补充

攻击分类

被动攻击

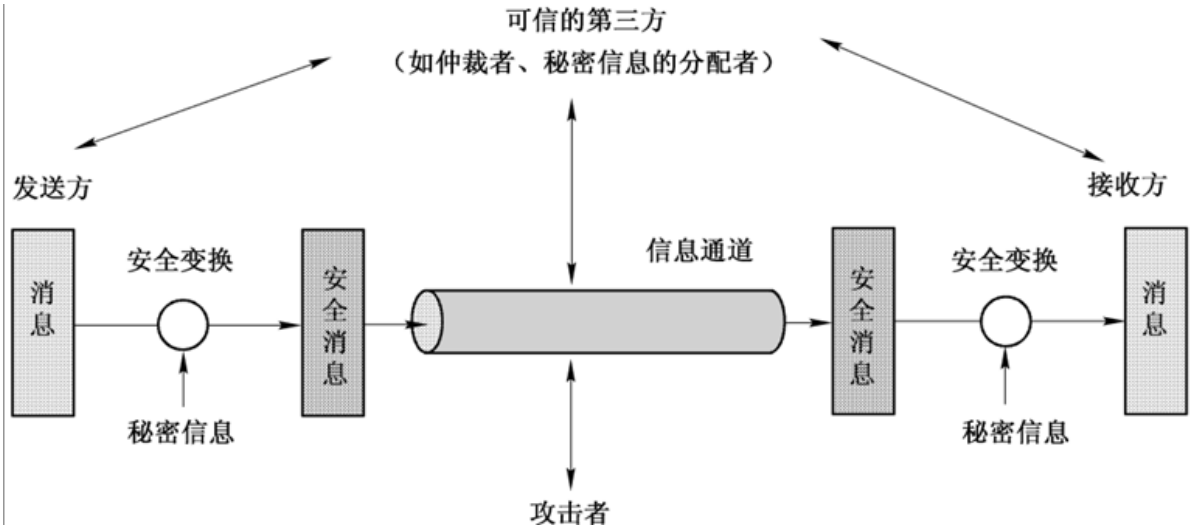
1. 对传输中的明文进行窃听，导致信息内容泄露
2. 针对密文进行流量分析：统计和模式信息 (身份、位置、频度、长度等)

主动攻击

1. 重放：信息再次发送
2. 伪装：假装成其他实体
   包含其他形式攻击，例如：截获认证信息，在验证完信息后进行重放，无权限的实体便可以冒充有权限的实体获取权限
3. 篡改：修改、延迟传送、改变顺序
4. 拒绝服务

网络安全模型

1. 安全变换
   加密消息，使内容不可读；附加消息编码，以验证发送者身份
2. 共享秘密
   密钥等



安全服务包括四个内容：

1. 设计(选定)算法，执行相关变换
2. 产生算法使用的密钥
3. 密钥分发分配
4. 指定安全通信协议

memo1

01. one-time pad，计算安全性 vs. 理论安全性

一次一密one-time pad

生成与明文一样长的随机密钥，只使用一次，加密完就丢弃
唯一有理论安全性的算法
坏处：密钥太长，难以生成、分发、管理密钥

计算安全性 vs. 理论安全性

- 理论安全性：可以在信息理论和数学上证明安全性
  只有当密钥和明文一样长时才能完全保密
- 计算安全性：
  破译消耗的成本 > 密文信息包含的价值
  破译时间 > 密文需要保密的时间

02. 几种对称算法内外特性比较

除了RC4都是对称分组加密

DES

- 分组密码
- 明文分组：64位 / 密钥：56位
  初始置换、16轮Feistel结构与轮函数、循环左移+置换生成子钥、初始置换的逆
- 密钥空间太小，容易穷举出所有密钥，被AES替代
- 除了关键场合外，个人还是可以使用的

2DES

- 使用两个密钥，加密两次DES
  密钥：56 * 2 = 112 位
- 与DES相比密钥空间变大，更安全
  但由于两次加密产生中间值，存在中间相遇攻击，攻击复杂度介于暴力破解DES和2DES之间

3DES

- 使用3个密钥：密钥1加密，密钥2解密，密钥3加密
  使用2个密钥：密钥1加密，密钥2解密，密钥1再次加密
- 安全性高
  是在DES的基础上发生的调整，实现方便
  但是计算成本较高，速度较慢

AES

- 高级加密标准
- 分组密码
- 明文分组：128位 / 密钥：128，192，256位可选
  密钥扩展、字节代换substitution、行位移、列混合、轮密钥加
- 安全，可以抵抗各种攻击
  在1秒内crackDES的机器，需要花149万亿年crackAES
  结构简单、速度快、空间复杂性小，适合软硬件实现 (比3DES更快)

