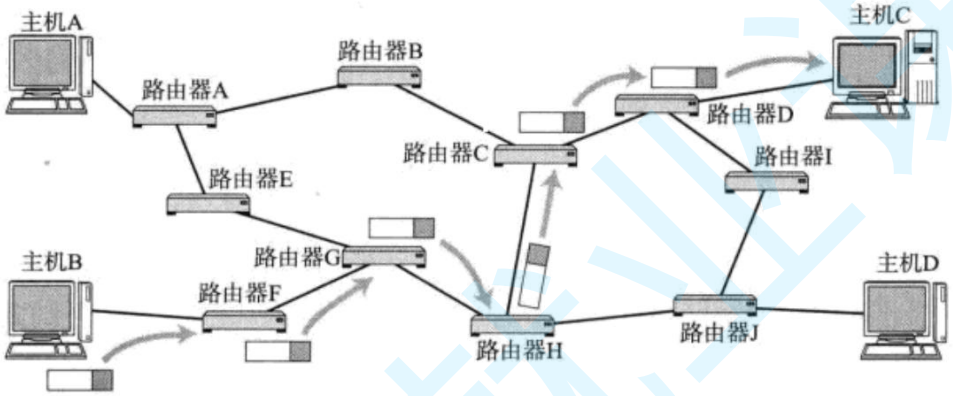
8 网络层

网络层

在复杂的网络环境中确定一个合适的路径.

IP 协议



基本概念

主机: 配有 IP 地址, 但是不进行路由控制的设备;

路由器: 即配有 IP 地址, 又能进行路由控制;

节点: 主机和路由器的统称;

协议头格式

- 4 位版本号(version): 指定 IP 协议的版本, 对于 IPv4 来说, 就是 4.
- *4 位头部长度(header length): IP 头部的长度是多少个 32bit, 也就是 length* 4 的字节数. 4bit 表示最大的数字是 15, 因此 IP 头部最大长度是 60 字节.
- 8 位服务类型(Type Of Service): 3 位优先权字段(已经弃用), 4 位 TOS 字段, 和 1 位保留字段(必须置为 0). 4 位 TOS 分别表示: 最小延时, 最大吞吐量, 最高可靠性, 最小成本. 这四者相互冲突, 只能选择一个. 对于 ssh/telnet 这样的应用程序, 最小延时比较重要; 对于 ftp 这样的程序, 最大吞吐量比较重要.
- 16 位总长度(total length): IP 数据报整体占多少个字节.
- 16 位标识(id): 唯一的标识主机发送的报文. 如果 IP 报文在数据链路层被分片了, 那么每一个片里面的这个 id 都是相同的.
- 3 位标志字段: 第一位保留(保留的意思是现在不用, 但是还没想好说不定以后要用到). 第二位置为 1 表示禁止分片, 这时候如果报文长度超过 MTU, IP 模块就会丢弃报文. 第三位表示"更多分片", 如果分片了的话, 最后一个分片置为 0, 其他是 1. 类似于一个结束标记.
- 13 位分片偏移(framegament offset): 是分片相对于原始 IP 报文开始处的偏移. 其实就是在表示当前分片在原报文中处在哪个位置. 实际偏移的字节数是这个值 8 得到的. 因此, 除了最后一个报文之外, 其他报文的长度必须是 8 的整数倍(否则报文就不连续了).
- 8 位生存时间(Time To Live, TTL): 数据报到达目的地的最大报文跳数. 一般是 64. 每次经过一个路由, TTL -= 1, 一直减到 0 还没到达, 那么就丢弃了. 这个字段主

