理】
 $\Rightarrow x' - x - r = k(p-1)$ (若 $g^t \equiv 1 \pmod{p}$ 则 $t = k\phi(p) = k(p-1)$)
 $\Rightarrow x' - x - r \equiv 0 \pmod{p-1}$ 【阶的性质】
 $\Rightarrow x \equiv (x' - r) \pmod{p-1} \quad \checkmark$

协议应用 3：姚氏百万富翁问题（保密数据的比较）

场景与参数

Alice 有 x , Bob 有 y , 在不泄露 x 和 y 的情况下比较 x 和 y 的大小。

设 $1 \leq x, y \leq n$, Bob 公钥 k_p , 私钥 k_v 。

课件算例： $1 \leq x, y \leq 4$, $x=4$, $y=2$, $k_p=7$, $k_v=23$, $n=55$

协议流程

【Alice 侧】随机选 k , 计算 $c = E_{k_p}(k)$

课件： $k=39$, $c = 39^7 \pmod{55} = 19$

发送 $c - x = 19 - 4 = 15$ 给 Bob

【Bob 侧】计算 n 个数： $z_i = D_{k_v}(c - x + i)$, $1 \leq i \leq n$

$z_1 = (15+1)^{23} \pmod{55} = 26$

$z_2 = (15+2)^{23} \pmod{55} = 18$

$z_3 = (15+3)^{23} \pmod{55} = 2$

$z_4 = (15+4)^{23} \pmod{55} = 39$

并且：对 $i > y$ 的项，Bob 把 z_i 加 1

($y=2$, 所以 z_3 、 z_4 各加 1: $2+1$ 、 $39+1$)

发送：26, 18, 3, 40

【Alice 侧】检查第 $x = 4$ 个数是否等于 $k = 39$

第 4 个数是 $40 \neq 39 \Rightarrow$ 不等于，所以 $x > y \quad \checkmark$ ($4 > 2$)

判定规则：

检查结果	结论
第 x 个数 等于 k	$x \leq y$
第 x 个数 不等于 k	$x > y$

协议特点：利用两个数据构造相关的位置关系，采用加密的方式实现在保密的前提下进行数据比较。

原理：若 $x \leq y$, 则第 x 个数没被加 1, 等于 $D_{k_v}(c) = k$; 若 $x > y$, 则第 x 个数被加了 1, 等于 $k+1 \neq k$ 。

4.2 小结：安全多方计算协议的流程与特点

首先将数据盲化，数据处理完后再去掉盲化因子。在计算的同时保证数据的保密性。

协议	协议特点
平均薪资的保密计算	采用盲化因子，数据需要保密传输，实现在保密个人薪资的前提下计算平均工资

离散对数的保密计算	采用盲化因子，根据费马小定理还原真实结果，实现在保密数据的前提下计算离散对数值
姚氏百万富翁问题	利用两个数据构造相关的位置关系，采用加密的方式实现在保密的前提下进行数据比较

4.3 密钥分发协议

应用背景

无线传感器网络中的节点可能新加入、可能退出、可能叛变。

思考：如何解决在广播或组播环境下节点之间的保密通信问题？

事例

在一个网络中，A 广播消息 m ，让一部分人获得。假设网络中有 A, B, C, D, E, F 共计 6 人，每两个人之间有共享密钥。

问题：A 将消息 m 广播给 B、C、E，怎么做通信代价和计算代价最小？

传统密钥分发协议

协议思想：A 将消息 m 用会话密钥 k_s 加密，然后分别用和 B、C、E 的共享密钥 k_B, k_C, k_E 加密 k_s 后广播。

广播内容： $E_{k_s}(m) \parallel E_{k_B}(k_s), E_{k_C}(k_s), E_{k_E}(k_s)$

协议分析：A 做 4 次加密计算，广播数据包 4 个；每个接收者做 3 次或 4 次解密计算。

基于 CRT 的密钥分发协议

协议思想：A 基于 k_s 根据中国剩余定理计算得到一个数 R 并广播，使得 B、C、E 通过简单的求模运算获得 k_s ，而其他人不能得到。

$R \equiv k_s \pmod{k_B} \quad \leftarrow B \text{ 有 } k_B$
 $R \equiv k_s \pmod{k_C} \quad \leftarrow C \text{ 有 } k_C$
 $R \equiv 0 \pmod{k_D} \quad \leftarrow D \text{ 无权，得不到}$
 $R \equiv k_s \pmod{k_E} \quad \leftarrow E \text{ 有 } k_E$
 $R \equiv 0 \pmod{k_F} \quad \leftarrow F \text{ 无权，得不到}$

广播内容： $E_{k_s}(m) \parallel R$

协议分析：A 做 1 次加密计算，1 次中国剩余定理计算，广播数据包 2 个；每个接收者做 1 次模运算，0 次或 1 次解密计算。

协议性能分析（假设网络有 n 人，向其中 k 人传送）

密码协议	通信代价	计算代价
传统的密钥分发	$k+1$	发送方： $(k+1)$ 次加密 接收方： $(k+1)$ 或 k 次解密
基于中国剩余定理	2	发送方：1 次加密，1 次 CRT 接收方：1 次模运算，1 或 0 次解密

小结

- 密钥分发：广播或组播环境下节点间的保密通信，也可以应用于密钥更新。
- 传统密钥分发：通信代价与计算代价与通信用户数 k 成线性关系，为 $O(k)$ 。
- 基于 CRT 的密钥分发：通信代价与计算代价与通信用户数 k 成常数关系，为 $O(1)$ 。

4.4 位承诺协议

协议目的与描述

协议目的：敏感信息保护机制中，实现**机密性和不可否认性**。如网络竞标中如何保证投标者不变更价格。

协议描述：Alice 对 Bob 承诺一个位，直到某个时间后才揭示，承诺后不能改变想法。

方案一：使用单向函数的位承诺协议

- ① Alice 产生位承诺 b 和随机位串 r_1, r_2
Alice $\rightarrow H(r_1, r_2, b), r_1 \rightarrow$ Bob

【协议的承诺部分】
- ② Alice $\rightarrow r_1, r_2, b \rightarrow$ Bob
Bob 计算 hash 值并与 H 比较

【协议的揭示部分】

方案二：使用对称密码算法

- ① Alice 产生位承诺 b 和随机密钥 k
Alice 产生随机位串 r ，计算 $c = E_k(r, b)$
Alice $\rightarrow r \rightarrow$ Bob

【协议的承诺部分】
- ② Alice $\rightarrow c \rightarrow$ Bob

③ Alice $\rightarrow k \rightarrow$ Bob

Bob 解密获得 b ，用 r 验证有效性

【协议的揭示部分】

两种方法的比较（必背）

采用算法	位承诺协议特点
散列函数	随机数 r_1 使得 Alice 不能篡改承诺结果
	随机数 r_2 让 Bob 不能猜测承诺结果
对称密码	随机数 r 保证 Alice 不能篡改承诺结果
	密钥 k 用来让 Bob 验证承诺结果

第 5 章 高级密码协议

5.1 零知识证明协议

协议目的与描述

协议目的：A 要向 B 证明她拥有一些秘密，且在证明中不泄漏任何信息。

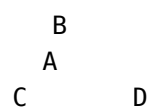
协议描述：采用交互式协议的形式：B 问 A 一系列与秘密无关的问题，若 A 知道秘密，她能回答所有问题；否则只有 50% 的机会欺骗。

著名的洞穴协议

A 向 B 证明她知道打开 CD 门的咒语。

- (1) B 站在洞外，A 进入洞内
- (2) A 选择进入洞穴左边或右边
- (3) B 进洞指定 A 从左边或右边出来
- (4) A 按照要求执行
- (5) 重复 (1)~(4) n 次

洞穴结构：



(C、D 之间有门，只有会咒语的人才能从一边到另一边)

关键：重复 n 次后，欺骗概率为 $(1/2)^n$ ，趋近于 0。但 B 始终没有获得关于咒语的任何信息。

课件拓展：综述至少三个零知识证明的例子。

5.2 盲签名协议

协议目的

签名者不知道所签文件的内容。如公证员只要证明某一时刻公证过这个文件，并不关心文件内容。

完全盲签名

前提：签名函数和乘法函数满足交换律。

Alice (将消息盲化) Bob (对 m' 签名)

