cshao@gdut.edu.cn

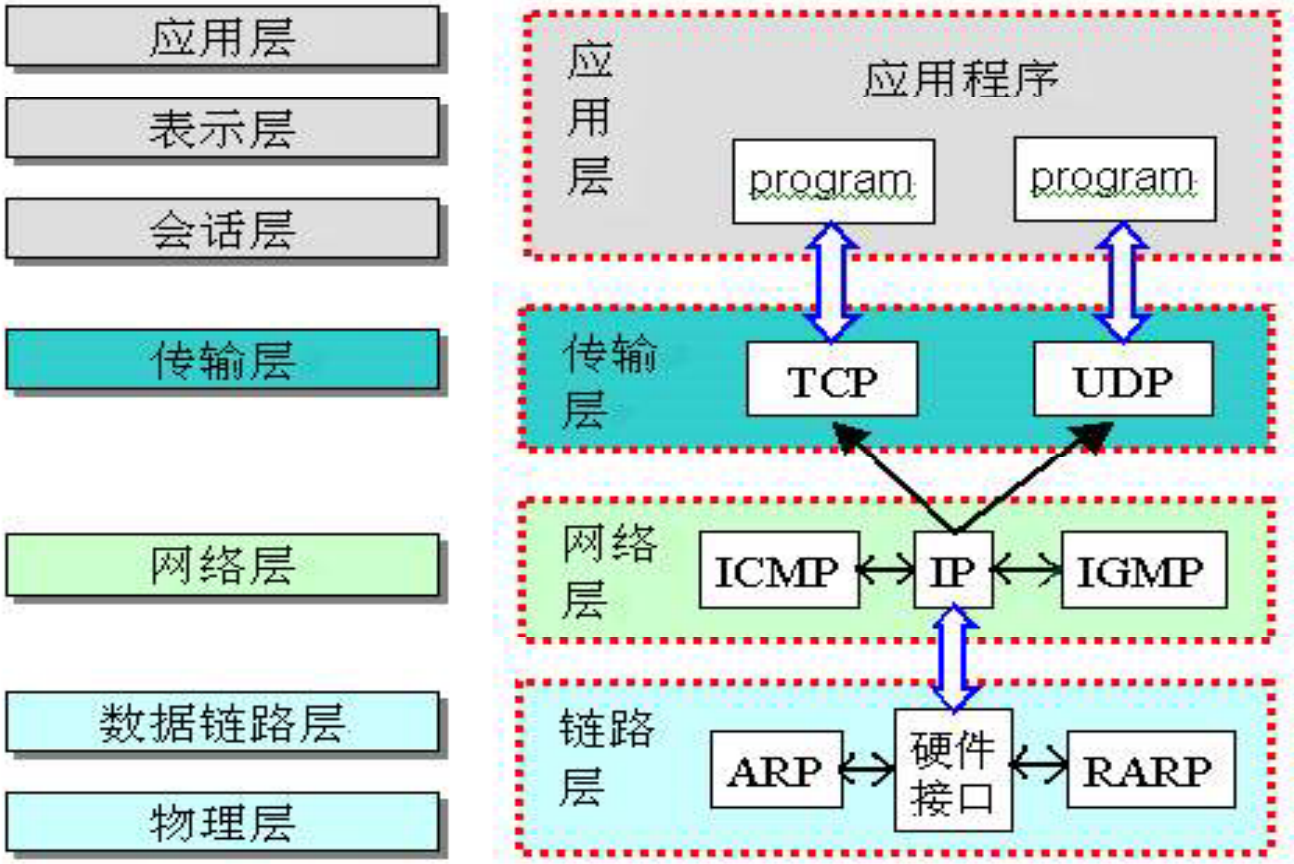
第四章 网络安全协议

第四章 网络安全协议

- 因特网与TCP/IP安全
- SSL协议
- SET协议
- IPSec协议

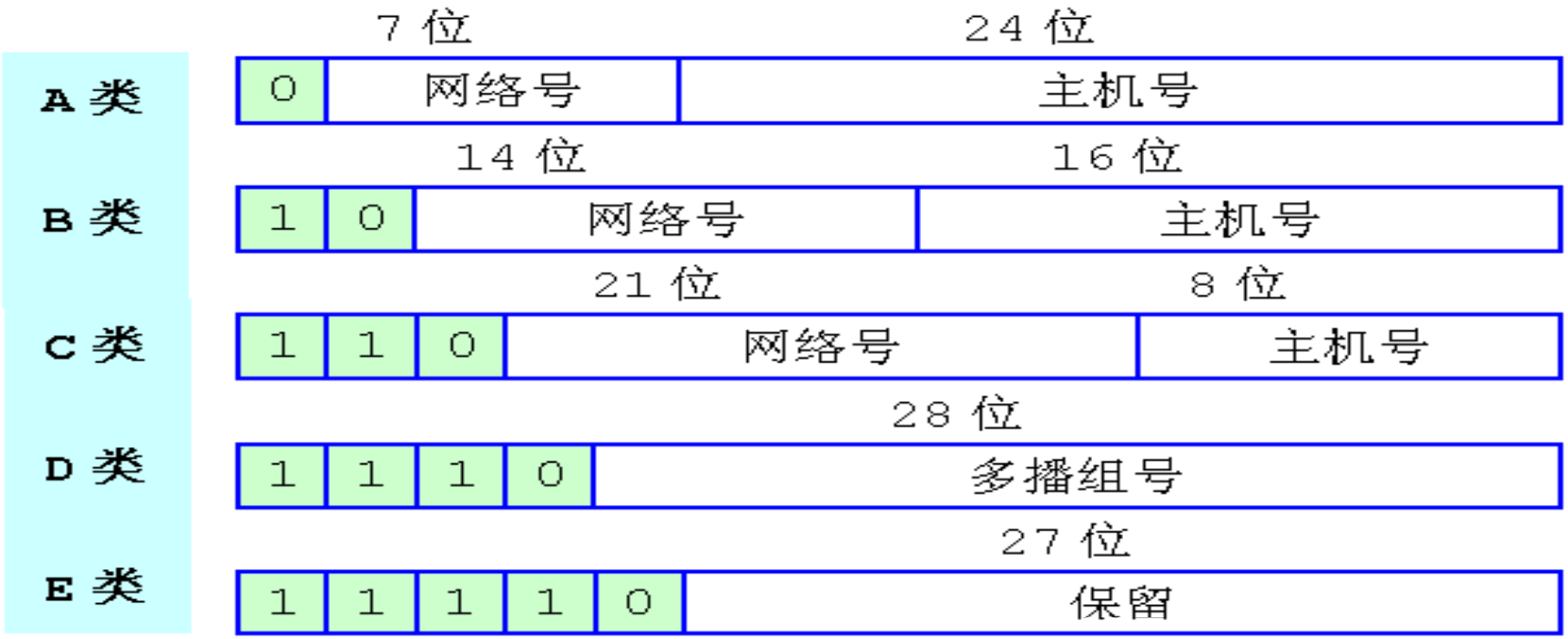
因特网与TCP/IP安全-1. TCP/IP协议栈

- **TCP/IP是一组通信协议的缩写**
- **ISO/OSI模型及其与TCP/IP的关系**



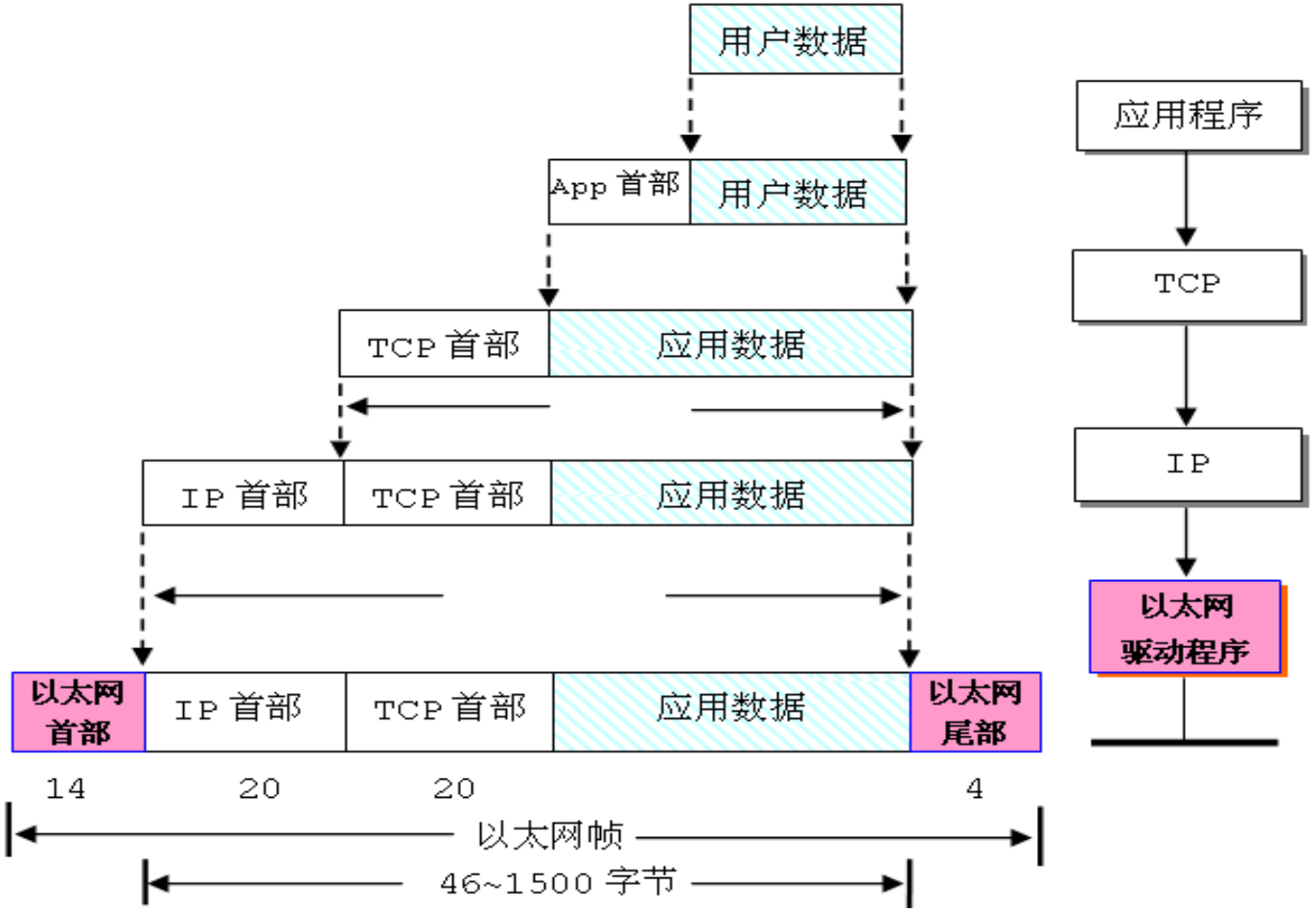
因特网与TCP/IP安全-2. 互联网地址

- 互联网上每个接口必须有一个唯一的互联网地址(IP地址)，长32bit，层次结构
- 32bit写成4个十进制数，点分十进制表示法



因特网与TCP/IP安全-3. 协议封装

- 当应用程序用**TCP/IP**传送数据时，数据送入协议栈，通过各层，直到被当作一串比特流送入网络。



因特网与TCP/IP安全-4. IP协议

- **1 IP**协议的作用
- **2 IP**数据包中含有源地址和目的地址
- **3** 不可靠协议，没有做任何事情来确认数据包是否按顺序或是未被破坏
  - 而高层在接受服务时通常假设源地址是有效的
  - **IP**地址成为认证的基础
- **4 IP**协议存在的安全问题不少

因特网与TCP/IP安全-4. IP协议

- **ping of death攻击—Ping的工作原理**
  - 当网络出现问题时，最常用的测试工具就是“**Ping**”命令
  - 假定主机**A**的**IP**地址是**192.168.1.1**，主机**B**的**IP**地址是**192.168.1.2**，都在同一子网内，则当你在主机**A**上运行“**Ping 192.168.1.2**”后，都发生了些什么呢**?**

- ***ping of death攻击—Ping的工作原理***
  - 首先，**Ping**命令会构建一个固定格式的**ICMP**请求数据包
  - 然后由**ICMP**协议将这个数据包连同地址“**192.168.1.2**”一起交给**IP**层协议，**IP**层协议将以地址“**192.168.1.2**”作为目的地址，本机**IP**地址作为源地址，加上一些其他的控制信息，构建一个**IP**数据包
  - 并在一个映射表中查找出**IP**地址**192.168.1.2**所对应的物理地址一并交给数据链路层。
  - 后者构建一个数据帧，目的地址是**IP**层传过来的物理地址，源地址则是本机的物理地址，还要附加上一些控制信息，依据以太网的介质访问规则，将它们传送出去。

- ***ping of death攻击—Ping的工作原理***
  - 主机**B**收到这个数据帧后，先检查它的目的地址，并和本机的物理地址对比，如符合，则接收;否则丢弃。
  - 接收后检查该数据帧，将**IP**数据包从帧中提取出来，交给本机的**IP**层协议。
  - 同样，**IP**层检查后，将有用的信息提取后交给**ICMP**协议，后者处理后，马上构建一个**ICMP**应答包，发送给主机**A**，其过程和主机**A**发送**ICMP**请求包到主机**B**一模一样。

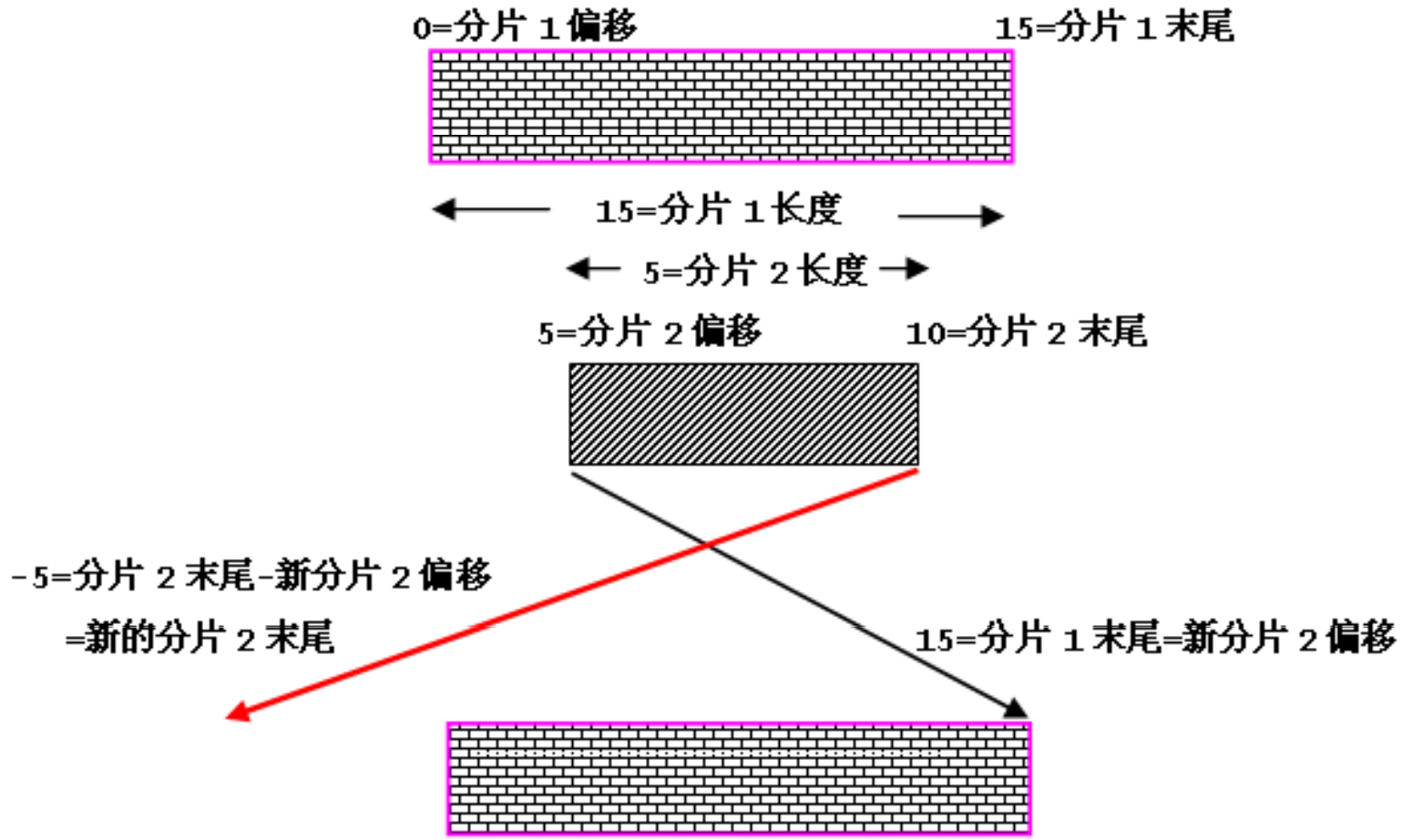
从**Ping**的工作过程，我们可以知道，主机**A**收到了主机**B**的一个应答包，说明两台主机之间的去、回通路均正常。也就是说，无论从主机**A**到主机**B**，还是从主机**B**到主机**A**，都是正常的。

- ***ping of death*攻击—攻击的方式**
  - 由于单个包的长度超过了**IP**协议规范所规定的包的长度
  - 以太网帧长度有限，**IP**必须被分片传输
  - 接受方必须接收所有分片，在进行重组**(**包的重组代码**)**
  - **IP**协议中规定了**IP**的最大尺寸，而大多数包的处理程序又假设超过这个最大尺寸这种情况不存在
  - 包的重组代码所分配的内存区域也不超过这个区域。一旦超大包出现，包中额外数据会被写入其他正常内存区域，使系统进入非稳定状态（缓冲区溢出）
  - **ping of death** 攻击向被攻击主机发送大量的碎片包，使这些碎片组装起来后构成一个超出最大限制的**ping**请求包从而造成被攻击主机的缓冲区溢出

- ***ping of death攻击—解决办法***
  - 可以在防火墙上过滤掉**ICMP**报文，或者在服务器上禁止**Ping**，并且只在必要时才打开**ping**服务。
  - 对操作系统进行补丁，内核将不再对超长包进行重组

因特网与TCP/IP安全-4. IP协议

- 泪滴(Teardrop)攻击
  - IP数据包在网络传递时，数据包可以分成更小的片段。攻击者可以通过发送两段(或者更多)数据包来实现Teardrop攻击。
  - 第一个包的偏移量为0，长度为N，设数据包中第二片IP包的偏移量小于第一片结束的位移，而且算上第二片IP包的Data，也未超过第一片的尾部，这就是重叠现象。


0=分片 1 偏移
15=分片 1 末尾
15=分片 1 长度
5=分片 2 长度
5=分片 2 偏移
10=分片 2 末尾
-5=分片 2 末尾-新分片 2 偏移
=新的分片 2 末尾
15=分片 1 末尾=新分片 2 偏移

- 为了合并这些数据段，**TCP/IP**堆栈会分配超乎寻常的巨大资源，从而造成系统资源的缺乏甚至机器的重新启动。对于**Windows**系统会导致蓝屏死机，并显示**STOP 0x0000000A**错误。
- 及时为操作系统打补丁，但是**Teardrop**攻击仍然会耗费处理器的资源和主机带宽。

因特网与TCP/IP安全-4. IP协议

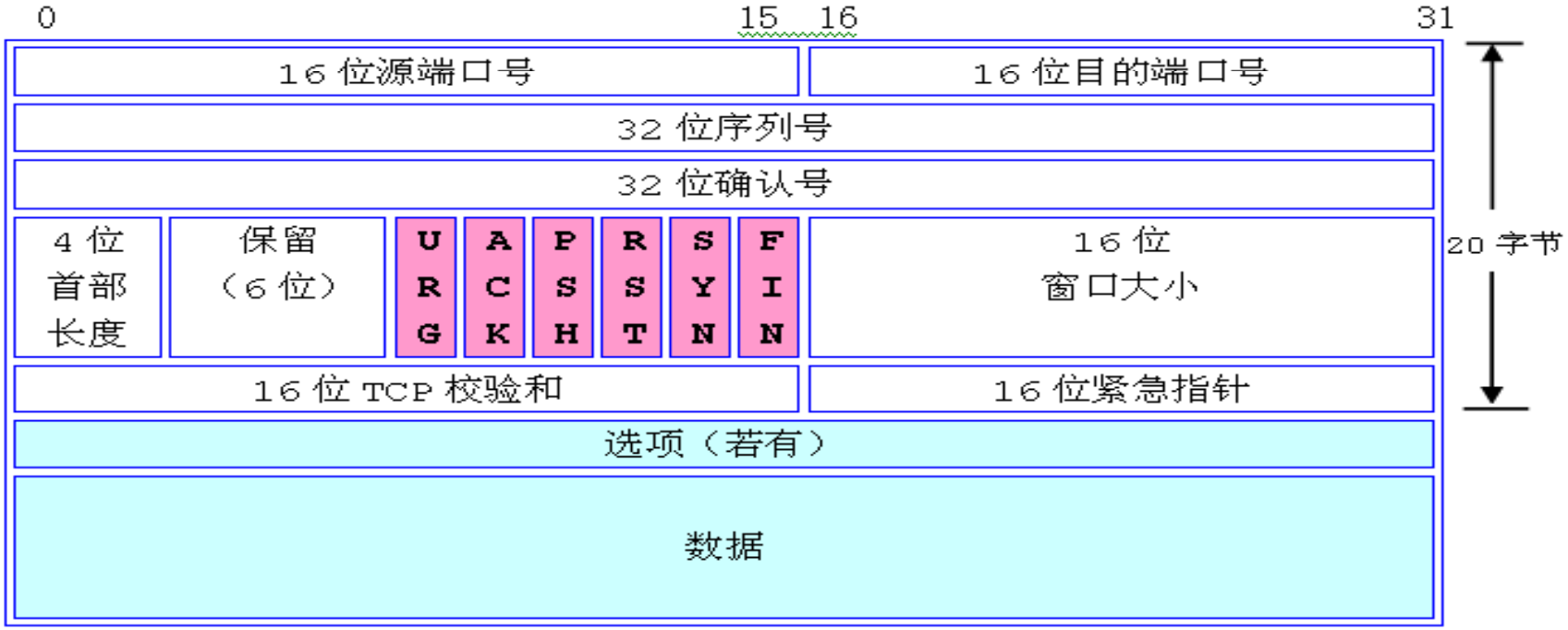
- **Smurf**：该攻击向一个子网的广播地址发一个带有特定请求(如ICMP回应请求)的包，并且将源地址伪装成想要攻击的主机地址。子网上所有主机都回应广播包请求而向被攻击主机发包，使该主机受到攻击。

- ***Pingflood:***该攻击在短时间内向目的主机发送大量**ping**包，造成网络堵塞或主机资源耗尽。

- *源路由*
  - 通过**IP**协议的额外选项－**IP**源路由选项
  - 允许发送者指定包的一条源地址和目的地址的路由。**A-c-d-b**
  - 使用该选项的**IP**包好像是从该路径上的最后一个系统传递过来的。
  - 允许黑客伪装成其他可信任的机器**(d)**
  - 幸运的是，大多数主要服务都不使用源路由选项，我们可以使用防火墙来过滤掉任何的源路由选项数据包。

因特网与TCP/IP安全-5. TCP协议

- 1 作用
- 2 可靠的面向连接的
- 3 格式

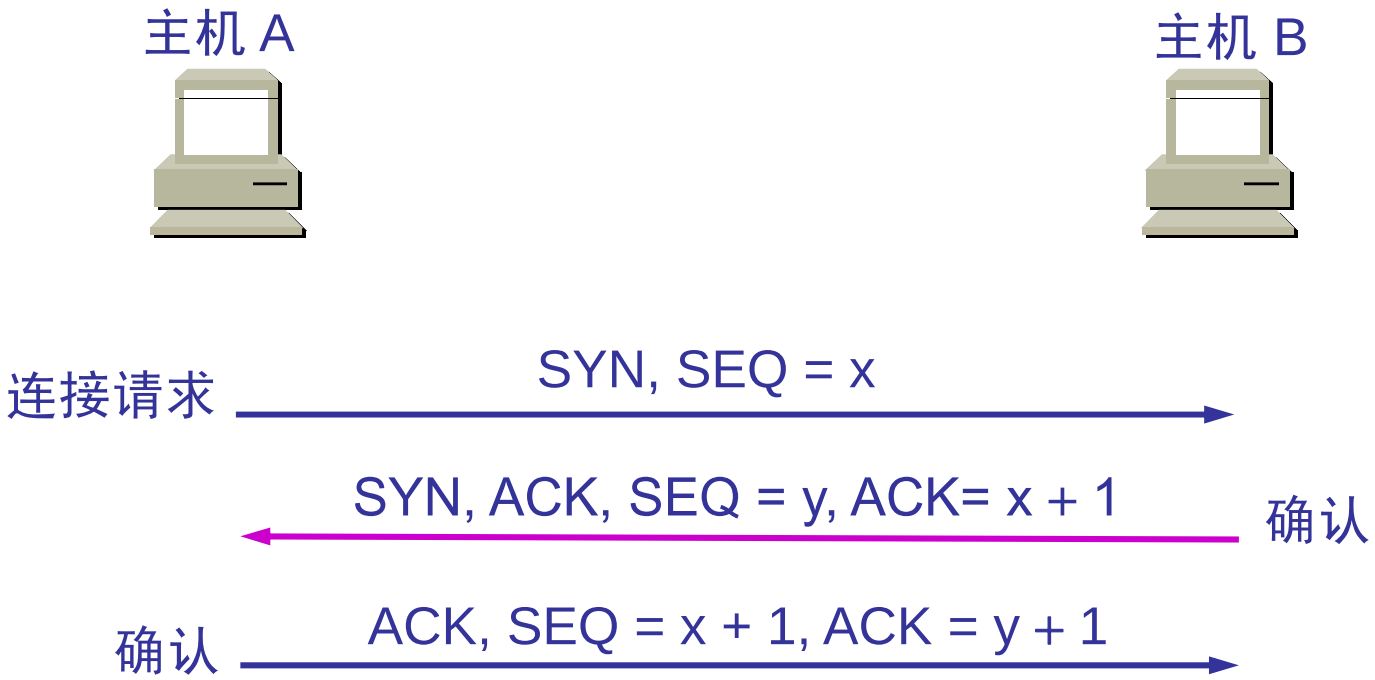


- **4 每个TCP都包含源端和目的端的端口号，与IP**
- **首部的源和目的端的IP地址可以唯一的确定一条TCP连接**
- **5 6个标志比特**

URG	紧急指针有效
ACK	确认序列号有效
PSH	接收方应当尽快将这个报文交给应用层
RST	连接复位
SYN	同步序列号用来发起一个连接
FIN	发送端完成发送任务

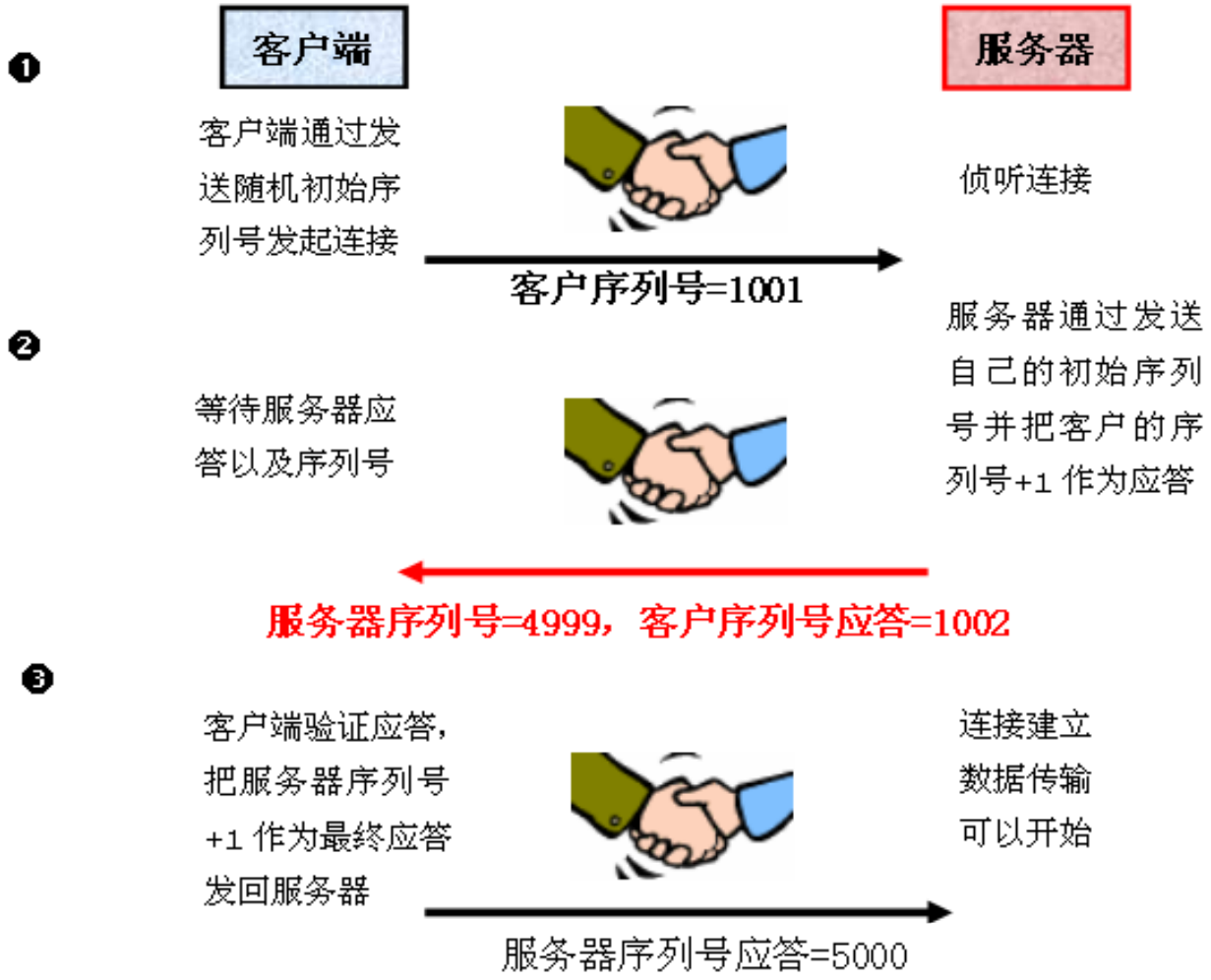
因特网与TCP/IP安全-5. TCP协议

- **TCP安全缺陷—TCP连接的可靠性**
  - 初始化连接：三次握手，确保双方做好传输准备，统一序列号。



因特网与TCP/IP安全-5. TCP协议

- **TCP安全缺陷—TCP连接的可靠性**



- ***TCP安全缺陷—TCP连接的可靠性***
  - 应答每个接受到的数据包：若在规定的时间内应答没有接受到，包被重传。
- ***Land攻击***：攻击者将一个**TCP**包的源地址和目的地址，源端口和目的端口都设置为目标主机的地址，这种包可以造成被攻击主机因试图与自己建立连接而陷入死循环，从而很大程度地降低了系统性能。

