

网络互连

主要内容

网络层提供的服务：为高层提供节点到节点的传输，经过多跳传输最终到达目的地

5.1 交换和路由：internet的工作方式（虚电路和数据报）

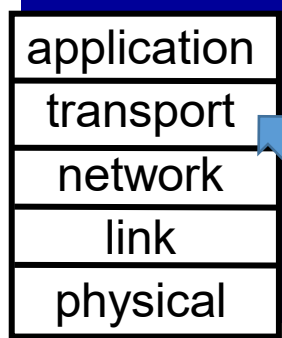
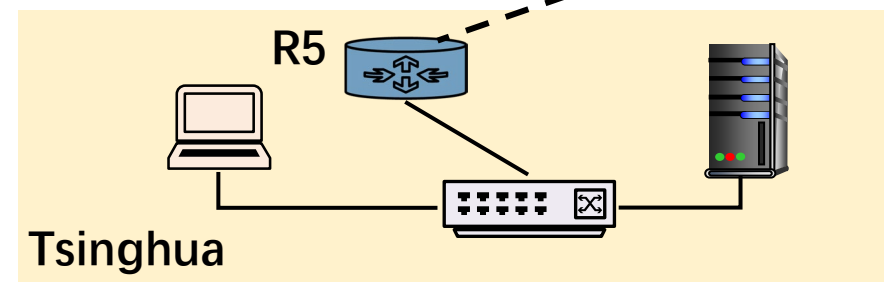
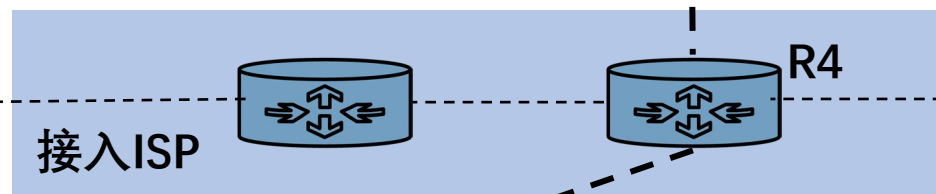
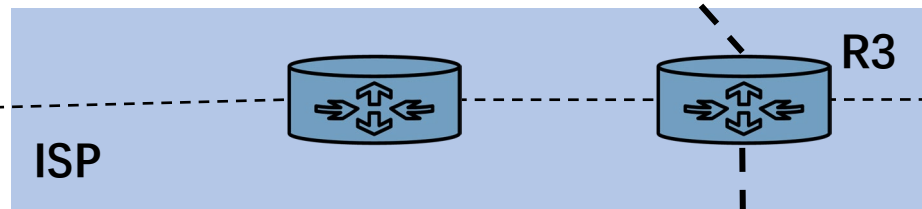
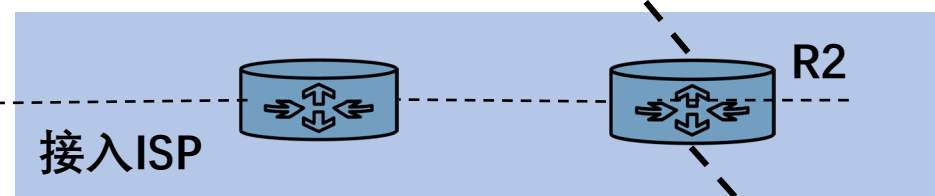
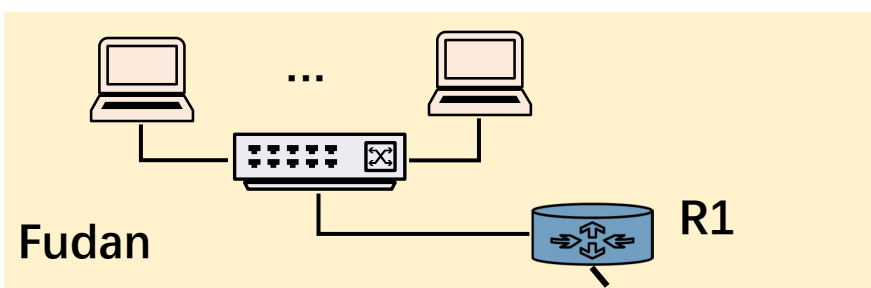
5.1 交换和路由：路由方式

- 源路由和逐跳路由
 - ~~扩散法~~
 - ~~逆向学习法~~
- IP协议
 - ARP
 - ICMP

~~5.2 网桥~~

5.3 Internet网络层

- DHCP
- NAT
- IP隧道
- IP组播
- IPv6

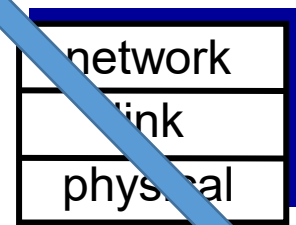
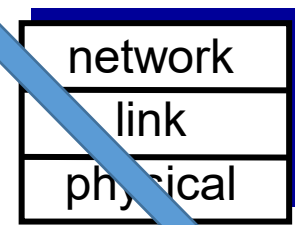


Client/Server 建立TCP连接, 交换HTTP消息: GET / ...

