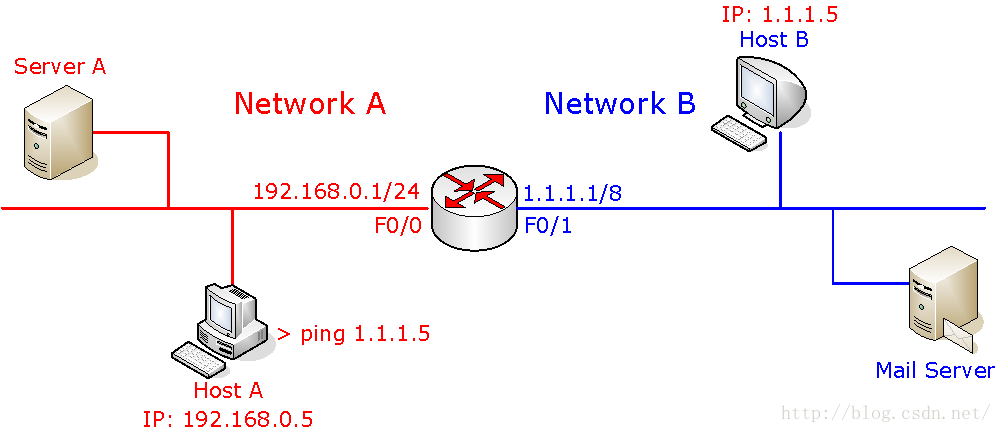
  TCP/IP协议簇是目前互联网应用最广的协议栈，谈到TCP/IP协议栈就不能不讲一讲IP路由的问题，因为在我们使用的网络通信中几乎每时每刻都在发生着IP路由的事件……。当你在网络世界中还是一位新手的时候，你也许认为设备之间实现IP路由所需的仅仅是一台设备的IP地址而已，如果你真的这样认为那就错了。每台设备在进行IP路由的时候除了必需的IP地址外还需要很多其他信息来完成数据的封装！我将通过以下网络场景提供的案例一步一步地来分析设备在进行IP路由时将会发生哪些通信事件。

  
      在这个场景中有两个网络，这两个网络通过一台具有两个以太网接口(F0/0，F0/1)的路由器连接在一起。路由器的这两个端口和你自己电脑上的RJ-45型号的网卡端口是一样的，都是内建在路由器中的。

现在我们来看一看当位于网络A的主机A想要和位于网络B的主机B进行通信时都发生了什么？

      主机A打开命令提示符(具体方法：开始——运行——cmd)，命令提示符(>)后输入ping 1.1.1.5  当“ping”这个指令被计算机执行的时候，TCP/IP协议栈会调用该协议栈互联网层的“ICMP”协议封装“ping 1.1.1.5”这个请求。

ICMP协议封装如下：

|  |  |
| --- | --- |
| ICMP报头 | (主机A) Ping 1.1.1.5 |

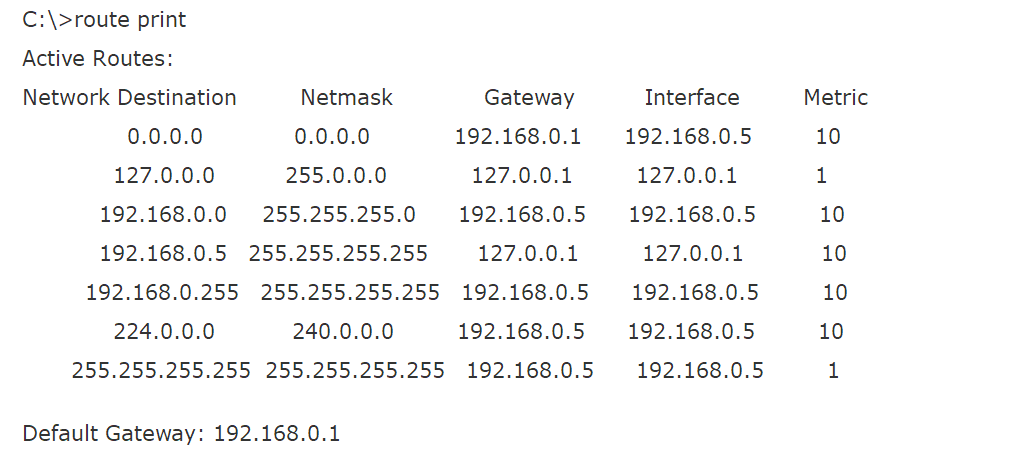
      Host A(主机A)若要将这个ICMP协议封装的请求发送给目标主机B(Host B)就必须要**使用IP协议来“运输”该请求**。这样，IP协议这个时候就充当了一个邮差的角色，将ICMP这个数据包写上发送者(主机A)的IP地址和接收者(主机B)的IP地址，然后就可以根据这个目标地址(主机B的地址)来“投递”这个数据包了。  
  