攻击者

服务器

❶

攻击者发起LAND攻击，源地址和端口设置为服务器端

侦听连接

攻击包的序列号=1001

服务器通过发送自己的初始序列号并把客户的序列号+1 作为应答

❷

服务器序列号=4999

客户序列号应答=1002

服务器等待客户端发送回服务器的序列号+1作为应答。这儿是 5000 作为应答包的序列号

❸

服务器只看到序列号=1002 应答包。因此重发。

服务器序列号=4999

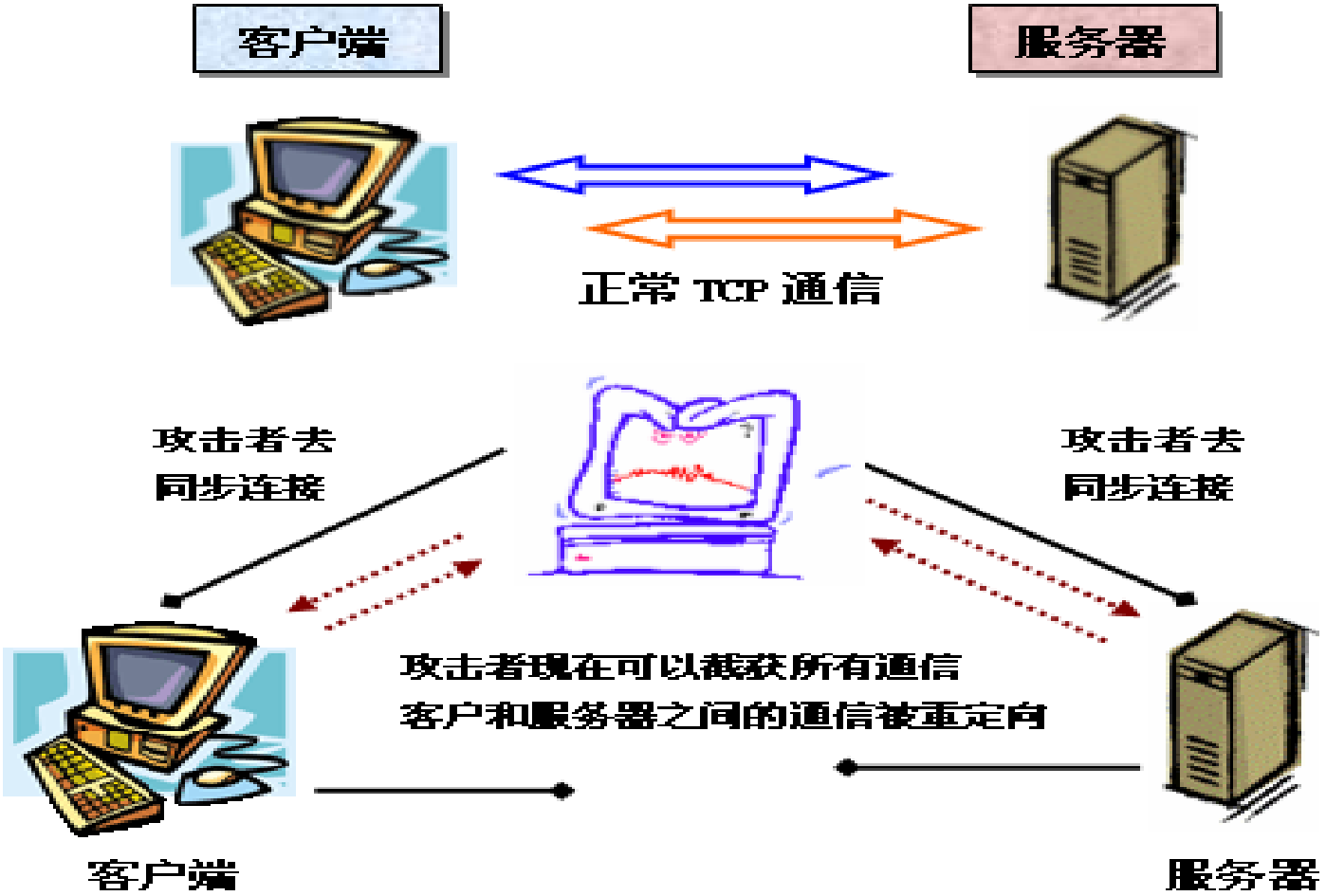
客户序列号应答=1002

服务器等待客户端发送回服务器的序列号+1作为应答。这儿是 5000 作为应答包的序列号

服务器只看到序列号=1002 应答包。因此重发。

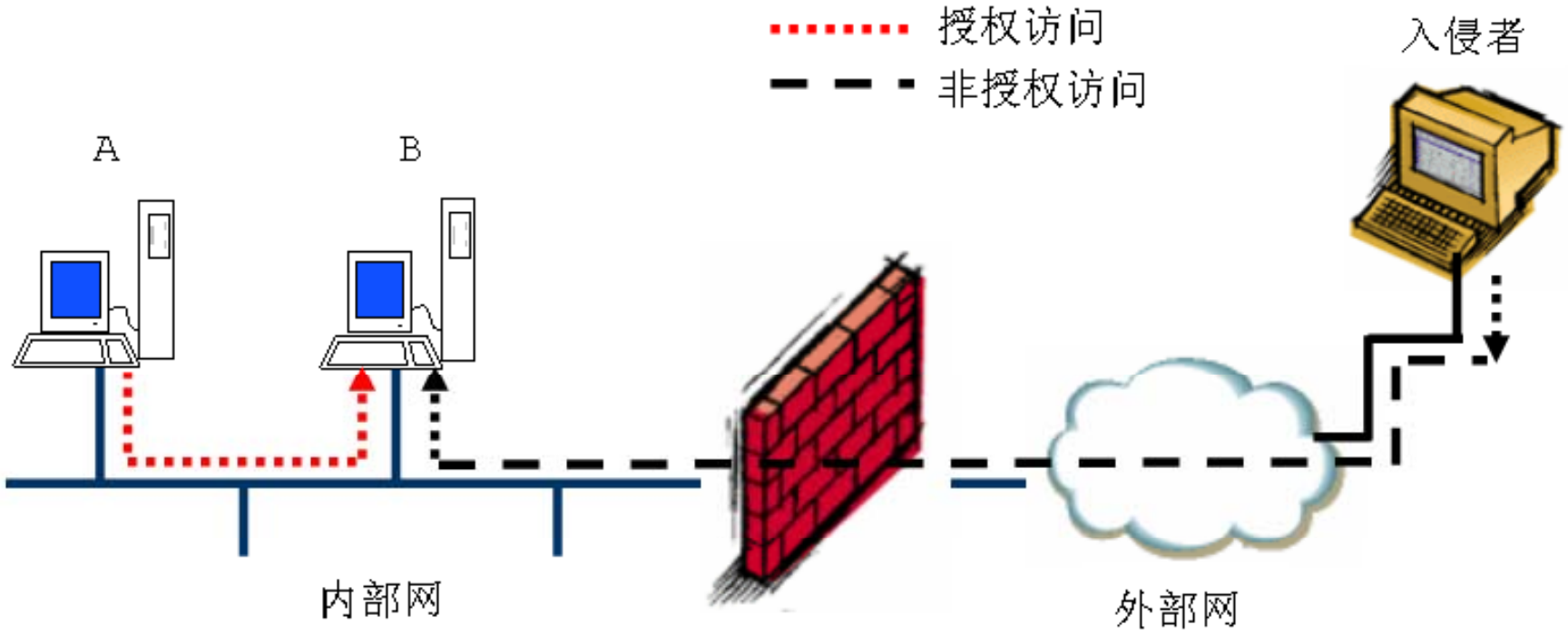
由于TCP是具有高优先权的内核级进程，将中断其它正常的系统操作已获得更多的资源来处理进入的数据。

- ***TCP安全缺陷—没有任何认证机制***
- **TCP**假定只要接受到的数据包含有正确的序列号就认为数据是可以接受的。一旦连接建立，服务器无法确定进入的数据包是否来自真实的机器。



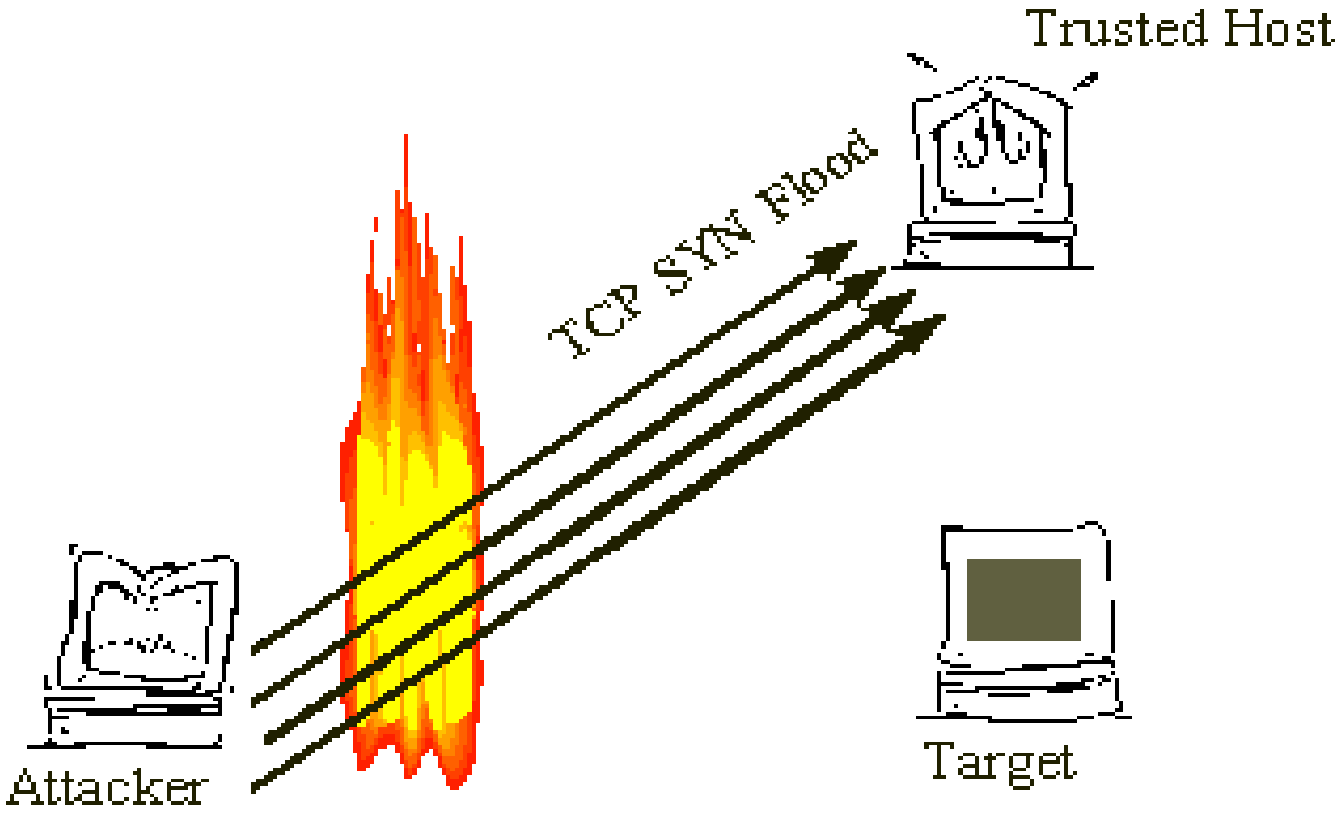
- ***TCP安全缺陷—没有任何认证机制***
- **TCP会话劫持**
- **(1)**截获客户和服务器之间的数据流
- **(2)**可采取被动攻击
- **(3)**攻击者可看到序列号，可伪造数据包放**TCP**流中，这将允许攻击者以被骗客户具有的特权来访问服务器。
- **(4)**攻击者不需要知道口令，只需等待用户登录到服务器，劫持会话数据流就可。

- ***IP欺骗***
- 攻击者通过伪造一个可信任地址的数据包以使一台机器认证另一台机器的复杂技术
- 攻击是利用应用程序之间基于IP地址的认证机制，攻击者通过IP地址欺骗获得远程系统的非法授权访问。



- ***IP欺骗***
- 过程描述：
- **1、X->B: SYN（序列号为M）,IP=A**
- **2、B->A: SYN（序列号为N）,ACK(应答序号＝M+1),此时A没有同B发起连接，当A收到B的包后，会发送RST置位的包给B，从而断开连接，此时攻击者事先让A无法对来自B的任何数据包进行应答。**
- **3、X->B: ACK(应答序号＝N+1) IP=A。攻击者猜测B发送给A的序列号。若猜测成功则，X将获得主机B赋予A的权利。**

- ***IP欺骗***
- 辅助技术1：如何阻止A响应B的包－ **SYN 淹没**



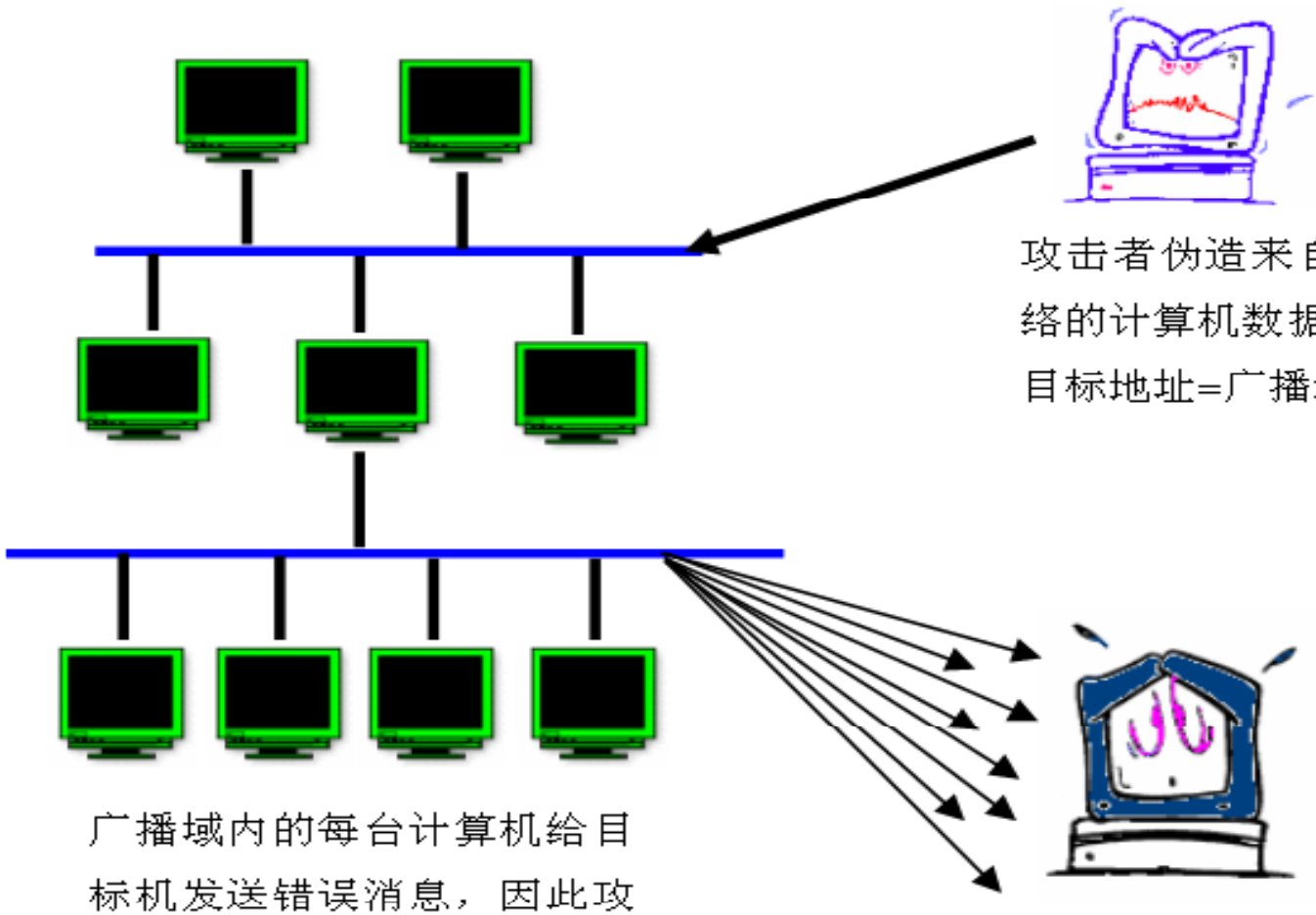
- ***IP欺骗***
- 辅助技术2：序列号预测
- 攻击者事先要进行连接试验：
- (1)同目标主机进行连接
- (2)目标主机应答
- (3)记录应答包所含的序列号，继续步骤1测试
- 分析序列号产生的模式

- *攻击检测*
- 上述攻击过程中，攻击者必须依照一定的顺序来完成网络入侵，这往往是攻击的一种先兆。可安装一个网络嗅探器。
- **(1)**最初，会检测到大量的**TCP SYN**从某个主机发往**A**的登录端口。
- **(2)**大量的**TCP SYN**包将从主机**X**经过网络发往主机**B**，相应的有**SYN+ACK**包从主机**B**发往**X**，然后主机**X**用**RST**作应答

- *其他措施*
- 配置路由器和网关，使他们拒绝网络外部与本网内具有相同**IP**地址的连接请求
- 当包的源**IP**地址不在本地子网内时，路由器和网关不应该把本网主机的包发出去，以阻止内部用户去破坏他人网络
- 在包发送到网络上之前，加密

因特网与TCP/IP安全-6. UDP协议

- **UDP**与**TCP**处于同一层，给应用程序提供不可靠、无连接的分组传输服务。会出现丢失、重复、延迟及乱序。
- **Fraggle**的拒绝服务攻击
  - 单播
  - 广播
  - 多播



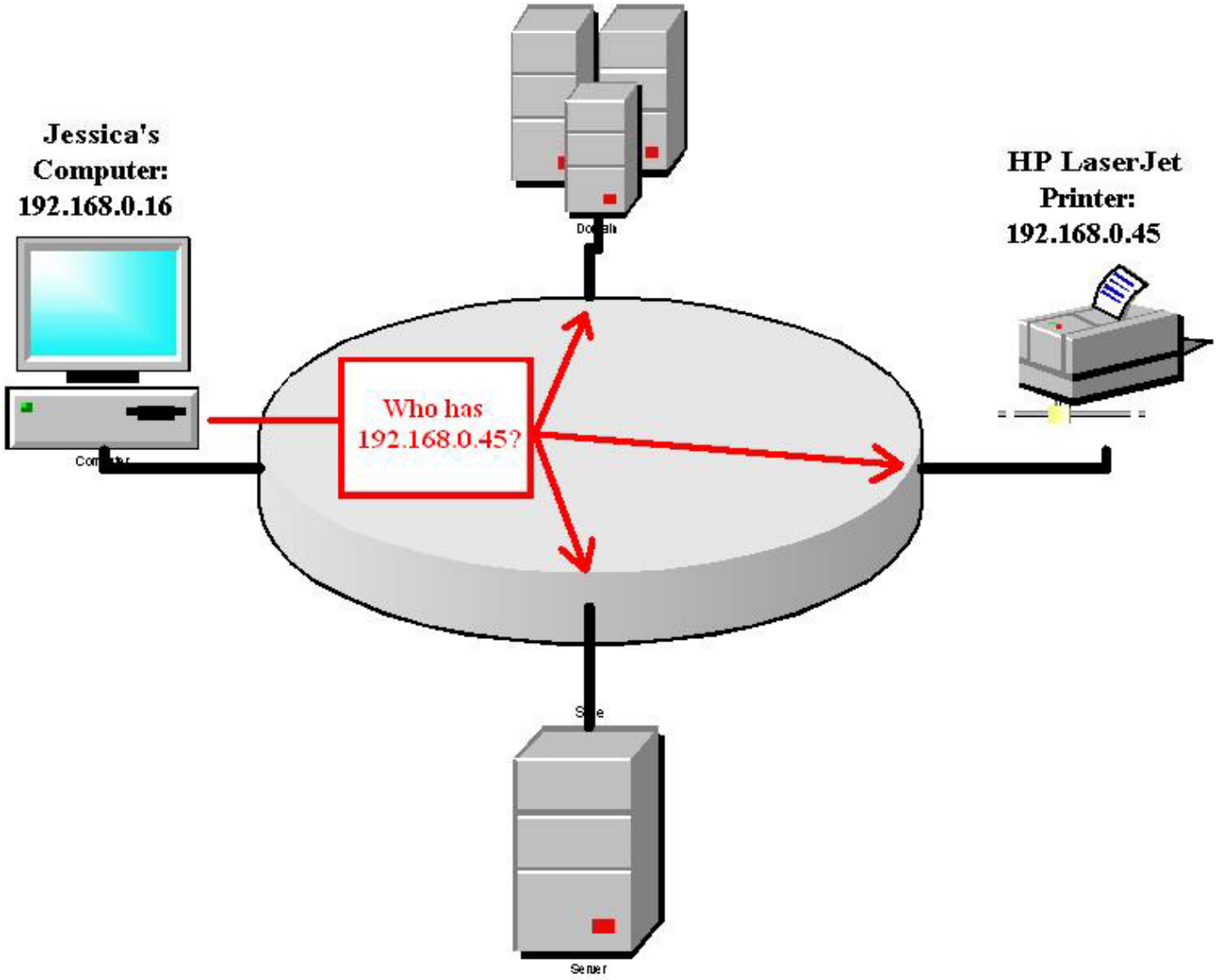
攻击者伪造来自目标网络的计算机数据包，IP目标地址=广播地址

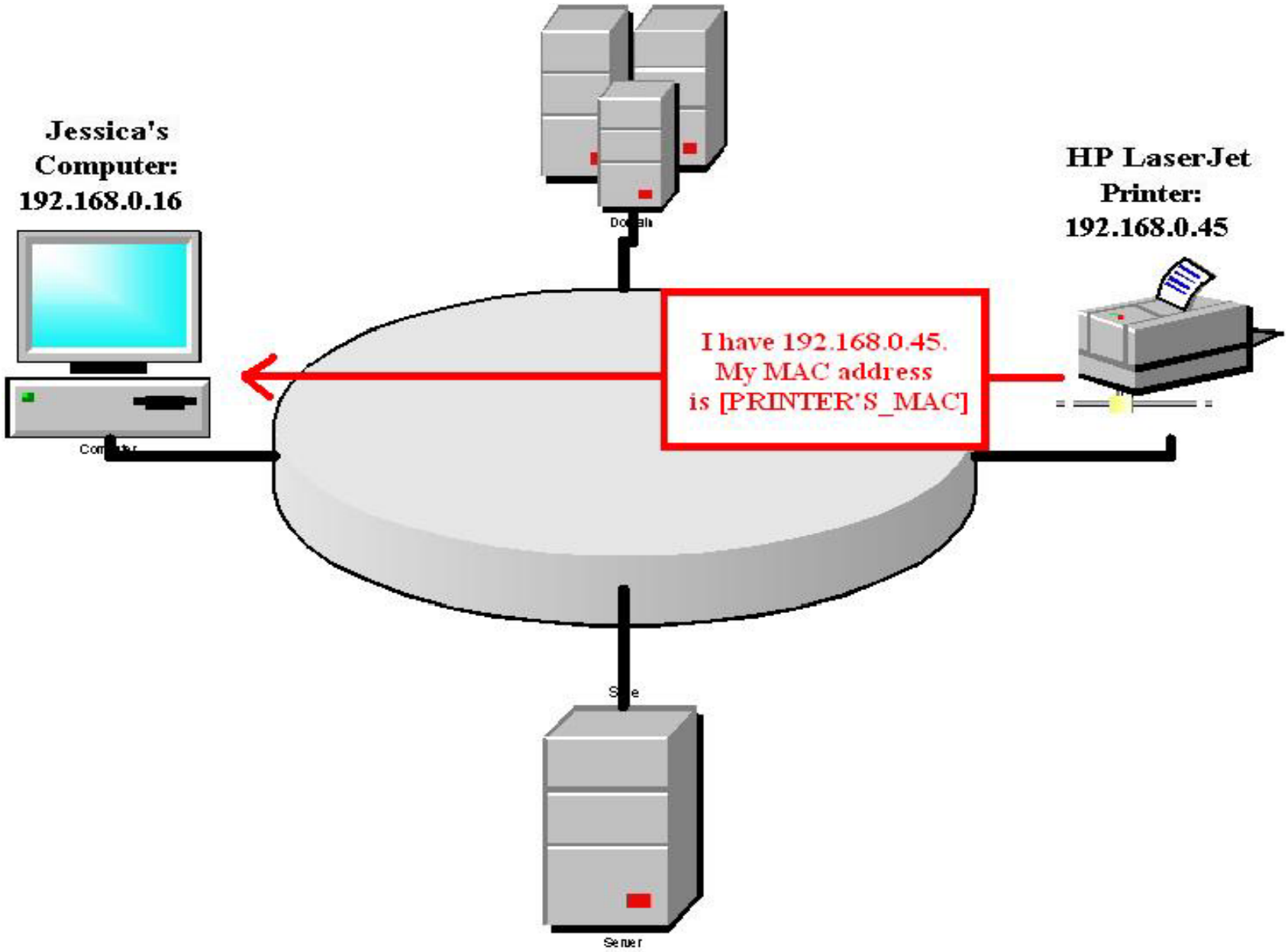
广播域内的每台计算机给目标机发送错误消息，因此攻击效果同广播域内的机器数目成正比，结果形成拒绝服务攻击。

因特网与TCP/IP安全-7. ARP/RARP协议

- **Address Resolution Protocol (ARP)**
- **is how network devices associate MAC addresses with IP Addresses so that devices on the local network can find each other.**

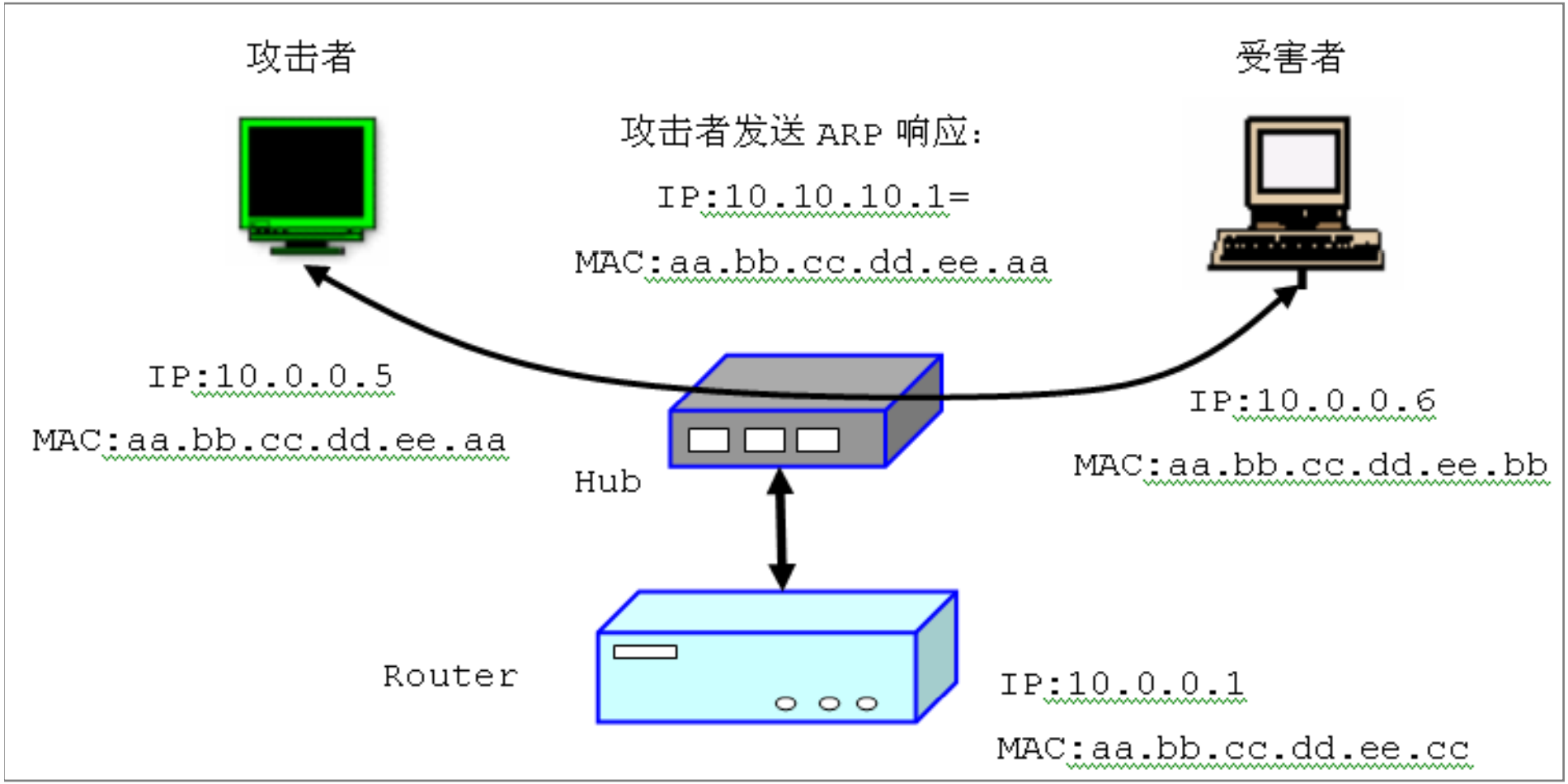
- **ARP, a very simple protocol, consists of merely four basic message types:**
  - An ARP Request**. Computer A asks the network, "Who has this IP address?"**
  - An ARP Reply**. Computer B tells Computer A, "I have that IP. My MAC address is [whatever it is]."**
  - A Reverse ARP Request (RARP)**. Same concept as ARP Request, but Computer A asks, "Who has this MAC address?"**
  - A RARP Reply**. Computer B tells Computer A, "I have that MAC. My IP address is [whatever it is]"**


Jessica's
Computer:
192.168.0.16
HP LaserJet
Printer:
192.168.0.45
Who has
192.168.0.45?
Server


Jessica's
Computer:
192.168.0.16
HP LaserJet
Printer:
192.168.0.45
I have 192.168.0.45.
My MAC address
is [PRINTER'S_MAC]

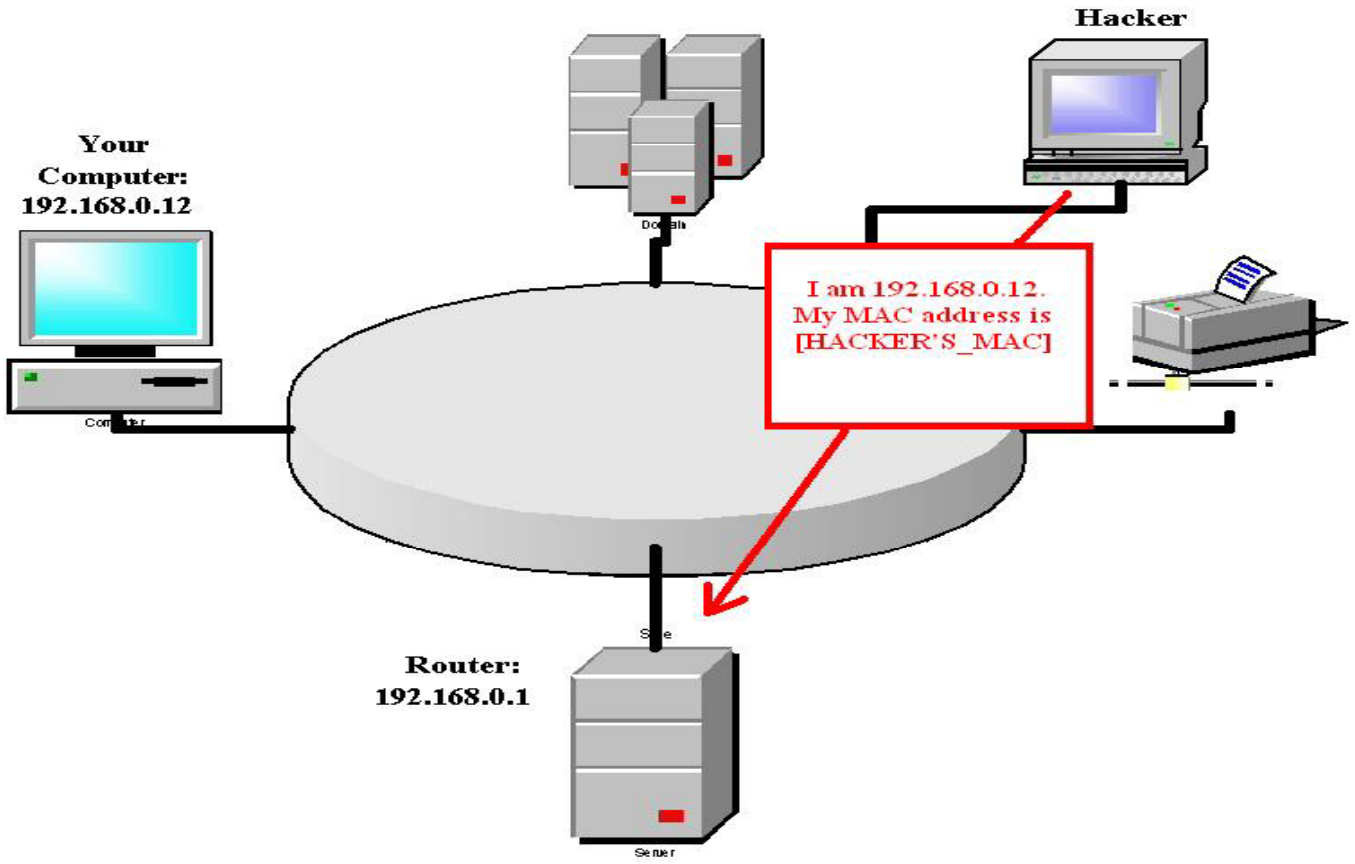
- ***Anatomy of an ARP Poisoning Attack***
- **ARP CACHE POISONING**
- **Man in the Middle**

