**1. OSI与TCP/IP各层的结构与功能，都有哪些协议。**

答案：

（1）OSI模型

OSI（Open System Interconnection，开放系统互连）七层网络模型称为开放式系统互联参考模型 ，是一个逻辑上的定义，一个规范，它把网络从逻辑上分为了7层。每一层都有相关、相对应的物理设备，比如路由器，交换机。OSI 七层模型是一种框架性的设计方法 ，建立七层模型的主要目的是为解决异种网络互连时所遇到的兼容性问题，其最主要的功能使就是帮助不同类型的主机实现数据传输。它的最大优点是将服务、接口和协议这三个概念明确地区分开来，通过七个层次化的结构模型使不同的系统不同的网络之间实现可靠的通讯。



**模型优点**

建立七层模型的主要目的是为解决异种网络互连时所遇到的兼容性问题。它的最大优点是将服务、接口和协议这三个概念明确地区分开来：服务说明某一层为上一层提供一些什么功能，接口说明上一层如何使用下层的服务，而协议涉及如何实现本层的服务；这样各层之间具有很强的独立性，互连网络中各实体采用什么样的协议是没有限制的，只要向上提供相同的服务并且不改变相邻层的接口就可以了。网络七层的划分也是为了使网络的不同功能模块（不同层次）分担起不同的职责，从而带来如下好处：

● 减轻问题的复杂程度，一旦网络发生故障，可迅速定位故障所处层次，便于查找和纠错；

● 在各层分别定义标准接口，使具备相同对等层的不同网络设备能实现互操作，各层之间则相对独立，一种高层协议可放在多种低层协议上运行；

● 能有效刺激网络技术革新，因为每次更新都可以在小范围内进行，不需对整个网络动大手术；

● 便于研究和教学。

**分层介绍**

1.物理层：主要定义物理设备标准，如网线的接口类型、光纤的接口类型、各种传输介质的传输速率等。它的主要作用是传输比特流（就是由1、0转化为电流强弱来进行传输,到达目的地后在转化为1、0，也就是我们常说的数模转换与模数转换）。这一层的数据叫做比特。

2.数据链路层：定义了如何让格式化数据以进行传输，以及如何让控制对物理介质的访问。这一层通常还提供错误检测和纠正，以确保数据的可靠传输。

3.网络层：在位于不同地理位置的网络中的两个主机系统之间提供连接和路径选择。Internet的发展使得从世界各站点访问信息的用户数大大增加，而网络层正是管理这种连接的层。

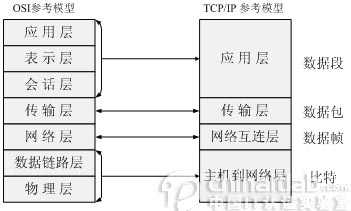
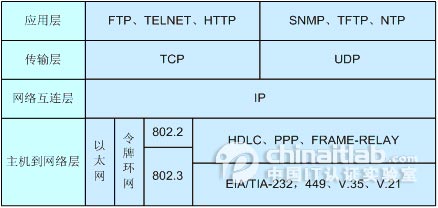
4.传输层：定义了一些传输数据的协议和端口号（WWW端口80等），如：TCP（传输控制协议，传输效率低，可靠性强，用于传输可靠性要求高，数据量大的数据），UDP（[用户数据报协议](http://baike.baidu.com/view/468464.htm)，与TCP特性恰恰相反，用于传输可靠性要求不高，数据量小的数据，如QQ聊天数据就是通过这种方式传输的）。 主要是将从下层接收的数据进行分段和传输，到达目的地址后再进行重组。常常把这一层数据叫做段。

5.会话层：通过传输层（端口号：传输端口与接收端口）建立数据传输的通路。主要在你的系统之间发起会话或者接受会话请求（设备之间需要互相认识可以是IP也可以是MAC或者是主机名）

6.表示层：可确保一个系统的应用层所发送的信息可以被另一个系统的应用层读取。例如，PC程序与另一台计算机进行通信，其中一台计算机使用扩展二一十进制交换码（EBCDIC），而另一台则使用美国信息交换标准码（ASCII）来表示相同的字符。如有必要，表示层会通过使用一种通格式来实现多种数据格式之间的转换。

7.应用层： 是最靠近用户的OSI层。这一层为用户的应用程序（例如电子邮件、文件传输和终端仿真）提供网络服务。

（2）TCP/IP模型

ISO制定的OSI参考模型的过于庞大、复杂招致了许多批评。与此对照，由技术人员自己开发的TCP/IP协议栈获得了更为广泛的应用。如图2-1所示，是TCP/IP参考模型和OSI参考模型的对比示意图。  
  
　　TCP/IP协议栈是美国国防部高级研究计划局计算机网（Advanced Research Projects Agency Network，ARPANET）和其后继因特网使用的参考模型。ARPANET是由美国国防部（U.S．Department of Defense，DoD）赞助的研究网络。最初，它只连接了美国境内的四所大学。随后的几年中，它通过租用的电话线连接了数百所大学和政府部门。最终ARPANET发展成为全球规模最大的互连网络-因特网。最初的ARPANET于1990年永久性地关闭。　　  
　　TCP/IP参考模型分为四个层次：应用层、传输层、网络互连层和主机到网络层。如图2-2所示。  
  
图2-2　 TCP/IP参考模型的层次结构  
  
　　在TCP/IP参考模型中，去掉了OSI参考模型中的会话层和表示层（这两层的功能被合并到应用层实现）。同时将OSI参考模型中的数据链路层和物理层合并为主机到网络层。下面，分别介绍各层的主要功能。  
　　  
　　1、主机到网络层　　  
　　**实际上TCP/IP参考模型没有真正描述这一层的实现，**只是要求能够提供给其上层-网络互连层一个访问接口，以便在其上传递IP分组。由于这一层次未被定义，所以其具体的实现方法将随着网络类型的不同而不同。　　  
　　2、网络互连层　　  
　　**网络互连层是整个TCP/IP协议栈的核心。它的功能是把分组发往目标网络或主机。**同时，为了尽快地发送分组，可能需要沿不同的路径同时进行分组传递。因此，分组到达的顺序和发送的顺序可能不同，这就需要上层必须对分组进行排序。　　  
　　网络互连层定义了分组格式和协议，即IP协议（Internet Protocol）。　　  
　　网络互连层除了需要完成路由的功能外，也可以完成将不同类型的网络（异构网）互连的任务。除此之外，网络互连层还需要完成拥塞控制的功能。　　  
　　3、传输层　　  
　　**在TCP/IP模型中，传输层的功能是使源端主机和目标端主机上的对等实体可以进行会话。**在传输层定义了两种服务质量不同的协议。即：传输控制协议TCP（transmission control protocol）和用户数据报协议UDP（user datagram protocol）。　　  
　　TCP协议是一个面向连接的、可靠的协议。它将一台主机发出的字节流无差错地发往互联网上的其他主机。在发送端，它负责把上层传送下来的字节流分成报文段并传递给下层。在接收端，它负责把收到的报文进行重组后递交给上层。TCP协议还要处理端到端的流量控制，以避免缓慢接收的接收方没有足够的缓冲区接收发送方发送的大量数据。　　  
　　UDP协议是一个不可靠的、无连接协议，主要适用于不需要对报文进行排序和流量控制的场合。　　  
　　4、应用层　　  
　　**TCP/IP模型将OSI参考模型中的会话层和表示层的功能合并到应用层实现。**　　  
　　应用层面向不同的网络应用引入了不同的应用层协议。其中，有基于TCP协议的，如文件传输协议（File Transfer Protocol，FTP）、虚拟终端协议（TELNET）、超文本链接协议（Hyper Text Transfer Protocol，HTTP），也有基于UDP协议的。

**2. TCP与UDP的区别。**

答案：

TCP和UDP是OSI模型中的运输层中的协议。TCP提供可靠的通信传输，而UDP则常被用于让广播和细节控制交给应用的通信传输。

**UDP(User Datagram Protocol)**

**UDP不提供复杂的控制机制，利用IP提供面向无连接的通信服务。**并且它是将应用程序发来的数据在收到的那一刻，立刻按照原样发送到网络上的一种机制。

即使是出现网络拥堵的情况下，UDP也无法进行流量控制等避免网络拥塞的行为。此外，传输途中如果出现了丢包，UDP也不负责重发。甚至当出现包的到达顺序乱掉时也没有纠正的功能。如果需要这些细节控制，那么不得不交给由采用UDP的应用程序去处理。换句话说，UDP将部分控制转移到应用程序去处理，自己却只提供作为传输层协议的最基本功能。UDP有点类似于用户说什么听什么的机制，但是需要用户充分考虑好上层协议类型并制作相应的应用程序。

