# 密码编码技术原理与应用

何立宝

holybao@youquan.io

深圳市优权天成科技有限公司

2017 年 7 月 8 日

# 内容提要

# 背景知识



- 发展历程
- 古典密码
- 基本术语
- 密码体制

# 发展历程

1. 1949 年之前：古典密码
   - 密码学还不是科学，而是一门艺术
   - 密码算法的基本手段（替代 & 置换）出现，针对的是字符
   - **数据安全基于算法的保密**

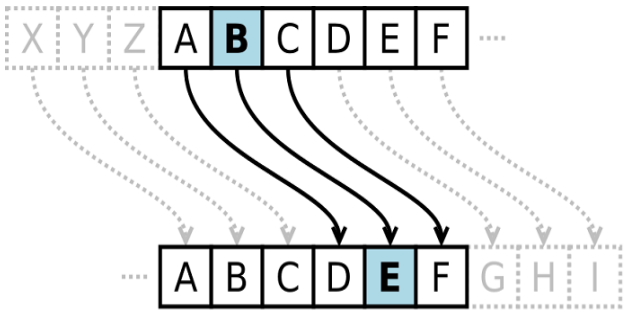



Figure 1.1: Caesar 密码

Figure 1.2: Sycatle 密码

# 发展历程

2. 1949 年 ~1975 年：密码学成为一门科学
   - 标志：1949 年 Shanon 发表 “The Communication Theory of Secret Systems”
   - 1967 年 David Kahn 的《The Codebreakers》
   - 1971~1973 年 IBM Watson 实验室的 Horst Feistel 等的几篇技术报告
     (1) J.L. Smith, *The Design of Lucifier: A Cryptographic Device for Data Communication*, 1971
     (2) J.L. Smith, W.A. Notz, and P.R. Osseck, *An Experimental Application of Cryptography to a Remotely Accessed Data System*, Auh. 1972
     (3) H. Feisstel, *Cryptography and Computer Privacy*, May 1973
   - Kerckhoffs 准则：数据安全基于密钥非算法的保密

# 发展历程

3. 1976 年后：密码学的新方向—公钥密码学
   - 标志：1976 年 Diffie 和 Hellman 的“New Directions in Cryptography”提出非对称密码
   - 1977 年 Rivest, Shamir & Adleman 提出了 RSA 公钥算法
   - 80 年代逐步出现离散对数、代数编码理论、椭圆曲线等其他公钥算法
   - **公钥密码解决了密钥分发难题，使得发送端和接收端不需要事先通过保密信道共享密钥**

# 古典密码

1. 替代密码：构造一个或多个密文字母表，然后用密文字母表中的字母或字母组来代替明文字母或字母组，各字母或字母组的相对位置不变，但其本身改变了
   - 单字母：明文的一个字符用相应的一个密文字符代替
     (1) 单表：任何密文可以看成由相应明文的各组信息单元使用同一个代替表进行替换而得
     (2) 多表：密文依次对相应明文的各组信息单元使用有限个周期性重复的或无限多的固定代替表进行替换而得
     (3) 多名：映射是一对多的，每个明文字母可以加密成多个密文字母
   - 多字母：明文的一个或多个字符用相应的多个密文字符代替
2. 置换密码：把明文中的字母重新排列，字母本身不变，但其位置改变了

# 单表替代密码

- 描述为可逆映射 $f:\{A \to B\}$, $f(a_i)=b_j$, $f^{-1}(b_j)=a_i$
  - 明文空间：$A=\{a_1,a_2,\ldots,a_m\}$
  - 密文空间：$B=\{b_1,b_2,\ldots,b_n\}$
  - 加密、解密变换：$f$、$f^{-1}$
  - 明文中单字母出现的频率分布与密文中相同
- Ex1. 移位密码
  - 加密变换：$\{E:Z26 \to Z26, E_k(m)=m+k \pmod{26}\}$
  - 解密变换：$\{D:Z26 \to Z26, D_k(c)=c-k \pmod{26}\}$
- Ex2. 仿射密码
  - 密钥空间：$K=\{(k_1,k_2)|k_1,k_2 \in Z26, \gcd(k_1,26)=1\}$
  - 加密变换：$E_k(m)=k_1m+k_2 \pmod{26}$
  - 解密变换：$D_k(c)=k_1(c-k_2) \pmod{26}$

# 多表替代密码

- 使用两个以上替代表，对明文成组替代，同一字母有不同密文，隐藏单字母出现的频率分布
- Ex1. Vigenère 密码：由偏移量不同的多个 Caesar 密码组成
  - 明文：$m = (m_1, m_2, \ldots, m_n)$
  - 密文：$c = (c_1, c_2, \ldots, c_n)$
  - 密钥：$k = (k_1, k_2, \ldots, k_n)$，周期性重复
  - 加密变换：$E_k(m) = (c_1, c_2, \cdots, c_n), c_i = m_i + k_i \pmod{26}$
  - 解密变换：$D_k(c) = (m_1, m_2, \cdots, m_n), m_i = c_i - k_i \pmod{26}$
- Ex2. 一次一密密码
  - 明文：$m = (m_1, m_2, \ldots, m_n, \cdots)$
  - 密文：$c = (c_1, c_2, \cdots, c_n, \cdots)$
  - 密钥：$k = (k_1, k_2, \cdots, k_n, \cdots)$
  - 加密、解密：$c = m \oplus k$、$m = c \oplus k$

# 多名替代密码

- 与单表替代密码类似，只是映射是一对多的，每个明文字母可以加密为多个字母
- Ex. 映射 $f : \{a, b, \ldots, z\} \to \{1, 2, \ldots, 100\}$
  - 加密变换
    - $a \to 1, 2, \ldots, 10$
    - $b \to 11, 12, \ldots, 20$
    - $\ldots$
    - $z \to 99, 100$
  - 解密变换
    - $1, 2, \ldots, 10 \to a$
    - $11, 12, \ldots, 20 \to b$
    - $\ldots$
    - $99, 100 \to z$

# 置换密码

- 置换密码，又称为换位密码
  - 简单换位，可由置换矩阵 $E_k$ 来表示，每个置换都有一个与之对应的逆置换 $D_k$
  - 仅有一个发送方和接受方知道的加密置换（用于加密）及对应的逆置换（用于解密）
  - 对明文 $L$ 长字母组中的字母位置进行重新排列，而每个字母本身并不改变
- 数学描述
  - 前提：明文 $m = (m_1, m_2, \ldots, m_n)$、置换矩阵决定的置换为 $\pi$
  - 加密置换：$c = E_k(m) = (c_1, \ldots, c_L) = m\pi(1), \ldots, m\pi(L)$
  - 解密置换：$m = D_k(c) = (m_1, \ldots, m_L) = c\pi^{-1}(1), \ldots, c\pi^{-1}(L)$

# 古典密码分析

1. 频率分析
   - 在任何一种书面语言中，不同的字母或字母组合出现的频率各不相同
   - 对于以这种语言书写的任意一段文本，都具有大致相同的特征字母分布
2. 穷举攻击：分组长度为 $L$ 的置换密码总共只有 $L!$ 种置换

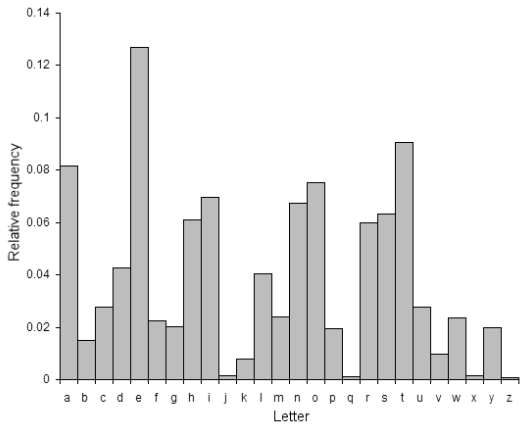



Figure 1.3: 英文字母统计分布

# 基本术语

- 密码学（Cryptology）
  - 密码编码学（Cryptography）：主要研究对信息进行编码，实现对信息的隐藏
  - 密码分析学（Cryptanalytics）：主要研究加密消息的破译或消息的伪造
- Cipher：密码
- 加密（Encipher）：将明文转换成密文的过程
- 解密（Decipher）：将密文还原成明文的过程
- 明文（Plaintext）：原始的可读数据
- 密文（Ciphertext）：加密后的不可解读数据
- 密钥（Key）：对加密与解密过程进行控制的参数
- $E(m)$：加密变换
- $D(c)$：解密变换