- ***ARP CACHE POISONING***
- **A hacker can easily associate an operationally significant IP address to a false MAC address.**
- **For instance, a hacker can send an ARP reply associating your network router's IP address with a MAC address that doesn't exist.**


攻击者
受害者
攻击者发送 ARP 响应：
IP:10.10.10.1=
MAC:aa.bb.cc.dd.ee.aa
IP:10.0.0.5
MAC:aa.bb.cc.dd.ee.aa
IP:10.0.0.6
MAC:aa.bb.cc.dd.ee.bb
Hub
Router
IP:10.0.0.1
MAC:aa.bb.cc.dd.ee.cc

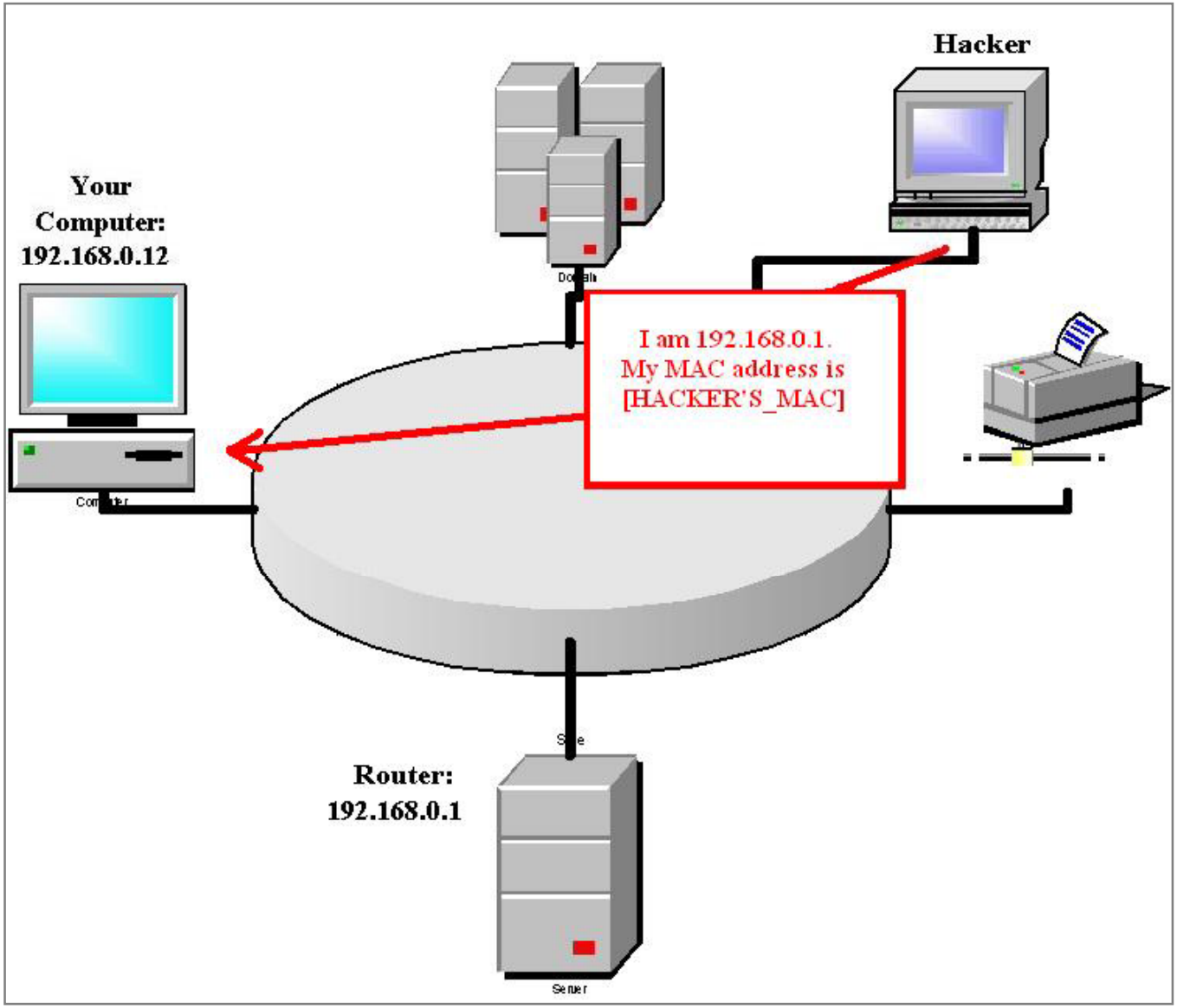
- ***Man in the Middle***
- **A hacker can exploit ARP Cache Poisoning to intercept network traffic between two devices in your network.**

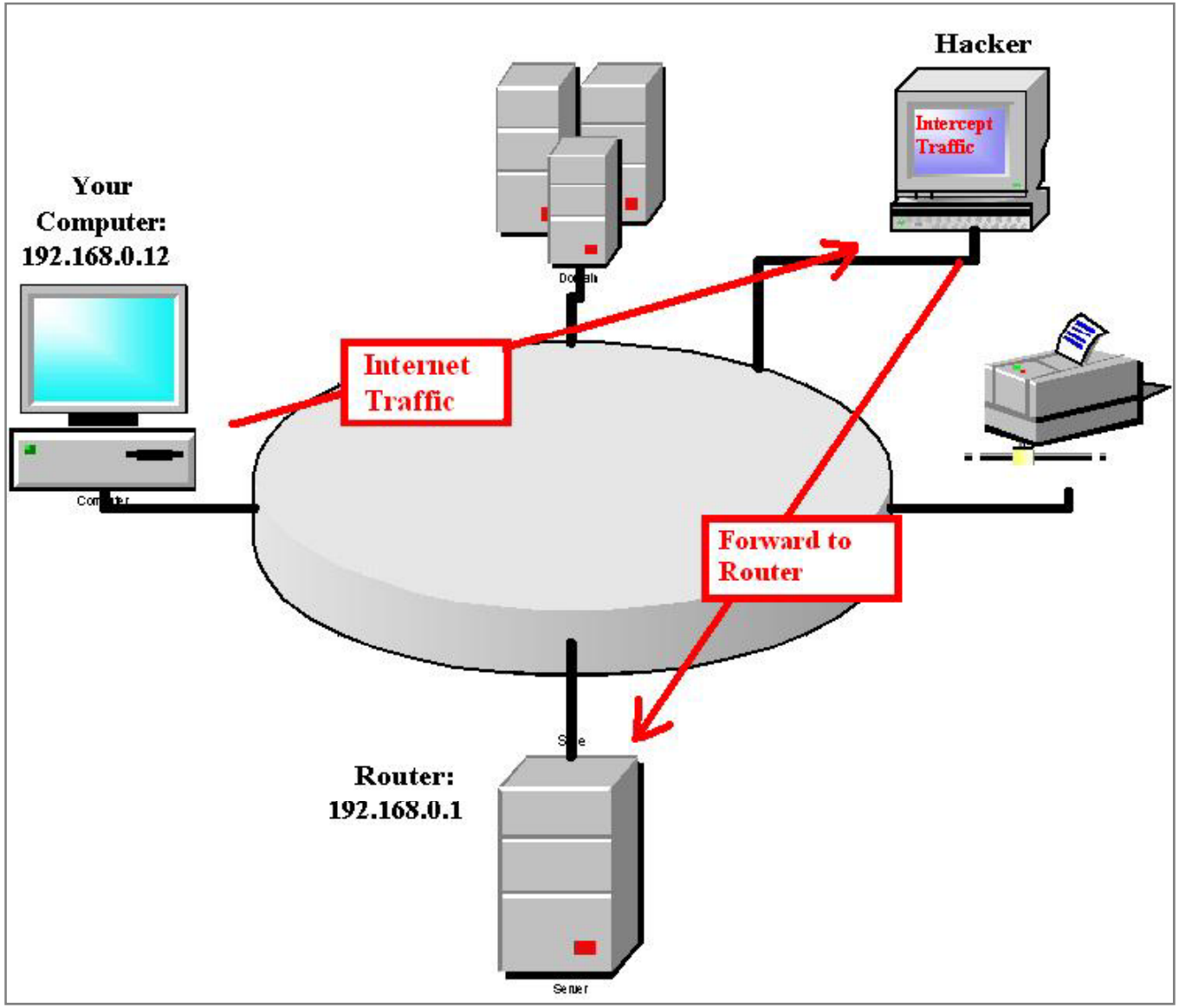
Sending a malicious ARP "reply“ to your Router



Now your router thinks the *hacker's* computer is *your* computer.

Next, a malicious ARP reply to *your* computer




Hacker
Intercept Traffic
Your Computer: 192.168.0.12
Internet Traffic
Forward to Router
Router: 192.168.0.1

分析：

1 ARP攻击只能被用于本地网络的假冒，这意味者攻击者必须已经获得网络中某台机器的访问权(已被IP欺骗)

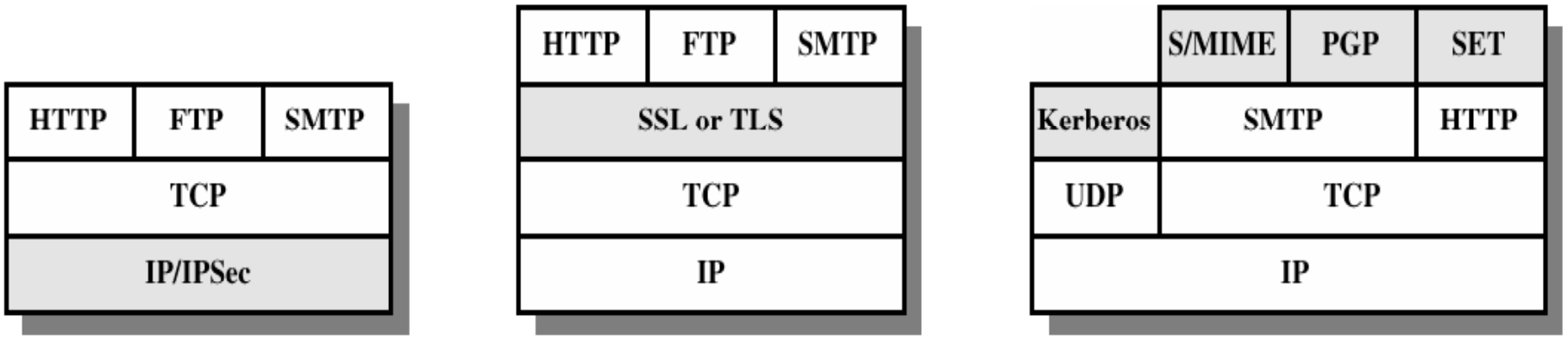
2 ARP请求从来不会送到路由器以外，因此被假冒的机器必须同攻击者所攻击的机器位于同一网段，也就是可通过集线器连接。

解决办法：

1 使用交换机而不是集线器，很多交换机可以配置成硬件地址和特定的端口相连

2 入侵监测系统可以识别LAN中的IP攻击

- 因特网与TCP/IP安全
- SSL协议
- SET协议

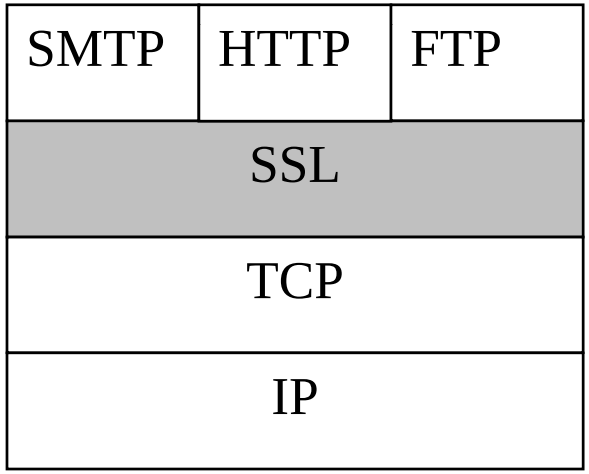


SSL协议—1 概况

- 安全套接字层协议(**Security Socket Layer，SSL**)是网景(**Netscape**)公司于**1994**年提出的基于**web**应用的安全协议。主要介绍SSLv3
- SSL主要采用公开密钥体制和x. 509数字证书技术，其目标是保证两个应用间通信的保密性、完整性和可靠性，可在服务器和客户端两端同时实现支持。

SSL协议—1 概况

- **SSL**介于可靠的传输层协议和应用层协议之间，能为客户端和服务器之间的**TCP／IP**连接提供服务器认证、消息完整性、通信数据加密和可选的客户端认证服务。
- **SSL**可以用于任何面向连接的安全通信，但通常用于安全**web**应用的**HTTP**协议。



SSL协议—1 概况

•SSL提供一个安全的“握手”来初始化一个TCP / IP连接，完成客户端和服务器之间关于安全等级、密码算法．通信密钥的协商，以及执行对连接端身份的认证工作。在此之后，SSL连接上所传送的应用层协议数据都会被加密，从而保证通信的机密性。

•**(1)SSL服务器认证**

•**(2)SSL客户端认证**

•**(3)加密的SSL连接**

SSL协议—2 协议结构

- 协议栈

- 两个状态
  - 连接状态(Connection State)
  - 会话状态(Session State)

SSL协议—2 协议结构

- 通信步骤

(1)建立TCP“连接”；

(2)SSL握手，建立SSL“会话”(Session)；

(3)通过“会话”传送加密数据包；

(4)释放连接，“会话”过期。

- 连接和会话的关系
- 安全连接的特性

- *连接和会话的关系*
  - 连接是传输层中的概念。
  - **SSL**会话联系客户端和服务器，由**SSL**握手协议建立，制定一系列加密的安全参数。这些安全参数可被连接多次使用。
  - 连接是短暂的，对应一个会话，而一个会话可被多次连接使用。
- *安全连接的特性*
  - 连接是私有的。握手协议商量一个会话密钥，然后用对称的加密算法对数据加密
  - 连接的标识符用非对称算法加密
  - 连接是可靠的。

SSL协议—2 协议结构

- SSL规范定义了如下的会话状态要素：
- ①会话标识符：由服务器选择的一个任意的字节序列，用于标识活动的或可恢复的会话状态。
- ②对方证书：对方的X.509v3证书。状态的这个元素可以为空。
- ③压缩方法：在加密之前用来压缩数据的算法。
- ④密码规范：指明大块数据加密算法(例如，空、DES等等)，用于MAC计算的散列算法(例如MD5或SHA-1)，它还定义了加密属性。
- ⑤主密码：48个字节长的Client和Server共享的密码。
- ⑥可重用否：指示会话是否可以用来初始化新的连接的标志。

•SSL规范也定义了如下的连接状态要素：

•①服务器和客户端的随机数：服务器和客户端为每个连接选择的字节序列。

•②服务器写MAC密码：用于对服务器发送数据进行MAC操作的密钥。

•③客户端写MAC密码：用于对客户端发送数据进行MAC操作的密钥。

•④服务器写密钥：用于服务器对数据加密和客户端对数据解密的常规加密密钥。

•⑤客户端写密钥：用于客户端对数据加密和服务器对数据解密的常规加密密钥。

•⑥初始化向量

•⑦序列号：对于每一个连接，每一方都分别维护用于传出的和收到的消息的序列号

SSL协议—3记录层协议

- 功能
- 记录协议层的功能是根据当前会话状态给出压缩算法、对称加密算法、MAC算法、密钥长度、Hash长度、IV长度等参数，以及连接状态中给出的Client和Server的随机数、加密密钥、MAC secrets、IVs、消息序列号对当前的连接中要传送的高层数据实施压缩/解压缩、加/解密、计算/校验MAC等操作。

SSL协议—3记录层协议

- 封装的高层协议

SSL 握手协议	SSL 告警协议	SSL修改 密文协议	应用数据协议 HTTP、FTP、Telnet 等
SSL记录协议			
TCP			
IP			

SSL协议—3记录层协议

- 为SSL连接提供两种服务

•机密性：握手协议定义了共享的可以用于对SSL有效载荷进行常规加密的密钥。

•报文完整性；握手协议还定义了共享的、可以用来形成报文的鉴别码(MAC)的密钥。

SSL协议—3记录层协议

- SSL记录协议的发送过程(接收过程反之)

应用数据

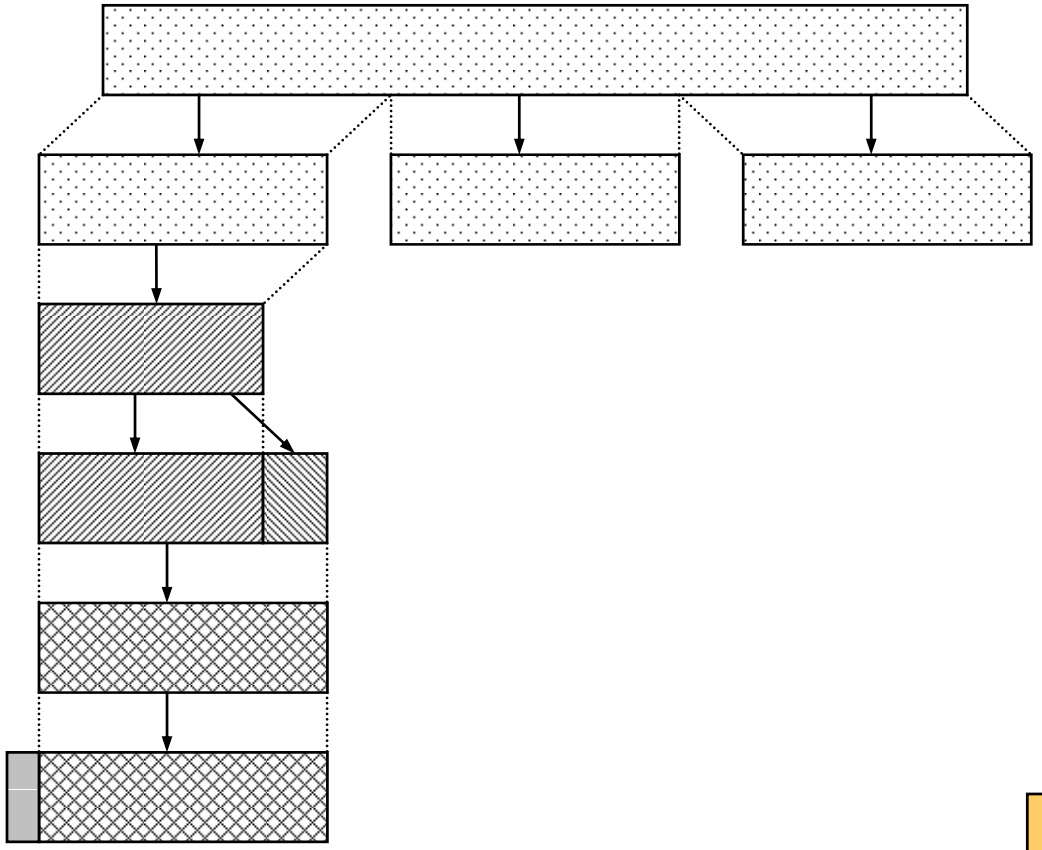
分片

压缩

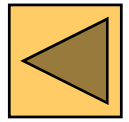
加MAC

加密

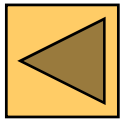
附另SSL记录报头



- 将信息分片
- 将信息分片成不超过2^{14}字节(16384字节)的明文记录(Planitext Records)。其数据结构为：
- struct {
  - ContentType type; //用来处理分片的高层协议
  - ProtocolVersion version; //采用的版本，这里是SSL V3
  - uint16 length; //SSL分片的长度，不能超过2^{14} bytes
  - opaque fragment[SSLPlaintext.length]; //应用数据
- } SSLPlaintext; //分片数据格式



- 用当前Ciper Spec中指定的MAC算法生成MAC。为此目的，需使用先前建立的共享密钥。
- 生成MAC的公式如下：
- hash(MAC_write_secret||pad_2||
  hash(MAC_write_secret||pad_1||seq_num||
  SSLCompressed.type||
  SSLCompressed.length||
  SSLCompressed.fragment))



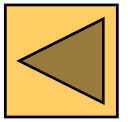
•在加密数据上附加一个首部，该首部由下面一些字段组成：

• 内容类型（8bit）：表明用来处理这个包装的数据片的更高层协议。

• 主要版本（8bit）：指示使用SSL的主要版本。对于SSLV3字段值为3。

• 次要版本（8bit）：指示使用的次要版本。对于SSLV3字段值为0。

压缩长度（**16bit**）：明文数据片以字节为单位的长度（如果使用压缩就是压缩数据片）。最大的值是$2^{14}+2048$。

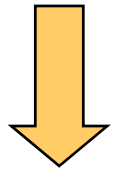


SSL协议—4 修改密文规约协议

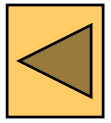
- 它是最简单的。它存在的目的是为了表示密码策略的变化。
- 在完成握手协议之前，客户端和服务器端都要发送这一消息，以便通知对方其后的记录将用刚刚协商的密码规范及相关的密钥来保护。
- 所有意外的更改密码规范消息都将产生一个“意外消息”（**unexpected message**）警告

SSL协议—5 告警协议

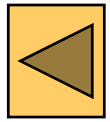
- 作用
- 包括若干个告警消息，告警消息的作用是当握手过程或数据加密等操作出错误或发生异常情况时，向对方发出警告或中止当前连接。
- 组成
- 第一个字节的值为warning(警告)或fatal(致命)，表示消息的严重性等级。第二个字节是具体的告警消息。



- 第一个字节为fatal时，也就是说该消息为致命的告警消息，致命的告警消息包括：
- ①unexpected_message(意外消息)：如果接收了不合适的报文，就返回该消息。
- ②bad_record_mac(错误的记录MAC)：如果收到了不正确的MAC，就返回该消息。
- ③decompression_failure(解压缩失败)：如果解压缩函数收到不适当的输入(例如数据不能解压缩或者解压缩后的数据超出了最大允许的长度)，就返回该消息。
- ④handshake_failure(握手失败)：如果在给定的选项可用时，发送者不能协商可接受的安全参数集合，则返回该消息。
- ⑤illegal_parameter(非法参数)：如果握手报文中的某数据域超出范围或者与其它的数据域不兼容，则返回该消息。



- 如果第一个字节为warning，则表明该消息是警告的告警消息，警告的告警消息包括：
- ①close_notify(关闭通知)：通知接收方发送者在这个连接上将不会再发送任何报文。在关闭一个连接的写方之前，每个交互实体都要需发送该消息。
- ②no_certificate(无证书)：如果没有合适的证书可用，则可以发送该消息作为对证书请求的响应。
- ③bad_certificate(错误证书)：返回该消息表明收到的证书是错误的(例如包含了一个无法通过验证的签名)。
- ④unsupported_certificate(不支持的证书)：返回该消息表明收到的证书类型不被支持。
- ⑤certificate_revoked(证书已吊销)：返回该消息表明证书已被它的签发者废弃。
- ⑥certificate_expired(证书已过期)：返回该消息表明证书已经过期。
- ⑦certificate_unknown(未知证书)：返回该消息表明在处理证书时，出现了其他一些没有说明的情况，使得它不能被接受。



SSL协议—5 握手协议

- 作用
- 验证实体身份
- 协商密钥交换算法、压缩算法和加密算法、
- 完成密钥交换
- 生成密钥
- 客户端和服务器要建立一个连接，就必须进行握手过程。每次握手都存在一个会话和一个连接，连接一定是新的，但会话可能是新的，也可能是已存在的会话。

SSL协议—5 握手协议

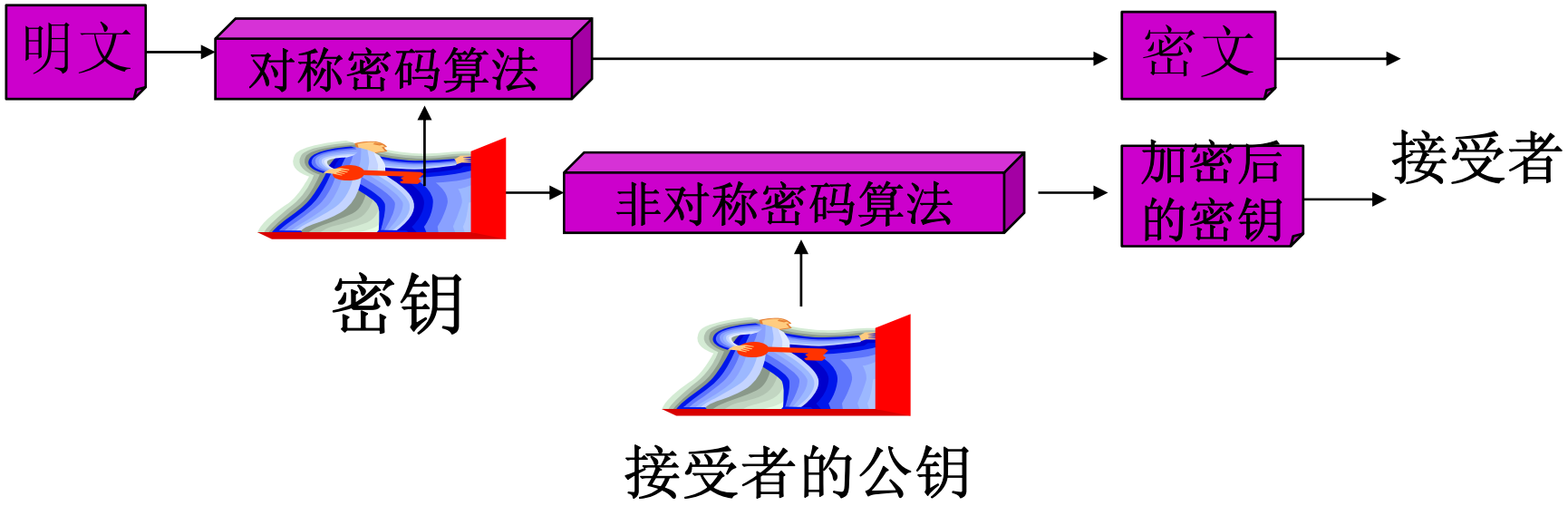
- *消息描述*
- 握手协议由一系列在客户端和服务器端之间交换的消息组成
- ⑴ **Hello Request(问候请求)**
- ⑵ **Client Hello(客户端问候)消息**
- ⑶ **Server Hello(服务器问候)消息**
- ⑷ **Server Certificate(服务器证明)消息**
- ⑸ **Server Key Exchange(服务器密钥交换)消息**
- ⑹ **Certificate Request(证明请求)消息**
- ⑺ **Server Hello Done(服务器问候结束)消息**

- ⑻ **Client Certificate(客户端证明)消息**
- ⑼ **Client Key Exchange(客户端密钥交换)消息**
- ⑽ **Certificate Verify(证明检查)消息**
- ⑾ **Finished(结束)消息**

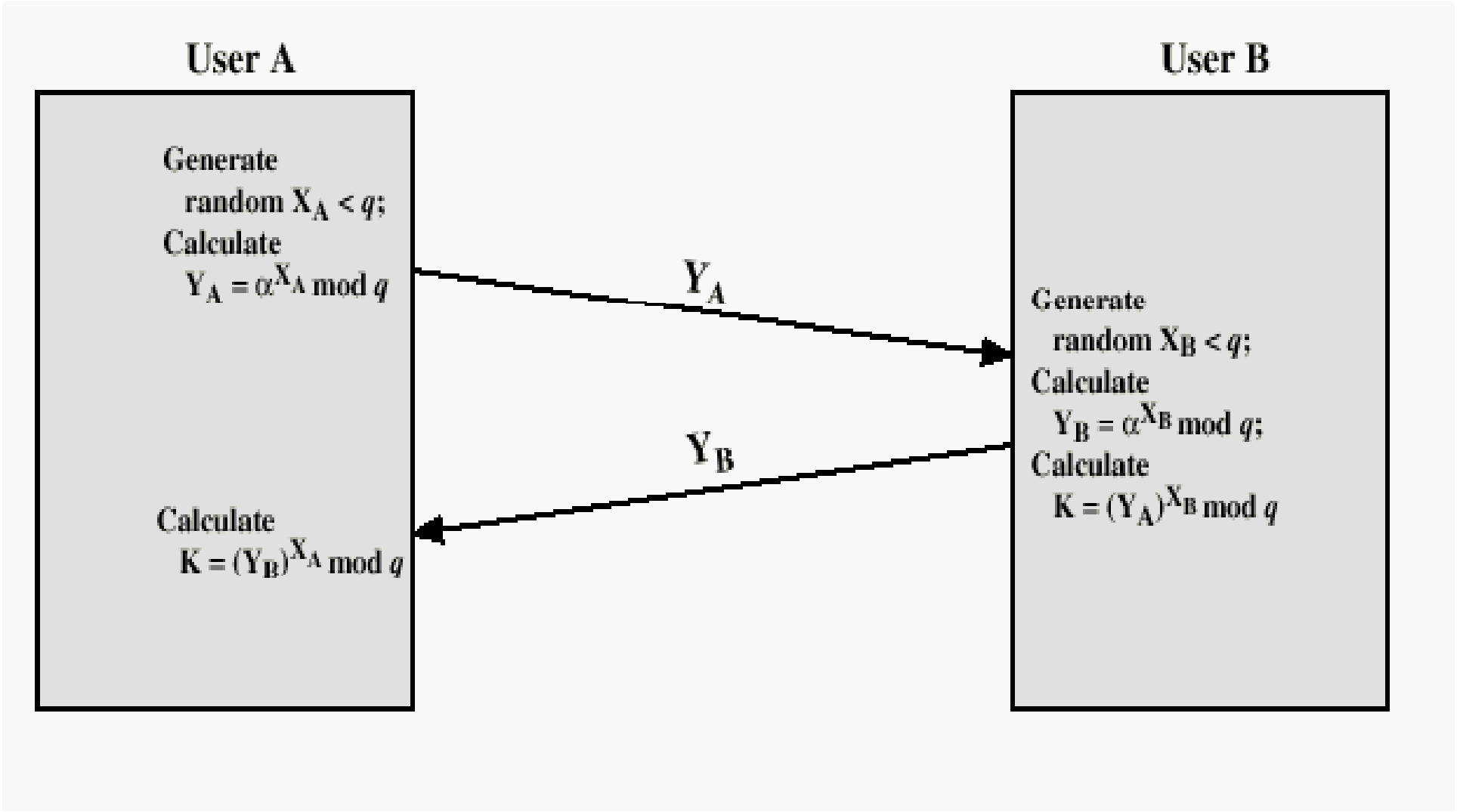
- ⑴ **Hello Request(问候请求)**
- 信息发送者：服务器
- 接受者的反应：
  - 如果客户端正在进行握手协议，则该信息被忽略；
  - 如果客户端有空，则发送Client Hello信息。
- 在随后的握手协商结束后，服务器才能再次发送Hello Request信息。

- (2)Client Hello(客户端问候)消息
- 发送者：客户端
- 内容：
  - 客户端版本号(Client_version)：
  - 随机数（Random）：用于SSL协议中后面的密码学计算。它由一个4字节的时间戳和安全随机数生成器生成的28字节的随机数组成。
  - 会话ID（Session id）：可变长度的会话标识符。
  - 密码组（Cipher_suite）：一个客户端支持的密码算法组合的列表。密码组参数的第一个元素是密钥交换方法；之后的是CipherSpec。
  - 压缩算法（Compression_methods）：与密码组域相似，该域列出客户端已知的所有压缩算法。同样，该列表也依照客户端的偏爱来排序。尽管该域通常在SSLV3中不使用，但以后的TLS将要求支持它。

- 密钥交换
  - 在非安全的通信通道中双方交换会话密钥，以加密通信双方后续连接所传输的信息。每次逻辑连接使用一把新的会话密钥，用完就丢弃。
- 密钥交换的方法
  - RSA：使用接收者的RSA公开密钥对密钥进行加密。接收者密钥的公开密钥证书必须提供。