**TCP(Transmission Control Protocol)**

**TCP充分实现了数据传输时各种控制功能**，可以进行丢包的重发控制，还可以对次序乱掉的分包进行顺序控制。而这些在UDP中都没有。此外，**TCP作为一种面向有连接的协议，只有在确认通信对端存在时才会发送数据，从而可以控制通信流量的浪费。**TCP通过检验和、序列号、确认应答、重发控制、连接管理以及窗口控制等机制实现可靠性传输。

**TCP与UDP如何加以区分使用？**

TCP用于在传输层有必要实现可靠性传输的情况。由于它是面向有连接并具备顺序控制、重发控制等机制的。所以它可以为应用提供**可靠传输**。

另一方面，UDP主要用于那些对**高速传输和实时性**有较高要求的通信或广播通信。举一个IP电话进行通话的例子。如果使用TCP，数据在传送途中如果丢失会被重发，但是这样无法流畅地传输通话人的声音，会导致无法进行正常交流。而采用UDP，**它不会进行重发处理。从而也就不会有声音大幅度延迟到达的问题。即使有部分数据丢失，也只是影响某一小部分的通话。**此外，在多播与广播通信中也使用UDP而不是UDP。RIP、DHCP等基于广播的协议也要依赖于UDP。

**TCP与UDP区别总结：**

1、TCP**面向连接**（如打电话要先拨号建立连接）;UDP是**无连接的**，即发送数据之前不需要建立连接

2、TCP**提供可靠**的服务。也就是说，通过TCP连接传送的数据，无差错，不丢失，不重复，且按序到达;UDP尽最大努力交付，即**不保证可靠交付**

3、TCP面向字节流，实际上是TCP把数据看成一连串无结构的字节流;UDP是面向报文的

UDP没有拥塞控制，因此网络出现拥塞不会使源主机的发送速率降低（对实时应用很有用，如IP电话，实时视频会议等），并且UDP速度更快

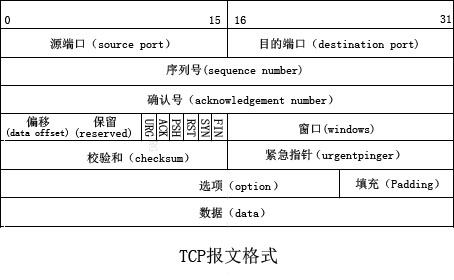
4、**每一条TCP连接只能是点到点的;UDP支持一对一，一对多，多对一和多对多的交互通信**

5、TCP首部开销20字节;UDP的首部开销小，只有8个字节

6、TCP的逻辑通信信道是全双工的可靠信道，UDP则是不可靠信道

**3. TCP报文结构。**

答案：



l 源端口（Source Port）：16位的源端口字段包含初始化通信的端口号。源端口和IP地址的作用是标识报文的返回地址。

l 目的端口(Destination Port):16位的目的端口字段定义传输的目的。这个端口指明接收方计算机上的应用程序接口。

l 序列号（Sequence Number，seq）:该字段用来标识TCP源端设备向目的端设备发送的字节流，它表示在这个报文段中的第几个数据字节。序列号是一个32位的数。

l 确认号（Acknowledge Number，ack）：**TCP使用32位的确认号字段标识期望收到的下一个段的第一个字节，并声明此前的所有数据已经正确无误地收到，因此，确认号应该是上次已成功收到的数据字节序列号加1！！！！。**收到确认号的源计算机会知道特定的段已经被收到。确认号的字段只在ACK标志被设置时才有效。

l 数据偏移（Data Offset）:这个4位字段包括TCP头大小。由于首部可能含有选项内容，因此TCP首部的长度是不确定的。首部长度的单位是32比特或4个八位组。首部长度实际上也指示了数据区在报文段中的起始偏移值。

l 保留（Reserved）:6位置0的字段。为将来定义新的用途保留。、

l 控制位(Control Bits):共6位，每一位标志可以打开一个控制功能。

URG(Urgent Pointer Field Significant,紧急指针字段标志):表示TCP包的紧急指针字段有效，用来保证TCP连接不被中断，并且督促中间齐备尽快处理这些数据。

ACK（Acknowledgement field significant,确认字段标志）: 取1时表示应答字段有效，也即TCP应答号将包含在TCP段中，为0则反之。

PSH(Push Function,推功能)：这个标志表示Push操作。所谓Push操作就是指在数据包到达接收端以后，立即送给应用程序，而不是在缓冲区中排队。

RST（Reset the connection,重置连接）：这个标志表示感谢连接复位请求，用来复位那些产生错误的连接，也被用来拒绝错误和非法的数据包。

**SYN（Synchronize sequence numbers,同步序列号）:表示同步序号，用来建立连接。**

**FIN（No more data from sender）:表示发送端已经发送到数据末尾，数据传送完成，发送FIN标志位的TCP段，连接将被断开。**

l **接受窗口（Window）:目的主机使用16位的窗口字段告诉源主机它期望每次收到的字节数。**

l 校验和（Checksum）:TCP头包括16位的校验和字段用于错误检查。源主机基于部分IP头信息，TCP头和数据内容计算一个校验和，目的主机也要进行相同的计算，如果收到的内容没有错误过，两个计算应该完全一样，从而证明数据的有效性。

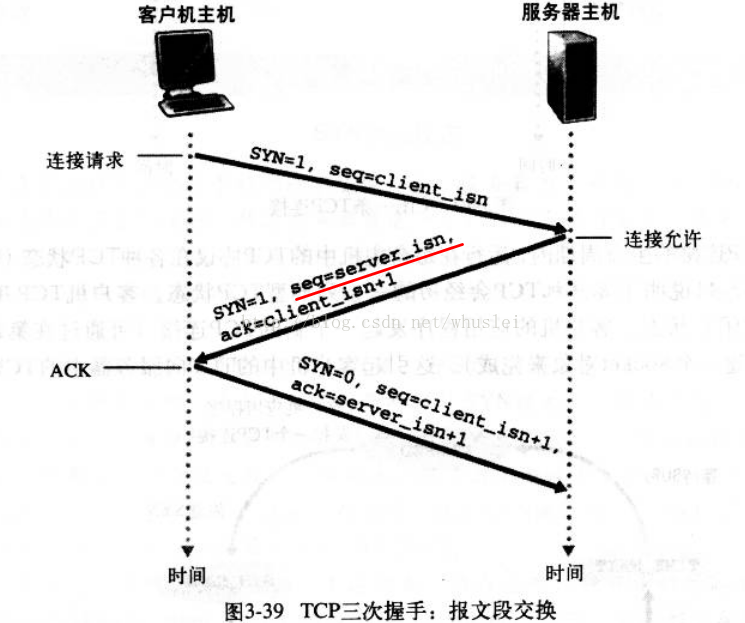
l 紧急指针（Urgent Pointer）:紧急指针字段是一个可选的16位指针，指向段内的最后一个字节位置，这个字段只在URG标志被设置时才有效。

l 选项（Option）:至少1字节的可变长字段，标识哪个选项（如果有的话）有效。如果没有选项，这个字节等于0,说明选项的结束。这个字节等于1表示无需再有操作；等于2表示下四个字节包括源机器的最大长度（Maximum Segment Size,MSS）.

l 填充（Padding）:这个字段中加入额外的零，以保证TCP头是32的整数倍。

4. TCP的三次握手与四次挥手过程，各个状态名称与含义，TIMEWAIT的作用。

（1）TCP的三次握手——**建立TCP连接**



**第一步：**

客户端TCP向服务端TCP发送一个特殊的TCP报文段，不包含应用层数据，但在报文段首部的SYN标志位被设为1，表示建立连接的报文段，因此被称为SYN报文段。另外，客户端会选择一个初始序号client\_isn，**记录在此报文段的序列号seq**中。该报文段会封装在一个IP数据报中被发送到服务器端。这个报文段表达的就是希望建立的信息。

