

网络层

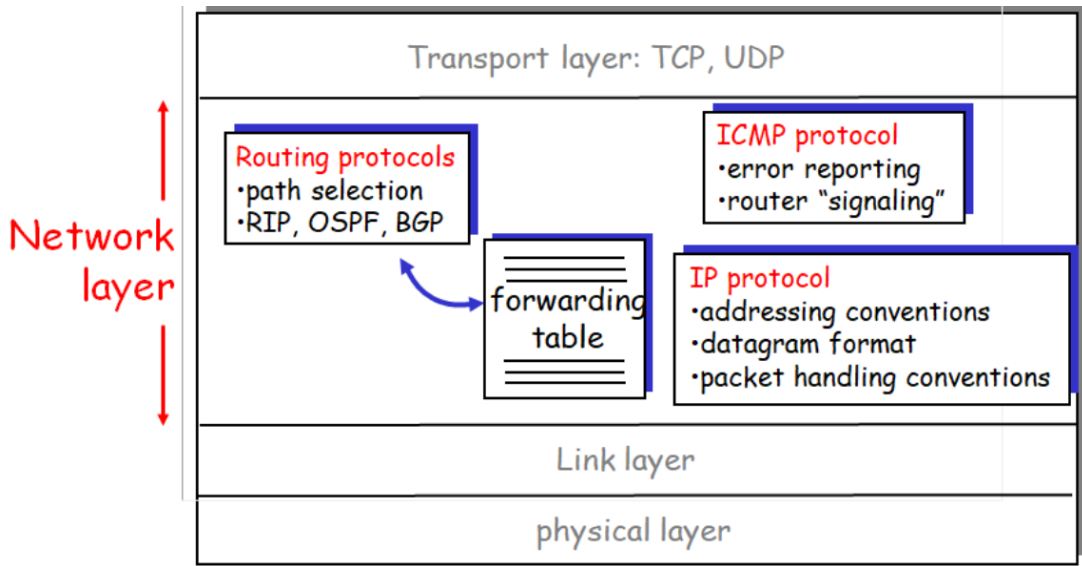
笔记源文件: [Markdown](#), [长图](#), [PDF](#), [HTML](#)

每个主机/路由器都有网络层部分, 网络层实现主机↔主机的通讯, 最复杂

注意: 笔记中所说数据报=独立的/无连接的网络通信单元

1. 网络层概述

1.0. 网络层主要协议



1 控制报文协议(ICMP): 差错报告, 检查连通性

2 IP(Internet Protocol)

3 路由算法

⚠ 网络层可分为两个子层: IP位于底下子层, ICMP位于上面子层

1.1. 网络连接

1 背景: 网络千千万, 要将其互联, 需要让路由器连接的网络共同遵守一种协议, 互联成虚拟互联网

2 虚拟互联网: 即逻辑互联网, 消除物理网络的客观异构性, 在协议层面逻辑统一

3 将网络互联所需的几种中继

层级	物理层	数据链路层	网络层	传输层
对应中继系统	集线器/中继器	网桥/交换机	路由器	网关

互联网: 用路由器进行互连的网络

⚠ 传输中检测出差错的IP数据报都被丢弃了

1.2. 网络层功能

1.2.1. 路由选择(核心)

- 1 含义：确定分组从源→目的地端到端路径
- 2 特点：时间尺度长(几秒)，软件实现
- 3 实现：基于路由选择算法构造路由表，最优化网络拓扑计算
- 4 何时进行路由选择
 - 1. 子网内部使用数据报：来一个分组就重选一次路径
 - 2. 子网内部使用虚电路：新建虚电路时才重选路径

1.2.2. 分组转发(核心)

- 1 含义：将分组从一个输入链路接口转移到适当的输出链路接口
- 2 特点：时间尺度短(几纳秒)，硬件实现，是路由器的核心功能
- 3 实现：基于路由表构造转发表，最优化查找过程，往往不区分转发表/路由表
- 4 类比：路由选择好比选择了 $A \rightarrow B$ 的一条公路(路由选择)，每条公路上都有很多立交桥(分组转发)

1.2.3. 连接建立(非核心)

- 1 路由器参与：数据报流动前，两端主机和中间路由器建立虚拟连接
- 2 网络&传输层服务：网络层是在两主机之间的，传输层是在两进程间不涉及路由器

1.3. 网络层服务模型

网络架构	服务模型	带宽	保证丢包	保证顺序	保证时序	拥塞反馈
互联网	Best Effort	无	否	否	否	否
ATM(异步)	CBR恒比特率	恒定速率	是	是	是	无拥塞
ATM	VBR变比特率	保证速率	是	是	是	无拥塞
ATM	ABR可用比特率	保证最小	否	是	否	是
ATM	UBR未指定比特率	无	否	是	否	否

1.4. 虚电路网络(VC)