# 密码体制

- 密码体制：加密系统采用的基本工作方式
  - 基本要素是密码算法和密钥
    (1) 密码算法是一些公式、法则或程序
    (2) 密钥是密码算法的控制参数
  - 符号描述：$S = \{P, C, K, E, D\}$
    (1) $P$：明文空间
    (2) $C$：密文空间
    (3) $K$：密钥空间
    (4) $E$：加密变换
    (5) $D$：解密变换
    (6) $k \in K$
    (7) $C = E_{k_1}(P), P = D_{k_2}(C) = D_{k_2}(E_{k_1}(P))$
  - 作用：机密性、鉴别、完整性、抗抵赖

# 对称密码体制和非对称密码体制

**根据密钥的特征分类：**

1. 对称密码体制（Symmetric System, One-key System, Secret-key System）
   - 加密密钥和解密密钥相同，或者一个密钥可由另一个推导出
   - 加密能力和解密能力耦合，开放性差
2. 非对称密码体制（Asymmetric System, Two-key System, Public-key System）
   - 加密密钥和解密密钥不同，从一个密钥导出另外一个密钥是计算上不可行的
   - 加密能力和解密能力解耦，开放性好

# 序列密码体制和分组密码体制

**根据对明文消息的加密方式分类：**

1. 流密码（Stream Cipher）
   - 密文不仅与最初给定的算法和密钥有关，同时也与明文的位置有关（是所处位置的函数）
   - 加密以明文比特为单位，以伪随机序列与明文序列模 2 加后，作为密文序列
2. 分组密码（Block Cipher）
   - 经过加密所得到的密文仅与给定的密码算法和密钥有关，与被处理的明文数据在整个明文中的位置无关
   - 通常以大于等于 64 位的数据块为单位，加密得到相同长度的密文

# 密码体制其他分类

**确定型密码体制和概率密码体制：**

1. 确定型：当明文和密文确定后，密文也就唯一确定
2. 概率型：明文和密钥确定后，密文通过客观随机因素从一个密文集合中产生，密文形式不确定

**单向函数型密码体制和双向变换型密码体制：**

1. 单向函数型：适用于不需要解密的场合，容易将明文加密成密文，如哈希函数
2. 双向变换型：可以进行可逆的加密、解密变换

# 计算安全性和无条件安全性

1. 计算安全
   - 如果一个算法用约定期限内可得到的资源都不能破译，则被认为是计算上安全的
   - Ex. 公钥密码学方法基于安全性假设，而这种安全性假设在计算上是安全的，如大整数因式分解、离散对数假设等等
2. 无条件安全也称信息论安全
   - 不论密码分析者有多少密文，都没有足够的信息恢复出明文
   - Ex. 一次一密，依赖于随机序列发生器，理论上达到无条件安全

# 密码系统设计

- 基本方法
  - 扩散（Diffusion）、混淆（Confusion）
  - 目的：抵抗敌手对密码系统的统计分析
- 基本要求
  (1) 系统应该是实际上安全的，截获密文或已知明文-密文对时，要决定密钥或任意明文在计算上是不可行的
  (2) 加密解密算法适用于密钥空间中的所有元素
  (3) 符合 Kerckhoffs 原则：数据的安全依赖于密钥而非算法的保密
  (4) 系统易于实现，使用简单
  (5) 系统的使用不应使通信网络的效率过分降低

# 密码分析

1. 唯密文攻击（Ciphertext-only Attack）
   - 已知：$C_1 = E_K(P_1), \ldots, C_i = E_K(P_i)$
   - 推导：$P_1, \ldots, P_i$；$K$ 或从 $C_{i+1} = E_K(P_{i+1})$ 推导出 $P_{i+1}$
2. 已知明文攻击（Known-plaintext Attack）
   - 已知：$P_1, C_1 = E_K(P_1), \ldots, P_i, C_i = E_K(P_i)$
   - 推导：密钥 $K$，或从 $C_{i+1} = E_K(P_{i+1})$ 推导出 $P_{i+1}$ 的算法
3. 选择明文攻击（Chosen-plaintext Attack）
   - 已知：$P_1, C_1 = E_K(P_1), \ldots, P_i, C_i = E_K(P_i)$
   - 推导：密钥 $K$，或从 $C_{i+1} = E_K(P_{i+1})$ 推导出 $P_{i+1}$ 的算法
4. 选择密文攻击（Chosen-ciphertext Attack）
   - 已知：$C_1, P_1 = D_K(C_1), \ldots, P_i, C_i = D_K(C_i)$
   - 推导：$K$
5. 自适应选择明文攻击（Adaptive-chosen-plaintext Attack）

# 算法理论



- 安全层级模型
- 单向散列函数
- 对称加密算法
- 公钥加密算法
- 量子密码算法
- 后量子密码算法

# 安全层级模型

- 经典密码：基于图灵计算模型下的陷门单向函数，如：大整数因式分解、离散对数等
- 量子密码：基于量子力学的特性来加密
- 后量子密码：基于图灵计算模型、量子计算模型下的数学难解问题，如格密码

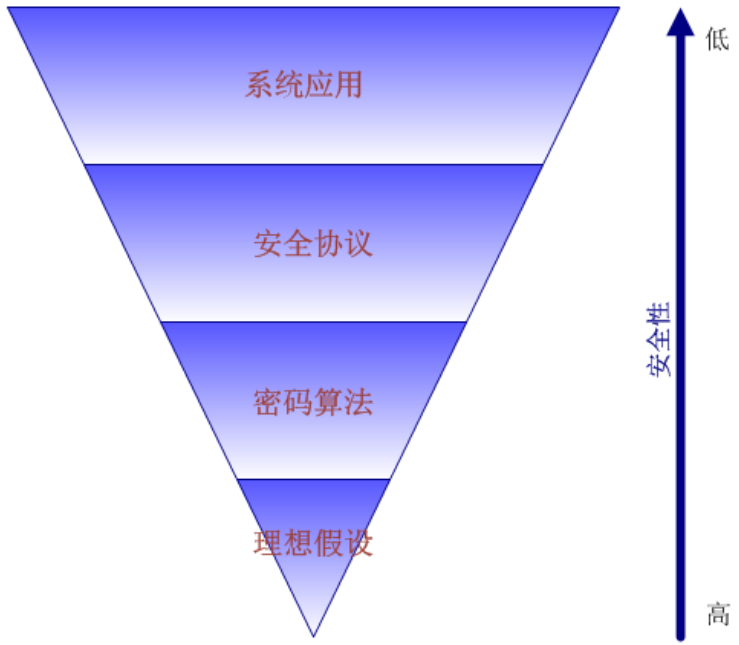



Figure 2.1: 安全层级模型

# 定义及特性

**定义 1 (单向函数，One-way Function)**

函数 $f : A \to B$ 称为单向函数，若它满足：

(1) 对所有 $x \in A$，易于计算 $f(x)$

(2) 对几乎所有 $x \in A$，由 $f(x)$ 求 $x$ 极为困难

- 别名：压缩函数、收缩函数、消息摘要、指纹、密码校验和、信息完整性校验、操作检验码
- 多到一映射：可变长度输入串转换为固定长度输出串
- 无冲突（Collision-free）：难于产生预映射的值，使它们的散列值相同
- 四大系列：MD、SHA、HMAC、CRC，用于数据完整性校验和数字签名