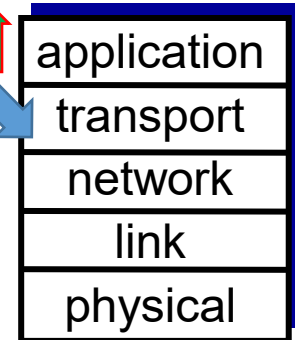
HTTP协议

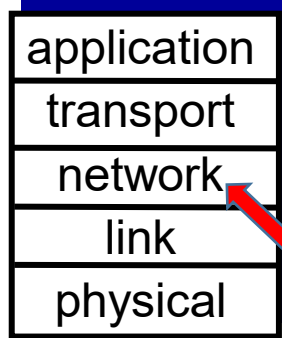
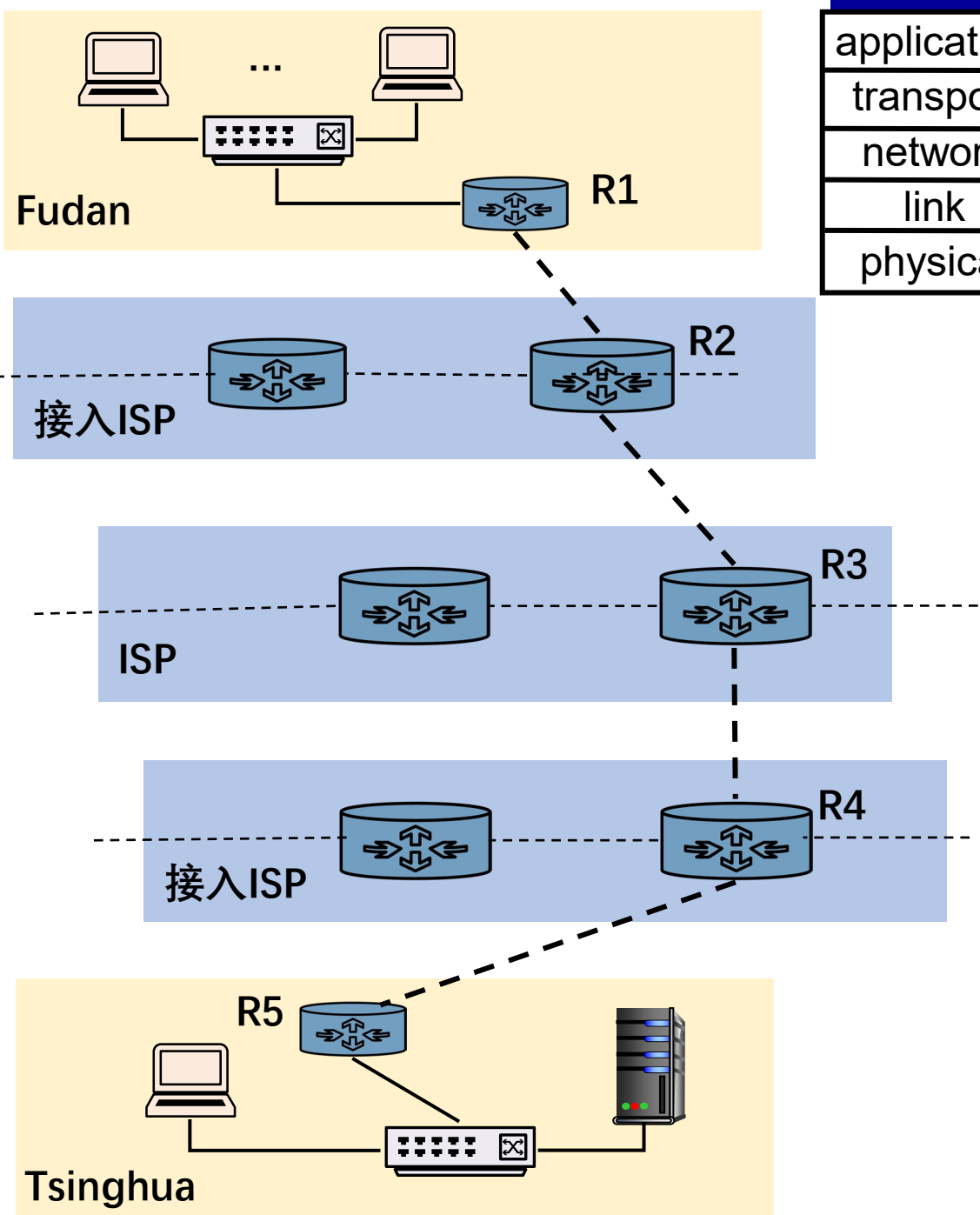
GET / HTTP/1.1
Host: www.tsinghua.edu.cn

HTTP/1.1 200 OK
...

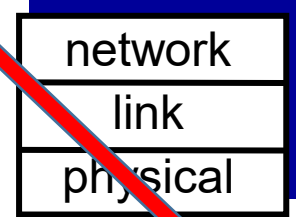
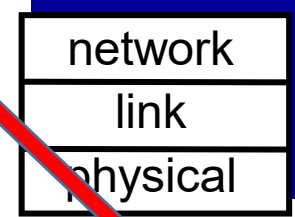
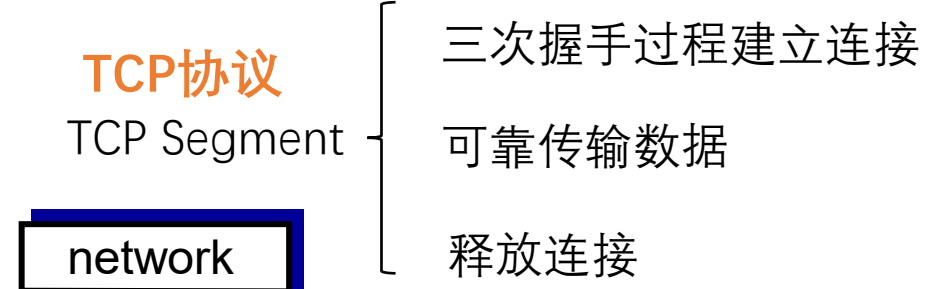


HTTP: GET / ...

