TCP/IP详解学习笔记2-数据链路层

数据链路层主要有三个目的：

* 为IP模块发送和 接收IP数据报。
* 为ARP模块发送ARP请求和接收ARP应答。
* 为RARP发送RARP请 求和接收RARP应答

数据链路层的协议：以太网协议，PPP协议（adsl宽带）和loopback协议

TCP/IP详解之IP协议ARP协议和RARP协议

把这三个协议放到一起学习是因为这三个协议处于同一层，ARP协议用来找到目标主机的Ethernet网卡Mac地址，IP则承载要发送的消息。数据链路层可以从ARP得到数据的传送信息，而从IP得到要传输的数据信息。

1.IP协议

　　IP协议是TCP/IP协议的核心，所有的TCP，UDP，IMCP，IGCP的数据都以IP数据格式传输。要注意的是，IP不是可靠的协议，这是说，IP协议没有提供一种数据未传达以后的处理机制--这被认为是上层协议--TCP或UDP要做的事情。所以这也就出现了TCP是一个可靠的协议，而UDP就没有那么可靠的区别。这是后话，暂且不提

　　1.1.IP协议头

　　如图所示



**序列号**：TCP序号，即本报文段所发送的数据的第一个字节的序号。

**确认号**：即希望下次收到对方传送的数据的第一个字节的序号。

**数据偏移**：指出TCP报文段的数据起始处距离TCP报文段的距离，实际就是TCP头部长度。注意，数据偏移的单位不是字节而是32bit，即4字节。TVP首部最大长度为(2^4-1)\*4=60字节。

**保留**：保留为今后使用，目前统一置为0.

**代码位**：

1、 URG：紧急比特。当URG=1时，表明紧急指针字段有效。该报文应尽快传送。而不要按原来的队列顺序来传送。

2、 ACK：确认比特。当ACK=1时确认号字段才有效，ACK=0时，表明确认号无效。

3、 PSH：推送比特：按对队传输到对方，不用待缓存填满后再提交给上层，而是立即提交。

4、 RST：复位比特。TCP连接中出现严重差错，必须立即释放并重新建立连接。也用于拒绝一个非法的报文段或拒绝打开一个连接。

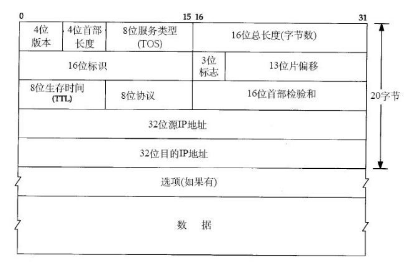
5、 SYN：同步比特。在连接建立时用来同步序号。当SYN=1而ACK=0时，表明这是一个连接请求报文段。对方若同意建立连接，则应在响应的报文段中使用SYN=1和ACK=1。因此，SYN=1时，就表明这是一个连接请求或连接接受。

6、 FIN：终止比特。用来释放一个连接。当FIN=1时，表明数据发送完毕，要求释放连接

**窗口**：接收端告知自己的接收能力，即自己接收窗口的大小，发送方将按这个大小发送数据。

**校验和**：检验的范围包括首部和数据这两部分。在计算检验和时，要在TCP报文段的前面加上12字节的伪首部。

**紧急指针**：当紧急指针代码位被设置时为有效字段。如果有效，这个值指明了当前序列号的八位组的偏移值，即第一个非紧急数据的起始位置。



TTL字段Time To Live

这个字段规定该数据包在穿过多少个路由之后才会被抛弃(这里就体现出来IP协议包的不可靠性，它不保证数据被送达)，某个ip数据包每穿过一个路由器，该数据包的TTL数值就会减少1，当该数据包的TTL成为零，它就会被自动抛弃。这个字段的最大值也就是255，也就是说一个协议包也就在路由器里面穿行255次就会被抛弃了，根据系统的不同，这个数字也不一样，一般是32或者是64，Tracerouter这个工具就是用这个原理工作的，tranceroute的-m选项要求最大值是255，也就是因为这个TTL在IP协议里面只有8bit。

现在的ip版本号是4，所以也称作IPv4。现在还有IPv6，而且运用也越来越广泛了。

1.2.IP路由选择

　　当一个IP数据包准备好了的时候，IP数据包(或者说是路由器)是如何将数据包送到目的地的呢?它是怎么选择一个合适的路径来"送货"的呢?