# 信息摘要算法-MD

- 消息摘要算法（Message Digest Algorithm）

| 算法 | 年份 | RFC | 输出散列值长度 (bits) | 中继散列值长度 (bits) | 数据区块长度 (bits) | 单字长度 (bits) | 最大输入信息长度 (bits) | 循环次数 | 碰撞攻击 |
|---|---|---|---|---|---|---|---|---|---|
| MD2 | 1982 | 1319 | 128 | 384 | 128 | 32 | 不限 | 864 | 是 |
| MD4 | 1982 | 1320 | 128 | 128 | 512 | 32 | $2^{64}-1$ | 48 | 是 |
| MD5 | 1991 | 1321 | 128 | 128 | 512 | 32 | $2^{64}-1$ | 64 | 是 |
| MD6 | 2008 | | $(0, 512]$ | 1024 | 512 | 64 | $2^{64}-1$ | $40+(d/4)$ | 尚无 |

# MD5 算法

- MD5 以 512 位分组来处理输入的信息，且每一分组又被划分为 16 个 32 位子分组，经过了一系列的处理后，算法的输出由四个 32 位分组组成，将这四个 32 位分组级联后将生成一个 128 位散列值

(1) 填充，加 1 补 0，确保字节长度模 512 为 448
(2) 附上 64-bit 的消息长度
(3) 初始化标准幻数
   - $A = 0x01234567$
   - $B = 0x89abcdef$
   - $C = 0xfedcba98$
   - $D = 0x76543210$
(4) 四轮主循环共 64 步操作

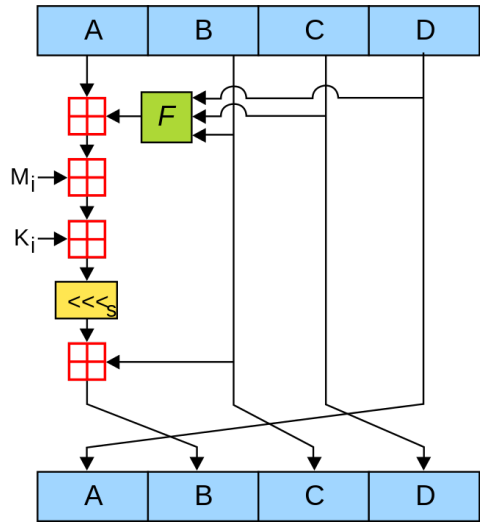


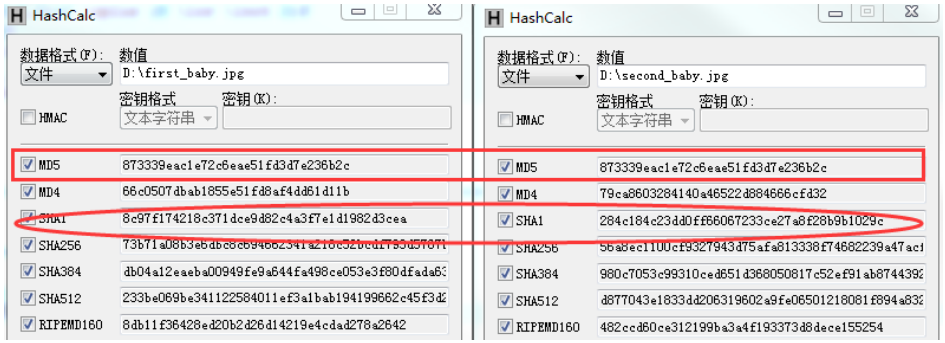

Figure 2.2: MD5 算法

# MD5 算法

- 主循环 $XX(a, b, c, d, M_j, s, t_i)$
  (1) $a = b + ((a + X(b, c, d) + M_j + t_i) \lll s)$，单轮 16 步操作
  (2) 每轮主循环的 $X$ 分别代表四个非线性函数 $F$、$G$、$H$、$I$
- 非线性函数
  (1) $F(B, C, D) = (B \wedge C) \vee (\neg B \wedge D)$
  (2) $G(B, C, D) = (B \wedge D) \vee (C \wedge \neg D)$
  (3) $H(B, C, D) = B \oplus C \oplus D$
  (4) $I(B, C, D) = C \oplus (B \vee \neg D)$
- $M_i$、$K_i$：32-bit 的输入信息块和常量
- $\lll_s$：左移 $s$ 位
- $\boxplus$：模 $2^{32}$ 加法

# MD5 碰撞攻击

- 2004 年发现完整 MD5 算法的碰撞
- 碰撞前缀构造法： http://www.win.tue.nl/hashclash/




```
D:\>fastcoll_v1.0.0.5.exe -p baby.jpg -o first_baby.jpg second_baby.jpg
MD5 collision generator v1.5
by Marc Stevens (http://www.win.tue.nl/hashclash/)

Using output filenames: 'first_baby.jpg' and 'second_baby.jpg'
Using prefixfile: 'baby.jpg'
Using initial value: 31df1f0f91cae06ff0d7cd5740f6eb0f

Generating first block: ...
Generating second block: S10.............................
Running time: 2.968 s
```

# 安全哈希算法-SHA

- 安全散列算法（Secure Hash Algorithm），美国国家安全局设计，美国国家标准与技术研究院发布
- SHA-1 在许多安全协议中使用，包括 SSL、PGP、SSH 等

Table 1: SHA 算法家族

| 算法 | 年份 | 输出散列值长度 (bits) | 中继散列值长度 (bits) | 数据区块长度 (bits) | 最大输入信息长度 (bits) | 单字长度 (bits) | 循环次数 | 碰撞攻击 |
|---|---|---|---|---|---|---|---|---|
| SHA-0 | 1993 | 160 | 160 | 512 | $2^{64}-1$ | 32 | 80 | 是 |
| SHA-1 | 1993 | 160 | 160 | 512 | $2^{64}-1$ | 32 | 80 | 是 [6] |
| SHA-256/224 | 2001 | 256/224 | 256 | 512 | $2^{64}-1$ | 32 | 64 | 尚无 |
| SHA-512/384 | 2001 | 512/384 | 512 | 1024 | $2^{128}-1$ | 64 | 80 | 尚无 |

# 消息认证码-HMAC

- 哈希运算消息认证码（Hash-based Message Authentication Code）
- 定义：$HMAC(K, m) = H((K \oplus opad)|H((K \oplus ipad)|m))$
  (1) $H$、$K$：散列函数、密钥
  (2) $m$：待认证消息
  (3) $\oplus$：异或运算
  (4) $opad$：$0x5c$ 重复 $N$ 次
  (5) $ipad$：$0x36$ 重复 $N$ 次
- 运算作用
  (1) 验证 TPM 接收的授权数据和认证数据
  (2) 确认 TPM 接收到的明清请求是已授权请求，并且命令在传输过程中未被篡改

# 循环冗余校验码-CRC

- 循环冗余校验码（Cyclic Redundancy Check）
  - 数据通信领域中常用的一种差错校验码
  - 碰撞概率较高，无法检出所有错误，但计算高效

Table 2: CRC 算法家族

| 名称 | 多项式 | 应用场景 |
|---|---|---|
| CRC-1 | x+1 | 硬件的奇偶校验位 |
| CRC-5-CCITT | $x^5 + x^3 + x + 1$ | ITUG.704 标准 |
| CRC-5-USB | $x^5 + x^2 + 1$ | USB 信令包 |
| CRC-7 | $x^7 + x^3 + 1$ | 通信系统 |
| CRC-8-ATM | $x^8 + x^2 + x + 1$ | ATM HEC |
| CRC-8-CCITT | $x^8 + x^7 + x^3 + x^2 + 1$ | 1-Wire 总线 |
| CRC-12 | $x^{12} + x^{11} + x^3 + x^2 + x + 1$ | 通信系统 |
| CRC-16-BBS | $x^{16} + x^{15} + x^{10} + x^3$ | XMODEM 协议 |
| CRC-16-CCITT | $x^{16} + x^{12} + x^5 + 1$ | Bluetooth，PPP |
| CRC-32 | $x^{32} + x^{26} + x^{23} + \ldots + x^2 + x + 1$ | ZIP，RAR |

