

密码学基本概念

两种加密形式

- 传统加密：又称为**对称加密**、单钥加密，安全性在于保持算法本身的保密性
- 现代加密：又称为**非对称加密**、公钥加密，把算法和密钥分开，密码算法公开，密钥保密，安全性在于保持密钥的保密性

基本概念

- 明文/密文
- 加密/解密
- 密码算法/密码：用来加密和解密的数学函数
- 密钥：密码算法中的一个变量， $D_{Kd}(E_{Ke}(m)) = m$

密码学的基本模型

- 密码编码学
- 密码分析学
- 密码编码学和密码分析学统称为密码学

密码编码学

- 研究各种加密方案的学科称为密码编码学
- 密码编码学系统有三个独立的特征：
 1. 转换明文为密文的运算类型，基于两种原理：
 - **置换**：将明文中的元素重新排列
 - **代换**：将明文中每个元素映射成为另外一个元素
 - 原则是不允许丢失信息，即所有的运算都是可逆的
 2. 所用的密钥数：
 - **对称密码**：发送方和接收方使用相同的密钥
 - **非对称密码**：发送方和接收方使用不同的密钥
 3. 处理明文的方法：
 - **分组密码/块密码**：每次处理一个输入分组，相应地输出一个输出分组
 - **流密码/序列密码**：连续地处理输入元素，每次输出一个元素
- 无条件安全和计算安全：
 - 无条件安全：无论有多少可以使用密文，都不足以唯一地确定由该体制产生密文所对应的明文，则加密机制是无条件安全的
 - 计算安全：满足以下条件之一：
 - 破译密码的代价大于加密数据本身的价值
 - 破译密码的时间超过了密文信息的生命期

古典密码

代换技术

- 代换技术是将明文字符替换成其它字母、数字或者符号的方法
- **Caesar密码**: $c = (m + 3) \text{ Mod } 26$, 单表代换密码, 只有25个密钥 (1~25)
- **秘钥词密码**: 设一个密钥词放在前面, 其余字母按顺序排列, 单表代换密码
- **Playfair密码**: 多表代换密码, 将明文中的双字母作为一个单元并将其转换为密文的双字母音节, 即将单字母映射关系变为了一个字母对到另一个字母对的映射关系
 - 构造密钥词的方法是:
 - 加密规则是 (一次加密两个字母) :
- **Hill密码**: 多表代换密码, 将 m 个连续的明文字符代换成 m 个密文字符, 如下图:
 - n 维矩阵能够隐藏 ($n-1$) 大小的字母对的频率特征
- **Vigenere密码**: 是使用一系列Caesar密码组成密码字母表的加密算法, 属于多表代换密码
- **Vernam密码**: 选择一个与明文毫无统计关系并且和明文一样长的密钥, 其运算是基于二进制数据而非字母的
 - $c_i = p_i \oplus k_i$
 - 一次一密 (One-time Pad; OTP), 基于Vernam密码的改进方案, 使用与消息一样长且无重复的随机密钥来加密消息, 无条件安全

置换技术

- 置换密码是通过置换形成新的排列
- 单步置换还是容易被识破, 一般采用多步置换密码就安全多了
- 转轮机采用多层加密原理

破译举例

- 穷举法
- 频率分析法: 单码/双码/三码统计特征

对称密码算法

简介

- 加密和解密使用相同的密钥: $K_E = K_D$
- 密钥必须使用秘密的信道分配

S-DES

- 简化DES(S-DES)是为教学使用的一个加密算法, 与DES有着相似的性质和结构, 但是参数要小很多, 便于理解
- 加密/解密算法的输入为一个8位明文/密文组和一个10位密钥, 输出为8位密文组/明文组
- 加密/解密:
- 加密流程和解密流程几乎相同, 不同之处在于密钥 K_1, K_2 输入的位置
- 加密过程的函数:
 - IP: 初始置换函数, $IP(n1, n2, n3, n4, n5, n6, n7, n8) = (n2, n6, n3, n1, n4, n8, n5, n7)$
 - E/P: 扩展位宽的置换函数, $E/P(n1, n2, n3, n4) = (n4, n1, n2, n3, n2, n3, n4, n1)$
 - S盒:
 - 第1、4位作为二进制数决定S盒的行
 - 第2、3位作为二进制数决定S盒的列
 - 输出即是二进制的2位输出
 - S_0 :
 - S_1 :
 - P_4 : 置换函数, $P_4(n1, n2, n3, n4) = (n2, n4, n3, n1)$
 - SW: 交换函数, SW将输入的左4位和右4位交换

- IP^{-1} : 末尾置换函数, $IP^{-1}(n1, n2, n3, n4, n5, n6, n7, n8) = (n4, n1, n3, n5, n7, n2, n8, n6)$
- 子密钥产生过程中的函数:

Feistel密码结构

- 现在使用的对称分组密码算法都基于Feistel分组密码结构的
- Feistel建议使用乘积密码的概念来逼近简单代换密码
- 乘积密码是指依次使用两个或以上的基本密码
- Feistel 建议交替使用代换和置换
- 混淆和扩散:
 - 扩散是指使明文的统计特征消散在密文中, 让每个明文数字尽可能地影响多个密文数字
 - 混淆是尽可能地使密文和加密密钥间的统计关系更复杂, 以挫败推导出密钥的企图
- Feistel密码每轮迭代都有相同的结构

DES

- DES采用分组加密, 是一种对称密钥算法
- 分组长度是64 bits
- 除了初始置换和末尾置换, DES的结构与Feistel密码结构完全相同
- E盒扩展置换: 将 R_i 从32位扩展到48位, 输入的一位影响下一步的两个替换, 使得输出对输入的依赖性传播得更快, 密文的每一位都依赖于明文的每一位
- S-盒代换选择: 将48比特压缩成32比特

最常用的对称密码

- 3DES: 三重DES加密, 密钥长度为112比特。
- Blowfish: 分组长度是64位, 密钥长度可以在32 ~ 448位之间变化
 - 加密过程:
 - Blowfish算法的子密钥和S盒都是用Blowfish算法本身生成的, 这使得数据完全不可以辨认, 对它的密钥分析也就异常困难
 - Blowfish算法每轮运算都是对数据的左右两个部分同时执行运算, 与古典Feistel结构不同; 这使得密码的强度又增强了
- RC5:
 - RC5的分组长度可以位32、64或者128位, 密钥长度则可取0~2040位
 - RC5是一个由三个参数确定的加密算法族 :
 - w: 分组长度
 - r: 迭代次数
 - b: 密钥K的8位字节数
 -
- AES: AES的分组长度128位, 密钥长度128位
 - AES不是Feistel结构, AES的每一轮都使用代换和置换并行的处理整个数据分组
 - AES的每一轮迭代都包括四个阶段:
 - 字节代换: 用一个S盒完成分组中的字节代换
 - 行移位: 一个简单的置换
 - 列混淆: 算术代换
 - 轮密钥加: 利用当前分组和扩展密钥的一部分进行按位异或
 -

- 密钥被扩展成44个32位字所构成的数组
- 仅仅在轮密钥加阶段使用密钥；轮密钥加实际上是一种Vernam密码

非对称密码算法

公钥密码原理

- 对称密码的缺陷：
 - 密钥必须经过安全的信道分配
 - 无法用于数字签名
 - 密钥管理复杂
- 公钥密码是基于数学函数而不是代换和置换
- 公钥密码的组成部分：
 - 明文：可读的信息，做为加密算法的输入
 - 加密算法：对明文进行的各种变换
 - 公钥/私钥：一个用于加密，一个用于解密。加密算法执行的变换依赖于公钥和私钥
 - 密文：加密算法的输出，不可读信息
 - 解密算法：根据密文和相应的密钥，产生出明文
- 符号说明：
- 公钥密码体制：每个用户产生一对密钥：用于加密和解密。其中一个密钥存于公开的寄存器或者文件中，即公钥；另外一个是私有的，称为私钥
 - 公钥公开，用于加密和验证签名
 - 私钥保密，用作解密和签名
 - 利用这种方法，通信各方皆可以访问公钥，而私钥是个通信方在本地产生的，所以不必进行分配
 - 只要系统控制了私钥，那么它的通信是安全的
 - 任何时刻，系统都可以改变自己的私钥，而公布相应的公钥代替原来的公钥
 - **加密原理示例图：**
 - **签名原理示例图：**
 - **数字签名和加密同时使用：**
- 公钥密码的数学原理：陷门单向函数
 - 单向函数是求逆困难的函数；单向陷门函数，是在不知陷门信息下求逆困难的函数，当知道陷门信息后，求逆是易于实现的
 - 单向陷门函数 $f(x)$ ，必须满足以下三个条件：
 - 给定 x ，计算 $y=f(x)$ 是容易的
 - 给定 y ，计算 x 使 $y=f(x)$ 是困难的
 - 存在 δ ，已知 δ 时对给定的任何 y ，若相应的 x 存在，则计算 x 使 $y=f(x)$ 是容易的
 - 仅满足前两条的称为单向函数，第三条为陷门性， δ 称为陷门信息
- 公钥密码系统的应用：
- 在实际应用中，公钥密码目前仅局限于密钥管理和数字签名
-
- 数论基础：
 - 欧拉函数： $\phi(n)$ ： n 是正整数， $\phi(n)$ 是比 n 小且与 n 互素的正整数个数，如果 p 是素数，则 $\phi(p)=(p-1)$
 - 欧拉定理：若整数 m 和 n 互素，则 $m^{\phi(n)} \equiv 1 \pmod{n}$ → $M^{k\phi(n)+1} \equiv m \pmod{n}$

RSA算法

- RSA体制是一种分组密码，其明文和密文都是 $0 \sim n-1$ 之间的整数，通常 n 的大小为1024位二进制数或者309位十进制数
- 密钥的产生：
- 加密/解密过程：
- RSA算法举例1：
- RSA算法举例2：

- RSA算法的安全性：对RSA算法的攻击方法：蛮力攻击、数学攻击、计时攻击
 - 蛮力攻击：对所有密钥都进行尝试
 - 数学攻击：实质是两个素数乘积(n)的因子分解
 - 计时攻击：攻击者可以通过记录计算机解密消息所用的时间来确定私钥
- RSA算法的性能：
 - 软件实现比DES慢100倍
 - 硬件实现比DES慢1000倍

DH密钥交换算法

- Diffie-Hellman密钥交换算法的目的是使两个用户能够安全地交换密钥，该算法本身也只局限于进行密钥交换
- 数学背景：
 - 本原根： a 是素数 p 的一个本原根，如果 $a \bmod p, a^2 \bmod p, \dots, a^{p-1} \bmod p$ 是1到 $p-1$ 的排列，即各不相同，是整数1到 $p-1$ 的一个置换
 - 对于整数 b ($b < p$) 和素数 p 的一个本原根 a ,可以找到一个唯一的指数 i ，使得： $b \equiv a^i \bmod p$ ，其中 $0 \leq i \leq (p-1)$
 - i 称为 b 的 以 a 为底 模 p 的离散对数或指数 ，记为 $\text{ind}_{a,p}(b)$
 - $\text{ind}_{a,p}(1) = 0$
 - $\text{ind}_{a,p}(a) = 1$
 - 对于 $b = a^x \bmod p$
 - 已知 a, x, p , 计算 b 是容易的
 - 已知 a, b, p , 计算 x 是非常困难的
- DH算法举例：

其他公钥密码算法

- DSA：数字签名算法，算法的安全性是基于计算离散对数的难度
- 椭圆曲线密码系统
- RSA是事实上的标准

密钥的分配

- 存在两种加密方法：
 - 链路加密
 - 端到端加密
- 密钥分配方法就是将密钥发放给希望交换数据的双方而不让别人知道的方法

传统的对称密码分配

- 方法一：密钥由A选择，并亲自交给B
- 方法二：第三方选择密钥后亲自交给A和B
- 方法三：如果A和B以前或者最近使用过某个密钥，其中一方可用它加密一个新密钥后再发送给另外一方
- 方法四： A和B与第三方C均有秘密渠道，则C可以将一密钥分别秘密发送给A和B
- 方法一和二需要人工传送密钥，适用于链路加密；对于端到端加密，则使用密钥分配中心
- 密钥分配中心KDC模式：
 - 假定每个用户与密钥分配中心KDC共享唯一的一个主密钥：
 - A有一个除了自己只有KDC知道的主密钥 K_a
 - B有一个除了自己只有KDC知道的主密钥 K_b
 -

- 在网络规模很大的时候，密钥的分配功能不限定在单个的KDC上面，而是使用层次式的KDC
- 层次式KDC密钥分配使得主密钥分配的代价变小了。而且，如果一个本地KDC出错，或者被攻击了，破坏只是集中在一个区域中，不会影响全局

公钥的分配

- 公钥密码可用于下面两个方面：
 - 公钥的分配
 - 公钥密码用于传统密码体制的密钥分配
- 常用的公钥分配方法有四种：
 - 公开发布：缺点：任何人都可以伪造这种公钥的公开发布
 - 公开可访问目录：维护一个动态可访问的公钥目录可以获得更大程度的安全性；某个可信的实体或组织负责这个公开目录的维护和分配
 - 公钥授权：缺点在于公钥管理员就会成为系统的瓶颈
 - 公钥证书：

利用公钥分配传统密码的密钥

- 简单的密钥分配：
- 具有保密性和真实性的密钥分配：

Lettuce4 认证技术

消息认证

消息认证基本概念

- 泄密：将消息透露给没有合法密钥的接收方
- 传输分析
- 伪装
- 网络环境中的攻击：
- **消息认证就是验证所收到的消息确实是来自真正的发送方且未被修改的消息**
- 任何消息认证都能在功能上看做两层：
 - 下面一层中有某种产生认证符的函数，认证符是一个用来认证消息的值
 - 上面一层将该函数作为原语，使得接收方可以验证消息的真实性

认证函数

- 认证函数：产生认证符的函数
- 可以分为三类：
 - 消息加密：将整个消息的密文作为认证符
 - 消息认证码MAC： MAC是消息和密钥的公开函数，它产生定长的值，该值作为认证符
 - Hash函数：它是将任意长的消息映射为定长的hash值的公开函数，以该hash值作为认证符

认证函数1：消息加密

- 对称加密：
 - 发送方A用A和B共享的密钥K，对发送到接收方B的消息M进行加密，如果没有其他人知道该密钥，就可以提供保密性
 - B可以在接收到消息后，用密钥K解密，确认该消息是由A发出的
- 对称加密的例子：
- 对称加密时需要先构造FCS，再加密，顺序不能变化
- 对称加密的协议：TCP协议，可以对除了IP报头之外的所有数据进行加密
- 公钥加密：
 - 公钥加密只提供保密性，不能提供认证
 - 如果既要提供保密性，又要提供认证，发送方A可以先用其私钥加密(数字签名)，然后用B的公钥加密
 - 这种方法的缺点是执行了四次附加的公钥算法运算
-

认证函数2：消息认证码MAC

- 消息认证码：
 - 消息认证码MAC也是一种认证技术，它利用密码生成一个固定长度的短数据块，并将该数据块附加在消息之后
 - MAC是消息和密钥的函数： $MAC = C_k(M)$
 - MAC函数与加密类似，区别就是MAC算法不要求可逆性，而加密算法必须可逆
- MAC认证的过程：
- 使用MAC的情况：

认证函数3：Hash函数

- 单向的Hash函数是消息认证码的一种变形
- hash函数的输入是长度可变的消息M，输出是长度固定的hash码
- hash函数不使用密钥，仅是消息M的函数，因此也称为消息摘要(Message Digest， MD)
- Hash码是所有消息位的函数，它具有错误检测能力
- 将hash码用于消息认证的多种形式：
 1. 用对称密码对消息以及hash码进行加密：提供了**认证**和**保密性**
 2. 用对称密码仅对hash加密：仅提供**认证**（相当于MAC），对于不要求保密性的应用，会减少处理代价
 3. 用公钥密码和发送方的私钥对hash码加密：提供了**认证**和**数字签名**（数字签名有发送方的私钥所保证）
 4. 先用发送方的私钥对hash码进行加密，再用对称密码中的密钥对消息和上述加密结果进行加密：提供了**认证**、**数字签名**和**保密性**
 5. 假定通信双方共享公共的秘密值S， A将M和S联接后再计算hash值，并将其附加在M后面，由于B也知道S， B可以计算hash值，并验证其正确性：提供了**认证**
 6. 在假设（5）的基础上对整个消息和hash码加密：提供了**认证**和**保密性**
- 对hash函数的要求：
 - 可以操作任意大小的报文m
 - 生成的h长度固定
 - 单向性
 - 抗弱碰撞性
 - 抗强碰撞性

安全hash算法的一般结构

- Hash函数将输入消息分为L个固定长度的分组，每个分组长度为b位，最后一个分组不足b位时，需要填充成b位
- 输入中包含长度，增加了攻击的难度
-
- Hash函数中重复使用了压缩函数f：

- 它的输入是前一步中得出的n位结果（即连接变量）和一个b位分组，输出为一个n位分组
- 连接变量的初始值在算法开始的时候指定，其终值即为hash值。通常 $b > n$ ，因此称为压缩函数
- 设计安全hash函数可以归纳为设计具有抗碰撞能力的压缩函数问题，并且该压缩函数的输入是定长的

主要的hash算法概述

- MD族：
 - MD = Message Digest（消息摘要）
 - MD2、MD4和MD5都产生一个128位的消息摘要
- SHA族：
 - SHA = Secure Hash Algorithm
 - SHA-0：正式地称作SHA系列，发行后不久即被指出存在弱点
 - SHA-1：1994年发布的，与MD4和MD5散列算法非常相似，被认为是MD4和MD5的后继者。
 - SHA-2：实际上分为SHA-224、SHA-256、SHA-384和SHA-512算法。
 - SHA-3：方案正在征集中。
- HAVAL：加密哈希算法
- Gost：是一套俄国标准
- RIPEMD-128/160/320：RIPEMD由欧洲财团开发和设计的MD算法
- 以上算法中，仅有SHA-2和RIPEMD-160未被攻破

