**1、OSI参考模型：**

       OSI（Open System Interconnect），即开放式系统互联。一般都称为OSI参考模型，是ISO（国际标准化组织）组织在1985年研究的网络互联模型。该体系结构标准定义了网络互连的七层框架（物理层、数据链路层、网络层、传输层、会话层、表示层和应用层），即ISO开放系统互连参考模型。第一层到第三层属于[OSI参考模型](http://baike.baidu.com/view/38361.htm)的低三层，负责创建网络通信连接的链路；第四层到第七层为OSI参考模型的高四层，具体负责端到端的数据通信。在这一框架下进一步详细规定了每一层的功能，以实现开放系统环境中的互连性、互操作性和应用的可移植性。

OSI参考模型中，对等层协议之间交换的信息单元统称为协议数据单元(PDU,Protocol Data Unit)。 　　

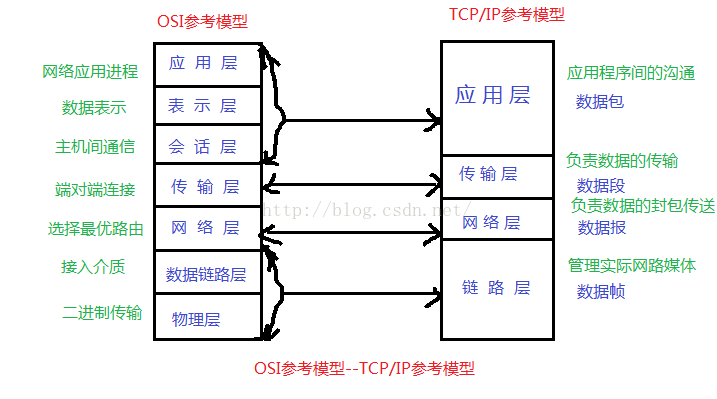
传输层——数据段（Segment） 　　网络层——分组（数据包）（Packet） 　　数据链路层——数据帧（Frame） 　　物理层——比特（Bit）

**2、TCP/IP参考模型：**

        TCP/IP参考模型是计算机网络的祖父ARPANET和其后继的因特网使用的参考模型。ARPANET是由美国国防部DoD（U.S.Department of Defense）赞助的研究网络。逐渐地它通过租用的电话线连结了数百所大学和政府部门。当无线网络和卫星出现以后，现有的协议在和它们相连的时候出现了问题，所以需要一种新的参考体系结构。这个体系结构在它的两个主要协议出现以后，被称为TCP/IP参考模型（TCP/IP reference model）。

       TCP/IP是一组协议的代名词，它还包括许多协议，组成了TCP/IP协议簇。TCP/IP协议簇分为四层（从下到上分为第一、二、三、四层的），IP位于协议簇的第二层(对应OSI的第三层：网络层)，TCP位于协议簇的第三层(对应OSI的第四层：传输层)。TCP/IP通讯协议采用了4层的层级结构，每一层都呼叫它的下一层所提供的网络来完成自己的需求。这4层分别为：

应用层：应用程序间沟通的层，如简单电子邮件传输（SMTP）、文件传输协议（FTP）、网络远程访问协议（Telnet）等。   
传输层：在此层中，它提供了节点间的数据传送服务，如传输控制协议（TCP）、用户数据报协议（UDP）等，TCP和UDP给数据包加入传输数据并把它传输到下一层中，这一层负责传送数据，并且确定数据已被送达并接收。   
互连网络层：负责提供基本的数据封包传送功能，让每一块数据包都能够到达目的主机（但不检查是否被正确接收），如网际协议（IP）。   
网络接口层：对实际的网络媒体的管理，定义如何使用实际网络（如Ethernet、Serial Line等）来传送数据。



       OSI七层模型结构的协议体系结构结构完整，概念明确，但其只能算是理想的一种体系结构，在实际应用中显得发杂，定义的层数越多，实现的难度也就随之越大，所以TCP/IP的四层模型结构成了至今被大量运用的网络体系的模型结构，但这并不意味着OSI就被搁置一旁甚至弃而不用。

3、两种模型结构的比较：

共同点：

1）、OSI参考模型和TCP/IP参考模型都采用了层次结构的概念。

2）、都能够提供面向连接和无连接两种通信服务机制。

不同点：

1）、OSI采用的七层模型，而TCP/IP是四层结构。

2）、TCP/IP参考模型的网络接口层实际上并没有真正的定义，只是一些概念性的描述。而OSI参考模型不仅分了两层，而且每一层的功能都很详尽，            甚至在数据链路层又分出一个介质访问子层，专门解决局域网的共享介质问题。

3）、OSI模型是在协议开发前设计的，具有通用性。TCP/IP是先有协议集然后建立模型，不适用于非TCP/IP网络。

4）、OSI参考模型与TCP/IP参考模型的传输层功能基本相似，都是负责为用户提供真正的端对端的通信服务，也对高层屏蔽了底层网络的实现细节。            所不同的是TCP/IP参考模型的传输层是建立在网络互联层基础之上的，而网络互联层只提供无连接的网络服务，所以面向连接的功能完全在TCP          协议中实现，当然TCP/IP的传输层还提供无连接的服务，如UDP；相反OSI参考模型的传输层是建立在网络层基础之上的，网络层既提供面向连          接的服务，又提供无连接的服务，但传输层只提供面向连接的服务。