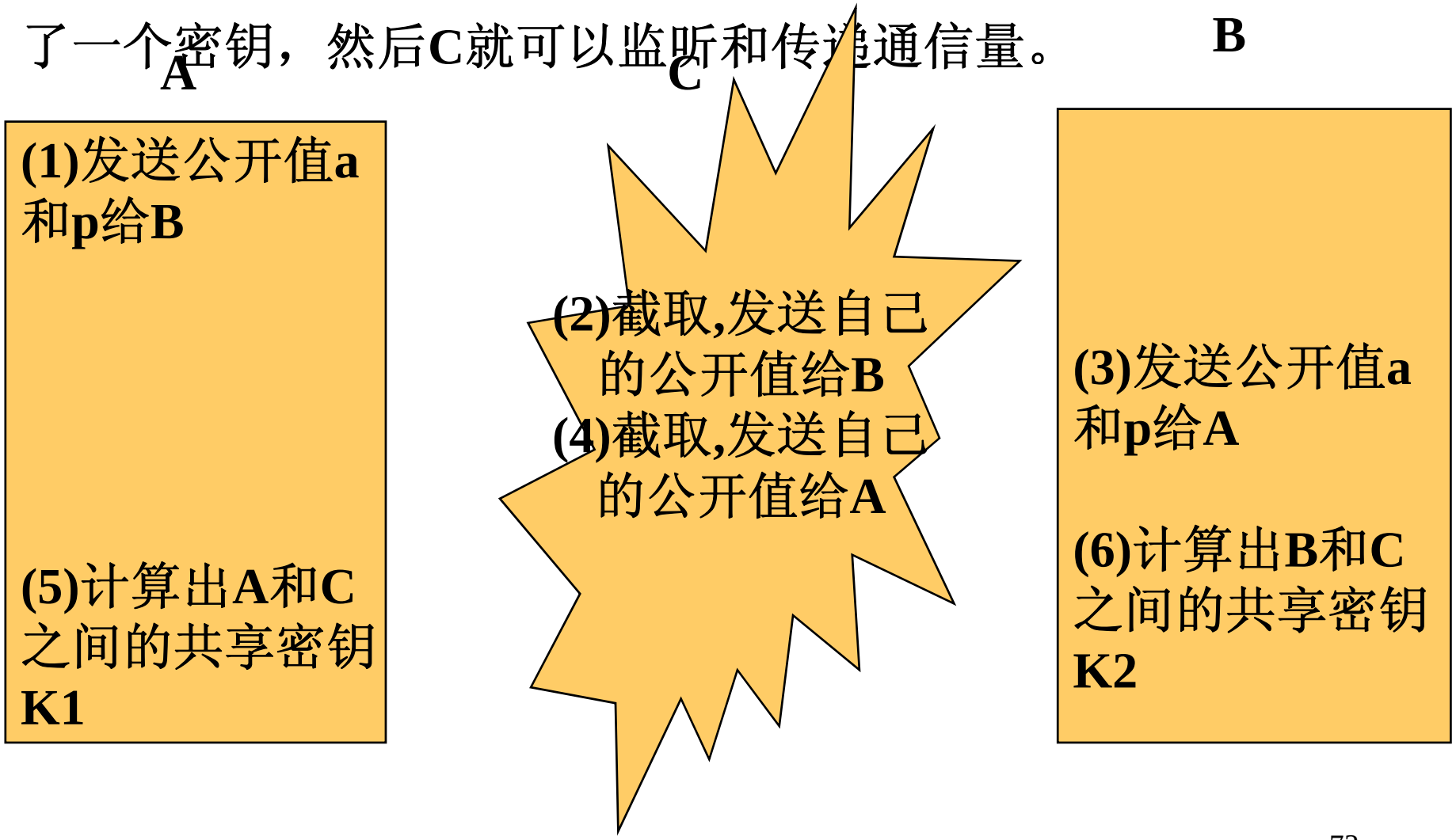
- 密钥交换的方法
  - Diffie-Hellman:有效性依赖于计算离散对数的难度。a和q公开，Y_A和Y_B分别是A和B的公钥



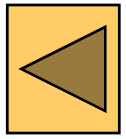
容易遭受中间人的攻击。

第三方C在和A通信时扮演B；和B通信时扮演A。A和B都与C协商

了一个密钥，然后C就可以监听和传递通信量。



- 固定的Diffie-Hellman：其中服务器的公开密钥证书包含了Diffie-Hellman公开密钥参数。客户也可在证书中提供它的Diffie-Hellman公开密钥参数(如果要求客户鉴别)。
- 短暂的Diffie-Hellman：产生临时的Diffie-Hellman公开密钥。用发送者的私有RSA或DSS密钥签名的Diffie-Hellman公开密钥。接收者可以使用对应的公开密钥来验证签名，使用证书来鉴别签名所对应的公开密钥。这种方法看起来是三种Diffie-Hellman选项中最安全的，因为它导致了临时性的、经过鉴别的密钥。



•CipherSpec包括下面一些字段：

•①加密算法：以前提到的任何算法：RC4、RC2、DES、3DES、DES40，IDEA，Fortezza。

•②MAC算法：MD5或SHA-1。

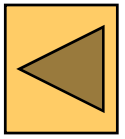
•③加密类型：流或分组。

•④可输出的：真或假。

•⑤散列大小：0,16字节(对于MD5)，或者20字节(对于SHA-1)。

•⑥密钥素材：包含了用来生成写密钥数据的字节序列。

•⑦IV大小：用于密文块链(CBC)加密的初始值的大小。



(3) Server Hello(服务器问候)消息

- 服务器处理了 Client Hello消息之后，可以用一个握手失败警告或者一个 Server Hello消息来响应。
- 信息的结构类似Client Hello，区别在于：Client Hello消息用于列出客户端的能力，而Server Hello消息则用于作出决定，并将该决定传输回客户端。

- 一个Server Hello消息包含下面的域：
- 服务器版本号(Server version)
- 随机数（Random）
- 会话ID（Session id）：该域提供了与当前连接相对应的会话的标识信息。如果从客户端接收到的会话标识符非空，则服务器将查找其缓存以便找一个匹配。如果找到了一个匹配，服务器就可以重用指定的会话状态来建立一个新的连接。在此情况下，服务器返回由客户端提供的同样的值，用以标识一个重用的会话。否则，该域中将包含一个不同的值，标识一个新的会话。
- 密码组（Cipher suite）：该域标识由服务器从客户端提供的列表中选择出来的一个单独的密码组。
- 压缩方法（Compression method）：与密码组消息相似，该域标识由服务器从客户端提供的列表种选择出来的一个单独的压缩方法。

- (4) Server Certificate(服务器证明)消息
- 如果要求验证服务器，则服务器立刻在Server Hello信息后发送其证明Certificate。通常为通常是一个X.509V3服务器证书。该消息也可以使客户端得到服务器的公钥。客户端将用该公钥加密实际的会话密钥。

- ⑸Server Key Exchange(服务器密钥交换)消息
- 当服务器没有**Certificate**或有一个仅用于签名的**Certificate(**如**DSS**，**RSA)**，或采用**Fortezza**密钥交换时，服务器才会发送一个**Server Key Exchange**消息。该消息补充以前在**Server Hello**消息声明中的密码组，它为客户端提供了继续通信而需要的算法变量。这些值依赖于所选定的算法。例如，对于**RSA**密钥交换（此时**RSA**仅用于签名），消息中可能包含一个临时的**RSA**公钥指数和模，以及对这些数据的一个签名。

- (6) Certificate Request(证明请求)消息
- 为了认证的目的，可以用一个可选的Certificate Request消息来向客户端请求一个证书。它由两个参数组成：第一个参数表明可以接受的证书类型；第二个参数表明可以接受的CA的可接受的特定名称（DN）。如果服务器是匿名的、则在请求客户端Certificate时会导致致命错。

- (7) Server Hello Done(服务器问候结束)消息
- 服务器发出该信息表明**Server Hello**结束，然后等待客户端响应。客户端收到该信息后检查服务器提供的**Certificate**是否有效，以及服务器的**Hello**参数是否可接受。

- (8) Client Certificate(客户端证明)消息
- 该信息是客户端收到服务器的**Server Hello Done**后可以发送的第一条信息。只有当服务器请求**Certificate**时才需发此信息。如果客户端没有合适的**Certificate**，则发送“没有证明”的告警信息，如果服务器要求有“客户端验证”，则收到告警后宣布握手失败。

(9) Client Key Exchange(客户端密钥交换)消息

- Client Key Exchange消息允许客户端向服务器发送密钥信息。本密钥信息是关于双方在会话中使用的对称密钥算法的。该消息的内容取决于密钥交换的类型，如下所示：

⑩● RSA：客户端生成一个48字节的预主密码（Pre—master secret），它要么使用服务器证书中的公钥加密，要么使用一个 Server Key Exchange中的临时RSA密钥加密。然后将结果发送给服务器用来计算主秘密密钥。

⑩●匿名Diffie-Hellman：报文内容由两个全局Diffie-Hellman值组成(一个素数和这个数的一个初始根)，加上服务器的公开Diffie-Hellman密钥。

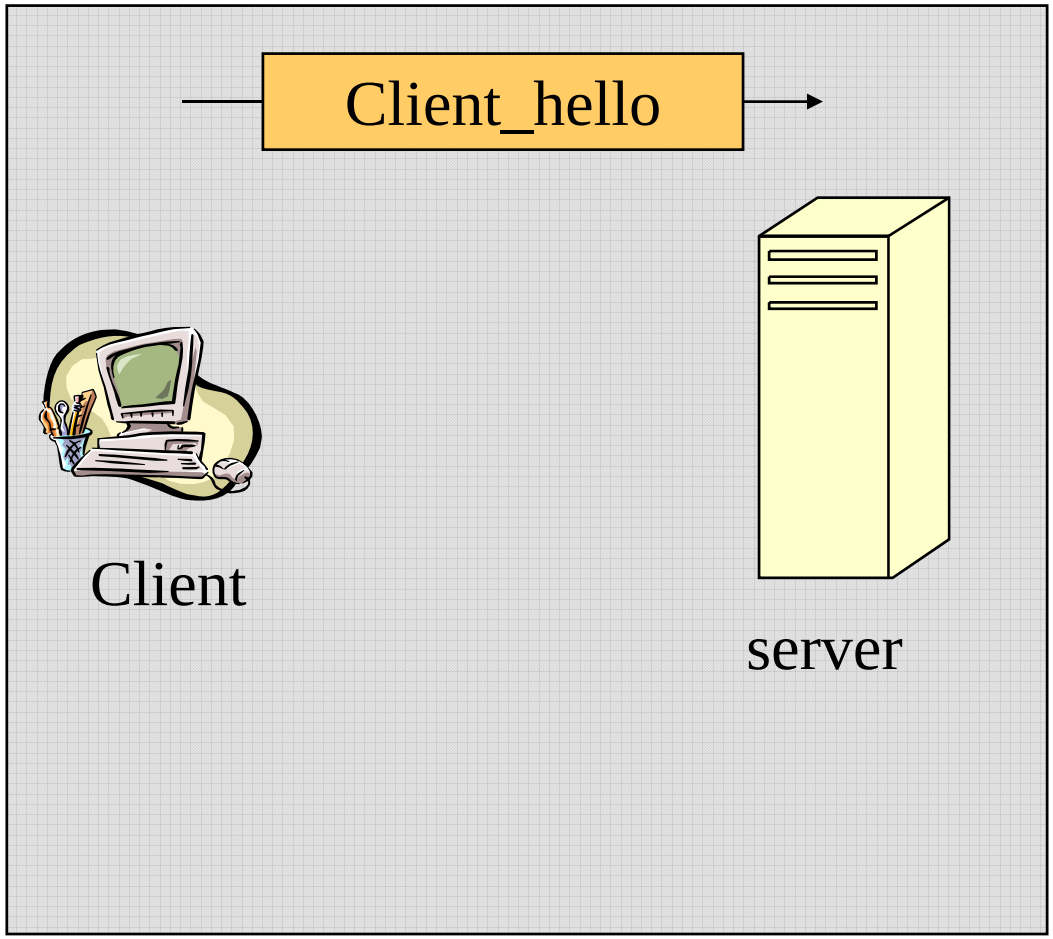
⑩●短暂Diffie-Hellman：报文内容包括了用于匿名Diffie-Hellman的三个Diffie-Hellman参数，加上这些参数的签名。

- ⑽ Certificate Verify(证明检查)消息
- Certificate Verify消息用于提供对客户端证书的显式认证。当使用客户端认证时，服务器使用其私钥来对客户端认证。该消息中包含有经客户端私钥签名的预主秘密密钥。服务器不需要向客户端认证它自己。因为预主秘密是使用服务器的公钥发送给服务器的，只有合法的服务器使用相应的私钥才可以解密它。

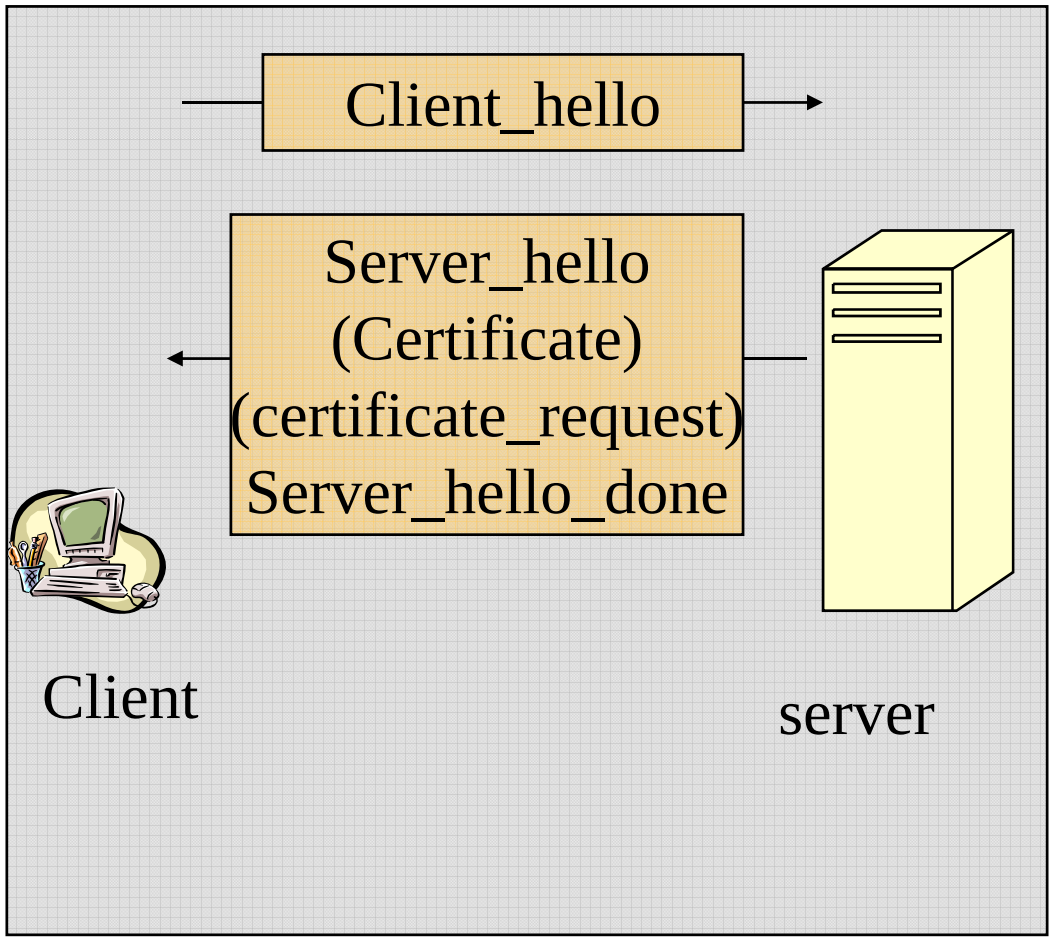
- ⑾ Finished(结束)消息
- 该信息在Change Cipher Spec之后发送，以证明密钥交换和验证的过程已顺利进行。当服务器接收到该Finished消息时，它同样发送一个Change Cipher Spec消息，然后发送它的Finished消息。此时握手协议完成，双方可以安全地传输应用数据了。
- Finished消息是第一个用刚刚协商的算法、密钥和秘密保护的消息。因此，通信的双方就可以验证密钥交换和认证过程是否成功。不需要对Finished消息进行确认；紧接在发送了Finished消息之后，双方可以开始发送加密的数据。Finished消息的接收者必须验证其内容的正确性。

SSL协议—5 握手协议

- 建立一个新的握手过程



包括：
客户端支持的**ssl**版本号
产生的随机数
会话**ID**
密码算法列表
压缩方法列表



Server_hello包括：

从客户端列表中选出的**ssl**版本号

产生的随机数

会话**ID**

从客户端列表中选出密码算法

选定的压缩算法

Certificate:

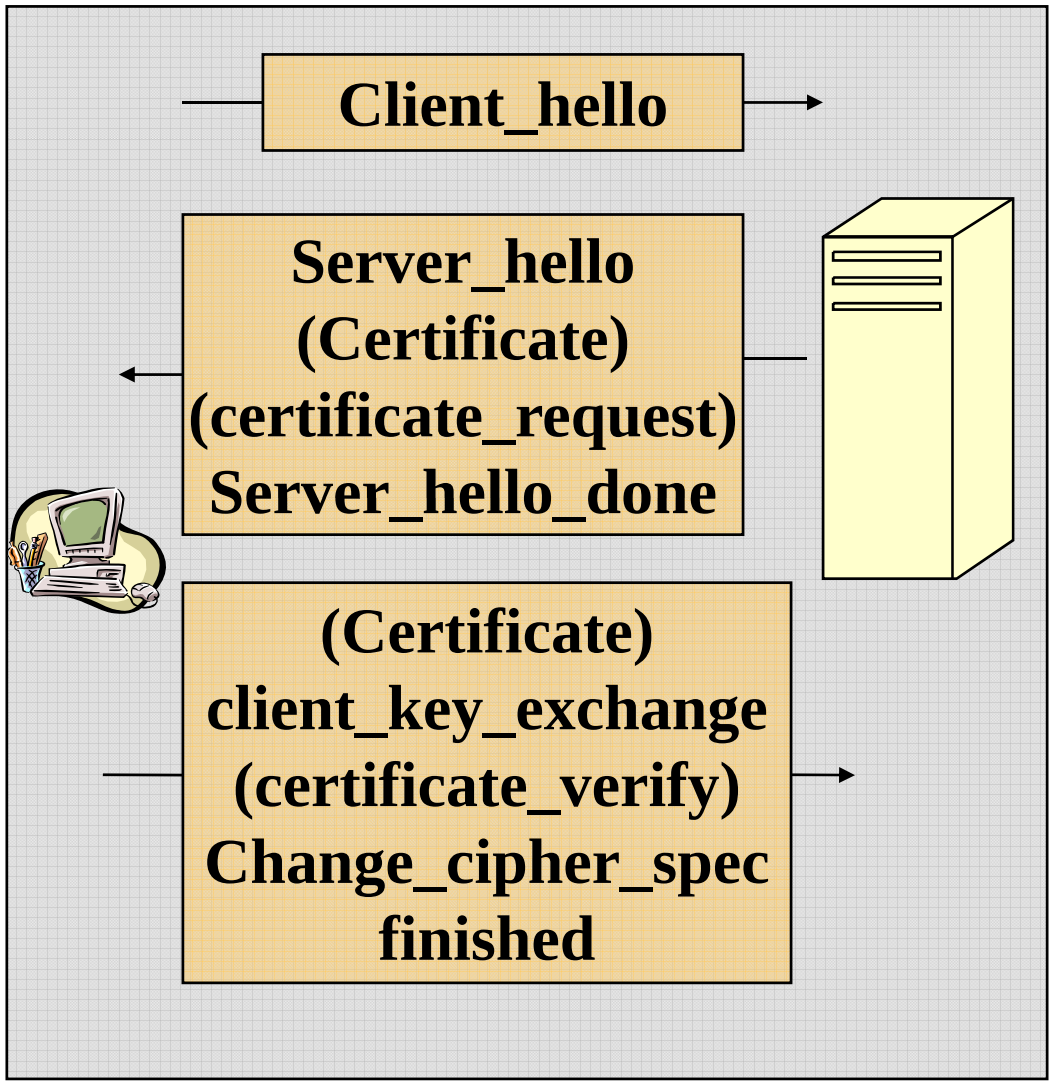
由认证中心签发的**X.509**证书。包括服务器的信息和公钥，客户端可以使用这个公钥向服务器发送加密信息

certificate_request：

一般客户端是匿名的

Server_hello_done

完成通信，等待客户端应答



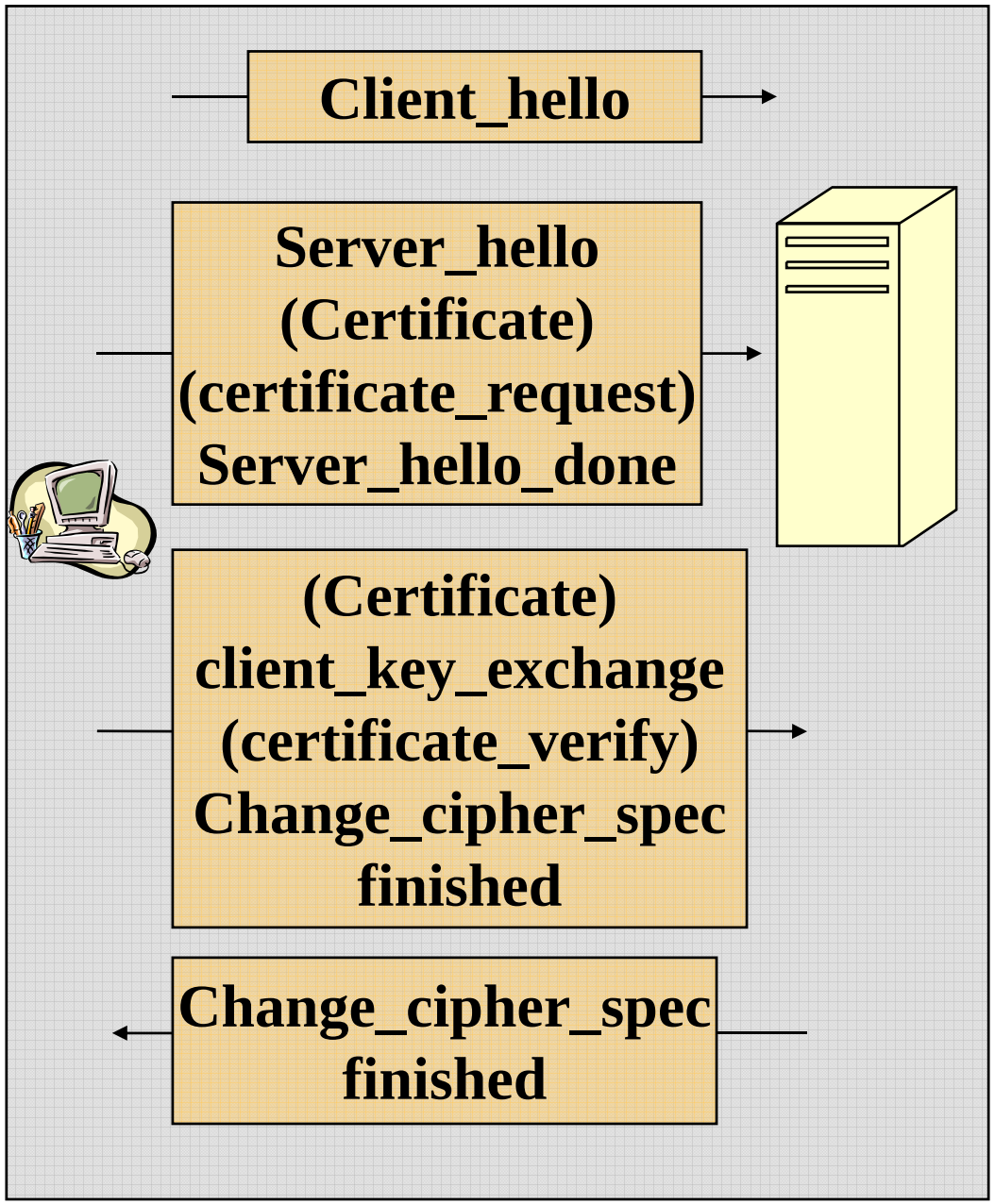
Certificate:
客户的**X.509**证书。

client_key_exchange：
随机数，用于会话密钥的建立

certificate_verify：
若用户提供了证书，将用私钥对信息签名

Change_cipher_spec
表示将使用选定的密码算法和参数处理将来的通信。但是客户端不需要将密钥告诉服务器，因为服务器可使用前面的随机数和算法计算出同样的密钥。

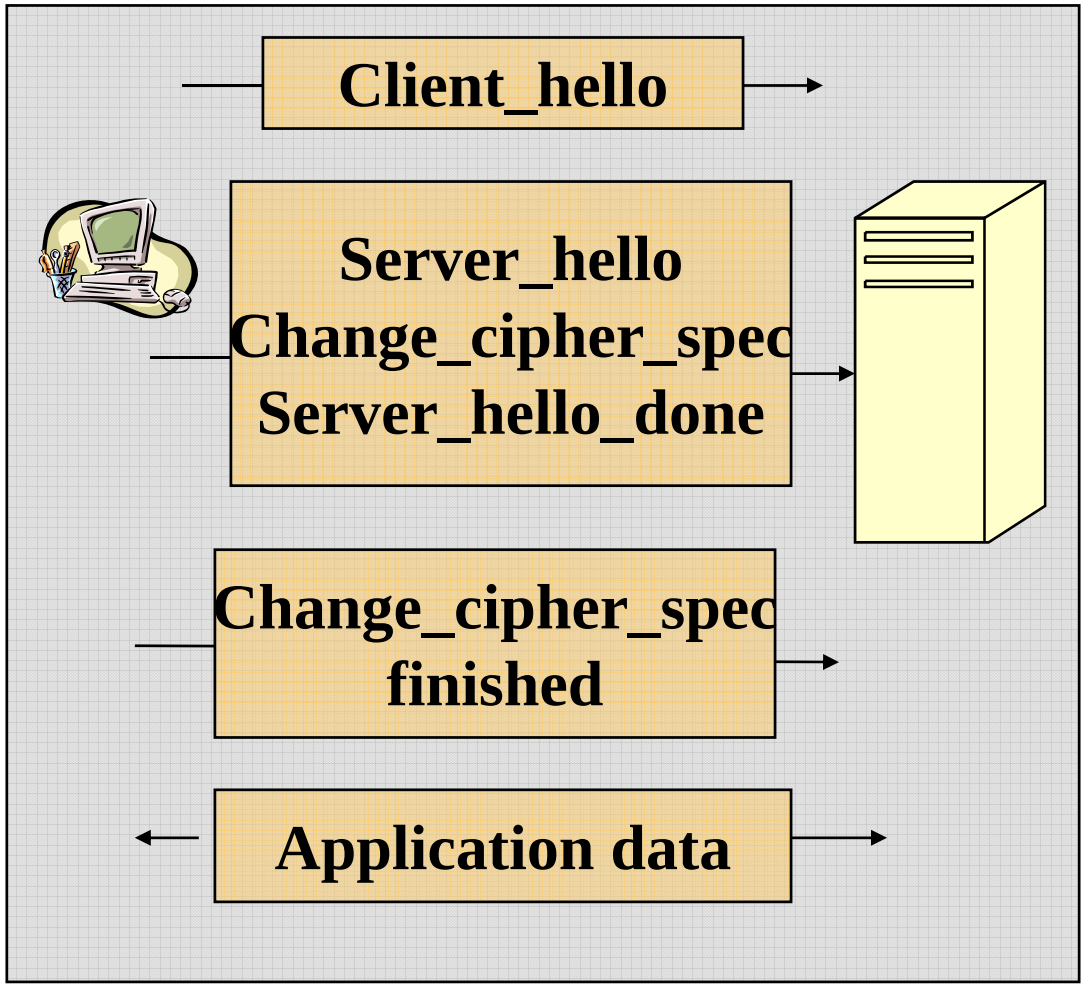
***Finished*：**
使用会话密钥加密



Change_cipher_spec：
向客户端显示它也将使用与客户端相同的参数来加密将来所有的通信

Finished：
使用会话密钥加密

- 恢复一个已存会话的握手过程



Client_hello:
会话ID是要恢复的会话的session_id
服务器发送一个含有相同session_id的Server Hello消息_id

当通过恢复一个会话建立一个连接时，这一新的连接继承这个会话状态下的压缩算法、Cipher Spec和master secret。但这个连接所产生的新的ClientHello.random和ServerHello.random，二者和当前会话的master secret用来生成此连接使用的密钥、MAC secrets。

SSL协议—5 握手协议

要注意的问题：

由于公钥加密算法的速度非常慢。为了提高性能，双方可以缓存并重用在握手协议过程中交换的信息。该过程称为会话ID重用。

如果在握手协议的过程中可以确定协议的客户端和服务器所共享的一个会话ID的话，就可以略过公钥和认证操作，同时可以在密钥生成的过程中重用先前生成的共享秘密。

SSL协议—6 密码计算

- 主密码：从预主密码衍生出来的 P50
- 预主密码：在**RSA**密码技术预主密码是由客户端生成，然后通过**Client Key Exchange**消息发送给服务器的。在**Diffie-Hellman**密码中，预主密码是由每一方（服务器和客户端）使用对方的**Diffie-Hellman**公开值来生成的。p80
- **MAC**写密钥和写密钥：对于一个连接，从主密码中生成的。用于每次连接的加密 p51

- **master_secret=MD5(pre_master_secret || SHA('A' || pre_master_sectert||ClientHello.random || ServerHello.random)) ||MD5(pre_master_secret || SHA('BB' || pre_master_sectert ||ClientHello.random || ServerHello.random)) ||MD5(pre_master_secret || SHA('CCC' || pre_master_sectert ||ClientHello.random || ServerHello.random))**
- 其中，**ClientHello.random**和**ServerHello.random**是在初始**hello**报文中交换的两个现时值。

- 因特网与TCP/IP安全
- SSL协议
- SET协议
- IPSec协议

Secure Electronic Transactions

- An open encryption and security specification.
- Protect credit card transaction on the Internet.
- Companies involved:
  - MasterCard, Visa, IBM, Microsoft, Netscape, RSA, Terisa and Verisign
- Not a payment system.
- Set of security protocols and formats.

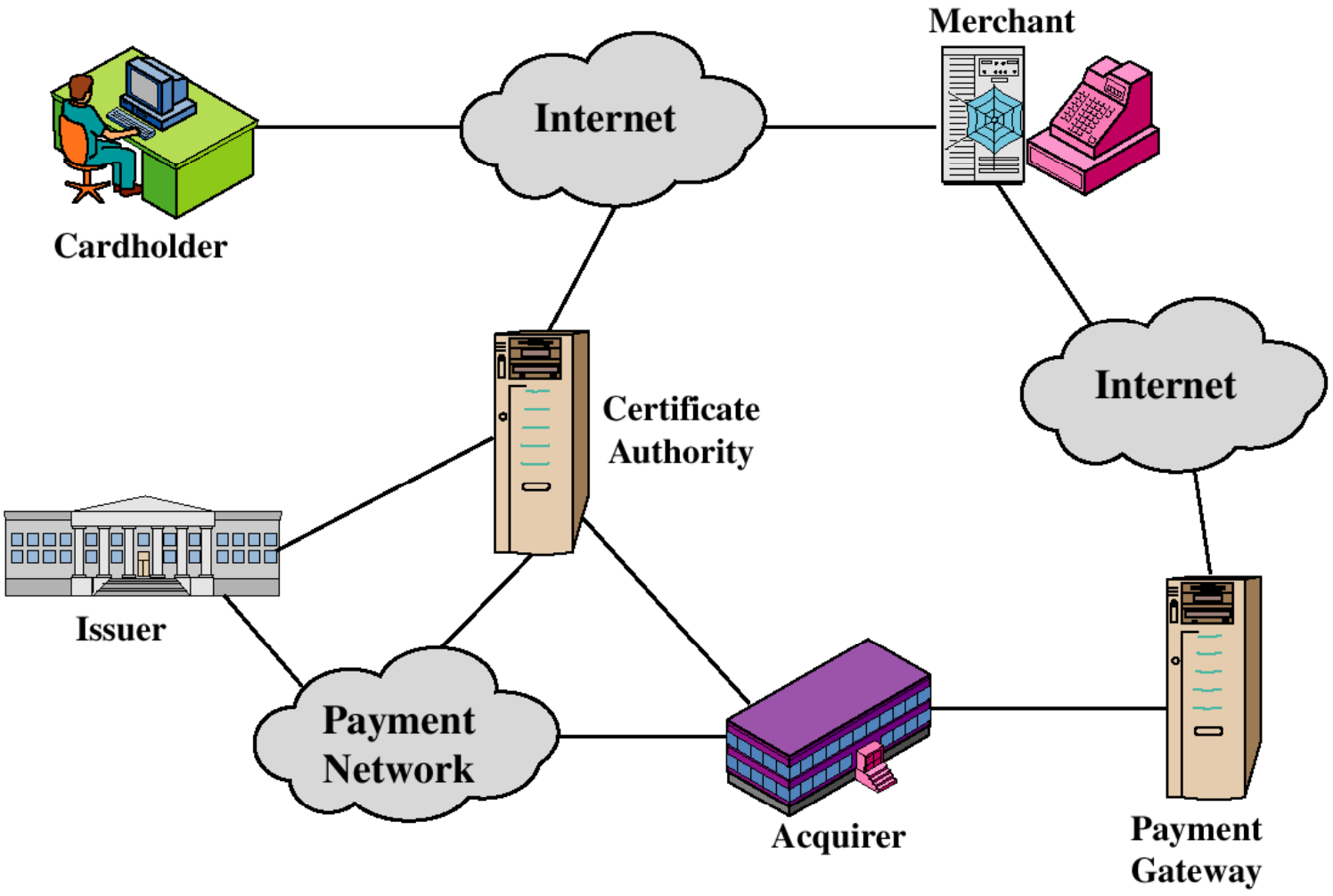
SET Services

- Provides a secure communication channel in a transaction.
- Provides tust by the use of X.509v3 digital certificates.
- Ensures privacy.