Blowfish

- 明文分组：64位 / 密钥：变长32 - 448位
  子钥生成算法冗长，动态生成S盒，Feistel结构
- 密钥可以足够长，强力攻击更费时
  但是相应的产生子钥的时间会很长

RC5

- 明文分组：3种可选 (半个分组的大小可以是16 / 32 / 64位)
  密钥：长度可变 (0 ~ 2040位)
  循环轮数：可变 (0 ~ 255轮)
  Feistel结构
- RC5-32/12/16推荐组合：64位分组 (半个分组32位)、12轮循环、128位密钥 (16字节)
- 参数适应机器字长，适合硬件和软件实现
  在安全性和速度之间可取舍：轮数和密钥长度越长，安全性越强，速度约慢
- RC6：基于RC5进行改进，Feistel中从两半变成四半

IDEA

- 明文分组：64位 / 密钥：128位固定
  分组被分为4半
  Feistel结构

现代对称密码算法共性

- Feistel结构 (或类似)：DES、Blowfish、RC5/6、IDEA、CAST128
- 密钥长度可变：AES、Blowfish、RC5、CAST-128/256、RC2、RC4
- S盒依赖于密钥
  固定S盒：DES、CAST
  变化的S盒：Blowfish，同时使用冗长的子钥生成算法
- 循环移位依赖于数据/密钥：RC5、CAST
- 轮函数F/循环次数/分组长度可变：CAST-128、RC5
- 每轮同时操作两半：IDEA、Blowfish、RC5
- 混合操作：都使用了算术和布尔运算（除DES）

RC4流算法

- 流密码算法
- 密钥：长度可变
  无限循环生成密钥流，明文流 XOR 密钥流
- 和DES、3DES相比，速度快

流算法和分组算法的比较

- 粒度：8字节分组 vs. 1比特或1字节
  各自适应不同的应用数据格式
- 分组算法：对相同的明文分组，输出相同的密文分组
  流密码：输出不同的密文比特
- 速度：流密码更快
- 适用范围：分组密码是主流 (分组密码也可以用作流模式)
- 安全性：分组密码更好 (理由不充分，只是看起来是)

03. RSA原理和计算

建立参数

- 选取两个素数**p**和**q** (512bits)
- 计算模**n** = p × q
- $\varphi(n)$ = (p-1)(q-1)
- 选取一个公开参数**e**，且与$\varphi(n)$互素
  回避p-1、q-1的因子，从最小的素数开始选取
- 计算出保密参数**d**，使得满足ed mod $\varphi(n)$ = 1
  枚举法可以简化运算

- 发布公钥 **(e, n)**
  保留私钥 **(d, n)**

RSA加解密

- 加密：C = m^e mod n
  明文分组m做为整数须小于n
- 解密：m = C^d mod n

实现考虑

- 素数：必须足够大，否则对手可能很快分解n
  强素数：(p-1)/2和(q-1)/2应是素数

评价RSA

- 简单、好理解，支持广泛
- 既可以加密又可以签名
- 安全性模糊，等价于因子分解
- 不容易产生随机的素数
- 运算量大、速度慢，硬件会受限

04. Hash算法特性、比较、用途

Hash算法：将任意长度的输入数据映射为固定长度散列值

特性

- 单向性质：给定h，找x使H(x) = h，困难
- 弱抗碰撞特性：给定y，找x使H(x) = H(y)，困难
- 强抗碰撞特性（生日攻击）：找x和y使H(x) = H(y)，困难

用途

- 验证数据完整，没有被篡改
- 数字签名
- hashcash：反垃圾邮件和防御分布式拒绝服务攻击的工具，通过要求发送者解决计算密集的哈希碰撞问题，引入计算成本，降低了大规模滥用和攻击的可能性
- 使用Hash和HMAC实现伪随机数发生器
- 登录验证：注册时输入的口令 (password)，通过hash生成哈希值，存储哈希值到数据库，登陆时验证口令的哈希值是否于数据库中一直；为存储口令的数据库增加安全性，即使被非法访问，也只会看到hash值

**补充：碰撞攻击“

碰撞攻击：找到两个不同的输入，它们产生相同的哈希值
当哈希函数的输出位数为n位时，理论上需要大约$2^{n/2}$个不同的输入，有约50%概率出现碰撞。

Hash算法比较

1. MD5（Message Digest Algorithm 5）
   - 输入：任意长
   - 输出：128位
   - 优点：速度快、简单
   - 缺点：容易受到碰撞攻击 (2^{64})，不安全
   - 初始化向量ABCD
2. SHA-1（Secure Hash Algorithm 1）：
   - 输入：任意报文 < 2^{64}