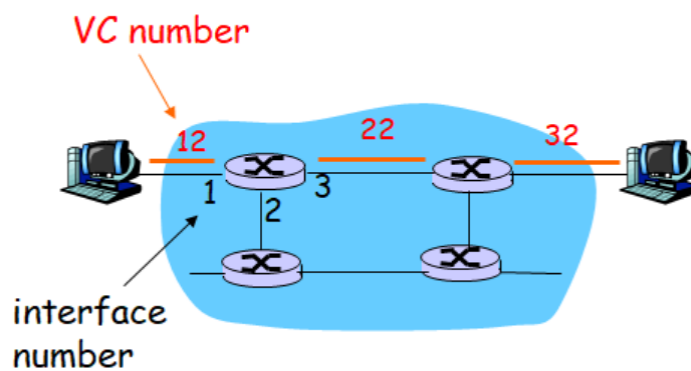
1.4.1. 虚电路的工作原理

- 1 数据传输前建立虚电路，通信完就撤销
- 2 每个包，含有一个虚电路ID
- 3 源-目的间，所有路由器维持连接状态
- 4 通信前就知道通信质量

1.4.2. 虚电路结构

- 1 发送→接收端的路径
- 2 虚电路号，每条路径都有一个号
- 3 沿路中路由的转发表项，转发表由交换设备维护，转发后虚电路号改变

虚电路/转发表实例：



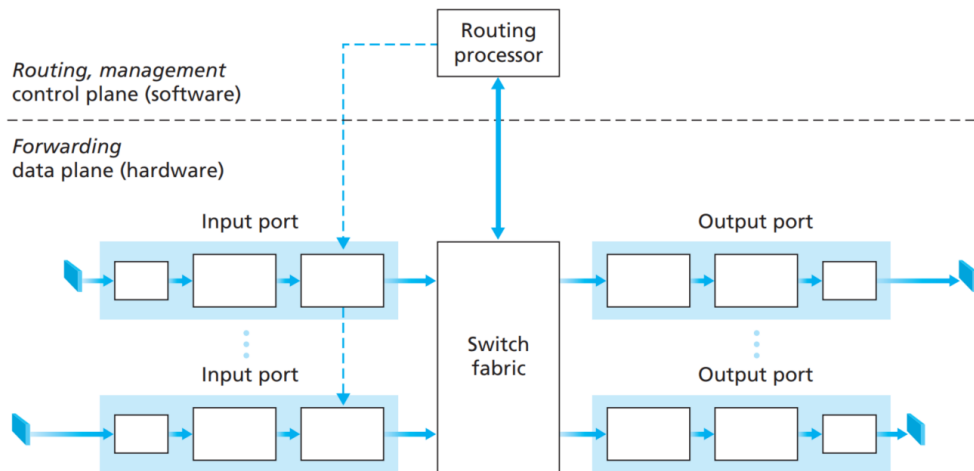
输入端口	输入VC	输出端口	输出VC
1	12	3	22
2	63	1	18
3	7	2	17
.....

1.4.3. 信令协议

- 1 作用：建立/维护/撤销VC
- 2 应用于：ATM, frame-relay, X.25, IPv6注意IPv4 中没有应用

2. 路由器

2.1. 路由器结构功能概述



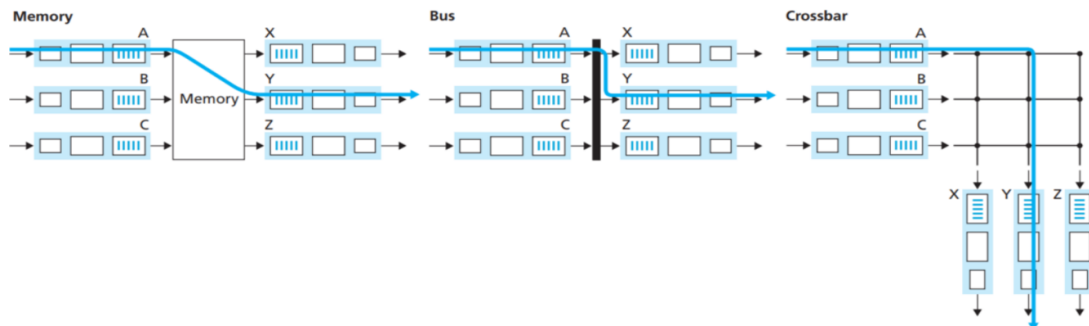
1 结构:

1. 路由选择部分
2. 分组转发部分: 输入端口, 交换结构, 输出端口

2 功能:

1. 运行路由算法/协议(RIP/OSPF/BGP), 只有边界网关路由器才运行BGP
2. 将数据包从输入链路转发到输出链路

2.2. 路由器的三种交换



1 通过存储器交换:

1. 原理: 输入端口将数据包放进内存, 输出端口从内存中取出
2. 特点: CPU控制这一过程, 两次访问存储器效率低(被内存带宽限制), 多见于早期路由器

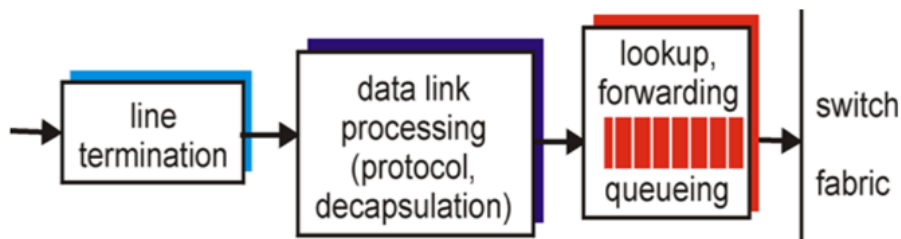
2 通过总线交换:

1. 原理: 任一输入端的数据都放总线上, 输出端从总线读数据
2. 特点: 路由性能取决于总线带宽, 可达32Gps

3 通过交叉开关网络交换: 网络横竖交叉与一个Bar, 需要通信时对应线路的Bar会导通

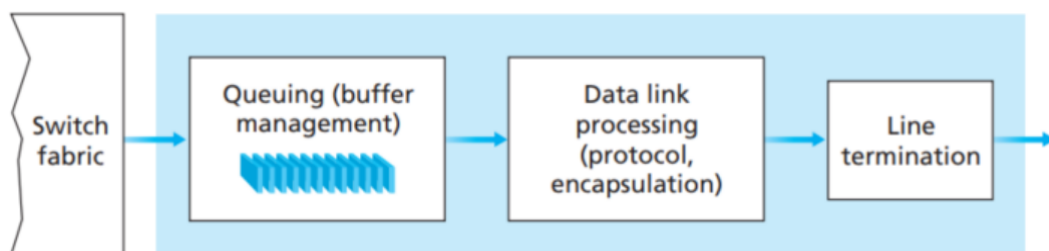
2.3. 输入输出端口

- 1 输入端口: 从左到右为物理层/链路层/网络层, 有一个输入队列



1. 输入队列：多个输入端口欲转发至同一输出端口，产生等待队列
2. 行头阻塞：等待队列满后，丢包

2 输出端口：



缓冲队列：缓冲交换结构处理速度>传输速度的那部分，转而从队列中选择数据进行传输

3. IPv4

3.1. IPv4特点

- 1 无连接性：发送数据报前，两端不必预先建立连接，数据包中含有目标IP可以自己传过去
- 2 无状态+路由选择性：路由器不需要记住经手数据包信息，只负责根据目标IP转发出去
- 3 乱序性：不同数据包会选择不同路径从发送端→接收端，不一定先发先收到，接收端需要重排

🌿乱入：转发表

<u>Prefix Match</u>			<u>Link Interface</u>
11001000	00010111	00010	0
11001000	00010111	00011000	1
11001000	00010111	00011	2
otherwise			3

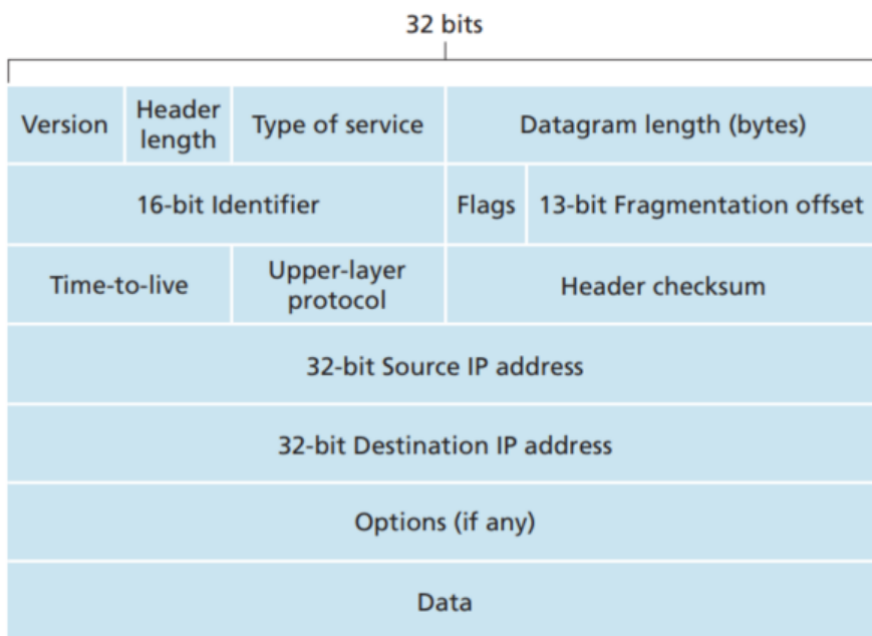
采用最长前缀原则，例如前缀为110010000001011100010时，IP的范围为

11001000000101110001000000000000

11001000000101110001011111111111

在这个范围内的IP会被转发给端口0

3.2. 数据包格式：首部+数据



Data之前都是数据包的首部，每行代表首部的一部分，每部分都是32Bit

1 第一行

1. **Version (4 bits)**: 决定是用IPv4还是IPv6
2. **Header Length (4 bits)**: IP数据包首部长
3. **Type of Service (8 bits)**: 指示数据包的服务类型，IPv6才有区别，IPv4未启用
4. **Datagram Length (16 bits)**: IP数据包的首部+数据总长

2 第二行：与IP切片有关

1. IP切片：IP数据包太大，超过链路的MTU(最大传输单元)，会被分为小片
2. 这些结构有助于切片后的IP数据包，在接收端重新组织
 - **16-bit Identifier**: 识别来自同一数据包的切片，便于在接收后组装
 - **Flags**: 控制/状态标志，是否允许切片
 - **Fragmentation Offset**: 切片在原始数据包中的位置