SET Overview

- Key Features of SET:
  - Confidentiality of information
  - Integrity of data
  - Cardholder account authentication
  - Merchant authentication

SET Participants



Sequence of events for transactions

1. 持卡人打开一个银行账号。
2. 持卡人收到一个作为在线购买或其他交易的信用卡的电子证书，它包括一个公钥和一个有效期限。
3. 商家也收到来自银行的证书，包括商家的公钥和银行的公钥。
4. 持卡人定购货物。
5. 持卡人的浏览器得到来自商家证书的关于商家有效性的确认。
6. 浏览器发送定购信息，信息用商家的公钥加密，支付信息用银行的公钥加密以保证支付只为特定的订购者使用。
7. 商家通过检查持卡人证书中的数字签名来验证持卡人。
8. 商家把订购者的消息传送到收单行，收单行可以解密信用卡号，并通过认证验证签名；
9. 收单行向发卡行查问，确认用户信用卡是否属实；
10. 发卡行认可并签证该笔交易；
11. 收单行认可商家并签证此交易；
12. 商家向持卡人传送货物和收据；
13. 交易成功，商家向收单行索款；
14. 收单行按合同将货款划给商家。

Dual Signature

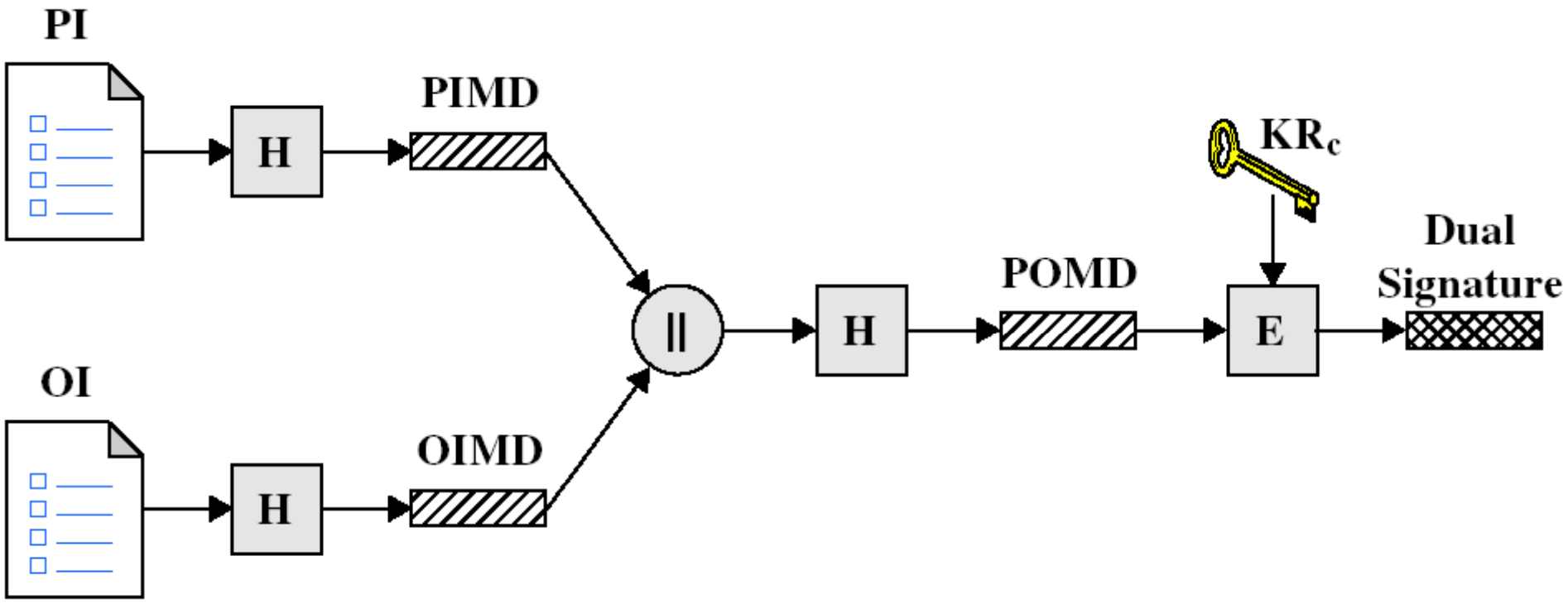
- 它允许将两种数据连接在一起并交给两个不同的实体处理。
- 在这种情况下，消费者想要将订购信息(OI)发送给商家，将支付信息(PI)发送给银行。
  - 商家不必知道消费者的信用卡号码
  - 银行也不必知道消费者订单的细节
  - 消费者可以证明这个支付是用于这次订购而不是用于其他某种货物或服务。

Dual Signature

- 为了看看这种连接的必要性，假设消费者将两个报文发送给商家—签名的OI和签名的PI，接着商家将PI继续传递给银行。如果商家可以截获来自这个消费者的另一个OI，商家可以声明后一个OI是和那个PI一起的而不是原来的OI。连接可以防止这个问题。
- OI PI
- 第一次 电视机 卡号＋￥**5000**
- 第二次 电话 卡号＋￥**50**

Dual Signature

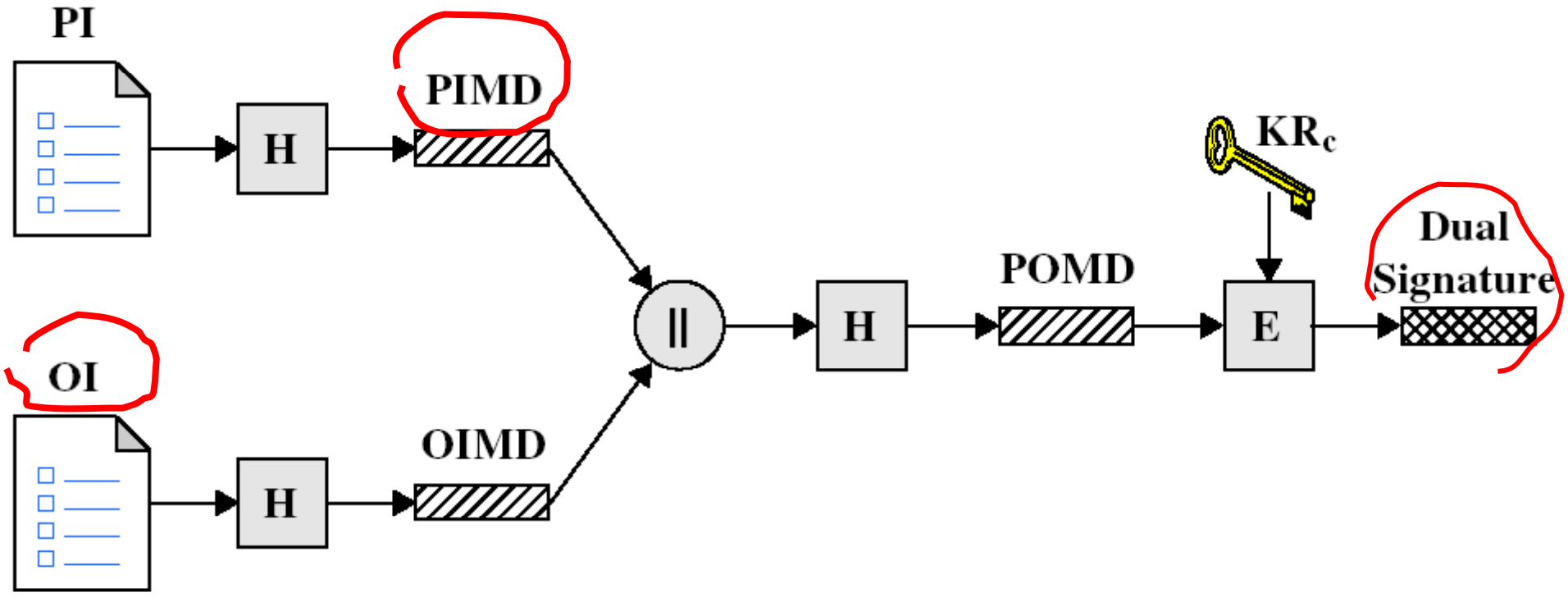
$$DS = E_{KR_c}[H(H(PI) \| H(OI))]$$



PI = Payment Information
OI = Order Information
H = Hash function (SHA-1)
|| = Concatenation

PIMD = PI message digest
OIMD = OI message digest
POMD = Payment Order message digest
E = Encryption (RSA)
KR_c = Customer's private signature key

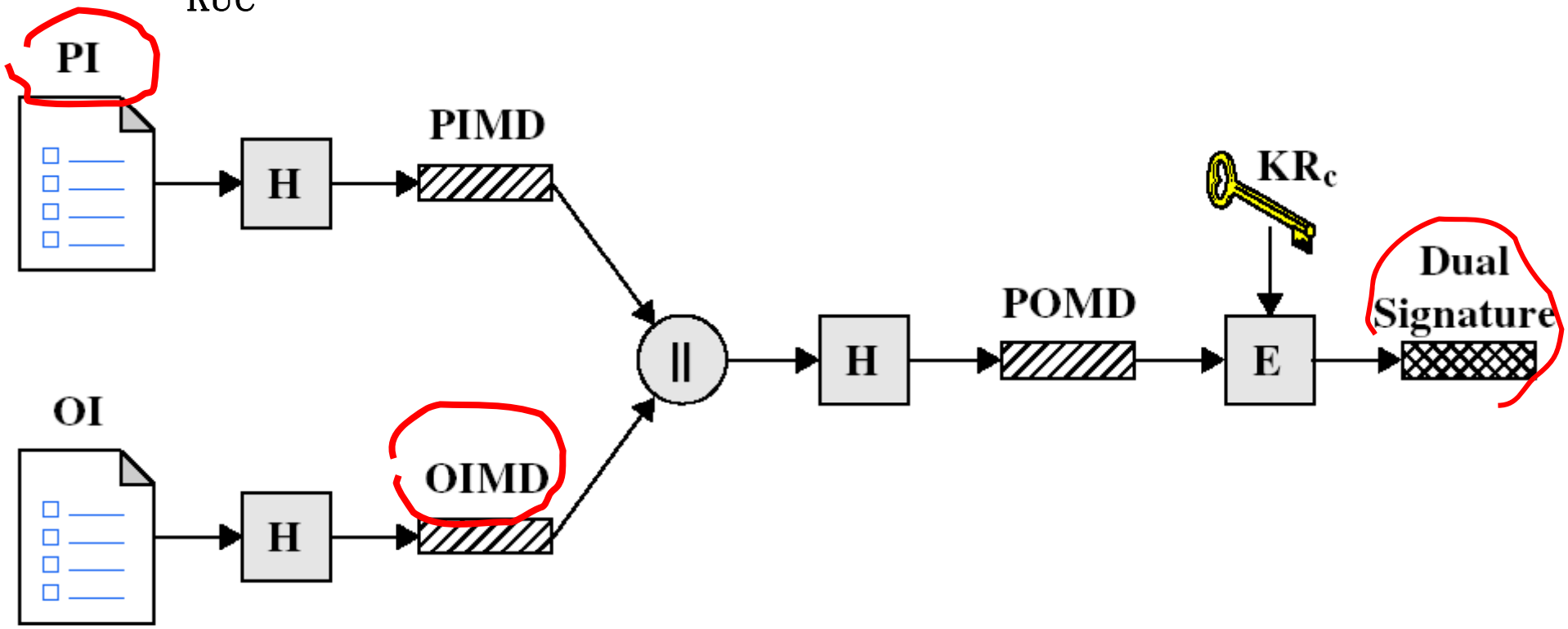
•1 商家收到PIMD、OI、DS，计算H(PIMD||H(OI))和D_{KUc}[DS]，验证该签名。Kuc持卡人的公钥



PI = Payment Information
OI = Order Information
H = Hash function (SHA-1)
|| = Concatenation

PIMD = PI message digest
OIMD = OI message digest
POMD = Payment Order message digest
E = Encryption (RSA)
KR_c = Customer's private signature key

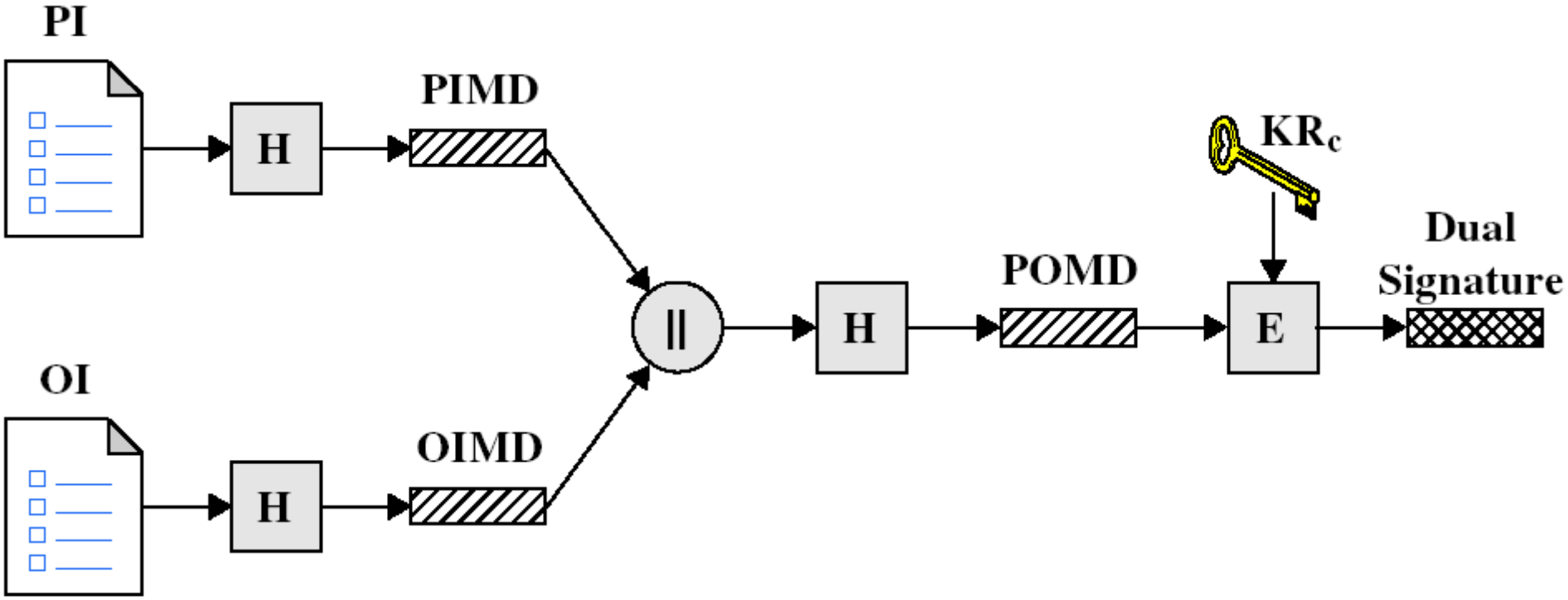
- 2银行收到OIMD、PI、DS,计算H(OIMD||H(PI))和D_{KUc}[DS]，验证该签名。



PI = Payment Information
OI = Order Information
H = Hash function (SHA-1)
|| = Concatenation

PIMD = PI message digest
OIMD = OI message digest
POMD = Payment Order message digest
E = Encryption (RSA)
KR_c = Customer's private signature key

- 3 消费者将OI和PI连接起来并且能够证明这个连接关系。

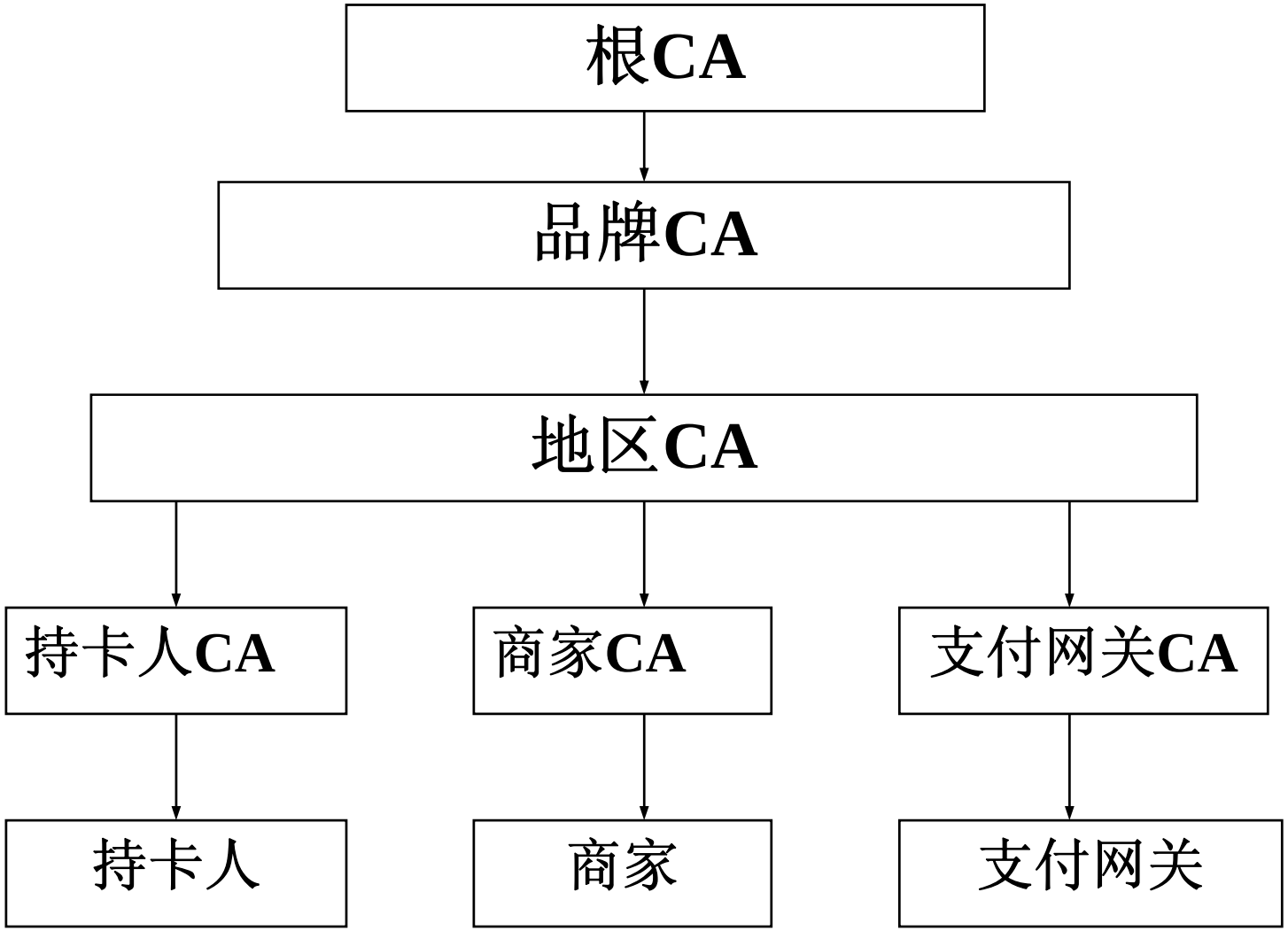


PI = Payment Information
OI = Order Information
H = Hash function (SHA-1)
|| = Concatenation

PIMD = PI message digest
OIMD = OI message digest
POMD = Payment Order message digest
E = Encryption (RSA)
KR_c = Customer's private signature key

- 例如，假设商家为了自己的利益，想要在这个交易中用另一个OI来代替，那么它就必须找到另一个散列数据与已经存在的OIMD相匹配的OI，这被认为是不可行的。因此，商家不能将另一个OI与这个PI相连接。

基于SET的CA的分级结构

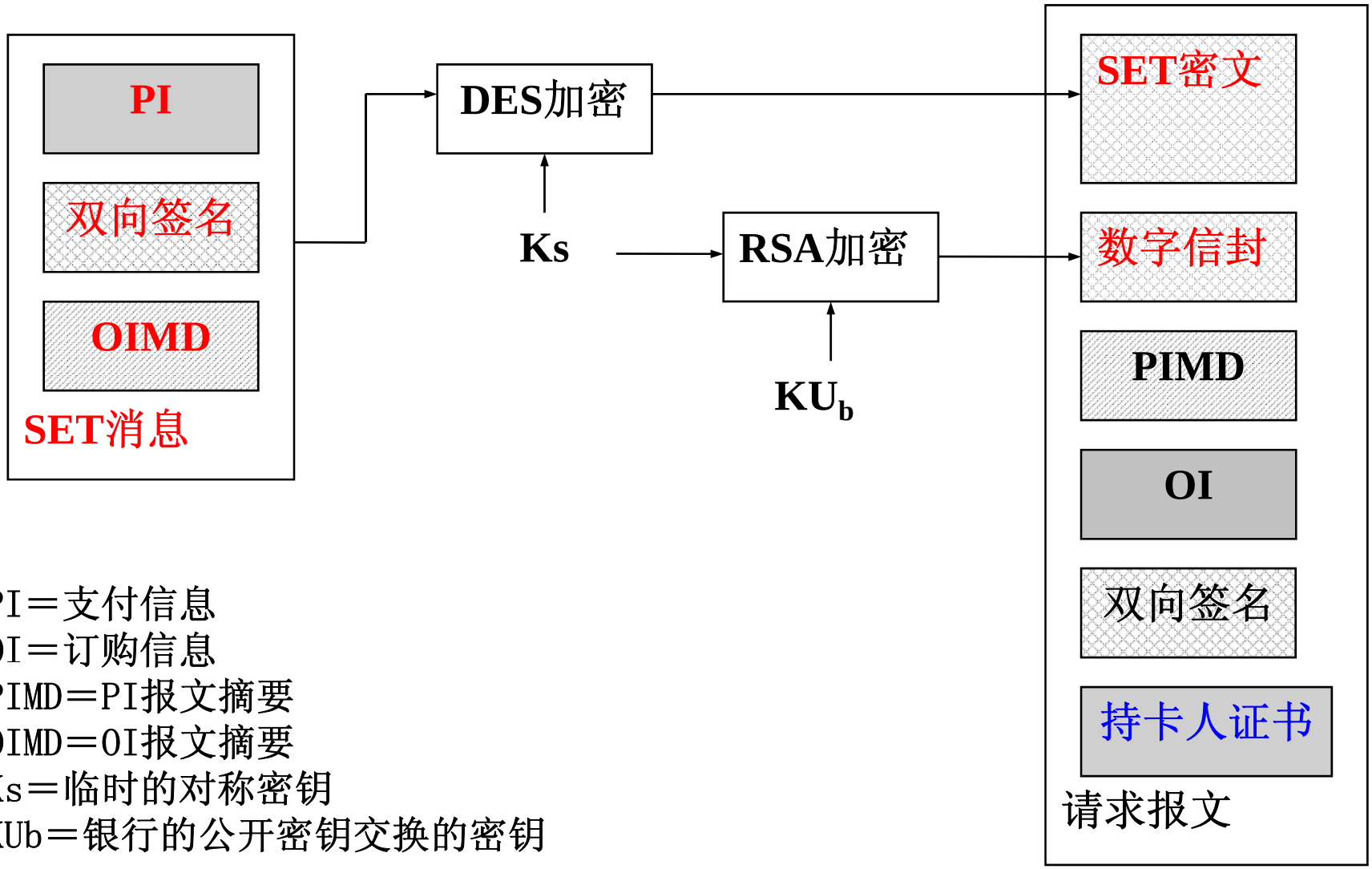


Payment processing

- 说明SET如何在电子商务交易中为支付处理提供安全性，讨论以下交易类型中每一个：
  - 购买请求：用户与商家确定所用支付方式的细节；
  - 支付授权：商家会与银行核实，随着交易的进展，他们将得到付款
  - 支付获取：商家向银行出示所有交易的细节，然后银行以适当方式转移货款。

Payment processing

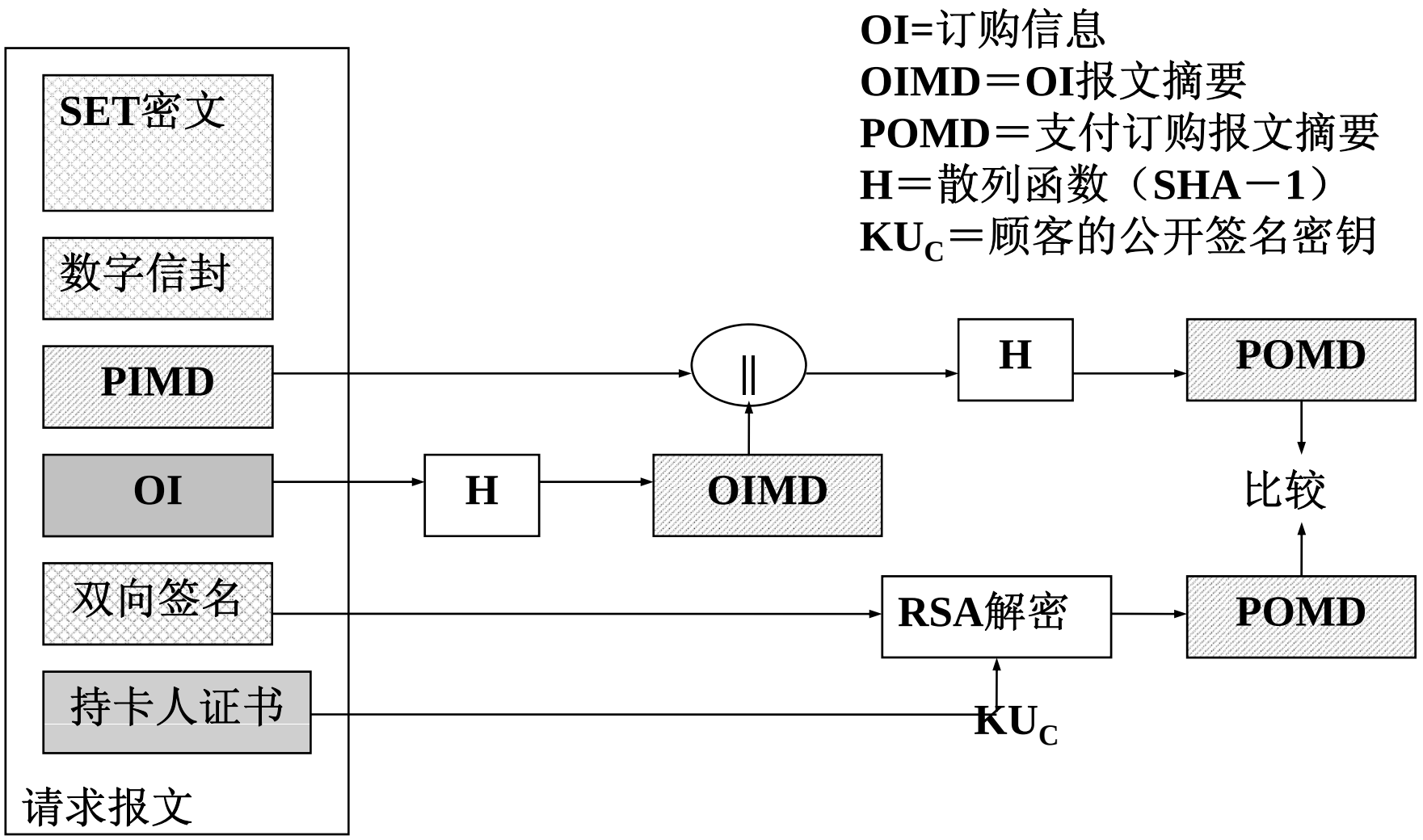
- *1. 购买请求*
- 购买请求由持卡人和商家之间交换的4个消息组成：
- (1)*发起请求*：持卡人软件请求网关证书的一个拷贝；该交易中将使用何种品牌的支付卡。
- (2)*发起响应：*
  - 消息一个唯一的交易标识符。
  - 商家证书以及支付网关的证书
  - 然后用商家的私钥对该信息进行数字签名后发送给持卡人。
- (3)*购买请求*：
  - 收到发起响应消息之后，持卡人软件验证商家和支付网关的证书。
  - 持卡人软件使用OI和PI来产生一个双重签名。
  - 最后，持卡人软件生成一个购买请求消息。
  - 持卡人生成了一次性的对称加密密钥Ks。



PI＝支付信息
OI＝订购信息
PIMD＝PI报文摘要
OIMD＝OI报文摘要
Ks＝临时的对称密钥
KUb＝银行的公开密钥交换的密钥

①与购买相关的信息。这个信息将被商家转发给支付网关
②与订购有关的信息。商家需要这个信息
③持卡人的证书。

- (3)*购买响应*：
  - ①通过持卡人的ＣＡ签名来验证持卡人的证书。
  - ②使用消费者的公开签名密钥来验证双向签名。这样可以确保订购信息在传输过程中没有被篡改，并且它使用了持卡人的私有签名密钥进行了签名。
  - ③处理订购信息，并将支付信息转交给支付网关进行认可。
  - ④最后，商家生成一个购买响应消息，它表明商家已经收到持卡人的请求。
  - 一旦收到了商家的购买响应，持卡人软件就验证商家的证书以及消息内容的数字签名。在此时刻，持卡人软件基于这些消息将采取一些行动，诸发向持卡人显示一个消息，或者用订单的状态来更新数据库。


OI=订购信息
OIMD＝OI报文摘要
POMD＝支付订购报文摘要
H＝散列函数（SHA－1）
KUC＝顾客的公开签名密钥
SET密文
数字信封
PIMD
OI
双向签名
持卡人证书
请求报文
||
H
OIMD
H
POMD
比较
RSA解密
POMD
KUC

Payment processing

- *2. 支付授权*
- ①授权请求：商家软件生成并数字签发一个授权请求，其中包括授权的数量、来自OI的交易标识符以及其他的关于交易的信息。然后，使用支付网关的公钥对这一信息生成一个数字信封。授权请求和持卡人的PI（它也以数字信封的形式传给支付网关）被传送给支付网关。
- ②授权响应：当收到授权请求时，支付网关解密并验证消息（也就是证书和PI）的内容。如果一切都是合法的，那么支付网关会生成一个授权响应消息；然后用商家的公钥对之生成一个数字信封并将它传送加商家。
- 一旦收到了支付网关的授权响应消息，商家就解密数字信封并验证里面的数据。如果本购买是授权过的，则商家通过发送货物或者完成在订单内指明的服务来完成对持卡人订单的处理。