- 输出：160位
- 优点：比MD5更安全，但是速度慢
- 使用大端模式 *(数据的高位保存在内存的低地址中)*
- 初始化向量ABCDE

05. OS Kernel中密码子系统规划

操作系统内核：OS Kernel

1. 身份验证：验证用户身份，确保只有合法用户可以登录系统
    - 身份验证方式：口令、密钥、生物特征等
2. 访问控制：确保对系统资源的访问受到适当的限制
3. 密码管理：安全存储和管理用户密码
4. 密钥管理：
    - 安全生成、存储和分发密钥，用于加密和解密操作
    - 管理证书，支持公钥基础设施
    - 只有经过授权的用户才能使用密钥
5. 安全存储和传输：
    - 加密敏感信息，如存储在系统中的密码、密钥等。
    - 保障密码传输的安全性，尤其是在网络通信中
6. 更新和漏洞管理：
    - 定期更新密码子系统，确保系统安全性
    - 及时修复发现的漏洞，保护系统免受已知攻击

06. hybrid混合体制 (混合加密)

1. 一方用随机数生成对称密钥 (会话密钥)，使用对方公钥传递会话密钥，
2. 对方收到消息后用自己的私钥机密

此时双方共享会话密钥

3. 一方使用自己的私钥对消息进行签名，再用对称密码加密消息
4. 另一方使用对称密钥解密出明文，然后用对方的公钥验证签名

07. MAC和HMAC

MAC 消息认证码（Message Authentication Code)

- 附在消息后的固定大小代码，用于验证消息的完整性
- 将消息和对称密钥作为参数，用消息认证函数生成MAC
    - HMAC：基于哈希函数和密钥生成的MAC
    - CMAC：基于密码算法和密钥生成的MAC
- 接收方也使用相同的对称密钥生成MAC，考察是否与附带的MAC一致
- MAC的安全特性：
    - 生成MAC不需要考虑可逆，没有保密负担，目标是对消息进行验证
    - MAC通常比较短，提高效率
    - 为防范重放攻击，会在生成MAC时加注时间戳或报文序号等内容
    - 对称MAC使用相同密钥进行生成和验证，并不提供签名特性