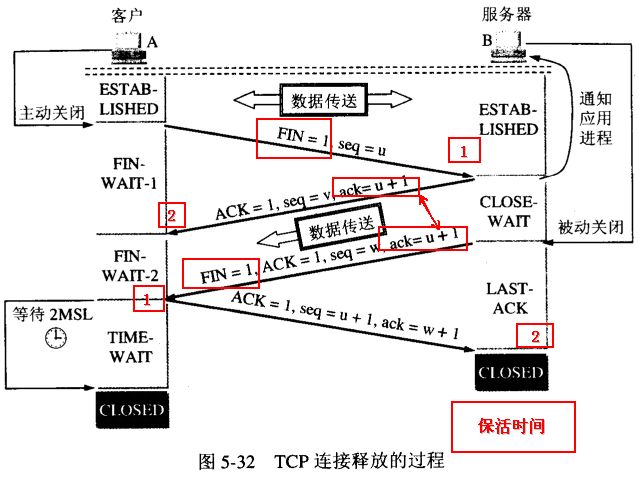
**第二步：**

一旦包含SYN报文段的IP数据报到达服务器主机，服务器从IP数据报中提取出TCP SYN报文段，为该TCP连接分配需要的缓存和变量，并向客户端发送表示允许连接的报文段。这个允许连接的报文段也不包含任何应用层数据，但包含三个重要的信息：首先，SYN被置为1，其次，**该报文段的确认号ack被置为client\_isn+1**，表示下一个希望接受的序列号为**client\_isn+1**，最后，服务器选择自己的初始序号server\_isn，**记录在序列号seq中**。这个报文表达的就是允许建立该连接，我自己的初始序号是server\_isn。**有时也被成为SYNACK报文段。**

**第三步：**

在收到SYNACK报文段之后，客户端也要给该连接分配缓存和变量，客户端向服务器再发送一个报文段，对允许连接的报文段进行了确认，其ack被置为server\_isn+1，表示下一个希望从服务器获得的报文段的序列号为此。**并且由于连接已经建立了，SYN标志位被置为0，并且已经可以携带被传送到服务器的应用层数据。**

（2）TCP的四次挥手——**断开TCP连接**



**简单过程：**

当客户A 没有东西要发送时就要释放 A 这边的连接，A会发送一个报文（没有数据），其中 FIN 设置为1,  服务器B收到后会给应用程序一个信，这时A那边的连接已经关闭，即A不再发送信息（但仍可接收信息）。  A收到B的确认后进入等待状态，等待B请求释放连接， B数据发送完成后就向A请求连接释放，也是用FIN=1 表示， 并且用 ack = u+1(如图）， A收到后回复一个确认信息，并进入 TIME\_WAIT 状态， 等待 2MSL 时间。

为什么要等待呢？

为了这种情况： B向A发送 FIN = 1 的释放连接请求，但这个报文丢失了， A没有接到不会发送确认信息， B 超时会重传，这时A在 WAIT\_TIME 还能够接收到这个请求，这时再回复一个确认就行了。（A收到 FIN = 1 的请求后 WAIT\_TIME会重新记时）

**深入理解：**

由于TCP连接是全双工的，因此每个方向都必须单独进行关闭。这原则是当一方完成它的数据发送任务后就能发送一个FIN来终止这个方向的连接。收到一个 FIN只意味着这一方向上没有数据流动，一个TCP连接在收到一个FIN后仍能发送数据。首先进行关闭的一方将执行主动关闭，而另一方执行被动关闭。  
TCP协议的连接是全双工连接，一个TCP连接存在双向的读写通道。   
简单说来是 “先关读，后关写”，一共需要四个阶段。以客户机发起关闭连接为例：  
1.服务器读通道关闭  
2.客户机写通道关闭  
3.客户机读通道关闭  
4.服务器写通道关闭  
关闭行为是在发起方数据发送完毕之后，给对方发出一个FIN（finish）数据段。直到接收到对方发送的FIN，且对方收到了接收确认ACK之后，双方的数据通信完全结束，过程中每次接收都需要返回确认数据段ACK。  
详细过程：  
    第一阶段   客户机发送完数据之后，向服务器发送一个FIN数据段，序列号为i；  
    1.服务器收到FIN(i)后，返回确认段ACK，序列号为i+1，关闭服务器读通道；  
    2.客户机收到ACK(i+1)后，关闭客户机写通道；  
   （此时，客户机仍能通过读通道读取服务器的数据，服务器仍能通过写通道写数据）  
    第二阶段 服务器发送完数据之后，向客户机发送一个FIN数据段，序列号为j；  
    3.客户机收到FIN(j)后，返回确认段ACK，序列号为j+1，**在TIME\_WAIT后才能关闭客户机读通道！！！！！！！！！！；**  
    4.服务器收到ACK(j+1)后，关闭服务器写通道。  
这是标准的TCP关闭两个阶段，服务器和客户机都可以发起关闭，完全对称。

从上面可以看到，主动发起关闭连接的操作的一方将达到TIME\_WAIT状态，而且这个状态要保持Maximum Segment Lifetime的两倍时间。为什么要这样做而不是直接进入CLOSED状态？

原因有二：  
**一、保证TCP协议的全双工连接能够可靠关闭  
二、保证这次连接的重复数据段从网络中消失**

**先说第一点**，如果Client直接CLOSED了，那么由于IP协议的不可靠性或者是其它网络原因，导致Server没有收到Client最后回复的ACK。那么Server就会在超时之后继续发送FIN，此时由于Client已经CLOSED了，就找不到与重发的FIN对应的连接，最后Server就会收到RST而不是ACK，Server就会以为是连接错误把问题报告给高层。这样的情况虽然不会造成数据丢失，但是却导致TCP协议不符合可靠连接的要求。所以，Client不是直接进入CLOSED，而是要保持TIME\_WAIT，当再次收到FIN的时候，能够保证对方收到ACK，最后正确的关闭连接。

**再说第二点**，如果Client直接CLOSED，然后又再向Server发起一个新连接，我们不能保证这个新连接与刚关闭的连接的端口号是不同的。也就是说有可能新连接和老连接的端口号是相同的。一般来说不会发生什么问题，但是还是有特殊情况出现：假设新连接和已经关闭的老连接端口号是一样的，如果前一次连接的某些数据仍然滞留在网络中，这些延迟数据在建立新连接之后才到达Server，由于新连接和老连接的端口号是一样的，又因为TCP协议判断不同连接的依据是socket pair，于是，TCP协议就认为那个延迟的数据是属于新连接的，这样就和真正的新连接的数据包发生混淆了。所以TCP连接还要在TIME\_WAIT状态等待2倍MSL，这样可以保证本次连接的所有数据都从网络中消失。

**5. TCP拥塞控制。**

（1）拥塞控制和流量控制的区别

流量控制针对的是点对点之间的（发送方和接收方）之间的速度匹配服务，因为接收方的应用程序读取的速度不一定很迅速，而接收方的缓存是有限的，就需要避免发送的速度过快而导致的问题。拥塞控制是由于网络中的路由和链路传输速度限制，要避免网络的过载和进行的控制。

（2）拥塞控制的几个指导原则

**第一：**一个丢失的报文段意味着拥塞，因此当报文段丢失时应该降低TCP发送方的速率；

**第二：**当一个确认的报文段到达时，表示当前网络是顺畅的，即可以提升发送方的速率。

**第三：**ACK和丢包事件充当了隐式的信号，指导着拥塞控制。

（3）拥塞控制算法

拥塞控制是使用拥塞窗口来进行控制的，运行在发送方的拥塞控制机制通过跟踪一个额外的变量，**即拥塞窗口（cwnd），**它对于发送方能够向网络中发送流量的速率进行了限制，对于发送方：