3 第三行

1. **Time to Live**: 限制数据包在网络中的存在时间，经过一个路由器就会减一
2. **Upper-layer Protocol**: 携带数据所用的协议(TCP/UDP)
3. **Header Checksum**: 检错，但没用

4 四五行：源IP/目标IP/额外信息

3.3. IPv4地址

3.3.1. 概述

- 1 IP地址^{一一对应}←——→网络接口：不与主机——对应，路由器有多少接口就有多少IP
- 2 IP地址组成：主机部分+网络部分，通过子网掩码分开
- 3 IP位于同一网段 ⇔ IP地址的网络部分相同

3.3.2. IP地址的特点

1 IP是分等级的地址结构

- 1. IP地址管理机构分配网络号，再由网络地址组织者分配主机号
- 2. 路由器转发分组只关注网络号，由此精简了路由表

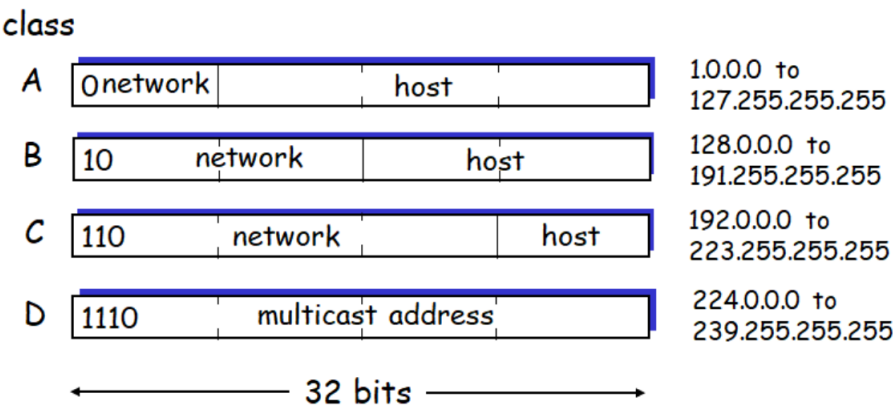
2 IP的含义

- 1. 标识一个主机/路由器的一个链路接口
- 2. 多接口主机：主机连接两个网络(两个接口)，就有两个网络号不同的IP，比如路由器的每个接口

3 用中继器/网桥连接的n个局域网，仍为一个网络，有相同的网络号

4 所有网络号对应的网络，地位平等

3.3.3. A/B/C/D类IP



类型	最高位固定值	可指派网络数	最多主机数	备注
A	0	$2^7 - 1$	$2^{24} - 2$	网络号0为保留地址，127为环回地址*
B	10	$2^{14} - 1$	$2^{16} - 2$	128.0.0.0实际上不指派
C	110	$2^{21} - 1$	$2^8 - 2$	192.0.0.0实际上不指派
D	1110	\	\	多播通信时作为目的地址**

- 1. 环回地址：用于向自己通讯
- 2. 多播通信：一对多(这个多对应IP地址)，类似于微信群聊

3.3.4. 特殊IP地址

特殊地址	结构	源or目的地址	应用
网络地址	网络号特定，主机号全为0	都不	标识整个网络

特殊地址	结构	源or目的地址	应用
直接广播地址	网络号特定, 主机号全为1	目的地址	发数据包到特定网络所有主机
受限广播地址	255.255.255.255	目的地址	发数据包到当前网络所有主机
此网络此主机	0.0.0.0	都可	主机不知自己IP时, 将其作为IP
此网络特定主机	网络号全为0, 主机号特定	目的地址	指定同一网络内的特定主机
环回地址	网络号127, 主机号全0/1	二者都为本机	本机内部通信测试

3.5. 网络地址变换(NAT): 节省IP消耗

3.5.1. 专用网

1 问题背景: 100个机构, 每个机构有100台主机, 一个机构的主机只需要内部通信(不连接Internet)

1. 每个主机分配一个共10000个IP: 显然浪费
2. 每个机构分配一个共100个专用IP: aka专用地址

2 专用地址(可重用地址)

1. 特点: 不会被路由器转发, 其作为目的地址不会被Internet传送
2. 专用地址范围
 - A类网络的一块: 10.0.0.0~10.255.255.255
 - B类网络的连续16个: 172.16.0.0~172.31.255.255
 - C类网络的连续256个: 192.168.0.0~192.168.255.255