# 循环冗余校验码-CRC

## 定义 2 (CRC 原理)

若设码字长度为 $N$，信息字段为 $K$ 位，校验字段为 $R$ 位，则对于 CRC 码集中的任一码字，存在且仅存在一个 $R$ 次多项式 $g(x)$，使得：

$$V(x) = A(x)g(x) = x^R m(x) + r(x) \tag{2.1}$$

其中：$m(x)$ 为 $K$ 次原始的信息多项式，$r(x)$ 为 $R-1$ 次校验多项式（即 CRC 校验和），$g(x)$ 为生成多项式：

$$g(x) = g_0 + g_1 x^1 + g_2 x^2 + \ldots + g_R x^R \tag{2.2}$$

发送方和接收方通过指定的 $g(x)$ 来生成、验证 CRC 码字

# 对称加密算法

单密钥加密算法，按照明文处理方式分类：

1. 分组密码（Block Cipher）：明文数字划分为长度为 $n$ 的组，每组分别在相同密钥控制下变换为等长密文数字，如 DES、AES、IDEA 等
2. 流密码（Stream Cipher）：对明文中单比特进行加密运算，依赖于伪随机数发生器，如 RC4、SEAL 等

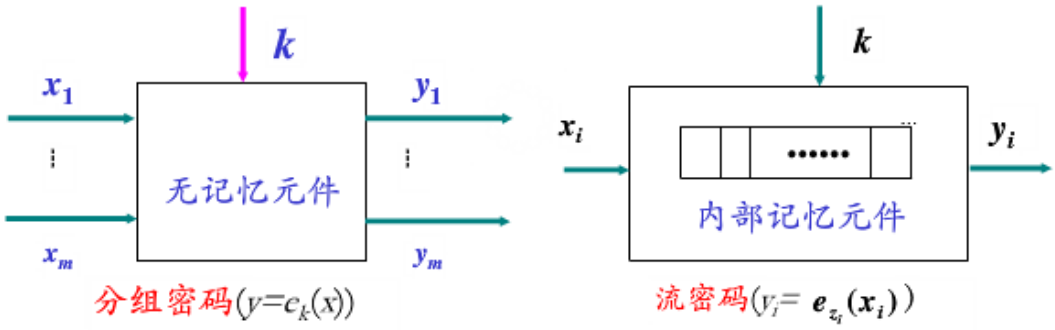



Figure 2.3: 分组密码与流密码

# 分组密码工作模式

分组密码主要有以下四种工作模式：

1. 电码本模式（Electronic Code Book，ECB）：所有分组使用同一密钥独立加密
2. 分组链接模式（Cipher Block Chaining，CBC）：前一个分组的加密结果被反馈到当前分组
3. 密码反馈模式（Cipher Feedback，CFB）
4. 输出反馈模式（Output Feedback，OFB）

# 分组密码工作模式

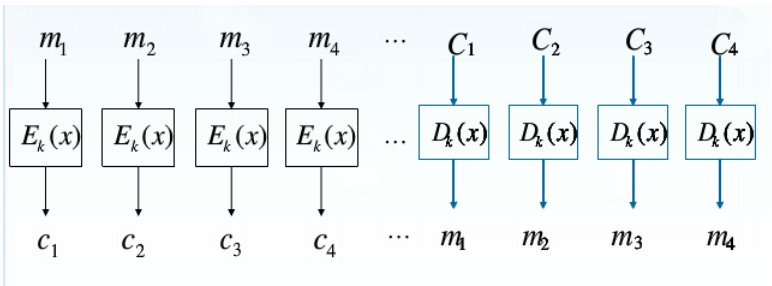



Figure 2.4: ECB 模式

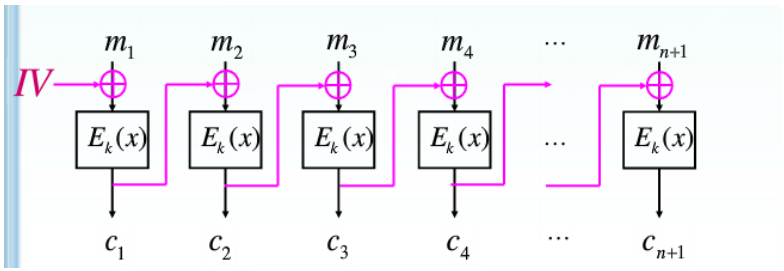



Figure 2.5: CBC 模式

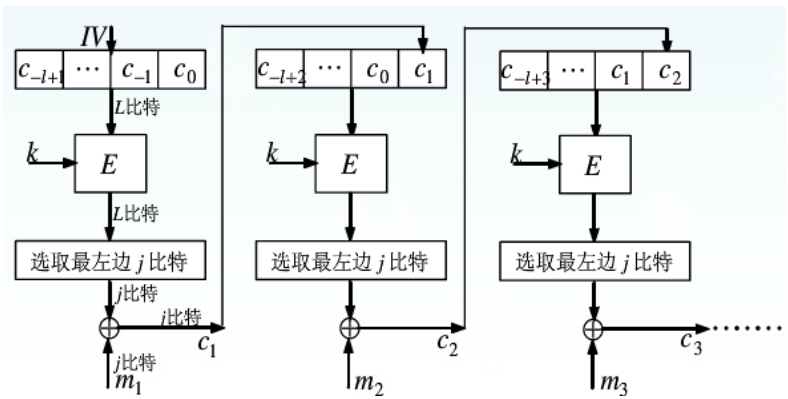



Figure 2.6: CFB 模式

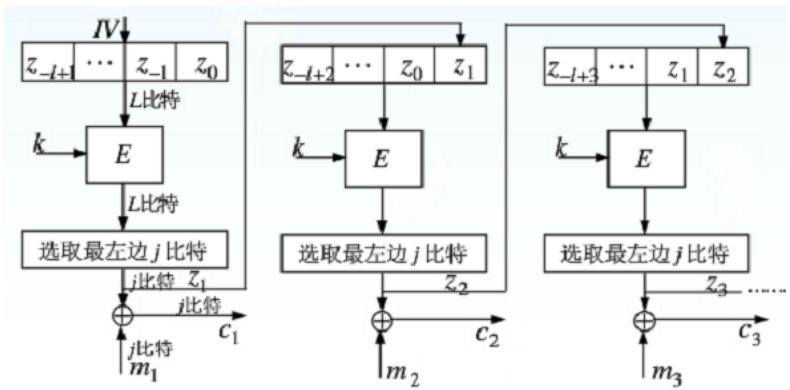



Figure 2.7: OFB 模式

# 扩散和混淆

扩散和混淆的目的是抗击敌手对密码系统的统计分析。

1. 扩散：将明文的统计特性扩散到密文中去，实现方式是使得密文的每一位由明文中的多位产生
   - 明文的统计特性就被散布开，因而在密文中每一字母出现的概率将更接近于相等，使敌手难以通过统计分析得到有用的信息
2. 混淆：就是使密文和密钥之间的统计关系变得尽可能复杂，使敌手无法得到密钥
   - 敌手即便得到了密文之间的某些统计关系，也难以得到密钥