5）、OSI参考模型的抽象能力高，适合与描述各种网络；而TCP/IP是先有了协议，才制定TCP/IP模型的。

6）、OSI参考模型的概念划分清晰，但过于复杂；而TCP/IP参考模型在服务、接口和协议的 区别上不清楚，功能描述和实现细节混在一起。

7）、TCP/IP参考模型的网络接口层并不是真正的一层；OSI参考模型的缺点是层次过多，划分意义不大但增加了复杂性。

8）、OSI参考模型虽然被看好，由于技术不成熟，实现困难；相反，TCP/IP参考模型虽然有许多不尽人意的地方，但还是比较成功的。



       关于TCP/IP的四层模型结构，也可以是五层结构，即在最底层加上了物理层，因为物理层注重的是信号的传递以及具有某些电机特性，所以严格的来讲，应该将其加上，放在网络接口层的下层位置。

**4、关于各层间的作用(基于TCP/IP)：**

1）、物理层：

        指的是电信号传递方式，比如现在以太网通用的网线(双绞线)、早期以太网采用的同轴电缆(现在主要用于有线电视)、光纤等都属于物理层的概念。 物理层的能力决定了最大传输速率、传输距离、抗干扰性等。集线器(Hub)是工作在物理层的网络设备，用于双绞线的连接和信号中继(将已衰减的信号再次放大使之传得更远)。

2）、链路层：

        链路层有以太网、令牌环网等标准。链路层负责网卡设备的驱动、帧同步(就是说从网线上检测到什么信号算作新帧的开始)、冲突检测(如果检测到冲突就自动重发)、 数据差错校验等工作。交换机是工作在链路层的网络设备，可以在不同的链路层网络之间转发数据帧(比如十兆以太网和百兆以太网之间、以太网和令牌环网之间)，由于不同链路层的帧格式不同，交换机要将进来的数据包拆掉链路层首部重新封装之后再转发。

       网络层的IP协议是构成Internet的基础。Internet上的主机通过IP地址来标识，Internet上有大量路由器负责根据IP地址选择合适的路径转发数据包，数据包从Internet上的源主机到目的主机往往要经过十多个路由器。路由器是工作在第三层的网络设备，同时兼有交换机的功能，可以在不同的链路层接口之间转发数据包。因此路由器需要将进来的数据包拆掉网络层和链路层两层首部并重新封装。IP 协议不保证传输的可靠性,数据包在传输过程中可能丢失，可靠性可以在上层协议或应用程序中提供支持。

3）、网络层：

         负责点到点的传输(这里的“点”指主机或路由器)，而传输层负责端到端 (end-to-end)的传输(这里的“端”指源主机和目的主机)。

4）、传输层：

       该层定义了两个重要的协议，即传输控制协议（TCP）或用户数据报协议（UDP）。TCP是一种面向连接的、可靠的协议。就像打电话，双方拿起电话互通身份之后就建立了连接，然后说话就行了，这边说的话那边保证听得到，并且是按说话的顺序听到的，说完话挂机断开连接。也就是说TCP传输的双方需要首先建立连接，之后由TCP协议保证数据收发的可靠性,丢失的数据包自动重发，上层应用程序收到的总是可靠的数据流,通讯之后关闭连接。UDP协议不面向连接，也不保证可靠性，有点像寄信，写好信放到邮筒里，既不能保证信件在邮递过程中不会丢失，也不能保证信件是按顺序寄到目的地的。使用UDP协议的应用程序需要自己完成丢包重发、消息排序等工作。

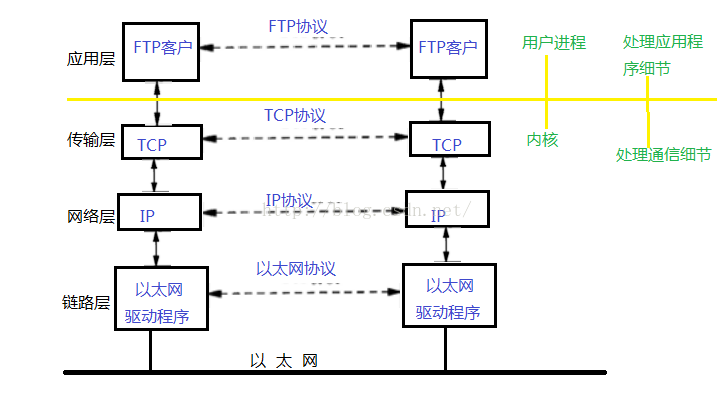
5）、应用层：

         为用户提供各种所需的服务。比如：FTP，DNS，Telnet，SMTP等。

**5、TCP/IP协议栈与数据封装**

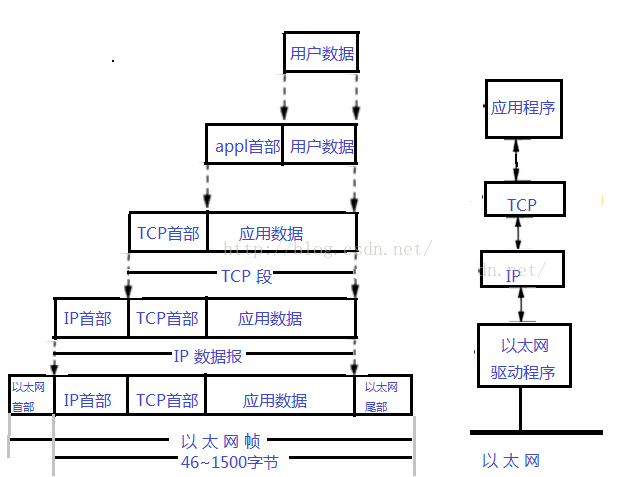
       TCP/IP网络协议栈分为应用层(Application)、传输层(Transport)、网络层(Network)和链路层(Link)四层。