$m' = m \times b$

→

$SBv(m')$

←

去掉盲因子 b ，获得签名后的消息 $SBv(m)$

如 RSA 算法中： $m' = m \cdot k^e \bmod n$

特点：

- Alice 在文件上签名有效
- Bob 无法将签署行为与所签文件联系起来

盲签名协议思想（分割选择法）

签名者打开并检查除一个文件之外的所有文件，然后对最后一个文件签名。

Alice 准备 n 个文件 $m_1 \sim m_n$ ，各配随机数 $k_1 \sim k_n$
计算 $ci = (m_i, k_i)$ 的某种盲化形式

发送 $c_1 \sim c_n$ 给 Bob

Bob: 随机选 $n-1$ 个 c_i , 要求 Alice 公开对应的 k_i
Alice 公开 $n-1$ 个 k_i (Bob 验证这些没做手脚)
剩下 1 个未选的 c_p , Bob 对它签名

Alice 收到 $S(c_p)$ 后还原出 $S(m_p)$

分割选择避免欺骗, Alice 能成功欺骗的概率为 $1/n$

课件还给出了**数字现金协议**中用盲签名的完整流程 (见 5.6 区块链部分)。

5.3 同时签约协议

协议事例

Alice 和 Bob 想签订合同, 已经协商好内容, 但**每个人都想等对方签名后再签**。

方案一: 带仲裁者的签约

Alice $\xrightarrow{\textcircled{1}}$ 已签合约 $\xrightarrow{\quad}$ TTP $\xrightarrow{\textcircled{1}}$ 已签合约 $\xrightarrow{\quad}$ Bob
 $\xleftarrow{\quad}$ $\xrightarrow{\textcircled{2}}$ 签合约两份副本 $\xrightarrow{\quad}$
 $\textcircled{3}$ 一份两人签好的合约
 $\xleftarrow{\quad}$ $\xrightarrow{\textcircled{4}}$
 $\textcircled{5}$ 已有合约 $\qquad\qquad\qquad \textcircled{5}$ 已有合约

方案二: 无需仲裁者的同时签约 (面对面)

Alice 和 Bob 依次在合约上逐个签上自己名字的字母。

方案三: 无需仲裁者的同时签约 (非面对面, 概率递增法)

协议中 Alice 和 Bob 交换一系列下面形式的签名: **我同意以概率 p 接受这个合约约束**。

- (1) Alice 和 Bob 协商一个完成日期, 以及双方都愿意接受的概率差
- (2) Alice 发出 $p_A = a$
Bob 回复 $p_B = p_A + b$
Alice 回复 $p_A = p_B + a$
- (3) 重复到双方都收到 $p = 1$ 的消息或已到完成日期

判定: 如果在完成日期之前完成协议, 双方受合约约束; 否则法官随机选 p , 若 $p < (p_A, p_B)$ 则双方受约束。

核心思想: 每一步只增加一点点"接受概率", 谁先违约就吃亏, 从而达到"同时生效"的效果。

方案四: 使用密码技术

- (1) Alice 和 Bob 随机选择 $2n$ 个密钥分成 n 组 (k_{li}, k_{ri}) ,
对 n 对消息 L_i 和 R_i 分别用 k_{li} 和 k_{ri} 加密, 并传送给对方
- (2) 采用不经意传输协议将 $2n$ 个密钥中的一半 k_{li} 或 k_{ri} 传送给对方
- (3) 用收到的密钥解密出一半消息
- (4) 如果能解密其中的任意一对消息, 合约被签署
- (5) 双方交换在第(2)步中使用的私钥, 验证未欺骗

5.4 匿名投票协议

投票需求

生活中的投票需求：授权、防篡改、投一次、确保选票计算、保密（防止欺骗、保护隐私）。

课前思考的问题：如何设计投票协议，使投票具有现场投票的效果（即实现“匿名”）？

投票协议 I（简单网络投票协议）

协议思想：采用中央制表机构 CTF（Central Tabulating Facility）。

Voter 填写选票 t ，用 CTF 公钥加密： $V, \text{ETp}(t) \longrightarrow \text{CTF}$
CTF 将选票解密，统计，公布结果

特点：主办方不知道投票者身份，投票次数不定

问题：无法保证身份和次数

投票协议 II（协议 I + 数字签名）

Voter 填写选票 t ，签名后用 CTF 公钥加密： $V, \text{SVv}(\text{ETp}(t)) \longrightarrow \text{CTF}$
CTF 解密、验证签名后统计，公布结果

特点：保证投票者身份和投票次数

问题：CTF 知道谁投了谁的票（无法匿名）

匿名投票协议设计（用盲签名）

现场投票流程：Voter 领票 $\rightarrow$ Voter 填票 $\rightarrow$ Voter 投出选票 $\rightarrow$ CTF 唱票计票 $\rightarrow$ CTF 公布结果

网络投票流程：Voter 构建选票 $\rightarrow$ CTF 签名选票 $\rightarrow$ Voter 密投选票 $\rightarrow$ CTF 解密选票 $\rightarrow$ CTF 核实选票 $\rightarrow$ CTF 公布结果

选票的构建：每张选票带一个识别号（如 11011011）和选择标记（ $\checkmark$ ），由 CTF 签名。

匿名的保证：

- 采用盲签名协议进行选票签名
- 切断投票者与选票的对应关系，但能鉴别
- 签名者知道在干什么，但不知签名的内容

课件场景：从张三和李四中选取一个人作为歌王胜出。投票者获取 CTF 签名的 100 组选票（号码各不相同），投票时用其中一张，CTF 通过“验证重复”保证一人一票。

投票效果

授权，防篡改，保密，投一次，确保选票计算 —— 使投票具有现场投票的效果。

5.4 小结

投票协议	特点
投票协议 I	投票者身份和次数 不确定
投票协议 II	CTF 知道投票者与选票的关系（ 无法匿名 ）
匿名投票协议	保证了投票的五个特性（授权、防篡改、投一次、确保计票、保密）

5.5 匿名通信协议

匿名消息广播

事例：Alice、Bob 和 Carol 在餐馆进餐，账单匿名支付，能否确定是他们三人中的一个付账还是 NSA 付账？

游戏：两两抛币，每个人说出与其他两人抛币的正反面是“相同”或“不同”；付账的人说假话。

判断依据：说“不同”的人数为
奇数 $\Rightarrow$ 三人中的一个在付账
偶数 $\Rightarrow$ NSA 付账

匿名通信 (DC-Net 思想)

事例：Alice、Bob 和 Carol 发送消息，其中一个人在发送，外面的人不知道是谁发送
规则：不发消息的人发送共享密钥的异或值；发消息的人发送共享密钥与消息 m 的异或值

密钥分配：两两之间共享 k_{12} , k_{13} , k_{23}

Alice 发送： $m_1 = k_{13} \oplus k_{12}$

Bob 发送： $m_2 = k_{12} \oplus k_{23} \oplus m$ $\leftarrow$ 真正发消息的人

Carol 发送： $m_3 = k_{13} \oplus k_{23}$

网关 GW 计算： $m_1 \oplus m_2 \oplus m_3 = m$ $\leftarrow$ 密钥抵消，只剩消息

三个基本概念 (必背)

- 匿名性 (Anonymity)：一种在**一组对象中不被识别**的状态。
- 不可关联性 (Unlinkability)：不能获得系统中对象之间**是否相关**的判断。
- 不可观察性 (Unobservability)：不能观察到对象**是否发送或接收了消息**。

匿名集： $S = \{s_1, s_2, \dots, s_n\}$ ，经过若干中间节点 $P_1, P_2, \dots, P_l$ 到达 R 。

三种匿名

- 通信关系安全：消息的发送者与接收者的保密性
- 身份匿名 (Identity Privacy)：源节点和目的节点的身份信息是保密的，它们也不知道路径上中间节点的真实身份（发送者匿名、接收者匿名、通信关系匿名）
- 位置匿名 (Location Privacy)：源节点和目的节点的具体位置是保密的，其他节点不知道它们到源节点或目的节点的跳数
- 路由匿名 (Route Privacy)：不管攻击者是否在转发路径上，它都不能进行数据包追踪到源节点或者目的节点

四种实现机制

① 代理机制：借助可信赖第三方的身份来隐蔽自己的真实身份。

- 单代理：Penet, Anonymizer 和 LPWA 等
- 多代理：如 OR, Hordes, Morphemix 等

② MIX 机制：MIX 是一些特殊节点，接收一定数量的消息，通过**加密或填充**等改变消息外观，通过**延迟或重排序**等改变消息顺序，隐藏输入输出对应关系，使得攻击者无法推断通信参与者的通信关系，从而实现匿名。