3 专用网: 采用专用IP的互联网

4 NAT的引入: 使专用网的主机和因特网的主机通信

3.5.2. NAT概述

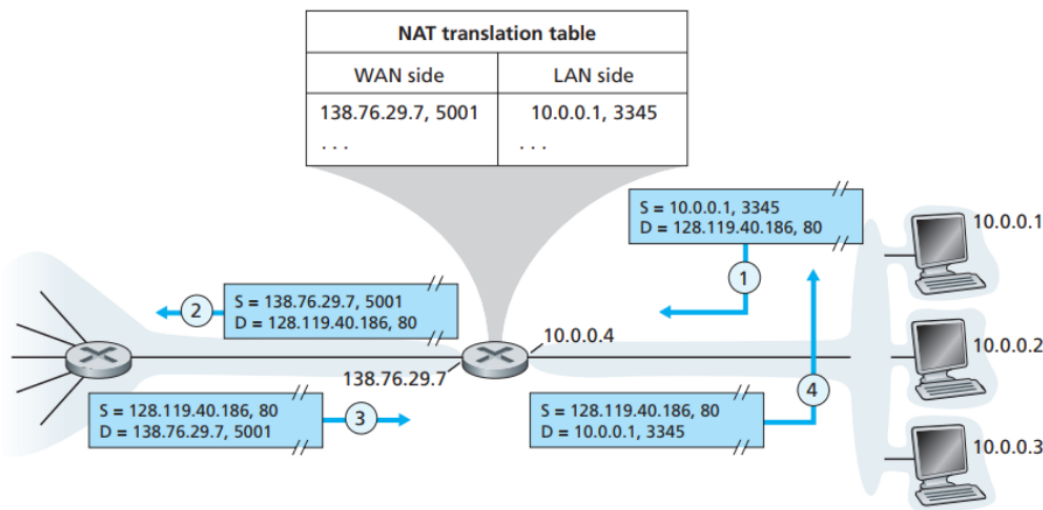
1 功能: 将专用网内部本地IP→有效外部全球IP, 使专用网络只需一个全球IP就可连接Internet

2 使用NAT: 专用网 $\xleftrightarrow{\text{NAT路由器}}$ Internet

3 NAT路由器: 装有NAT软件的路由器, 特点为

1. 至少有一个全球IP
2. 对外界隐藏了专用网内部的细节
3. 外界看来NAT路由器就是具有单一IP的单一设备

3.5.3. NAT转换表与示例



1 NAT转换表解决的问题：广域网到达NAT路由器的目标IP相同，所以要把分组转发给内部哪个主机呢

2 NAT转换表表项：端口号+IP

3 示例分析：用户处于家庭网络10.0.0.1主机

1. 用户请求128.119.40.186(端口80)的某台Web网页服务器，主机随便挑了个源端口(如3345)发出数据报
2. NAT路由器接到数据报，将源IP替换为家庭网络WAN一侧的接口IP地址138.76.29.7，选择任一个还未在NAT转换表中的源端口号(比如5001)替换原有端口号，替换后增加如下表项

NAT translation table	
WAN side	LAN side
138.76.29.7, 5001	10.0.0.1, 3345

3. Web服务器响应请求，然后返回报文，其目的地址时NAT路由器的IP，端口5001
4. 返回报文到达NAT路由器时，检索到添加过的表项WAN侧，转化回LAN侧IP，转发给家庭用户

4 特点：涉及转换，效率低

3.5.4. NAT网关穿越问题

1 静态配置NAT表：每有一个新应用就要往NAT添加新表项，由网管配置

2 动态配置NAT表：软件自动添加，借助IGD协议

3.6. 子网掩码

3.6.1. 子网划分：三级IP地址

1 格式：<网络号><子网号><主机号>

2 范围：子网划分在单位内部，单位对外仍是没划分的网络

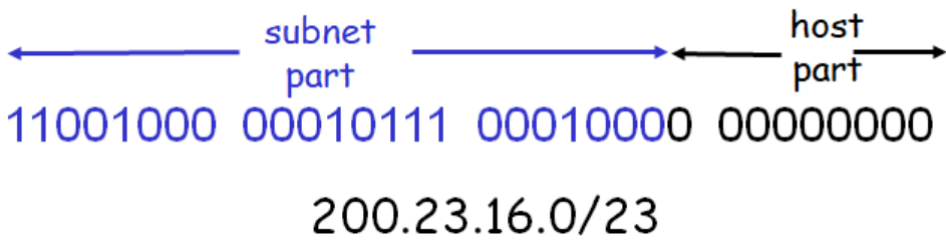
3 思路：网络号不变，主机号中借用若干bit作为子网号

3.6.2. 子网掩码

- 1 作用：告诉主机/路由器，是否对网络进行划分
- 2 结构：
 - 1. 将IP中的网络号/子网号对应位全置1，主机号对应位全置0
 - 2. IP&&子网掩码=子网地址(网络号+子网号)
- 2 默认子网掩码：对于A/B/C三类网络分别为255.0.0.0 / 255.255.0.0 / 255.255.255.0

3.6.3. 无类别域间路由选择(CIDR)

- 1 允许网管动态调整子网号/主机号长度
- 2 格式：a.b.c.d/x，x代表子网号的位数



3.7. 动态主机配置协议(DHCP)

- 1 概述：其实是一个应用层协议，DHCP报文用UDP传输，目的是给主机动态分配IP
- 2 即插即用联网+自动IP分配机制：DHCP客户端-DHCP服务器的交换过程如下

新设备(DHCP客户端)加入网络，执行以下操作

步骤	参与方	操作	说明
1	Clinet	广播 DHCP discover	申请，客户端不知道DHCP服务器在哪①，只能广播
2	Server	接收 DHCP discover	收到申请，服务器要选一个IP准备分配②
3	Server	广播 DHCP offer	发Offer，报文包含分配的IP+配置信息③
4	Clinet	接收 DHCP offer	收到Offer，客户端会决定是否接受
5	Clinet	广播 DHCP request	接受Offer，客户端请求分配Offer中的IP地址
6	Server	接收 DHCP request	收到确认
7	Server	广播 DHCP ack	你有学上了，确认分配IP地址给客户端
8	Clinet	接收 DHCP ack	我有学上了，客户端得到IP