　　最特殊的情况是目的主机和主机直连，那么主机根本不用寻找路由，直接把数据传递过去就可以了。至于是怎么直接传递的，这就要靠ARP协议了，后面会讲到。

　　稍微一般一点的情况是，主机通过若干个路由器(router)和目的主机连接。那么路由器就要通过ip包的信息来为ip包寻找到一个合适的目标来进行传递，比如合适的主机，或者合适的路由。路由器或者主机将会用如下的方式来处理某一个IP数据包

　　如果IP数据包的TTL(生命周期)以到，则该IP数据包就被抛弃。

　　搜索路由表，优先搜索匹配主机，如果能找到和IP地址完全一致的目标主机，则将该包发向目标主机

　　搜索路由表，如果匹配主机失败，则匹配同子网的路由器，这需要“子网掩码(1.3.)”的协助。如果找到路由器，则将该包发向路由器。

　　搜索路由表，如果匹配同子网路由器失败，则匹配同网号(第一章有讲解)路由器，如果找到路由器，则将该包发向路由器。

　　搜索陆游表，如果以上都失败了，就搜索默认路由，如果默认路由存在，则发包

　　如果都失败了，就丢掉这个包。

　　这再一次证明了，ip包是不可靠的。因为它不保证送达。

**1.3.子网寻址**

IP地址的定义是网络号+主机号。但是现在所有的主机都要求子网编址，也就是说，把主机号在细分成子网号+主机号。最终一个IP地址就成为 网络号码+子网号+主机号。例如一个B类地址：210.30.109.134。一般情况下，这个IP地址的红色部分就是网络号，而蓝色部分就是子网号，绿色部分就是主机号。至于有多少位代表子网号这个问题上，这没有一个硬性的规定，取而代之的则是**子网掩码**，校园网相信大多数人都用过，在校园网的设定里面有一个255.255.255.0的东西，这就是子网掩码。子网掩码是由32bit的二进制数字序列,形式为是一连串的1和一连串的0，例如：255.255.255.0(二进制就是11111111.11111111.11111111.00000000)对于刚才的那个B类地址，因为210.30是网络号，那么后面的109.134就是子网号和主机号的组合，又因为子网掩码只有后八bit为0，所以主机号就是IP地址的后八个bit，就是134，而剩下的就是子网号码－－109。

### 2. ARP协议

还记得数据链路层的以太网的协议中，每一个数据包都有一个MAC地址头么？我们知道每一块以太网卡都有一个MAC地址，这个地址是唯一的，那么IP包是如何知道这个MAC地址的？这就是ARP协议的工作。

ARP（地址解析）协议是一种解析协议，本来主机是完全不知道这个IP对应的是哪个主机的哪个接口，当主机要发送一个IP包的时候，会首先查一下自己的ARP高速缓存（就是一个IP-MAC地址对应表缓存），如果查询的IP－MAC值对不存在，那么主机就向网络发送一个ARP协议广播包，这个广播包里面就有待查询的IP地址，而直接收到这份广播的包的所有主机都会查询自己的IP地址，如果收到广播包的某一个主机发现自己符合条件，那么就准备好一个包含自己的MAC地址的ARP包传送给发送ARP广播的主机，而广播主机拿到ARP包后会更新自己的ARP缓存（就是存放IP-MAC对应表的地方）。发送广播的主机就会用新的ARP缓存数据准备好数据链路层的的数据包发送工作。

一个典型的arp缓存信息如下，在任意一个系统里面用“arp -a”命令:

都会得到这样的结果。

  Interface: 192.168.11.3 --- 0x2  
  Internet Address      Physical Address      Type  
  192.168.11.1          00-0d-0b-43-a0-2f     dynamic  
  192.168.11.2          00-01-4a-03-5b-ea     dynamic

这样的高速缓存是有时限的，一般是20分钟（伯克利系统的衍生系统）。

[TCP/IP详解学习笔记(4)-ICMP协议，ping和Traceroute](http://blog.csdn.net/goodboy1881/article/details/670761)

### 1.IMCP协议介绍

前面讲到了，IP协议并不是一个可靠的协议，它不保证数据被送达，那么，自然的，保证数据送达的工作应该由其他的模块来完成。其中一个重要的模块就是ICMP(网络控制报文)协议。