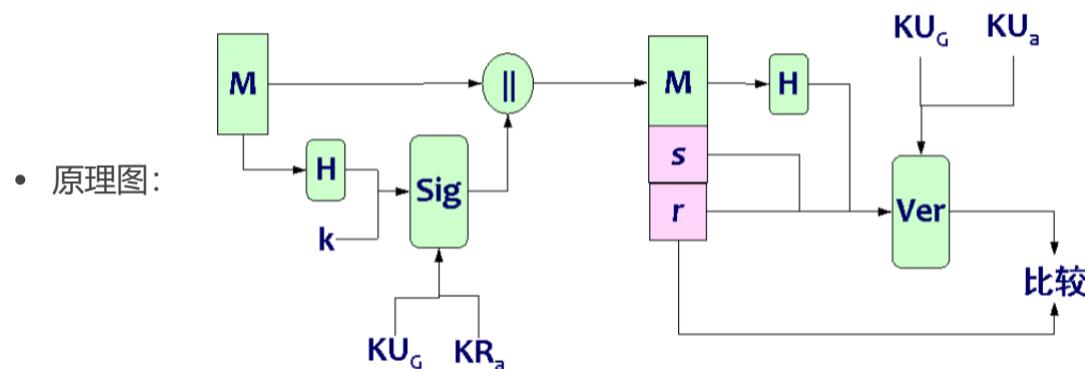
Hash算法：MD5

- MD5的输入是任意长度的消息，对输入按照512位的分组为单位进行处理，算法的输出是128位的消息摘要
 - 使用小端结构（低端结构）
- MD5算法步骤：
 - Step1：增加填充位
 - Step2：填充长度
 - Step3：初始化MD缓存
 - Step4：以512位的分组处理信息
 - Step5：输出
- MD5算法过程：
- Step1：增加填充位
 - 填充bit：第1位为1，其后所有位都为0
 - 填充bit后，使得消息长度与 $448 \ (512-64) \ mod \ 512$ 同余
- Step2：填充长度
 - 用64位表示填充前的报文长度，附加填充比特的后面，如果长度大于 2^{64} ，则对 2^{64} 取模
 - 以低端结构方式表示被填充前的消息长度
 - 完成填充后，消息的总长度是512的整数倍
- Step3：初始化MD缓存
 - Hash函数的中间结果和最终结果都保存于128位的缓冲区中，缓冲区用4个32位的寄存器(A, B, C, D)表示
 - A, B, C, D寄存器以低端格式存储，初始值为 $0 \sim F \ F \sim 0$
- Step4：
 - 计算Hash值
 - MD5的压缩函数
 - MD5 每轮处理512位分组的过程
 - 续上图
- Step5：输出
 - 所有的L个512位的分组处理完之后，第L个分组的输出即是128位的消息摘要
- MD5算法中，hash函数的每一位都是输入的每一位的函数；基本逻辑函数的复杂迭代使得输出对输入的依赖性非常小

Hash算法：各类算法比较

数字签名算法DSS

- 数字签名：消息认证可以保证通信双方不受第三方的攻击，但是它不能处理通信双方自身发生的攻击
- 数字签名的特征：
 - 它必须能够验证签名者、签名日期和时间
 - 它必须能认证被签的消息内容
 - 签名应能由第三方仲裁，以解决争执
- 数字签名可以分成两类：直接数字签名和仲裁数字签名
 - 直接数字签名只涉及通信双方：
 - 假定接收方已知发送方的公钥，则发送方可以通过用自己的私钥对整个消息或者消息的hash码加密来产生数字签名
 - 用接收方的公钥(公钥密码)和共享的密钥(对称密码)对整个消息和签名进行加密，则可以获得保密性
 - 直接数字签名的弱点在于方法的有效性依赖于发送方私钥安全性
 - 仲裁数字签名：
 - 从发送方X到接收方Y的每条已签名的消息都先发给仲裁者A， A对消息及其签名进行检查以验证消息源及其内容，然后给消息加上日期，并发给Y，同时指明该消息已通过仲裁者的检验
 - 通过A的加入解决了直接数字签名的问题
- DSS使用SHA-1算法给出的数字签名方法叫做数字签名算法DSA
- DSS只提供数字签名功能，不能用于加密或者密钥分配



- 用hash函数产生的hash码和随机数k，以及发送方的私钥 KR_a 和 KU_G 作为数字签名函数sig的输入
- 签名函数sig的输出分为s和r两部分，将m, s, r, 拼接为新的消息
- 接收方对接收到的消息产生hash码，此hash码和签名以其作为验证函数Ver的输入，验证函数依赖于全局公钥 KU_G 和 KU_a ，若验证函数Ver的输出等于签名中的r，则签名有效
-

- 全局公钥 (p, q, g)
 - p : 为 L 位长的素数。其中， L 为 512~1024 之间且是 64 倍数的数。
 - q : 是 160 位长的素数，且为 $p-1$ 的因子。
 - g : $g = h^{(p-1)/q} \bmod p$
其中， h 是满足 $1 < h < p-1$ 且 $h^{(p-1)/q} \bmod p$ 大于 1 的整数。

- 用户私钥 x : x 为在 $0 < x < q-1$ 内的随机数

- 用户公钥 y : $y = g^x \bmod p$

- 用户每个消息用的秘密随机数 k , $0 < k < q$

参数 p 、 q 、 g 是公开的； x 为私钥， y 为公钥；
 对于每一次签名都应该产生一次 k ； x 和 k 用于数字签名，必须保密；

签名过程与验证过程

签名过程

用户随机选取 k , 计算：

- $r = (g^k \bmod p) \bmod q$
- $s = [k^{-1}(H(M) + xr)] \bmod q$

(r, s) 即为消息 M 的数字签名

验证过程

接收者收到 M, r, s 后，首先验证 $0 < r < q$, $0 < s < q$, 如通过则计算：

- $w = (s)^{-1} \bmod q$
- $u_1 = [M^w] \bmod q$
- $u_2 = [r^w] \bmod q$
- $v = [(g^{u_1} y^{u_2}) \bmod p] \bmod q$

如果 $v=r$, 则确认签名正确

电子身份认证

- 身份认证是指在主客体交互行为过程中确认行为参与（主体或客体，通常称为用户）身份的解决方法
- 电子身份认证是在计算机及网络中确认操作者身份的解决方法

网站身份认证技术

- 最简单的网站用户认证：HTTP的Basic认证
- HTTP的Basic认证：
 - 用户身份凭证：账号+静态口令 (U, P)
 - 由于HTTP协议是面向一次连接的无状态网络协议，因此需要在每次发出HTTP请求时，把用户身份凭证的明文发送到服务器端，服务器与存储在服务器端的用户凭证进行比较
 - 发送时，对账号口令进行Base64编码，服务器端接收后进行Base64解码
- HTTP的Basic认证：改进1
 - 解决账号口令明文传输可能被监听盗取的问题
 - 使用加密技术
 - 使用消息认证技术
 - 解决账号口令长期本地保存以及服务器端每次都进行账号口令验证的问题
 - 使用表单验证+session的机制
- HTTP的Basic认证：改进2
 - 将口令 P 作为密钥 E_k , 每次发送 $U || E_k[U]$
- HTTP的Basic认证：改进3
 - 使用MAC认证
 - 采用
 - 挑战/响应(Challenge/Response)机制，需要进行两次HTTP请求：
 - 第一次请求：服务器向客户端返回一个随机生成的挑战码 M
 - 第二次请求：服务器向客户端返回一个挑战码 M ，服务器端进行验证

基于表单的身份认证

- Web的Session机制:
 - 一个Session包括特定的客户端、特定的服务器端和特定的操作时间段
- Session的工作原理:
 - 当某个Session首次启用时，服务器会产生一个唯一的标识符发送到客户端
 - 唯一标识符通常是一串随机字符串
 - 在客户端浏览器上，唯一标识符通常用Cookie技术存储
 - 在Session存活期间，客户端每次向服务器发送的HTTP请求都会包含上述唯一标识符，使得服务器能够把前后多次请求关联起来
 - 当Session结束时，服务器端和客户端都应当销毁上述唯一标识符
- 基于表单的Web身份认证过程通常包括三个步骤:
 1. 客户端向服务器发送请求，服务器返回包含表单的页面
 2. 用户按要求填写表单的内容完成后，客户端把表单内容发送到服务器；服务器获取表单中的内容后，进行验证，验证通过则启动Session并返回给客户端（这里账号和口令通过明文传输）
 3. 客户端后续的请求包含Session的唯一标识符，服务端验证唯一标识符的合法性
- 缺点是安全性低，账号和口令明文在网络上传输
- 基于表单的身份认证：改进1
 - 引入Challenge/Response机制，避免口令明文通过网路传输，并且能避免重放攻击：
 - 第一步，服务器返回表单页面时，包含一个Challenge（通常为随机字符串+时间戳，记为M）
 - 第二步，用户完成表单填写提交请求之前，以口令为加密密钥（k=口令）计算Ek[M||U]，并将账号U和Ek[M||U]发送到服务器，服务器收到后，用存储的口令k'计算Ek'[M||U]，与收到的Ek[M||U]比较，完成用户的身份验证
- 基于表单的身份认证：改进2
 - 使用传输层SSL协议传输HTTP请求
- 网站身份增强认证的其它方式:
 - 手机短信口令
 - 动态口令
 - USB KEY
 - 数字证书

Lettuce5 WLAN安全

无线网络概述

- 无线电通信技术是在没有物理连接的情况下多个设备之间能够互相通信的技术

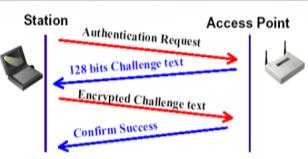
无线网络	网络类型	应用范围/典型技术	传输距离
	无线个人域网WPAN (Wireless Personal Area Network)	点对点短距离链接，个人办公家庭环境 IEEE 802.15x/Bluetooth/ZigBee等	10m左右
	无线局域网WLAN (Wireless Local Area Network)	点对多点无线连接，支持AP间切换 IEEE 802.11 (a.b.g.n) 等	100m
	无线城域网WMAN (Wireless Metropolitan Area Network)	点对多点无线连接，支持基站间漫游与切换 IEEE 802.16等	1-50KM
	无线广域网WWAN (Wireless Wide Area Network)	全球通讯，通信卫星 IEEE 802.20等	1-15KM

- WLAN的安全威胁:

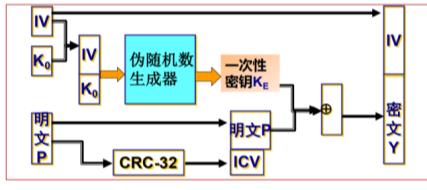
- 无线窃听
- 假冒攻击
- 信息篡改
- 重放、重路由、错误路由、删除消息
- 网络泛洪

无线局域网加密认证技术

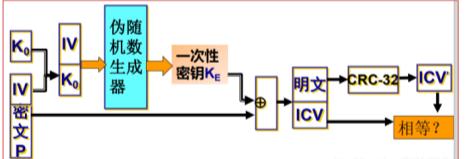
- 无线局域网加密认证技术
 - 无加密认证
 - WEP
 - WPA
 - 无加密认证:
 - 无线接入点AP：网络的中心节点
 - STA站点：连接到无线网络中的终端
 - SSID：便于用户识别，为每个AP配置的标志名，通常情况下，AP会向外广播SSID，可以通过disable Broadcast提高安全性
 - 有线等效保密协议WEP：
 - 有线等效保密协议WEP (Wired Equivalent Privacy) 目的是为无线局域网提供与有线网络相同级别的安全保护，协议标准为IEEE 802.11b
 - 使用对称加密算法
- WEP的安全措施**

 - 认证:
 - 开放系统认证
 - 默认认证方式
 - 对请求认证的任何人提供明文认证
 - 共享密钥认证
 - RC4流密码加密，密钥长度40bit/104bit
 - 完整性：循环冗余校验CRC32
 - 密钥管理:
 - 各个设备与接入点共享一组默认密钥，可能被泄露
 - 每个设备与其他设备建立密钥对关系，密钥人工分发困难

WEP 加密过程

 - 加密:
 - 将24位初始化向量IV和40位共享密钥K0连接得到64位的数据，输入到虚拟随机数产生器中，产生一次性密钥KE
 - 将KE和上一步的计算结果进行按位异或运算，得到密文Y
 - 传输:
 - 将24位初始化向量IV和密文Y串联起来，在无线链路上传输

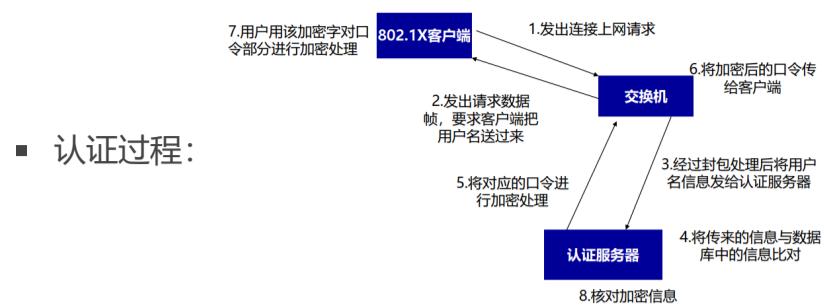
WEP 解密过程

 - 解密:
 - 将收到的信息中初始化向量IV和共享密钥K0串接，输入到虚拟随机数产生器中，产生一次性密钥KE
 - 将收到信息中的密文P和一次性密钥KE进行按位异或运算，得到明文P和ICV串接信息
 - 认证:
 - 将计算得到的明文P进行完整性CRC-32计算，得到校验和ICV'
 - 比较ICV和ICV'，相等接收，不相等丢弃

WEP的安全性分析

 - 同一个SSID中，所有STA和AP共享同一个密钥，容易泄露
 - 生成RC4密码流的初始化向量IV明文发送
 - 24 bits的初始化向量IV过短，容易重复
 - 完整性校验算法CRC-32是非加密的线性运算，主要用于检测消息中的随机错误，不是安全的杂凑函数，无法实现消息认证
 - RC4是一个序列密码加密算法，发送者用一个密钥序列和明文异或产生密文，接收者用相同的密钥序列与密文异或恢复出明文，容易被破译
 - WEP协议中不含序列号，无法确定帧顺序，无法提供抵抗重放攻击
 - WPA1：因为WEP的严重安全漏洞，2002年Wi-Fi联盟制定了WPA1，这是一个过度性的中间标准，其核心就是IEEE 802.1x和TKIP
 - 临时密钥完整性协议TKIP：包在已有的WEP密码外围的一层“外壳”
 - TKIP使用WEP同样的加密引擎和RC4算法，但是TKIP中密钥使用的密钥长度是128位
 - 动态变化每个数据包所使用的密钥
 - 利用TKIP传送的每一个数据包都具有独有的48位序列号
 - IEEE 802.1x：
 - IEEE 802.1x是针对以太网而提出的，基于端口的网络访问控制利用物理层特性对连接到无线端口的设备进行身份认证

- IEEE 802.1x基于客户/服务器模式，可以在无线终端与AP建立连接之前，对用户身份的合法性进行认证



■ 认证过程：

• WEP-WPA1-WPA2比较：

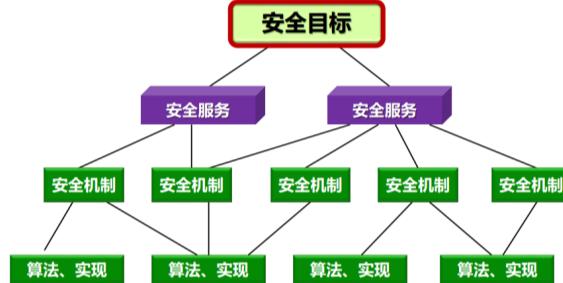
- WPA使公共场所和学术环境安全地部署无线网络成为可能
 - WEP使用一个静态的密钥来加密所有的通信，这意味着为了更新密钥，技术人员必须亲自访问每台机器，而这在学术环境和公共场所是不可能的
 - WPA采用有效的密钥分发机制，不断转换密钥

加密技术	全称	加密算法	协议
WEP	Wired Equivalent Privacy (有线对等保密)	RC4	IEEE 802.11b
WPA	Wi-Fi Protected Access (WiFi安全存取)	RC4, 使用TKIP	IEEE 802.11i draft 3
WPA2	Wi-Fi Protected Access 2 (WiFi安全存取 第二版)	支持AES, 使用CCMP 需要新硬件支持	IEEE 802.11i

Lettuce 6 CIA & VPN

安全目标

• 安全目标、服务、机制的关系：



安全目标：CIA

- Confidentiality: 保密性、机密性
- Integrity: 完整性
- Availability: 可用性

OSI 安全框架

OSI安全框架主要关注安全服务、安全机制和安全攻击

- 安全服务：一种由系统提供的对系统自愿进行特殊的处理或通信服务，安全服务通过安全机制来实现安全策略
- 安全机制：用来保护系统免受监听、阻止安全攻击及恢复系统的机制
- 安全攻击：主动攻击、被动攻击

安全服务

- X.800提供了以下安全服务：
 - Authentication: 认证服务
 - Confidentiality: 保密服务
 - Integrity: 数据完整性保护
 - Access Control: 访问控制服务
 - Non-repudiation: 抗抵赖服务
 - Availability: 可用性服务

Authentication 认证服务

- 认证服务Authentication与保证通信的真实性有关
- 在单向通信时：认证服务向接受方保证发送方的真实性
- 在双方通信时：

- 在连接的初始化阶段，认证服务保证双方的真实性
- 认证服务还需要保证该连接不受第三方非法干扰
- 两个特殊的认证服务：
 - 对等实体认证：
 - 参与通信的实体的身份是真实的
 - 一个实体不能试图进行伪装或者对以前连接进行非授权的重放
 - 面向连接的应用
 - 数据源认证：
 - 对数据的来源提供确认，但是对数据的复制和修改不提供保护
 - 保证接收到的信息的确来自它所宣称的来源
 - 面向无连接的应用