# FEISTEL 网络结构

大多数分组密码的结构本质上基于 Feistel 网络结构，迭代函数：

$$L_i = R_{i-1}, R_i = L_{i-1} + F(R_{i-1}, K_i) \tag{2.3}$$

Feistel 结构安全性参数：

- 分组大小
- 密钥长度
- 子密钥生成算法
- 轮数
- 轮函数

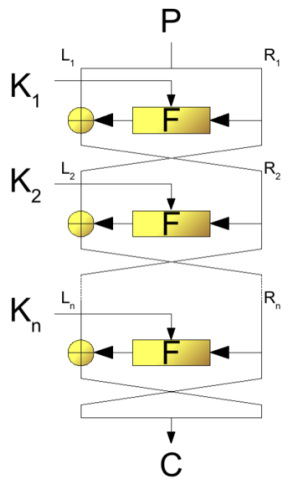



Figure 2.8: Feistel 网络结构

# DES 加密算法

DES 加密基于 Feistel 网络结构，算法步骤：

(1) 初始置换

(2) 密钥置换

(3) 扩展置换

(4) S 盒替代

- 非线性变换，不容易分析，安全性的关键
- 提供密码算法所必须的混淆，改变 S 盒的输入位至少要引起两位的输出变化

(5) P 盒替代

- 使得 S 盒的输出对下一轮多个 S 盒产生影响，形成雪崩效应：明文或密钥的一点小的变化都会引起密文较大的变化

(6) 逆置换

# DES 加密算法

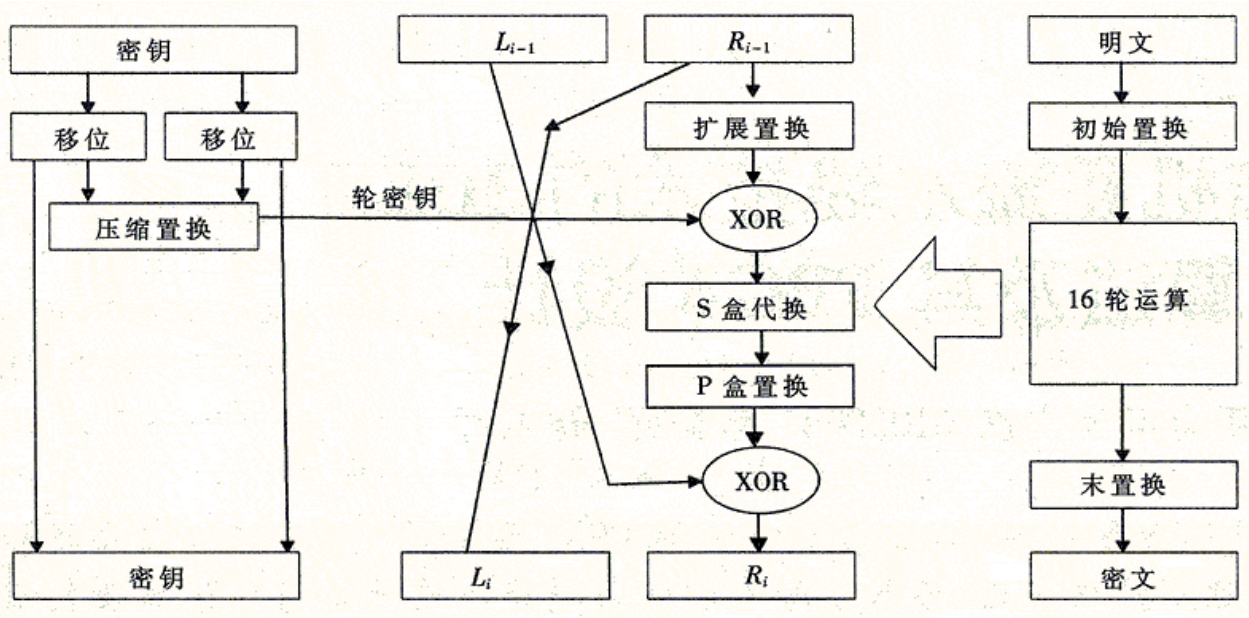



Figure 2.9: DES 算法结构流程图

# 公钥加密算法

公钥加密，也叫非对称加密（Public Key Encryption）

(1) 发送者首先获得接收者的公钥，并使用该公钥加密原文，然后将密文传输给接收者

(2) 接收者使用自己的私钥解密密文

- 优缺点分析
  - 避免了对称密钥系统中容易产生的任何一方单方面密钥泄漏问题以及分发密钥时的不安全因素和额外开销
  - 安全性高，密钥持有量大大减少且易于管理，但计算量大，加密和解密速度慢
- 使用原则：公开密钥算法不用来加密消息，而用来加密会话密钥
- 基本服务：加密解密、数字签名、密钥交换

# 公钥系统满足的条件

约定：Alice 为发送方，Bob 为接收方

- Bob 容易通产生出一对密钥 $(K_{Pub}, K_{Priv})$，并将 $K_{Pub}$ 公开
- Alice 知道公开密钥 $K_{Pub}$ 和消息 $M$ 的情况下，很容易计算出对应的密文 $C = E_{K_{Pub}}(M)$
- Bob 知道私钥 $K_{Priv}$ 和密文 $C$ 的情况下，很容易恢复出明文 $D_{K_{Priv}}(C)$
- 除了 Bob 以外的其他人，要确定私钥在计算上是不可行的
- 除了 Bob 以外的其他人，即使知道 $K_{Pub}$ 和 $C$，要恢复 $M$ 在计算上也是不可行的
- 最终归结为设计一个陷门单向函数：不知道陷门信息求逆困难、知道陷门信息求逆易于实现的函数

# 常用的陷门单向函数

- 大整数因式分解
- 离散对数
- 多项式求根
- 背包问题
- Diffie-Hellman 问题
- 二次剩余问题

# RSA 算法

MIT 三位年青数学家 R.L.Rivest，A.Shamir 和 L.Adleman 于 1977 年提出了一种用数论构造双钥的方法，称作 MIT 算法，后来被广泛称为 RSA 算法。

(1) 首个比较完善的公钥密码算法。既可用于加密数据，又可用于数字签名，容易理解和实现

(2) 经受多年密码分析的攻击，具有较高的安全性和可信度

(3) 简单使用，几乎满足任何应用场合，如：PGP（Pretty Good Privacy）中将 RSA 作为传送会话密钥和数字签名的标准

(4) 国际上一些标准化组织 ISO、ITU、及 SWIFT 等均已接受 RSA 算法作为标准

# RSA 算法

- 系统参数：独立地选取两个大素数 $p$ 和 $q$ 计算

$$n = p \times q \tag{2.4}$$

其欧拉函数值

$$\phi(n) = (p-1) \times (q-1) \tag{2.5}$$

随机选择一整数 $e$，$1 \leq e \leq \phi(n)$，$(\phi(n), e) = 1$。因而在模 $\phi(n)$ 下，$e$ 有逆元

$$d = e^{-1} \pmod{\phi(n)} \tag{2.6}$$

取公钥为 $(n, e)$，秘密密钥为 $d$。

# RSA 算法

用 $x$、$y$ 分别代表明文和密文，RSA 加解密过程描述如下：

- 加密：

$$y = e^x \pmod{n} \tag{2.7}$$

- 解密：

$$x = y^d \pmod{n} \tag{2.8}$$

# DH 密钥分发协议

Diffie-Hellman（DH）密钥交换协议主要用于通信双方的密钥交换或称密钥协商过程。

- 使用该协议密钥交换需要在该协议基础上增加消息认证以增加消息认证以抵抗中间人攻击即可
- 应用非常简单，而且具有前向保密性的安全特性
- 中国无线局域网安全标准 WAPI 采用 DH 密钥交换协议来实现接入点 AP 和终端 STA 之间的密钥协商

# DH 密钥分发协议

- 输入：$(p, g)$，$p$ 为大素数，$g$ 为 $F_p^*$ 的生成元
- 输出：Alice 和 Bob 共享 $F_p^*$ 中的一个元素
- 协议步骤：
  (1) Alice 选择 $x \in_U [1, p-1)$，计算 $g_x \leftarrow g^x \pmod p$ 并发送给 Bob
  (2) Bob 选择 $y \in_U [1, p-1)$，计算 $g_y \leftarrow g^y \pmod p$ 并发送给 Alice
  (3) Aclie 计算 $k \leftarrow g_y^x = g^{xy} \pmod p$
  (4) Bob 计算 $k \leftarrow g_x^y = g^{xy} \pmod p$

# 量子密码算法

量子信息科学的研究包含两部分：

1. 量子密码（Quantum Cryptography）
   - 采用量子态作为信息载体，利用量子的优异特性构建密码算法，安全性基于量子力学基本原理
   - Ex. 无条件安全的 BB84 密钥分配协议
2. 量子计算（Quantum Computation）
   - 利用量子计算的并行性，构建量子计算模型下的高效算法
   - Ex. 经典计算环境下大整数因式分解，已经寻找到多项式时间的量子算法