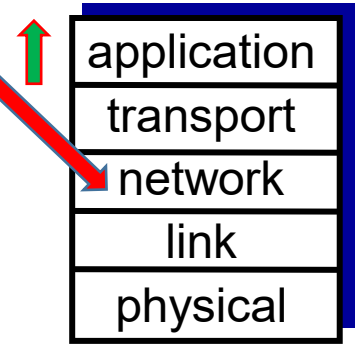


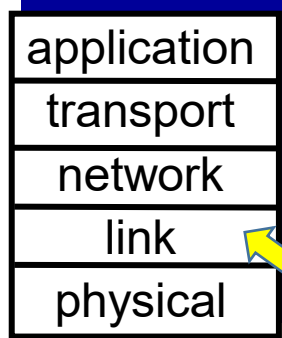
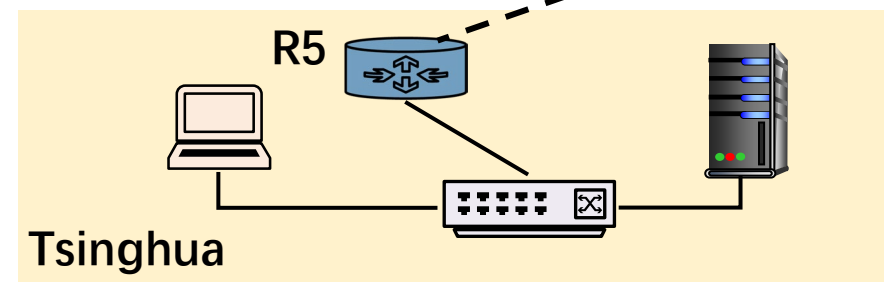
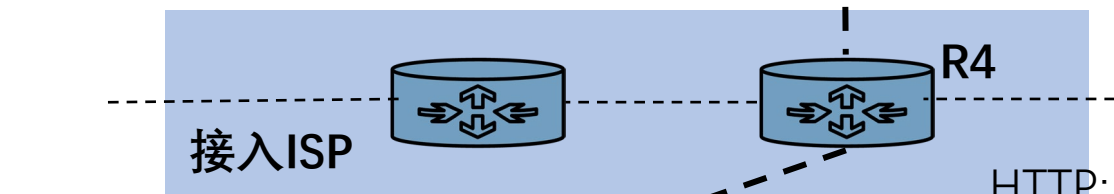
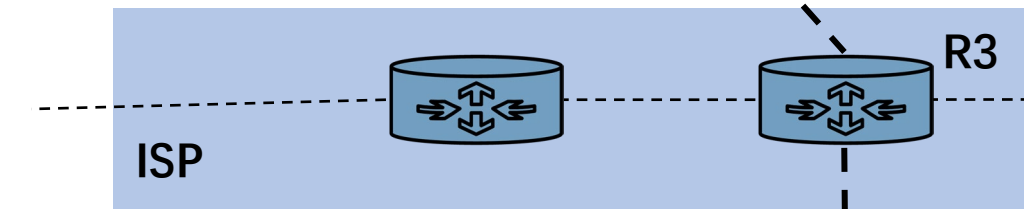
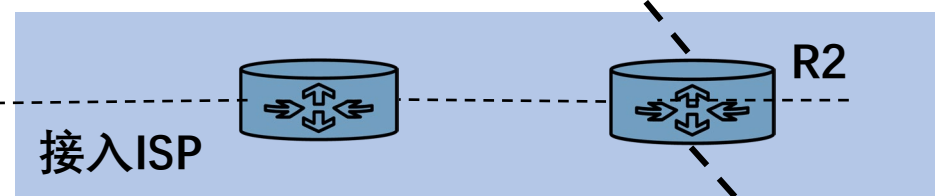
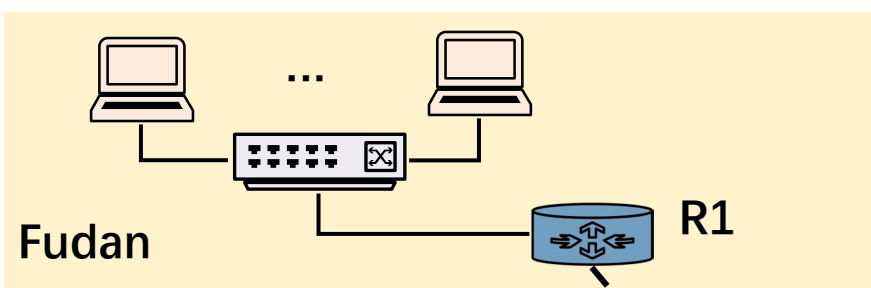


HTTP: GET / ...
TCP在网络层提供的节点间(C--S) 的数据传输服务基础上实现可靠的数据传输。<port: 1234→80>, HTTP: GET / ...



HTTP: GET / ...
TCP: <port: 1234→80>, HTTP: GET / ...

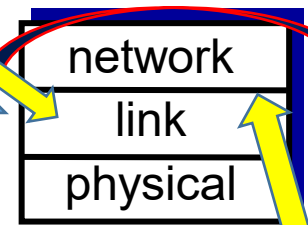




HTTP: GET / ...
TCP: <port: 1234→80>, HTTP: GET / ...
IP: <fudan→Tsinghua> <port: 1234→80> HTTP GET...
在链路层提供的服务基础上通过**多跳传输**将数据递交到最终目的地

数据平面: 从接口收到一个IP分组后查找转发表, 转发到下一跳

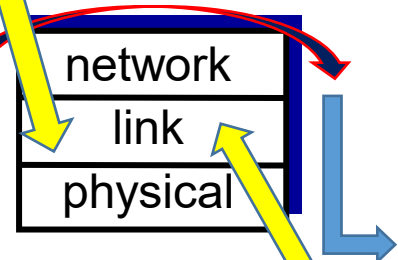
控制平面: 维护转发表



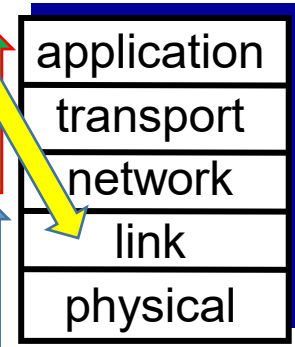
IP: <fudan→Tsinghua>

转发表	
目的网络	下一跳
166.111.0/16	R3

IP: <fudan→Tsinghua>



HTTP: GET / ...
TCP: <port: 1234→80>, HTTP: GET / ...
IP: <fudan→Tsinghua> <port: 1234→80> HTTP GET...