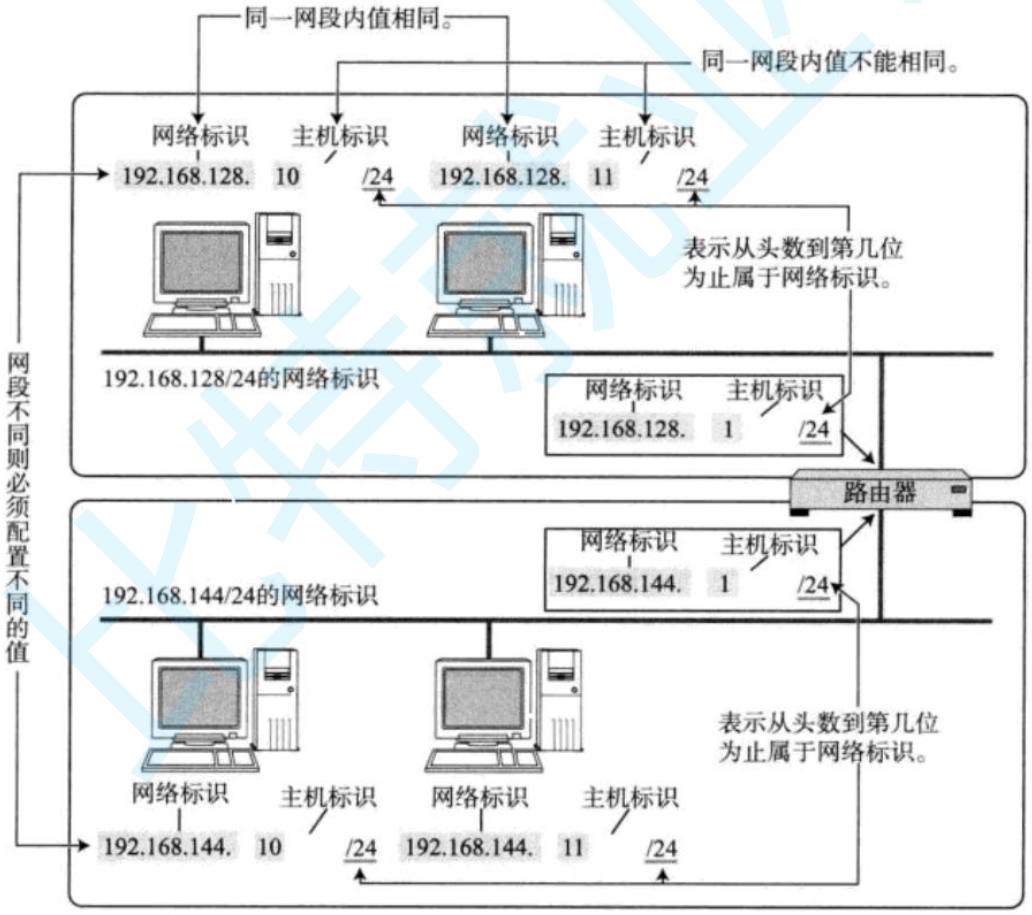
要是用来防止出现路由循环

- 8 位协议: 表示上层协议的类型
- 16 位头部校验和: 使用 CRC 进行校验, 来鉴别头部是否损坏.
- 32 位源地址和 32 位目标地址: 表示发送端和接收端.
- 选项字段(不定长, 最多 40 字节): 略

网段划分(重要)

IP 地址分为两个部分, 网络号和主机号

- 网络号: 保证相互连接的两个网段具有不同的标识;
- 主机号: 同一网段内, 主机之间具有相同的网络号, 但是必须有不同的主机号;