当传送IP数据包发生错误－－比如主机不可达，路由不可达等等，ICMP协议将会把错误信息封包，然后传送回给主机。给主机一个处理错误的机会，这 也就是为什么说建立在IP层以上的协议是可能做到安全的原因。ICMP数据包由8bit的错误类型和8bit的代码和16bit的校验和组成。而前 16bit就组成了ICMP所要传递的信息。书上的图6－3清楚的给出了错误类型和代码的组合代表的意思。

尽管在大多数情况下，错误的包传送应该给出ICMP报文，但是在特殊情况下，是不产生ICMP错误报文的。如下

1. ICMP差错报文不会产生ICMP差错报文（出IMCP查询报文）（防止IMCP的无限产生和传送）
2. 目的地址是广播地址或多播地址的IP数据报。
3. 作为链路层广播的数据报。
4. 不是IP分片的第一片。
5. 源地址不是单个主机的数据报。这就是说，源地址不能为零地址、环回地址、广播地 址或多播地址。

虽然里面的一些规定现在还不是很明白，但是所有的这一切规定，都是为了防止产生ICMP报文的无限传播而定义的。

ICMP协议大致分为两类，一种是查询报文，一种是差错报文。其中查询报文有以下几种用途:

1. ping查询（不要告诉我你不知道ping程序）
2. 子网掩码查询（用于无盘工作站在初始化自身的时候初始化子网掩码）
3. 时间戳查询（可以用来同步时间）

而差错报文则产生在数据传送发生错误的时候。就不赘述了。

### 2.ICMP的应用--ping

ping可以说是ICMP的最著名的应用，当我们某一个网站上不去的时候。通常会ping一下这个网站。ping会回显出一些有用的信息。一般的信息如下:

ping这个单词源自声纳定位，而这个程序的作用也确实如此，它利用ICMP协议包来侦测另一个主机是否可达。原理是用类型码为0的ICMP发请 求，受到请求的主机则用类型码为8的ICMP回应。ping程序来计算间隔时间，并计算有多少个包被送达。用户就可以判断网络大致的情况。我们可以看到， ping给出来了传送的时间和TTL的数据。我给的例子不太好，因为走的路由少，有兴趣地可以ping一下国外的网站比如sf[**.NET**](http://lib.csdn.net/base/dotnet)，就可以观察到一些 丢包的现象，而程序运行的时间也会更加的长。  
ping还给我们一个看主机到目的主机的路由的机会。这是因为，ICMP的ping请求数据报在每经过一个路由器的时候，路由器都会把自己的ip放到该数 据报中。而目的主机则会把这个ip列表复制到回应icmp数据包中发回给主机。但是，无论如何，ip头所能纪录的路由列表是非常的有限。如果要观察路由， 我们还是需要使用更好的工具，就是要讲到的Traceroute(windows下面的名字叫做tracert)。

### 3.ICMP的应用--Traceroute

Traceroute是用来侦测主机到目的主机之间所经路由情况的重要工具，也是最便利的工具。前面说到，尽管ping工具也可以进行侦测，但是，因为ip头的限制，ping不能完全的记录下所经过的路由器。所以Traceroute正好就填补了这个缺憾。

Traceroute的原理是非常非常的有意思，它受到目的主机的IP后，首先给目的主机发送一个TTL=1（还记得TTL是什么吗？）的UDP(后面就 知道UDP是什么了)数据包，而经过的第一个路由器收到这个数据包以后，就自动把TTL减1，而TTL变为0以后，路由器就把这个包给抛弃了，并同时产生 一个主机不可达的ICMP数据报给主机。主机收到这个数据报以后再发一个TTL=2的UDP数据报给目的主机，然后刺激第二个路由器给主机发ICMP数据 报。如此往复直到到达目的主机。这样，traceroute就拿到了所有的路由器ip。从而避开了ip头只能记录有限路由IP的问题。

有人要问，我怎么知道UDP到没到达目的主机呢？这就涉及一个技巧的问题，TCP和UDP协议有一个端口号定义，而普通的网络程序只监控少数的几个号码较 小的端口，比如说80,比如说23,等等。而traceroute发送的是端口号>30000(真变态)的UDP报，所以到达目的主机的时候，目的 主机只能发送一个端口不可达的ICMP数据报给主机。主机接到这个报告以后就知道，主机到了，所以，说Traceroute是一个骗子一点也不为过:)