Confidentiality 保密服务

- 保密服务Confidentiality是防止传输数据遭到被动攻击
- 分为连接保密服务和无连接保密服务
- 保密力度：流(stream)、消息(message)、选择字段(field)
- 保密服务防止攻击者观察到消息的源、目的、频率、长度或者其它流量特征

Integrity & Access Control

- 数据完整性服务Integrity：可对消息流、单条消息或消息的选定部分进行保护
- 访问控制服务Access Control：是指限制实体的访问权限，通常是经过认证的合法的实体才可以访问

Non-repudiation & Availability

- 抗抵赖服务Non-repudiation：防止发送方或者接收方否认传输或者接收过某条消息
- 可用性服务Availability：根据系统的性能说明，能够按照授权的系统实体的要求存取或使用系统或系统资源的性质

安全机制

分类

安全机制分为两类：

- 普通安全机制：不属于任何协议层或者安全服务的安全机制
- 特定安全机制：在特定的协议层实现的安全机制

普通的安全机制

- 可信功能(trusted functions)
 - 根据某些标准被认为是正确的
- 安全标签(security Labels)
 - 资源的标志，指明该资源的安全属性
- 事件检测 (Event Detection)
 - 检测与安全相关的事件
- 审计跟踪 (security audit Trail)
 - 收集用于安全审计的数据，对系统记录和行为的独立回顾和检查
- 安全恢复 (security recovery)
 - 处理来自安全机制的请求，如事件处理、管理功能和采取恢复行为

特定的安全机制

● 八种特定的安全机制包括

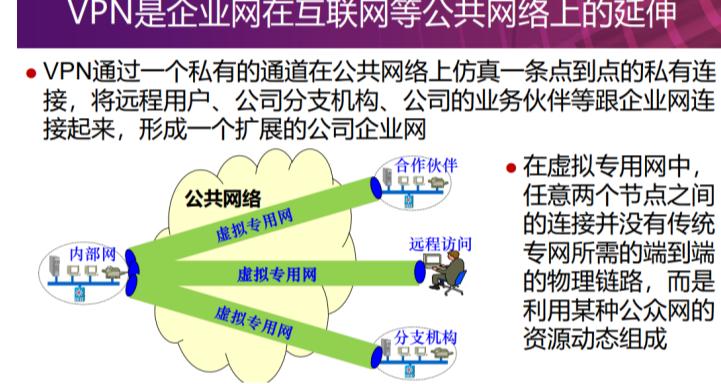
- ① 加密机制
- ② 数字签名机制
- ③ 访问控制机制
- ④ 数据完整性机制
- ⑤ 认证机制
- ⑥ 业务流填充机制
- ⑦ 路由控制机制
- ⑧ 公证机制

安全性攻击

- 主动攻击：试图改变系统资源或者影响系统运行
- 被动攻击：试图了解或者利用系统的信息但不影响系统资源
 - 窃听
 - 流量分析

虚拟专用网VPN

- 需求背景：TCP/IP协议本身存在许多固有的安全缺陷，架设物理专用网难度大
- VPN的定义：Virtual Private Network，虚拟专用网
-



- VPN提供的安全功能
 - 数据机密性保护
 - 数据完整性保护
 - 数据源身份验证
 - 重放攻击保护
- VPN的解决方案
 - 基于数据链路层的VPN解决方案:L2TP/PPTP/L2F
 - 基于网络层的VPN解决方案:IPsec/IKE
 - 基于传输层的VPN解决方案:SSL
- 基于数据链路层的VPN解决方案：
 - 由于数据链路层的VPN技术在认证、数据完整性以及密钥管理等方面的应用不足，现在已经很少应用
- 基于传输层的VPN解决方案:
 - 基于传输层SSL协议的VPN解决方案，零客户端是其最大优势
 - 在实际应用中，SSL VPN和IPsec VPN两种方案往往结合实行
 - 像视频会议这样的非B/S (Browser-Server) 结构的业务是无法通过SSL VPN建立和开展的

Lettuce 7 IPsec & IKE

■ 网络层安全协议：IPsec

IP级安全性

- IPsec保证了IP级的安全性，包括：
 - 认证
 - 保密
 - 密钥管理
- 现有IP协议的安全特性：
 - 无连接，不保证顺序到达；存在着重复包、丢失包；设备简单、无状态
 - 所提供的安全服务：无认证、完整性、保密性，基于IP地址的访问控制，不完备
- IPsec的原理在于可以在IP层加密和/或认证所有流量
- IPsec可以在主机、防火墙或路由器上实施

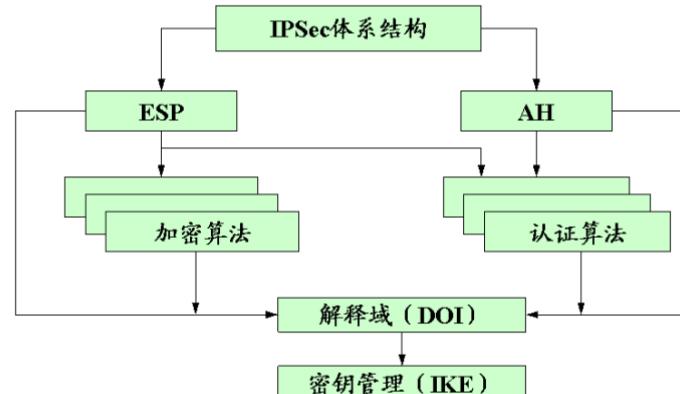
IPsec：文档

- IPsec文档：
 - 重要的有1998年发布的RFC 2401/2402/2406/2408

◦ 整个IPsec文档被分为7个部分：

- 体系结构
- 认证头AH
- 封装安全载荷ESP
- 加密算法
- 认证算法
- 密钥管理
- 解释域

◦



IPsec提供的服务

- 它们提供的安全服务包括：

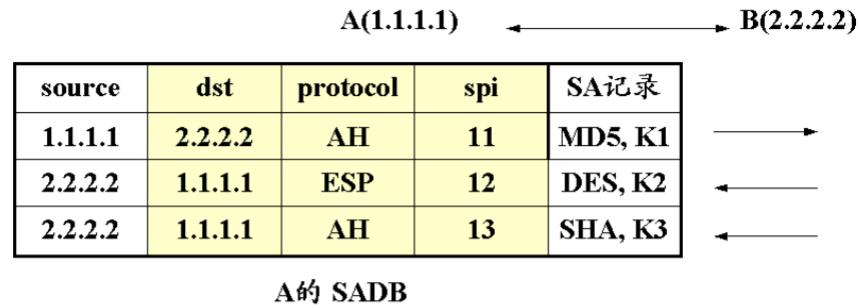
- 访问控制
- 无连接完整性
- 数据源发认证
- 拒绝重放数据包
- 保密性（加密）
- 有限的信息流保密性

	AH	ESP(只加密)	ESP(加密并认证)
访问控制	✓	✓	✓
无连接的完整性	✓		✓
数据源发认证	✓		✓
检测重放攻击	✓	✓	✓
机密性		✓	✓
有限的通信流保密		✓	✓

安全关联

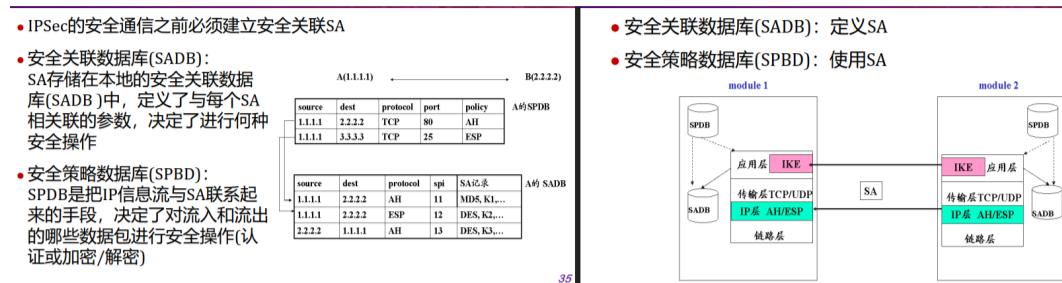
- IP认证和保密机制中的一个核心概念是安全关联SA
- 安全关联SA是IPsec通信双方之间对某些要素的一种协商，一组安全信息参数集合，包括：协议、操作模式、密码算法、认证算法、密钥、密钥生存期等
- 关联是发送方和接收方之间的单向关系，该关联为双方的通信提供了安全服务
 - 如果需要双方安全交换，则建立两个安全关联
 - 安全服务可由AH或者ESP提供，但不能两者都提供
- 一个安全关联SA由3个参数唯一确定：

- 安全参数索引SPI：一个和SA相关的位串，仅在本地有意义，由AH和ESP携带，使得接收方能够选择合适的SA处理接收包
- IP目的地址IPDA：只允许使用单一地址
- 安全协议标识：标识该关联是一个AH安全关联或ESP安全关联
- 安全关联数据库SADB：包含了 (SPI, IPDA, AH/ESP) 三元组索引
 - 在任何IPSec实现中，都有一个安全关联数据库SADB，它定义了与每个SA相关联的参数



- SA的参数：
 - 序列号计数器：一个32位整数，刚开始通常为0，用于生成AH或ESP头中的序列号域；每次用SA保护一个包时增1；在溢出之后，SA会重新进行协商
 - 序列号溢出标志：表明序列号是否溢出，序列号计数器的溢出时，该值为1时，产生审查事件并阻止该SA继续下发数据包
 - 反重放窗口：决定输入AH或ESP报文是否是replay的32位计数器
 - AH信息组：(AH必须实现) 认证算法，密钥，密钥生存期和AH的相关参数
 - ESP信息组：(ESP必须实现) 加密和认证算法，密钥，初始值，密钥生存期和ESP的相关参数
 - SA的生存期：(可选) 一个时间间隔或字节计数 生存期结束时，SA必须终止，或用一个新的SA替换
 - IPSec协议模式：隧道模式或者传输模式
 - Path MTU：任何遵从的最大传送单位路径和迟滞变量
- SADB的工作过程：
 - 在IP数据包中，安全管理SA由IPv4或IPv6报头中的目的地址唯一标识，SPI被封装在AH或者ESP扩展头中
 - 任何IPSec实现中都必须实现安全关联数据库SADB；对于收到的数据包，解析出三元组【SPI、目的地址、AH/ESP】，并据此查找SADB
 - 如果查找到一个匹配的SA条目，则将该SA的参数与AH或ESP头中相关域进行比较：参数相一致，就处理该数据包；参数不一致，就丢弃该数据包
 - 如果没有查找到匹配的SA条目：如果数据包是输入包，丢弃；如数据包是输出包，将创建一个新的SA，并将其存入SADB中
- SA选择子：每个SPDB入口由一个IP集合和上层协议定义，称为选择子(SA selector)
 - IP流量与特定SA相关是通过安全策略数据库SPDB定义的
 - SPDB至少应该包括定义IP流量子集的入口、指向该流量SA的指针
 - IPSec对需要IPSec保护的流量和不需要IPSec保护的流量进行了大粒度区分
 - SA选择子用于过滤输出流量，并将它们映射到某个特定的SA
 - 每个IP包的输出过程如下：
 - 在SPDB中比较相应域的值，寻找匹配的入口，可能是零或多个
 - 如果存在SA，则选定SA和其关联的SPI执行所需的IPSec处理 (AH或ESP)
 - 目的IP地址、源IP地址
 - 用户标识
 - 数据敏感性级别
 - 传输层协议
- SPDB入口由下列选择子确定：
 - IPSec协议
 - 源端口和目的端口
 - IPv6报类
 - IPv6报流标签
 - IPv6报服务类型
- 安全策略数据库SPDB：
 - 在IPSec中，安全策略通过SPDB来定义、标识、管理和维护

- 对于数据包的操作策略包括：
 - Discard**: 不让这个包进入或外发
 - bypass IPsec**: 不对进入或外发的数据包进行安全服务
 - apply IPsec**: 对外发的数据包提供安全服务，同时认为接收的数据包已经进行过安全服务
- SADB与SPDB:

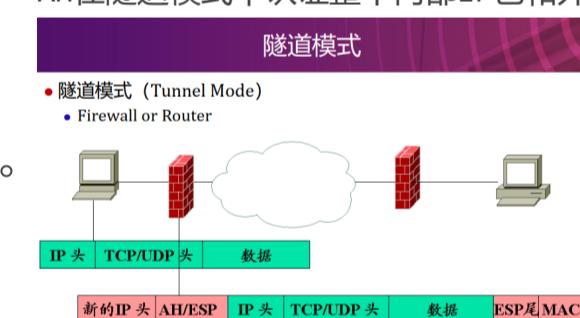


传输模式/隧道模式

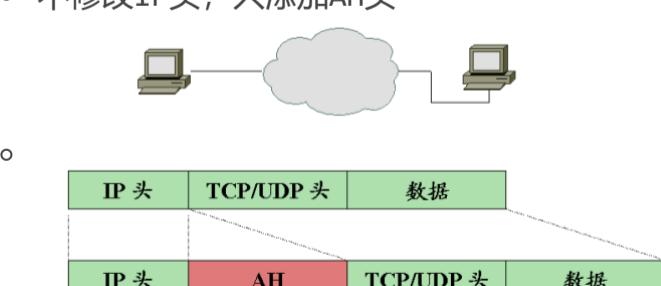
- AH和ESP均支持两种模式：传输模式和隧道模式
- 传输模式：主要为上层协议提供保护，同时增加了IP包载荷的保护
 - 典型的传输模式用于两台主机之间进行的端到端通信
 - 传输模式的ESP加密和认证（可选）IP载荷，不包括报头
 - 传输模式的AH认证IP载荷和报头的选中部分



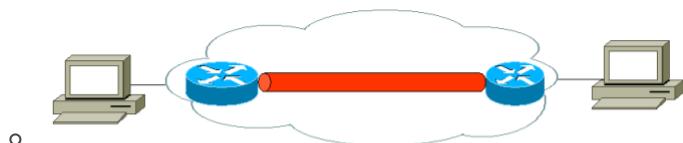
- 隧道模式：对整个IP包提供保护
 - 当IP包加上AH/ESP域后，整个数据包和安全域被当作一个新的IP载荷，并拥有一个新的外部IP报头
 - 新的IP数据包利用隧道在网络中传输，途中的路由器不能检查内部IP报头
 - ESP在隧道模式中加密和认证（可选）整个内部IP包，包括内部IP报头
 - AH在隧道模式中认证整个内部IP包和外部IP报头的选中部分



- 在传输模式和隧道模式中使用AH
 - 途径1：直接在服务器和客户工作站之间提供传输模式的认证
 - 途径2：在服务器不支持认证的情况下，远程工作站向防火墙证明自己身份，以便访问整个内部网络的时候，使用隧道模式SA
- 在传输模式中使用AH：
 - 通常以端到端方式实现（在主机上实现）
 - 不修改IP头，只添加AH头



- 在隧道模式中使用AH：
 - 通常在防火墙或路由器上实现
 - 把整个IP包作为数据，增加一个新的IP头、AH头



IP 头 TCP/UDP 头 数据

新的IP头 AH IP 头 TCP/UDP 头 数据

- 在传输模式和隧道模式中使用ESP：

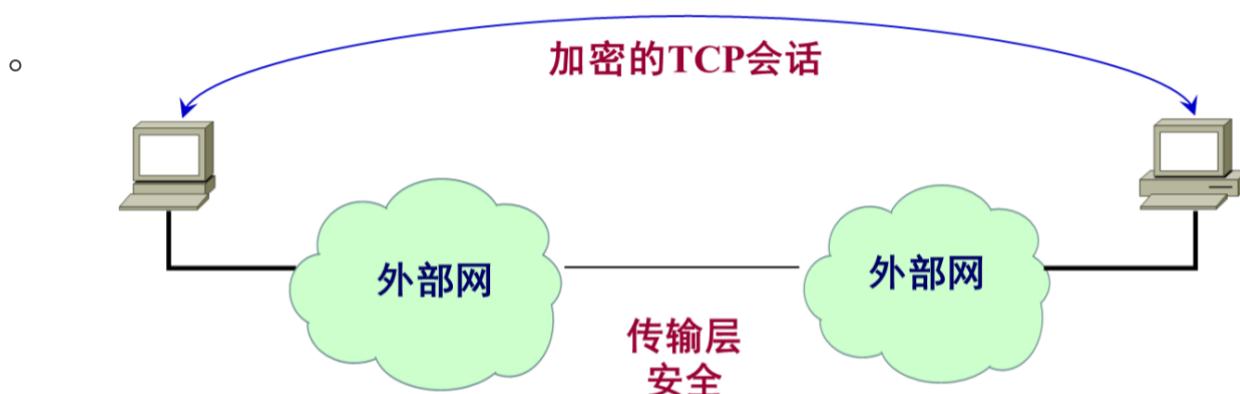
- 途径1：加密和认证（可选）直接由两个主机提供，采用传输模式
- 途径2：隧道模式的ESP用于加密整个IP包，可以创建VPN

- 在传输模式中使用ESP：

- 在源端，包括ESP尾和整个传输层分段的数据块被加密，块中的明文被密文替代，形成要传输的IP包；如果选择了认证，则加上认证
- 将包送往目的地：中间路由器需要检查和处理IP报头以及任何附加的IP扩展头，但不需要检查密文
- 目的节点对IP报头和任何附加的IP扩展报头进行处理后，利用ESP报头中的SPI解密包的剩余部分，恢复传输层分段数据