①网络中可能不止一台DHCP服务器，广播后可能有多个服务器响应，但只有一台(一般是最先的)被选定

②DHCP服务器选择IP的方法：向数据库中搜索该设备信息

- 能找到信息，就选定找到的IP
- 找不到的话，就从IP池中选定一个IP

③配置信息有：子网掩码，默认网关，本地DN服务器

3.8. 网络控制报文协议(ICMP)

3.8.1. 路由器/主机的差错控制

- 1 检测到数据首部出错：直接弃疗吧，因为这种情况下哪个IP发来的都不可知了
- 2 检测到其他错误：
 1. 收到数据的主机/路由器：通过ICMP报文，把错误报告送回来错误数据的主机
 2. 发送数据的主机：根据ICMP报文确定错误类型，换种方式(如重新路由)把数据重发一遍

3.8.2. ICMP报文分类

3.8.2.1. ICMP差错报告报文

1 再分类

类型	路由器/主机出问题的地方	路由器/主机采取的操作
终点不可达	无法把数据交到目的地	向源点发不可达报文
源站抑制	因为拥塞丢包	向源点发源点抑制报文，源点放慢发数据报速率
时间超过	所含IP报文生命周期TTL到头	丢弃分组，向源点发超时报文①
参数问题	收到数据报首部有问题	丢掉数据报，发送参数问题报文(其实不会发)
重定向	发现有更好路由	向源点发重定向报文，源点下次用更好路由发送

①当终点在预定时间内，收不到数据报所有数据片时，也会向源点发超时报文，然后丢弃已收到的片

2 不应发送ICMP差错报告报文的几种情况

1. 当收到的报文就是ICMP差错报告报文时
2. 数据报具有组播地址
3. 数据报具有特殊地址(127.0.0.0或0.0.0.0)
4. 当报文被切片时，支队第一个分片发送差错报告

3.8.2.2. ICMP询问报文

有回送请求和回答报文/时间戳请求和回答报文/掩码地址请求和回答报文/路由器询问和通告报文

3.8.3. ICMP应用

- 1 ping: 在应用层使用了ICMP回送请求与回送回答报文, 测试两主机的连通性
- 2 tracer: 在网络层使用UDP, 跟踪分组经过的路由

4. IPv6: 彻底结局IPv4耗尽问题

4.1. IPv6特点

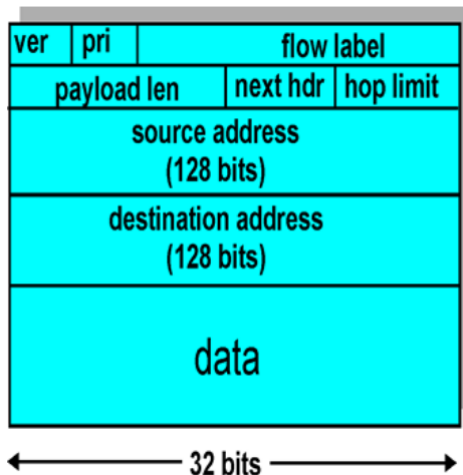
4.1.1. 基本特点

- 1 空间扩展: 长128位, 可划分层次也更多了
- 2 首部格式灵活, 首部长度必须是64bit的倍数, 且简化为了8段
- 3 允许协议扩充
- 4 支持即插即用(自动配置), 支持资源与分配
- 5 支持选项: 必要项变成可选项, 使路由器能跳过无关项

4.1.2. 兼容性

- 1 与IPv4不兼容
- 2 与其他网络协议兼容, 只需改变处理地址长即可

4.2. IPv6数据包首部: 一共40B



- 1 priority: 数据包优先级的标识符
- 2 flow label: 识别数据报是否属于相同流
- 3 next header: 下一个报头, 识别数据的上层协议

4.3. IPv6→IPv4

双栈技术&隧道技术

5. 路由算法

5.1. 路由算法概述

- 1 什么是路由算法：生成路由表的算法，路由表控制分组转发
- 2 以是否能更具信息量/拓扑自适应调整，分为：
 1. 静态/非自适应路由选择：手动搭建每条路由，多用于小网络，简单开销小
 2. 动态/自适应路由选择：多用于复杂网络，复杂开销大，又分为：
 - 链路状态路由算法(LS)：具全局性，维护一个全局的拓扑图
 - 距离-向量路由算法(DV)：具有分布性，无需维护一个全局的拓扑图
- 3 算法数据结构：用图，结点标识路由器，线表示链路

5.2. 链路状态路由算法

- 1 算法概述：主动测试所有邻结点连接状态，定期传播链路状态给其他点，是的人每个系欸但那都有完全网络拓扑信息
- 2 算法核心：Dijkstra算法，这个自不必多说
- 3 其他
 1. 算法改进：引入优先队列，复杂度可 $O(n^2) \rightarrow O(n \log n)$
 2. 算法缺点：存在震荡，会有随即延迟