LastByteSent – LastByteAcked <= cwnd

拥塞控制算法主要包含了三个部分：**慢启动、拥塞避免和快速回复**

**第一、慢启动**

慢开始算法的思路就是，不要一开始就发送大量的数据，先探测一下网络的拥塞程度，也就是说由小到大逐渐增加拥塞窗口的大小。一般一开始为1个MSS，之后翻倍这样来增加，呈指数增长。其中1、慢启动过程有一个**阈值ssthresh，**一旦到达阈值就进入**拥塞避免模式**。这是第一种离开结束慢启动的方式2、如果收到了一个丢包提示，就将cwnd设为1并且重新开始慢启动过程，这时要把**阈值ssthresh**设为当前cwnd值的一半。3、如果收到了三次冗余的ACK，就执行一次快速重传并且进入快速恢复状态，这是最后一种结束慢启动的过程。

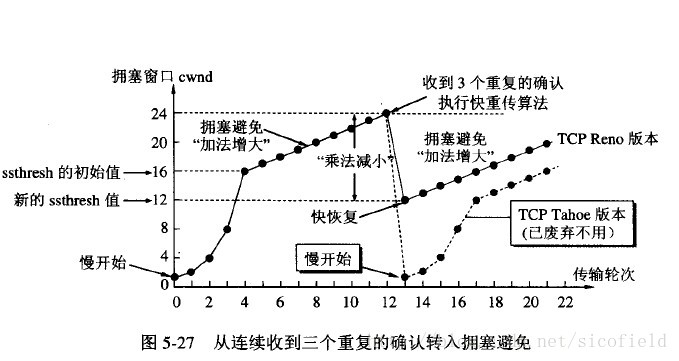
**第二、拥塞避免**

进入拥塞避免说明cwnd值大约是上一次遇到拥塞是的一半，这时候不能翻倍，而是将cwnd的值每次增加一个MSS。结束的过程有两种可能：1、当出现超时时，将cwnd值设为1个MSS，并且将ssthresh阈值设为当前cwnd值的一半。2、当收到三个冗余ACK时，将ssthresh阈值设为当前cwnd值的一半，并且将cwnd值设为当前cwnd值的一半加3，即ssthresh阈值加3，并且进入快速恢复状态。

**第三、快速恢复**

快速恢复就是指进入快速恢复前的一系列操作，即将ssthresh阈值设为当前cwnd值的一半，并且将cwnd值设为当前cwnd值的一半加3，即ssthresh阈值加3，之后进入拥塞避免状态，即每次cwnd的值加1个MSS。

见下图：

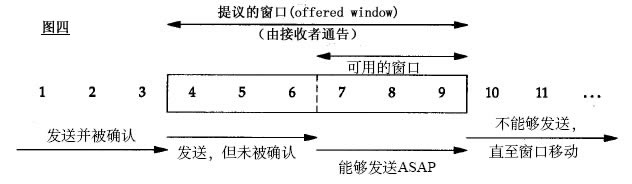


**6. TCP滑动窗口与回退N针协议。**

**针对TCP流量控制：**

（1）滑动窗口协议

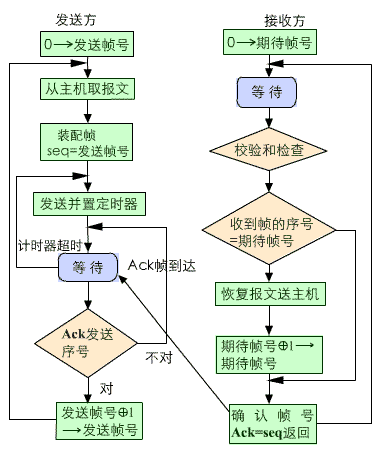
发送和接受方都会维护一个数据帧的序列，这个序列被称作窗口。发送方的窗口大小由接受方确定，目的在于控制发送速度，以免接受方的缓存不够大，而导致溢出，同时控制流量也可以避免网络拥塞。下面图中的4,5,6号数据帧已经被发送出去，但是未收到关联的ACK，7,8,9帧则是等待发送。可以看出发送端的窗口大小为6，这是由接受端告知的（事实上必须考虑拥塞窗口cwnd，这里暂且考虑cwnd>rwnd）。此时如果发送端收到4号ACK，则窗口的左边缘向右收缩，窗口的右边缘则向右扩展，此时窗口就向前“滑动了”，即数据帧10也可以被发送。

[](http://blog.csdn.net/images/blog_csdn_net/dotjox/slidingwindow.jpg)

（2）

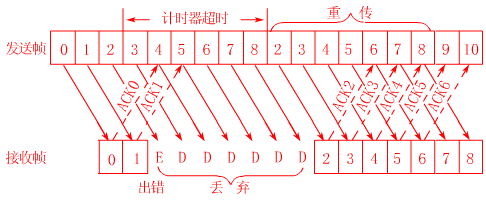
**第一种：比特滑动窗口协议（停等协议）**

当发送窗口和接收窗口的大小固定为1时，滑动窗口协议退化为停等协议（stop－and－wait）。该协议规定发送方每发送一帧后就要停下来，等待接收方已正确接收的确认（acknowledgement）返回后才能继续发送下一帧。由于接收方需要判断接收到的帧是新发的帧还是重新发送的帧，因此发送方要为每一个帧加一个序号。由于停等协议规定只有一帧完全发送成功后才能发送新的帧，因而只用一比特来编号就够了。其发送方和接收方运行的流程图如图所示。



**第二种：回退N帧协议**

由于停等协议要为每一个帧进行确认后才继续发送下一帧，大大降低了信道利用率，因此又提出了后退n协议。后退n协议中，发送方在发完一个数据帧后，不停下来等待应答帧，而是连续发送若干个数据帧，即使在连续发送过程中收到了接收方发来的应答帧，也可以继续发送。且发送方在每发送完一个数据帧时都要设置超时定时器。只要在所设置的超时时间内仍收到确认帧，就要重发相应的数据帧。如：当发送方发送了N个帧后，若发现该N帧的前一个帧在计时器超时后仍未返回其确认信息，则该帧被判为出错或丢失，此时发送方就不得不重新发送出错帧及其后的N帧。



从这里不难看出，后退n协议一方面因连续发送数据帧而提高了效率，但另一方面，在重传时又必须把原来已正确传送过的数据帧进行重传（仅因这些数据帧之前有一个数据帧出了错），这种做法又使传送效率降低。由此可见，若传输信道的传输质量很差因而误码率较大时，连续测协议不一定优于停止等待协议。此协议中的发送窗口的大小为k，接收窗口仍是1。

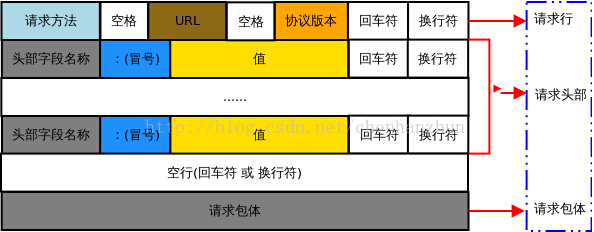
**第三种：选择重传协议**

在后退n协议中，接收方若发现错误帧就不再接收后续的帧，即使是正确到达的帧，这显然是一种浪费。另一种效率更高的策略是当接收方发现某帧出错后，其后继续送来的正确的帧虽然不能立即递交给接收方的高层，但接收方仍可收下来，存放在一个缓冲区中，同时要求发送方重新传送出错的那一帧。一旦收到重新传来的帧后，就可以原已存于缓冲区中的其余帧一并按正确的顺序递交高层。这种方法称为选择重发(SELECTICE REPEAT)，其工作过程如图所示。显然，选择重发减少了浪费，但要求接收方有足够大的缓冲区空间。

**7. Http的报文结构。**

**（1）HTTP 请求报文**

　　HTTP 请求报文由请求行、请求头部、空行 和 请求包体 4 个部分组成，如下图所示：



　　下面对请求报文格式进行简单的分析：

　　请求行：请求行由方法字段、URL 字段 和HTTP 协议版本字段 3 个部分组成，他们之间使用空格隔开。常用的 HTTP 请求方法有 GET、POST、HEAD、PUT、DELETE、OPTIONS、TRACE、CONNECT;

　　● GET：当客户端要从服务器中读取某个资源时，使用GET 方法。GET 方法要求服务器将URL 定位的资源放在响应报文的[数据](http://ad.doubleclick.net/ddm/trackclk/N7442.5006CHINABYTE/B10313247.138166523;dc_trk_aid=310538354;dc_trk_cid=74205219)部分，回[送给](http://www.chinabyte.com/keyword/%E9%80%81%E7%BB%99/)客户端，即向服务器请求某个资源。使用GET 方法时，请求参数和对应的值附加在 URL 后面，利用一个问号(“?”)代表URL 的结尾与请求参数的开始，传递参数长度受限制。例如，/index.jsp?id=100&op=bind。