HMAC Hash-based Message Authentication Code

- 使用带密钥的哈希函数生成MAC

- 可以直接用现有的Hash函数，也可以很容易地用新升级的Hash函数替代
- 保持哈希函数的安全性
- 简单，并易进行密码学分析

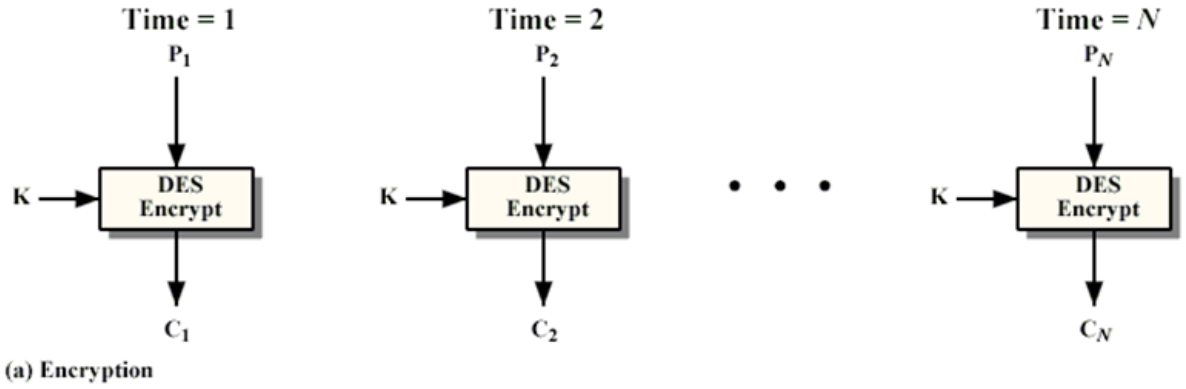
MAC认证和加密分离的意义

- 分离带来灵活性
- 有些场合只需要认证，不需要对数据进行保密或加密
  如：公文、软件完整性鉴别、网络管理广播包围、不可以加密的场合、存档期间的保护

08. ECB CBC CTS XTS CTR

ECB 电子密码本electronic code book

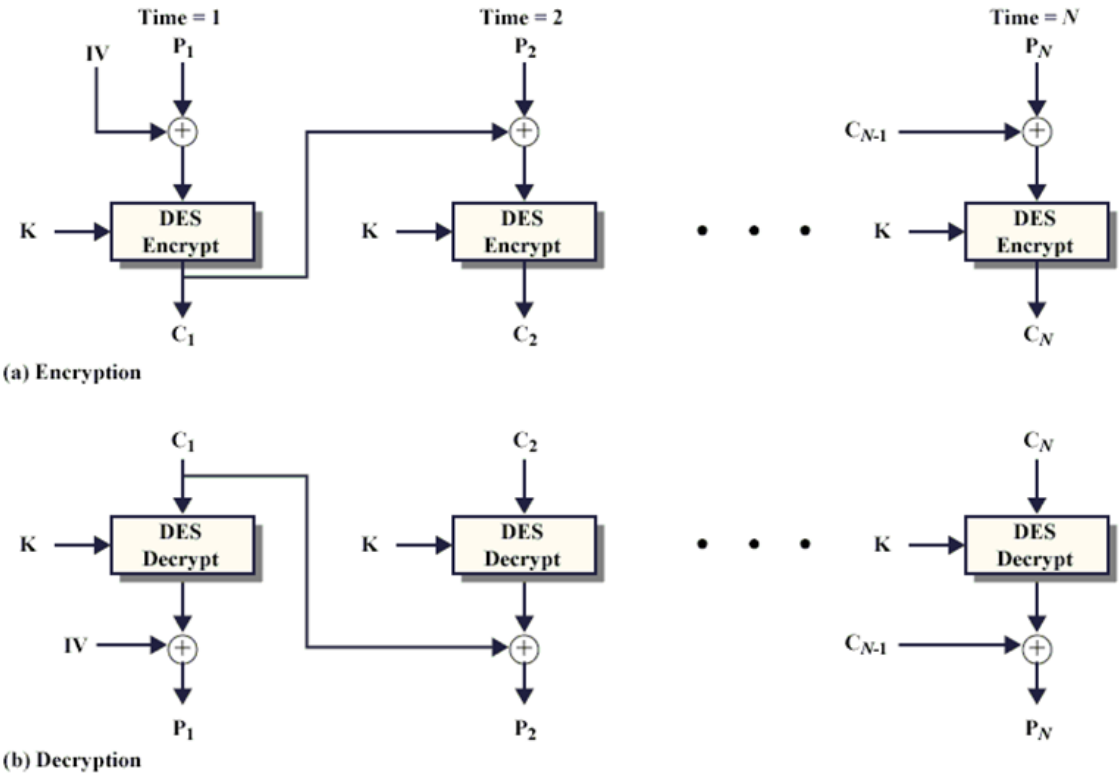
- 报文被顺序分割分成8字节分组
- 各个分组独立加密，密钥相同；解密时需等齐整个分组
- 填充：最后一组不满足分组长度的明文，有必要进行填充



- 优点：并行加密、随机存取
- 缺点：
  - Padding：明文不满足加密时的分组长度，要进行处理 *(此概念的引出见3.1节)*
  - 若有明文分组相同，则对应的密文分组也相同 ⇒ 暴露统计规律
  - 改进：替换、窜改、乱序重排

CBC 密码分组链接 cipher block chaining

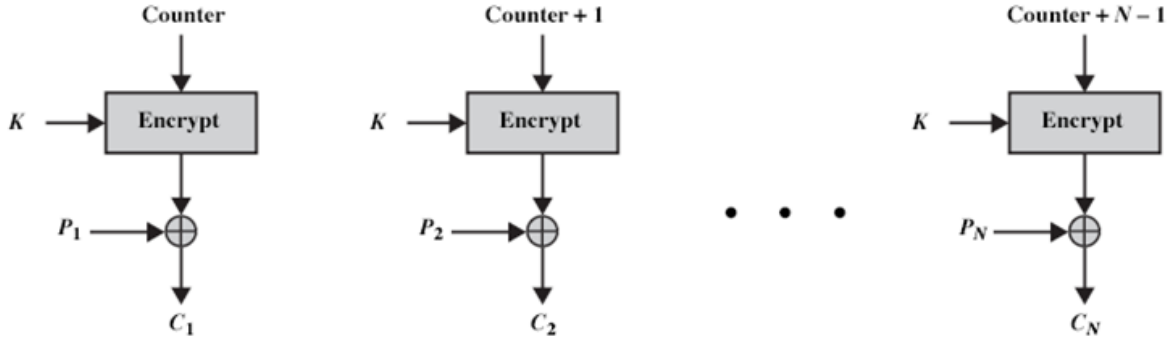
- 将当前的明文组和上一个密文组异或，再进行加密
- 初始向量IV：和第一个明文分组异或
-