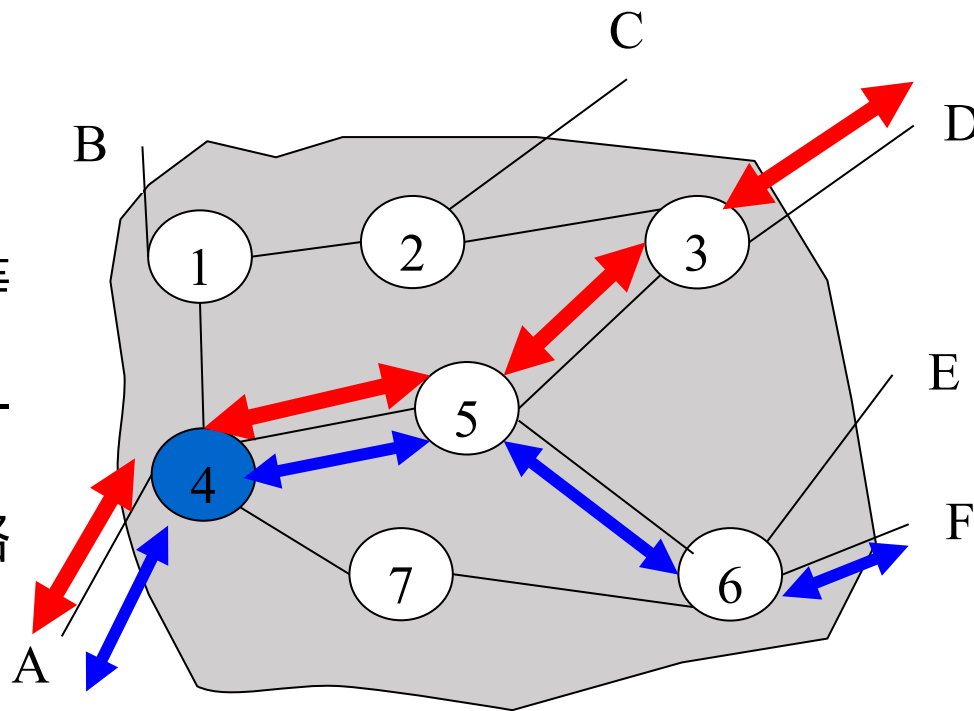


互连网(internet)的工作方式

- 互连网内部可有两种工作方式：虚电路和数据报方式
 - 网络互连设备(比如路由器)收到一个分组进行转发时所遵循的路由**何时决定**
- 数据报(datagram)方式：无连接方式
 - 无需建立连接，在**数据传输时决定路由**
 - 每个分组根据头部包括的目的地址等控制信息来独立决定路由
 - 分组经过路径可能会各不相同，可能会丢失、延迟、失序
- 虚电路(virtual circuit) 方式：面向连接的方式。X.25、帧中继(Frame Relay)和ATM（异步传输模式）
 - 连接建立、数据传输和连接释放三个阶段
 - 仅仅在**连接建立时进行一次路由选择**，以后该连接上的所有分组沿着预先建立的路径传输
 - 虚电路和电路交换的区别：
 - 虚电路: 建立了一条**逻辑上的连接**
 - 仍然采用**分组交换（存储转发）**方式，收到分组后在往下一个节点方向的端口处排队，等待链路空闲后发送
 - 电路交换：通过路由选择建立了一条电路，同时为该电路分配了供其**独占使用的带宽资源**

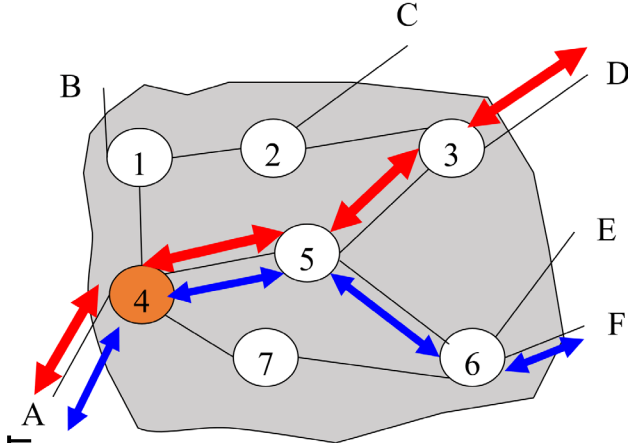
互连网(internet)的工作方式：虚电路

- 一条虚电路途中经过多个中间节点，每个节点需要维护虚电路上的上一节点和下一节点信息
- 每个链路上可能会有多条虚电路通过，要能够区分收到的分组属于哪个虚电路，要转发给哪个下一跳节点
 - 分组中包含虚电路号
 - 虚电路号全局唯一：节点之间交换信息以知道哪些虚电路号可用
 - 虚电路号在链路唯一
 - 建立和释放虚电路的协议称为**信令协议**
- **交换虚电路SVC**(Switched VC): 建立虚电路
 - 节点根据连接建立请求携带的源和目的地址等信息查找路由表，决定下一跳节点
 - 节点决定链路上区分虚电路的虚电路号，上一跳链路和下一跳链路采用不同的虚电路号
 - 实践中，一般由下一跳节点选择一个在该链路上唯一的虚电路号
- **永久虚电路PVC**(Permanent Virtual Circuit): 管理员静态配置