　　● POST：当客户端给服务器提供信息较多时可以使用POST 方法，POST 方法向服务器提交数据，比如完成表单数据的提交，将数据提交给服务器处理。GET 一般用于获取/查询资源信息，POST 会附带用户数据，一般用于更新资源信息。POST 方法将请求参数[封装](http://www.chinabyte.com/keyword/%E5%B0%81%E8%A3%85/)在HTTP 请求数据中，以名称/值的形式出现，可以传输大量数据;

　　请求头部：请求头部由关键字/值对组成，每行一对，关键字和值用英文冒号“:”分隔。请求头部通知服务器有关于客户端请求的信息，典型的请求头有：

　　● User-Agent：产生请求的[浏览器](http://www.chinabyte.com/keyword/%E6%B5%8F%E8%A7%88%E5%99%A8/)类型;

　　● Accept：客户端可识别的响应内容类型列表;星号 “ \* ” 用于按范围将类型分组，用 “ \*/\* ” 指示可接受全部类型，用“ type/\* ”指示可接受 type 类型的所有子类型;

　　● Accept-Language：客户端可接受的自然语言;

　　● Accept-Encoding：客户端可接受的编码压缩格式;

　　● Accept-Charset：可接受的应答的字符集;

　　● Host：请求的主机名，允许多个[域名](http://www.chinabyte.com/keyword/%E5%9F%9F%E5%90%8D/)同处一个IP 地址，即虚拟主机;

　　● connection：连接方式(close 或 keepalive);

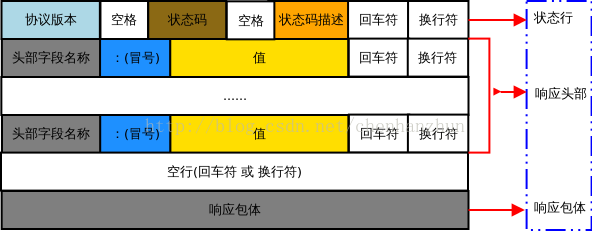
　　● Cookie：[存储](http://storage.chinabyte.com/)于客户端扩展字段，向同一域名的服务端发送属于该域的cookie;

　　空行：最后一个请求头之后是一个空行，发送回车符和换行符，通知服务器以下不再有请求头;

请求包体：请求包体不在 GET 方法中使用，而是在POST 方法中使用。POST 方法适用于需要客户填写表单的场合。与请求包体相关的最常使用的是包体类型 Content-Type 和包体长度 Content-Length;

（2）HTTP 响应报文

　　HTTP 响应报文由状态行、响应头部、空行 和 响应包体 4 个部分组成，如下图所示：



　　下面对响应报文格式进行简单的分析：

　　状态行：状态行由 HTTP 协议版本字段、状态码和状态码的描述文本 3 个部分组成，他们之间使用空格隔开;

　　● 状态码由三位数字组成，第一位数字表示响应的类型，常用的状态码有五大类如下所示：

　　1xx：表示服务器已接收了客户端请求，客户端可继续发送请求;

　　2xx：表示服务器已成功接收到请求并进行处理;

　　3xx：表示服务器要求客户端重定向;

　　4xx：表示客户端的请求有非法内容;

　　5xx：表示服务器未能正常处理客户端的请求而出现意外错误;

　　● 状态码描述文本有如下取值：

　　200 OK：表示客户端请求成功;

　　400 Bad Request：表示客户端请求有语法错误，不能被服务器所理解;

　　401 Unauthonzed：表示请求未经授权，该状态代码必须与 WWW-Authenticate 报头域一起使用;

　　403 Forbidden：表示服务器收到请求，但是拒绝提供服务，通常会在响应正文中给出不提供服务的原因;

　　404 Not Found：请求的资源不存在，例如，输入了错误的URL;

　　500 Internal Server Error：表示服务器发生不可预期的错误，导致无法完成客户端的请求;

　　503 Service Unavailable：表示服务器当前不能够处理客户端的请求，在一段时间之后，服务器可能会恢复正常;

　　响应头部：响应头可能包括：

　　Location：Location响应报头域用于重定向接受者到一个新的位置。例如：客户端所请求的页面已不存在原先的位置，为了让客户端重定向到这个页面新的位置，服务器端可以发回Location响应报头后使用重定向语句，让客户端去访问新的域名所对应的服务器上的资源;

　　Server：Server 响应报头域包含了服务器用来处理请求的软件信息及其版本。它和 User-Agent 请求报头域是相对应的，前者发送服务器端软件的信息，后者发送客户端软件(浏览器)和[操作系统](http://soft.chinabyte.com/os/)的信息。

　　Vary：指示不可缓存的请求头列表;

　　Connection：连接方式;

　　对于请求来说：close(告诉[WEB 服务器](http://www.chinabyte.com/keyword/web%E6%9C%8D%E5%8A%A1%E5%99%A8/)或者[代理服务器](http://www.chinabyte.com/keyword/%E4%BB%A3%E7%90%86%E6%9C%8D%E5%8A%A1%E5%99%A8/)，在完成本次请求的响应后，断开连接，不等待本次连接的后续请求了)。keepalive(告诉WEB服务器或者代理服务器，在完成本次请求的响应后，保持连接，等待本次连接的后续请求);

　　对于响应来说：close(连接已经关闭); keepalive(连接保持着，在等待本次连接的后续请求); Keep-Alive：如果浏览器请求保持连接，则该头部表明希望WEB 服务器保持连接多长时间(秒);例如：Keep-Alive：300;

　　WWW-Authenticate：WWW-Authenticate响应报头域必须被包含在401 (未授权的)响应消息中，这个报头域和前面讲到的Authorization 请求报头域是相关的，当客户端收到 401 响应消息，就要决定是否请求服务器对其进行验证。如果要求服务器对其进行验证，就可以发送一个包含了Authorization 报头域的请求;

　　空行：最后一个响应头部之后是一个空行，发送回车符和换行符，通知服务器以下不再有响应头部。

　　响应包体：服务器返回给客户端的文本信息;

　　HTTP 工作原理

　　HTTP 协议采用请求/响应模型。客户端向服务器发送一个请求报文，服务器以一个状态作为响应。

　　以下是 HTTP 请求/响应的步骤：

　　● 客户端连接到web服务器：HTTP 客户端与web服务器建立一个 TCP 连接;

　　● 客户端向服务器发起 HTTP 请求：通过已建立的TCP 连接，客户端向服务器发送一个请求报文;

　　● 服务器接收 HTTP 请求并返回 HTTP 响应：服务器解析请求，定位请求资源，服务器将资源副本写到 TCP 连接，由客户端读取;

　　● 释放 TCP 连接：若connection 模式为close，则服务器主动关闭TCP 连接，客户端被动关闭连接，释放TCP 连接;若connection 模式为keepalive，则该连接会保持一段时间，在该时间内可以继续接收请求;

　　● 客户端浏览器解析HTML内容：客户端将服务器响应的 html 文本解析并显示;

　　例如：在浏览器地址栏键入URL，按下回车之后会经历以下流程：

　　1、浏览器向[DNS 服务器](http://www.chinabyte.com/keyword/DNS%E6%9C%8D%E5%8A%A1%E5%99%A8/)请求解析该 URL 中的域名所对应的 IP 地址;

　　2、解析出 IP 地址后，根据该 IP 地址和默认端口 80，和服务器建立 TCP 连接;

　　3、浏览器发出读取文件(URL 中域名后面部分对应的文件)的HTTP 请求，该请求报文作为 TCP 三次握手的第三个报文的数据发送给服务器;

　　4、服务器对浏览器请求作出响应，并把对应的 html 文本发送给浏览器;

　　5、释放 TCP 连接;

　　6、浏览器将该 html 文本并显示内容;

**8. Http的状态码含义。**

其中，HTTP-Version表示服务器HTTP协议的版本；Status-Code表示服务器发回的响应状态代码；Reason-Phrase表示状态代码的文本描述。状态代码由三位数字组成，第一个数字定义了响应的类别，且有五种可能取值。