IP协议封装如下：

|  |  |  |
| --- | --- | --- |
| IP报头 | ICMP报头 | (主机A) Ping 1.1.1.5 |

      经过IP协议封装的数据包只是完成了TCP/IP协议簇网络层上的封装，接下来要**完成的是数据链路层的封装**。在进行数据链路层的信息封装之前主机A还需启动**IP路由进程来判断目标主机B在本地网络还是在远程网络**。若主机B在本地网络，说明主机B和主机A在同一个逻辑子网中，也就是同一个IP网段内。若主机B在远程网络则说明主机B与主机A不在同一个网段内，比如本例中，主机A位于网络A而主机B位于网络B。显而易见，在本例的网络场景中主机A会判断出主机B位于远程网络。

那么问题是**主机A是如何做出判断的呢**？  
      首先主机A**将自己的IP地址和子网掩码作“与运算”**。所谓与运算就是无论是二进制的“0”和“1”相与还是二进制的“0”和“0”相与，得出的结果都为“0”，而只有二进制“1”和“1”相与得出的结果为“1”。比如本例中，主机A的IP地址为192.168.0.5，将这个IP地址转换为二进制就是：11000000 10101000 00000000 00000101  
      主机A的子网掩码为255.255.255.0，将该掩码转换为二进制就是：11111111 11111111 11111111 00000000 将主机A二进制形式的IP地址和子网掩码作“与运算”的结果是：  11000000 10101000 00000000 00000000  
      为了方便查看将这个“与运算”的结果再转换为点分十进制的数就是：192.168.0.0      这样主机A就通过“与运算”完成了**IP路由判断的第一步**。

接下来的第二步是将目标主机(主机B)的IP地址与发送方(主机A)的子网掩码作“与运算”，比如本例中，主机B的IP地址为1.1.1.5，将这个IP地址转换为二进制就是：00000001 00000001 00000001 00000101  
      主机A的子网掩码为255.255.255.0将该掩码转换为二进制就是：11111111 11111111 11111111 00000000 将以上两组二进制的数进行“与运算”的结果是：00000001 00000001 00000001 00000000 将这个“与运算”的结果转换为点分十进制的数就是：1.1.1.0 这样我们就得到了**两组“与运算”**的结果，分别是**：192.168.0.0和1.1.1.0**      对于主机A来说，经过这样一番“与运算”后，**如果两个“与运算”结果相同，则说明目标主机B和自己在同一个网络内，即目标主机B在本地网络。**相反地，对于主机A来说，经过这样一番“与运算”后，如果**两个“与运算”结果不同**，则说明目标主机B和自己不在同一个网络内，即目标主机B在远程网络。在本例中，很显然**目标主机B对于发送方(主机A)来说在远程网络——也就是在网络B。**  
      主机A通过以上与目标主机B略显复杂的“与运算”后，完成了路由判断的第一步——判断出了目标主机B在远程网络。接下来主机A需要“思考”的是如何才能够到达主机B所在的网络？主机A首先会**查询自己的路由表**，看在自己的路由表中能否找到去往目标网络(网络B)的路由条目。为了方便大家理解，我将主机A的路由表输出显示如下：



      从主机A路由表的输出显示中我们没有找到与目标主机B所在的网络相匹配的具体路由，也就是说**主机A没有到达1.1.1.0这个网络的路由**。但是我们注意到在该路由表的最后一行输出的是：  Default Gateway(默认/缺省网关): 192.168.0.1 路由表最后一行的**输出说明主机A在没有找到能够到达目标网络的具体路由的情况下，会将发往目标网络的数据包发送到默认/缺省网关(192.168.0.1)，由这个默认/缺省网关再将该数据包转发到目标网络。**  
      好了，现在主机A知道了默认网关(192.168.0.1)是自己的下一跳地址，主机A开始执行封装。封装信息如下：

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 目的MAC: 下一跳设备的MAC | 源MAC: 主机A的MAC地址 | 源IP: 192.168.0.5 | 目的IP： 1.1.1.5 | ICMP报头 | (主机A) Ping 1.1.1.5 |