Sender $\rightarrow E_1(M_1, E_2(\dots(M_i, E_{i+1}(\dots(M_l, E_R(M_R))\dots)))\dots)$
 $\rightarrow \text{MIX}_1 \rightarrow \text{MIX}_2 \rightarrow \dots \rightarrow \text{MIX}_i \rightarrow \dots \rightarrow \text{MIX}_l \rightarrow \text{Recipient}$
每一层 MIX 剥掉一层加密

应用：匿名电子邮件 (Babel, MixMaster, MixMinion);
匿名连接服务 (OR, Tarzan, MorphMix, WonGoo);
匿名 Web 浏览 (Crowds, Webmix)

③ Crowds 系统：基于 P2P 网络对等系统组群思想构建的匿名通信系统。

消息在不同的 Jondo 之间随机转发，转发给目的节点的概率为 pf ，
最终被发送给目的节点，发送者就被隐藏。
问题：对称加密机制、转发路径过长

④ **广播或者多播机制**：基于广播技术的 DC-Net、组播技术的 Hordes，以及采用分级广播思想的 P5 系统等。每个节点发送所有共享密钥的异或值，发送者加入消息。

5.6 区块链协议

应用背景

2019 年 10 月 24 日，国家领导人在中共中央政治局学习中指出要加快推动区块链技术和产业创新发展，把区块链作为核心技术自主创新。

依赖大量的第三方中介机构才能保证电子货币的流通，不仅引入中心化的风险，也提升了传输成本，必然会要求一个与之匹配的高效可靠的价值传输系统。

- **电子支付**：银行账户上做数字的加法和减法，只有银行有记账权。
- **新型货币**：如何创造新型支付体系，大家都有权来记账，如比特币。

通过构造一个以**竞争 — 记账 — 奖励**为核心的经济系统解决了去中心化记账的难题。

应用领域：金融领域、票据管理、产品溯源、存证取证、数据共享。

数字现金协议（用盲签名）

协议要求与流程

协议要求：要求匿名，消息可以确认，但不可跟踪。银行和政府不愿意放弃审计追踪控制。

协议实现：盲签名

- ① Alice 准备 n 个装有 1000 元的汇票和复写纸的信封
- ② 把 n 个信封交给银行
- ③ 银行开启 $n-1$ 个确认，在最后 1 个上签名，并从 Alice 账上扣除 1000 元
- ④ 银行把签名的信封还回
- ⑤ Alice 把汇票给商人
- ⑥ 商人检查银行签名
- ⑦ 商人把汇票交给银行
- ⑧ 银行验证汇票并将 1000 元划入商人帐户

三个问题及解决方案

问题 1：双重花费。

解决：在每一张汇票上包含一个**不同且唯一的随机字符串**，银行在签名时确认汇票或验证签名时检查随机字符串的唯一性。

问题 2：当银行发现有人进行双重花费，但不知道是 Alice 欺骗商人还是商人试图欺骗银行。

解决：当商人检查银行签名以确信汇票合法后，要求 Alice 在汇票上写上**识别串**（商人无法改变）。

银行验证签名并检查数据库中的唯一串：

无：划帐，并保存识别串和唯一串

有：拒收，根据识别串 — 与数据库相同：商人欺骗

— 与数据库不同：Alice 欺骗

问题 3：Alice 可能会第二次花一张有同样识别串的汇票陷害商人。

解决：采用**秘密分割技术**用来在数字汇票中隐藏 Alice 的名字。

Alice 准备 n 张 1000 元的汇票：

面值：1000

唯一字符串： x

识别字符串： $I_i = (I_{iL}, I_{iR})$

...

$I_n = (I_{nL}, I_{nR})$

任意一对 $I_i = (I_{iL}, I_{iR})$ 都能揭示 Alice 的身份

流程：银行随机选 $n-1$ 张验证，在最后一张上签名，扣除 1000 元
用汇票时，Alice 给出相应识别串的左半或右半
银行核对 — 无：划帐；有：根据识别串判断
相同：商人欺骗
不同：找出一对 I_i ，揭示 Alice 身份

欺骗概率： $1/n$ （签名阶段）、 $1/2n$ （使用阶段）

什么是区块链

课件原文（两个定义）

科学百科（2019）：区块链是分布式数据存储、点对点传输、共识机制、加密算法等计算机技术的新型应用模式。

中国区块链技术和应用发展白皮书（2016）：广义来讲，区块链技术是利用块链式数据结构来验证与存储数据，利用分布式节点共识算法来生成和更新数据，利用密码学的方式保证数据传输和访问的安全，利用由自动化脚本代码组成的智能合约来编程和操作数据的一种全新的分布式基础架构与计算范式。

- **基于密码学分布式网络账本**：一个点对点的分布式网络（节点同步数据）+ 一份所有交易的账簿数据（区块链）+ 一个去中心化的交易验证（基于密码技术）
- **核心思想**：信任一个有足够信用度的个体，或信任一群人（51%）
- **基本步骤**：广播交易 → 节点记载 → 公开账簿
 - **广播交易**：将每笔交易都在全网广播
 - **节点记载**：节点接收交易信息，记载此次信息，获得奖励
 - **公开账簿**：其他节点核对确认，交易达到 **6 个确认以上**就成功记录在案 ⇒ 一条不可更改的完整区块数据链，可溯源

共识机制

定义与分类

用于在分散的计算机网络中达成协议、信任和安全的方法。

作用：确保所有参与者都能够对网络事件进行有效记录。

分类：竞争式或投票式原理，包括权益证明、工作证明等。

机制	原理
工作量证明 POW (Proof of Work)	节点通过计算随机哈希散列的数值解争夺记账权，达成共识周期长，资源浪费
权益证明 POS (Proof of Stake)	根据每个节点拥有代币的比例和时间，依据算法等比例地降低节点的挖矿难度
股权授权证明 DPOS (Delegated Proof of Stake)	全体节点投票选举出一定数量的节点代表来代理全体节点确认区块

课件拓展：从原理、特点和应用角度综述区块链协议中的共识机制。

比特币

要点

- **起源**：2008 年，一位化名为中本聪的人，在一篇为《比特币：一个点对点的电子现金系统》的论文中首先提出比特币
- **运行**：2009 年，系统开始运行，用于交易和炒作
- **记账权**：工作量证明 POW（Proof of Work）
- **挖矿**：寻找 Nonce 满足 $h(\text{Nonce} \parallel \text{Prev_Hash} \parallel \text{Tx}) < \text{Bits}$
 - $h()$ —— **SHA256** 哈希函数
 - Bits —— 目标值，**48 个 0 开始的比特串**；每 10 分钟产生一个块；每 2016 个块后（两周 = 2016×10 分钟）调整
 - Nonce —— 是一个答案且对于每个区块是唯一的，**第一个找出 Nonce 的用户得到奖金**

- 账本：全世界的矿机共同维护一个比特币历史交易记录
- 奖励减半：50 BTC（210000 个块 / 4 年）→ 25 BTC → 12.5 BTC → 即 $50 \times (1/2)^n$ ；预计 2140 年全部 BTC 被挖完（ 21×10^6 个）

区块链中的密码技术（五类）

#	技术	作用
1	哈希函数 SHA256	用于数据完整性和加密，共识计算的工作量证明，块之间链接
2	Merkle 树	保存信息的哈希值，验证区块中的交易
3	公钥密码	去中心化的保证
4	数字签名 ECDSA	交易拥有者的证明
5	零知识证明	在不公开秘密号码的情况下验证其在已知限制内

区块链特点

特点	说明
高透明性	公有区块链系统开放性，节点共同维护，除私有信息被加密外，任何人可通过公开接口查询。 （同时也带来隐私泄露问题）
高自治性	基于协商一致的规范和协议，使各参与方在不需要信任任何人的环境下进行自由安全的交易
高信任性	按照时间先后顺序标记且难以篡改数据记录，既能保证数据的可追溯性，又能减少造假。 交易不可撤销
高安全性	节点参与数据变动记录，保留一份相同且完整的账本，单个节点不会影响整个账本及记录的完整性
交易确认高延迟	—
淡化监管	—

第 6 章 分组密码加密模式

分组加密模式概述

为什么需要“模式”：对称密码算法的输入固定，而明文消息长度不定，因此加密时要对明文分组再进行加密运算。

- DES：64 位
- AES：128 位

四种基本模式：ECB 模式（基本密码）、CBC 模式（一些反馈）、CFB 模式（一些简单运算）、OFB 模式。

思考的问题：如果密文被替换，是否影响解密？在实际应用中，是否存在安全问题？

填充方案

两种填充方案

消息尾部可能会出现一个短分组；采用一些方式将其填充成一个完整的分组。规则模式和密文挪用两种填充方案。

① 规则模式：全 0、全 1 或者 0,1 交替填充。

X X X 00 0 0 5 ← 最后的 5 表示填充了 5 个字节

② 密文挪用：把前一个分组的密文“借”一部分来补足最后一个短分组。

明文分组	P1	P2	...	P _{n-2}	P _{n-1}	P _n
	E _k	E _k		E _k	E _k	E _k
密文分组	C1	C2	...	C _{n-2}	C _{n-1}	C _n

6.1 ECB 模式

模式机制（Electronic Codebook，电子密码本模式）

ECB 模式又称为电子密码本模式，最简单，每一分组独立进行加密或解密处理。

明文分组	P1	P2	...	P _{i-1}	P _i	...
	E _k	E _k		E _k	E _k	
密文分组	C1	C2	...	C _{i-1}	C _i	...