* 1xx：指示信息--表示请求已接收，继续处理。
* 2xx：成功--表示请求已被成功接收、理解、接受。
* 3xx：重定向--要完成请求必须进行更进一步的操作。
* 4xx：客户端错误--请求有语法错误或请求无法实现。
* 5xx：服务器端错误--服务器未能实现合法的请求。

常见状态代码、状态描述的说明如下。

* 200 OK：客户端请求成功。
* 400 Bad Request：客户端请求有语法错误，不能被服务器所理解。
* 401 Unauthorized：请求未经授权，这个状态代码必须和WWW-Authenticate报头域一起使用。
* 403 Forbidden：服务器收到请求，但是拒绝提供服务。
* 404 Not Found：请求资源不存在，举个例子：输入了错误的URL。
* 500 Internal Server Error：服务器发生不可预期的错误。
* 503 Server Unavailable：服务器当前不能处理客户端的请求，一段时间后可能恢复正常，举个例子：HTTP/1.1 200 OK（CRLF）。

**9. Http request的几种类型。**

GET、POST、HEAD、PUT、DELETE、OPTIONS、TRACE、CONNECT

**10. Http1.1和Http1.0的区别**

1，HTTP/1.0协议使用非持久连接,即在非持久连接下,一个tcp连接只传输一个Web对象；  
2，HTTP/1.1默认使用持久连接(然而,HTTP/1.1协议的客户机和服务器可以配置成使用非持久连接)。

**11. Http怎么处理长连接。**

**（1）在HTTP/1.0中，默认使用的是短连接**。也就是说，浏览器和服务器每进行一次HTTP操作，就建立一次连接，但任务结束就中断连接。如果客户端浏览器访问的某个HTML或其他类型的 Web页中包含有其他的Web资源，如JavaScript文件、图像文件、CSS文件等；当浏览器每遇到这样一个Web资源，就会建立一个HTTP会话。

但从 **HTTP/1.1起，默认使用长连接**，用以保持连接特性。使用长连接的HTTP协议，会在响应头有加入这行代码：**Connection**:keep-alive

**（2）优缺点**

由上可以看出，**长连接**可以**省去较多的TCP建立和关闭的操作，减少浪费，节约时间**。对于频繁请求资源的客户来说，较适用长连接。不过这里**存在一个问题**，**存活功能的探测周期太长**，还有就是它只是探测TCP连接的存活，属于比较斯文的做法，遇到恶意的连接时，保活功能就不够使了。在长连接的应用场景下，client端一般不会主动关闭它们之间的连接，**Client与server之间的连接如果一直不关闭的话，会存在一个问题，随着客户端连接越来越多，server早晚有扛不住的时候**，这时候server端需要采取一些策略，如关闭一些长时间没有读写事件发生的连接，这样可 以避免一些恶意连接导致server端服务受损；如果条件再允许就可以以客户端机器为颗粒度，限制每个客户端的最大长连接数，这样可以完全避免某个蛋疼的客户端连累后端服务。

**短连接**对于服务器来说管理较为简单，存在的连接都是有用的连接，不需要额外的控制手段。但如果客户**请求频繁**，将在**TCP的建立和关闭操作上浪费时间和带宽**。

长连接和短连接的产生在于client和server采取的关闭策略，具体的应用场景采用具体的策略，没有十全十美的选择，只有合适的选择。

**（3）什么时候用长连接，短连接？**

**长连接**多用于操作频繁，点对点的通讯，而且连接数不能太多情况，。每个TCP连接都需要三步握手，这需要时间，如果每个操作都是先连接，再操作的话那么处理速度会降低很多，所以每个操作完后都不断开，次处理时直接发送数据包就OK了，不用建立TCP连接。例如：数据库的连接用长连接， 如果用短连接频繁的通信会造成socket错误，而且频繁的socket 创建也是对资源的浪费。

而像WEB网站的http服务一般都用**短链接**，因为长连接对于服务端来说会耗费一定的资源，而像WEB网站这么频繁的成千上万甚至上亿客户端的连接用短连接会更省一些资源，如果用长连接，而且同时有成千上万的用户，如果每个用户都占用一个连接的话，那可想而知吧。所以并发量大，但每个用户无需频繁操作情况下需用短连好。

12. Cookie与Session的作用与原理。

**13. 电脑上访问一个网页，整个过程是怎么样的：DNS、HTTP、TCP、OSPF、IP、ARP。**

答案：

**1.本地过程：**

1. 若DNS缓存中没有相关数据，则IE浏览器先向DNS服务器发出DNS请求：
2. 这一过程的目的是获取www.sina.com这个域名所对应的IP地址；
3. IE浏览器向本机DNS模块发出DNS请求，DNS模块生成相关的DNS报文；
4. DNS模块将生成的DNS报文传递给传输层的UDP协议单元；
5. UDP协议单元将该数据封装成UDP数据报，传递给网络层的IP协议单元；
6. IP协议单元将该数据封装成IP数据包，其中目的IP地址为DNS服务器的IP地址；
7. 封装好的IP数据包将传递给数据链路层的协议单元进行发送；
8. 发送时如果ARP缓存中没有相关数据，则发送ARP广播请求，等待ARP回应；
9. 得到ARP回应后，将IP地址与路由下一跳MAC地址对应的信息写入ARP缓存表；
10. 写入缓存后，以路由下一跳地址填充目的MAC地址，并以数据帧形式转发；
11. 这个转发过程可能会进行多次，这取决于DNS服务器在校园网中的位置；
12. DNS请求被发送到DNS服务器的数据链路层协议单元；
13. DNS服务器的数据链路层协议单元解析收到的数据帧，将其内部所含有的IP数据包传递给网络层IP协议单元；
14. DNS服务器的IP协议单元解析收到的IP数据包，将其内部所含有的UDP数据报传递给传输层的UDP协议单元；
15. DNS服务器的UDP协议单元解析收到的UDP数据包，将其内部所含有的DNS报文传递给该服务器上的DNS服务单元；
16. DNS服务单元收到DNS请求，将域名解析为对应的IP地址，产生DNS回应报文；
17. （所有应用层报文必须通过传输层、网络层和数据链路层，因此在下面的叙述中，我将简化这一过程的叙述，简化形式如下面的样子，其中单箭头为本机内部传递，双箭头为网络上的发送）
18. DNS回应报文→UDP→IP→MAC→→请求域名解析的主机；
19. 请求域名解析的主机收到数据帧，该数据帧→IP→UDP→DNS→IE浏览器；
20. 将域名解析的结果以域名和IP地址对应的形式写入DNS缓存表。

**2.IE浏览器与**[www.sina.com.cn](http://www.sina.com.cn/)**建立TCP连接：TCP建立连接的三次握手**

1. IE浏览器向www.sina.com.cn发出TCP连接请求报文；
2. 该请求TCP报文中的SYN标志位被设置为1，表示连接请求；
3. 该TCP请求报文→IP(DNS)→MAC(ARP)→→校园网关→→www.sina.com.cn主机；
4. 该TCP请求报文经过IP层时，填入的目的IP地址就是上面DNS过程获得的IP地址；
5. 经过数据链路层时，若MAC地址不明，还要进行上面所叙述的ARP过程；
6. www.sina.com.cn收到的数据帧→IP→TCP，TCP协议单元会回应请求应答报文；
7. 该请求应答TCP报文中的SYN和ACK标志位均被设置为1，表示连接请求应答；
8. 该TCP请求应答报文→IP→MAC(ARP)→→校园网关→→请求主机；
9. 请求主机收到数据帧→IP→TCP，TCP协议单元会回应请求确认报文；
10. 该请求应答TCP报文中的ACK标志位被设置为1，表示连接请求确认；
11. 该TCP请求确认报文→IP→MAC(ARP)→→校园网关→→www.sina.com.cn主机；
12. www.sina.com.cn收到的数据帧→IP→TCP，连接建立完成；