      关于两台计算机通过TCP/IP协议通讯的过程（局域网的两个主机可以通信）见下图：



      只有应用层由用户提供，传输层及其以下的机制由内核提供。应用程序对通讯数据的含义进行解释，而传输层及其以下的机制处理通讯的细节，将数据从一台计算机通过一定的路径发送到另一台计算机。

      应用层数据通过协议栈发到网络上时，每层协议都要加上一个数据首部(header)，称为封装(Encapsulation)，如下图所示:



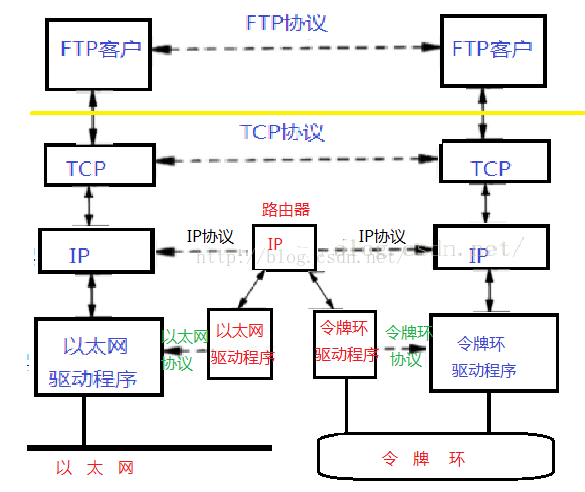
                                                                            TCP/IP数据包的封装

过程：

       应用层将数据通过TCP/IP协议栈逐层向下传递，其下的每层接到来自上层的数据时，根据每层的协议都要在其数据的前端添加首部信息进行封装。不同的协议层对数据包有不同的称谓，在传输层叫做段(segment)，在网络层叫做数据报(datagram),在链路层叫做帧(frame)。在经过链路层时，数据封装成帧后发给物理层的传输介质上，到达目标主机后，每层协议再逐层剥掉其首部，最后递交给目标主机应用层，进行应用程序处理，数据与源主机发送的数据一致。

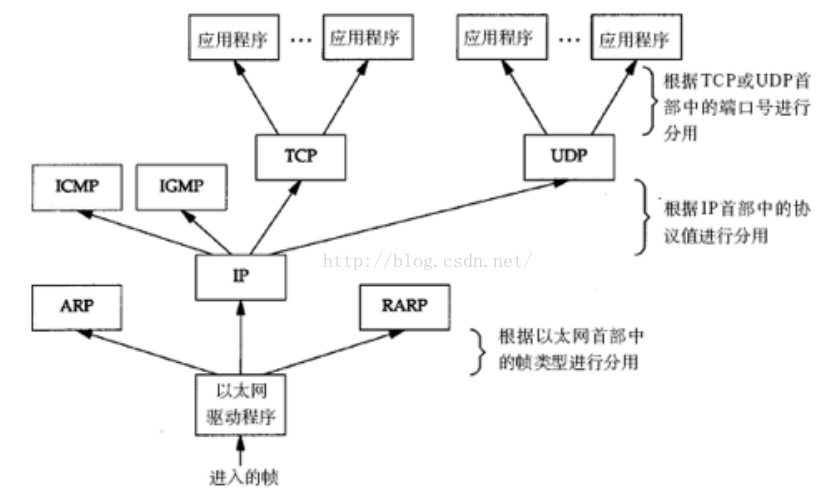
       每层都将数据包分为：有效头部信息和有效载荷（数据）。对于有效数据称之为“有效载荷”，添加的部分称其为“报头”。下层收到来自上层交付的数据时，对于下层来说并不关心其具体内容，因为站在当前层的角度来看，此时已添加上层报头信息的数据整个都被视为有效载荷，当前层要做的只是继续封装该数据，即添加自己的报头。而解包的过程亦是如此，每层只负责剥离自己能识别的报头信息。任何数据报中的报头信息必须包含"上层协议"，即要把当前的有效载荷交付给上层的谁。

上图适应两台计算机在同一网段的情况，如果是不同网段，从源主机到目标主机间的数据传输则须经过一个或多个路由器，以实现数据的跨网络传输。



        路由器是工作在第三层的网络设备，同时兼有交换机的功能,可以在不同的链路层接口之间转发数据包。因此路由器需要将进来的数据包拆掉网络层和链路层两层首部并重新封装。在此过程中，仍认为在同一层进行了网络连接。

       目的主机收到数据包后，如何经过各层协议栈最后到达应用程序呢？整个过程如下图所示



1、以太网驱动程序根据以太网首部中的“上层协议”字段确定该数据帧的有效载荷(指除去协议首部之外实际传输的数据)是IP、ARP还是RARP协议的数据报。

2、交给相应的协议处理。假如是IP数据报，IP协议再根据IP首部中的“上层协议”字段确定该数据报的有效载荷是TCP、UDP、ICMP还是IGMP ，然后交给相应的协议处理。假如是TCP段或UDP段，TCP或UDP协议再根据TCP首部或UDP首部的“端口号”字段确定应该将应用层数据交给哪个用户进程。