Payment processing

- *3. 支付回复*

- ①回复请求：商家软件生成回复请求，它包括交易的最终数量、交易标识符以及其他的关于交易的信息。然后使用支付网关的公钥对这一消息生成数字信封并传送给支付网关。
- ②回复响应：接收到回复请求并验证了它的内容之后，支付网关将生成一个回复响应。回复响应包括与该请求交易的支付有关的信息。然后使用商家的公钥对这一响应生成数字信封并将之传送回商家。
- 一旦收到了支付网关的回复响应，商家软件就解密数字信封，验证签名和消息数据，用于同从收单行那里接收的支付相匹配。

- 因特网与TCP/IP安全
- SSL协议
- SET协议
- IPSec协议

IPSec 协议 —1 概述

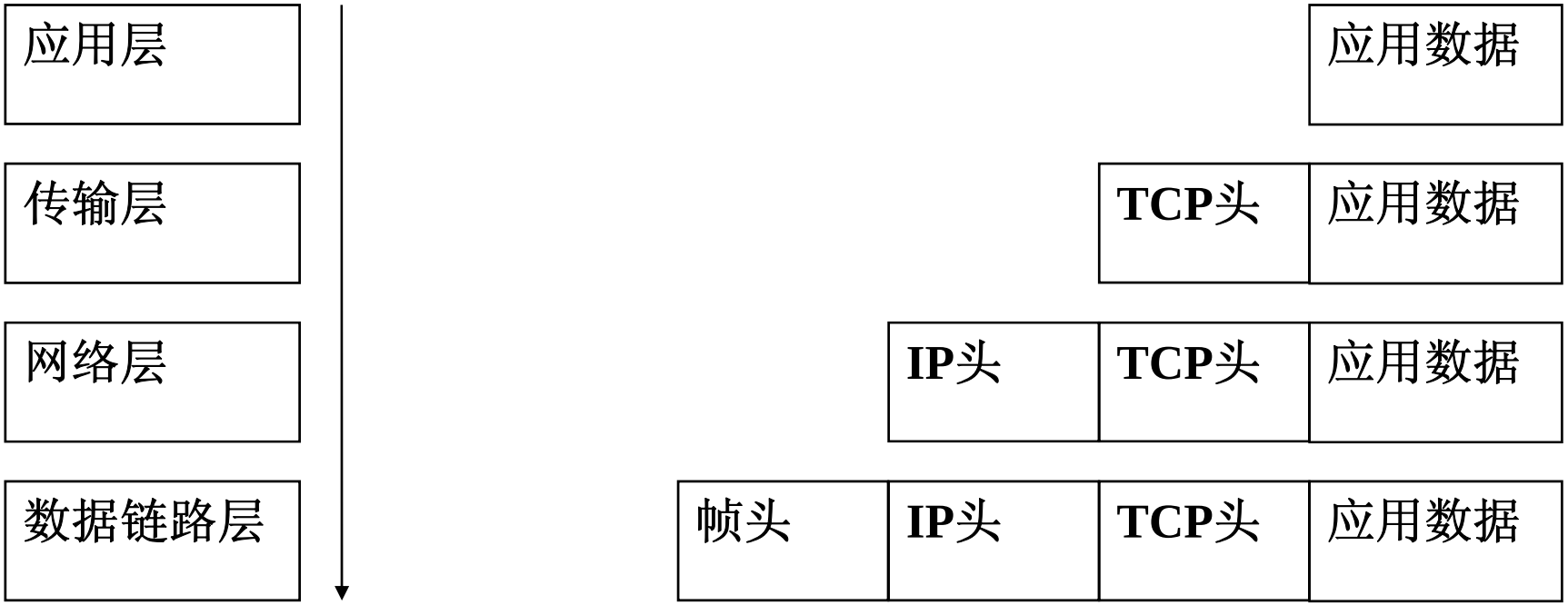
- 在发送IP数据包之前，对IP包的一些重要部分进行加密和验证计算，由接收端对这部分进行解密和验证，从而实现安全传输的目的。
- IPSec实现了网络层的加密和认证，它在网络体系结构中提供了一种端到端的安全解决方案。

IPSec 协议 —1 概述

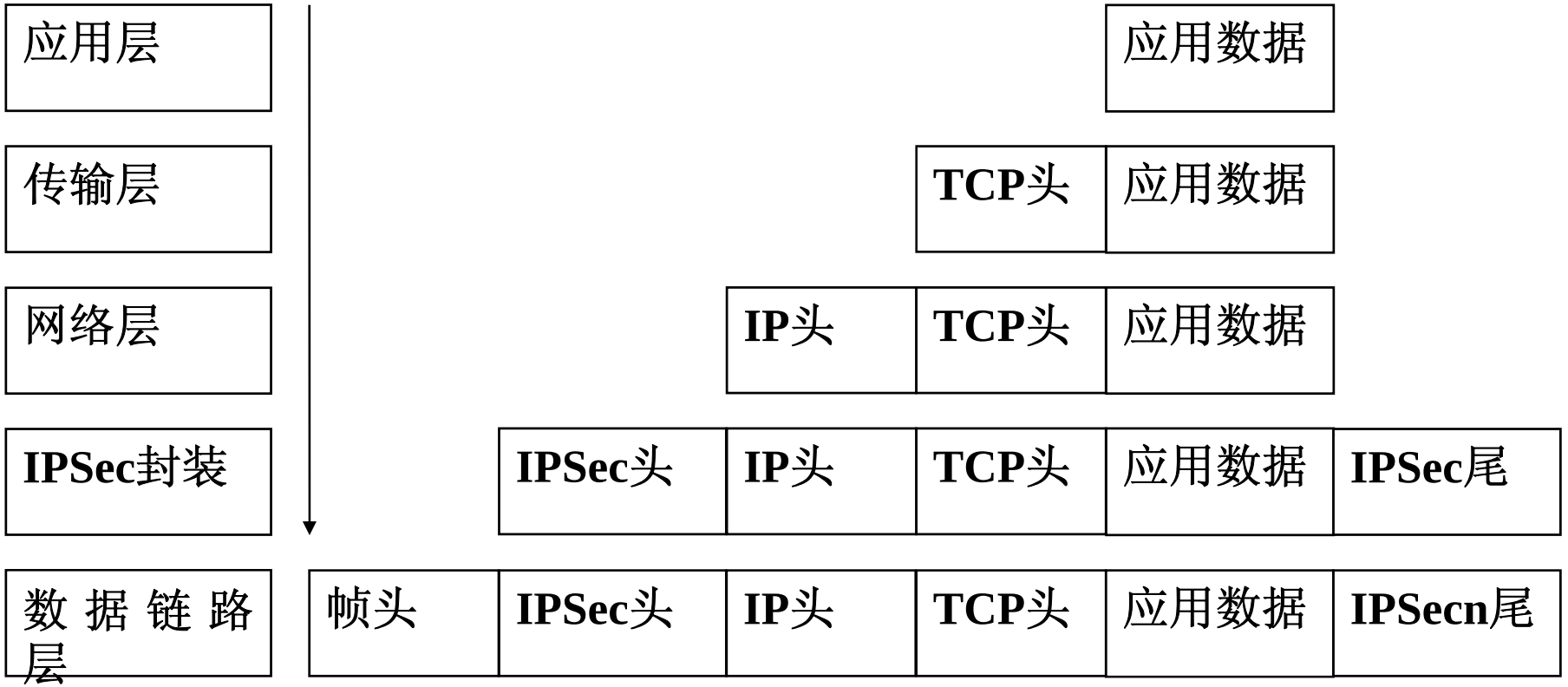
- IPSec提供三种不同的形式来保护通过公有或私有IP网络传送的私有数据。
- (1)认证：通过认证可以确定所接受的数据与所发送的数据是否一致，同时可以确定申请发送者在实际上是真实的，还是伪装的发送者；
- (2)数据完整性验证：通过验证，保证数据在从发送者到接收者的传送过程中没有被修改；
- (3)保密：使相应的接收者能获取发送的真正内容，而无关的接收者无法获知数据的真正内容。

IPSec 协议 —1 概述

- **IPSec**由三个基本要素来提供以上三种保护形式
  - 验证头**(AH)**
  - 封装安全载荷**(ESP)**
  - 互联网密钥管理协议**(IKMP)**。
- 认证协议头和安全加载封装是一套协议中两个不同的机制。所以，它们可以被单独使用，也可以被组合起来使用，可以满足不同的安全要求。

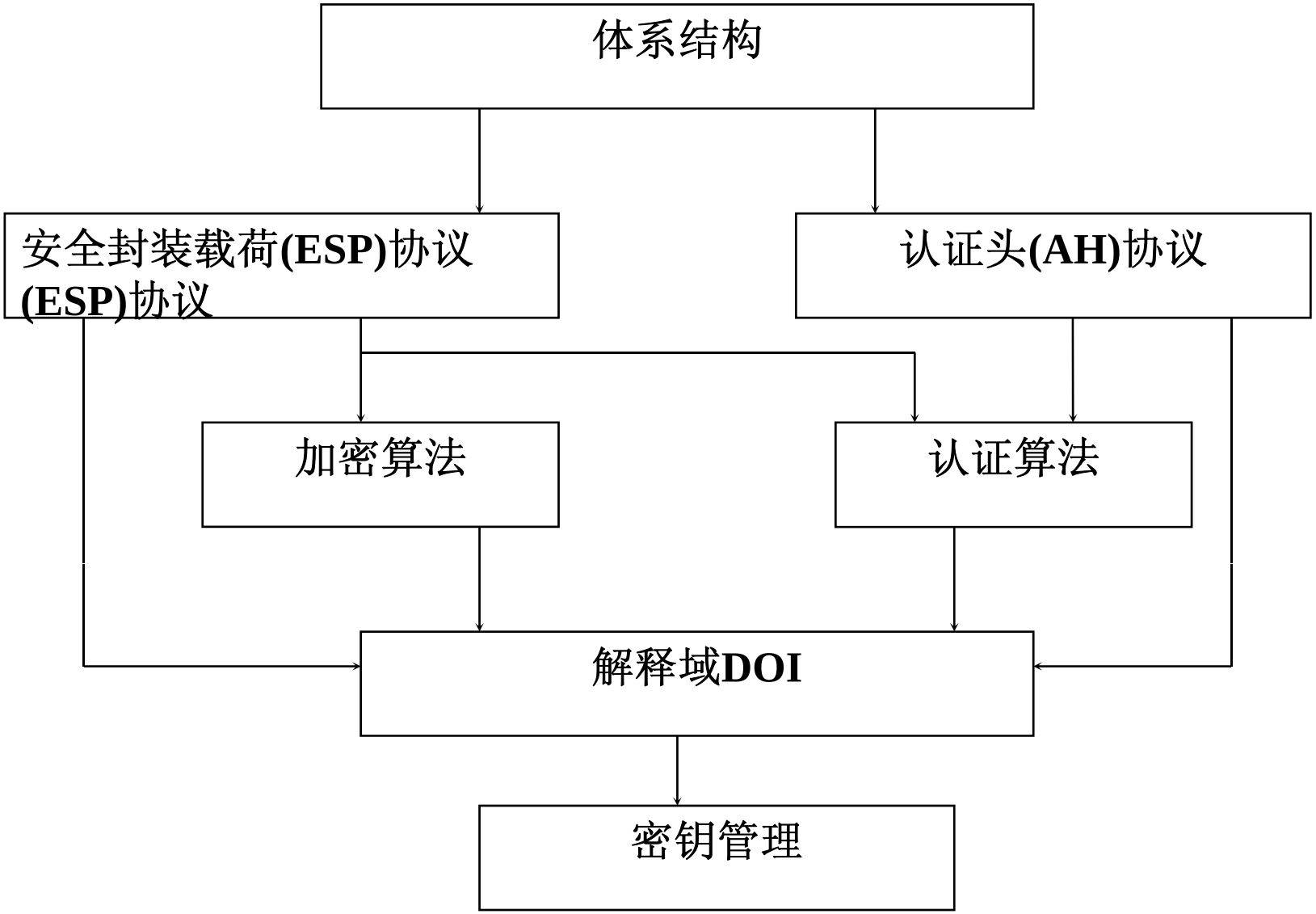


没有IPSec的数据封装


应用层
传输层
网络层
IPSec封装
数 据 链 路 层
应用数据
TCP头
应用数据
IP头
TCP头
应用数据
IPSec头
IP头
TCP头
应用数据
IPSec尾
帧头
IPSec头
IP头
TCP头
应用数据
IPSecn尾

有IPSec数据封装

IPSec 协议 —2 IPSec的安全体系结构



- 各个模块的功能是：
  - 体系结构：覆盖了定义IPSec技术的一般性概念、安全需求、定义和机制
  - 安全封装载荷(ESP)：是插入IP数据包内的一个协议头，具有为IP数据包提供机密性、数据完整性、数据源认证和抗重传攻击等功能。
  - 认证头(AH)：是插入IP数据包内的一个协议头，具有为IP数据包提供数据完整性、数据源认证和抗重传攻击等功能。
  - 加密算法：一组文档描述了怎样将不同的加密算法用于ESP。
  - 认证算法：一组文档描述了怎样将不同的认证算法用于AH和ESP可选的鉴别选项。
  - 密钥管理：描述密钥管理机制的文档。
  - 解释域DOI：包含了其他文档需要的为了彼此间相互联系的一些值。这些值包括经过检验的加密和认证算法的标识以及操作参数，例如密钥的生存期。

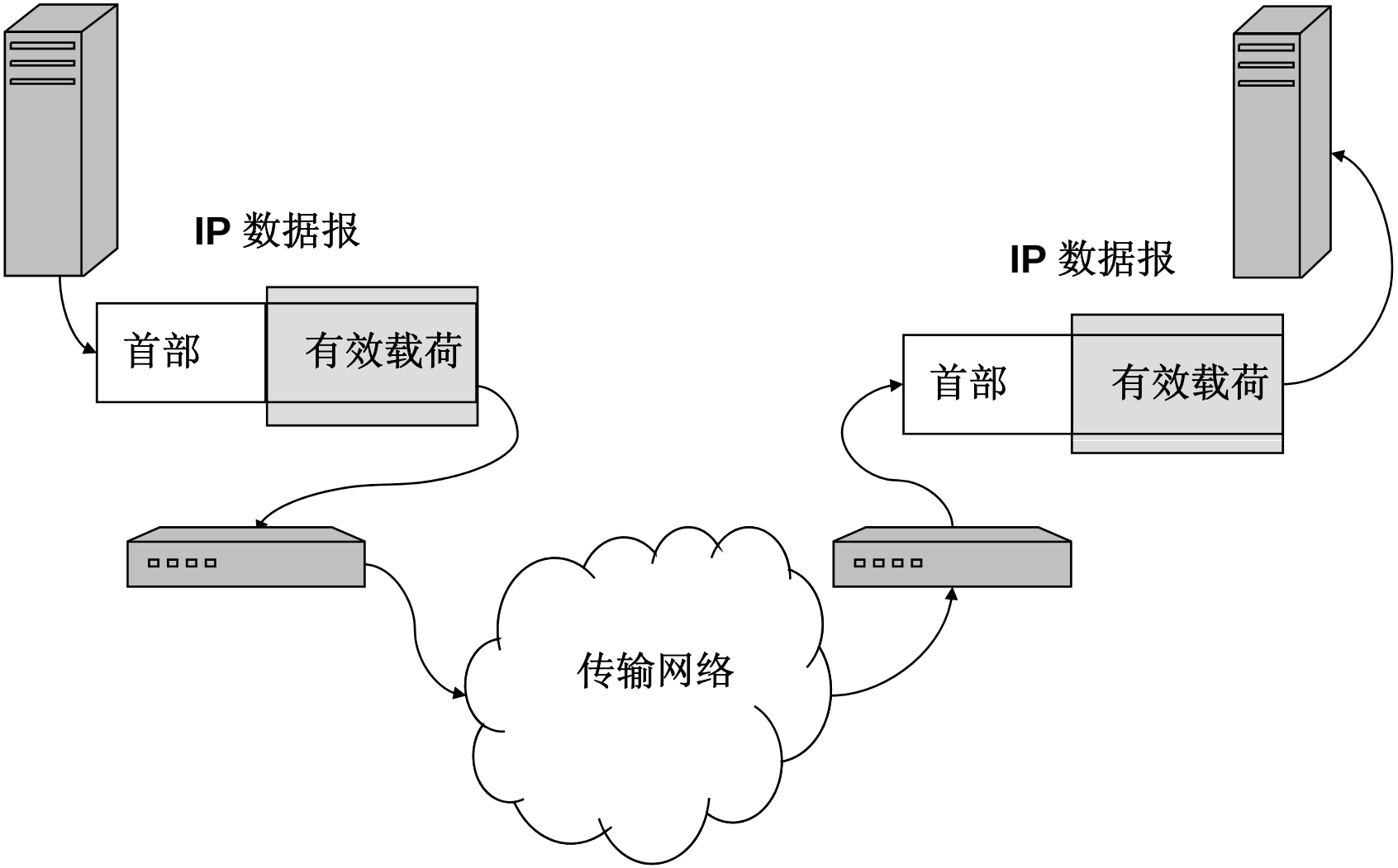
IPSec 协议 —3 IPSec服务

服务类型	AH	ESP(只加密)	ESP(加密并认证)
访问控制	Y	Y	Y
无连接完整性	Y	N	Y
数据源的鉴别	Y	N	Y
拒绝重放攻击	Y	Y	Y
保密性	N	Y	Y
有限的通信流保密性	N	Y	Y

IPSec 协议 —4 IPSec的工作模式

1传输模式

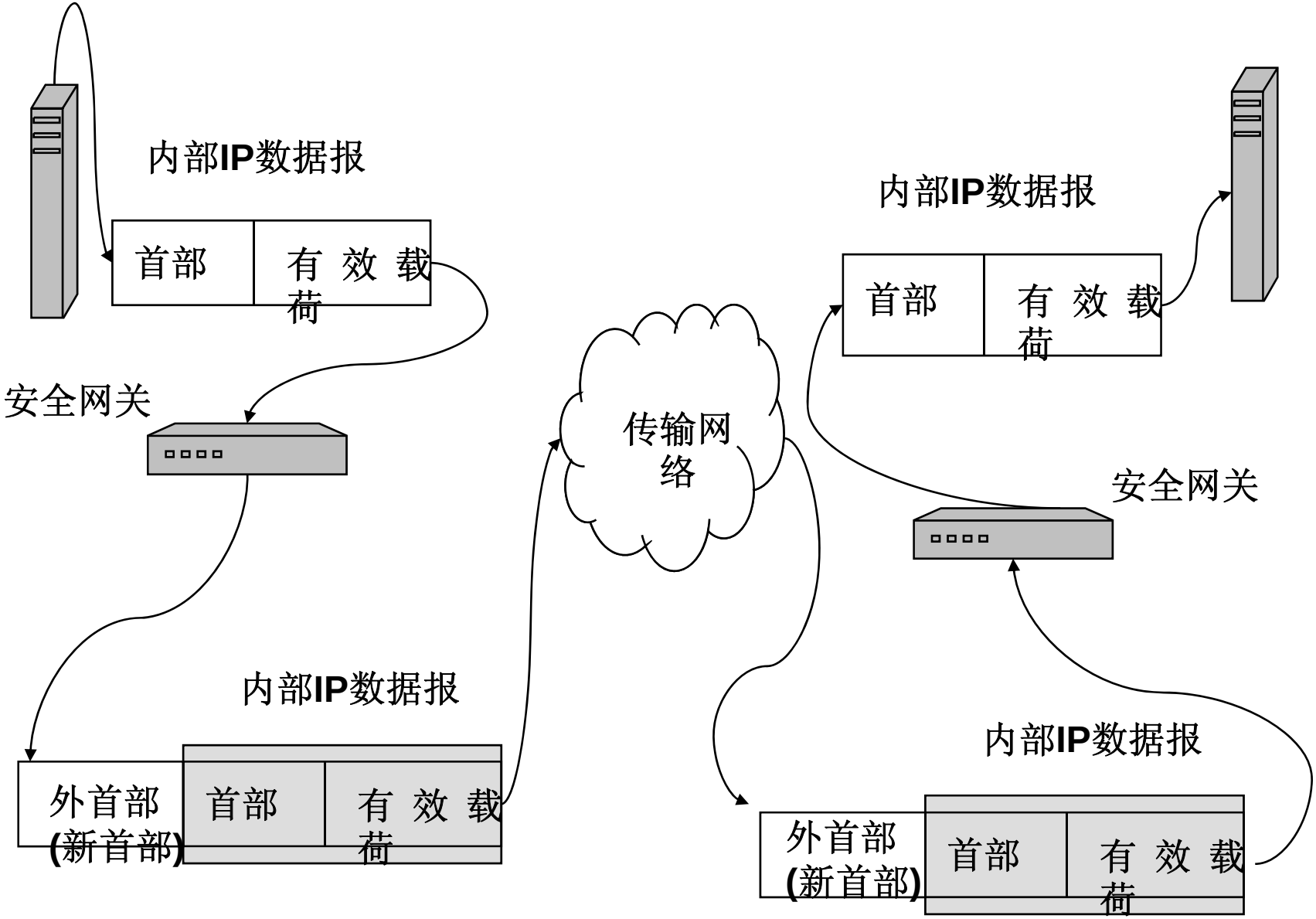
- IPSec传输模式主要对IP包的部分信息提供安全保护，即对IP数据包的上层协议数据提供安全保护。
- 当采用AH传输模式时，主要为IP数据包提供认证保护；而采用ESP传输模式时，主要对IP数据包的上层信息提供加密和认证双重保护。这是一种端到端的安全，IPSec在端点执行加密认证、处理，在安全通道上传输，因此主机必须配置IPSec。


IP 数据报
首部
有效载荷
IP 数据报
首部
有效载荷
传输网络

IPSec 协议 —4 IPSec的工作模式

隧道模式

- IPSec隧道模式对整个IP数据包提供保护。其基本原理是构造新的IP数据包
- 当采用AH遂道模式时，主要为整个IP数据包提供认证保护(可变字段除外)；当采用ESP遂道模式模式时，主要为整个IP数据包提供加密和认证双重保护。
- 这时，对IPSec的处理是在安全网关执行的，因此两端主机不必知道IPSec协议的存在。


内部IP数据报
首部
有 效 载荷
安全网关
传输网络
内部IP数据报
首部
有 效 载荷
安全网关
内部IP数据报
外首部
(新首部)
首部
有 效 载荷
内部IP数据报
外首部
(新首部)
首部
有 效 载荷

IPSec 协议 —5 认证头协议(AH)

- 1. 认证头的功能

- 为IP数据包提供数据完整性、数据源认证和抗重传攻击等功能。
  - 数据完整性是通过消息认证码产生的校验值来保证的；
  - 数据源认证是通过在数据包中包含一个将要被认证的共享秘密或密钥来保证的；
  - 抗重传攻击是通过使用了一个经认证的序列号来实现的。

IPSec 协议 —5 认证头协议(AH)

- 2. 认证头格式
- AH为IP数据包提供了数据完整性和认证服务。
- 注意AH并不提供保密性保护，因此当数据通过一个网络的时候仍然可以被看到。

下一个首部	载荷长度	保留
安全参数索引(SPI)		
序列号		
认证数据ICV(长度可变)		

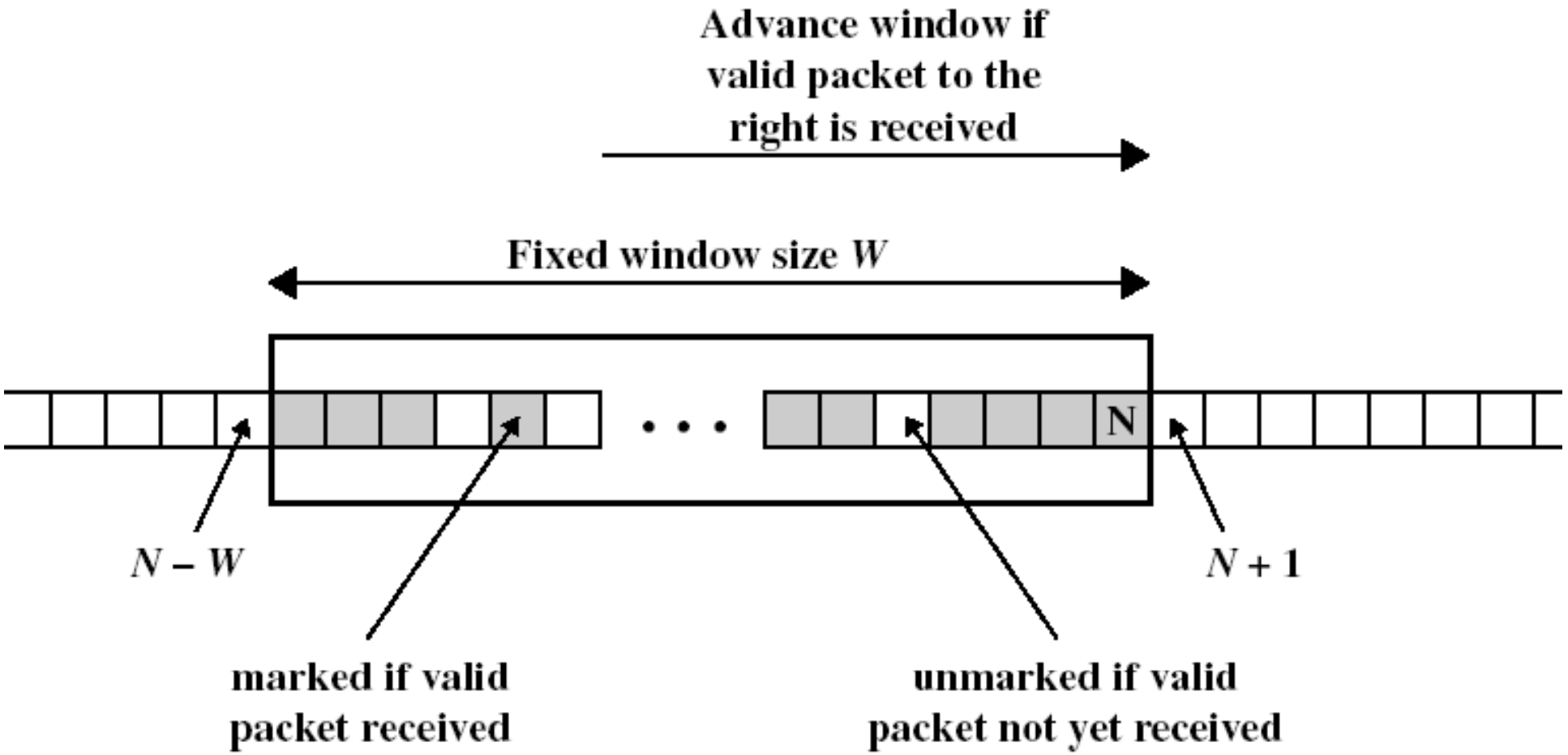
IPSec 协议 —5 认证头协议(AH)

- 3完整性校验值的计算

- ICV是消息认证码(message authentication code，MAC)的一种截断版本，它是用MAC算法计算的。
- 只使用前96bit。
- ICV使用下面的域来计算：
  - 在传输中不变化的IP头域，或者其值在到达使用该AH关联的端点时可以预测的IP头域。
  - AH首部，为了在源和目的地进行计算，必须将认证数据域置为0。
  - 所有的上层协议数据，假定它们在传输中不变化(例如TCP报文段或隧道方式的一个内部IP分组)。

IPSec 协议 —5 认证头协议(AH)

- 4. 抗重放攻击
  - 重放攻击指的是攻击者获得了认证分组的副本，并且以后将它传输到它的目的地址。收到了重复的认证IP分组可能会以某种方式中断服务或者出现其他一些意想不到的后果。序号字段是设计用来阻挡这种攻击的。
  - IPSec数据包专门使用了一个序列号，以及一个“滑动”的接收窗口。
  - 因为IP是无连接、不可靠的服务，协议不能保证分组的按序交付，也不能保证所有的分组都会被交付，因此，IPSec认证文档规定接收者应该实现大小为W的接收窗口。


Advance window if
valid packet to the
right is received
Fixed window size W
N
N – W
N + 1
marked if valid
packet received
unmarked if valid
packet not yet received

- 窗口最左端对应于窗口起始位置的序列号，窗口的右边界则代表目前已经收到的合法分组的最高序号。对于任何已经正确接收的(即经过了正确鉴别的)其序号处于N-W+1到N范围之间的分组，窗口的相应插槽被标记。当收到一个分组时，按照如下步骤进行进入处理：

- ①．如果收到的分组落在窗口之内并且是新的，就进行MAC检查。如果分组被鉴别，就对窗口的相应插槽做标记。
- ②．如果收到的分组落在窗口的右边并且是新的，就进行MAC检查。如果分组被鉴别，那么窗口就向前走，使得该序号成为窗口的右边界，并对窗口的相应插槽做标记。
- ③．如果收到的分组落在窗口的左边，或者鉴别失败，就丢弃该分组。

IPSec 协议 —5 认证头协议(AH)

- 5. 传输模式和遂道模式

•AH服务可以以两种方式来使用：传输(transport)模式和隧道(tunnel)模式。AH的实际位置决定于使用何种模式以及AH是应用于一个IPv4还是一个IPv6数据包。

标准IPv4数据报

原始的IP头 (任何选项)	TCP	数据

传输模式下的IPv4

原始的IP头 (任何选项)	AH(载荷长度, SPI, 序号, MAC)	TCP	数据

认证范围：所有不变的域

标准的IPv6数据报

原始的IP头 (任何选项)	扩展项头(如果有)	TCP	数据

传输模式下的IPv6

原始的IP头 (任何选项)	逐跃点、目的 、路由、分段	AH(载 荷 长 度 , SPI, 序号, MAC)	目的 选项	TCP	数据

认证范围：所有不变的域

隧道模式下的IPv4

认证范围：所有不变的域

隧道模式下的IPv6

认证范围：所有不变的域

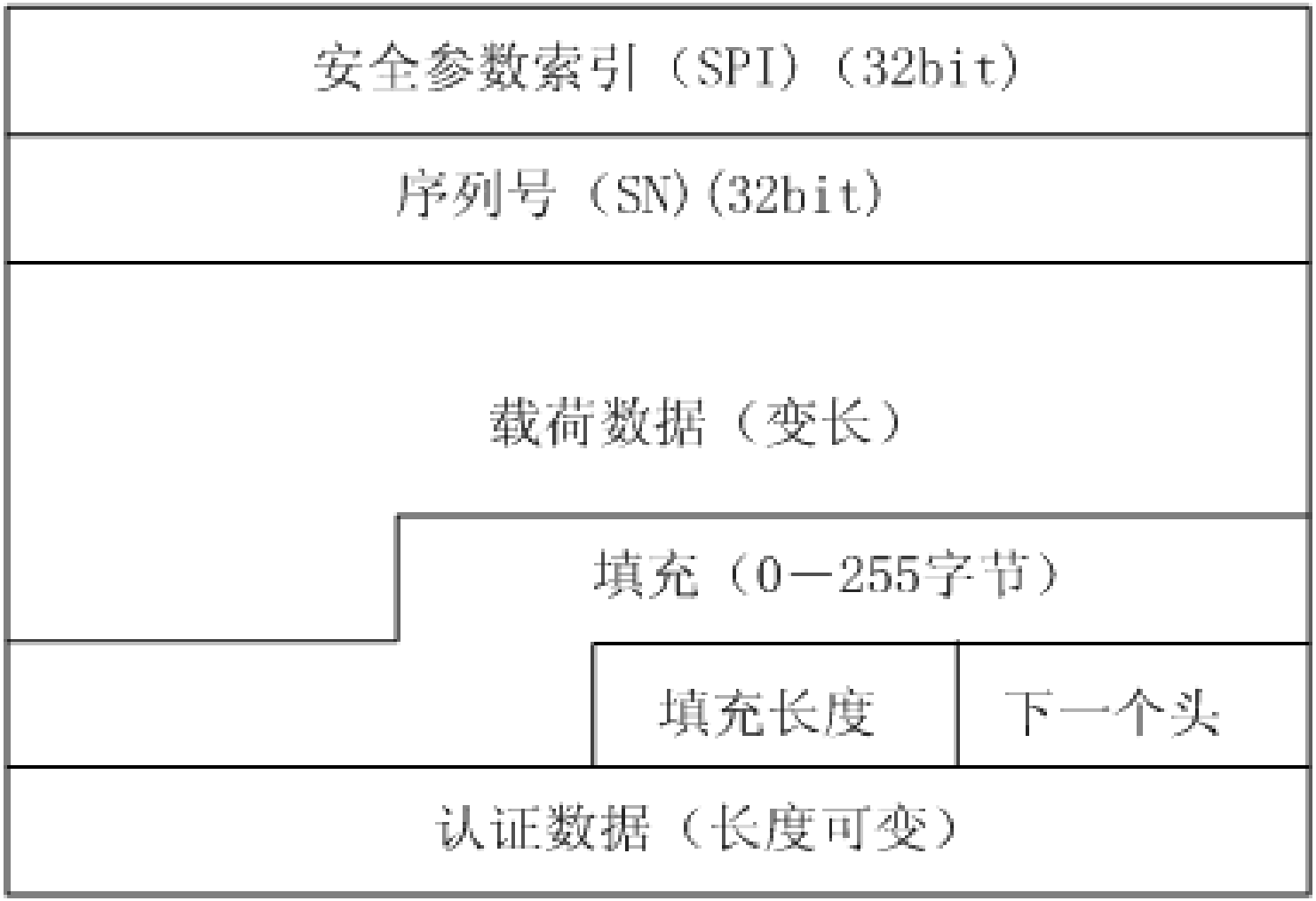
IPSec 协议 —6封装安全载荷协议(ESP)