互连网(internet)的工作方式：虚电路

- 每个节点维护了虚电路表
 - 路由信息：前一个节点(端口)和后一个节点(端口)的标识
 - 虚电路号：前一个链路和后一个链路上唯一的虚电路号
- 虚电路号的选取要考虑到双向数据传输，节点i为链路ij选择虚电路号要考虑：
 - 本节点i在对应的**外出链路ij**上用过的虚电路号
 - **到来链路ji**上邻居节点j选择的虚电路号。可以从之前的收到的信令消息中了解到，并且已经保存在虚电路表中
- 假设目前节点A到F间建立了两条虚电路，现在D到A要建立虚电路，节点4要选择一个在链路4-5间唯一的虚电路号：
 - 查找**前一节点为5的表项**和**下一节点为5的表项**中已经使用的虚电路号，不能使用那些虚电路号

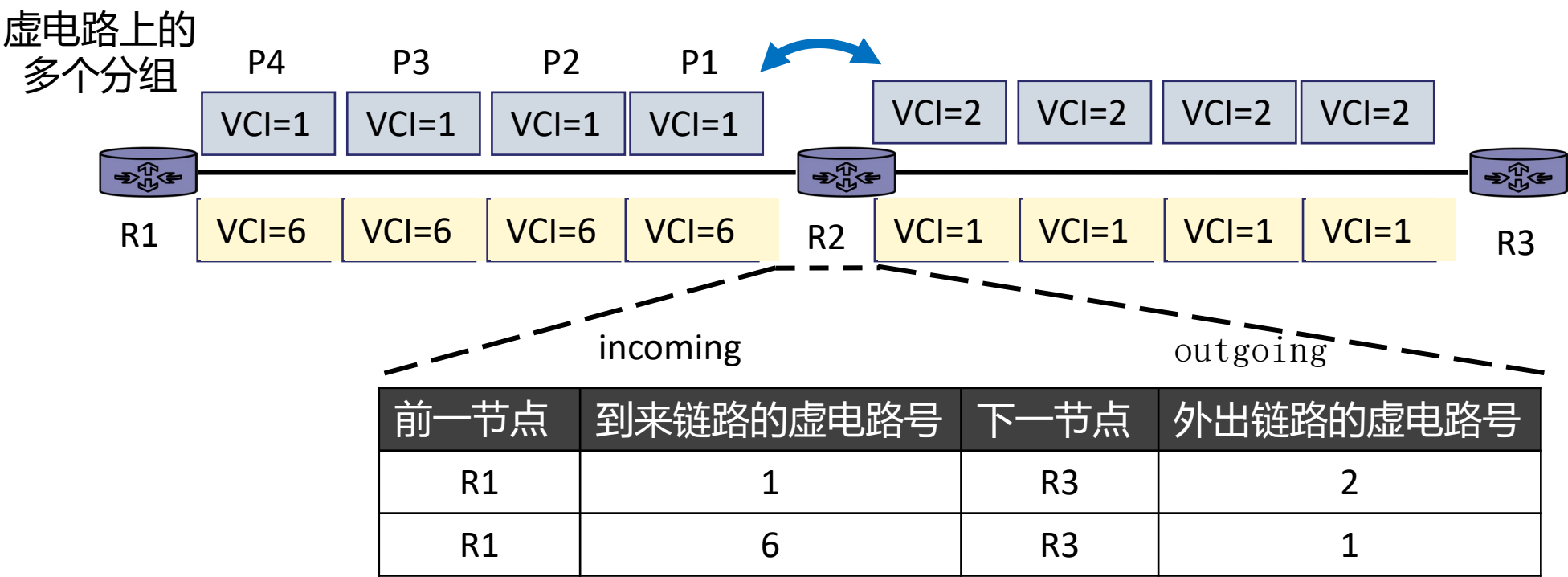


节点4的虚电路表

前一节点	到来链路的虚电路号	下一节点	外出链路的虚电路号	对应的虚电路
A	100	5	200	A-4-5-6-F
A	101	5	199	A-4-5-6-F
5	201	A	102	D-3-5-4-A

互连网(internet)的工作方式：虚电路

- 每个节点基于虚电路号进行交换。对于MPLS(Multi-Protocol **Label Switching**)而言，称为标签交换
- 从某个端口收到分组时：
 - 查找虚电路表，确定下一跳节点（外出端口）以及虚电路号
 - 将分组的虚电路号替换为新的虚电路号

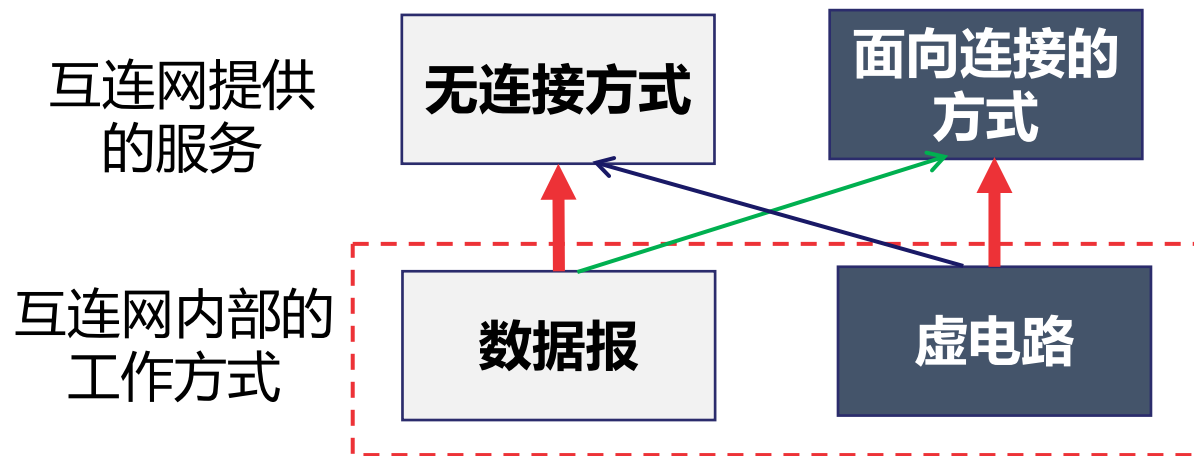


互连网(internet)的工作方式：数据报和虚电路

	数据报	虚电路
路由选择	每个分组单独选择路由，查找转发表	建立虚电路时选择路由，构建虚电路表，以后所有分组都使用该路由
延时	分组传输延时	虚电路建立，分组传输延时
节点失败的影响	除了在崩溃时正在由该节点处理的分组会丢失，其他分组无影响	所有经过失效节点的虚电路都要被终止
拥塞控制和服务质量QoS (Quality of Service)	难，分组可能失序到达	可进行准入控制，可进行差错和流量控制
地址（头部开销）	每个分组包括源端和目的端的完整地址	每个分组含有一个短的虚电路号
状态信息	子网无需为分组流保存状态信息	每个节点要保存一张虚电路表