加密： $C_i = E_k(P_i)$

解密： $P_i = D_k(C_i)$

模式特点（优点）

- 对加密如数据库等随机存取文件非常方便（每一块独立，可随机访问）
- 密文块损坏，仅对应明文块损坏（错误不传播）

模式缺点：易受重放攻击

攻击者可以在不知道密钥的情况下修改被加密的消息，以欺骗接收者。

举例：银行转账（报文按分组划分）：

1~4	5~8	9~12	13~16	17~20	21~24
时间标记	A 银行	B 银行	储户姓名	帐号	金额

黑客攻击：从 A 银行转账到他在 B 银行的帐上，重复该过程并记录消息；挑选与他名字和帐号相对应的密文分组，替换其他人的转账信息。

黑客转账信息：

20160801 A银行 B银行 黑客 999999 10元

Alice 转账信息：

20160815 A银行 B银行 张三 888888 10000元

黑客伪造的信息（把"黑客 999999"那两块换进 Alice 的报文）：

20160815 A银行 B银行 黑客 999999 10000元 ← 可解密吗？

因为 ECB 每一块独立加密，密文分组可以任意拼接、替换，且解密后仍然合法。

防范

- 银行频繁改变密钥
- 最好的方法是采用分组链接方式（如 CBC）

6.2 CBC 模式

模式机制（Cipher Block Chaining，密码分组链接）

CBC 模式是将前一分组的加密结果反馈到当前分组的加密中。

明文分组	P1	P2	...	Pi-1	Pi	...
IV ⊕		⊕		⊕	⊕	
	Ek	Ek		Ek	Ek	
密文分组	C1	C2	...	Ci-1	Ci	...

加密： $C_i = E_k(P_i \oplus C_{i-1})$

解密： $P_i = C_{i-1} \oplus D_k(C_i)$

初始向量 IV

- 采用初始化向量 IV 将每个消息唯一化
- IV 不需保密，可以以明文形式与密文一起传送

为什么：IV 的作用只是“让相同明文在不同消息中得到不同密文”，它本身不参与保密；解密时 IV 也只是异或回去，泄露 IV 不泄露密钥。

课件思考题：① CBC 模式中为什么 IV 可以明文传输？② 分析 CBC 模式中的错误传播。

6.3 CFB 模式

模式机制（Cipher FeedBack，密码反馈）

CFB 模式中的反馈在分组内和分组间进行。

C_{i-1}
↓
加密 (k)
↓
最左端字节 j=1~8 k_j C'_{i-1}
↓
 $P_{ij} \oplus \longrightarrow C_{ij}$

模式分析：一开始 C_0 用 IV 填充，然后加密成 C'_0 ，将 C'_0 和 P_1 的低 8 位异或，形成密文的 8 位，并将此 8 位加入到 C_0 的序列中，重复此过程，直到 P_1 加密完毕，形成 C_1 。

n 位 CFB 模式 (类似 CBC)

CFB 模式

明文分组	P1	P2	...	P _{i-1}	P _i	...
IV	E _k	⊕		⊕	⊕	
		E _k		E _k	E _k	
密文分组	C1	C2	...	C _{i-1}	C _i	...

加密: $C_i = P_i \oplus E_k(C_{i-1})$

解密: $P_i = C_i \oplus E_k(C_{i-1})$

(说明: 课件上解密式写作 $D_k(C_{i-1})$, 但 CFB 的正确形式是加解密都用加密函数 E_k —— 因为 CFB 把分组密码当作“密钥流发生器”来用, 两个方向都只做加密运算。答题时按课件写法即可, 理解上按标准形式。)

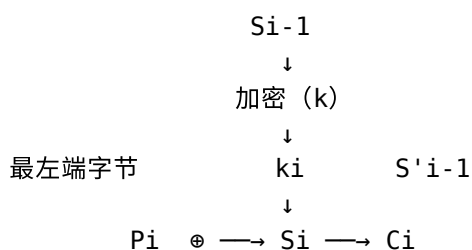
与 CBC 的对比 (必考)

	CBC 模式	CFB 模式
加密	$C_i = E_k(P_i \oplus C_{i-1})$	$C_i = P_i \oplus E_k(C_{i-1})$
解密	$P_i = C_{i-1} \oplus D_k(C_i)$	$P_i = C_i \oplus E_k(C_{i-1})$
反馈来源	前一分组的密文	前一分组的密文
用到解密函数吗	解密要用 D_k	两个方向都只用 E_k

6.4 OFB 模式

模式机制 (Output FeedBack, 输出反馈)

与 CFB 模式类似, 但其反馈机制独立于明文和密文, 因此可以将大部分工作于明文前 (即可以预先算出密钥流)。



模式分析: 一开始 S_0 用 IV 填充, 然后加密成 S'_0 , 将 S'_0 和 P_1 的低 8 位异或, 形成密文的 8 位, 并将 S'_0 的 8 位加入到 S_0 的序列中, 重复此过程, 直到 P_1 加密完毕, 形成 C_1 。

n 位 OFB 模式

明文分组	P1	P2	...	P _{i-1}	P _i	...
IV	E _k	⊕		⊕	⊕	
		E _k		E _k	E _k	
密文分组	C1	C2	...	C _{i-1}	C _i	...

加密: $C_i = P_i \oplus S_i, S_i = E_k(S_{i-1})$

解密: $P_i = C_i \oplus S_i, S_i = E_k(S_{i-1})$

CFB 与 OFB 的关键区别

CFB: 反馈量来自密文 (C_{i-1}) $\Rightarrow$ 必须收到密文才能算下一段, 不能预计算。

OFB: 反馈量独立于明文和密文（只和密钥、IV 有关） $\Rightarrow$ 可以提前把整条密钥流算出来，等明文来了直接异或即可。

6.5 其他模式

分组链接模式 BC (Block Chaining)

简单地将分组密码算法的输入与前面所有密文分组的异或值相异或。

加密: $C_i = E_k(P_i \oplus F_i)$ $F_{i+1} = F_i \oplus C_i$
解密: $P_i = F_i \oplus D_k(C_i)$ $F_{i+1} = F_i \oplus C_i$
其中: $F_1 = IV$

扩散分组密码链接模式 PCBC

与 CBC 模式不同的是，在加密前将前面的明文分组、密文分组及当前明文分组相异或。

明文	P_{i-1}	P_i	P_{i+1}
	$\oplus$	$\oplus$	$\oplus$
	$\oplus$	$\oplus$	$\oplus$
	E_k	E_k	E_k
密文	C_{i-1}	C_i	C_{i+1}

加密: $C_i = E_k(P_i \oplus C_{i-1} \oplus P_{i-1})$
解密: $P_i = C_{i-1} \oplus P_{i-1} \oplus D_k(C_i)$

PCBC 模式被用在 Kerberos 4 中，但 Kerberos 5 发现了其存在问题，用 CBC 模式取代。

PCBC 的问题：交换两个密文分组，错误将抵消

原因：由于明文和密文异或的性质。

原文本: $P_1 P_2 P_3 P_4$ 原密文: $C_1 C_2 C_3 C_4$

$C_2 = E_k(P_2 \oplus C_1 \oplus P_1)$
 $C_3 = E_k(P_3 \oplus C_2 \oplus P_2)$
 $C_4 = E_k(P_4 \oplus C_3 \oplus P_3)$

【攻击】把中间两个密文分组交换：

新明文: $P_1' = P_1$ $P_2' = P_2$ $P_3' = P_3$ $P_4' = P_4$
新密文: $C_1' = C_1$ $C_2' = C_3$ $C_3' = C_2$ $C_4' = C_4$

解密：

$P_2' = C_1' \oplus P_1' \oplus D_k(C_2') = C_1 \oplus P_1 \oplus P_3 \oplus C_2 \oplus P_2$
 $P_3' = C_2' \oplus P_2' \oplus D_k(C_3')$
 $\quad = C_3 \oplus C_1 \oplus P_1 \oplus P_3 \oplus C_2 \oplus P_2 \oplus P_2 \oplus C_1 \oplus P_1$
 $\quad = C_3 \oplus P_3 \oplus C_2$
 $P_4' = C_3' \oplus P_3' \oplus D_k(C_4')$
 $\quad = C_2 \oplus C_3 \oplus P_3 \oplus C_2 \oplus P_4 \oplus C_3 \oplus P_3 = P_4$ (正确!)