# 量子力学基本假设

量子力学基本假设数学描述：

1. 对于一个量子系统，其状态和有关情况可以用复内积向量空间（即 Hilbert 空间）的一个态矢量 $|\Psi\rangle$ 完全描述
2. 量子力学中表示力学量的算符 $F$ 对应 Hilbert 空间中的一个线性厄米算符 $\hat{F}$，相应算符的本征值即为力学量的取值
3. 封闭量子系统的态矢量的运行规律遵循薛定谔方程：

$$i\hbar\frac{\partial\psi(t)}{\partial t} = \hat{H}\psi(t) \tag{2.9}$$

其中 $\hat{H}$ 为哈密顿（Hamiltonian）算符。$\hbar = h/2\pi$，$h$ 是普朗克常数

# 量子力学基本假设

4. 算符 $\hat{F}$ 的本征值谱由测量力学量 $F$ 的可能值谱决定。若量子系统处于归一化态矢

$$|\psi\rangle = \sum_i c_i|\psi_i\rangle, c_i = \langle\psi_i|\psi\rangle \tag{2.10}$$

所描述的状态，测量得到的值为本征值 $F_i$ 的概率为 $|c_i|^2$，$c_i$ 是态矢量 $|\psi\rangle$ 在基矢 $|\psi_i\rangle$ 上的展开系数。测量结束，系统的态塌缩到与本征值相对应的本征态

5. 对于任意一对交换粒子而言，描述全同多粒子系统的态矢量都是对称或反对称的

# 量子比特

对于单量子比特，我们约定：

$$|0\rangle = \begin{pmatrix} 1 \\ 0 \end{pmatrix} \tag{2.11}$$

$$|1\rangle = \begin{pmatrix} 0 \\ 1 \end{pmatrix} \tag{2.12}$$

因此量子比特可以表示为

$$|\phi\rangle = \alpha|0\rangle + \beta|1 = \rangle \begin{pmatrix} \alpha \\ \beta \end{pmatrix} \tag{2.13}$$

其中复常数 $\alpha$ 和 $\beta$ 表示概率幅，满足：

$$|\alpha|^2 + |\beta|^2 = 1 \tag{2.14}$$

# BB84 协议

BB84 量子密钥分发协议步骤：

(1) 共享量子纠缠粒子

(2) 随机选择测量基

(3) 分别测量，公布测量基

(4) 选择若干粒子用于窃听检测，如无窃听则剩余粒子的测量结果作为共享密钥，否则重新发起协议

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| Alice的随机比特 | 0 | 1 | 1 | 0 | 1 | 0 | 0 | 1 |
| Alice随机选择的基 | + | + | × | + | × | × | × | + |
| Alice所传光子的偏振态 | ↑ | → | ↘ | ↑ | ↘ | ↗ | ↗ | → |
| Bob随机选择测量的基 | + | × | × | × | + | × | + | + |
| Bob测量的光子的偏振态 | ↑ | ↗ | ↘ | ↗ | → | ↗ | → | → |
| 在公共信道中对比基 | | | | | | | | |
| 共有的密钥 | 0 | | 1 | | | 0 | | 1 |

Figure 2.10: BB84 协议

# 后量子密码学

- 后量子密码（Post-quantum Cryptography）
  - 原因：大整数因式分解、离散对数在量子计算模型下发现多项式时间求解算法
  - 研究基于图灵计算模型、量子计算模型下数学难解问题的密码，如格密码（Lattice-based Cryptography）

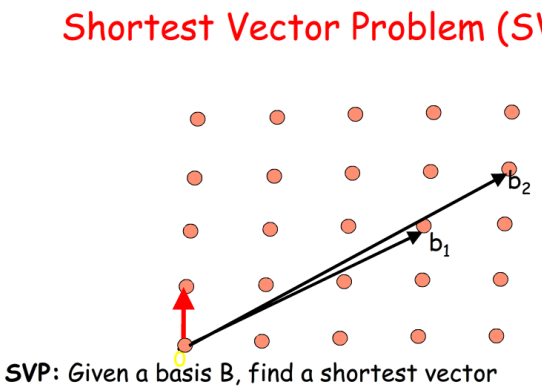



Figure 2.11: SVP 问题

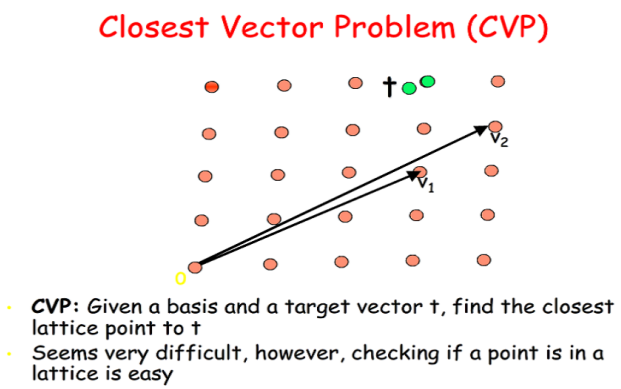



Figure 2.12: CVP 问题

# 实例分析



- 登录认证
- 令牌原理
- openapi 签名
- SSL 协议

# 登录认证

- 注册流程和登录流程

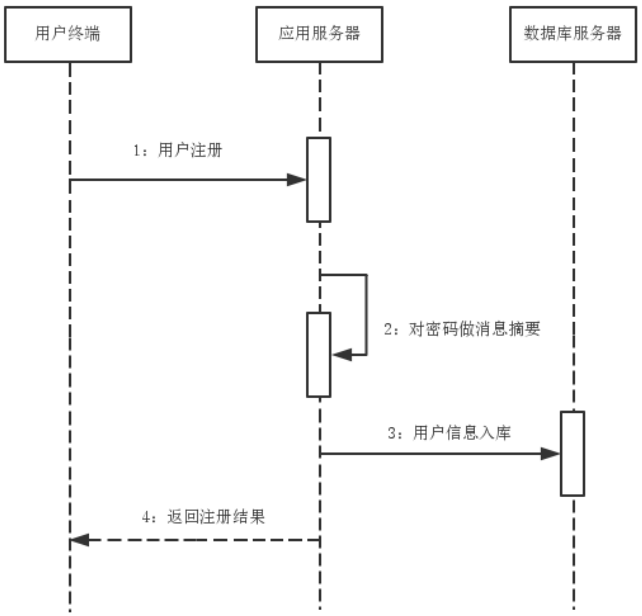



Figure 3.1: 用户注册

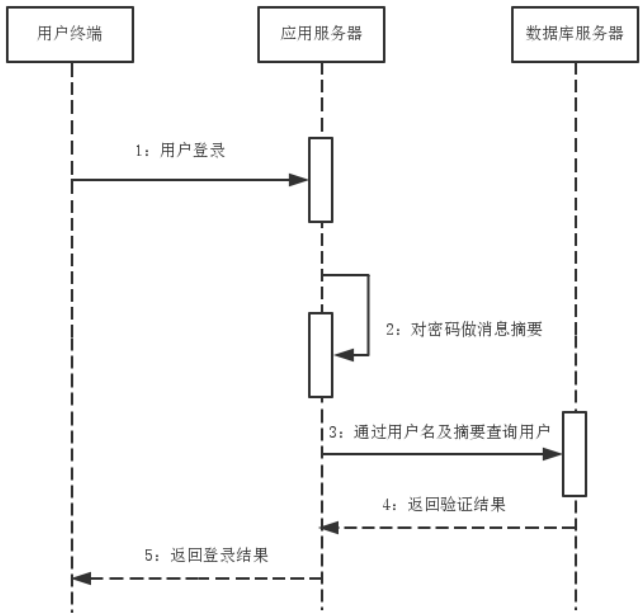



Figure 3.2: 用户登录

# 如何对密码做消息摘要?

- 基本要素：加盐（Salt）哈希
  (1) 选择合适的哈希算法，如 SHA-2
  (2) 选择合适的伪随机数发生器(Cryptographically Secure Pseudo-Random Number Generator，CSPRNG）生成盐值
  (3) 盐值的长度必须足够长
- 尽量避免
  (1) 选择过时的哈希算法，如 MD5、SHA-1 等
  (2) 多次哈希和组合哈希算法
  (3) 不加盐、短盐值、盐值重复或盐值不够随机