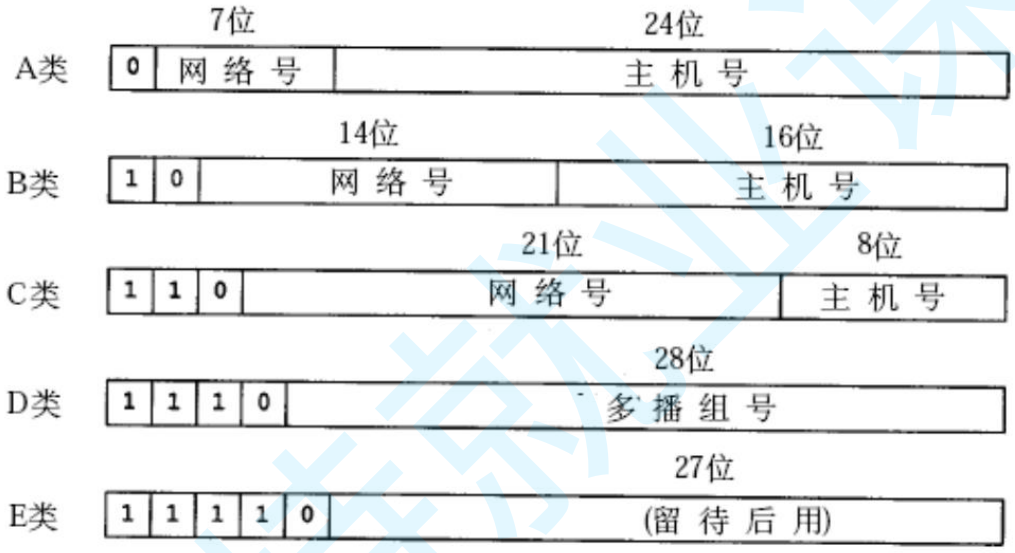
- 不同的子网其实就是把网络号相同的主机放到一起.
- 如果在子网中新增一台主机, 则这台主机的网络号和这个子网的网络号一致, 但是主机号必须不能和子网中的其他主机重复.

通过合理设置主机号和网络号, 就可以保证在相互连接的网络中, 每台主机的 IP 地址都不相同.

那么问题来了, 手动管理子网内的 IP, 是一个相当麻烦的事情.

- 有一种技术叫做 DHCP, 能够自动的给子网内新增主机节点分配 IP 地址, 避免了手动管理 IP 的不便.
- 一般的路由器都带有 DHCP 功能. 因此路由器也可以看做一个 DHCP 服务器.

过去曾经提出一种划分网络号和主机号的方案, 把所有 IP 地址分为五类, 如下图所示(该图出 自[TCPIP])。



- 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

随着 Internet 的飞速发展,这种划分方案的局限性很快显现出来,大多数组织都申请 B 类网络地址, 导致 B 类地址很快就分配完了, 而 A 类却浪费了大量地址;

- 例如, 申请了一个 B 类地址, 理论上一个子网内能允许 6 万 5 千多个主机. A 类地址的子网内的主机数更多.
- 然而实际网络架设中, 不会存在一个子网内有这么多的情况. 因此大量的 IP 地

> 址都被浪费掉了.

针对这种情况提出了新的划分方案, 称为 CIDR(Classless Interdomain Routing):

> - 引入一个额外的子网掩码(subnet mask)来区分网络号和主机号;
> - 子网掩码也是一个 32 位的正整数. 通常用一串 "0" 来结尾;
> - 将 IP 地址和子网掩码进行 "按位与" 操作, 得到的结果就是网络号;
> - 网络号和主机号的划分与这个 IP 地址是 A 类、B 类还是 C 类无关;

下面举两个例子:
划分子网的例子1

IP地址	140.252.20.68	8C FC 14 44
子网掩码	255.255.255.0	FF FF FF 00
网络号	140.252.20.0	8C FC 14 00
子网地址范围	140.252.20.0~140.252.20.255	

划分子网的例子2

IP地址	140.252.20.68	8C FC 14 44
子网掩码	255.255.255.240	FF FF FF F0
网络号	140.252.20.64	8C FC 14 40
子网地址范围	140.252.20.64~140.252.20.79	