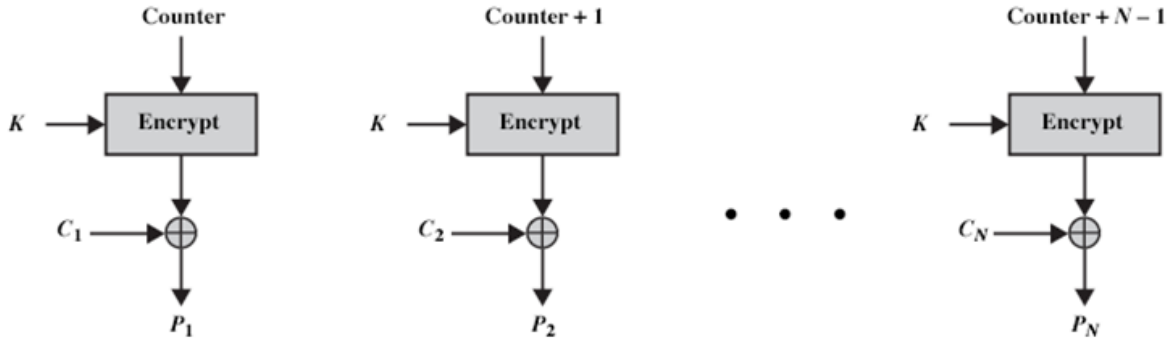
- 优点：
  - 避免明密对应 (看不出来哪些明文是相同的了)
  - 可以用作校验 authentication
- 缺点：
  - 不够分组长度需要padding
  - 不能并行加密、随机存取 (明文前后是链接起来的)

CTR 计数方式 counter mode

- 有一个counter，对其加密再与明文分组异或，得到密文
- 下一次使用的counter是原先+1
- Counter的初值须不能预测
- 流方式的应用，可以非顺序存取，允许并行加密