     IP 地址是标识网络中特定主机的地址，而端口号就是同一台主机上标识特定进程的地址，IP 地址和端口号合起来标识网络中唯一的进程。  
注意：

       虽然IP、ARP和RARP数据报都需要以太网驱动程序来封装成帧，但是从功能上划分，ARP和RARP属于链路层，IP属于网络层。虽然ICMP、IGMP、TCP、UDP的数据都需要IP协议来封装成数据报，但是从功能上划分，ICMP、IGMP与IP同属于网络层，TCP和UDP属于传输层。

**6、以太网帧格式**

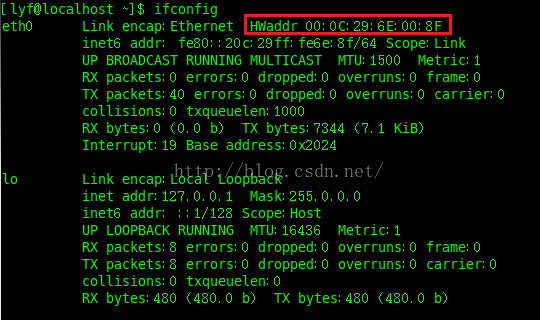


其中的源地址和目的地址是指网卡的硬件地址(也叫MAC地址),长度是48位（12个字节）,是在网卡出厂时固化的。

帧协议类型字段有三种值，分别对应IP、ARP、RARP，而后是有效载荷的大小（数据长度）。帧末尾是CRC校验码，用于差错校验的。

帧协议其类型值分别为0800的IP协议，0806的ARP协议，0835的RARP协议。

用ifconfig命令看一下MAC地址，“HWaddr 00:1C:29:6E:00:8F”部分就是硬件地址。

。

     以太网帧中的数据长度规定最小46字节,最大1500字节，ARP和RARP数据包的长度不够46字节，要在后面补填充位。

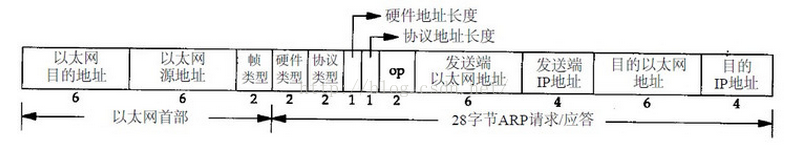
     最大值1500称为以太网的最大传输单元(MTU)，不同的网络类型有不同的MTU，如果一个数据包从以太网路由到拨号链路上，数据包长度大于拨号链路的MTU了，则需要对数据包进行分片(fragmentation)。ifconfig命令的输出中也有“MTU:1500”。注意，MTU这个概念指数据帧中有效载荷的最大长度，不包括帧首部的长度。

7、ARP协议

**概念：**ARP协议原名为“地址解析协议”，是根据IP地址获取物理地址的一个TCP/IP协议。主机发送信息时将包含目标IP地址的ARP请求广播到网络上的所有主机，并接收返回消息，以此确定目标的物理地址。收到返回消息后将该IP地址和物理地址存入本机ARP缓存中并保留一定时间，下次请求时直接查询ARP缓存以节约资源。

**目的：**在网络通讯时,源主机的应用程序知道目的主机的IP地址和端口号,却不知道目的主机的硬件地址,而数据包首先是被网卡接收到再去处理上层协议的,如果接收到的数据包的硬件地址与本机不符,则直接丢弃。因此在通讯前必须获得目的主机的硬件地址，ARP协议就起到这个作用。

       每台主机都维护一个ARP缓存表,可以用arp -a命令查看。缓存表中的表项有过期时间(一般为20分钟),如果20分钟内没有再次使用某个表项,则该表项失效,下次还要发ARP请求来获得目的主机的硬件地址。



      源MAC地址、目的MAC地址在以太网首部和ARP请求各出现一次,对于链路层为以太网的情况是多余的,但如果链路层是其它类型的网络则有可能是必要的。帧协议类型字段有三种值，分别对应IP、ARP、RARP。硬件类型指链路层网络类型,1为以太网。协议类型指要转换的地址类型，0x0800为IP地址。后面两个地址长度对于以太网地址和IP地址分别为6和4(字节)，即0604。op字段：op字段为1表示ARP请求,op字段为2表示ARP应答。

例如：A主机（源地址）向B主机（目的地址）进行ARP请求，A的MAC为：00:05:5d:61:58:a8；A的IP地址：c0 a8 00 37(192.168.0.55)。

          B的MAC为：00:05:5d:a1:b8:40；B的IP地址：c0 a8 00 02(192.168.0.2).

请求帧为

以太网首部(14字节)  ff ff ff ff ff 00 05 5d 61 58 a8 08 06           //广播地址的目标地址是全1。08 06：ARP帧

APR帧(28字节)

00 01                          //硬件类型表示为以太网；硬件地址(MAC地址)长度为6,协议地址(IP地址)长度为4（0604）

08 00 **06 04** 00 01 00 05 5d 61 58 a8 c0 a8 00 37    //协议类型0800表示为IP地址；op为0x0001表示请求；

00 00 00 00 00 00 c0  a8 00 02   //此处目的以太网地址为全0，IP地址为B的。A主机ARP的目的就是获得B主机硬件地址（MAC地址/物理地址）。

填充位（18字节）

应答帧为：

以太网首部 （字节） 00 05 5d 61 58 a8 00 05 5d a1 b8 40 08 06

APR帧(28字节)