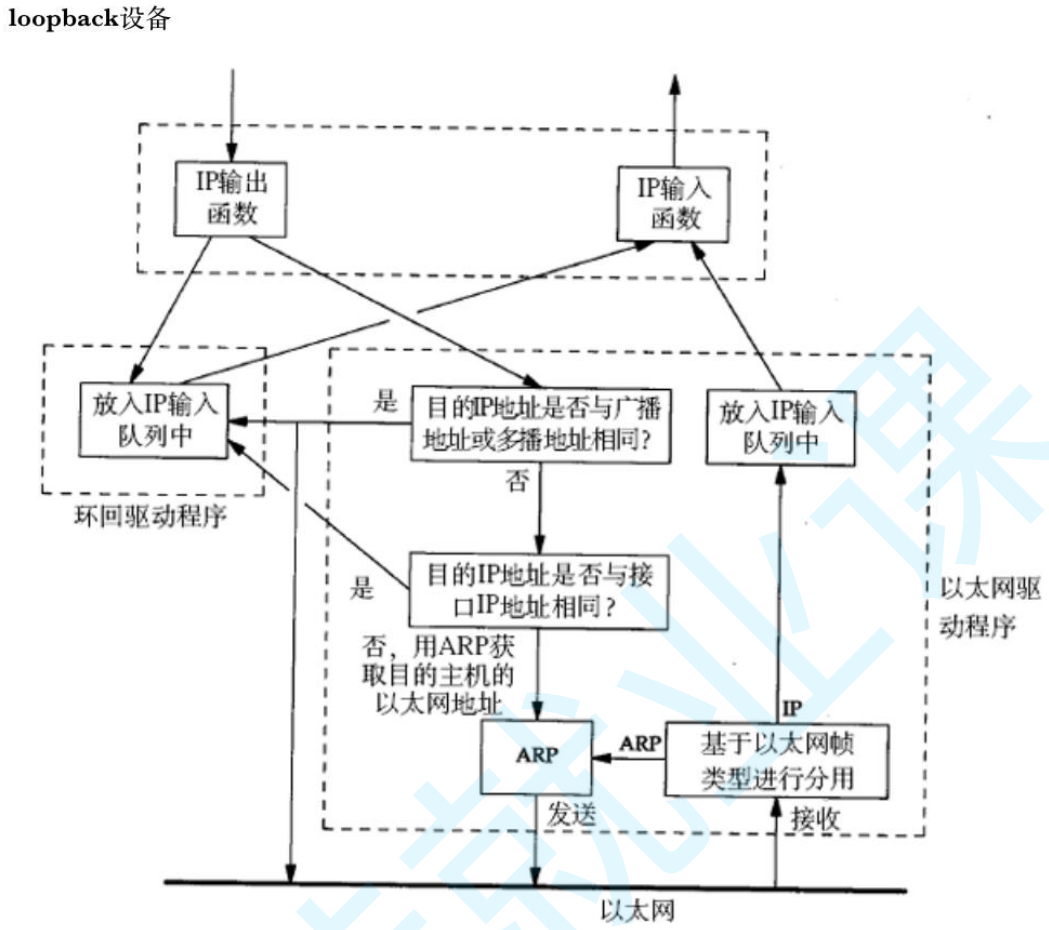
可见,IP 地址与子网掩码做与运算可以得到网络号, 主机号从全 0 到全 1 就是子网的地址范围;

IP 地址和子网掩码还有一种更简洁的表示方法,例如 140.252.20.68/24,表示 IP 地址为 140.252.20.68, 子网掩码的高 24 位是 1,也就是 255.255.255.0

特殊的 IP 地址

> - 将 IP 地址中的主机地址全部设为 0, 就成为了网络号, 代表这个局域网;
> - 将 IP 地址中的主机地址全部设为 1, 就成为了广播地址, 用于给同一个链路中相互连接的所有主机发送数据包;

- 127.*的 IP 地址用于本机环回(loop back)测试,通常是 127.0.0.1



IP 地址的数量限制

我们知道, IP 地址(IPv4)是一个 4 字节 32 位的正整数. 那么一共只有 2 的 32 次方 个 IP 地址, 大概是 43 亿左右. 而 TCP/IP 协议规定, 每个主机都需要有一个 IP 地址.

这意味着, 一共只有 43 亿台主机能接入网络么?