看出来了吗：交换 C_2 和 C_3 之后， P_4 居然完好无损（错误在异或中相互抵消），攻击者可以借此愚弄接收方认为收到的是 P_4 ，而实际上中间内容已被改动。

六种模式总览表

模式	加密公式	解密公式	特点
----	------	------	----

ECB	$C_i = E_k(P_i)$	$P_i = D_k(C_i)$	分组独立；随机存取方便；易受重放攻击
CBC	$C_i = E_k(P_i \oplus C_{i-1})$	$P_i = C_{i-1} \oplus D_k(C_i)$	需要 IV；IV 可明文传输
CFB	$C_i = P_i \oplus E_k(C_{i-1})$	$P_i = C_i \oplus E_k(C_{i-1})$	反馈来自密文；不能预计算
OFB	$C_i = P_i \oplus S_i, S_i = E_k(S_{i-1})$	$P_i = C_i \oplus S_i, S_i = E_k(S_{i-1})$	反馈独立于明文密文；可预计算密钥流
BC	$C_i = E_k(P_i \oplus F_i), F_{i+1} = F_i \oplus C_i$	$P_i = F_i \oplus D_k(C_i)$	$F_1 = IV$ ；与前面所有密文异或
PCBC	$C_i = E_k(P_i \oplus C_{i-1} \oplus P_{i-1})$	$P_i = C_{i-1} \oplus P_{i-1} \oplus D_k(C_i)$	Kerberos4 用过；交换密文分组错误会抵消

第 7 章 密码管理

7.1 密钥生命周期

密钥管理概述

密码系统 = 明文、密文、加解密算法、**密钥**、密钥空间。

思考：密钥如此重要，如何管理？

密码系统的安全性

安全性来自三方面：**密码算法**、**密钥空间**。

- **对称密码**：靠算法强度
- **公钥密码**：靠数学难题（因子分解、离散对数、单向陷门）
- **密钥空间**：靠**穷举攻击**的难度（即密钥长度）

两个权衡：

- 数学难题的困难性 —— 数论和复杂度理论的发展使其改变
- 密钥长度与代价之间的权衡 —— 破译的明文消息的价值和破译所花的代价之间的权衡

密钥生命周期的组成（必背）

密钥生成 → 密钥存储 → 密钥传输 → 密钥使用 → 密钥更新 → 密钥销毁

阶段	性能需求
密钥生成	有效性
密钥存储	安全性
密钥传输	可靠性
密钥使用	多样性
密钥更新	高效性
密钥销毁	彻底性

密钥生成 —— 普通密钥

口令选择与攻击

普通密钥泛指人们生活中的各种登录口令。

口令选择：人们选择自己的密钥时，通常选择一个**弱密钥**，如选择的是 zs111 而不是 *9(hH/A。

字典攻击：根据用户信息尝试最可能的密钥。

- 用户姓名、电话、身份证、帐户等有关信息
- 各种单词的不同置换形式，如大小写、简写

撞库攻击：攻击者先通过收集互联网已泄露的用户+密码信息，生成对应的字典表；再用字典中罗列的用户和密码，尝试批量登录其他网站。

拖库攻击：攻击者入侵有价值的网站，把注册用户的资料数据库全部盗走的行为。（互联网各种信息泄露事件）

安全口令的建议

密钥既容易记忆，又难以被猜中。

建议：

- 加入标点符号
- 由较长的短语的首字母组成字母串
- 根据安全级别设置 3 个左右的口令

Stand high and see far. → ^Shasf?520^
Chuan guo shui wu hen. → ^Cgsw#118^

最高级别：网银等重要密码
中等级别：QQ、微信、重要邮箱等登录密码
普通级别：临时注册使用的一次性密码

密钥生成 —— 随机密钥

PRNG

随机密钥一般应用于各种网络通信中。

任意由确定过程生成的随机数序列，从实际意义上讲都不是随机的。

伪随机数发生器 PRNG (pseudo-random number generators)：随机数产生的周期足够长。

- 线性同余发生器（见 2.2.2）
- 反馈移位寄存器（见 2.2.3）

[参考] Paul Garrett 著, 吴世忠等译. 密码学导引. 机械工业出版社

其余各阶段

阶段	要点
密钥的存储与备份	如何保存密钥？如何避免被窃取？⇒ 密钥托管、门限方案
密钥的传输与验证	如何保证密钥传输的安全和可靠性？⇒ 密钥交换、密钥分发
密钥使用	密钥使用的期限：长期密钥与会话密钥；通信中使用密钥交换协议——会话密钥不保存
密钥更新	考虑密钥的安全性，不能长期高频率地使用一个密钥；从旧密钥中产生新的密钥，如用 Hash 函数
密钥销毁	物理的：粉碎载体；软件的：采用特殊的删除程序

7.1 小结

- 密钥的生命周期：密钥生成 → 存储 → 传输 → 使用 → 更新 → 销毁；密钥每个阶段的性能需求。
- 产生安全的密钥：安全口令 —— 密钥既容易记忆，又难以被猜中；随机密钥 —— 保证随机的周期足够长。

7.2 对称密钥管理 (Kerberos)

Kerberos 概述

Kerberos：应用于 TCP/IP 网络的可信第三方鉴别协议。

最初是在麻省理工学院为 Athena 项目开发的模型，基于对称密码。

Kerberos 有一个保存所有客户及其共享密钥的数据库，能：

- 产生消息向一个实体证实另一个实体的身份
- 产生会话密钥，加密双方之间的通信

两类凭证

① 票据 (Ticket Granting Ticket, TGT)：用作向服务器提供身份鉴别，包括客户名、服务器名、网络地址、时间标记和会话密钥。

$$T_{c,tgs} = \{ c, tgs, addr, realm, t, K_{c,tgs} \}$$

② 鉴别码 (Authenticator)：客户每次需要使用服务器上的服务时，都要产生一个鉴别码，包括用户名、时间标记和网络范围。

$$A_{c,s} = \{ c, s, realm, t \}$$

密钥管理过程 (六步协议，必背)

两个服务器：AS（身份认证服务器）、TGS（票据发放服务器）

- ① Client → AS :

c, tgs, n
- ② AS → Client :

{Kc,tgs, n}Kc , {Tc,tgs}Ktgs
- ③ Client → TGS:

{Ac,s}Kc,tgs, {Tc,tgs}Ktgs , s, n
- ④ TGS → Client :

{Kc,s, n}Kc,tgs , {Tc,s}Ks
- ⑤ Client → Server:

{Ac,s}Kc,s , {Tc,s}Ks
- ⑥ Server → Client:

{m, t, k}Kc,s

三步概括：

1. 获得票据许可票据 TGT（Ticket Granting Ticket）—— 第 ①② 步
2. 获得服务票据 ST（Service Ticket）—— 第 ③④ 步
3. 请求服务（双向验证）—— 第 ⑤⑥ 步

思考：几个密钥？如何分配与验证？

涉及的密钥： K_c （客户与 AS 的长期密钥）、 K_{tgs} （TGS 的长期密钥）、 $K_{c,tgs}$ （客户与 TGS 的会话密钥）、 K_s （服务器的长期密钥）、 $K_{c,s}$ （客户与服务间的会话密钥）——会话密钥用完即弃，长期密钥事先分配。

7.3 公钥管理与数字证书

引入

使用不同浏览器访问银行网站，可能出现两种不同的结果。

思考：为什么出现两种不同的结果？（一种是受信任的证书 ⇒ 安全连接；一种是证书不受信任 ⇒ 警告）

网络中公钥的获得（三条途径）：

- 从对方处获得
- 从可信的中央数据库获得
- 从自己的私人数据库获得

问题：如何保证可靠性？⇒ 避免中间人攻击⇒ 数字证书

数字证书概念与定义

概念：公钥证书 ⇒ 身份鉴别（像“身份证”、“名片”）。

定义：简称证书，由可信的证书机关 CA 将公钥的值绑定到持有对应私钥的个人、设备或服务的身份。

X.509 标准

- ISO 鉴别框架：X.509 标准，于 1988 年公布
- 国际电信联盟-电信（ITU-T）部分标准和国际标准化组织（ISO）的证书格式标准
- 定义了公开密钥的证书结构
- 是 PKI（公钥基础设施）的重要组成部分