(a) Encryption

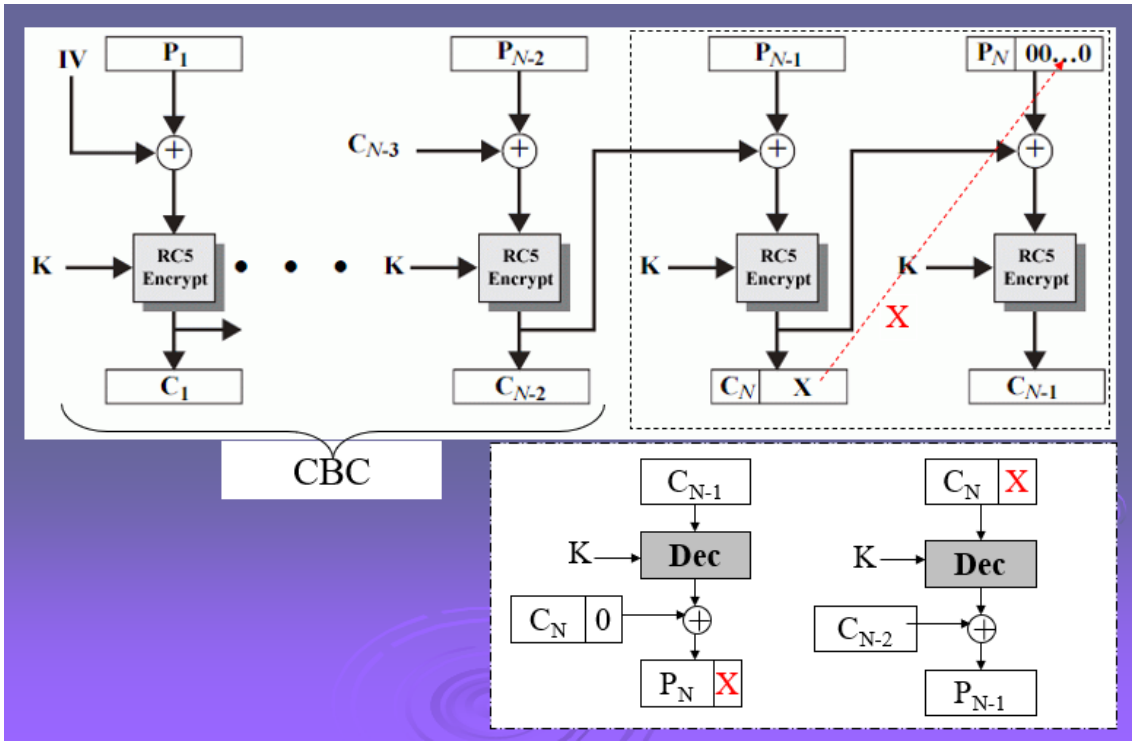


(b) Decryption

- 优点：适合随机存取，因为counter序列是对应的、而且密文不是链接产生的

CTX 密文窃取方式 cipher text stealing

- 最后一组明文的长度小于密文分组长度时，取出倒数第二组密文的后几位，填充进最后一组中
  示例：RC5加密在CBC模式下的密文窃取

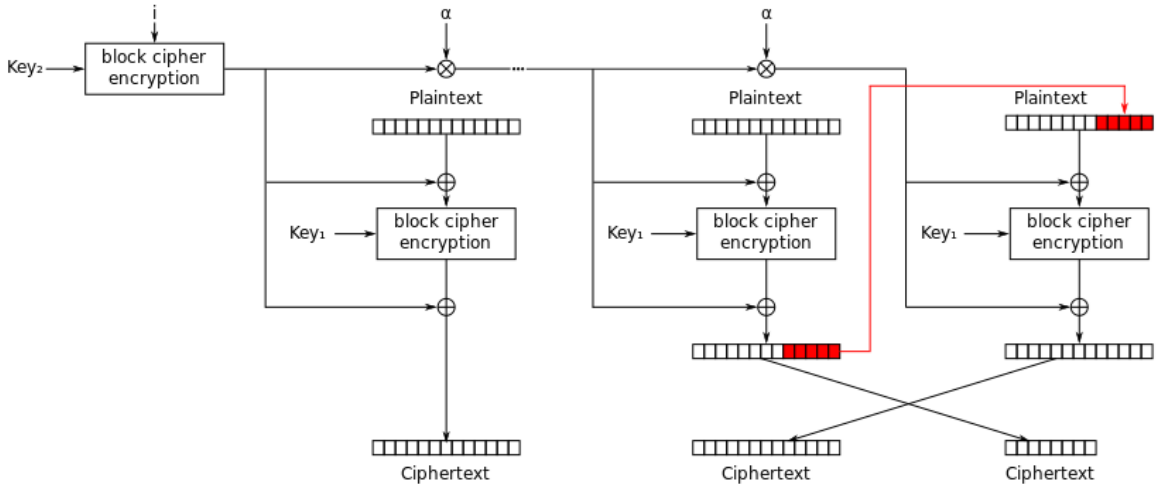


- 优点：
  - 最后一个块不满足大小时不用进行填充
  - 保持加密、解密过程的数据完整性

XTS模式 = XEX+CTS

- 用于分组算法对磁盘上的数据进行加密，采用两个独立的加密过程提供更好的安全性

- 基于XEX：混合模式，数据会进行两次独立加密，分别使用相同的密钥和不同的调整值；
  调整值：保证相同的明文块加密后结果不同
- 使用密文窃取CTS技术



XEX with tweak and ciphertext stealing (XTS) mode encryption

09. *_init/update/final

1. **初始化函数** `_init`：
   用于初始化加密算法的上下文或状态，通常涉及到分配内存、设置密钥、初始化向量等
   初始化函数只在加密/解密开始前调用一次
2. **更新函数** `_update`：
   用于对输入数据进行部分加密或解密
   在处理大文件或流时，数据通常会被分块处理，更新函数以数据块为单位进行处理
3. **结束函数** `_final`：
   用于完成加密或解密过程，处理最后的数据块并生成最终的输出
   在 _final 被调用后，上下文通常不再可用

10. 生日攻击

- 生日攻击是一种密码学攻击，利用了生日悖论的概念，尝试寻找两个不同输入产生相同的哈希值
- 当哈希函数的输出位数为n位时：
  - 最多尝试2^n+1个报文，必有至少一对碰撞
  - 平均尝试$2^{n/2}$个报文，可以以1/2的概率找到一对碰撞
- 抵御生日攻击
  - 增加哈希函数输出长度、
  - 使用更强大的哈希函数
  - 引入一些额外的随机性（如盐值）