实际上, 由于一些特殊的 IP 地址的存在, 数量远不足 43 亿; 另外 IP 地址并非是按照主机台数来配置的, 而是每一个网卡都需要配置一个或多个 IP 地址.

CIDR 在一定程度上缓解了 IP 地址不够用的问题(提高了利用率, 减少了浪费, 但是 IP 地址的绝对上限并没有增加), 仍然不是很够用. 这时候有三种方式来解决:

- 动态分配 IP 地址: 只给接入网络的设备分配 IP 地址. 因此同一个 MAC 地址的设备, 每次接入互联网中, 得到的 IP 地址不一定是相同的;
- NAT 技术(后面会重点介绍);

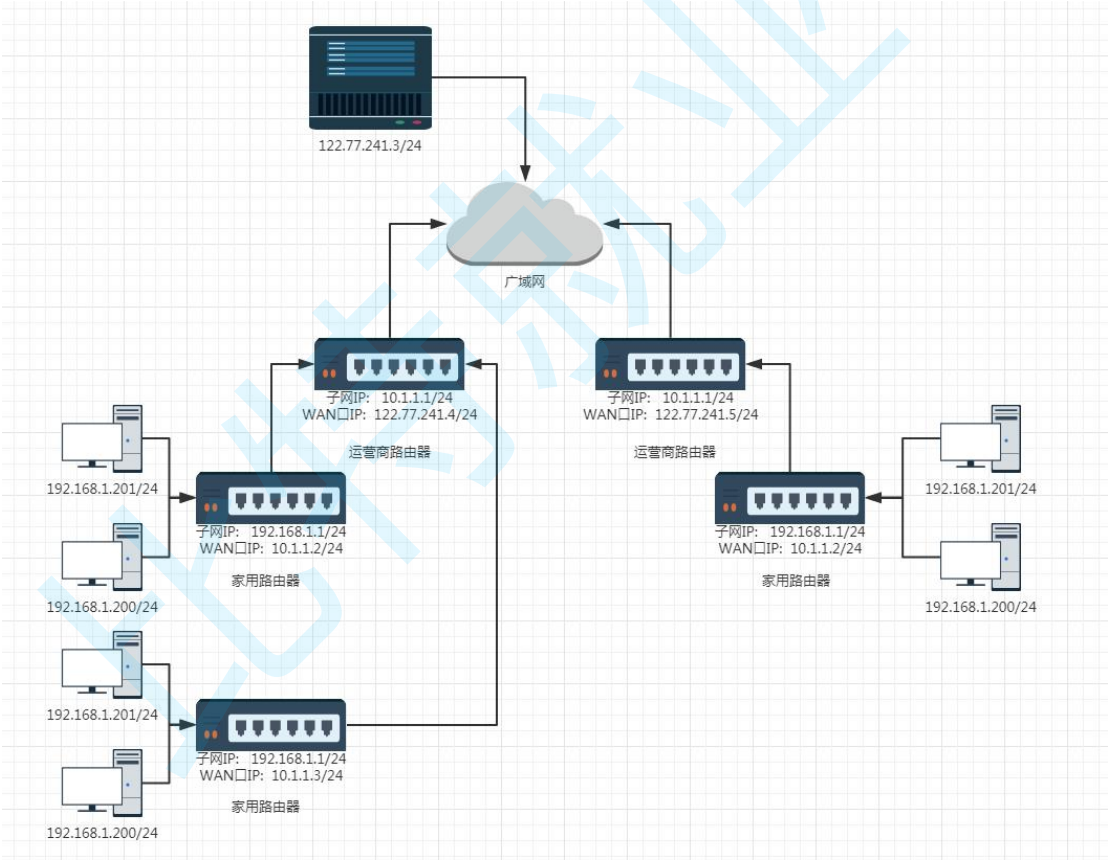
- IPv6: IPv6 并不是 IPv4 的简单升级版. 这是互不相干的两个协议, 彼此并不兼容; IPv6 用 16 字节 128 位来表示一个 IP 地址; 但是目前 IPv6 还没有普及;