数字证书的结构（V1~V3）

字段	说明
版本	V1~V3
用于签名的算法	签发者签名所用的算法
序列号	Internet 中唯一
颁发者信息	包括通用名、组织单位、组织和国家
有效期	生效与失效时间
证书所有者信息（公钥）	持有者身份 + 公钥值
扩展项	V2 中引入，解决重复名称问题
扩展 / 密钥用途等	V3 中引入

颁发者的签名

保证证书的完整性（防篡改）

举例：访问银行网站

用户 — 启用连接 —→ 服务器
←— 发送银行证书 —
检查本机证书存储区
若证书颁发机构是可信任的，则存储银行证书
←— 建立安全会话 —

数字证书的发布：证书链

三个问题：通过证书对身份的信任程度？谁可以作为唯一可信实体？证书链可以有多长？

答案：一个唯一可信任的实体 → 可信任代理机构 CA。

- 主机将颁发者证书放到主机的受信任根证书颁发机构的证书存储区，将其指定为受信任的根颁发机构。
- 中间的或从属的证书颁发机构受到信任的条件是：他们拥有受信任根证书颁发机构的有效证书路径。

7.3 小结

- 数字证书的结构：颁发者、使用者的信息（公钥）、颁发者的签名……
- 数字证书的作用：用来为实现安全信息交换建立身份并创建信任。
- 数字证书的应用：受信任根证书的存储和中间证书的信任条件。

课件作业：① 在设置各种登录口令时，为什么不能简单地使用一个口令？给出理由。② 在自己主机上进行证书的存储、导入导出等相关操作，提交操作报告。③ 查阅资料了解网络撞库攻击的现状。④ 查阅资料了解不同行业的证书权威机构。

结语：密码学及应用的应用场景 —— (1) 保密通信 (2) 访问控制 (3) 隐私保护 (4) 安全支付……

A. 知识点索引（速查）

按关键词字顺排列，右边的章节号可直接点击跳转。

知识点 / 关键词	所在章节	一句话要点
AES	2.1.3	128 位分组，密钥 128/192/256，10/12/14 轮，4 个子步骤， $GF(2^8)$ 多项式运算
Asmuth-Bloom 门限	3.6	用 CRT 恢复； $a_i = x \bmod m_i$
BC 模式	6.5	$C_i = E_k(P_i \oplus F_i)$ ， $F_{i+1} = F_i \oplus C_i$
CBC 模式	6.2	$C_i = E_k(P_i \oplus C_{i-1})$ ；IV 可明文传输
CFB 模式	6.3	$C_i = P_i \oplus E_k(C_{i-1})$ ；反馈在分组内与分组间
CRT 中国剩余定理	2.4.1 / 4.3	$x \equiv \sum a_i T_i W_i \pmod{M_n}$ ；用于密钥分发可把 $O(k)$ 降到 $O(1)$
DES	2.1.1	64 位分组，56 位密钥，16 轮，混淆与扩散
DH 密钥协商	3.3.2	共享密钥 $g^{xy} \bmod p$ ；易受中间人攻击
DSA 签名	2.5.2	$r = (g^k \bmod p) \bmod q$ ； $s = k^{-1}(H(m) + dr) \bmod q$
ECC	2.4.4	$y^2 = x^3 + ax + b$ ； $C_m = \{kG, P_m + kE\}$
ECB 模式	6.1	$C_i = E_k(P_i)$ ；易受重放攻击
EKE 协议	3.3.3	用共享密钥加密公钥，安全性较好
ElGamal	2.4.3	密文对 $(g^k, m \cdot e^k)$ ；同明文可不同密文
Euclid 算法	1.3	辗转相除 + 回代，求最大公因子与乘法逆元
IV 初始向量	6.2	把每个消息唯一化；不需保密，可明文传输
Kasiski 方法	1.3	量重复片段距离，公约数 = 密钥长度
Kerberos	7.2	AS + TGS，TGT 与 Authenticator，六步协议
Knapsack	2.4.5	超递增背包，已证明不安全
LFSR	2.2.3	移位寄存器 + 反馈函数；最大周期 $2^n - 1$
MD5	2.3.1	128 位输出，5 个步骤，4 轮 16 次
MIX 机制	5.5	加密/填充改外观，延迟/重排序改顺序
OFB 模式	6.4	$C_i = P_i \oplus S_i$ ， $S_i = E_k(S_{i-1})$ ；可预计算
OTP 一次一密	1.3	密钥与明文等长、只用一次；理论安全
PCBC 模式	6.5	Kerberos4 用过；交换密文分组错误会抵消
Pollard's Rho / P-1	2.4.2	针对 RSA 的因子分解攻击
RC4	2.2.1	256 字节 S 盒；字节流
RSA	2.4.2	$c = m^e \bmod n$ ；安全性基于因子分解
RSA 签名	2.5.1	$s = m^d \bmod n$ ；验证 $s^e = m$
SHA 系列	2.3.2	SHA1-160、SHA256-256；512 位分组 4 轮 80 次
Shamir 门限方案	3.6	$k-1$ 次多项式， $D_i = f(i)$ ， $m = f(0)$
Shamir 三次传输	4.1	重重加密依次解密；加解密顺序可交换
SKEY 协议	3.5	单向 hash 链反向使用；抗重放、省存储
SM 系列	2.6	SM1/2/3/4/9
Schnorr 签名	2.5.2	$x = a^k$ ， $r = H(m, x)$ ， $s = (k + dr) \bmod q$
Vigenère	1.3	多表代替；密钥空间 26^m ；有周期性弱点
X.509	7.3	1988 年公布；PKI 重要组成部分

安全服务（五类）	1.1	认证、访问控制、保密性、完整性、抗否认性
安全机制（八类）	1.1	加密、数字签名、认证、完整性、访问控制、路由控制、业务流填充、公证
攻击类型（四种）	1.2	唯密文、已知明文、选择明文、针对密钥
比特承诺	4.4	r_1 防篡改、 r_2 防猜测（hash）； r 防篡改、 k 做验证（对称）
保密通信协议	3.2	四种设计：古典/对称/公钥/混合
并集基数 / lcm	1.4	置换的阶 = 各不相交循环长度的最小公倍数
不经意传输 OT	4.1	传两个中的一个，发送方不知对方收到哪个
乘法逆元	1.3	$mx \equiv 1 \pmod n$ ；存在条件 $(n,m)=1$
匿名通信	5.5	匿名性、不可关联性、不可观察性；代理/MIX/Crowds/广播
匿名投票	5.4	盲签名切断投票者与选票的对应关系
盲签名	5.2	$m'=m \times b$ ；签名者不知内容；分割选择法
欧拉定理	1.4	$(x,n)=1 \implies x^{\varphi(n)} \equiv 1 \pmod n$ ；用于降指数
欧拉函数	1.4	$\varphi(p)=p-1$ ； $\varphi(p^e)=(p-1)p^{e-1}$
费马小定理	1.4	$x^{p-1} \equiv 1 \pmod p$ ；欧拉定理的素数特例
仿射密码	1.3	$y=(ax+b)\%26$ ；密钥空间 311； a 必须与 26 互素
分区 / 剩余类	1.4	模 m 的剩余类有 m 个
公钥密码 vs 对称密码	1.2	前者密钥管理简单、慢；后者快、密钥管理复杂
哈希函数性质	2.3	公开性、单向性、扩散性（雪崩）、抗碰撞性
区块链	5.6	POW/POS/DPOS； $h(\text{Nonce} \parallel \text{Prev_Hash} \parallel \text{Tx}) < \text{Bits}$
换位密码	1.3	列换位（按行填按列读）、矩阵换位（置换矩阵）
混合密码 / 数字信封	3.2	公钥保护会话密钥，会话密钥加密消息
基数 / 群	1.4	封闭、结合、单位元、逆元；加交换律为交换群
阶	1.4	回 1 的最少步数；阶整除 $\varphi(n)$
密钥生命周期	7.1	生成 → 存储 → 传输 → 使用 → 更新 → 销毁
快速指数算法	1.4	三元组 (X,E,Y) ；奇减一，偶平分
离散对数 DLP	2.4.1	$g^t \equiv x$ ；由 g,t,n 算 x 易，反向难
门限方案	3.6	(k,n) ： $\geq k$ 份可恢复， $< k$ 份不可
密钥分发（CRT）	4.3	$R \equiv ks \pmod{k_i}$ ；无权者 $R=0$ ； $O(1)$
密钥保管 / 密钥托管	3.6	DH 密钥托管；门限分割 + 托管备案
零知识证明	5.1	洞穴协议；重复 n 次欺骗概率趋 0
离散对数 / BSGS / Pollard	2.4.1	两支求离散对数的算法
对称密码算法对比	2.1	DES 64/56/16；IDEA 64/128/8；AES 128/(128,192,256)/(10,12,14)
素性检测	2.4.1	Fermat / Lehmann / Solovay-Strassen / Miller-Rabin
素数与生成	2.4.1	梅森素数 2^n-1 ；费马素数；伪素数；卡迈克尔数
同余	1.3	$x \equiv y \pmod m \iff m (x-y)$ ；6 条定理
同态加密	4.2	$D(E(a) \otimes E(b)) = a \otimes b$ ；乘法/加法/全同态
同时签约	5.3	带仲裁者 / 概率递增 / 密码技术
网络攻击（口令）	3.5 / 7.1	攻击数据库、网络窃听、重放攻击；字典/撞库/拖库
位承诺协议	4.4	单向函数 / 对称密码两种方案
洗牌	1.4	完美快速洗牌 $f(x)=(2x) \bmod (2n+1)$
序列密码	2.2	明文 $\oplus$ 密钥流 = 密文；密钥流不能重复
姚氏百万富翁问题	4.2	构造位置关系，加密比较大小
一次同余式	2.4.1	$ax \equiv y \pmod m$ 有解 $\iff d y$ ；解的个数是 d