IP 头 TCP/UDP 头 数据

IP 头 ESP 头 TCP/UDP 头 数据 ESP 尾 MAC

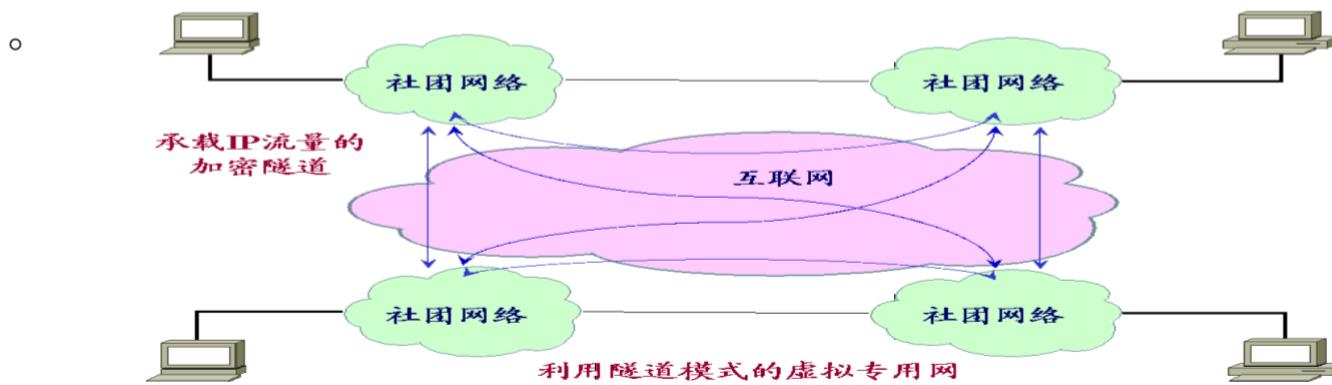


- 在隧道模式中使用ESP：

- 隧道模式的ESP用于加密整个IP包，可用于建设虚拟专用网
- 将ESP头作为包的前缀，并在包后附加ESP尾，然后对其进行加密

IP 头 TCP/UDP 头 数据

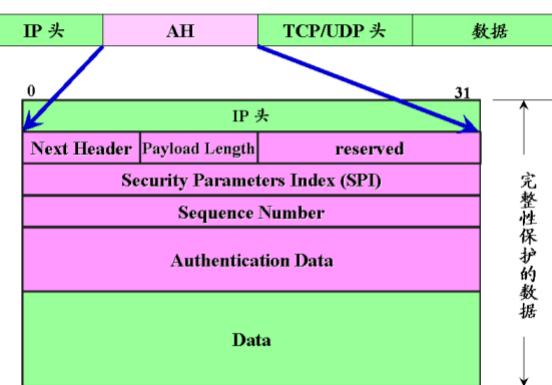
新的IP头 ESP 头 IP 头 TCP/UDP 头 数据 ESP 尾 MAC



IPsec：认证头AH

- 认证头AH支持数据完整性和IP包的认证
- 数据完整性确保在包的传输过程中内容不可更改
- 认证基于消息认证码MAC，双方必须共享一个公钥
- 认证确保末端系统或者网络设备对用户或者应用程序进行认证，并提供相应的流量过滤功能，同时还能防止地址欺诈攻击和重放攻击
-

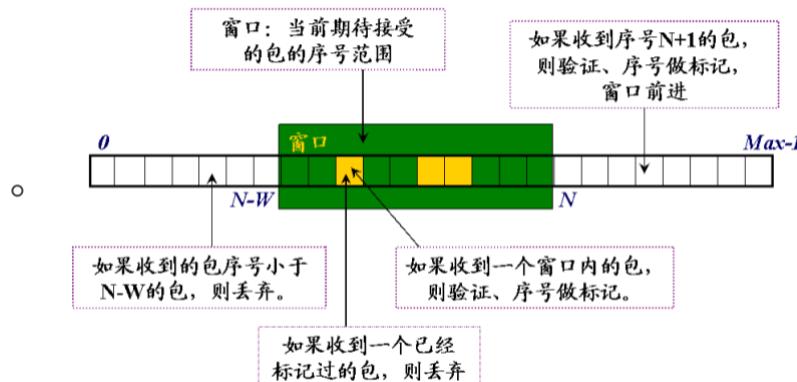
认证头的组成



- Next header(邻接头)
 - 8 bits, 标识数据载荷中的封装方式或协议
 - 例如: TCP/UDP/ICMP、AH、ESP
 - Payload length(有效载荷长度)
 - 8 bits, 以32位字为单位的认证数据字段长度
 - Reserved(保留字)
 - 16 bits, 保留以供将来使用
 - 发送时必需设置为全零
 - 这个值包含在认证数据计算中, 否则被接收方忽略
- Security Parameters Index (安全参数索引)
 - SPI为数据识别安全联合的32位伪随机值, SPI=0被保留, 表明“没有安全关联存在”
- Sequence Number(序列号)
 - 32bits, 单调递增计数器, 用于防范重放类型攻击
- Authentication Data(认证数据AD)
 - 变长域, 必须是32 bits的整数倍
 - 包含完整性校验值ICV或者包的MAC

• 反重放攻击:

- 重放攻击是指攻击者在得到一个经过认证的包后, 又将其传送到目的站点的行为
- 重复接收经过认证的IP包可能会以某种方式中断服务或者产生不可预料的后果, 序列号域就可以防止重放攻击
- 当建立了一个新的SA时, 发送方将序列号初值设为0, 每次在SA上发送一个包, 则计数器加1并将值放入序列号域, 则使用的第一值就是1
- 发送方不允许循环计数, 否则, 同一个序列号就可以产生多个合法的包; 如果该序列号到达 $2^{32}-1$, 则SA必须中止, 用新的密钥协商声称新的SA
- 由于IP协议是无连接的, IPsec认证文档中声明, 接收方应该实现一个大小为w的窗口 (w 的默认值为64), 如下图:

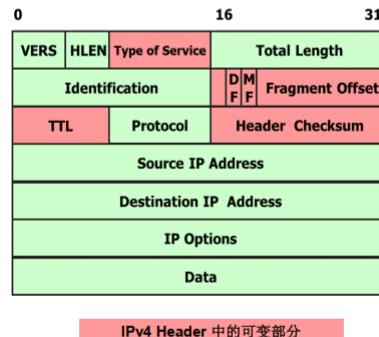


• 完整性校验值:

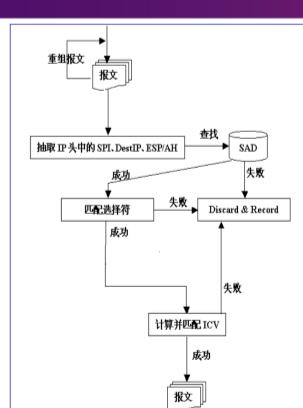
- 认证数据AD域包含完整性校验值ICV, ICV是一种报文认证编码MAC或者MAC算法生成的截断代码

在以下字段计算MAC值:

- IP报头: 传输过程中不变的部分和AH SA终点可以预测的部分参与计算MAC
 - 对于可变部分和不可预测的部分全部置0, 便于在源端和目的端计算
- AH报头中不计算认证数据域; 认证数据域被置成0, 便于在源端和目的端计算
- 整个上层协议数据, 如TCP/UDP数据, 假设在传输过程中不变

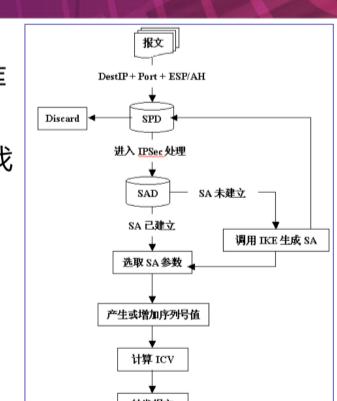


• AH对接收数据包 (Inbound)的处理



- 从端口收到输入的数据包, 解析出其SA三元组, 查找SADB
 - 如查找到一个匹配的SA条目, 将该SA参数与数据包相关域参数进行比较: 参数一致, 处理该数据包; 参数不一致, 丢弃该数据包
 - 如果没有找到匹配的SA条目, 丢弃该数据包
- 检查序列号(可选, 针对重放攻击)
 - 使用滑动窗口来检查序列号重放
- 计算数据包的ICV, 将其和数据包中的值进行比较:
 - 相等, 恢复数据包, 转IP协议栈进行路由
 - 不相等, 丢弃该数据包并审计事件

• AH对输出数据包(Outbound)的处理过程



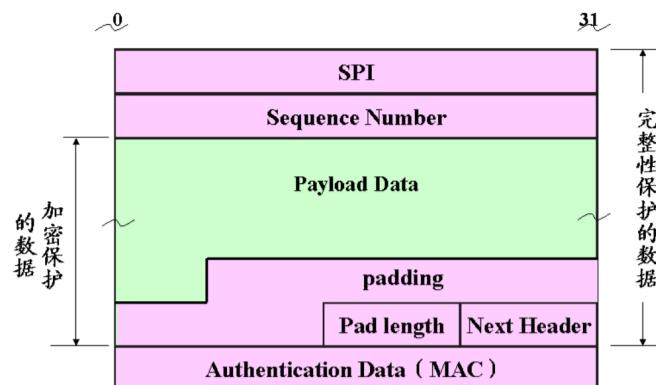
- 从IP协议栈中收到需要转发的数据包, 使用相应的选择子查找安全策略数据库SPDB, 获取对数据包的安全策略
- 如果确定对数据包实施IPsec处理, 查找安全关联数据库SADB
 - 如SA已经建立, 选取参数, 计算ICV, 转发报文
 - 如SA未建立, 调用IKE协商新的SA; 在选取参数, 计算ICV, 转发报文

IPsec:封装安全载荷ESP

- 封装安全载荷ESP提供保密性服务包括报文内容保密和流量限制保密, ESP还可以提供和AH同样的认证服务
- ESP的格式:

● Security Parameters Index (安全参数索引)

- SPI为数据报识别安全联合的32位伪随机值，
- SPI=0被保留，表明“没有安全关联存在”



● Sequence Number (序列号)

- 32bits, 单调递增的计数器，用于防范重放类型的攻击

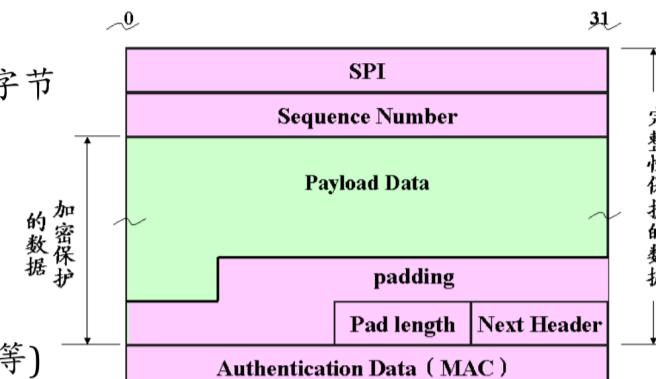
● Padding(填充域)

- 可变长域，范围在0~255字节

● Pad Length(填充域长度)

● Next header(邻接头)

- 8 bits, 标识载荷中的封装方式或协议 (TCP/UDP/ICMP/AH/ESP等)



● Authentication Data (认证数据AD)

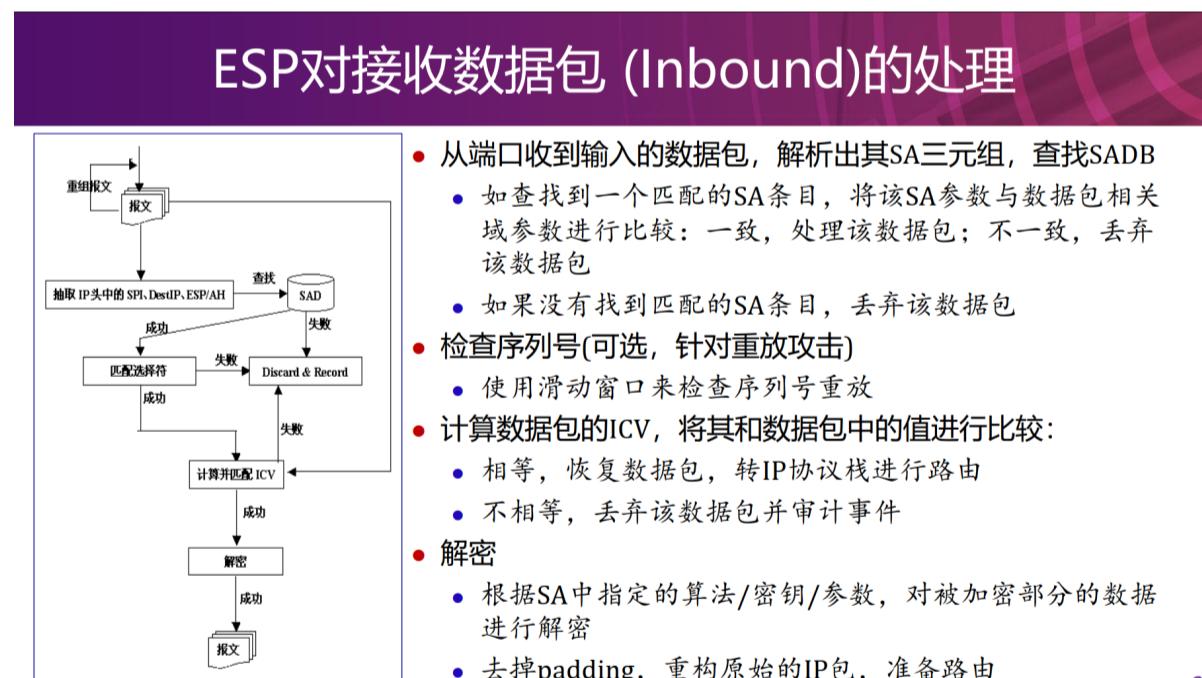
- 变长域，必须是32 bits的整数倍
- 包含完整性校验值ICV或者包的MAC

• ESP的加密算法、认证算法和填充域

- 载荷数据、填充数据、填充长度和邻接头域都在ESP中被加密，可用加密算法有3DES、RC5、IDEA、CAST、Blowfish
- 与AH相同，ESP使用默认截断至96位的MAC
- 填充域的功能如下：

- 如果加密算法需要明文是某个字节的倍数，则填充域可以用于扩展明文长度
- 填充域用来保证ESP格式需要填充长度和邻接头域为右对齐的32位字
- 增加额外的填充域可以隐藏载荷的实际长度，并提供部分流量保护

• ESP对接收数据包的处理：

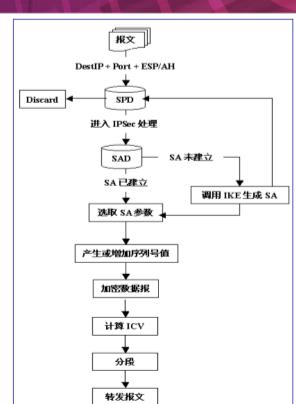


- 从端口收到输入的数据包，解析出其SA三元组，查找SADB
 - 如查找到一个匹配的SA条目，将该SA参数与数据包相关域参数进行比较：一致，处理该数据包；不一致，丢弃该数据包
 - 如果没有找到匹配的SA条目，丢弃该数据包
- 检查序列号(可选，针对重放攻击)
 - 使用滑动窗口来检查序列号重放
- 计算数据包的ICV，将其和数据包中的值进行比较：
 - 相等，恢复数据包，转IP协议栈进行路由
 - 不相等，丢弃该数据包并审计事件
- 解密
 - 根据SA中指定的算法/密钥/参数，对被加密部分的数据进行解密
 - 去掉padding，重构原始的IP包，准备路由

64

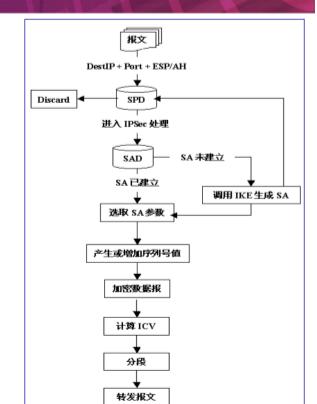
• ESP对输出数据包(Outbound)的处理过程[1]

- 从IP协议栈中收到需要转发的数据包，使用相应的选择子查找安全策略数据库SPDB，获取对数据包的安全策略
- 如果确定对数据包实施IPsec处理，查找安全关联数据库SADB
 - 如果SA已经建立，选取参数，计算ICV，转发报文
 - 如果SA未建立，调用IKE协商新的SA；在选取参数，计算ICV，转发报文
- 产生序列号：防止重放攻击



ESP对输出数据包(Outbound)的处理过程[2]

- 加密
 - 封装必要的数据，放到payload data域中
 - 不同的模式，封装数据的范围不同
 - 增加必要的padding数据
 - 加密操作
- 计算ICV
 - 针对加密后的数据计算ICV
- 分片
 - 根据最大传输单元MTU，将数据包分成适当大小的包发往目的节点



IPsec：安全关联（SA）组合

- 单个SA可以实现AH或者ESP，但是不能全都实现
- 安全关联组合（安全关联束）是指提供特定的IPSec服务集所需的一个SA序列，SA可通过两种方式组合成束：
 - 传输邻接：在不使用隧道的情况下，对一个IP包使用多个安全协议；组合AH和ESP的方法仅允许一级组合
 - 隧道迭代：指通过IP隧道应用多层安全协议；由于每个隧道可以在路径上的不同IPSec节点起始和结束，因此该方法允许多层嵌套
- AH和ESP典型组合

Transport	Tunnel
1. [IP1][AH][upper]	4. [IP2][AH][IP1][upper]
◦ 2. [IP1][ESP][upper]	5. [IP2][ESP][IP1][upper]
3. [IP1][AH][ESP][upper]	

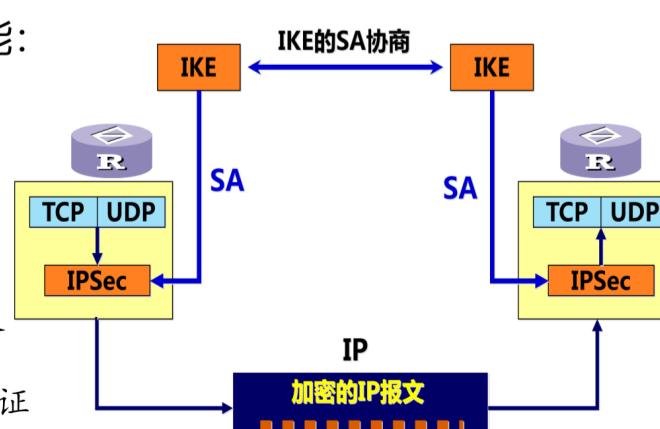
• 这里upper指上层协议数据
• IP1指原来的IP头
• IP2指封装之后的IP头