私有 IP 地址和公网 IP 地址

如果一个组织内部组建局域网,IP 地址只用于局域网内的通信,而不直接连到 Internet 上,理论上 使用任意的 IP 地址都可以,但是 RFC 1918 规定了用于组建局域网的私有 IP 地址

- 10.*,前 8 位是网络号,共 16,777,216 个地址
- 172.16.*到 172.31.*,前 12 位是网络号,共 1,048,576 个地址
- 192.168.*,前 16 位是网络号,共 65,536 个地址

包含在这个范围中的, 都成为私有 IP, 其余的则称为全局 IP(或公网 IP);



- 一个路由器可以配置两个 IP 地址, 一个是 WAN 口 IP, 一个是 LAN 口 IP(子网 IP).
- 路由器 LAN 口连接的主机, 都从属于当前这个路由器的子网中.
- 不同的路由器, 子网 IP 其实都是一样的(通常都是 192.168.1.1). 子网内的主机

IP 地址不能重复. 但是子网之间的 IP 地址就可以重复了.

- 每一个家用路由器, 其实又作为运营商路由器的子网中的一个节点. 这样的运营商路由器可能会有很多级, 最外层的运营商路由器, WAN 口 IP 就是一个公网 IP 了.
- 子网内的主机需要和外网进行通信时, 路由器将 IP 首部中的 IP 地址进行替换(替换成 WAN 口 IP), 这样逐级替换, 最终数据包中的 IP 地址成为一个公网 IP. 这种技术称为 NAT(Network Address Translation, 网络地址转换).
- 如果希望我们自己实现的服务器程序, 能够在公网上被访问到, 就需要把程序部署在一台具有外网 IP 的服务器上. 这样的服务器可以在阿里云/腾讯云上进行购买.

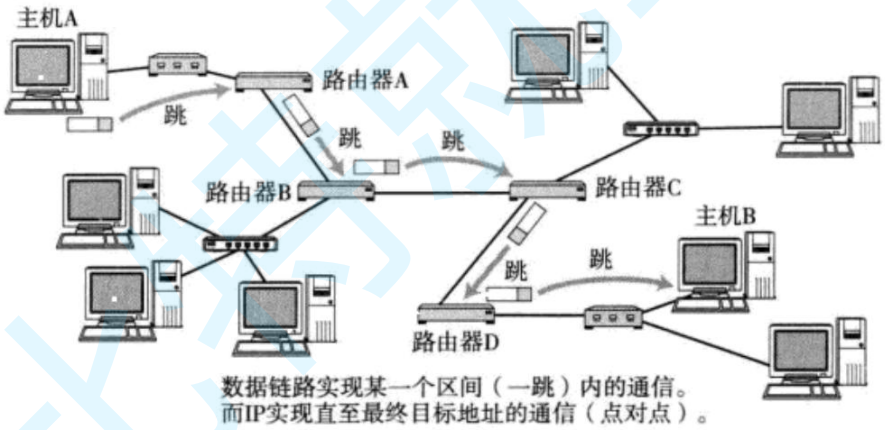
路由

在复杂的网络结构中, 找出一条通往终点的路线;

[唐僧问路例子 1]

路由的过程, 就是这样一跳一跳(Hop by Hop) "问路" 的过程.

所谓 "一跳" 就是数据链路层中的一个区间. 具体在以太网中指从源 MAC 地址到目的 MAC 地址之间的帧传输区间.