5.3. 距离-向量路由算法

5.3.1. 算法特征

- 1 分布式：每个结点从邻居获得信息→计算→将结果发给邻居
- 2 迭代+自终止：重复上述过程直到邻居无更多信息要交换，算法结束
- 3 异步：所有节点迭代的步伐不必一致

5.3.2. 算法思想&伪代码

- 1 Bellman-Ford方程： $d_x(y) = \min_v \{c(x, v) + d_v(y)\}$
 1. 符号： $d_x(y)$ 是 $x \rightarrow y$ 路径最小开销， $c(x, v)$ 是 x 到其某一邻居 v 的路径
 2. 思想： $x \rightarrow y$ 的最短路径，一定要经过邻居 v 中的个，遍历所有邻居就有可能求得
- 2 结点 x 的距离向量： $D_x = [D_x(y) : y \in N]$
 1. y ：除 x 以外的其他所有节点之一
 2. $D_x(y)$ ：结点 x 到其他结点 y 的开销估计
 3. D_x ：即 $[D_x(y_1), D_x(y_2), \dots]$
- 3 结点 x 维护的信息：所有邻居的 $c(x, v)$ ，自生的距离向量 D_x ，所有邻居的距离向量 D_v
- 4 算法操作
 1. 每个节点周期性地，向每个邻居发送其距离向量副本

2. 当 x 收到 v 新的距离向量, 则用Bellman-Ford方程更新距离向量

$$D_x(y) = \min_v \{c(x, v) + D_v(y)\}, y \in N$$

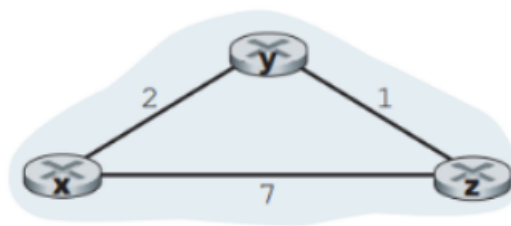
3. 如果 D_x 因此改变, 则向所有邻居立即更新其距离向量

4. 重复下去, 最终迭代得到最低路径开销

5 算法伪代码, 背下来

```
//初始化x的距离向量
for(所有其他结点y)
{
    if(y是邻居)  $D_x(y)=c(x,y)$ ;
    else  $D_x(y)=\text{infinity}$ ;
}
//将x的距离向量发给所有邻居
for(所有邻居w)
{
    将  $D_x=[D_x(y):y \text{ in } N]$  送给 w;
}
//无限循环, 处理网络变化
while(1)
{
    wait_until(x到邻居w的链路成本变化 || 收到邻居w的更新信息)
    for(所有其他结点y)
    {
         $D_x(y)=\min_v \{c(x,v)+D_v(y)\}$ ; //更新x到其余所有节点距离, 以此更新x距离
        向量
    }
    if(对任何结点y,  $D_x(y)$ 变化)
    {
        将 $D_x(y)$ 最小值发给所有邻居
    }
}
```

5.3.3. 算法示例



1 初始化: 初始化每个结点的路由选择表, 包括结点自己的距离向量, 邻居的距离向量全部设为无穷

完成后结果为图中第一列

Node x table

		cost to		
		x	y	z
from	x	0	2	7
	y	∞	∞	∞
	z	∞	∞	∞

Node y table

		cost to		
		x	y	z
from	x	∞	∞	∞
	y	2	0	1
	z	∞	∞	∞

Node z table

		cost to		
		x	y	z
from	x	∞	∞	∞
	y	∞	∞	∞
	z	7	1	0

Time

2 周期性更新+计算：完成后结果为图中第二列

1. 每个结点向邻居更新距离向量：如图中箭头的指向
2. 结点收到更新后：重新计算自身的距离向量，以 x 结点为例有

$$D_x(x) = 0$$

$$D_x(y) = \min\{c(x, y) + D_y(y), c(x, z) + D_z(y)\} = \min\{2 + 0, 7 + 1\} = 2$$

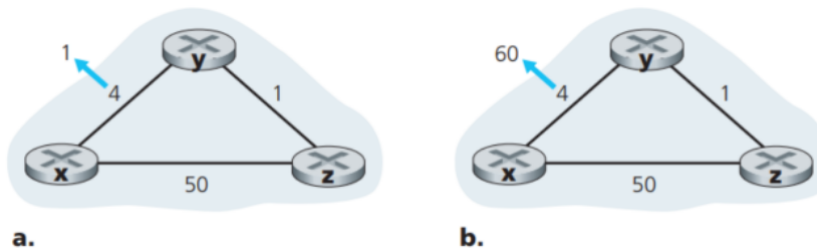
$$D_x(z) = \min\{c(x, y) + D_y(z), c(x, z) + D_z(z)\} = \min\{2 + 1, 7 + 0\} = 3$$

3 改变后更新+计算：

1. x, z 因为上一步，距离向量发生改变，所以立即让二者向邻居更新距离向量
2. 更新后再计算，算无可算算法静止，直到一条链路开销发生改变

5.3.4. 链路开销改变&链路故障

x 与邻居 v 的开销改变：更新距离，在最低开销变化时，告诉所有邻居新距离向量



1 好消息传达速度快：当 $4 \rightarrow 1$ 的变化发生时，以下时间挨个发生

1. y 检测到开销变化，更新 D_y ，向邻居更新新的距离向量
2. z 收到来自 y 的更新距离表，计算出 $z \rightarrow x$ 最小开销(从5减为2)，向邻居更新新的距离向量