互连网(internet)的工作方式

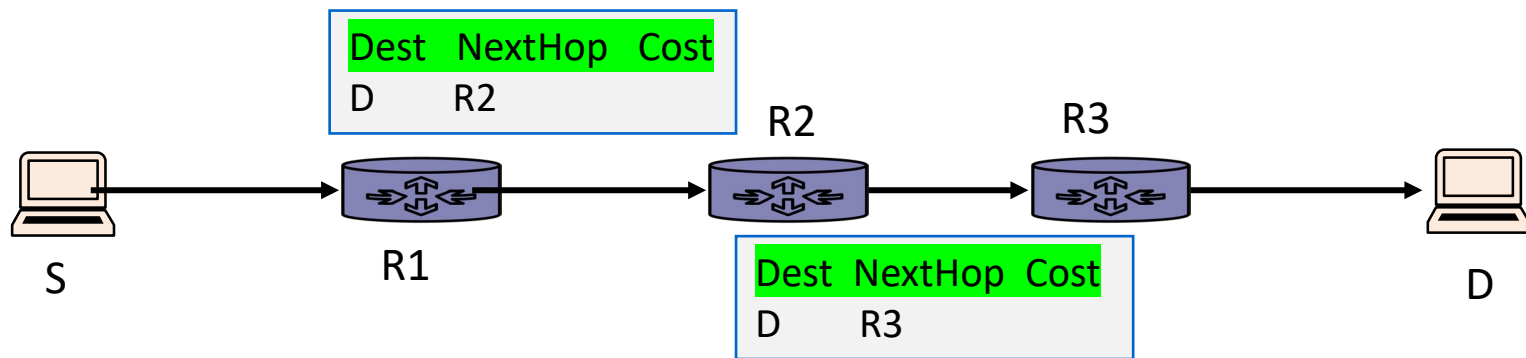
- 互连网内部可以采取数据报或者虚电路方式实现
 - Internet采用数据报方式
 - X.25、帧中继、ATM采用虚电路方式
- 互连网向更高层提供的服务也可以采取面向连接或无连接的方式
 - Internet可以连接X.25、帧中继、ATM等虚电路方式的网络
 - 可以在Internet的一部分采用MPLS技术提供分组的更快转发，保证服务质量



互连网的路由方式:逐跳路由

根据**选择路由的方式**可以分为逐跳路由和源路由

- 逐跳(per-hop)路由：每个节点仅知道路径上的下一跳节点
- 假设S到D的路径： $S \rightarrow R1 \rightarrow R2 \rightarrow R3 \dots \rightarrow Rn \rightarrow D$
- 每个节点只需要维护到目的地的下一跳节点信息即可
 - 可能短暂出现**路由回路**
- 单播(unicast)路由：**单个目的节点**
- 多播（组播, multicast）路由：一个或者多个源发送分组给**多个目的节点**，多个节点共同维护一棵组播树



互连网的路由方式:源路由

逐跳路由: 每个节点仅知道路径上的下一跳节点

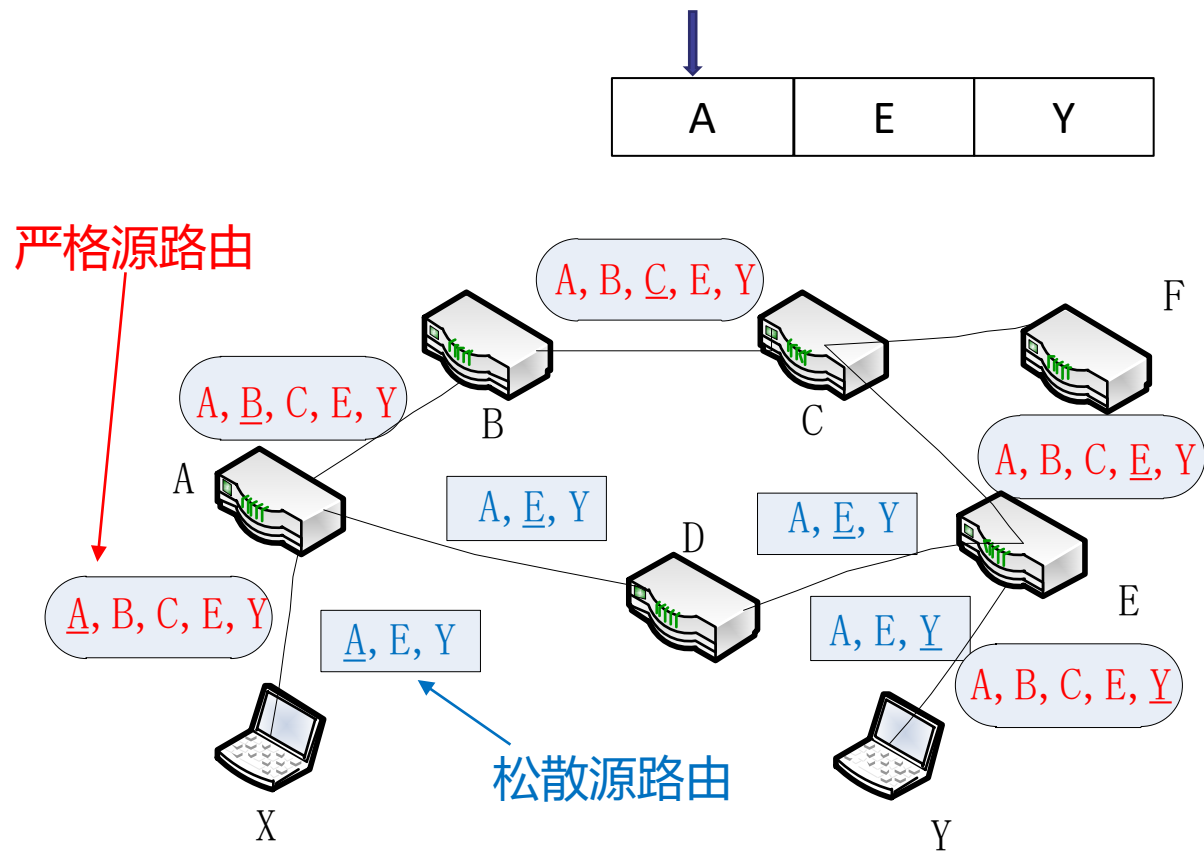
源路由:

- **发送者(源) 决定**途中经过的节点, 并将其**记录在分组头部**
 - 通过第三方 (如人工配置) 获得网络的拓扑, 决定路径
 - 或通过扩散等动态发现协议发现路径
 - 分组头部长度的限制了源路由的规模
- 途中节点根据分组中的源路由信息, 决定下一跳节点
- 源路由可用于数据报和虚电路网络等
- 适合**流量工程(traffic engineering)**, 可以根据网络的拓扑和流量情况, 决定分组流所经过的路径

互连网的路由方式:严格和松散源路由

- 严格源路由(strict source routing): 发送方给出路径上每个节点列表, 要求严格按照列表进行转发, 途中不允许经过其他节点。即要求节点列表中的相邻节点也是路径上的邻居节点
- 松散源路由(loose source routing): 给出路径上需要途经的节点列表, 允许通过不在列表中的节点
- 可结合在一起, 通过位图(bit map)来说明哪一段为严格或松散源路由

- X→Y, 松散源路由为AEY, X查找路由(转发)表以匹配A, A为下一跳
- A收到后
 - 已经到达源路由的当前节点, 将指针指向下一个节点E,**
 - 查找路由(转发)表以匹配E, 下一跳为D
- D收到后:
 - 当前指针指向的不是D, 查找转发表匹配E, 下一跳为E
- E收到后:
 - 已经到达源路由的当前节点, 指针指向下一个节点Y, 查找转发表匹配Y, 下一跳为Y**
- Y收到, 到达目的地



主要内容

网络层提供的服务：为高层提供节点到节点的传输，经过多跳传输最终到达目的地

5.1 交换和路由：internet的工作方式（虚电路和数据报）

5.1 交换和路由：路由方式

- 源路由和逐跳路由
- ~~扩散法~~
- ~~逆向学习法~~

5.2 网桥

5.3 Internet网络层

- IP协议
- ARP
- ICMP
- DHCP
- NAT
- IP隧道
- IP组播
- IPv6

Internet 网络层：Internet 设计原则

目标1：连接各种类型的异构网络

目标2：路由器和链路故障仍能通信

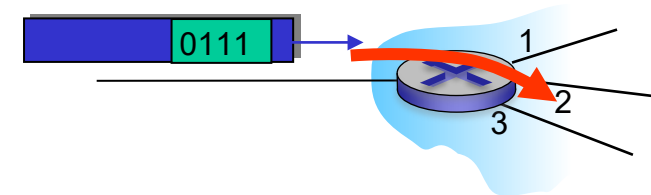
目标3：有效利用网络的资源，支持各种类型的应用

Internet设计原则

- (目标1)**沙漏模型，IP是核心**，要求下层物理网络提供基本数据递交功能、MTU至少68字节，一般不低于576字节
 - IP地址唯一标识Internet上某台主机
- (目标1)**无连接方式，IP提供尽力递交（best-effort）的数据传输服务**：无带宽保证，分组可能丢失、失序、延迟，无拥塞指示
- (目标2)命运共享(fate-sharing)策略：只要网络连通，端系统间的通信仍然可继续
 - 无状态的路由设计：路由器不需要为端系统间的**分组流**维护状态信息
 - 软状态机制：路由表的维护通过**定期地刷新**（路由信息交换）来完成，**超时**时状态被移走
 - 端到端的设计原则（end-to-end argument）：
 - 逐跳的差错控制和流量控制仍然会由于中间节点的问题需要在端系统实现这些复杂的功能
 - 在某跳中实现复杂的功能仅仅优化性能，而不是保证正确性
- (目标3)端系统提供面向连接的可靠、按序递交的**TCP服务**和无连接方式的不可靠**UDP服务**

Internet可用于连接各种异构网络，遵循命运共享、软状态和端到端的设计原则，采取松散、分布式、无状态的网络架构，提供无连接方式的尽力递交的数据传输服务

Internet 网络层：转发和路由



- 数据平面：如何**转发**分组，尽可能快转发，**微秒或纳秒级的延迟**
 - IP协议：给出了IP分组的格式，以及如何处理IP分组
 - NAT转换：内部网络中的主机的分组经过NAT设备如何出现在Internet上
 - IPv6协议：下一代的IP协议
- 控制平面：为数据平面的分组转发提供支持
 - **路由**协议：
 - 如何为IP分组找到到目的端的路径，能够处理网络的变化，**毫秒甚至是秒级的延迟**都是可以接受的
 - 传统上，路由器之间交换路由信息，按照相应的路由算法构造路由表
 - 软件定义网络(Software-Defined Networking): 原来的各个路由器的控制平面功能都集中到一个控制器，由一个集中的控制器为每个路由器计算路由，并分发到路由器
 - ARP协议：在通过物理链路传输IP分组前首先知道下一跳节点的硬件地址
 - ICMP协议：IP分组递交出错时报告差错给源端
 - IGMP协议：路由器了解到其连接的接口是否有**组播组成员存在**