- 1 ESP功能

•ESP主要支持IP数据包的机密性，它将需要保护的用户数据进行加密后再封装到新的IP数据包中。另外ESP也可提供认证服务，但与AH相比，二者的认证范围不同，ESP只认证ESP头之后的信息，比AH认证的范围要小。

IPSec 协议 —6封装安全载荷协议(ESP)

- 2．ESP格式



IPSec 协议 —6封装安全载荷协议(ESP)

- 3. 加密与认证算法

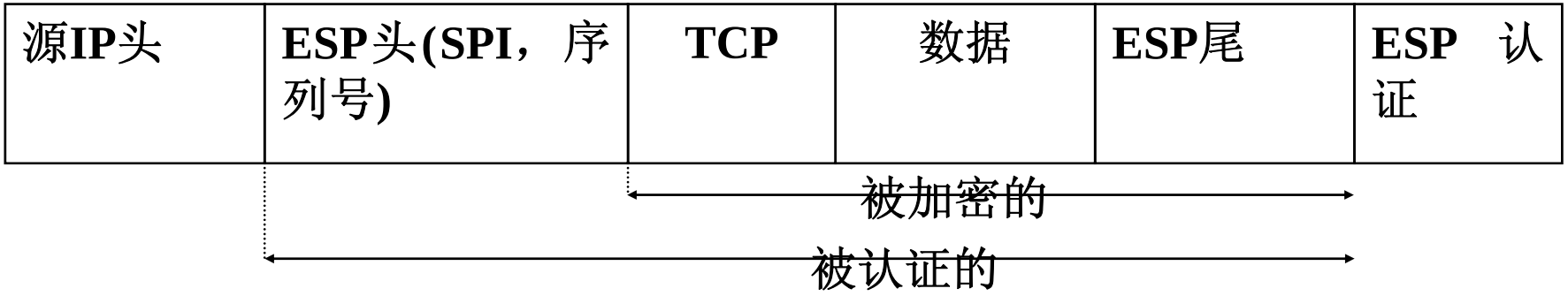
•ESP服务对有效载荷数据、填充、填充长度和下一个首部字段加密。下面是一些已定义的算法：

- 三密钥三重DES
- RC5
- IDEA
- CAST
- Blowfish
- 三密钥三重IDEA

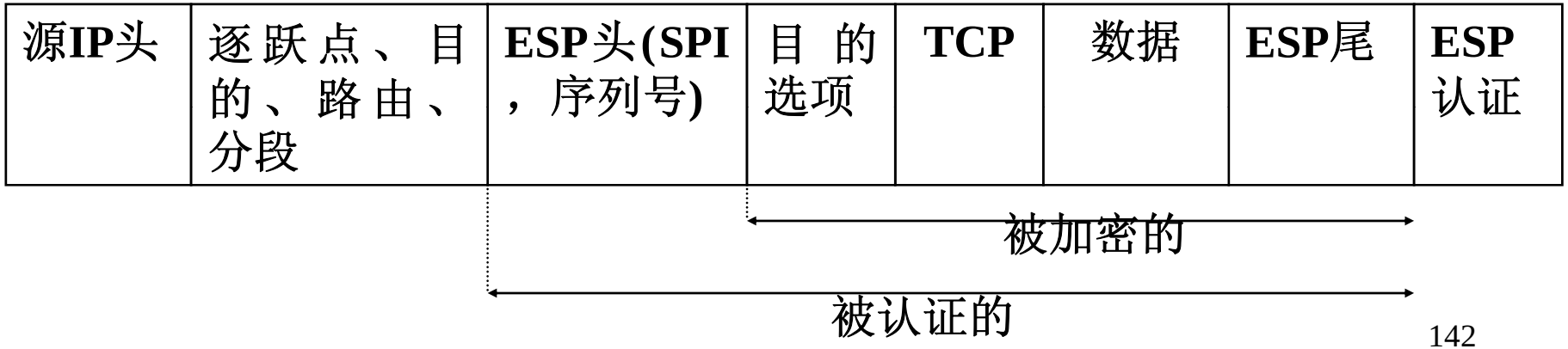
IPSec 协议 —6封装安全载荷协议(ESP)

- 4. 传输模式和隧道模式

传输模式下的**IPv4 ESP**



传输模式下的**IPv6 ESP**



•传输方式的操作总结如下：

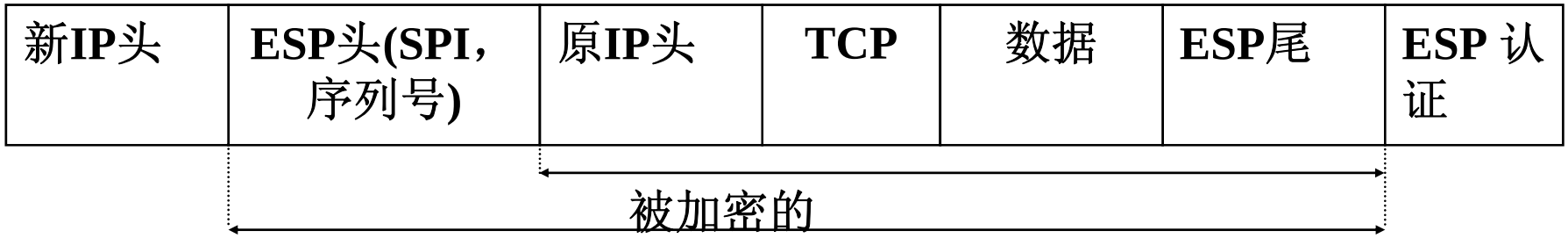
•①在源站，由ESP尾部加上整个传输级的报文段组成的数据块被加密，这个数据块的明文被其密文所代替，以形成用于传输的IP分组。如果认证选项被选中，还要加上认证。

•②然后分组被路由到目的站。每个中间路由器都要检查和处理IP首部加上任何明文的IP扩展首部，但是不需要检查密文。

•③目的结点检查和处理IP首部加上任何明文的IP扩展首部。然后，在ESP首部的SPI基础上目的结点对分组的其他部分进行解密以恢复明文的传输层报文段。

•传输方式操作为使用它的任何应用程序提供了机密性，因此避免了在每一个单独的应用程序中实现机密性，这种方式的操作也是相当有效的，几乎没有增加IP分组的总长度。

隧道模式下的IPv4 ESP



隧道模式下的IPv6 ESP

IPSec 协议 —7 安全关联

- 当利用IPSec进行通信时，采用哪种认证算法、加密算法以及采用什么密钥都是事先协商好的。一旦通信双方取得一致后，在通信期间将共享这些安全参数信息来进行安全信息传输。
- 为了使通信双方的认证算法、加密算法保持一致，相互间建立的联系被称为安全关联**(Security Association，SA)**。
- 也就是在使用AH或ESP之前，先要从源主机到目的主机建立一条网络层的逻辑连接。
- **SA是构成IPSec的基础。**

- 安全关联是单向的，意味着每一对通信系统连接都至少需要两个安全关联。一个是从**A**到**B**，一个是从**B**到**A**。
- 如**A**需要有一个**SA(out)**用来处理外发数据包；一个**SA(in)**用来处理进入数据包。
- 如**B**需要有**SB(out)** 和**SB(in)**
- **SA(out)** 和**SB(in)**共享一个加密参数
- **SA(in)** 和**SB(out)**共享一个加密参数
- 因此对外发和进入的**SA**分别需要维护一张单独的数据表

- 一个SA由三个参数来惟一地标识：

⑩● 安全参数索引(SPI)：该比特串惟一地标识一个与某一安全协议(例如AH或者置SP)相关的安全关联。SPI位于AH和ESP头内，因此接收系统可以挑选出用于处理接收到的IP数据包的SA。

⑩● 目的IP地址：该参数表明该SA的目的IP地址。端点可能会是最终用户系统或者一个如网关或防火墙这样的网络系统。尽管从概念上讲该参数可以是任意地址类型(多播、广播等)，但是目前它只能是一个单播地址。

● 安全协议标识符：该参数表明本关联是一个**AH**安全关联还是一个**ESP**安全关联。

- IPSec实施方案最终会构建一个SA数据库(SADB)，由它来维护IPSec协议用来保障数据包安全的SA记录。
- 在IPSec保护IP报文之前，必须先建立一个安全联盟。安全联盟可以手工或动态建立。Internet密钥交换(IKE)用于动态建立安全联盟，IKE代表IPSec对SA进行协商，并对SADB进行填充。

IPSec 协议 —8 安全数据库

- 安全策略数据库**(Security Policy Database，SPD)**和安全关联数据库**(Security Association Database，SAD)**。
- **SPD**指定了决定所有输入或者输出的**IP**通信部署的策略。
- **SAD**包含有与当前活动的安全关联相关的参数。
- 安全关联是双方所协商的安全策略的一种描述。

IPSec 协议 —8 安全数据库

- *1 安全策略数据库*
- 安全策略决定了为一个包提供的安全服务
- 根据“选择符”对数据库进行检索，选择符从网络层和传输层头提取出来
- **IP**包的外出和进入都需要查询**SPD**，以判断为这个包提供的安全服务有哪些。

•①目的IP地址：这可以是一个单一的IP地址、一个地址列表或者是一个通配的地址。多个地址和通配地址用于多于一个的源系统共享同一个SA的情况(例如，位于一个网关之后)。

•②源IP地址：这可以是一个单一的IP地址、一个地址范围或者是一个通配的地址。

•③名称：这可以是一个X．500特定名称(DN)或者是一个来自操作系统的用户标识符。

•④传输层协议：这可以从IPv4协议域或者IPv6的下一个头域中得到。它可以是一个单独的协议号码、一个协议号的列表，或者是一个协议号码范围。

•⑤源端口和目的端口：这些端口可以是单个的UDP或TCP端口值，真正应用协议的便是这些端口。如果端口不能访问，便需要使用通配符。

•⑥数据敏感级：这被用于提供信息流安全的系统(例如无分级的或者秘密的)。

IPSec 协议 —8 安全数据库

- 2．安全关联数据库
- 下面的参数用于定义一个SA：
- ①序列号计数器：用于生成位于AH或者ESP头中的序列号域的一个32比特值。
- ②序列号计数器溢出：这是一个标志。
- ③抗重放窗口：一个32位计数器，用于决定一个输入的AH 或者ESP数据包是否是一个重放包。
- ④AH信息：与使用AH有关的参数(如认证算法、密钥和密钥生存期)。
- ⑤ESP信息：与使用ESP有关的参数(如加密算法、密钥、密钥生存期和初始化值)。

- ⑥SA的生存期：一个时间间隔或者字节计数，用于指定一个SA使用的持续时间。当该持续时间已结束时，必须用一个新的SA(以及新的SPI)来替代该SA，或者终止该SA，同时该参数中包括一个应该采取何种动作的标识。
- ⑦IPSec协议模式：指定对于该SA的通信流所使用的操作模式(传输模式、隧道模式，或通配模式)。

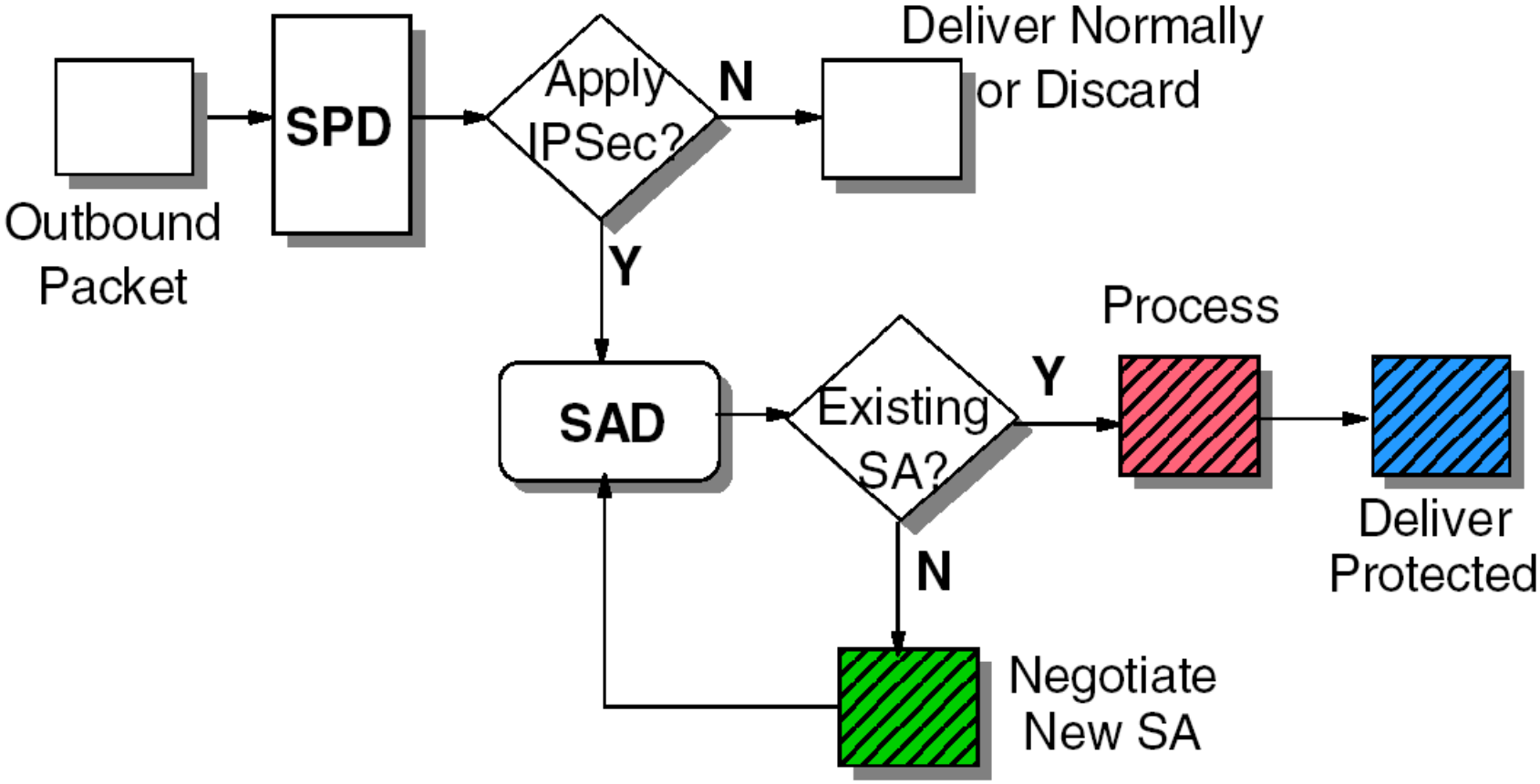
IPSec 协议 —8 安全数据库

•3．IPSec处理方式

•IPSec处理分为两类：外出处理和进入处理。

•外出处理过程中，传输层的数据包流进IP层，IP层检索SPD数据库，判断应为这个包提供哪些安全服务。需要输入SPD的是前面已经讨论过的“选择符”。

- 至于SPD检索的输出，则可能有下面几种情况：
- ①丢弃这个包。此时包不会得以处理，只是简单地丢掉。
- ②绕过安全服务。在这种情况下，IP层会在载荷内增添IP头，然后分发IP包。
- ③应用安全服务。在这种情况下，假如已建立了一个SA，就会返回指向这个SA的指针；
- 假如尚未建立SA，就会调用IKE，将这个SA建立起来。如果SA已经建立，SPD内便会包含指向SA或SA集束的一个指针(具体由策略决定)。如果策略的输出规定强行将IPSec应用于数据包，那么在SA正式建立起来之前，包是不会传送出去的。


Outbound
Packet
SPD
Apply
IPSec?
N
Deliver Normally
or Discard
Y
SAD
Existing
SA?
Y
Process
Deliver
Protected
N
Negotiate
New SA

- 进入处理有别于外出处理。如果收到的**IPSec**包是一个分段，必须把它保留下来，直到这个包的其他部分收完为止。
- 收到IP包后，*假如包内根本没有包含IPSec头*，那么安全层就会对策略进行检查，判断该如何对这个包进行处理。它会用选择符字段来检索SPD数据库。策略的输出可能是下述三种选择：丢弃、绕过或应用。如果策略的输出是丢弃，那么数据包就会被丢弃；如果是应用，但SA没有建立，包同样会被丢弃。否则，就将包传递给下一层，作进一步的处理。

- 如果IP包中包含了IPSec头，就会由IPSec层对这个包进行处理。
- IPSec层会从IP数据报中提取出SPI、源地址和目标地址。它会利用<SPI，目标地址，协议>字元组对SADB数据库进行检索。
- 协议值要么是AH，要么是ESP。根据这个协议值，这个包的处理要么由AH层，要么由ESP层进行。
- 协议载荷处理完之后，需要查询策略，对载荷进行检验。选择符则用作获取策略的依据。验证过程包括：检查SA的使用是否得当(也就是说，SA中的源和目标地址是否与策略对应)，以及SA保护的传送层协议是否和要求的相符。

IPSec 协议 —8 安全数据库

- 4. 组合安全关联

•使用一个单一的**SA**，**AH**或者**ESP**来实现**IP**数据包的安全。然而，并没有限制使用多个**SA**，这通常称为**SA**束**(bundle)**。**SA**集结的顺序是通过安全策略来定义的。

•IPSec体系结构文档列出了IPSec兼容的主机或安全网关必须支持的四种SA组合的例子。

•这些例子如下图所示，图中，每种情况的下面部分表示的是元素的物理连接；上面部分通过一个或多个嵌套SA表示了逻辑连接。每个SA可以是AH或者ESP。对于主机到主机的SA，可以是传输模式或者隧道模式；其他情况则必须是隧道模式。

•①端到端主机之间应用IPSec

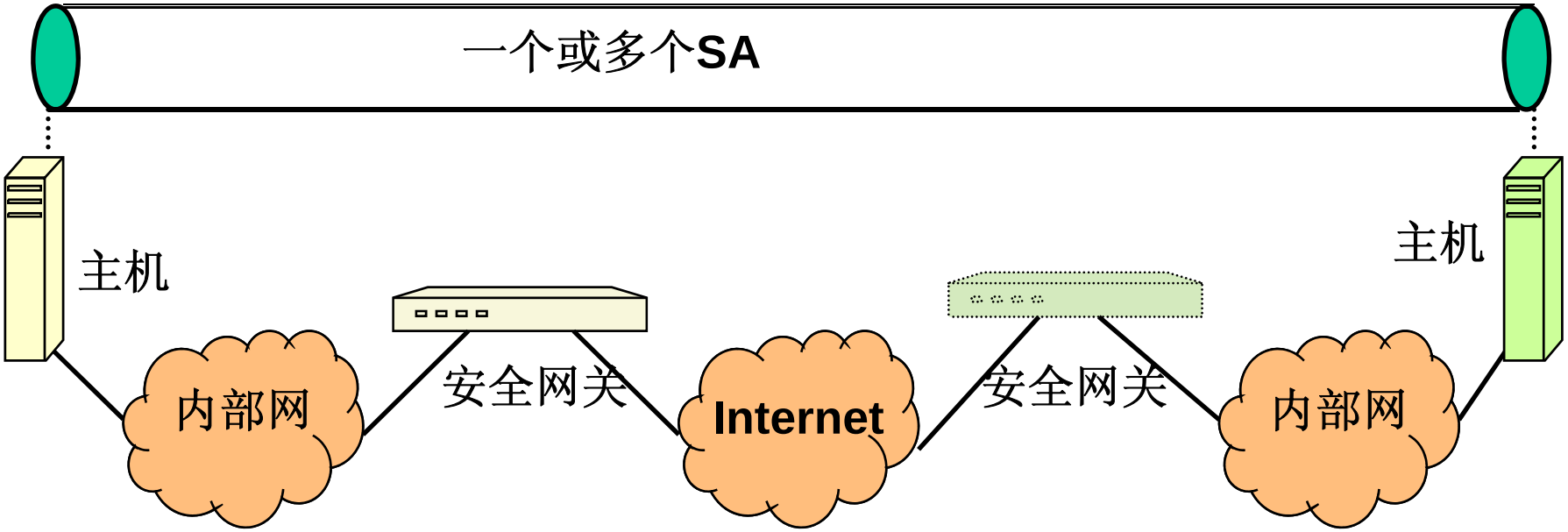
•该情况中，对实现IPSec的两个端系统提供安全业务，两个端系统必须有共享的秘密密钥。此时SA的组合方式可能有：

•传输模式下的AH；

•传输模式下的ESP；

•AH后跟ESP，两者都处于传输模式下（即AH　SA在ESP　SA的内部）；

•上面三种情况的任一种在隧道模式下的**AH或ESP**里面



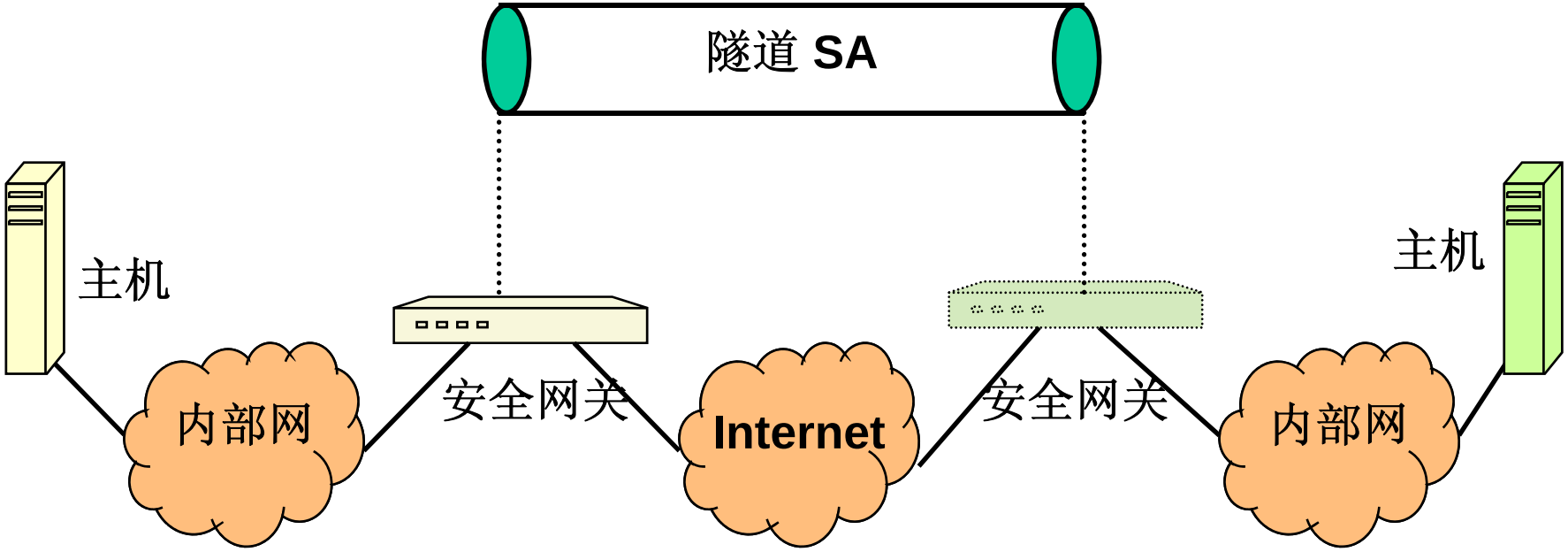
原始的IP头 (任何选项)	AH(载荷长度, SPI, 序号, MAC)	TCP	数据

源IP头	ESP头(SPI，序列号)	TCP	数据	ESP尾	ESP 认证

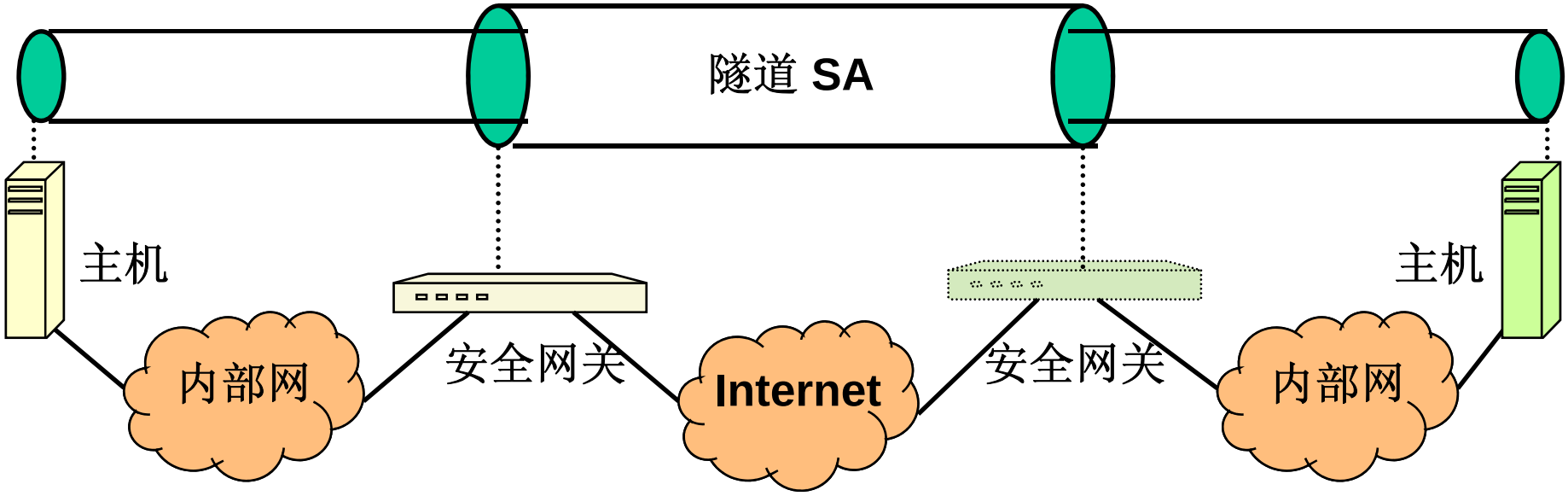
源IP头	ESP 头 (SPI ，序列号)	AH(载 荷 长 度 , SPI, 序 号 , MAC)	TCP	数据	ESP尾	ESP 认证

新的IP头 (任何选项)	AH(载荷长度，SPI, 序号, MAC)	原始的IP头 (任何选项)	TCP	数据

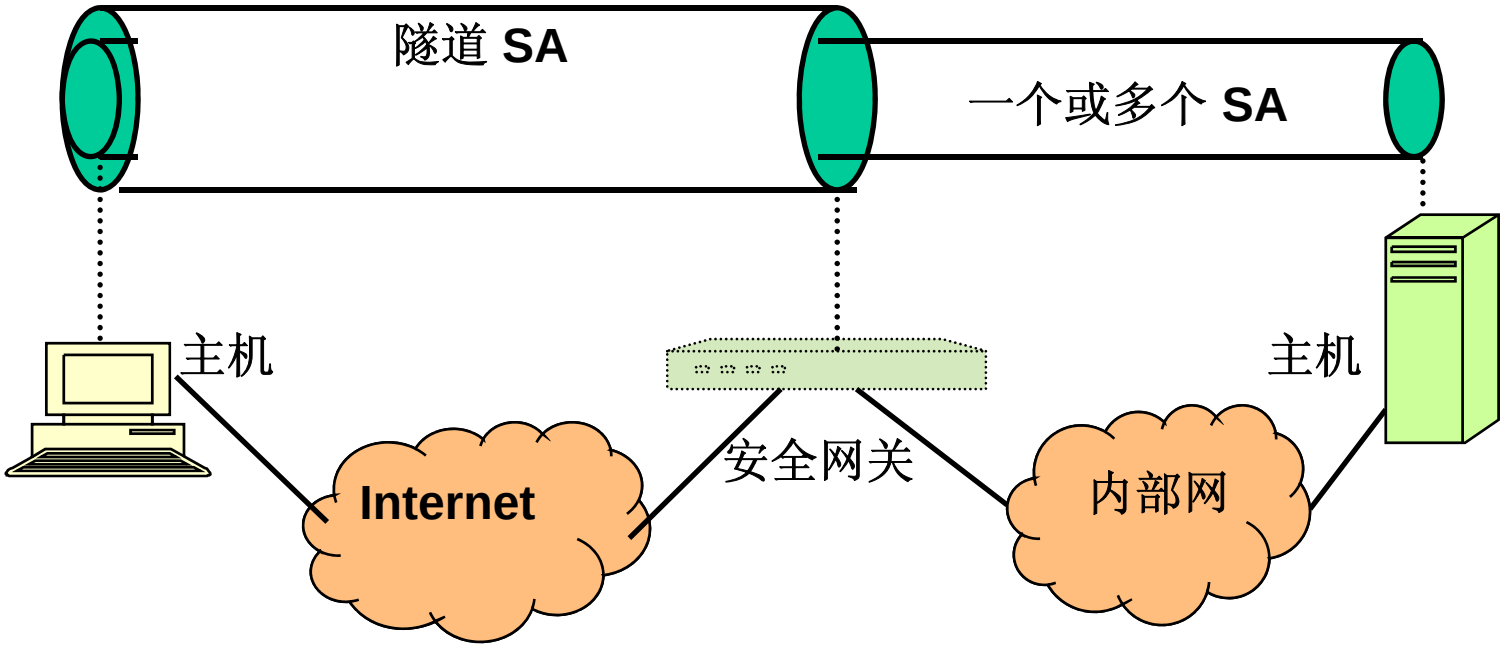
- ②网关到网关之间实现IPSec
- 该情况中，两个主机未实现IPSec，因此仅在两个网关之间提供安全业务。此时仅需要一个隧道模式下的SA，可用于支持AH、ESP或具有认证选择的ESP。由于IPSec安全业务作用于整个内部数据报，因此不需要使用嵌套的隧道模式。



- ③上面①和②的组合
- 该情况在情况②的基础上增加了端到端的安全性。在情况①和情况②中讨论的组合在这里同样允许。网关到网关的隧道为终端系统之间的所有通信量提供了鉴别服务或者机密性，或者两种服务都提供。当网关到网关的隧道是ESP时，它还提供了有限形式的通信量机密性。通过端到端的SA，单独的主机可以为给定的应用程序或给定的用户实现任何需要的补充IPSec。



- ④远程终端支持
- 该情况表示一个具有IPSec的远程主机通过Internet到达某一组织的防火墙，然后访问防火墙后面的本地主机（如服务器或工作站）。此时在远程主机和防火墙之间只要求使用隧道模式，而在远程主机和本地主机之间，则使用一到两个SA。



IPSec 协议 —9 密钥管理和密钥交换

- *1 概述*
- 必要性
- 方式：
  - 手工
  - 自动：通过使用自动的密钥管理协议，可以创建SA所需要的密钥。对于自动管理，可以使用各种各样的协议，但是IKE已经成为了当前的工业标准。
  -