IKE：为IPsec管理密钥

IKE简介

- IPsec体系结构支持两种密钥管理类型
 - 手工：系统管理员手动为每个系统配置所需各类密钥，SA永远存在，应用于小规模、结构简单的网络
 - 自动：在大型分布系统中，SA通过协商方式产生，SA过期后可以重新协商，适用于较复杂拓扑和较高安性的网络
- IKE(Internet Key Exchange)协议为IPSec提供了自动协商交换密钥、建立安全关联SA的服务，简化了IPSec的使用和管理
- IKE协议解决了在不安全的网络环境中安全地建立或更新共享密钥的问题
- IKE是非常通用的协议，不仅可为IPsec协商安全关联，而且可以为SNMPv3、RIPv2、OSPFv2等要求保密的协议协商安全参数
- 目前IKE协议只在IPsec协议中得到了应用
- IKE的精髓在于：**永远不在不安全的网络上直接传送密钥**，而是通过一系列的数据交换，通信双方最终计算出共享密钥
- 即使黑客截获了双方用于计算密钥的所有交换数据，也不足以计算出真正的密钥，其核心技术就是**DH秘钥交换算法**，使得IKE具备了完善的**前向安全性**
- 完善的前向安全性PFS(Perfect Forward Secrecy)是一种安全特性，指一个密钥被破解，并不影响其他密钥的安全性，这些密钥间没有派生关系
- IKE不但可自动地为参与通信的实体协商安全关联SA，还可以维护安全关联数据库SADB

- IKE为IPsec提供了如下功能：
 - 降低手工配置复杂度
 - 安全关联SA定时更新
 - 密钥定时更新
 - 允许IPSec提供反重放服务
 - 允许在端与端之间动态认证
- IKE和IPsec：
 - SA
 - IP
 - 加密的IP报文



- IKE是一个混合协议，IKE == “ISAKMP格式 + Oakley模式 + SKEME密钥交换”

IKE的报文格式

- IKE的报文格式继承自ISAKMP，所以也称为ISAKMP报文
- ISAKMP可以在任何传输层协议(UDP、TCP)或IP层上实现，利用UDP协议的端口500进行传输
- ISAKMP双方交换的信息以“报文头+载荷”的形式传输，每个ISAKMP报文由一个定长的报文头和不定数量的载荷组成
- ISAKMP报文头：

- Initiator Cookie (32 bits): 发起者Cookie
 - Cookie是一个随机数，发起者和应答者的Cookie一起可以唯一标识一个密钥交換会话和该会话生成的IKE SA

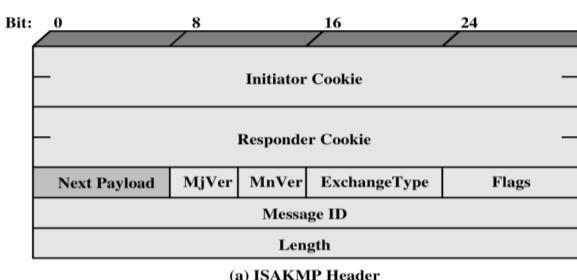
- Responder Cookie (32 bits): 应答者的Cookie

- Next Payload (8 bits): ISAKMP报文的第一个载荷类型

- MjVer (4 bits): 主版本号

- MnVer (4 bits): 次版本号

- Exchange Type (8 bits):
密钥交换的类型



Flags (8 bits): 每个标志位都代表密钥交换的一个特定属性，目前只定义了最低三位，其余位必须被置0

- 第1位是加密位(Encryption Bit)

- 置1表明ISAKMP消息头后面的载荷是被加密的
- 置0则表明是明文传输

- 第2位是约束位(Commit Bit) 用来同步密钥交换

- 如果密钥协商的一方A生成的ISAKMP报文将该位置1，那么对方B在安全关联建立起来后，仍要等待A发送一个安全关联建立成功的通知，才可以利用该安全关联进行保密的数据通信

- 第3位是认证位(Authentication Only Bit)

- 置1表明该ISAKMP消息头后面的载荷是有认证的
- 置0表明该ISAKMP报文没有做加密处理，只是做了认证处理

Message ID (32 bits): 由第二阶段密钥协商的发起者生成的随机数，用来标志一个第二阶段密钥协商会话

Total Message Length (32 bits): ISAKMP报文的总长度，即报文头和所有载荷的总长度，加密可能会导致报文长度增大

- IKE的载荷 (13种)：ISAKMP定义了13种载荷，具有相同格式的载荷头

Next Payload (8 bits):

该载荷的后继载荷类型；如果当前载荷最后一个，则该域被置为0

Reserved (8 bits):

保留字节，置0

Payload Length:

- 载荷长度（以字节为单位），该长度包括载荷头的长度。



- 编号1: SA载荷，用来协商安全属性，指明解释域和状态

- 编号2: Proposal载荷，包含在SA载荷中

- 编号3: Transform载荷，总是包含在Proposal载荷内

- 编号4: Key Exchange载荷，传送密钥的各类信息，可为Oakley/DH等所用

- 编号5: Identification身份认证载荷，传送身份信息

- 编号6: Certificate证书认证载荷，传送证书和相关信息

- 编号7: Certificate Request证书请求载荷，向对方要求证书和相关信息
 - 收到含有Certificate Request载荷的ISAKMP报文的接收方，要用Certificate载荷（编号6）发送它的证书

- 编号8：Hash载荷，传送Hash函数的结果
- 编号9：Signature签名载荷，传送数字签名信息
 - 可以用来对数据完整性进行检查，或者提供不可抵赖服务
- 编号10：Nonce载荷，传送大随机数，可以防止重放攻击
- 编号11：Notification通知载荷
 - 通知对方某些信息，比如报文格式出错，SA生成等
- 编号12：Delete删除载荷，通知对方删除某个或多个SA
- 编号13：Vendor ID供应商载荷，提供了一种扩展手段
 - 密钥交换双方可通过对方的Vendor ID来判断是否可以采用某种扩展

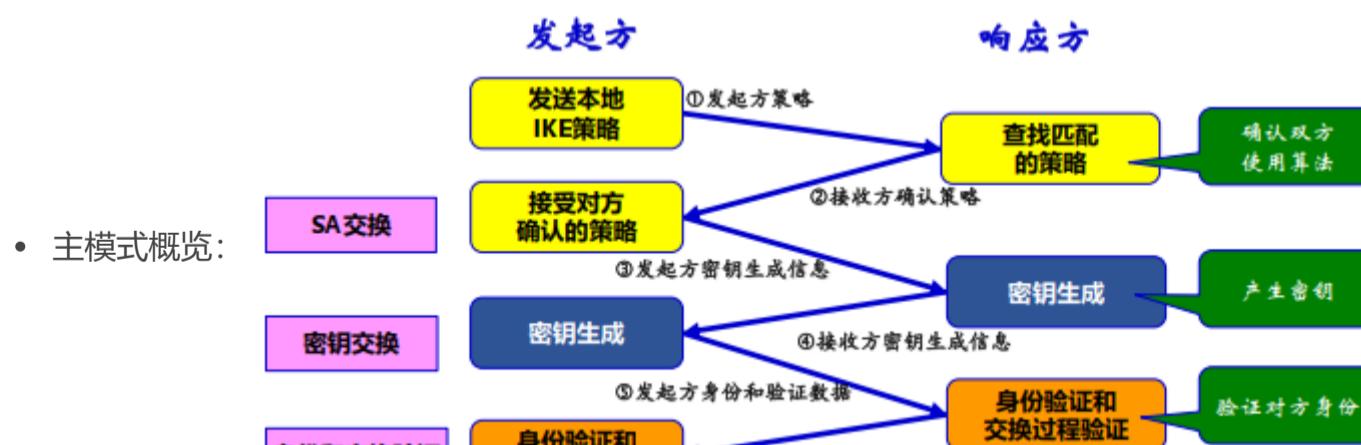
IKE的体系结构

- IKE使用了两个阶段的ISAKMP框架
- 第一阶段，协商创建一个通信信道（IKE SA），并对该信道进行验证，为双方进一步的IKE通信提供机密性、消息完整性以及消息源验证服务
 - 主模式： 6个消息交互
 - 快速模式： 3个消息交互
- 第二阶段，使用已建立的IKE SA建立IPsec SA
 - 快速模式： 3个消息交互
-



IKE的第一阶段：协商IKE SA

- IKE在第一阶段中要协商建立IKE SA，建立一个经过验证的安全通道，为后续的协商提供机密性和完整性保护
 - 必须协商的内容包括：加密算法、哈希算法、认证(验证)方法、进行DH操作所使用组的有关信息等
- IKE第一阶段中可以使用主模式和积极模式，这两种模式只能在第一阶段中使用
 - 主模式提供了对通信双方的身份保护
 - 当身份保护不必要时，可以使用积极模式以减少信息传输的数量，提高协商效率



- 主模式SA交换：

SA交换：消息①和②

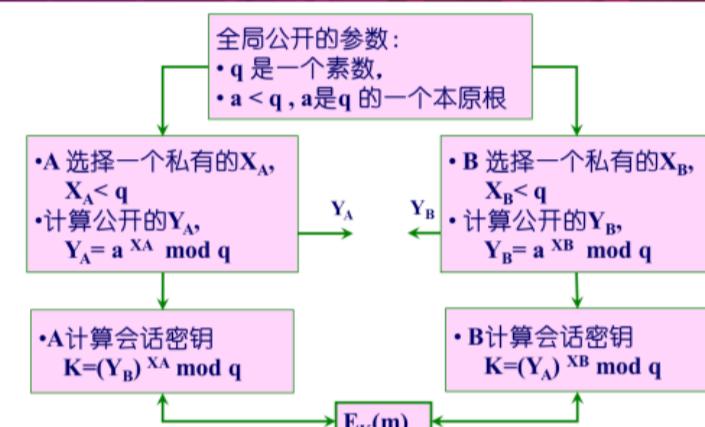
在协商之前，发起者和响应者必须计算产生cookie，用于唯一的标识每个单独的协商交换过程

- cookie使用源/目的IP地址、随机数字、日期和时间等信息进行MD5计算，其结果写入ISAKMP报文头的“cookie”域中

发起者发送消息①，接受者响应消息②；双方就散列函数、加密算法、认证方法、IKE SA协商的时间限制等进行了协商

密钥交换：消息③和④

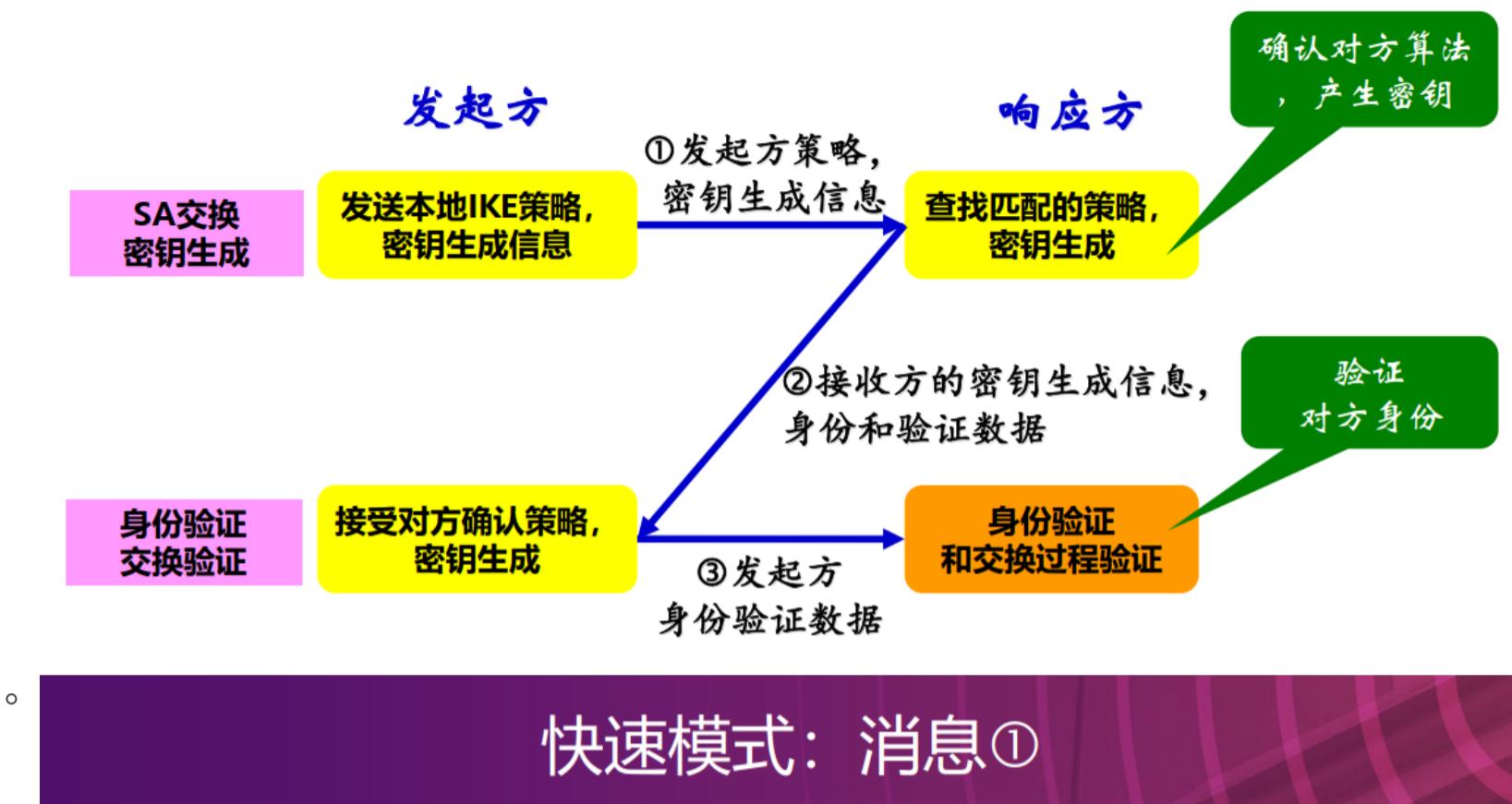
- 发起方和响应方采用Diffie-Hellman密钥交换算法计算共享密钥
- 发起方发送消息③，响应方发送消息④：
 - 消息③④的ISAKMP报文头和消息①和②相同，但载荷不同
 - 消息③包含了Key Exchange载荷和Nonce载荷
 - 消息类型4和消息类型10



身份和交换验证：消息⑤和⑥

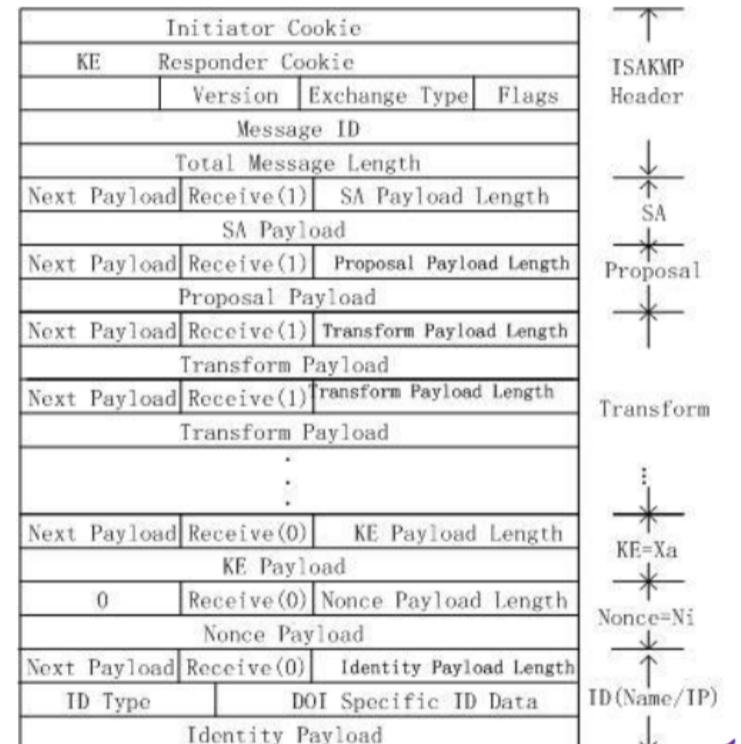
- 发起方发送消息⑤，ISAKMP头还是和前面的一样，但是载荷不同，消息⑤中的载荷包含标识载荷和散列载荷，并且加密传输
 - 身份认证载荷（消息类型5）包含了发起者的标识信息，IP地址或者主机名；散列载荷（消息类型8）包含对上一过程中产生的三组密钥进行Hash运算得出的值
 - 这两个载荷加密传输
- 消息⑥是响应方发送的，包含了响应者相应的信息
- 如果双方在消息⑤和消息⑥中的散列载荷中的hash值相同，那么双方的认证成功→→→ IKE第一阶段协商顺利完成
- 快速模式概览：
 - IKE为什么需要快速模式：
 - 适用于一方地址为动态的情况，主模式不能够应用于IP地址变化的情况
 - 快速模式传输的消息更少，效率更高