移位密码 / 凯撒	1.3	$y=(x+k)\%26$; 密钥空间 25
有限域 $GF(2^8)$	2.1.3	AES 的模多项式 $x^8+x^4+x^3+x+1$ (100011011)
原根	1.4	阶 = $\varphi(n)$; $g^1 \dots g^{\varphi(n)}$ 组成 Z_n^*
阈下信道	3.4	基于 ElGamal 签名隐藏秘密 k
置换群	1.4	不相交循环; 阶 = lcm
中间人攻击	3.2	交换公钥时被替换; 靠数字证书防范
中国剩余定理	2.4.1	孙子定理; 物不知数
重放攻击	3.5 / 6.1	口令场景靠动态口令/SKEY; ECB 场景靠分组链接
综合: 加密 vs 签名密钥对	2.5.4	保密通信的密钥对不同于数字签名的密钥对
数字证书	7.3	CA 绑定公钥与身份; 证书链与受信任根
数字摘要 / MAC	2.3	保完整性; 提高签名效率; 与密钥有关的散列是 MAC
散列函数	2.3	任意长 $\rightarrow$ 固定长; 数字指纹
哈希链	3.5	$x_{i+1}=h(x_i)$; 正易反难
零知识证明	5.1	证明拥有秘密而不泄露秘密

B. 课件习题总汇

B1. 第 1 章小结 · 知识点巩固 (10 题)

题目

- 求 -200 模 98 的约简。
- 已知明文攻击: $E_{a,b}(7)=1$ 且 $E_{a,b}(12)=4$, 求解密密钥。
- 集合 Z_{29}^* 中的元素个数是多少?
- 写出模 9 的最小正完全剩余系和最小正简化剩余系。
- 计算 $249^{197}\%199$ 。
- 计算 $\varphi(18)$, $\varphi(155)$ 。
- 求 101 模 234 的乘法逆。
- 验证 3 是否是模 41 的一个本原根。
- S_5 中任意元素的最大阶是多少? S_{10} 呢?
- 确定对一副 12 张牌的完美快速洗牌的循环分解。

参考解答

- $-200 + 3 \times 98 = 94$, 且 $0 \leq 94 < 98 \Rightarrow 94$
- 两式相减: $5a \equiv 3 \pmod{26}$; $5^{-1} = 21 \Rightarrow a = 63 \equiv 11$;
 $b = (1 - 7 \times 11) \% 26 = -76 \equiv 2 \Rightarrow$ 加密密钥 (11, 2);
 $11^{-1} = 19$; $-19 \times 2 = -38 \equiv 14 \Rightarrow$ 解密密钥 (19, 14), 即 $x = (19y+14)\%26$
- 29 是素数 $\Rightarrow \varphi(29) = 29 - 1 = 28$
- 最小正完全剩余系 {1,2,3,4,5,6,7,8,9};
最小正简化剩余系 (排除 3 的倍数) {1,2,4,5,7,8} ($\varphi(9)=6 \checkmark$)
- 199 为素数, $249^{198} \equiv 1$; $249^{197} \equiv 249^{-1}$; $249 \equiv 50$;
Euclid: $1 = 4 \times 50 - 199 \Rightarrow 50^{-1} = 4 \Rightarrow 4$
- $18 = 2 \times 3^2 \Rightarrow (2-1) \times 2^0 \times (3-1) \times 3^1 = 1 \times 1 \times 2 \times 3 = 6$;
 $155 = 5 \times 31 \Rightarrow 4 \times 30 = 120$
- Euclid (见模板 1) $\Rightarrow 101^{-1} \% 234 = 95$ (验算 $101 \times 95 = 9595 = 41 \times 234 + 1 \checkmark$)
- $\varphi(41) = 40 = 2^3 \times 5$, 素因子 2、5; 测 3^{20} 、 3^8 :
 $3^4 = 81 \equiv -1$; $3^8 \equiv 1 \Rightarrow 3$ 不是原根 ($\text{ord}_{41}(3) = 8, 8 \mid 40$)
- S_5 : 最优拆分 $3+2 \Rightarrow \text{lcm}(3,2) = 6$;
 S_{10} : 最优拆分 $5+3+2 \Rightarrow \text{lcm}(5,3,2) = 30$

10. $2n=12$, 模数 13, $f(x)=(2x) \bmod 13$:
 $1 \rightarrow 2 \rightarrow 4 \rightarrow 8 \rightarrow 3 \rightarrow 6 \rightarrow 12 \rightarrow 11 \rightarrow 9 \rightarrow 5 \rightarrow 10 \rightarrow 7 \rightarrow 1$
 $\Rightarrow (1\ 2\ 4\ 8\ 3\ 6\ 12\ 11\ 9\ 5\ 10\ 7)$, 12 阶循环, 洗 12 次复原

B2. 第 2 章 2.4.1 数学知识 · 知识点巩固 (9 题)

题目

- 用穷举法计算以 3 为底模 25 的 4 的离散对数。
- 利用 Baby-step 算法在 Z_{29}^* 中求 $\log_2 7$ 。
- 求一次同余式 $8x \equiv 4 \pmod{12}$ 的所有解。
- 求 2 模 47 的平方根。
- 求一个整数 x , 满足同余方程 $2x \equiv 1 \pmod{3}$, $3x \equiv 2 \pmod{5}$ 。
- 判断 $5x^2 \equiv 3 \pmod{77}$ 是否有解, 有则求出所有解。
- 利用二次符号判断 2 是否为模 97 的一个平方。
- 判断方程 $x^2 \equiv 83 \pmod{107}$ 是否有解。
- 判定 3 是否为 2009 的一个平方。

提示: 第 3 题 $d = (8, 12) = 4$, $4 \mid 4 \Rightarrow$ 有 4 个解; 第 9 题先分解 $2009 = 7^2 \times 41$ 。

B3. 各章"复习与作业"汇总 (课件原文)