Traceroute程序里面提供了一些很有用的选项，甚至包含了IP选路的选项，请察看man文档来了解这些，这里就不赘述了。

为什么需要arp得到mac地址

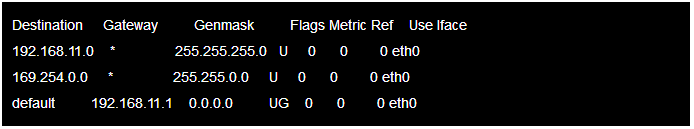
ICP/IP协议没有定义底层的协议标准，所以它需要基于某种底层协议来工作，一般来说是基于以太网协议。而以太网协议规定在网络中的标识是MAC地址，而不是IP地址，所以在数据传输时想要在以太网中把数据传输到最终的某一个具体目标，就需要这个目标在以太网中的MAC地址。而arp协议就是根据每个目标的ip地址得到对应的MAC地址的协议。

[TCP/IP详解学习笔记(5)-IP选路，动态选路，和一些细节](http://blog.csdn.net/goodboy1881/article/details/695304)

### 1.静态IP选路

#### 1.1.一个简单的路由表

选路是IP层最重要的一个功能之一。前面的部分已经简单的讲过路由器是通过何种规则来根据IP数据包的IP地址来选择路由。这里就不重复了。首先来看看一个简单的系统路由表。



对于一个给定的路由器，可以打印出五种不同的flag。

1. U表明该路由可用。
2. G表明该路由是到一个网关。如果没有这个标志，说明和Destination是直连的，而相应的Gateway应该直接给出Destination的地址。
3. H表明该路由是到一个主机，如果没有该标志，说明Destination是一个网络，换句话说Destination就应该写成一个网络号和子网号的组合，而不包括主机号(主机号码处为0)，例如 192.168.11.0
4. D表明该路由是为重定向报文创建的
5. M该路由已经被重定向报文修改

U没啥可说的，G说明这是一个网关，如果你要发数据给Destination，IP头应该写Destination的IP地址，而数据链路层的MAC地址就应该是GateWay的Mac地址了；反之，如果没有G标志，那么数据链路层和IP层的地址应该是对应的。H说明了Destination的性质，如果是H的，则说明该地址是一个完整的地址，既有网络号又有主机号，那么再匹配的时候就既要匹配网络号，又要匹配主机号；反之，Destination就代表一个网络，在匹配的时候只要匹配一下网络号就可以了。

这样，IP选路的方式就可以更加具体化了。如下

1. 首先用IP地址来匹配那些带H标志的DestinationIP地址。
2. 如果1失败就匹配那些网络地址。
3. 如果2失败就发送到Default网关

#### 1.3.ICMP的IP重定向报文和路由发现报文

当IP包在某一个地方转向的时候，都回给发送IP报的源主机一个ICMP重定向报文，而源主机就可以利用这个信息来更新自己的路由表，这样，随着网络通信的逐渐增多，路由表也就越来越完备，数据转发的速度也会越来越快。我们需要注意的是：

1. 重定向报文只能由路由器发出。
2. 重定向报文为主机所用，而不是为路由器所用。

在主机引导的时候，一般会发送在网内广播一个路由请求的ICMP报文，而多个路由器则会回应一个路由通告报文。而且，路由其本身不定期的在网络内发布路由通告报文，这样，根据这些报文，每一个主机都会有机会建立自己的路由表而实现网络通信。路由器在一份通告报文中可以通告多个地址，并且给出每一个地址的优先等级，这个优先等级是该IP作为默认路由的等级，至于怎么算的就不深究了。

路由器一般会在450-600秒的时间间隔内发布一次通告，而一个给定的通告报文的寿命是30分钟。而主机在引导的时候会每三秒发送一次请求报文，一旦接受到一个有效的通告报文，就停止发送请求报文。

在TCP/IP详解编写的时候，只有Solaris2.x支持这两种报文，大多数系统还不支持这两种报文。（后面还会讲到一些有用的路由报文）

### 动态选路协议

。。。

**TCP/IP详解学习笔记(6)-UDP协议**

### 1.UDP简要介绍

UDP是传输层协议，和TCP协议处于一个分层中，但是与TCP协议不同，UDP协议并不提供超时重传，出错重传等功能，也就是说其是不可靠的协议。

### 2.UDP协议头

#### 2.1.UDP端口号

由于很多软件需要用到UDP协议，所以UDP协议必须通过某个标志用以区分不同的程序所需要的数据包。端口号的功能就在于此，例如某一个UDP程序A在系统中注册了3000端口，那么，以后从外面传进来的目的端口号为3000的UDP包都会交给该程序。端口号理论上可以有2^16这么多。因为它的长度是16个bit