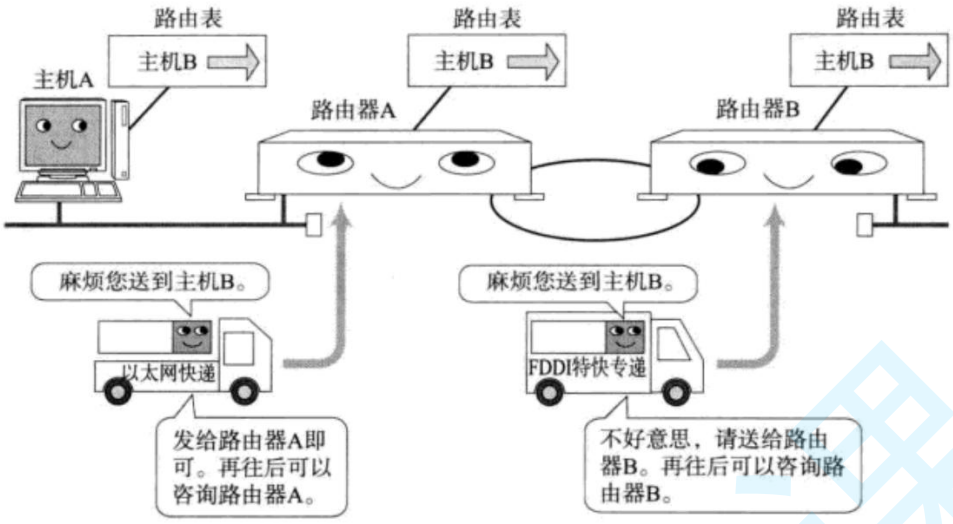


IP 数据包的传输过程也和问路一样.

- 当 IP 数据包, 到达路由器时, 路由器会先查看目的 IP;
- 路由器决定这个数据包是能直接发送给目标主机, 还是需要发送给下一个路由器;
- 依次反复, 一直到达目标 IP 地址;

那么如何判定当前这个数据包该发送到哪里呢? 这个就依靠每个节点内部维护一个路由表;

[唐僧问路例子 2]



- 路由表可以使用 route 命令查看
- 如果目的 IP 命中了路由表, 就直接转发即可;
- 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。

假设某主机上的网络接口配置和路由表如下:

```
Destination     Gateway         Genmask         Flags Metric Ref
Use Iface
192.168.10.0    *               255.255.255.0   U     0      0
0 eth0
192.168.56.0    *               255.255.255.0   U     0      0
0 eth1
127.0.0.0       *               255.0.0.0       U     0      0
0 lo
default         192.168.10.1    0.0.0.0         UG    0      0
0 eth0
```

- 这台主机有两个网络接口,一个网络接口连到 192.168.10.0/24 网络,另一个网络接口连到 192.168.56.0/24 网络;
- 路由表的 Destination 是目的网络地址,Genmask 是子网掩码,Gateway 是下一跳地址,Iface 是发送接口,Flags 中的 U 标志表示此条目有效(可以禁用某些 条目),G 标志表示此条目的下一跳地址是某个路由器的地址,没有 G 标志的条目表示目的网络地址是与本机接口直接相连的网络,不必经路由器转发;

转发过程例 1: 如果要发送的数据包的目的地址是 192.168.56.3

- 跟第一行的子网掩码做与运算得 到 192.168.56.0,与第一行的目的网络地址不

> 符
>
> - 再跟第二行的子网掩码做与运算得 到 192.168.56.0,正是第二行的目的网络地址,因此从 eth1 接口发送出去;
> - 由于 192.168.56.0/24 正 是与 eth1 接口直接相连的网络,因此可以直接发到目的主机,不需要经路由器转发;

转发过程例 2: 如果要发送的数据包的目的地址是 202.10.1.2

> - 依次和路由表前几项进行对比, 发现都不匹配;
> - 按缺省路由条目, 从 eth0 接口发出去, 发往 192.168.10.1 路由器;
> - 由 192.168.10.1 路由器根据它的路由表决定下一跳地址;

路由表生成算法(选学)

路由表可以由网络管理员手动维护(静态路由), 也可以通过一些算法自动生成(动态路由).

请同学们课后自己调研一些相关的生成算法, 例如距离向量算法, LS 算法, Dijkstra 算法等.