IKE的第一阶段：快速模式



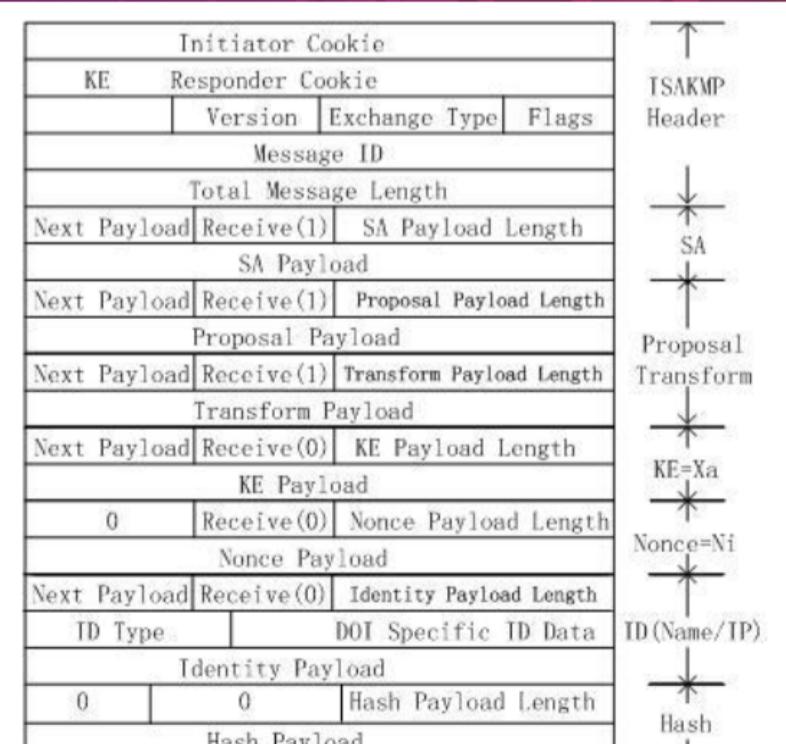
快速模式：消息①

- 在协商之前，发起者发送前计算好cookie
- 消息①交换SA、KEY、Nonce和ID等载荷
- 消息②在响应消息①内容的同时，增加了Hash认证
- 消息③是响应方对发起方的认证



快速模式：消息②

- 消息②在交换消息①内容的同时，增加了Hash认证
- 与主模式中的消息⑤⑥的准备工作一样，响应方计算好几个密钥，然后使用这些密钥以及其他信息计算hash值
- 消息②回答了消息①的相应内容，还包括了计算好的hash值



快速模式：消息③

- 发送方根据消息②传来的密钥，计算Hash值；如果这个Hash值和消息②中的Hash值一致，则发送方验证响应方成功；然后，再计算一个新的Hash值，加载在消息③中，发送给响应方
- 响应方收到了消息③后，计算自己的Hash值，与消息③的Hash值进行比对，如果一致，认证成功
- 快速模式顺利完成

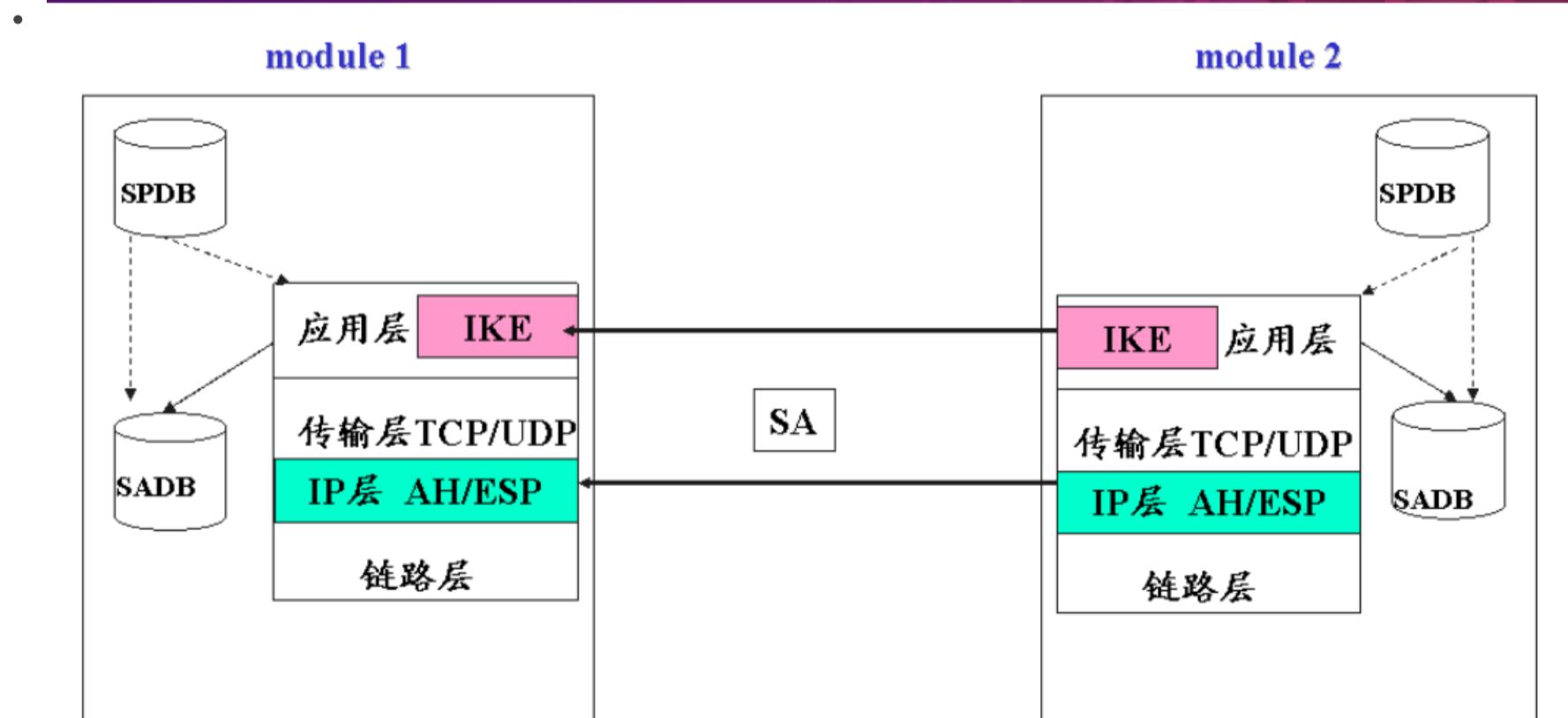
Initiator Cookie			
KE	Responder Cookie		
	Version	Exchange Type	Flags
Message ID			
Total Message Length			
0	0	Hash Payload Length	
Hash Payload			

↑ ISAKMP Header
↓ Hash

IKE的第二阶段：建立IPsec SA

- 一个IKE SA协商（第一阶段）可为多个IPsec SA协商（第二阶段）提供服务
- 第二阶段协商的内容与所协商的安全协议等相关，例如：IPsec AH 协议需要协商认证算法，IPsec ESP 协议需要协商认证和加密算法
- 第二阶段中使用快速模式进行信息交换，一个第二阶段协商可以建立多个安全关联
- 需要协商的参数：
 - 加密算法：包括DES、IDEA、Blowfish、3DES、CAST等
 - 哈希算法：包括MD5、SHA、Tiger等
 - 验证方法：包括共享密钥、RSA签名、DSS签名、RSA加密、RSA加密等
 - DH组
 - 存活周期类型及长度：包括秒、千字节等
 - 密钥长度

IKE的工作模式



- IKE在协议实现上，是以守护进程的方式在后台运行
- 两个守护进程通过UDP协议（端口500）来传递消息
- IKE协议使用两个数据库：安全关联数据库SADB和安全策略数据库SPDB，这两个数据库都保存在操作系统内核
- 可以通过两种方式来启动IKE服务：

- 由内核提交创建IKE SA请求
- 同级IKE守护进程提交协商SA请求
- SPDB在每个条目中隐藏有指针：无论对于外出IP包还是进入IP包，首先都要对SPDB表进行查询，以决定是否丢弃、绕过或应用IPsec
 - IPsec查询SADB，检查是否拥有合适的SA
 - 如果有，则进行相应的IPsec处理
 - 如果没有，就会向IKE守护进程发出创建SA请求
- IKE守护进程接收到内核发来的请求后，查询SPDB，得到所有协商参数；然后向远程IKE进程发出协商请求；IKE进程开始协商
 - 如果协商成功，把新协商的SA增加到SADB中
 - 如果因为SPDB参数问题，协商未成功，IKE进程给策略及SA管理模块提示，由管理员来配置SPDB参数
- 当管理员指示IKE守护进程不再使用某个SA时，IKE守护进程会从SADB中删除掉相应的记录SA，同时会向远地的IKE守护进程发送删除信息，表示本地已经不再使用此SA了

IKE协议总结

IKE协议的不足

- IKE是一组复杂混合协议集合，虽然发展成为Internet标准，但是主要用途局限于为IPsec通信双方建立安全关联SA
- 标准定义的复杂性（需要参考3个RFC）容易导致理解上的困难和实现上的混乱，最终会导致不同实现间的互操作困难
- 完成协商的消息往返次数过多，这样会消耗较多的计算以及网络带宽资源
- 容易受到各种攻击，由于设计缺陷，协议实现容易受到拒绝服务攻击、中间人攻击、重放攻击等多种攻击行为的威胁

Lettuce9 SSL-HTTPS-SET

传输层安全协议SSL

概述

- 网络层安全协议IPsec可以提供端到端的网络层安全传输，但是无法处理位于同一端系统之中的不同应用的安全需求，因此需要在传输层和更高层提供网络安全传输服务，来满足这些要求
- 常用协议包括：SP4、TLSP、SSH、SSL：
 - SP4：从属于安全数据网络系统SDNS，由NSA与NIST开发
 - SSH：通过强制认证+数据加密 实现 安全登陆+安全传输
 - TLSP：ISO开发和标准化的协议，通信加密+完整性验证
 - SSL：安全套接层安全机制
- 安全套接层协议SSL专门用于保护基于WEB的通信：
 - 服务器认证
 - 客户认证（可选）

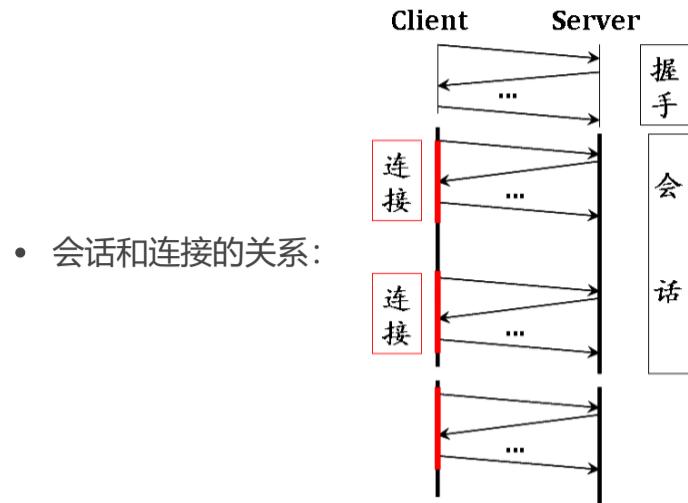
- SSL链路上的数据完整性和SSL链路上的数据保密性
- SSL的设计目标:
 - SSL工作在TCP协议上
 - SSL与应用层协议无关， SSL协议可用于保护正常运行于TCP之上的任何应用层协议
 - 在应用层协议传输之前， SSL协议已经完成了客户端和服务器的身份认证、加密算法和密钥的协商；在此之后，双方建立起了一条安全可信的通信信道，应用层协议所传送的所有数据都会被加密，从而保证了安全
- SSL协议概述:
 - SSL为端到端的应用提供保密性、完整性、身份认证等安全服务
 - 机密性保护
 - 完整性保护
 - 认证保护
- SSL的使用案例:

如果利用SSL来访问网页，其步骤如下：

- 用户：浏览器里输入 <https://www.sslserver.com>
- HTTP层：将用户需求翻译成HTTP请求
- SSL层：借助下层协议的信道安全的协商出一份加密密钥，并用此密钥来加密HTTP请求
- TCP层：与web server的443端口建立连接，传递SSL处理后的数据；接收端与此过程相反
- SSL在TCP之上建立了一个加密通道，通过这一层的数据经过了加密，因此达到保密的效果

SSL体系结构

- 两个实体：客户机+服务器
- 两个概念：会话+连接
 - 会话是虚拟连接，连接是特定通道，一个会话协商可以由多个连接共享
- 两层协议：握手协议+记录协议
 - 握手协议：认证+协商：算法、密钥、`secrets`、初始向量等
 - 记录协议服务：数据封装+安全通信
- 会话：
 - 会话是客户端和服务器之间的一个关联(*association*)，即一个虚拟的连接关系
 - 会话通过握手协议建立，用以协商密码算法、主密钥等，一个会话协商可以由多个连接共享
- 连接：
 - 连接是一个特定的通信信道，通常映射成一个TCP连接
 - 连接一般是短暂的，比如HTTPS中的一次访问中可能需要多个连接，共享同一个会话协商的密码算法、主密钥等信息

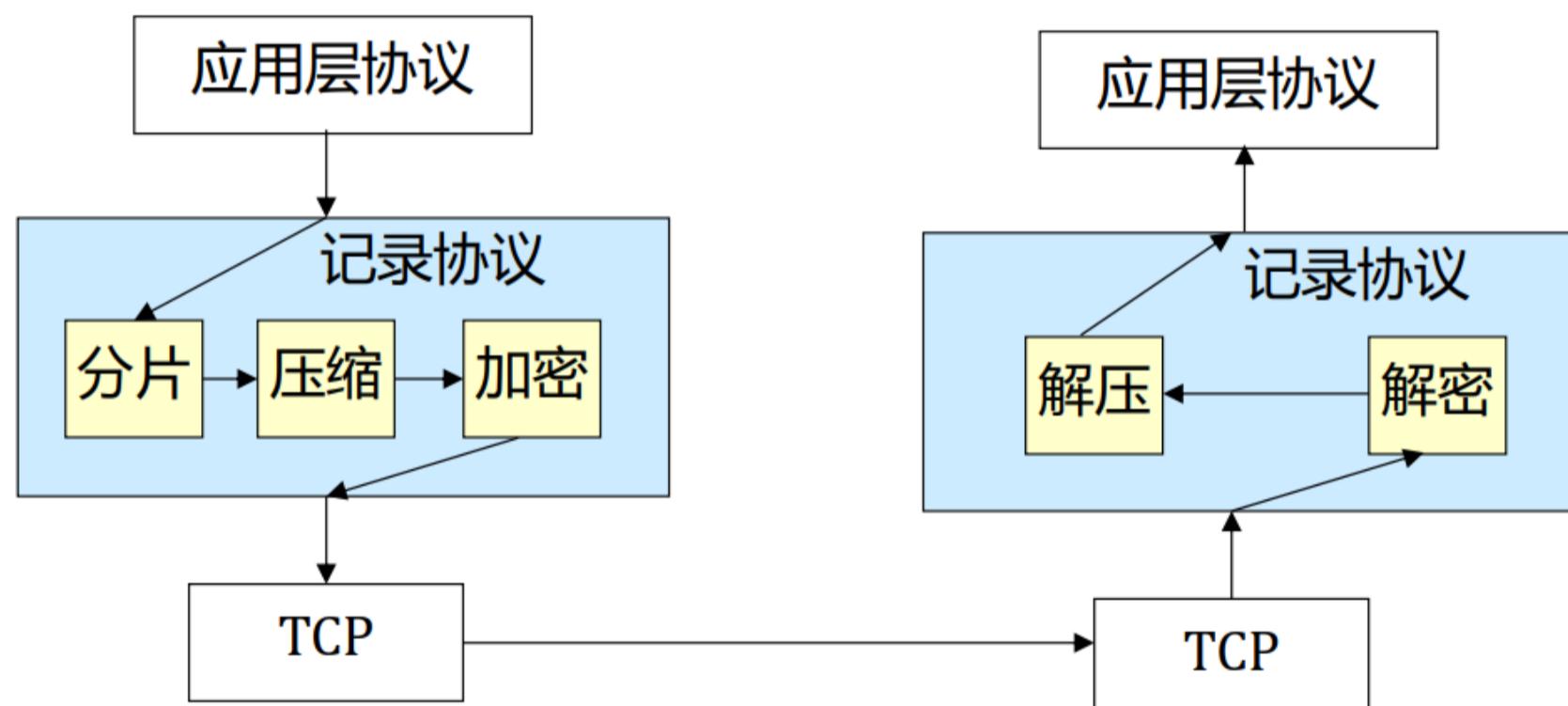


- 会话和连接的关系：

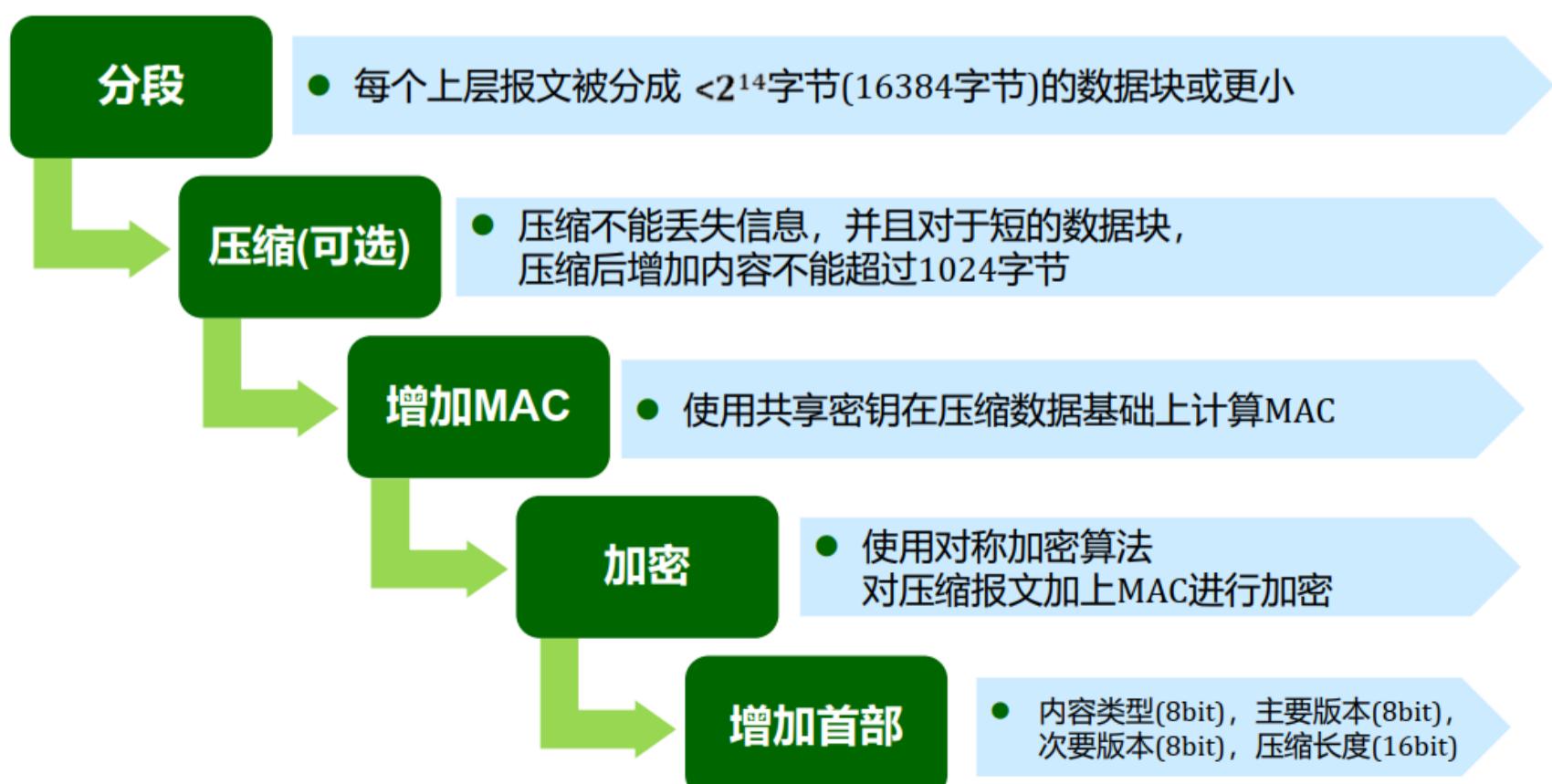
- SSL协议由以下协议组成：
 - SSL记录协议：定义传输格式，为高层协议提供数据封装、压缩、加密等基本功能
 - SSL握手协议：协商密钥，在数据传输前，进行身份认证、协商加密算法、交换密钥
 - SSL告警协议
 - SSL修改密码规约协议