**在这个过程中，任何一个报文出错或超时，都要进行重传；  
这个过程被称为TCP建立连接的三次握手。**

**3.IE浏览器开始HTTP访问过程**

1. IE浏览器向www.sina.com.cn发出HTTP-GET方法报文；
2. 该HTTP-GET方法报文→TCP→IP→MAC→→校园网关→→www.sina.com.cn主机；
3. www.sina.com.cn收到的数据帧→IP→TCP→HTTP，HTTP协议单元会回应HTTP协议格式封装好的HTML超文本形式数据；
4. HTTP-HTML数据→TCP→IP→MAC(ARP)→→校园网关→→请求主机；
5. 请求主机收到的数据帧→IP→TCP→HTTP→IE浏览器，浏览器会以网页形式显示HTML超文本，就是我们所看到的网页。

**4.断开TCP连接：TCP断开连接的四次握手**

1. IE浏览器向www.sina.com.cn发出TCP连接结束请求报文；
2. 该请求TCP报文中的FIN标志位被设置为1，表示结束请求；
3. 该TCP结束请求报文→IP→MAC(ARP)→→校园网关→→www.sina.com.cn主机；
4. www.sina.com.cn收到的数据帧→IP→TCP，TCP协议单元会回应结束应答报文；
5. 该结束应答TCP报文中的FIN和ACK标志位均被设置为1，表示结束应答；
6. 该TCP结束应答报文→IP→MAC(ARP)→→校园网关→→请求主机；

这个过程需要双向进行，因此[www.sina.com.cn](http://www.sina.com.cn/)主机也会按上述流程再做一次；  
整个过程被称为TCP断开连接的四次挥手

**14. Ping的整个过程。ICMP报文是什么。**

**（1）ICMP协议**

它 **用于TCP/IP网络中发送控制消息**，提供可能发生在通信环境中的各种问题反馈，通过这些信息，令管理者可以对所发生的问题作出诊断，然后采取适当的措施解决。它与传输协议最大的不同：它一般不用于在两点间传输数据，而常常 **用于返回的错误信息或是分析路由**。

ICMP控制的内容包括但不仅限于：echo响应（ping）、目标网络不可达、目标端口不可达、禁止访问的网络、拥塞控制、重定向、TTL超时...

它是[TCP/IP协议](http://baike.baidu.com/view/7649.htm)族的一个子协议，**属于网络层协议**，主要用于在主机与路由器之间传递控制信息，包括报告错误、交换受限控制和状态信息等。当遇到IP数据无法访问目标、IP[路由器](http://baike.baidu.com/view/1360.htm)无法按当前的传输速率转发[数据包](http://baike.baidu.com/view/25880.htm)等情况时，会自动发送ICMP消息。ICMP报文在IP帧结构的首部协议类型字段（Protocol 8bit)的值=1.

（2）Ping的过程

**类一：相同网段：**

1. 主机A要去Ping主机B， 主机A会封装两层报文，主机A先检查自己MAC地址中是否有B的MAC地址，如果没有就向外发送一个ARP广播包
2. 交换机收到这个ARP后，会检查在交换机中是否包含B的MAC地址，如果有就直接返回给A；如果没有就向所有端口发送ARP，该网段的主机的MAC如果与B的MAC地址不同就丢弃，如果主机B收到了该ARP就马上返回相同格式的ARP
3. 这时主机A已经有了B的MAC地址，就把B的MAC地址封装到ICMP报文中，向主机B发送一个回显请求
4. 主机B收到该报文后，知道是主机A的一个回显请求，就会返回一个相同格式的报文。这样就完成了同一个网段的Ping的过程

**不同网段：ping某个域名的过程**

* 首先本机发送域名请求数据到PC设置的DNS ip
* PC通过子网掩码判断DNS ip是本网段还是跨网段（这里只考虑跨网段）
* 由于是跨网段，PC发送DNS域名解析数据包到PC设置的网关ip上。（此时先要进行二层的mac转发，PC查看本机arp缓存表，如果表中有网关的mac地址，直接转发，如果没有，使用arp解析协议解析到网关的mac地址。之后封装成数据帧发送到三层网络层）此时PC发送三层数据到网关，源地址为PC内网地址，目的地址为DNS ip地址。而在二层源mac地址为PC mac地址，目的mac地址为网关mac地址。
* 路由内网网关收到数据包，根据数据包的目的地址，查看路由表。根据路由表发送数据到下一跳上。（发送前，数据到达路由外网端口，会根据nat地址转换配置。形成一条内网ip+port与外网ip+port的一一对应关系。）
* 发送到下一跳和内网通信都是一样的，查看路由arp缓存表，如果有下一跳mac地址，就直接发送，没有的话需要arp协议解析一下。
* 对端路由收到数据包，再接着根据路由表判断下一跳。这样一跳一跳地，最后到达DNS服务器。服务器将查询结果返回。
* 返回的数据包在ISP的网络里最后寻址到你的路由器上，你的路由器收到数据包后，会查询路由nat连接表，寻找ip+port关系对应的内网ip。拆分数据包，封装成帧，最后PC收到域名对应的ip地址。

【到这里，域名解析过程完成，接下来ping对方ip，过程与上面几乎一样】

* 再发起一次PC到目的域名ip地址的一次ping请求信息
* PC通过子网掩码判断对方ip是本网段还是跨网段（这里只考虑跨网段）
* 由于是跨网段，PC发送数据包到网关ip上。
* 路由内网网关收到数据包，根据数据包的目的地址，查看路由表。根据路由表发送数据到下一跳上。（发送前，数据到达路由外网端口，会根据nat地址转换配置。形成一条内网IP+port与外网ip+port的一一对应关系。）
* 发送到下一跳和内网通信都是一样的，查看路由arp缓存表，如果有下一跳mac地址，就直接发送，没有的话就是要arp协议解析一下。
* 服务器收到数据包后，会重新构建一个ICMP应答包，然后返回。
* 返回的数据包在ISP的网络里最后寻址到你的路由器上，你的路由器收到数据包后，会查询路由nat连接表，寻找ip+port关系对应的内网ip。拆分数据包，封装成帧，最后PC收到ICMP应答数据包。   
  整个过程到此结束。在整个这个过程中，源ip地址和目的ip地址是不变的（内网到路由器段不算在内）而mac地址是变的。

15. C/S模式下使用socket通信，几个关键函数。

**16. IP地址分类。**

A类:1.0.0.0~126.255.255.255,默认子网掩码/8,即255.0.0.0 (其中127.0.0.0~127.255.255.255为[环回地址](https://www.baidu.com/s?wd=%E7%8E%AF%E5%9B%9E%E5%9C%B0%E5%9D%80&tn=44039180_cpr&fenlei=mv6quAkxTZn0IZRqIHckPjm4nH00T1YzrjDknj7WP1wWuAuBmv7W0ZwV5Hcvrjm3rH6sPfKWUMw85HfYnjn4nH6sgvPsT6KdThsqpZwYTjCEQLGCpyw9Uz4Bmy-bIi4WUvYETgN-TLwGUv3EPHRznHmLrHn1),用于本地[环回测试](https://www.baidu.com/s?wd=%E7%8E%AF%E5%9B%9E%E6%B5%8B%E8%AF%95&tn=44039180_cpr&fenlei=mv6quAkxTZn0IZRqIHckPjm4nH00T1YzrjDknj7WP1wWuAuBmv7W0ZwV5Hcvrjm3rH6sPfKWUMw85HfYnjn4nH6sgvPsT6KdThsqpZwYTjCEQLGCpyw9Uz4Bmy-bIi4WUvYETgN-TLwGUv3EPHRznHmLrHn1)等用途)；  
  
B类:128.0.0.0~191.255.255.255,默认子网掩码/16,即255.255.0.0；  
  
C类:192.0.0.0~223.255.255.255,默认子网掩码/24,即255.255.255.0；  
  
D类:224.0.0.0~239.255.255.255,一般于用组播  
  
E类:240.0.0.0~255.255.255.255(其中255.255.255.255为全网[广播地址](https://www.baidu.com/s?wd=%E5%B9%BF%E6%92%AD%E5%9C%B0%E5%9D%80&tn=44039180_cpr&fenlei=mv6quAkxTZn0IZRqIHckPjm4nH00T1YzrjDknj7WP1wWuAuBmv7W0ZwV5Hcvrjm3rH6sPfKWUMw85HfYnjn4nH6sgvPsT6KdThsqpZwYTjCEQLGCpyw9Uz4Bmy-bIi4WUvYETgN-TLwGUv3EPHRznHmLrHn1)),[E类地址](https://www.baidu.com/s?wd=E%E7%B1%BB%E5%9C%B0%E5%9D%80&tn=44039180_cpr&fenlei=mv6quAkxTZn0IZRqIHckPjm4nH00T1YzrjDknj7WP1wWuAuBmv7W0ZwV5Hcvrjm3rH6sPfKWUMw85HfYnjn4nH6sgvPsT6KdThsqpZwYTjCEQLGCpyw9Uz4Bmy-bIi4WUvYETgN-TLwGUv3EPHRznHmLrHn1)一般用于研究用途..