位置	题目
第 1 章	理解: 设计一个简单的用于与家人通信的密码算法。 分析: 分析对称密码算法是否可用于数字签名? 拓展: 查阅资料, 综述密码算法的最新进展。 预习思考: 为什么对称密码比公钥密码运算速度快? 为什么说公钥很难推导出私钥?
2.1 对称密码	理解: 选择一种熟悉的编程语言, 实现 AES 算法。 分析: 分析 AES 算法中的雪崩效应。 拓展: 查阅资料, 综述三种用到 AES 算法的软件。
2.2 序列密码	理解: 编程实现 RC4 算法密钥流的生成。 计算: 同余发生器 $x_{n+1} = 6x_n + 9 \bmod 11$ 的坏种子。 拓展: 查阅资料, 了解 RC4 可能的替代算法。
2.3 数字摘要	理解: 编程实现 SHA256 算法。 分析: 基于散列函数的特点分析数字摘要的防篡改性。 拓展: 查阅资料, 综述消息鉴别码的原理与应用。
2.4 公钥密码	分析: 分析 RSA 算法中公钥选择较小素数的原因。 拓展: 查阅资料, 综述密码分析中的因子分解算法或离散对数计算方法。
2.5 数字签名	理解: 既然 RSA 签名算法存在指数运算攻击, 为什么仍然应用广泛? 分析: 分析 DSA 算法中私钥 d 泄露的后果。 拓展: 查阅资料, 综述数字签名算法的最近研究进展。
3.2 保密通信	理解: 设计一个简单的与家人通信的保密通信协议? 分析: 分析四种保密通信协议的实际应用场景。 拓展: 查阅资料, 综述 3 个保密通信协议的应用例子。
3.3 密钥协商	理解: 为什么说 EKE 协议的安全性较好? 分析: 分析 Diffie-Hellman 协议的安全性。 拓展: 查阅资料, 综述无线网络中的密钥协商协议。
3.4 数字签名协议	理解: 写出基于对称密码的数字签名协议的符号描述, 并分析签名的性质。 设计: 利用混合密码协议和哈希+公钥的数字签名协议优化带加密的数字签名协议。 拓展: 查阅资料, 了解数字签名协议的某个应用。
3.5 身份鉴别	理解: SKEY 协议中单向 hash 链的使用是反向的, 如果采用正向使用是否可行, 安全性如何? 拓展: 查阅资料, 综述单向 hash 链的一个应用。
3.6 秘密共享	理解: 已知 $m_1=3, m_2=5, m_3=7, m_4=8$, 求消息 $x=60$ 的 (3,4) 门限方案的秘密份额。 拓展: 查阅资料, 综述生活中一个门限方案的应用。

第 4 章	理解： 匿名密钥分配协议中能否采用对称密码算法？ 分析： 分析 Shamir 三次传输协议中的加密算法如果采用异或运算导致的安全隐患。 拓展： 查阅资料，综述不经意传输协议的一个应用。 预习思考： 广播环境下如何构建高性能的密钥分发协议？
第 6 章	理解： CBC 模式中为什么 IV 可以明文传输？ 分析： 分析 CBC 模式中的错误传播。
第 7 章	理解： 在设置各种登录口令时，为什么不能简单地使用一个口令？给出理由。在自己主机上进行证书的存储、导入导出等相关操作，提交操作报告。 拓展： 查阅资料，了解网络撞库攻击的现状；查阅资料，了解不同行业的证书权威机构。

B4. 课件"课内作业 / 练习"汇总

位置	题目	答案 / 提示
1.3 乘法逆元	求 $123^{-1} \% 421$	用 Euclid 模板
1.4 快速指数	计算 $2^{1000} \bmod 89$	三元组法
1.4 快速指数	计算 $2^{570} \% 59$	三元组法
1.4 模 n 的二次根	求 3 模 107 的主二次根	$107 \equiv 3 \pmod{4} \implies x = \pm 3^{(107+1)/4} \bmod 107$
2.4.1 二次符号	判断 $x^2 \equiv 137 \pmod{227}$ 是否有解	算 $(137/227)$
2.4.1 离散对数	在 Z_{227}^* 中计算 $\log_2 11$	BSGS
2.4.1 离散对数	在 Z_{59}^* 中计算 $\log_4 9$	Pollard's Rho
2.2.2 同余	计算 $x_{n+1} = 6x_n + 9 \bmod 11$ 的坏种子	$s_{\text{bad}} = 7$
2.2.3 LFSR	分析抽头序列 (4,1,0) 和 (4,2,1,0) 的周期	15 和 6
3.6 门限	已知 $m_1=3, m_2=5, m_3=7, m_4=8$ ，求 $x=60$ 的 (3,4) 门限方案份额	$a_i = 60 \bmod m_i$

C. 全课对比总表

C1. 密码算法总对比

算法	类型	分组/密钥	轮数	安全性基础
DES	对称分组	64 / 56	16	S 盒、混淆扩散
IDEA	对称分组	64 / 128	8	乘加结构
AES	对称分组	128 / 128,192,256	10,12,14	$GF(2^8)$ 多项式
SM4	对称分组（国密）	128 / 128	32	非平衡 Feistel
RC4	对称序列	— / 可变	—	256 字节 S 盒
RSA	公钥	— / ≥ 1024	—	因子分解难题
ElGamal	公钥	— / $p > 10^{150}$	—	离散对数难题
ECC	公钥	— / 160($\approx$ RSA1024)	—	椭圆曲线离散对数
Knapsack	公钥（已破）	—	—	子集和问题
MD5	摘要	— / 输出 128	4 轮 $\times$ 16	已破（王小云 2004）
SHA1	摘要	— / 输出 160	4 轮 $\times$ 20	已破
SHA256	摘要	— / 输出 256	—	目前主流
SM3	摘要（国密）	512 / 输出 256	64 次	基于 SHA-256 改进

C2. 密码协议总对比

协议	核心做法	特点 / 局限
保密通信（对称）	引入 TTP，共享密钥	快；TTP 瓶颈；扩展性差
保密通信（公钥）	$E_{Bp}(m)$	密钥管理简单；慢；依赖公钥可靠
混合密码 / 数字信封	公钥保护会话密钥	又快又简单；实际最常用

DH 密钥协商	$g^{xy} \bmod p$	不需预分配；易受中间人攻击
EKE	共享密钥加密公钥	安全性较好（有验证步骤）
双线性对协商	$\hat{e}$ 的性质	计算代价低；用于无线网络
数字签名（对称）	$E_k(m) + TTP$	需可信第三方；小规模
数字签名（公钥）	$E_{Av}(m)$	可验证、不可伪造、不可抵赖
hash + 公钥签名	sha256RSA	效率高；扩展性好
SKEY	单向 hash 链反向使用	抗重放；服务器只存一个数
挑战应答	$H_k(R_A, R_B, B)$	动态、防空口
Shamir 门限	$k-1$ 次多项式 + 拉格朗日插值	$\geq k$ 份可恢复； $m=f(0)$
Asmuth-Bloom	$a_i = x \bmod m_i + CRT$	$\geq k$ 份可恢复
Shamir 三次传输	重重加密，依次解密	不需协商密钥；不防中间人
匿名密钥分配	双层加密	减弱对 KDC 依赖
不经意传输 OT	两把公钥 + 对称密钥	发送方不知对方收到哪个
安全多方计算	盲化数据，算完去盲	保密的前提下完成计算
密钥分发（CRT）	$R \equiv ks \bmod k_i$	$O(1)$ 取代 $O(k)$
位承诺	$H(r_1, r_2, b)$ 或 $E_k(r, b)$	机密性 + 不可否认性
零知识证明	交互式提问 n 次	证明拥有秘密而不泄露
盲签名	$m' = m \times b$	签名者不知内容
同时签约	概率递增 / 不经意传输	双方“同时”生效
匿名投票	盲签名 + CTF 加密传输	授权、防篡改、投一次、可计票、保密
匿名通信	代理 / MIX / Crowds / 广播	身份匿名、位置匿名、路由匿名
区块链	POW 记账 + 链式结构	去中心化、不可篡改、可溯源

C3. 分组密码模式总对比

模式	加密	解密	IV	优点	缺点
ECB	$C_i = E_k(P_i)$	$P_i = D_k(C_i)$	无	随机存取方便；错误不传播	易受重放攻击
CBC	$C_i = E_k(P_i \oplus C_{i-1})$	$P_i = C_{i-1} \oplus D_k(C_i)$	需要	相同明文块得到不同密文	错误会传播
CFB	$C_i = P_i \oplus E_k(C_{i-1})$	$P_i = C_i \oplus E_k(C_{i-1})$	作为 C_0	可逐位/逐字节处理	不能预计算
OFB	$C_i = P_i \oplus S_i$	$P_i = C_i \oplus S_i$	作为 S_0	可预计算密钥流	—
BC	$C_i = E_k(P_i \oplus F_i)$	$P_i = F_i \oplus D_k(C_i)$	$F_1 = IV$	与前面所有密文异或	—
PCBC	$C_i = E_k(P_i \oplus C_{i-1} \oplus P_{i-1})$	$P_i = C_{i-1} \oplus P_{i-1} \oplus D_k(C_i)$	需要	Kerberos4 使用	交换密文分组错误会抵消

使用建议

- 开卷考试时：先翻到"★ 考前速查"，找对应类型的模板照抄格式；再用左侧搜索框查具体概念。
- 平时学习时：按章节顺序读，每章末尾的"复习与作业"是课件原题，可当练习。
- 打印时：直接 Ctrl+P，侧栏会自动隐藏，每章自动分页。