SSL记录协议

- SSL记录协议为SSL连接提供两种服务
- 保密性**：使用SSL握手协议定义的共享密钥，用传统密钥算法对SSL载荷进行加密
- 报文完整性**：用握手协议定义的共享密钥计算报文验证码
- SSL记录协议的工作流程：



- SSL记录协议的操作流程：



- 最后，形成SSL记录协议数据单元

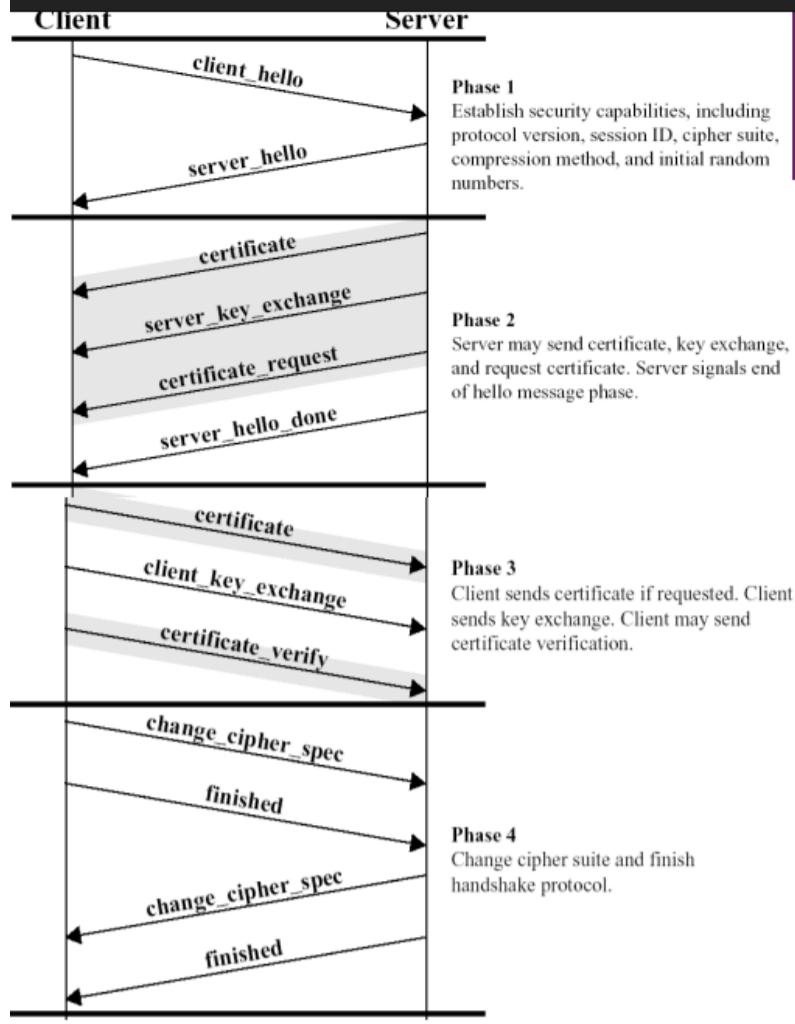
SSL握手协议

- SSL握手协议：
 - SSL的部分复杂性来自于握手协议，握手协议在传递应用数据之前使用
 - SSL握手协议允许客户端和服务器端相互认证、协商加密和MAC算法，保护数据使用密钥通过SSL记录传送
 - SSL握手协议的功能包括：
 - 协商密码算法
 - 协商主会话密钥
 - Client 和Server之间的相互认证：对服务器身份的认证要在Client身份认证之前； Client身份认证是可选的
- SSL握手协议报文格式
 - 握手协议由一系列在客户端和服务器之间交换的报文组成，完成认证、密钥和算法协商
 - 每个报文具有三个字段：
 - 类型(1字节)：指示10种报文中的一个
 - 长度(3字节)：以字节为单位的报文长度
 - 内容(>=1字节)：和这个报文有关的参数
- 握手协议消息类型：

消息	参数	描述
hello_request	Null	服务器发出此信息给客户端启动握手协议
client_hello	版本，随机数，会话id，密码参数，压缩方法	客户端发出 client_hello 启动SSL会话，该信息标识密码和压缩方法列表，服务器响应
server_hello		
certificate	X.509 v3证书链	服务器发出的向客户端验证自己的消息
server_key_exchange	参数，签名	密钥交换
certificate_request	类型， CAs	服务器要求客户端认证
server_done	Null	指示服务器的 Hello 消息发送完毕
certificate_verify	签名	对客户证书进行验证
client_key_exchange	参数，签名	密钥交换
finished	Hash值	验证密钥交换和鉴别过程是成功的

- SSL握手协议阶段：

SSL 握手协议



● 四个阶段

- Phase 1. Establish security Capabilities
- Phase 2. Server Authentication and Key Exchange
- Phase 3. Client Authentication and Key Exchange
- Phase 4. Finish

- 阶段1：建立安全能力

客户向服务器发送client_hello报文

- 参数包括：Version, Random , session id, cipher suite , compression method

服务器向客户端发送server_hello报文

- 参数与client_hello参数相同

Cipher suite 参数包括

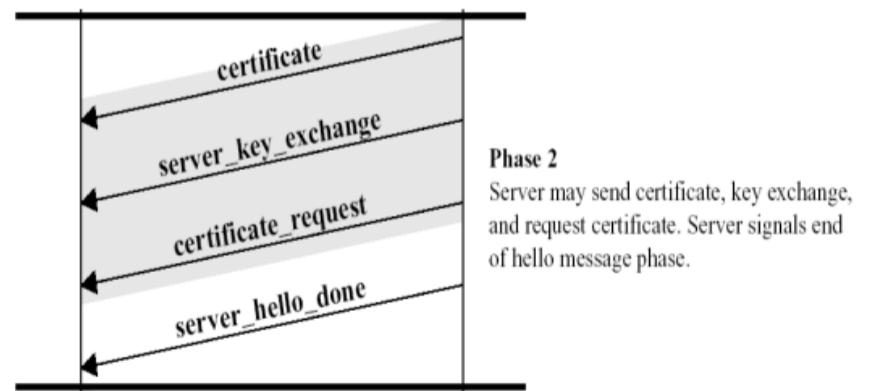
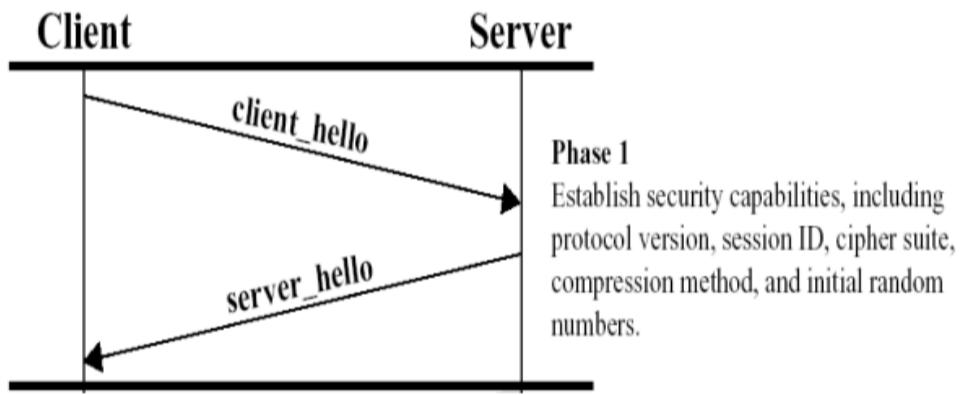
- 密钥交换方法：
RSA, Diffie-Hellman, Fortezza
- 加密算法：RC4, RC2, DES, 3DES, IDEA, Fortezza
- MAC算法：MD5, SHA-1

- 阶段2：服务器认证和密钥交换

服务器发送其SSL数字证书

Certificate，一般服务器使用带有SSL的X.509v3数字证书

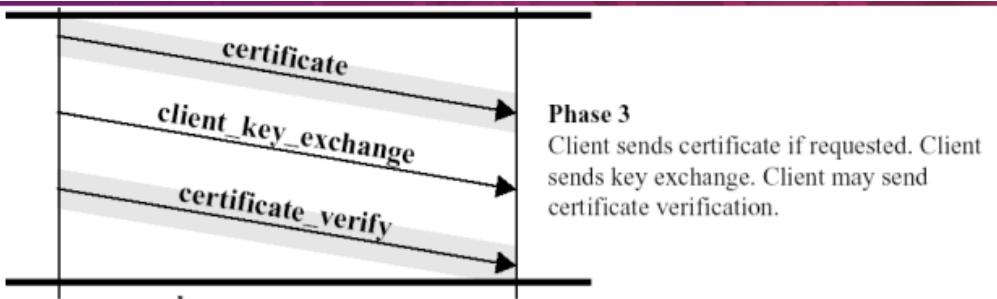
- (可选) 如果服务器使用 SSL 3.0, 服务器的web应用程序需要客户端提交数字证书进行认证，则发出certificate_request报文；在报文中，服务器给出可以支持的客户端数字证书类型列表



如果密钥交换算法使用匿名DH算法，就不需要服务器发送证书，但是很难防止中间人攻击；如果密钥交换算法使用DH、RSA等算法，需要发送server_key_exchange 报文，用来交换密钥
服务器发送sever_hello_done报文，等待客户端响应

- 阶段3：客户认证和密钥交换

收到服务器发来的 sever_hello_done 以后，客户端验证 server 提供的证书是否有效



- （可选）如果客户端收到了 certificate-request 报文，则发送一个SSL数字证书 Certificate；如果没有可用的数字证书，客户端将发送 no_certificate alert；如果客户端数字证书认证是强制性的，服务器应用程序将会使会话失败

客户端发送 client_key_exchange，此消息包含 pre_master_secret 和消息验证码密钥，在后续阶段，pre_master_secret 用来计算 master_secret

如果客户端发送了 Certificate 给服务器，客户端将发出签有客户端专用密钥的 certificate_verify，通过验证此消息的签名，服务器可以显示验证客户端数字证书的所有权

认证密钥和加密密钥

- 主密钥 master_secret 并不直接用于数据加密和认证，而是用来产生连接所需要的一系列密钥，包括：

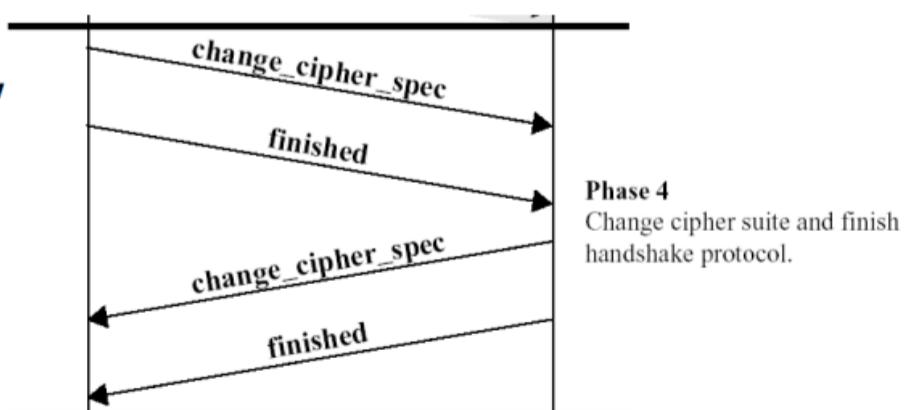
	Client	Server
MAC	client_write_secret	server_write_secret
加密	client_write_key	server_write_key
IV	client_write_iv	server_write_iv

- 阶段4：结束

客户端使用一系列加密运算将 pre_master_secret 转化为 master secret，派生出用于加密和消息认证的所有密钥

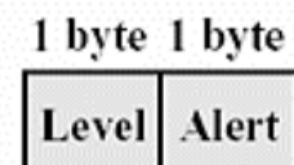
客户端发出 change_cipher_spec，服务器转换为新协商的密码对
客户发送在新算法、密钥的 finished，验证密钥交换和认证过程是否成功

服务器发送 change_cipher_spec，
将挂起状态迁移到当前的 cipher_spec，并发送结束报文
握手完成，客户和服务端可以交换应用层数据



SSL告警协议

- 告警协议用于向对等实体传递SSL相关的警报，和使用SSL的其它应用一样，告警消息按照当前状态压缩和加密
- 此协议的每个消息由两个字节组成
 - 第一个字节
 - 值为1标识告警
 - 值2表示致命错误来传递消息出错的严重程度
 - 如果结果为致命，SSL将立即中止会话中的其它连接将继续进行，但不会在此会话中建立新连接
 - 第二个字节，包含了描述特定告警信息的代码



(b) Alert Protocol

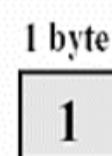
SSL修改密码规约协议

• 功能

- 在握手协议的结束阶段发送，通知接收方，以后的记录将使用刚才协商的密码算法和密钥进行加密/认证
- 接收方把会话的挂起状态复制到当前状态，使以后的连接使用这些密码参数

• 报文结构

- 只有一种报文，只有一个字节，只有一个值(1)是有效的



(a) Change Cipher Spec Protocol

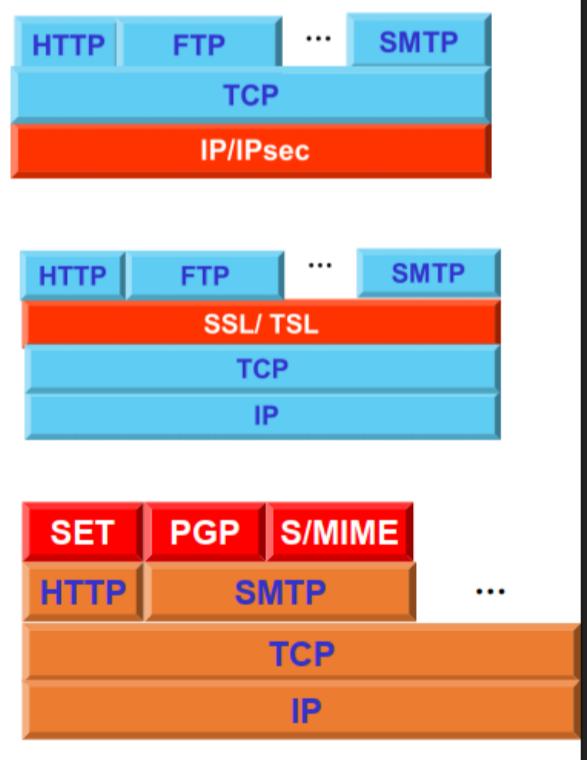
- SSL安全性分析：

- 一个安全协议除了所采用的加密算法安全性以外，更为关键的是其逻辑严密性、完整性、正确性，SSL比较好地解决了这一问题
 - SSL协议可以很好地防范“重放攻击”
 - SSL协议为每次安全连接产生一个128位长的随机数“连接序号”，攻击者无法提前预测此连接序号，因此不能对服务器请求做出正确应答
 - 几乎所有Web服务器以及各类操作系统上的web浏览器支持SSL协议，因此使用SSL协议开发成本小
- 保密问题：SSL协议的数据安全性其实建立在RSA等算法的安全性上，从本质上讲，攻破RSA等算法就等同于攻破此协议
- 认证问题：SSL对应用层不透明，只能提供交易中客户与服务器间的双方认证，在涉及多方的电子交易中，SSL协议并不能协调各方间的安全传输和信任关系