      在以上所封装的寻址信息当中，主机A唯一不知道的就是“下一跳设备的MAC地址”，这里的**“下一跳设备MAC地址”是指主机A的默认网关192.168.0.1的F0/0接口的MAC地址。**这里容易让人产生一个困惑，主机A的目的IP地址是主机B，那么为什么主机A封装的链路层的目的MAC地址却是路由器F0/0的MAC地址呢？原因是当初人们在设计数据链路层时主要考虑如何解决一条线路上相邻两端设备之间的通信。  
      从这个网络场景中我们看到主机A和主机B从各自所处的物理位置上看并不属于相邻的两台设备(中间隔着一台路由器)。既然主机B并不是主机A直接相邻的设备，那么主机A在数据链路层封装的目的MAC地址当然就不可能是主机B了。

**主机A通过IP路由的流程判断和查询路由表知道要想到达主机B就必须将数据包发给路由器A的F0/0接口(F0/0接口也是主机A到达主机B必经的直接相邻的接口)，因此F0/0的MAC地址就成为主机A在数据链路层封装的目的MAC地址。**在本例中假如主机A要给Server A发送数据而不是给主机B，那么主机A在数据链路层封装的目的**MAC地址就是Server A的MAC地址**，因为Server A与主机A是彼此直接相邻的设备。那么对于主机A来说它是否知道路由器F0/0接口的MAC地址呢？主机A是不知道的。所以接下来主机A需要做的是如何才能找到F0/0接口的MAC地址以完成数据链路层的封装成帧。

首先主机A**查看自己的ARP缓存表，**每台主机/电脑都有这个ARP缓存表，该**缓存表记录着与自己发生过通信的所有的直接相邻设备或主机的硬件地址(MAC地址)，ARP缓存表经过一段时间会自动删除，**比如Windows 的动态 ARP 缓存条目不超过 10 分钟就会被删除。如果主机A与路由器之前曾发生过通信，主机A自然能在ARP缓存表中找到路由器F0/0的硬件MAC地址，可是如果这是主机A与路由器的第一次通信，则**主机A会向网络A上发送ARP广播请求数据包**，该ARP请求包包含的关键信息如下：

|  |  |
| --- | --- |
| 发送方IP地址(192.168.0.5) | 发送方MAC地址(主机A MAC地址) |
| 目的IP地址(192.168.0.1) | **目的MAC地址(000000000000)** |

      以上ARP请求包中封装的三项信息都是已知的，只有“目的MAC地址”这一信息是未知的(全“0”填充)。当该ARP请求包发送到网络A上时，网络A上的设备接收该ARP包并查看该ARP包内的具体封装信息。因为该ARP包中封装的目的IP地址是192.168.0.1，**所以只有具有192.168.0.1这个IP的接口或设备才应答该ARP查询请求，应答者会将自己的硬件MAC地址封装到这个ARP应答数据包中。**

这个ARP应答数据包(本例中由路由器F0/0接口发出ARP应答)封装的关键信息如下：

|  |  |
| --- | --- |
| 发送方IP地址(192.168.0.1) | 发送方MAC地址(F0/0接口MAC地址) |
| 目的IP地址(192.168.0.5) | 目的MAC地址(主机A的MAC地址) |

      主机A通过ARP协议解析出路由器F0/0接口的MAC地址从而完成了数据链路层的封装。通过主机A这个解析过程大家可以看到主机A在第一次与默认网关通信时的确是费了一番周折，为了找到默认网关F0/0接口的MAC地址也“耽搁”了一些时间，这也是为什么我们在第一次ping网络上某台主机/设备的时候**通常会看到第一个数据包会出现“request timed out!”(请求超时！)**的情况。最后主机A完成**MAC地址解析后将该帧发送到网络A上**，该帧的具体封装如下：

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 目的MAC: 路由器F0/0接口 | 源MAC: 主机A的MAC地址 | 源IP: 192.168.0.5 | 目的IP： 1.1.1.5 | ICMP报头 | (主机A) Ping 1.1.1.5 |

      因为该帧的目的MAC地址是F0/0的MAC地址，**所以只有路由器的F0/0接口接收并处理该数据帧。**路由器将该帧解封装并查看网络层的封装信息，路由器看到该数据包网络层封装的目的地址是1.1.1.5。路由器为了将这个数据包转发到1.1.1.5，使用前面提到的“与运算”的方法进行计算，依据计算结果判断目标地址1.1.1.5对于路由器自己来说在本地网络还是在远程网络。路由器所做“与运算”过程简述如下：