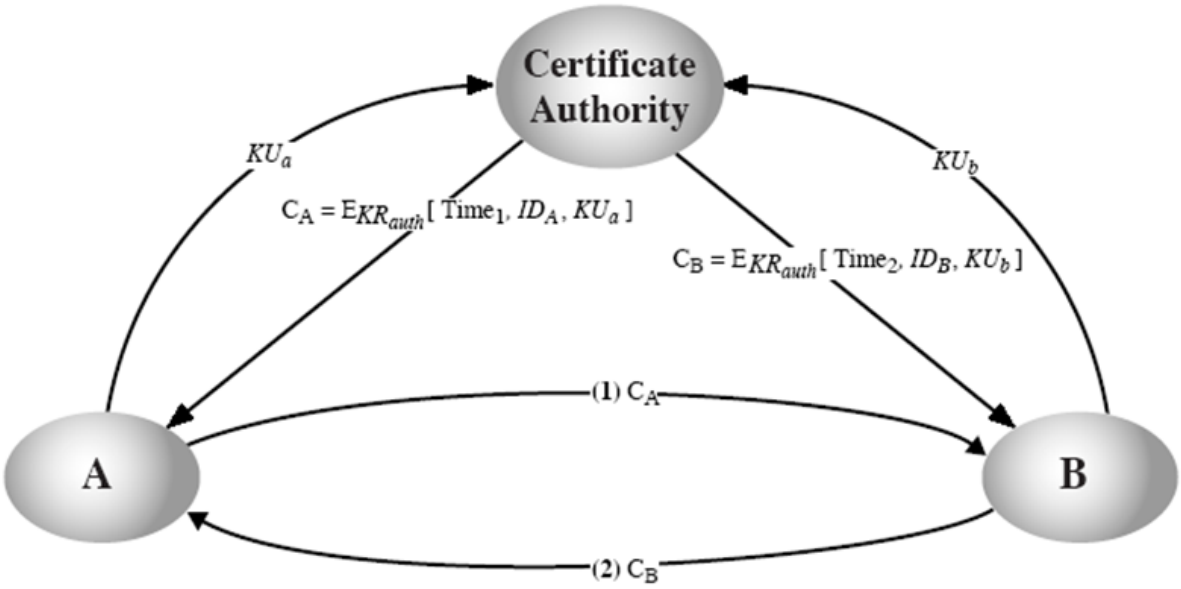
11. 公钥和证书

- **PKI（Public Key Infrastructure）：组织和管理公钥加密体系结构的框架**
- **CA (Certificate Authentication)：证书中心，收信任的权威机构**
  PKI 的核心角色，CA维护和管理数字证书，确保其安全性和可信度
  有一对CA自己的公钥私钥：
  - CA 颁发证书，用私钥签名
  - 用户用CA公钥验证

- **证书（Certificate）：数字身份证明**
  包含用户身份信息的数据结构，是公钥的载体
  证书内容：用户公钥，持有人和签发人的信息，用途，有效期间，签名等

基本流程

1. 每个用户自己产生一对公钥和私钥，并把公钥提交给CA申请证书
2. CA以某种可靠的方式核对申请人的身份及其公钥，并用自己的CA私钥"签发"证书
3. 需要通信时，双发临时交换证书，并用CA公钥验证
4. 获得CA签名保证的对方用户公钥后，可以进行下一步的身份验证和交换会话密钥等



Exchange of Public-Key Certificates

12. key agreement

密钥协商：通信双方协商出一个共享的会话密钥

- 依靠公钥算法传输会话密钥，如RSA
  参考06 混合密码，一方生成会话密钥，用对方公钥加密进行传输，另一方用私钥解密
- 依靠专门的密钥交换算法：如DH算法

Diffie-Hellman密钥交换

- 步骤
  1. 选取大素数p，生成元g；公开p和g
     如果p-1是小素数的乘积，则易求，p-1应含有大的素因子
  2. A选择随机数a，B选择随机数b；私钥
  3. A计算 Ya = g^a mod p，B计算 Yb = g^b mod p
  4. 双方交换Ya，Yb
  5. A计算 K = Y_b^a mod p，B计算 K' = Y_a^b mod p
     事实上，K = K' = g^{ab} mod p
- 离散对数问题：在本次算法中，对于公式Y = g^a mod p：我们可以得到公开的模p、生成元g，也知道交换过程中的Y，但是要计算出私钥a是很困难的
- 中间人攻击
  交换Y的过程中，Y有可能被替换假冒，而且不能发现；导致双方的交流都会经过中间人篡改
  我们需要对AB双方的身份进行验证

13. 公钥相关大数运算的方法和优化

14. hashcash bitcoin中的密码算法

Hashcash通过要求发送者解决计算密集的哈希碰撞问题，引入计算成本，减少大规模滥用和攻击

- 反垃圾邮件：要求发送者在发送电子邮件之前，先解决一个难以计算的哈希碰撞问题，通常是寻找一个哈希值，要求前几位必须满足一定的条件
  发送者用进行大量运算来寻找哈希值，但接收方只要进行简单的验证即可
- Bitcoin中用于工作量证明：矿工需要找到满足一定条件的哈希值，证明他们完成了一定的计算工作
  哈希值有一定难度要求，从而确保新区块的添加需要消耗大量计算能力，提高了网络的安全性和去中心化特性