应用层安全协议：HTTPS

WEB及其安全威胁

- 对于Web的安全威胁有两种分类方法：
 - 按照威胁方式分类：
 - 主动攻击
 - 被动攻击
 - 按照威胁位置分类
 - Web服务器安全
 - Web客户端安全
 - 服务器和客户端之间的通信安全
 - Web流量安全方法：
- 提供Web安全的方法1：IP级安全
 - 使用IPSec的优点在于它对终端用户和应用都是透明的，提供通用的解决方案，而且还具有过滤功能。
 - 提供Web安全的方法2：TCP级安全
 - 传输层安全协议SSL或TLS作为下层协议可以对应用透明，也可以在特定包中使用
 - 特定的安全服务在特定应用中实现，即方法3：应用级安全
 - 为应用定制安全协议，一个典型应用就是安全电子交易SET (Security Electronic Trade)



针对HTTP的攻击举例

- 监听嗅探
- 篡改劫持
- 伪造服务器
- 中间人攻击 (Man-In-The-Middle attack) 指攻击者与通讯双方分别建立独立的联系，并交换其所收到的数据
 - 地址解析协议ARP (Address Resolution Protocol)：工作在数据链路层，实现MAC地址与IP地址的映射

- ARP欺骗：ARP欺骗是一种中间人攻击的常用手段，攻击者可以通过制造伪造的ARP frame(request, reply)，修改网内任何计算机的映射表，从而切断目标主机的网络通讯，窃取目标主机关键信息

HTTPS=HTTP+SSL

- HTTPS是HTTP和SSL/TSL协议的合并，解决了数据加密、完整性校验、对服务器的身份认证等问题

- **HTTPS和HTTP在安全方面区别包括：**

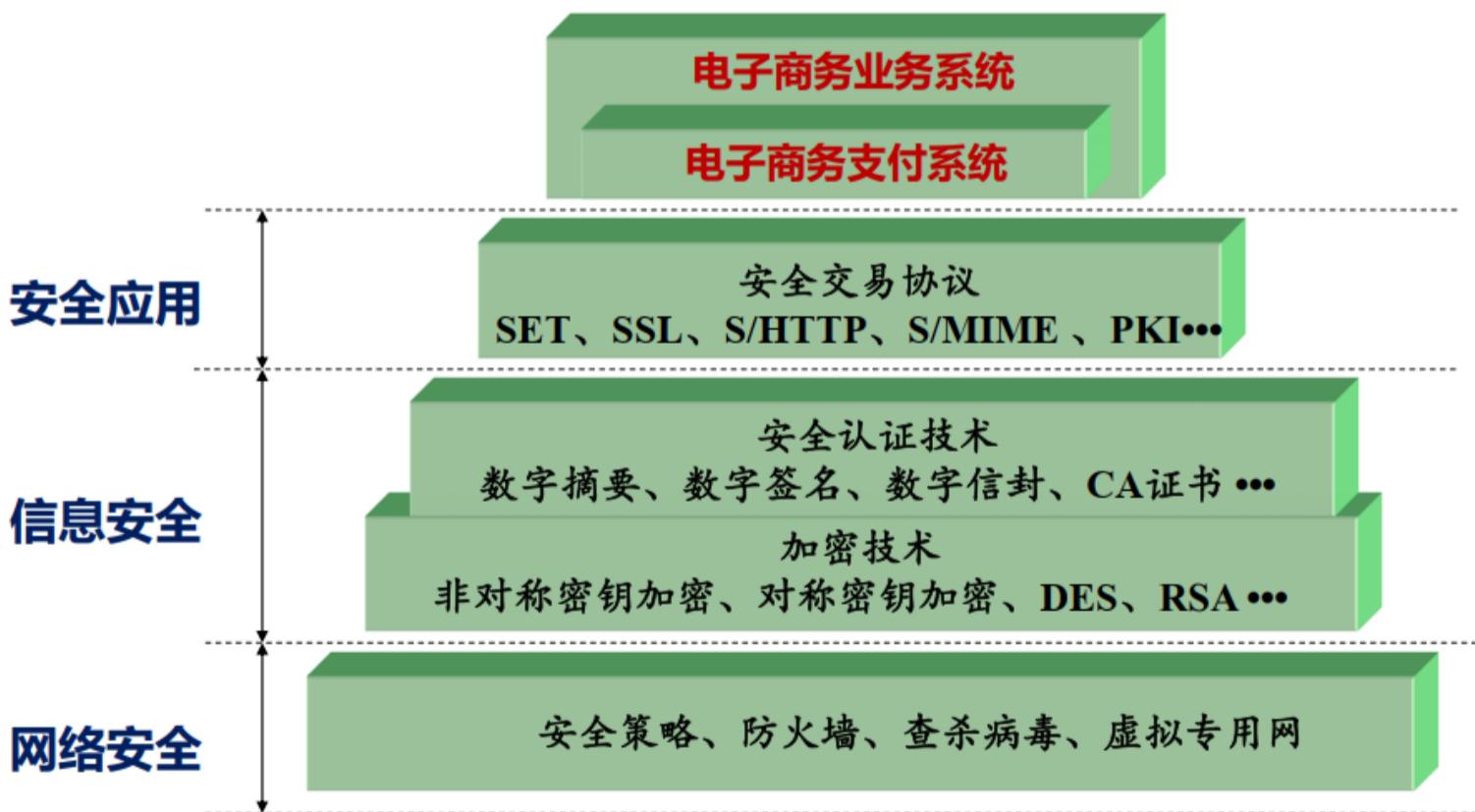
HTTP	HTTPS
HTTP协议采用明文传输	HTTPS协议采用SSL协议加密传输
HTTP端口号为80	HTTPS端口号为443
HTTP协议是简单的无状态连接	HTTPS协议是由SSL+HTTP协议构建的可进行加密传输、身份认证的连接

SSL能否保证HTTPS安全

- SSLStrip：针对https的攻击

安全电子交易协议SET

电子商务安全技术结构示意图



- 安全电子交易协议SET(Secure Electronic Transactions)是由世界上两大信用卡商Visa和Master Card联合制定的实现网上信用卡交易的模型和规范
- SET协议基于X.509v3证书的身份认证，保证交易信息的私密性、保密性、完整性、抗抵赖

SET协议的目标

- 真实性
- 保密性
- 隐私
- 实时性

SET协议的关键特征

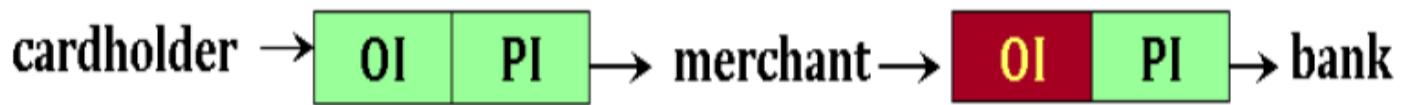
- 信息的机密性
- 数据的完整性
- 卡用户账号的认证
- 商家的认证

SET协议的参与方

- 发卡方
- 持卡人
- 商家
- 证书权威
- 支付网关
- 收款行

支付处理流程Step1：购买请求

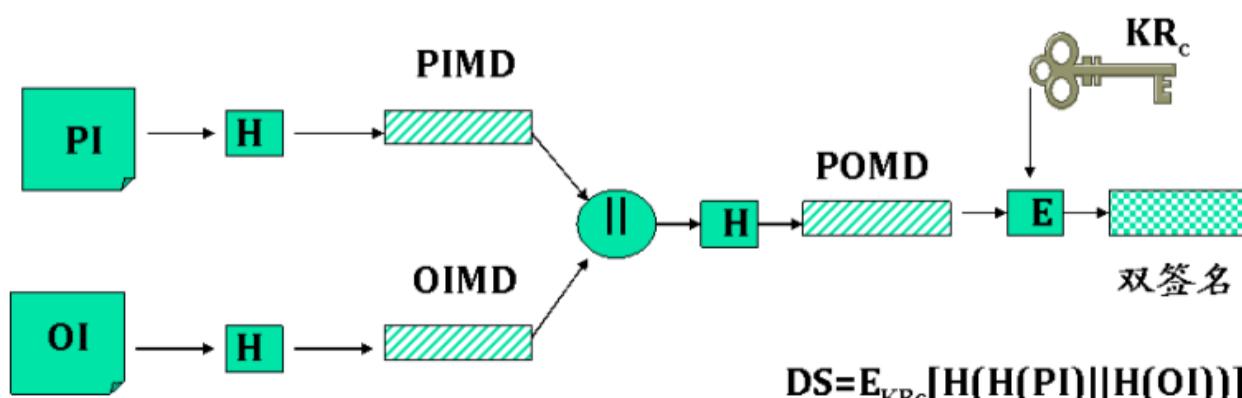
购买请求交换由四个报文组成：发起请求、发起响应、购买请求和购买响应



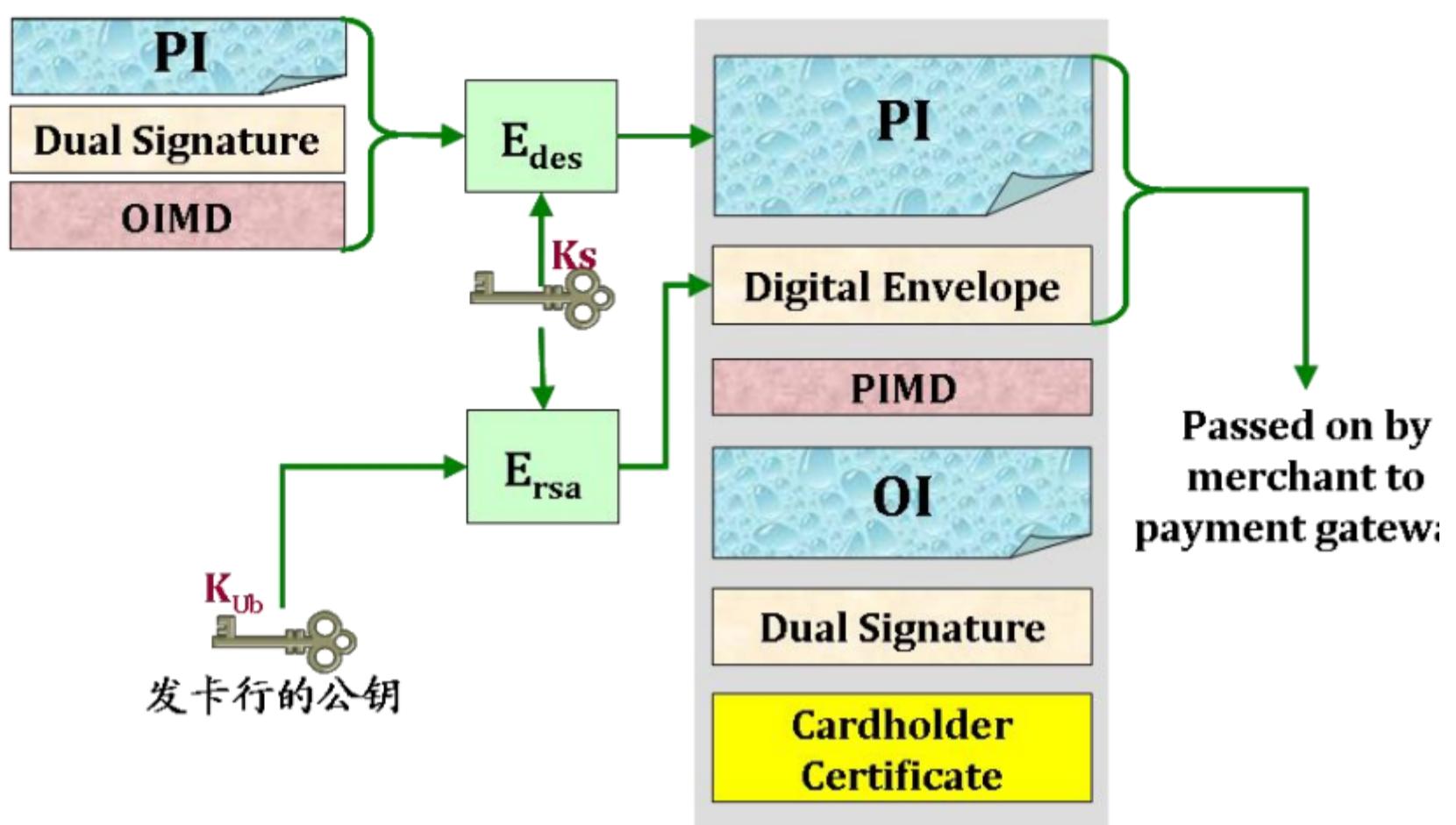
- Cardholder → Merchant : PI, OI
- Merchant → Acquirer : PI
 - Payment Information(PI) : Card number , money, etc.
 - Order Information(OI) : Order details
 - PI 和 OI 必须分开加密和签名，以保证用户的隐私不被泄漏
 - PI 和 OI 必须有联系，以防止商家篡改信息，产生纠纷

双签名

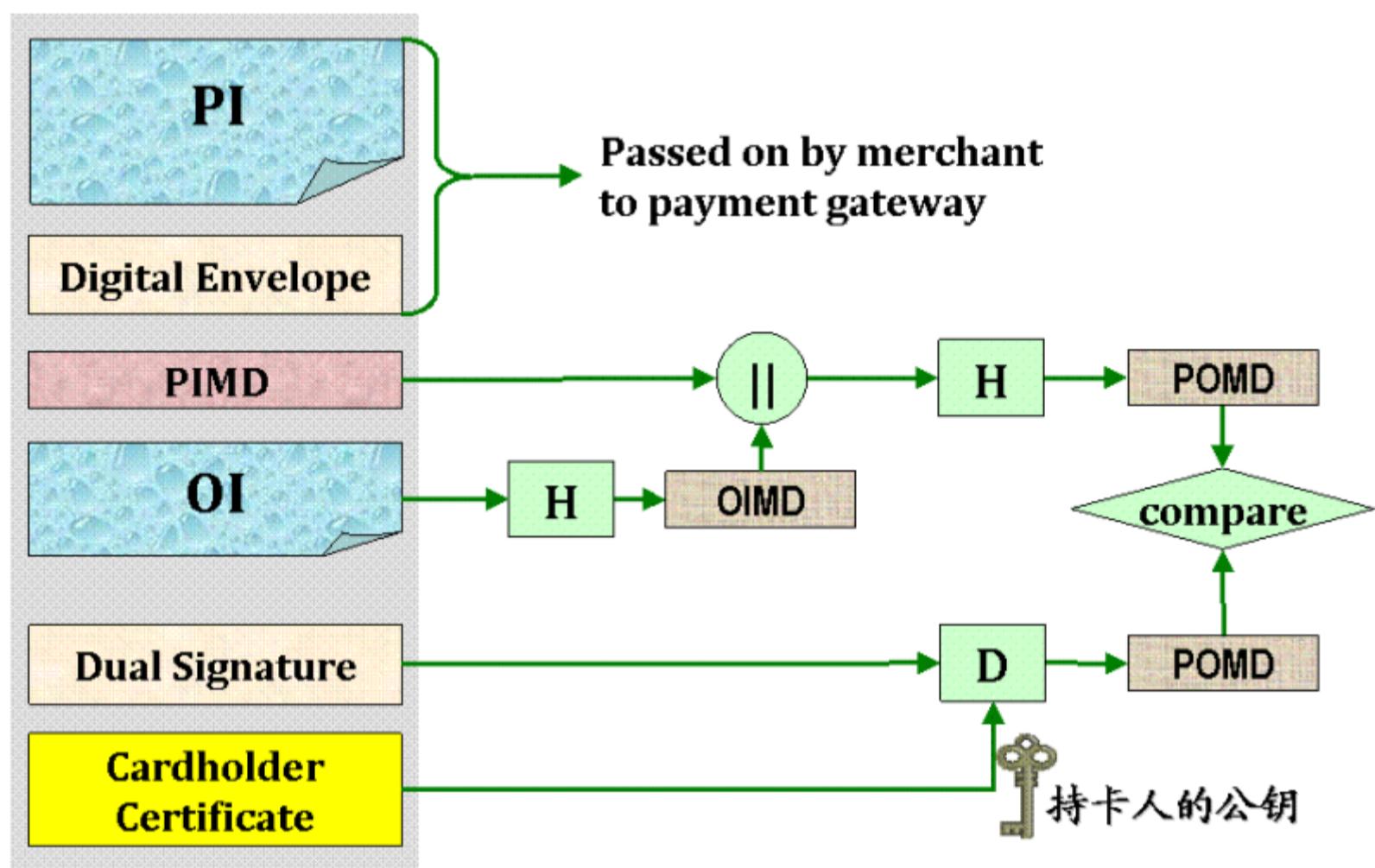
- 双签名的目的是为了连接两个发送给不同接收者的报文
 - PI=支付信息
 - PIMD=PI报文摘要
 - OI=定购信息
- OIMD=OI报文摘要
- H=散列函数（SHA-1）
- E=加密（RSA）
- ||=拼接
- KRc=顾客的私有签名密钥



客户生成购买请求



商家验证用户的订单



支付处理流程Step2: 支付授权

- 商家需要通过支付网关、发卡行得到授权，才能给用户发货
- 支付授权交换由两个报文组成：授权请求和授权响应
- 授权请求报文有以下几部分组成：
 - 与购买有关的信息： PI， 双签名

- 与授权有关的信息：Es
 - 证书：客户、商家
- 授权响应报文由以下几部分组成
 - 与授权有关的信息/获取权标信息/证书

支付网关对支付授权请求的处理

- 验证所有证书的合法性
- 解密数字信封，获得会话密钥，解密authorization block
- 验证商家对authorization block 的数字签名
- 解密Payment Block的数字信封，解密支付信息
- 验证双签名
- 验证transiaction ID 与PI中的是否一致
- 从发卡行申请支付

支付处理流程Step3：支付获取

- 商家只有通过支付获取，才能完成银行的转帐业务
- 获取请求报文由以下几部分组成：
 - 支付的数量、交易ID、获取权标、商人的签名密钥、证书
- 支付获取由获取请求和获取响应报文组成
- 获取响应报文由以下几部分组成：
 - 网关的签名、加密获取相应数据块、网关签名密钥证书