# 存储和校验密码

- 存储密码 $P$ 的步骤
  (1) 使用 CSPRNG 生成足够长的盐值 $S$
  (2) 将盐值混入密码，并使用标准的哈希算法进行加密，得到哈希值 $H(S, P)$
  (3) 将 $H(S, P)$ 和 $S$ 存入数据库对应此用户的记录
- 校验密码 $P'$ 的步骤
  (1) 从数据库中取出用户的 $H(S|P)$ 和 $S$
  (2) 计算哈希值 $H(S, P')$
  (3) 比较 $H(S|P')$、$H(S|P)$，相同则密码正确，反之密码错误

# 如何破解哈希加密?

1. 字典攻击、暴力破解
   - 将每个猜测值哈希值后的结果跟目标值比较，如果相同则破解成功
2. 查表法
   - 空间换时间，预计算密码字典中的每个密码，然后把哈希值跟对应的密码存储到一个用于快速查询的数据结构中
   - Crack Station：https://crackstation.net/
3. 反向查表法
   - 攻击者获得系统用户名数据后，构造基于密码-用户名的一对多表，然后猜测一系列哈希值并且从中查找拥有此密码的用户
   - 适用场景：许多用户有相同密码

# 如何破解哈希加密?

4. 彩虹表（Rainbow Tables）
   - 降低字典大小，选择存储一个较小的可逆向的长链密码的哈希值
   - 彩虹表：https://www.freerainbowtables.com/en/tables2/

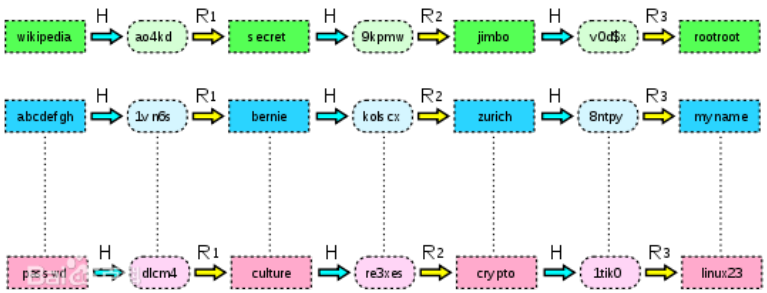



Figure 3.3: 彩虹表

# 进一步安全性

- 加盐可抵抗查表法、彩虹表攻击
- 让密码更难破解：慢哈希函数
  - 降低攻击者的攻击效率，而不至于引起用户的注意
- 密钥哈希和密钥哈希设备
  - TPM（Trusted Platform Module）：可信平台模块, 持有计算机生成的密钥进行加密安全设备。基于硬件的解决方案，防止黑客试图获取密码、加密密钥和其他敏感数据。
- 重置密码重置盐值
- 避免在线系统使用计时攻击

back

# 令牌原理

RSA SecureID 动态令牌：

- 基于 HMAC-SHA-1 算法
- HOTPC=Truncate(HMAC-SHA-1($K, C$))，$K$ 为共享密钥，$C$ 为计数器
- SHA-1 可以替换为 SHA-256、SHA-512 等


Figure 3.4: RSA SecureID 动态令牌

# 令牌原理

令牌协议：

- 符号约定
  - $P$、$V$：代表证明者（令牌）、验证方（令牌服务器）
  - $X$：代表计数单位时间，默认为 30 秒
  - $T$：当前的 unix 时间戳
  - $C = \lfloor (T - T_0)/X \rfloor$：当前计数器计数
  - $S$：$P$、$V$ 间的计数偏移量，初始为 0
- 协议前提：$P$、$V$ 共享密钥 $K$，并且时钟同步
- 协议步骤

(1) $P$ 计算 $C$、$HMAC(K, C)$，发送给 $V$

(2) $V$ 计算 $C'$、$HMAC(K, C' + S - 1)$、$HMAC(K, C' + S)$、$HMAC(K, C' + S + 1)$

(3) 比较哈希值，若与三个中的任何一个匹配，验证通过并更新 $S$，否则验证不通过

# OPENAPI 签名方案

- 密钥分配：验证方生成 *source* 和 *key*，并共享给证明者（*key* 必须绝对保密）
- 签名
  (1) 证明者将 *source*、当前 unix 时间戳 *ts*、指定签名方法 *sign_method* 附到接口参数列表 $PL_{api}$：$(k_1, k_2, \ldots, k_n)$，得到 *PL*：$(k_1, k_2, \ldots, k_{n+3})$
  (2) 证明者对 *PL* 中的参数进行字典排序，得到 $PL_{sort}$：$(k_1^{'}, k_2^{'}, \ldots, k_{n+3}^{'})$
  (3) 证明者根据 $PL_{sort}$ 次序拼接参数-值列表得到 $M$：$k_1^{'} = v_1^{'} \& k_2^{'} = v_2^{'} \& \cdots \& k_{n+3}^{'} = v_{n+3}^{'}$
  (4) 证明者根据 *sign_method* 指定的哈希算法 *H*(不妨设为 MD5) 计算签名 $sign = H(M, key) = MD5(M \& key = SECRET)$
  (5) 证明者将 *PL* 参数-值列表（记为 *PLV*）、*sign* 发送给验证方

# OPENAPI 签名方案

- 验签
  (1) 验证方接收到 $PLV'$，根据 $sign\_method'$ 选择哈希算法 $H'$
  (2) 按照加密步骤 (2)-(4) 计算 $M'$、$sign' = H'(M', key) = MD5(M'\&key = SECRET)$
  (3) 验证方比较 $sign'$ 与 $sign$ 是否相等，相等则验签通过，否则验签不通过

```php
public static function getMd5SignedParams($arrParam)
{
    $arrParam['sign_method'] = 1;
    $arrParam['source'] = self::SOURCE;
    $arrParam['ts'] = time();
    $arrKeyValue = array();
    unset($arrParam['sign']);
    ksort($arrParam);
    foreach ($arrParam as $k => $v) {
        $arrKeyValue[] = $k . '=' . $v;
    }
    $arrKeyValue[] = 'key=' . self::KEY;
    $params = implode('&', $arrKeyValue);
    $arrParam['sign'] = md5($params);
    $arrParam = array_map('urlencode', $arrParam);
    return $arrParam;
}
```

Figure 3.5: openapi 签名

# OPENAPI 签名安全性分析

1. 附上当前 unix 时间戳的作用?
2. 为什么密钥必须附在消息的末尾?
3. 消息篡改是否可能?
4. 是否有进一步的安全隐患?

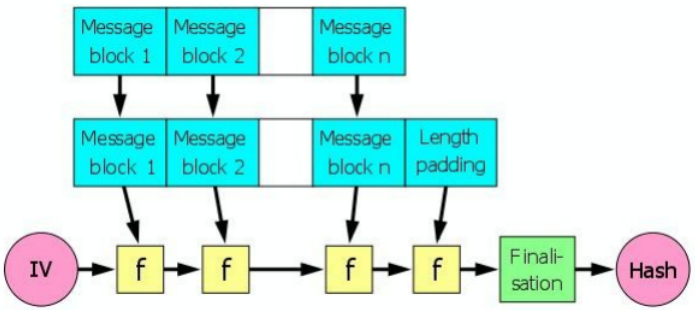



Figure 3.6: MD5 算法

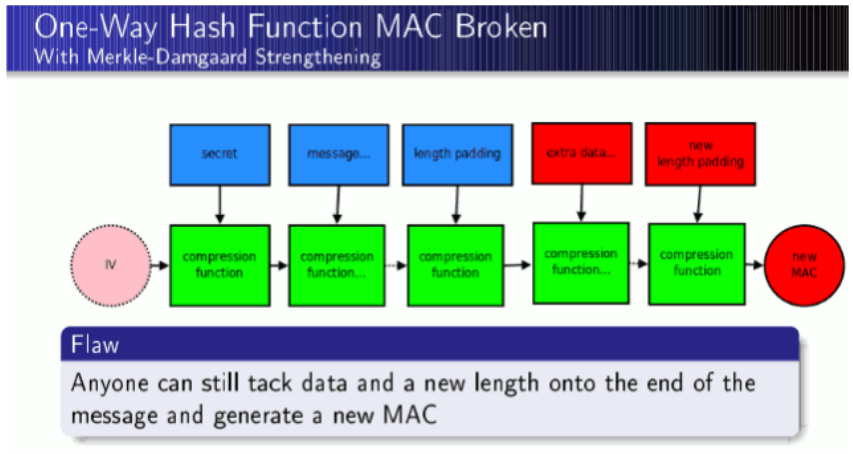



Figure 3.7: 哈希长度扩展攻击

# SSL 协议

SSL（Secure Sockets Layer，安全套接层）：为网络通信提供安全及数据完整性的一种安全协议，包含单向散列、公钥密码、对称密码等算法。协议简要描述：