3. y 收到来自 z 的更新距离表, y 开销未变所以不发送任何信息, 算法静止

2 坏消息传达速度慢: 当 $4 \rightarrow 60$ 后, 要迭代44此算法才会静止, 这样容易造成无穷计数

5.3.5. 算法改进: 增加毒性逆转

以 $z \rightarrow x$ 为例

1 操作: 当路由是 $z \rightarrow y \rightarrow x$ 时, z 就持续给 y 撒谎that有 $D_z(x) = \infty$

2 好处: 避免环路, 加快收敛

5.4. LS/DV算法比较

算法	报文多少	收敛速度	健壮性
LS	多	快	大(错误只影响单个顶点)
DV	少	慢	小(错误会影响整个网络)

6. 路由选择

6.1. 自治系统(AS): 一堆路由器聚集

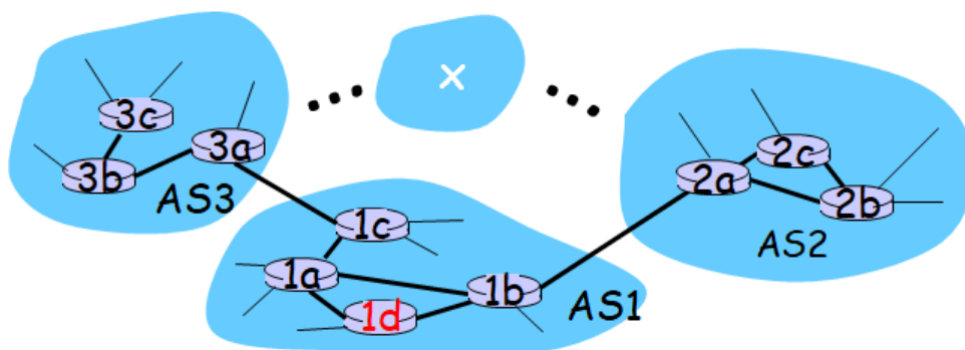
1 协议运行:

1. 同一AS内所有路由器运行相同协议, 即内部网关协议(IGP)
2. 不同AS的路由器运行不同协议
3. 所有AS运行相同的AS间路由协议, 即边界网关协议(BGP)

2 ASN: 一个自治系统由一个ASN唯一标识

3 网关路由器: 直接连接另一个AS的路由器

6.2. 网关路由选择



6.2.1. 起因

- 1 AS1中路由器收到发往其他AS的数据报
- 2 AS1搞清, 哪些路由可通过AS2/AS3访问, 此处X都可通过AS2/AS3访问
- 3 通过跨网关协议把可达性信息传给AS1内所有路由器
- 4 AS1内路由器将选一个网关路由器, 转发数据报

6.2.2. 热土豆路由选择

- 1 思想：以最小开销一股脑把分组送出AS1，至于送出AS1后分组到达目的地的成本，则完全不管
- 2 示例：上图分组发往1d后，1d就会选择将分组转给1b(近)而不是1c(远)

6.3. 路由协议


特点	RIP	OSPF	BGP
网关协议	内部	内部	外部
路由表内容	目的网络，下一跳，距离	目的网络，下一跳，距离	目的网络，完整路径
最优通路依据	跳数	费用	多种有关策略
算法	距离-矢量算法	链路状态算法	路径-矢量算法
传送方式	UDP	IP数据报	TCP连接
其他	简单，效率低	效率高，规模大	\

6.3.1. 内部网关协议IGP

- 1 路由信息协议RIP：基于DV算法(距离-向量)
 - 1. AS内部最远两点不超过15跳，直接不超过15跳，周长不超过25跳
 - 2. 每30s结点将距离向量广播给邻居
 - 3. 180s内没收到邻居的广播则认为邻居已断开，重新计算BF方程，广播链路故障信息
 - 4. 最优路径不唯一，则选其中一条
 - 5. 算法由应用层的route-d进程管理
- 2 开放最短路径优先OSPF：基于链路状态算法(Dijkstra)
 - 1. 周期性地广播自己跟谁连，代价多少，广播给整个AS，周期30min
 - 2. 运行在网络层上部，报文封装在IP报文中，是不可靠协议

6.3.2. 边界网关协议BGP

- 1 边界网关维护的不是路由表，而是路径表
- 2 BGP为每个AS提供如下方法
 - 1. 向邻居应用服务器获取子网可达性信息
 - 2. 将可达性信息传播到所有AS内部路由器
 - 3. 根据可达性信息和策略确定到AS的好路由
- 3 路由选择：优先使用本地策略，本地缺省时使用最短路径+热土豆
- 4 四种报文：
 - 1. OPEN：与相邻的BGP建立联系

- 
2. UPDATE: 发送某一路由信息
 3. KEEPALIVE: 打开报文+周期性证实邻居关系
 4. NOTIFICATION: 报告报文错误, 关闭TCP