00 01

08 00 **06 04** 00 02 00 05 5d a1 b8 40 c0 a8 00 02   //op为0x0002表示应答；

00 05 5d 61 58 a8 c0  a8 00 37  //目的以太网地址和IP地址为A的

填充位（18字节）

8、RARP协议



RARP协议

       原名为“反向地址解析协议”，允许局域网的物理机器从网关服务器的ARP表或者缓存上请求其IP地址。网络管理员在局域网网关路由器里创建一个表以映射物理地址（MAC）和与其对应的IP地址。当设置一台新的机器时，其RARP客户机程序需要向路由器上的 RARP 服务器请求相应的IP地址。假设在路由表中已经设置了一个记录，RARP 服务器将会返回 IP 地址给机器，此机器就会存储起来以便日后使用。

产生原因

     ARP（地址解析协议）是设备通过自己知道的IP地址来获得自己不知道的物理地址的协议。假如一个设备不知道它自己的IP地址，但是知道自己的物理地址，设备知道的只是网络接口卡上的物理地址时，又该怎么办呢？ 为解决这种情况，就有了与ARP对应的RARP协议。

工作方式

      RARP以与ARP相反的方式工作。RARP发出要反向解析的物理地址并希望返回其对应的IP地址，应答包括由能够提供所需信息的RARP服务器发出的IP地址。虽然发送方发出的是广播信息，RARP规定只有RARP服务器能产生应答。许多网络指定多个RARP服务器，这样做既是为了平衡负载也是为了作为出现问题时的备份。

9、IP协议（重点）

       IP数据报的格式，见下图：



       IP数据报的首部长度和数据长度都是可变长的,但总是4字节的整数倍。对于IPv4 ,4位版本字段是4。4位首部长度的数值是以4字节为单位的,最小值为5,也就是首部长度最小是4x5=20字节, 也就是不带任何选项的IP首部,4位能表示的最大值是15,也就是首部长度最大是60字节。8位TOS字段有3个位用来指定IP数据报的优先级(目前已经废弃不用),还有4个位表示可选的服务类型(最小延迟、最大呑吐量、最大可靠性、最小成本),还有一个位总是0。

       16位总长度是整个数据报(包括IP首部和IP层payload)的字节数。16位标识：每传一个IP数据报,16位的标识加1,可用于分片和重新组装数据报。

       3位标志和13位片偏移用于分片。

       TTL(Time to live)是这样用的:源主机为数据包设定一个生存时间,比如64,每过一个路由器就把该值减1,如果减到0就表示路由已经太长了仍然找不到目的主机的网络,就丢弃该包,因此这个生存时间的单位不是秒,而是跳(hop)。协议字段指示上层协议是TCP、UDP、ICMP还是IGMP。然后是校验和,只校验IP首部,数据的校验由更高层协议负责。IPv4的IP地址长度为32位。

下面主要进行说明IP分片和重组

3个标志位：

1）、最高位必须为0，该位必须复制到所有分组中。

2）、不分片位（no not fragment，DF位）。DF=0表示可以分片；DF=1表示接受主机对分组不能分片。如果长度超过MTU(最大传输单元)而又不可以分片的话，那么只能丢弃该组，并且用ICMP差错报文向源主机报告。

3）、分片位（More fragment,MF位）。该位表示是否是最后一个分片，MF=1表示接收的分组不是最后一片，否则表示是最后一片。

**IP如何分片：**

1）、检查DF标志位，检查是否允许分片。如果设置了该位，则数据报将被丢弃，并将一个ICMP错误返回给源主机。

2）、基于MTU，确定片长度，将数据长度分为若干部分。（除了最后的数据部分，所有新建的数据选项的长度必须是8个字节的整数倍。）

3）、每个数据部分被放入一个IP数据报，这些数的数据报的报头略微修改了原来的报文头。

4）、除了最后的数据报分片外，每个分片都设置了多个分片标志位。

5）、每个分片的偏移量字段设置为这个数据部分在原来的数据报所占的位置，这个位置相当于原来未分片的数据报的开头处。

6）、如果在原来的数据报中包括了选项，则选项类型字节的高位字节决定了这个信息是被复制到所有的分片数据段，还是只复制到了第一个数据报。

7）、设置新数据报的报文头字段及总长度字段。

8）、重新计算报文头部校验和。

IP的组装：

        为了高效第组装分片，用于保存分片的数据结构必须做到：

1）、快速定位属于某一数据报的一组分组。

2）、在属于某一数据报的一组分片中快速插入新的分片。

3）、有效地判断一个数据报的所有分片是否已经全部接收。

4）、具有组装超时机制，如果在重组完成之前定时器溢出，则删除该数据报的所有内容。

关于IP协议常见的问题：  
1、为什么IP不是可靠连接的协议  
     IP层没有超时和重传机制，没有连接握手协议。  
2、IP数据报经过路由器都做了哪些改动  
     IP没经过一个路由器，TTL数就会减一，如果最后TTL为0了，但是依然没有到达目的地，这个IP包将会被丢弃。这样做了目的是为了防止出现循环回路。并且最后一跳的路由器会发送一个ICMP不可达的数据包给源IP。  
3、关于分片的问题  
      IP数据包的长度为16，也就是说最大为65535个字节。但是我们知道数据链路层规定MTU最大传输单元为1500个字节。当IP数据包的大小超过了1500，IP数据包必须要进行分片处理。这还只是初始发送端的分片，在网络中如果某个路由器的MTU小于1500，那么还需要在路由器端做分片，这就给IP包的完整接收照成了很大的不确定性（我们知道数据包达到终点不一定是按照顺序的），因为万一某一个分片丢失，可能会造成真个IP重传，当然前提是高层协议支持重传。对于TCP协议，不存在分片的问题，因为TCP报头的选项字段有MSS字段，规定一个TCP包最大可传输的字节数，一般是1500-20-20=1460字节。也就是说到达IP封包的时候最大可承载数据只有1460个字节，这样就可以避免分片的问题。但是对于UDP这一类的协议，分片操作还是交给IP完成，虽然这样可以增加载荷的效率，但是稳定性会受到很大的影响。