**17. 路由器与交换机区别。**

**（1）交换机(Switch)**

是一种基于MAC（网卡的硬件地址）识别，能完成封装转发数据包功能的网络设备。交换机可以“学习”MAC地址，并把其存放在内部地址表中，通过在数据帧的始发者和目标接收者之间建立临时的交换路径，使数据帧直接由源地址到达目的地址。现在的交换机分为：[二层交换机](https://www.baidu.com/s?wd=%E4%BA%8C%E5%B1%82%E4%BA%A4%E6%8D%A2%E6%9C%BA&tn=44039180_cpr&fenlei=mv6quAkxTZn0IZRqIHckPjm4nH00T1YduWF9ujc1rHN9Pvf4rHD30ZwV5Hcvrjm3rH6sPfKWUMw85HfYnjn4nH6sgvPsT6KdThsqpZwYTjCEQLGCpyw9Uz4Bmy-bIi4WUvYETgN-TLwGUv3En1bLPWfvnHb3)，三层交换机或是更高层的交换机。三层交换机同样可以有路由的功能，而且比低端路由器的转发速率更快。它的主要特点是：一次路由，多次转发。 　　  
**（2）路由器(Router)**

是在网络层实现互连的设备。它比网桥更加复杂，也具有更大的灵活性。路由器有更强的异种网互连能力，连接对象包括局域网和广域网。过去路由器多用于广域网，近年来，由于路由器性能有了很大提高，价格下降到与网桥接近，因此在局域网互连中也越来越多地使用路由器。路由器是一种连接多个网络或网段的网络设备，它能将不同网络或网段之间的数据信息进行“翻译”，以使它们能够相互“读”懂对方的数据，从而构成一个更大的网络。路由器有两大典型功能，即数据通道功能和控制功能。数据通道功能包括转发决定、背板转发以及输出链路调度等，一般由特定的硬件来完成；控制功能一般用软件来实现，包括与相邻路由器之间的[信息交换](https://www.baidu.com/s?wd=%E4%BF%A1%E6%81%AF%E4%BA%A4%E6%8D%A2&tn=44039180_cpr&fenlei=mv6quAkxTZn0IZRqIHckPjm4nH00T1YduWF9ujc1rHN9Pvf4rHD30ZwV5Hcvrjm3rH6sPfKWUMw85HfYnjn4nH6sgvPsT6KdThsqpZwYTjCEQLGCpyw9Uz4Bmy-bIi4WUvYETgN-TLwGUv3En1bLPWfvnHb3)、系统配置、系统管理等。 　　  
**（3）区别**　  
 （1）工作层次不同 　　最初的的交换机是工作在OSI/RM开放体系结构的数据链路层，也就是第二层，而路由器一开始就[设计工作](https://www.baidu.com/s?wd=%E8%AE%BE%E8%AE%A1%E5%B7%A5%E4%BD%9C&tn=44039180_cpr&fenlei=mv6quAkxTZn0IZRqIHckPjm4nH00T1YduWF9ujc1rHN9Pvf4rHD30ZwV5Hcvrjm3rH6sPfKWUMw85HfYnjn4nH6sgvPsT6KdThsqpZwYTjCEQLGCpyw9Uz4Bmy-bIi4WUvYETgN-TLwGUv3En1bLPWfvnHb3)在OSI模型的网络层。由于交换机工作在OSI的第二层（数据链路层），所以它的工作原理比较简单，而路由器工作在OSI的第三层（网络层），可以得到更多的协议信息，路由器可以做出更加智能的转发决策。 　　  
 （2）数据转发所依据的对象不同 　　交换机是利用物理地址或者说MAC地址来确定转发数据的目的地址。而路由器则是利用不同网络的ID号（即IP地址）来确定数据转发的地址。IP地址是在软件中实现的，描述的是设备所在的网络，有时这些第三层的地址也称为协议地址或者网络地址。MAC地址通常是硬件自带的，由网卡生产商来分配的，而且已经固化到了网卡中去，一般来说是不可更改的。而IP地址则通常由[网络管理员](https://www.baidu.com/s?wd=%E7%BD%91%E7%BB%9C%E7%AE%A1%E7%90%86%E5%91%98&tn=44039180_cpr&fenlei=mv6quAkxTZn0IZRqIHckPjm4nH00T1YduWF9ujc1rHN9Pvf4rHD30ZwV5Hcvrjm3rH6sPfKWUMw85HfYnjn4nH6sgvPsT6KdThsqpZwYTjCEQLGCpyw9Uz4Bmy-bIi4WUvYETgN-TLwGUv3En1bLPWfvnHb3)或系统自动分配。 　　  
 （3）传统的交换机只能分割冲突域，不能分割广播域；而路由器可以分割广播域 　　由交换机连接的网段仍属于同一个广播域，广播数据包会在交换机连接的所有网段上传播，在某些情况下会导致通信拥挤和安全漏洞。连接到路由器上的网段会被分配成不同的广播域，广播数据不会穿过路由器。虽然第三层以上交换机具有VLAN功能，也可以分割广播域，但是各子广播域之间是不能通信交流的，它们之间的交流仍然需要路由器。 　　  
 （4）路由器提供了防火墙的服务 　　  
 路由器仅仅转发特定地址的数据包，不传送不支持路由协议的数据包传送和未知目标网络数据包的传送，从而可以防止[广播风暴](https://www.baidu.com/s?wd=%E5%B9%BF%E6%92%AD%E9%A3%8E%E6%9A%B4&tn=44039180_cpr&fenlei=mv6quAkxTZn0IZRqIHckPjm4nH00T1YduWF9ujc1rHN9Pvf4rHD30ZwV5Hcvrjm3rH6sPfKWUMw85HfYnjn4nH6sgvPsT6KdThsqpZwYTjCEQLGCpyw9Uz4Bmy-bIi4WUvYETgN-TLwGUv3En1bLPWfvnHb3)。 　　  
 交换机一般用于LAN-WAN的连接，交换机归于网桥，是数据链路层的设备，有些交换机也可实现第三层的交换。路由器用于WAN-WAN之间的连接，可以解决异性网络之间转发分组，作用于网络层。他们只是从一条线路上接受输入分组，然后向另一条线路转发。这两条线路可能分属于不同的网络，并采用不同协议。相比较而言，路由器的功能较交换机要强大，但速度相对也慢，价格昂贵，[第三层交换机](https://www.baidu.com/s?wd=%E7%AC%AC%E4%B8%89%E5%B1%82%E4%BA%A4%E6%8D%A2%E6%9C%BA&tn=44039180_cpr&fenlei=mv6quAkxTZn0IZRqIHckPjm4nH00T1YduWF9ujc1rHN9Pvf4rHD30ZwV5Hcvrjm3rH6sPfKWUMw85HfYnjn4nH6sgvPsT6KdThsqpZwYTjCEQLGCpyw9Uz4Bmy-bIi4WUvYETgN-TLwGUv3En1bLPWfvnHb3)既有交换机[线速转发](https://www.baidu.com/s?wd=%E7%BA%BF%E9%80%9F%E8%BD%AC%E5%8F%91&tn=44039180_cpr&fenlei=mv6quAkxTZn0IZRqIHckPjm4nH00T1YduWF9ujc1rHN9Pvf4rHD30ZwV5Hcvrjm3rH6sPfKWUMw85HfYnjn4nH6sgvPsT6KdThsqpZwYTjCEQLGCpyw9Uz4Bmy-bIi4WUvYETgN-TLwGUv3En1bLPWfvnHb3)报文能力，又有路由器良好的控制功能，因此得以广泛应用。

网络其实大体分为两块，一个TCP协议，一个HTTP协议，只要把这两块以及相关协议搞清楚，一般问题不大。