#### 2.2.UDP检验和

这是一个可选的选项，并不是所有的系统都对UDP数据包加以检验和数据(相对TCP协议的必须来说)，但是RFC中标准要求，发送端应该计算检验和。

UDP检验和覆盖UDP协议头和数据，这和IP的检验和是不同的，IP协议的检验和只是覆盖IP数据头，并不覆盖所有的数据。UDP和TCP都包含一个伪首部，这是为了计算检验和而摄制的。伪首部甚至还包含IP地址这样的IP协议里面都有的信息，目的是让UDP两次检查数据是否已经正确到达目的地。如果发送端没有打开检验和选项，而接收端计算检验和有差错，那么UDP数据将会被悄悄的丢掉（不保证送达），而不产生任何差错报文。

#### 2.3.UDP长度

UDP可以很长很长，可以有65535字节那么长。但是一般网络在传送的时候，一次一般传送不了那么长的协议（涉及到MTU的问题），就只好对数据分片，当然，这些是对UDP等上级协议透明的，UDP不需要关心IP协议层对数据如何分片，下一个章节将会稍微讨论一些分片的策略。

### 3.IP分片

IP在从上层接到数据以后，要根据IP地址来判断从那个接口发送数据（通过选路），并进行MTU的查询，如果数据大小超过MTU就进行数据分片。数据的分片是对上层和下层透明，而数据也只是到达目的地还会被重新组装，不过不用担心，IP层提供了足够的信息进行数据的再组装。

在IP头里面，16bit识别号唯一记录了一个IP包的ID,具有同一个ID的IP片将会被重新组装；而13位片偏移则记录了某IP片相对整个包的位置；而这两个表示中间的3bit标志则标示着该分片后面是否还有新的分片。这三个标示就组成了IP分片的所有信息，接受方就可以利用这些信息对IP数据进行重新组织（就算是后面的分片比前面的分片先到，这些信息也是足够了）。

因为分片技术在网络上被经常的使用，所以伪造IP分片包进行流氓攻击的软件和人也就层出不穷。

可以用Trancdroute程序来进行简单的MTU侦测。请参看教材。

### 3.UDP和ARP之间的交互式用

这是不常被人注意到的一个细节，这是针对一些系统地实现来说的。当ARP缓存还是空的时候。UDP在被发送之前一定要发送一个ARP请求来获得目的主机的MAC地址，如果这个UDP的数据包足够大，大到IP层一定要对其进行分片的时候，想象中，该UDP数据包的第一个分片会发出一个ARP查询请求，所有的分片都辉等到这个查询完成以后再发送。事实上是这样吗？

结果是，某些系统会让每一个分片都发送一个ARP查询，所有的分片都在等待，但是接受到第一个回应的时候，主机却只发送了最后一个数据片而抛弃了其他，这实在是让人匪夷所思。这样，因为分片的数据不能被及时组装，接受主机将会在一段时间内将永远无法组装的IP数据包抛弃，并且发送组装超时的ICMP报文（其实很多系统不产生这个差错），以保证接受主机自己的接收端缓存不被那些永远得不到组装的分片充满。

### 4.ICMP源站抑制差错

当目标主机的处理速度赶不上数据接收的速度，因为接受主机的IP层缓存会被占满，所以主机就会发出一个“我受不了”的一个ICMP报文。

### 5.UDP服务器设计

UDP协议的某些特性将会影响我们的服务器程序设计，大致总结如下：

1. 关于客户IP和地址：服务器必须有根据客户IP地址和端口号判断数据包是否合法的能力（这似乎要求每一个服务器都要具备）
2. 关于目的地址：服务器必须要有过滤广播地址的能力。
3. 关于数据输入：通常服务器系统的每一个端口号都会和一块输入缓冲区对应，进来的输入根据先来后到的原则等待服务器的处理，所以难免会出现缓冲区溢出的问题，这种情况下，UDP数据包可能会被丢弃，而应用服务器程序本身并不知道这个问题。
4. 服务器应该限制本地IP地址，就是说它应该可以把自己绑定到某一个网络接口的某一个端口上。