**IP地址与路由：**

IP地址分为5类：（IPv4的IP地址长度为4字节,通常采用点分〸十进制表示法）

A类 0.0.0.0到127.255.255.255

B类 128.0.0.0到191.255.255.255

C类 192.0.0.0到223.255.255.255  
D类 224.0.0.0到239.255.255.255

E类 240.0.0.0到247.255.255.255

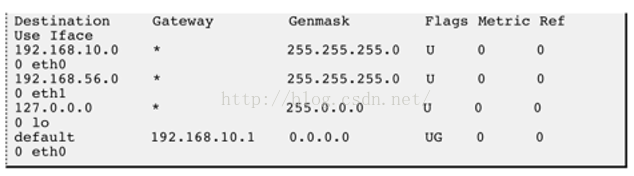
       IP地址与子网掩码做与运算可以得到网络号,主机号从全0到全1就是子网的地址范围。

       127.\*的IP地址用于本机环回(loop back)测试,通常是127.0.0.1。loopback是系统中一种特殊的网络设备,如果发送数据包的目的地址是环回地址或者与本机其它网络设备的IP地址相同，则数据包不会发送到网络介质上,而是通过环回设备再发回给上层协议和应用程序,主要用于测试。

一些不能用作主机IP 地址的特殊地址:

      目的地址为255.255.255.255 ,表示本网络内部广播,路由器不转发这样的广播数据包。

      主机号全为0的地址只表示网络而不能表示某个主机,如192.168.10.0(假设子网掩码 为255.255.255.0)。  
      目的地址的主机号为全1,表示广播至某个网络的所有主机,例如目的地址192.168.10.255表示广播至192.168.10.0网络(假设子网掩码为255.255.255.0)。



        路由表：由很多路由条目组成,每个条目都指明去往某个网络的数据包应该经由哪个接口发送,其中最后一条是缺省路由条目。路由条目路由表中的一行,每个条目主要由目的网络地址、子网掩码、下一跳地址、发送接口四部分组成,如果要发送的数据包的目的网络地址匹配路由表中的某一行,就按规定的接口发送到下一跳地址。缺省路由条目路由表中的最后一行,主要由下一跳地址和发送接口两部分组成,当目的地址与路由表中其它行都不匹配时,就按缺省路由条目规定的接口发送到下一跳地址。

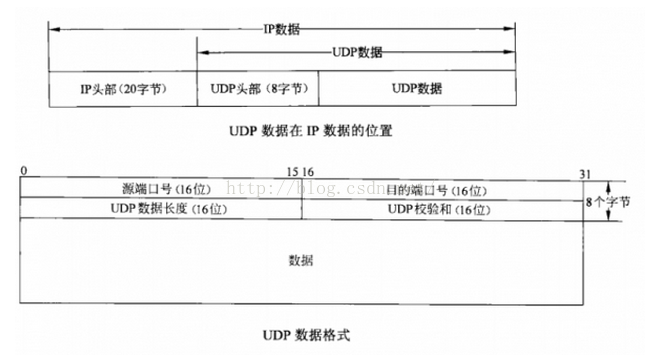
        路由是从寻找目标网络开始的，对于路由来说其找寻目标网络地址一般有三种可能：

        1、知道，但当前网段找不到（送至下一路由）；2、不知道，也找不到。（送至默认网关）；3、知道，也能在当前网段找到。

       路由如何确定要发送的网络号？

       IP数据报中存有目标IP地址，所以当路由获得该地址时，通过与子网掩码的“与”操作，便能得到一个网络号，将其与路由表中的目标地址（Destination）条目进行逐一比对，如果比对成功则通过该网络号对应的发送接口（Use Iface）发出；若未匹配到，则说明不在该局域网络，须将其从eth0的发送接口发出，将其送至默认网关处，再由其路由表决定下一跳的地址。

**10、UDP协议**



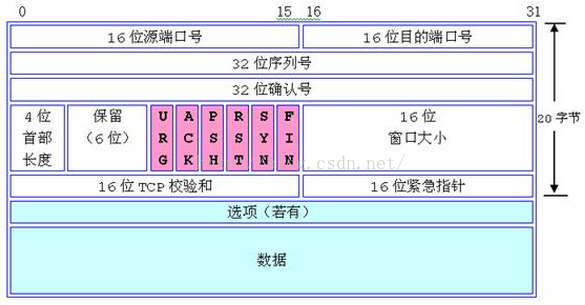
       UDP协议全称是用户数据报协议，在网络中它与TCP协议一样用于处理数据包，是一种无连接的协议。在OSI模型中，在第四层的传输层，处于IP协议的上一层。UDP有不提供数据包分组、组装和不能对数据包进行排序的缺点，也就是说，当报文发送之后，是无法得知其是否安全完整到达的。UDP用来支持那些需要在计算机之间传输数据的网络应用。包括网络视频会议系统在内的众多的客户/服务器模式的网络应用都需要使用UDP协议。UDP协议从问世至今已经被使用了很多年，虽然其最初的光彩已经被一些类似协议所掩盖，但是即使是在今天UDP仍然不失为一项非常实用和可行的网络传输层协议。