(1) 颁发证书和验证证书

- 颁发证书：证书信息哈希后用私钥签名，$Sign = E_{K_{Priv}}(H(M))$
- 验证证书：计算 $H(M)$、$D_{K_{Pub}}(Sign)$，两者一致则验证通过，否则验证不通过

(2) 协商会话密钥，结合 DH 密钥分发协议与数字签名

(3) 对称密钥机密传输数据

# SSL 协议安全性

SSL 协议薄弱点：

1. 单向散列算法破解，则可篡改证书
2. 签名算法破解，随意伪造证书
3. 密钥分发协议破解，推导出会话密钥
4. 对称加密算法破解，威胁传输数据的安全

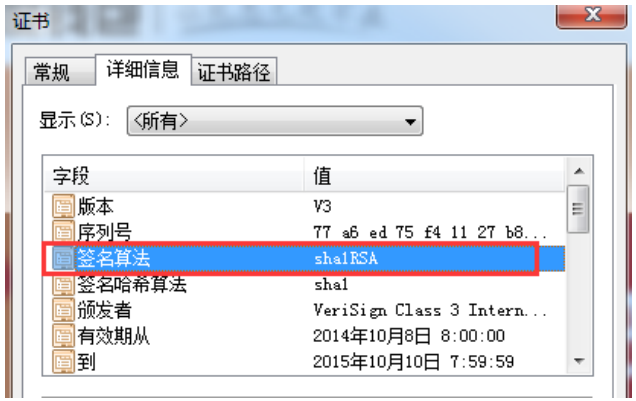



Figure 3.8: 证书信息

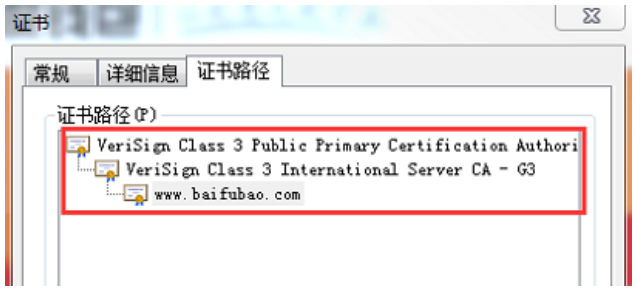
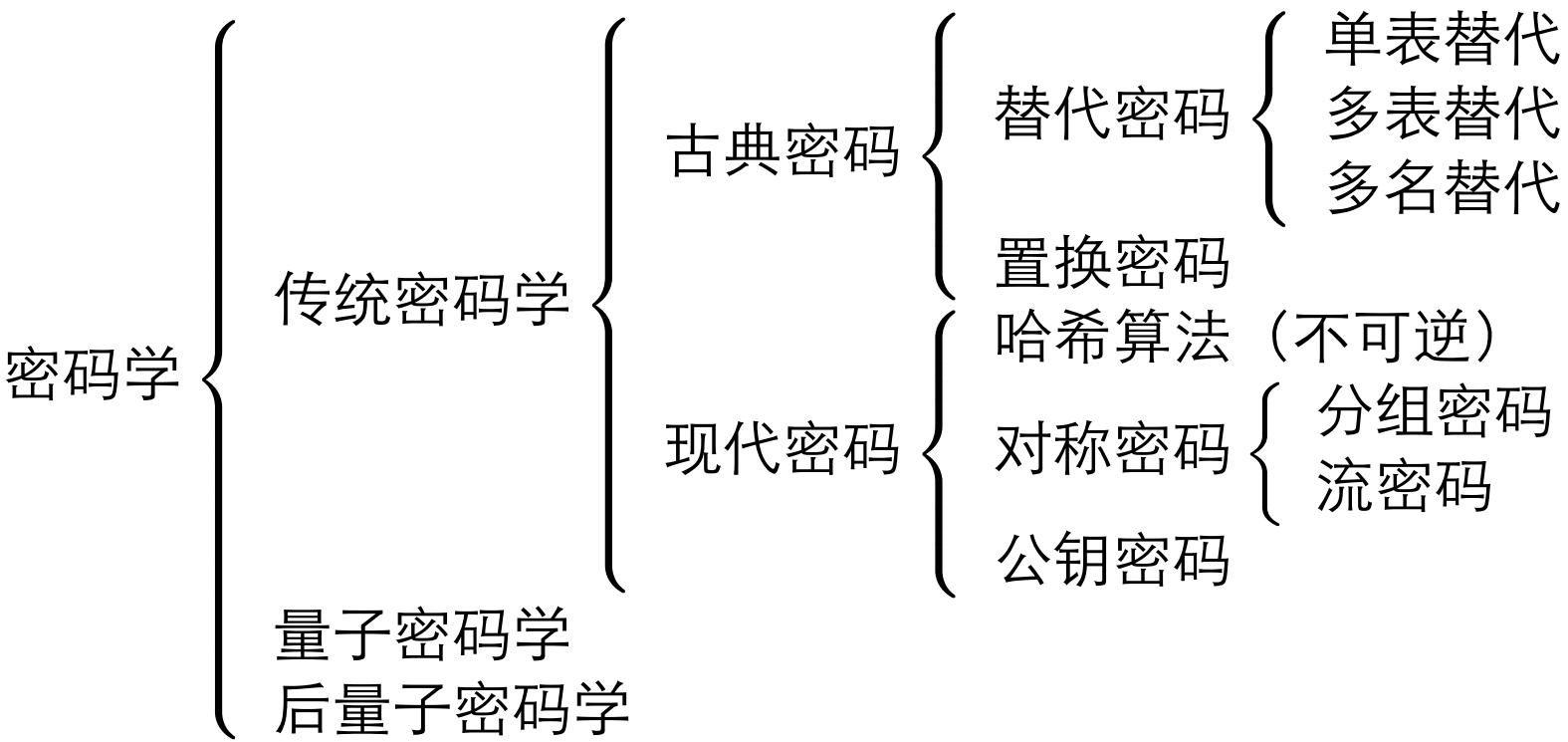



Figure 3.9: 证书路径

back

# 总结回顾


密码学
传统密码学
古典密码
替代密码
单表替代
多表替代
多名替代
置换密码
现代密码
哈希算法（不可逆）
对称密码
分组密码
流密码
公钥密码
量子密码学
后量子密码学

# 总结回顾

- 只有一次一密能做到理论上无条件安全
- 完整 MD5 算法的碰撞攻击已被发现，MD5 不应该再应用于完整性认证、数字签名场景
- 公开密钥算法不用来加密消息，而用来加密密钥
  - 公开密钥算法比对称密钥算法慢，慢 $10^3$ 数量级
  - 公开密钥算法对选择明文攻击是脆弱的
- 公开密钥算法使对称密钥的管理变得简单和更加安全
- 密码系统都立足于安全假设，安全性是相对的
- 量子计算对传统密码学造成威胁
- 后量子密码基于图灵计算模型、量子计算模型下的数学难解问题

back

# 参考文献


- Bruce Schneier 著，吴世忠等译，应用密码学—协议、算法与 C 程序，机械工业出版社 (2000)
- http://www.win.tue.nl/hashclash/
- http://commandlinefanatic.com/cgi-bin/showarticle.cgi?article=art012/
- http://tools.ietf.org/html/rfc6238
- h1ttp://tools.ietf.org/html/rfc6030
- http://thehackernews.com/2017/02/sha1-collision-attack.html


back