      路由器的F0/0接口上的IP地址192.168.0.1与F0/0接口的子网掩码255.255.255.0作“与运算”得到“结果1”；目标地址1.1.1.5与路由器F0/0接口的子网掩码255.255.255.0作“与运算”得到“结果2”；“结果1”与“结果2”并不相同，证明目标地址**1.1.1.5并不处在路由器的F0/0接口所在的网络A**。  
      路由器的另一个接口F0/1的IP地址1.1.1.1与F0/1接口的子网掩码255.255.255.0作“与运算”得到“结果3”；目标地址1.1.1.5与F0/1接口上  
的子网掩码255.255.255.0作“与运算”得到“结果4”；“结果3”与“结果4”相同，**证明目标地址1.1.1.5处在路由器的F0/1接口所在的网络B**。这也就意味着**路由器B只需将发送给1.1.1.5的数据包从F0/1接口发出去就OK了**。我们将路由器的这种路由**称为直接路由。**  
      路由器得出上述结论后立刻执行封装，封装的具体信息如下：

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 目的MAC： 主机B的MAC地址 | 源MAC： 路由器F0/1接口 | 源IP地址 192.168.0.5 | 目的IP地址 1.1.1.5 | ICMP报头 | (主机A) Ping 1.1.1.5 |

      在以上路由器封装的具体信息中的**目的MAC地址是主机B的MAC地址**。

**如果路由器在ARP缓存表中并未查到主机B的硬件MAC地址，则路由器仍然会像主机A查询路由器F0/0接口硬件MAC地址一样发送ARP请求来查询主机B的MAC地址。**

**关于路由器对主机B 硬件MAC地址的查询过程在这里就不再重复了。**      这样主机B就收到了该数据帧，主机B对该帧进行解封装看到了网络层封装的目的IP地址，知道这个数据包的目的地是自己后，主机B继续对其拆封装并查看ICMP报头内的信息得出结论：这是一个请求自己进行反馈的数据包，需要自己将这条“数据包已到达主机B”的信息回馈给主机A。

**因此主机B为了将回馈信息顺利送达主机A需要经过如封装：**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 目的MAC： 路由器F0/1接口 | 源MAC： 主机B的MAC地址 | 源IP: 1.1.1.5 | 目的IP: 192.168.0.5 | ICMP报头 | 对(主机A) Ping 1.1.1.5的应答 |

      主机B在发送应答数据包时封装的目的MAC地址是路由器F0/1接口的MAC地址，这也是主机B通过路由判断和与运算得出的结论，**对于主机B来说路由器F0/1接口是它的下一跳。**      **路由器F0/1接口**在收到主机B发来的数据包后为了将其转发给主机A再次进行封装，具体的封装信息如下：

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 目的MAC： 主机A的MAC地址 | 源MAC： 路由器F0/0接口 | 源IP: 1.1.1.5 | 目的IP: 192.168.0.5 | ICMP报头 | 对(主机A) Ping 1.1.1.5的应答 |

      从主机A到路由器再到主机B执行的封装过程中我们可以看出网络层封装的信息是始终不变的：源IP是192.168.0.5 目的IP是1.1.1.5。**始终变化的是一跳一跳的硬件MAC地址。**从主机B到路由器再到主机A执行的封装过程中我们同样可以看出网络层封装的信息也是始终不变的：源IP是1.1.1.5目的IP是192.168.0.5。**始终变化的还是一跳一跳的硬件MAC地址。**

**总结一下这个流程：**

**1：A:192.168.0.5要发送到B:1.1.1.5。首先与自己的子网掩码：255.255.255.0，分别做与运算。若结果相同，则B是同一网络，则直接通过ARP协议直接查MAC地址跳转即可。否则，继续以下步骤。**

**2：首先查询A的路由表，看是否去往目标网络(网络B)的路由条目，若有，则直接申请MAC地址跳转即可，否则，继续以下步骤。**

**3：直接跳转到缺省地址：路由地址：A’:192.168.0.1。此时A需要获得A’的MAC地址，通过ARP协议，发布通知，A’收到后，返回自己的MAC地址。然后A包装后，跳转。**

**4：A’收到A的跳转后，解封，看到了B的目的IP。B与A’做与，看A’是否为B的缺省地址，若是，则请求B的MAC地址，然后再进行跳转，若不是，则看有没有另外一个接口B’：1.1.1.1的，刚好是B的缺省地址，有的话，则B’再通过查找MAC步骤，把B的MAC地址返回，再从该B’的接口跳转到B，完成，该步骤称为直接路由。否则，继续以下步骤。**

**5：通过路由各种寻址方法找到找B，重复上面找的步骤。直到一跳一跳到B所在的却省接口B’，再通过MAC找到B。**

**6：B给反应，也是类似的，相反的步骤。**