      UDP协议不面向连接,也不保证传输的可靠性。例如: 发送端的UDP协议层只管把应用层传来的数据封装成段交给IP协议层就算完成任务了,如果因为网络故障该段无法发到对方,UDP协议层也不会给应用层返回任何错误信息。接收端的UDP协议层只管把收到的数据根据端口号交给相应的应用程序就算完成任务了,如果发送端发来多个数据包并且在网络上经过不同的路由,到达接收端时顺序已经错乱了,UDP协议层也不保证按发送时的顺序交给应用层。通常接收端的UDP协议层将收到的数据放在一个固定大小的缓冲区中等待应用程序来提取和处理,如果应用程序提取和处理的速度很慢,而发送端发送的速度很快,就会丢失数据包,UDP协议层并不报告这种错误。因此,使用UDP协议的应用程序必须考虑到这些可能的问题并实现适当的解决方案,例如等待应答、超时重发、为数据包编号、流量控制等。一般使用

UDP协议的应用程序实现都比较简单,只是发送一些对可靠性要求不高的消息,而不发送大量的数据。例如,基于UDP的TFTP协议一般只用 trivial ftp TCP FTP于传送小文件,而基于TCP的协议适用于各种文件的传输。

UDP协议的特点：UDP使用底层的互联网协议来传送报文，同IP一样提供不可靠的无连接数据包传输服务。它不提供报文到达确认、排序、及流量控制等功能。

11、TCP协议（重点）



                                                                                                 TCP段格式

       这里主要说一下六个标志位，URG、ACK、PSH、RST、SYN和FIN。

       SYN表示建立连接，FIN表示关闭连接；ACK表示响应；PSH表示有 DATA数据传输；RST表示连接重置；URG表示紧急指针字段有效。

       SYN、FIN、ACK位：其中ACK是可能与SYN，FIN等同时使用的，比如SYN和ACK可能同时为1，它表示的就是建立连接之后的响应。如果只是单个的一个SYN，它表示的只是建立连接。TCP的几次握手就是通过这样的ACK表现出来的。但SYN与FIN是不会同时为1的，因为前者表示的是建立连接，而后者表示的是断开连接。

        RST位：RST一般是在FIN之后才会出现为1的情况，表示的是连接重置。一般地，当出现FIN包或RST包时，我们便认为客户端与服务器端断开了连接；而当出现SYN和SYN＋ACK包时，我们认为客户端与服务器建立了一个连接。

        PSH位（Push）：PSH为1的情况，一般只出现在 DATA内容不为0的包中，也就是说PSH为1表示的是有真正的TCP数据包内容被传递。TCP的连接建立和连接关闭，都是通过请求－响应的模式完成的。当两个应用进程进行交互式的通信中，有时在一端的应用程序希望在键入一个命令后立即收到对应的响应。在这种情况下，TCP就可以使用推送操作。通常的数据中都会带有PSH，但URG只在紧急数据才设置，也称“带外数据”。

       URG位（URGent）：当URG=1时，表示紧急指针字段有效。他告诉系统次报文段有紧急指针，应该尽快的处理（相当于高优先级的数据），而不要按照原来的排序序列来传送。若不使用紧急指针，那么这两个字符将存储在接收TCP的缓存末尾。只有在所有数据段被处理完毕后这两个字符才能被交付到接收方的网应用进程。URG是一个正偏移，与TCP首部中序号字段的值相加表示紧急数据后面的字节，即紧急指针是指向紧急数据最后一个字节的下一字节。

TCP协议

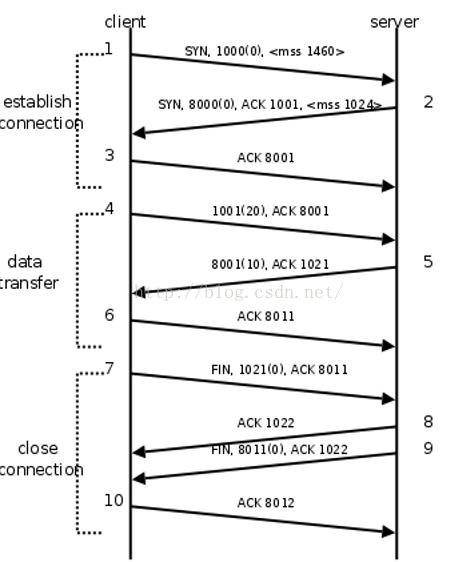
 1）、应用层向TCP层发送用于网间传输的、用8位字节表示的数据流，然后TCP把数据流分区成适当长度的报文段（通常受该计算机连接的网络的数据链路层的最大传输单元（MTU）的限制）。

 2）、TCP把结果包传给IP层，由它来通过网络将包传送给接收端实体的TCP层。TCP为了保证不发生丢包，就给每个包一个序号，同时序号也保证了传送到接收端实体的包的按序接收。