15. PKCS FIPS RFC

1. **PKCS 公钥密码学标准 Public Key Cryptography Standards**：
   - PKCS是一系列标准，定义了与公钥密码学相关的各种协议、格式和算法。PKCS包括数字签名、证书请求、加密等多个方面的标准，为公钥基础设施（PKI）提供了一组通用的规范。
2. **FIPS（Federal Information Processing Standards）**：
   - FIPS是由美国联邦政府颁布的一系列标准，涵盖了计算机安全、密码学、数据交换等多个领域。在密码学领域，FIPS标准通常与政府机构和承包商有关，确保他们的系统和流程满足一定的安全性要求。
3. **RFC（Request for Comments）**：
   - RFC是由互联网工程任务组（IETF）发布的一系列文件，用于定义和描述互联网相关的协议、流程、程序等。其中，与密码学和网络安全相关的RFC包括了许多重要的协议，如TLS（Transport Layer Security）和IPsec（Internet Protocol Security）。

这三者之间的关系是：

- **PKCS 和 FIPS**：PKCS 标准通常用于指导实现安全通信和数字签名的应用程序，而 FIPS 标准则涵盖了一系列计算机安全领域，其中包括了密码学标准，以确保联邦信息系统的安全性。
- **PKCS 和 RFC**：PKCS 和 RFC 可能在某些方面有重叠，因为它们都是为了推动互联网安全标准而存在的。例如，PKCS #1 定义了 RSA 加密和签名的格式，而相应的 RFC 中也可能包含与这些格式相关的协议。
- **FIPS 和 RFC**：FIPS 标准通常是由美国联邦政府制定的，而 RFC 是由全球社区制定的，但它们可能在某些领域存在交叉，特别是在互联网安全方面。

总体而言，这三者在密码学和安全通信领域都发挥着重要的作用，但它们的关系是复杂而多样的。

16. 国家标准和法规

国密算法

- SM1：对称加密算法，加密强度与AES相当，算法不公开
- SM2：椭圆曲线公钥密码算法 (非对称加密)
- SM3：密码杂凑算法，在安全和效率上与SHA-256相似，杂凑值长度256比特 (散列加密)
- SM4：分组对称密码算法，与AES算法具有相同的密钥长度分组长度128比特，在安全性上高于3DES算法

17. key passwd cryptography

- key 密钥：用于加密和解密数据的一组信息，有对称密钥和非对称密钥
- password 口令：用户为了验证身份而设置的机密字符串，用于访问系统、帐户或数据。密码可以是短字符串，需要是用户可以记住的
- cryptography 密码学：密码学是研究如何保护通信和数据的领域。它涵盖了加密算法、密钥管理、数字签名等技术

18. 单表破解 “数一数”

单表替代密码 Monoalphabetic Cipher（Substitution）

- 每个明文字符都被固定地替换为一个密文字符
- 移位密码和放射密码是单表替代的特殊形式
- 易被攻破：密文带有原始字母使用的频率统计规律
  英文中各个字母、双字母组合、三字母组合、某些单词出现的频率有概率规律

  1. 统计密文中各个字母的出现概率
  2. 猜测高频度密文字母对应明文字母e(或t、a)，最少的是z(或j)
     猜测出现得最多密文字母双组是th

19. “无纸化”的信息时代，如何理解电子数据的安全?

1. 数据加密：使用适当的加密算法保护数据的机密性，涵盖在数据存储、传输、处理等各个过程中
2. 访问控制：只有授权用户能够访问敏感数据
3. 网络安全：保护数据在网络上的传输。使用安全协议（如TLS/SSL）、防火墙和网络隔离来防范网络攻击。有些公司会使用内部网，不允许访问公网，防止泄密。
4. 备份：定期备份数据，并确保数据丢失或遭到破坏时，能够从备份数据中快速恢复系统
5. 安全培训和意识：进行数据安全的培训
6. 对员工行为进行管理：不把机密数据从公司带走、不将私人设备与公司设备连接、使用光盘拷贝数据并在使用后销毁等

20. 两个应用的安全需求分析，安全技术方案设计

21.以考促学新内容：登录(不是登陆)认证的两类方法

口令直传

用户在登录页面输入用户名和密码 (口令)，这些信息通过一个表单提交给服务器

HTTP身份验证

RFC 2617定义了HTTP的基本访问身份验证（Basic Access Authentication）和摘要访问身份验证（Digest Access Authentication）

- 基本访问身份验证
  为了防止用户名和密码被人直接读取，在传输前编码成base-64字符序列
  证书以明文形式传递，并不加密，容易被截获
- 摘要访问身份验证
  使用例如MD5哈希算法，对用户名、密码、以及其他一些参数进行哈希处理，在数据库中保存的是哈希值

使用公钥/私钥对登录

用户拥有一对密钥，公钥保存在服务器上，私钥保存在用户本地
用户在登录时使用私钥进行身份验证
服务器使用事先存储的与公钥相关联的信息来验证用户的身份