IPSec 协议 —9 密钥管理和密钥交换

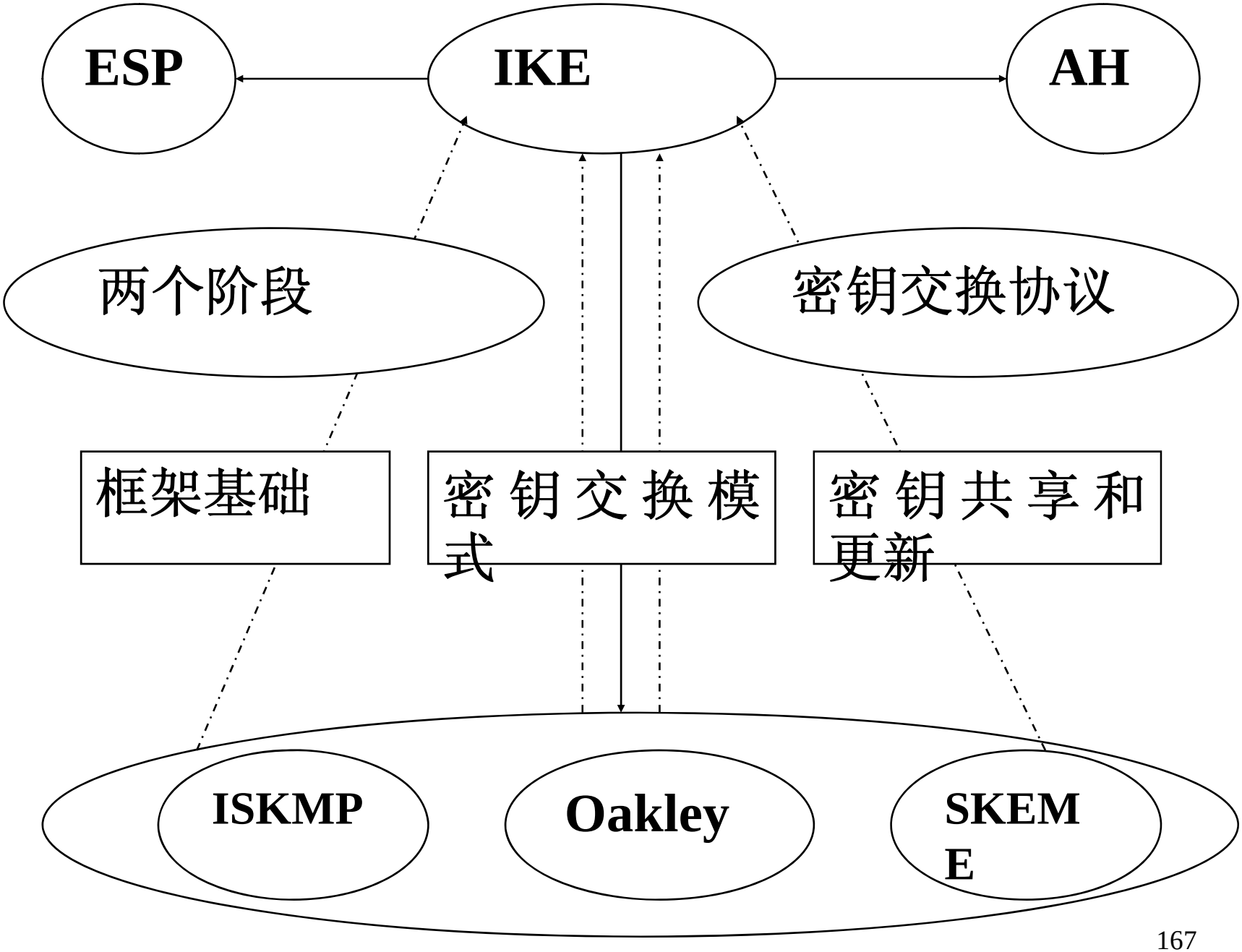
- 2 IKE概述

•默认的IPSec密钥协商方式是Internet密钥交换协议（即Internet Key Exchange，简记作IKE）。

•IKE允许两个实体通过一系列消息得到安全通信的会话密钥，IKE交换为通信双方的消息提供认证和（或）加密，并且针对洪流、重放、欺骗等攻击提供不同程度保护。

•IKE依赖于公私钥加密技术和密码散列函数等机制

•IKE是一个使用已知的UDP端口（端口为500）的应用层协议。


ESP
IKE
AH
两个阶段
密钥交换协议
框架基础
密钥交换模式
密钥共享和更新
ISKMP
Oakley
SKEME

IPSec 协议 —9 密钥管理和密钥交换

- 3 SKEME协议

- SKEME是一种密钥交换协议，SKEME提供了多种模式的密钥交换。
- SKEME的基本模式中，提供了基于公开密钥的密钥交换和Diffie-Hellman密钥交换。
- 它还包含了对Kerberos模型密钥交换的支持。

- SKEME协议的执行过程包含四个主要阶段：cookie阶段、共享阶段、交换阶段和认证阶段。
- (1)cookie阶段的目的是为了防止拒绝服务攻击。
- (2)共享阶段的目的是为了建立会话密钥K_{ab}
  - 前提：需要通信双方都拥有对方的公共密钥。
  - 在这个阶段，A和B先用对方的公共密钥把随机取得的半个密钥加密，然后，用单向哈希函数把这两个半个密钥合并成一个k_{ab}。如下所示。
  - ①：A→B：$\{K_a\}k_B$
  - ②：B→A：$\{K_b\}k_A$
  - 最后，A和B用事先协商好的一个单向哈希函数计算得到$K_{ab}=h(K_a,K_b)$。

•SKEME协议的执行过程包含四个主要阶段：cookie阶段、共享阶段、交换阶段和认证阶段。

•(3)在支持Diffie-Hellman密钥交换的SKEME模式中，交换阶段用来进行密钥交换。

- 假设A选择了一个随机数X_a作为Diffiee-Hellman的指数，B选择了另一个随机数X_b。
- A和B就可以通过下面的过程进行Diffie-Hellman密钥交换，并得到共享密钥$g^{XaXb}(\text{mod } p)$。

• ①：A→B：$g^{Xa}(\text{mod } p)$

• ②：B→A：$g^{Xb}(\text{mod } p)$

- 在不支持Diffie-Hellman密钥交换的SKEME模式中，交换阶段用来交换随机产生的即时时间值。

•(4)协议的执行过程中，双方的验证在认证阶段完成，这个阶段会用到共享阶段产生的密钥K_{ab}来验证从交换阶段得到的Diffie-Hellman指数或是即时时间值。如果交换阶段执行了Diffie—Hellman密钥交换，认证阶段可以表示如下。

• ①：A→B：<g^{Xa}，g^{Xb}，id_A，id_B>K_{ab}

• ②：B→A：<g^{Xb}，g^{Xa}，id_B，id_A>K_{ab}

•如果没有执行Diffie-Hellman密钥交换，Diffie-Hellman指数就被即时时间值所代替。

•SKEME支持四种模式，即：基本模式、只有共享模式、预先共享密钥模式和快速刷新模式。

•基本模式

- 包括了前面介绍的全部四个阶段，因此，基本模式下的SKEME支持完美向前保密(Perfect Forward Secrecy，PFS)。
- PFS指的是保证将来的通信受到密码学加密的保护，即使是长期使用的加密密钥被别人发现和破译，后来即时交换产生密钥，它的保密性也不会遭到危害，同时传输数据的保密性也不会被破坏。

•只有共享模式支持公开密钥加密法，但不支持Diffie-Hellman密钥交换技术。因此，这种模式下的SKEME不支持完美向前保密。

•SKEME支持四种模式，即：基本模式、只有共享模式、预先共享密钥模式和快速刷新模式。

•预先共享密钥模式中，SKEME没有使用公开密钥加密法。只进行了Diffie-Hellman密钥交换。在这种模式中，协议认为通信双方已经有了共享密钥，它们用这个共享密钥来产生后来新的会话密钥和进行密钥刷新。前面曾经提到，这个共享密钥是带外传送的，可以通过人工安装，也可以从KDC获得。

•快速刷新模式是最快速的SKEME协议。它既不使用公开密钥加密法，也不使用Diffie-Hellman密钥交换。在这个模式中，共享阶段完全被省略，交换阶段和验证阶段也只是交换即时时间。

IPSec 协议 —9 密钥管理和密钥交换

- 4 Oakley协议

•Oakley协议具有以下5个特性：

•①它采用称为cookie的程序机制来防止阻塞攻击。

•②它可使通信双方协商Diffie-Hellman密钥交换所用的群，即协商所需的全局参数。

•③使用一次性随机数防止重放攻击。

•④可使通信双方交换Diffie-Hellman公开密钥的值。

•⑤对Diffie-Hellman交换过程加以认证以防止中间人攻击。

•以Oakley为基础，IKE借鉴了不同模式的思想，在每种模式中，Oakley会进行不同的通过验证的密钥交换。

IPSec 协议 —9 密钥管理和密钥交换

- 4 Internet安全联盟和密钥管理协议（ISAKMP）
- IKE的报文格式就是建立在ISAKMP的框架下的。
- ISAKMP
  - 定义了双方如何沟通
  - 如何构建双方用以沟通的消息
  - 保障通信安全所需的状态交换
  - 提供了对对方的身份进行验证的方法
  - 密钥交换时交换信息的方法
  - 以及对安全服务进行协商的方法 (1)消息和载荷

- *4 Internet安全联盟和密钥管理协议（ISAKMP）*
- (1)消息和载荷
- 对一个用基于ISAKMP的密钥管理协议交换的消息来说，它的构建方法是：将ISAKMP所有载荷链接到一个ISAKMP头。ISAKMP的结构如图所示：

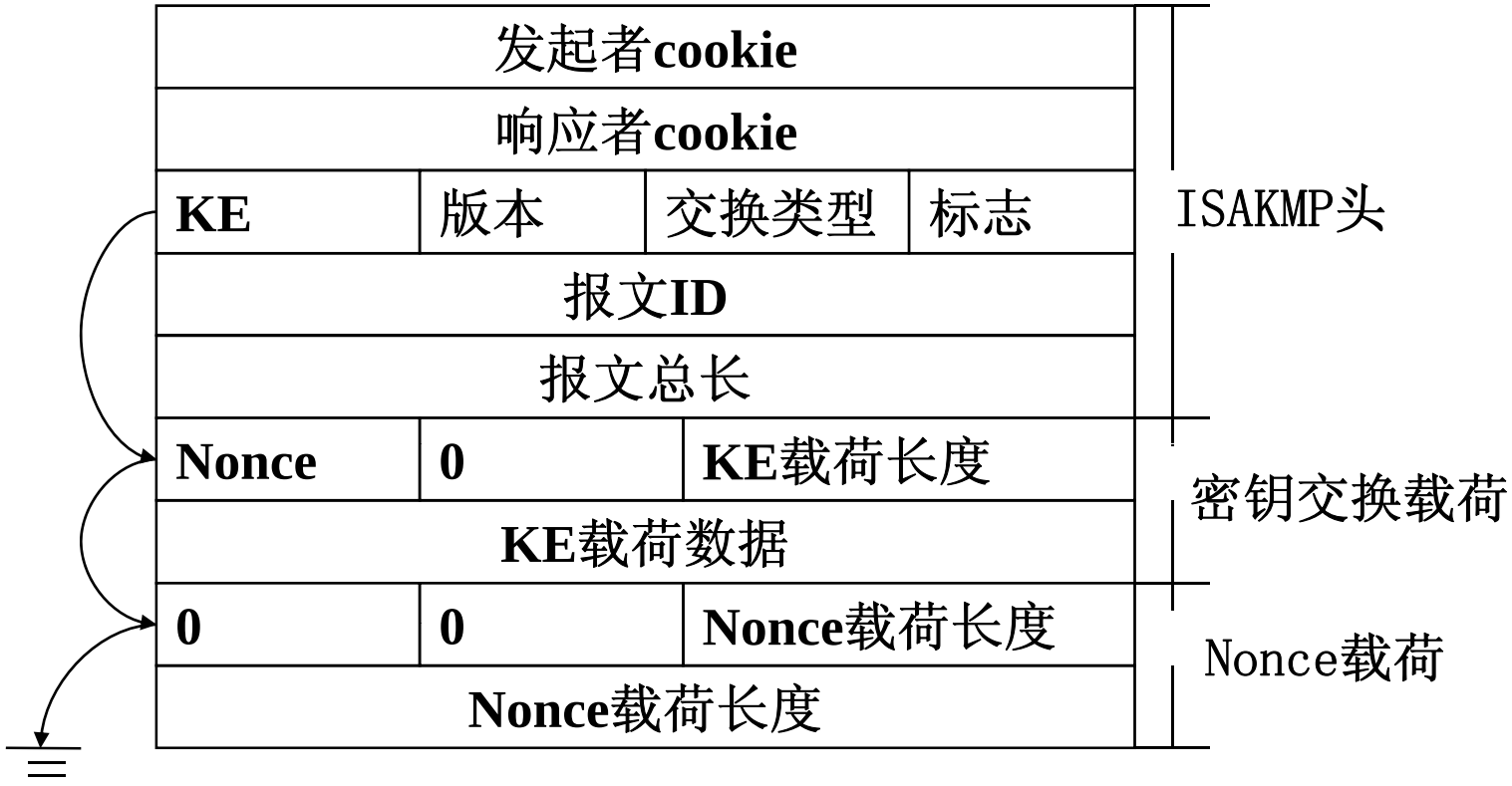
<table>
<tr><td colspan="5">发起者的cookie</td></tr>
<tr><td colspan="5">响应者的cookie</td></tr>
<tr><td>下个有效载荷</td><td>主要版本</td><td>次要版本</td><td>交换类型</td><td>标志</td></tr>
<tr><td colspan="5">报文ID</td></tr>
<tr><td colspan="5">报文长度</td></tr>
</table>

- 目前在**ISAKMP**中，总共定义了**13**种不同的载荷。它们都是以相同格式的头开始的。

下个有效载荷	保留	有效载荷长度

- 载荷标识符**p156**

- 在一条消息中，载荷之间链接到一起，这是用通用头中的“下个有效载荷”字段来实现的，如图所示。**ISAKMP**头指出了紧接在头之后的第一个载荷，同时每个载荷也指出了接下去的是哪个载荷。消息中的最后一个载荷为**0**。**ISAKMP**通信是通过**UDP**端口**500**来进行的。



- *4 Internet安全联盟和密钥管理协议（ISAKMP）*
- (2)交换阶段
- ISAKMP描述了协商的两个阶段。在第一阶段，通信各方彼此间建立了一个通过身份验证和安全保护的通道。在第二阶段，这个通道用于为另一个不同的协议（比如IPSec）协商安全服务。
- 阶段1的交换建立了一个ISAKMP“安全关联（SA）”。这个SA实际上是安全策略的一个抽象和一个密钥。
- 阶段2交换可为其他协议建立安全关联。
  - 由于ISAKMP SA已经验证通过了，所以可用它为一次阶段2交换中的所有消息提供源验证、完整性以及机密性保障。
  - 完成了一次阶段2交换之后，在ISAKMP处理过程中和它关联在一起的状态(例如密钥，验证方法等)便不复存在。但是ISAKMP SA可继续存在下去，以确保后续的阶段2交换的安全。

- *4 Internet安全联盟和密钥管理协议（ISAKMP）*
- (3)ISAKMP定义的五种密钥交换
- ① 基本交换(Base Exchange)
- 允许密钥交换和身份验证有关信息放在同一条信息中传输，这样就减少了协商的消息数，有利于快速建立SA。
- 但由于密钥交换和身份验证有关信息在同一条信息中传输，所以该交换不能提供身份保护。因为只有该消息发出去之后，双方才能获得共享的密钥。

- *4 Internet安全联盟和密钥管理协议（ISAKMP）*
- (3)ISAKMP定义的五种密钥交换
- 基本交换(Base Exchange)
- 以下是交换的具体消息：
- i：I→R：SA；NONCE　　开始ISAKMP-SA协商
- ii：R→I：SA；NONCE　　基本SA同意生成的密钥
- iii：I→R：KE；IDI；AUTH　密钥生成了；发起者的身份被响应者验证
- iv：R→I：KE；IDR；AUTH　　响应者的身份被发起者验证；密钥生成了；SA建立了

4 Internet安全联盟和密钥管理协议（ISAKMP）

- (3)ISAKMP定义的五种密钥交换
- 身份保护交换(Identity Protection Exchange)
- 将密钥交换与身份和鉴别信息分开，以提供对通信双方身份的保护。在这种交换中，对话各方先协商了一个共享密钥，然后用该密钥对稍后传输的身份和鉴别信息加密，以提供对身份的保护。但它是以多两条消息的代价实现的。

• i： I→R： SA	开始ISAMP-SA协商
• ii： R→I： SA	基本SA同意生成的密钥
• iii： I→R： KE； NONCE	密钥生成了
• iv： R→I： KE； NONCE	密钥生成了
• v： *I→R： ID_I； AUTH	发起者的身份被响应者验证
• vi： *R→I： ID_R； AUTH	响应者的身份被发起者验证；SA建立了

4 Internet安全联盟和密钥管理协议（ISAKMP）

- (3)ISAKMP定义的五种密钥交换
- 仅仅鉴别交换

•这种交换目的并不在交换生成一个共享的密钥，它只需要传输与鉴别有关的信息，最终形成安全关联，并确认对方正是自己要交流的人。这种交换还有一个特点是消息全部不加密，而且，由于目的只是验证，因此消息数只有三条。

• i ： I→R： SA； NONCE　　开始ISAKMP-SA协商

• ii ： R→I： SA； NONCE； ID_R； AUTH　基本SA同意建立；响应者的身份被发起者验证

•iii： I→R： ID_I； AUTH　　发起者的身份被响应者验证；SA建立了

4 Internet安全联盟和密钥管理协议（ISAKMP）

- (3)ISAKMP定义的五种密钥交换
- 野蛮交换(Aggressive Exchange)

•野蛮交换允许一方将安全联盟载荷、密钥交换载荷与验证相关的信息载荷放在一条消息中传输，这样做的后果与基本交换一样，都不能提供身份保护功能。这种交换方式企图在一条消息内建立所有与安全相关的信息。

• i ： I→R：SA；KE；NONCE；IDI　开始ISAKMP-SA协商和密钥交换

• ii ： R→I：SA；KE；NONCE；IDR；AUTH　发起者的身份被响应者验证；密钥生成了；基本SA同意建立

•iii：I→R：AUTH　　响应者的身份被发起者验证；SA建立了

4 Internet安全联盟和密钥管理协议（ISAKMP）

- (3)ISAKMP定义的五种密钥交换
- 信息交换(Informational Exchange)
- 信息交换比较特殊，它一般用于向对方提供特殊的信息，该信息是单向的，接收者不必做应答。
- i：*I→R：N/D　　差错或状态通知或删除
- 交换的载荷只有两种类型：通知载荷和删除载荷。如果信息交换在密钥交换之前，则不能对信息交换进行保护，否则要进行加密传输。

IPSec 协议 —9 密钥管理和密钥交换

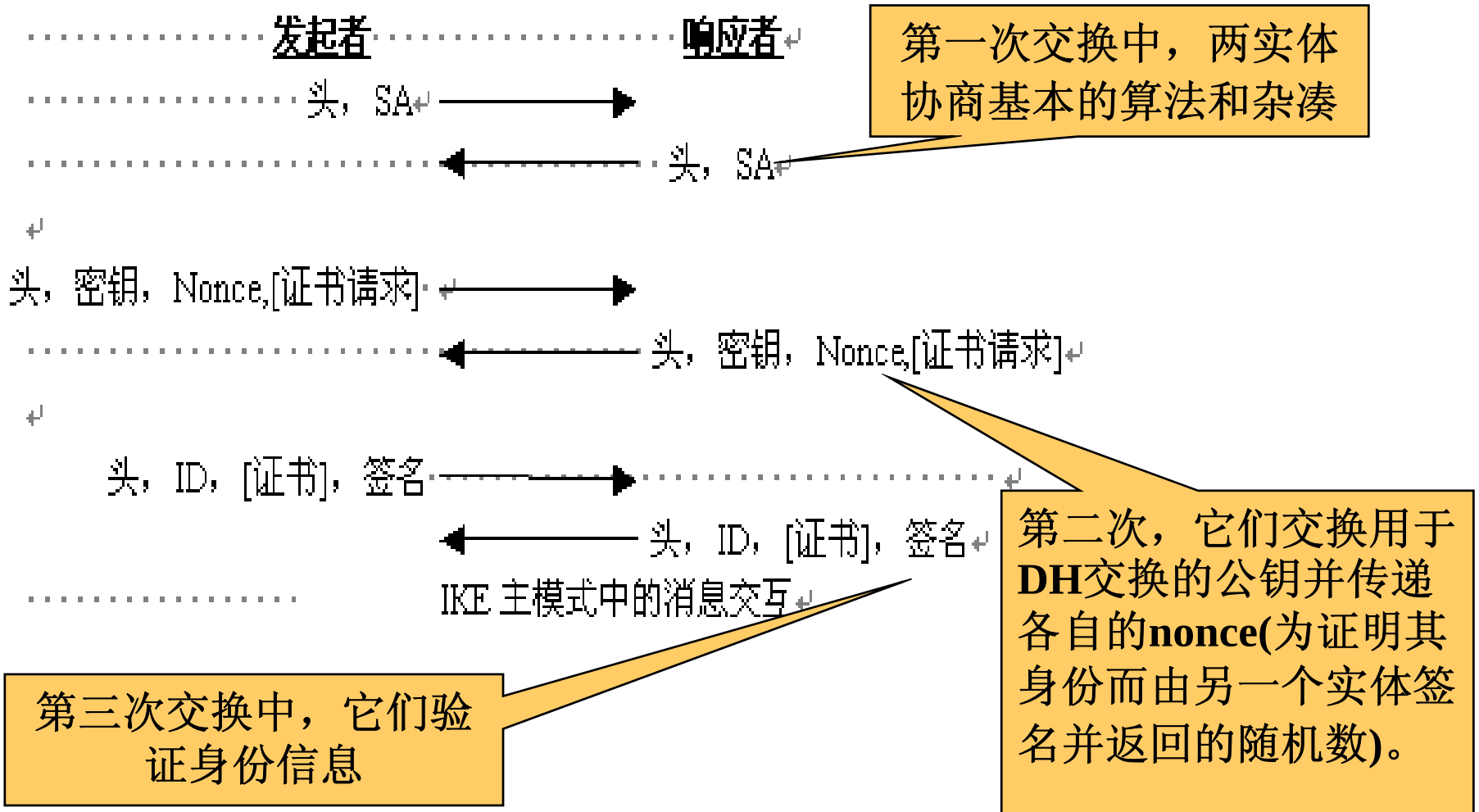
- 5Internet密钥交换协议(Internet Key Exchange，IKE)

- IKE是IPSec目前正式确定的密钥交换协议。IKE为IPSec的AH和ESP协议提供密钥交换管理和安全关联管理，同时也为ISAKMP提供密钥管理和安全管理。
- IKE定义了通信实体间进行身份认证、创建安全关联、协商加密算法以及生成共享会话密钥的方法。
- IKE用两个阶段来实现它的功能。在第一个阶段中，两个IKE对等方使用一个普通的IKE安全关联为通信建立一个安全的、经过认证的隧道。IKE提供了三种交换密钥信息和建立SA的模式；在第一阶段中，仅仅使用主模式和积极模式。
- 在第二个阶段中，将为像IPSec这样的服务或者是任何其他需要密钥信息和参数协商的服务协商SA。第二个阶段通过一个快速模式交换来完成。

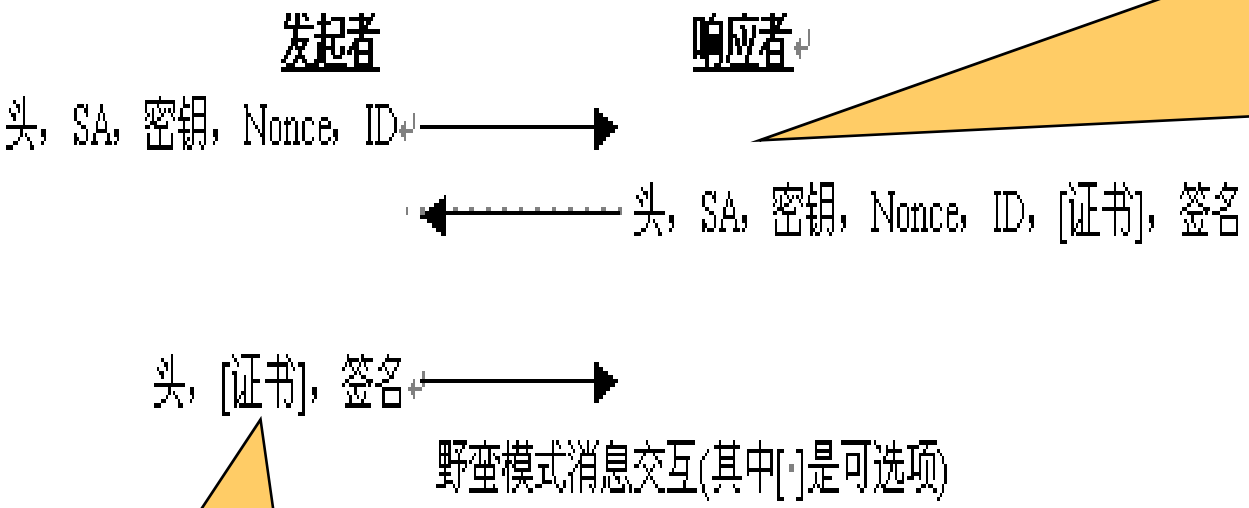
IPSec 协议 —9 密钥管理和密钥交换

- 5Internet密钥交换协议
- (1)主模式
  - IKE的主模式(main mode)为建立第一阶段的IKE SA提供了一个三阶段机制，它用于协商以后的通信。
  - 实体间要协商足够多的东西(诸如认证和加密算法、杂凑和密钥)。
  - 在该模式中，要在SA发起者与接受者之间交换三个双向消息。

- *5Internet密钥交换协议*
- (1) 主模式



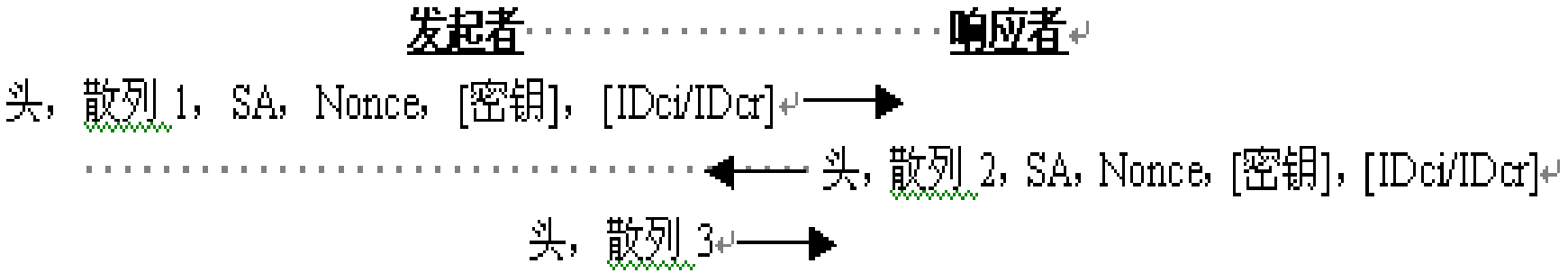
- *5 Internet密钥交换协议*
- (2)积极模式
  - 因为积极模式也是用于建立初始的IKE SA的。但是，在消息构建的方式上不同，从而将交换的数目从3减为2



1 发起者在交换的开始生成一个DH对；
2 建立一个SA，
3 传递DH公开值，
4 发送一个用于给对方签名的nonce，
5 并发送一个ID数据包；
6 然后，响应者发送回为了完成本次交换而需要的一切。

发起者剩下要做的一切就是确认本次交换

- *5 Internet密钥交换协议*
- (3)快速模式
  - 在两个通信实体使用主模式或者积极模式建立了一个IKE SA之后，它们就可以使用快速模式了。
  - 快速模式仅仅用于协商通用的IPSec安全服务并生成新的密钥信息。
  - 快速模式数据包都是加密的，并且以一个散列载荷开始，该散列载荷是由一致同意的伪随机函数和为IKESA而衍生出的认证密钥构成的。散列载荷用于认证数据包的其余部分。快速模式定义了数据包的哪一部分将包括在散列中。

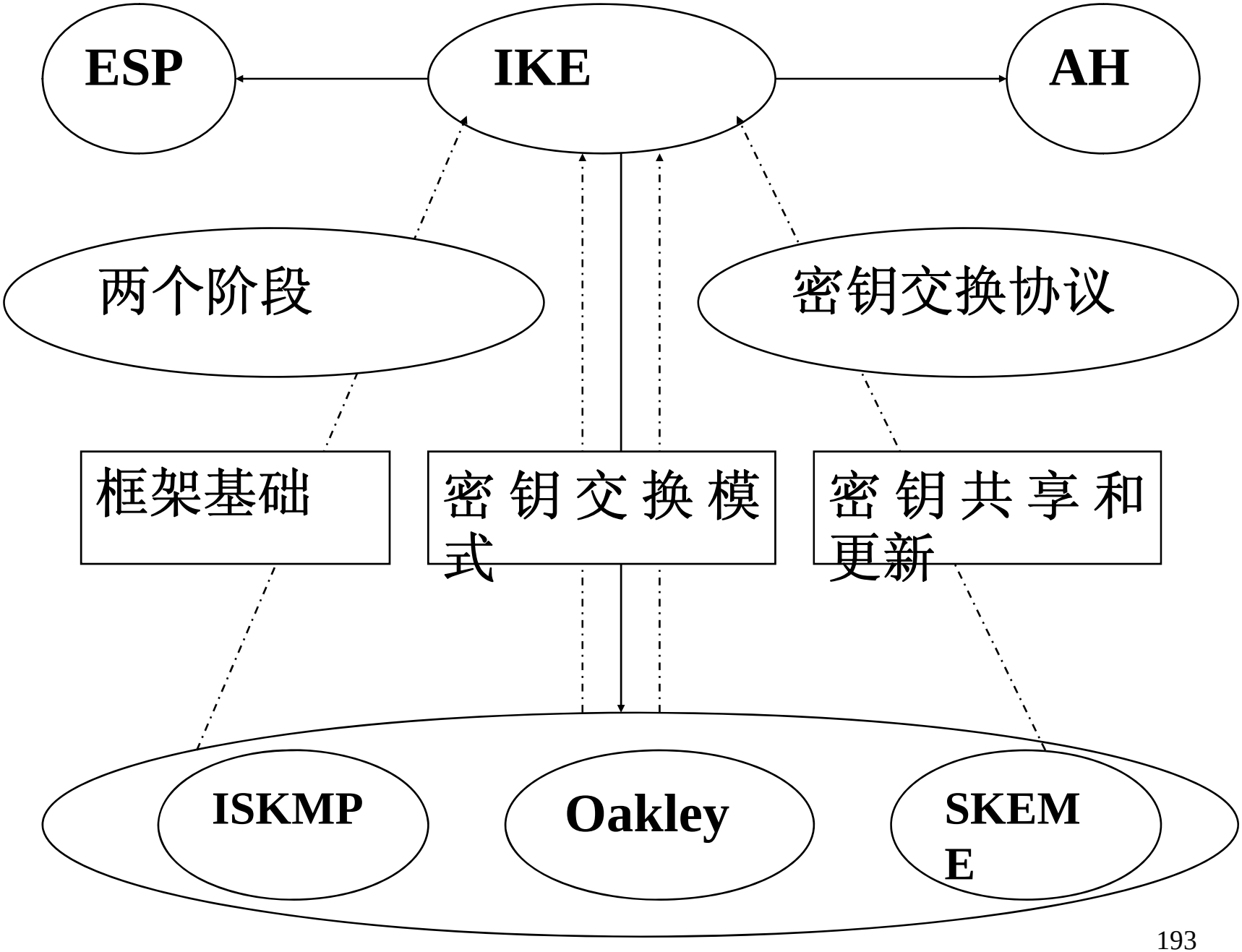


小结

- 因特网与TCP/IP安全
- SSL协议
  - 用处，所处位置
  - 协议栈
  - 握手协议，消息描述
- SET协议
  - 应用背景
  - 双向握手及应用（购买请求过程的描述）
- IPSec协议

小结

- 因特网与TCP/IP安全
- SSL协议
- SET协议
- IPSec协议
  - 通过3个要素提供认证、完整性和保密服务
    - 验证头（AH）
    - 封装安全载荷（ESP）
    - 互联网密钥管理协议（IKMP）
  - 工作模式：传输和隧道模式
  - 安全关联


ESP
IKE
AH
两个阶段
密钥交换协议
框架基础
密钥交换模式
密钥共享和更新
ISKMP
Oakley
SKEME