 3）、接收端实体对已成功收到的包发回一个相应的确认（ACK）。如果发送端实体在合理的往返时延（RTT）内未收到确认，那么对应的数据包就被假设为已丢失将会被进行重传。TCP用一个校验和函数来检验数据是否有错误，在发送和接收时都要计算校验和。

       TCP（Transmission Control Protocol 传输控制协议）是一种面向连接的、可靠的、基于字节流的传输层通信协议，由IETF的RFC793定义。在简化的计算机网络OSI模型中，它完成第四层传输层所指定的功能，用户数据报协议（UDP）是同一层内另一个重要的传输协议。在因特网协议族（Internet protocol suite）中，TCP层是位于IP层之上，应用层之下的中间层。不同主机的应用层之间经常需要可靠的、像管道一样的连接，但是IP层不提供这样的流机制，而是提供不可靠的包交换。

提起TCP，就不得不提及TCP的三次握手和四次挥手，其过程见下图用例：



      三次握手可概括为： SYN ----> SYN+ACK ----> ACK

3次握手除了进行链接以外，还会互相通知各自的最大报文段MSS的大小，以及各自的接收窗口的大小。

      四次挥手可概括为：主动请求释放，主端收到应答，对端请求释放，对端收到应答。    FIN--------->ACK-------->FIN--------->ACK

**为何要三次握手呢？两次可以吗？**

        三次握手完后的数据传输也是基于请求应答模式的，从而很好的解决了丢包问题。

1）TCP的三次握手最主要是防止已过期的连接再次传到被连接的主机。如果采用两次的话，会出现下面这种情况。

     比如是A机要连到B机，结果发送的连接信息由于某种原因没有到达B机；于是，A机又发了一次，结果这次B收到了，于是就发信息回来，两机就连接。传完东西后，断开。结果这时候，原先没有到达的连接信息突然又传到了B机，于是B机发信息给A，然后B机就以为和A连上了，这个时候B机就在等待A传东西过去。  
2）三次握手改成仅需要两次握手，死锁是可能发生  
       考虑计算机A和B之间的通信，假定B给A发送一个连接请求分组，A收到了这个分组，并发送了确认应答分组。按照两次握手的协定，A认为连接已经成功地建立了，可以开始发送数据分组。可是，B在A的应答分组在传输中被丢失的情况下，将不知道A是否已准备好，不知道A建议什么样的序列号，B甚至怀疑A是否收到自己的连接请求分组。在这种情况下，B认为连接还未建立成功，将忽略A发来的任何数据分组，只等待连接确认应答分组。而A在发出的分组超时后，重复发送同样的分组，资源会大大浪费。这样就形成了死锁。

**为何要四次挥手呢？**

       四次挥手：连接释放，确保数据能够完成传输。在创建连接时，发送端和接收端分别建立连接并以相应的数据结构进行建立维护，在一定程度上两端处于一个对等的关系。当一放释放连接时，并不能保证对端数据是否发送完毕，所以需要进行释放连接的请求，等待对端确认，所以一来一回，共四次。TCP协议是面向字节流的，是一种流式服务，全双工的。本端发送和接受数据两不误。

        挥手需要双发都同意才能结束。即关闭连接时，当收到对方的FIN报文通知时，它仅仅表示对方没有数据发送给你了；但未必你所有的数据都全部发送给对方了，所以你可以未必会马上会关闭SOCKET,也即你可能还需要发送一些数据给对方之后，再发送FIN报文给对方来表示你同意现在可以关闭连接了，所以它这里的ACK报文和FIN报文多数情况下都是分开发送的。

**关于公网地址和私网地址：**

公网地址是指能连接上互联网的地址，私网地址是局域网内部的地址。私网地址是不能穿透路由器到达互联网上的，那么局域网内部的多台电脑是如何通过同一个公网地址来链接互联网的？原理是使用了端口映射。比如局域网内部IP地址为192.1.1.10的主机中端口号为2000的应用程序要向外网发送IP报文，则IP报文在通过路由器时IP地址被替换成了路由器中能连接互联网的IP地址。然后路由器寻找映射表中还没有被使用的端口，并用这个端口替换掉IP报文中的端口号，并在映射表中作映射记录，则路由器的端口就对应上了子网中某个主机的IP地址和端口号。当路由器接收到远程主机回应的IP报文时，通过查表然后将报文中的公网IP转成对应的私网IP和对应的端口。也就是说私网中每一个主机中正在收发数据的端口都对应了路由器中的一个端口。一个C类子网最多能有255-2个主机，即使每个主机有100个端口在使用网络，那么也才占用24800个端口而已，完全不担心路由器端口不够用的情况。

**子网掩码：**

子网掩码的主要作用是为了将IP地址划分成网络地址和主机地址两部分。如果两个IP地址属于同一个子网，则它们和子网掩码相的结果相等，这样路由器在转发IP数据报时就可以进行路由选择，决定是将IP数据报发给外网还是发给子网内部的设备。

**域名系统：**

域名系统（DNS）是一个分布式的数据库，由它来提供IP地址和主机名之间的映射信息。

**封装:**

当应用程序用TCP传送数据时，数据被送入协议栈中，然后逐个通过每一层直到被当作一串比特流送入网络。其中每一层对收到的数据都要增加一些首部信息（有时还要增加尾部信息）。TCP传给IP的数据单元称作TCP报文段或简称为TCP段（TCP segment）。IP传给网络接口层的数据单元称作IP数据报(IP datagram)。通过以太网传输的比特

流称作帧(Frame)。