- 分析头部
- 查找转发表
- 从输入端口转发到输出端口，排队等待链路空闲

IP(Internet Protocol)协议

- IP 只要求下层物理网络提供基本的数据递交功能以及MTU一般不低于576字节，提供不可靠无连接方式的数据递交服务：
 - 单播 (unicast)：一到一
 - 组播/多播 (multicast)：一到多或者多到多
 - 广播 (broadcast)：一到所有
 - 联播/任播 (anycast)：一到多中任意一个
- RFC 791定义了IP协议。IP协议分组格式的设计考虑到：
 - 无连接方式，提供转发支持：**源地址和目的地址**
 - 多个高层协议：**协议** (ICMP 0x01 、 TCP 0x06、 UDP 0x11)
 - 路由回路：**生命期TTL(Time to Live)** 字段(一般初始设置为64或128) 。每经过一个节点**转发时，TTL减1**，为0时丢弃，通过ICMP协议发送通知给发送者。目的地收到TTL=0的分组不会丢弃！
 - 分组出错：**头部检验和**字段，每跳都要重新计算
 - MTU限制：**分段和重组**
 - 扩展性：支持IP选项，固定+可选+数据，通过头部长度确定可选部分。总长度和头部长度确定数据部分的长度
 - 服务质量：服务类型(ToS)字段，早期忽略，后为区分服务码字DSCP和拥塞通知相关字段

IP分组格式

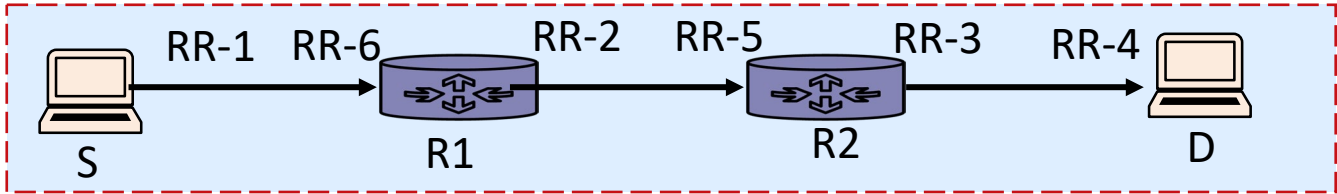
- 总长度：IP分组长度，最长65535字节
- IP头部长度：4个比特，单位为32比特，取值5-15，对应长度20-60字节
- 选项：可选的，可变长，最多40字节
- 填充：保证用户数据部分以32比特边界开始
- TTL(Time to Live): IP分组的生命期，经过中间节点转发时：TTL减1，为0时丢弃并发送通知给发送者
- 头部检验和：采用Internet检验和算法
 - 16-bit word反码加法求和后取反码
 - 检验：反码加法后验证是否为全1
 - 途中路由器转发时TTL减1，需要重新计算检验和
- 服务类型(Type of Service) ==> Class of Service
 - 早期忽略，一般为0
 - 前面6个比特用于区分服务，属于哪个负载类
 - 后面2个比特用于ECN

DSCP: Diffserv Code Point
ECN: Explicit Congestion Notification
ECT: ECN-capable terminal

6bit	1bit	1bit
DSCP	ECN	ECT
负载类	拥塞通知	是否支持ECN

4	8	16	32
版本	头部长度	服务类型	总长度
标识		D F	M F
生命期	协议	分段偏移	
头部校验和			
源地址			
目的地址			
IP选项 (如果有, 长度可变)			填充
用户数据			

IP分组格式：IP选项



记录路由RR选项

0 8 16 24 31

类型(7)	长度	指针	全0
第一个IP地址			
第二个IP地址			
...			

- IP选项最长40字节
- 当前实践中许多防火墙会过滤包含IP选项的分组
 - 单字节选项： 仅仅包含类型，与TCP选项类似
 - type=0: EOOl(End Of Option List)，表示选项结束
 - type=1: NOP(No Operation)，用于选项之间填充
 - 多字节选项：采用TLV方式描述
 - type的第一个比特（最高位）称为Copied标志，为1时所有分段都应该包含该选项，否则仅第1分段
 - 主要用来进行网络测试或调试
 - 记录路由RR选项(type=7):
 - 指针给出目前尚未使用的第一个IP地址槽的偏移，初始为4
 - 路由器转发时将外出链路的IP地址加入到列表中，指针+4。最多9个地址
 - 源路由：指定该IP分组从源到目的端必须经过的路由器列表
 - 严格源路由(type=9)：分组经过的路径必须与选项中给出的路径严格一致
 - 松散源路由(type=3)：分组转发时应该按照顺序经过那些路由器，但其中可经过其他路由器
 - 时间戳(type=68)：与RR选项类似，纪录IP地址和路由器的当前时间。 TCP有一个timestamp选项

IP分组格式：分段和重组



- 分组传输经过的物理网络都有MTU（帧携带的数据大小），至少68字节，一般超过576字节
 - 逐跳重组（透明分段法）：可能多次分段和重组，每跳必须是同一个出口
 - 接收方重组：途中可分段，最后在接收方重组
- ✓ IP采用**接收方重组**的策略，接收方要求能够处理长度至少为576字节的分段
- 源端和目的端之间会发送多个IP分组，如何知道分段是属于哪个原始的IP分组？
 - IP源地址和目的地址+**分段标识（16bit）**：标识原始分组。不是所有的分组可以分段
- 分段是原始IP分组的哪个部分？
 - 每个分段包含**分段偏移（13bit）**：分段携带的数据在原始IP分组中携带的数据部分所处位置的偏移量。**单位为8个字节**
 - 前面的分段携带数据长度为8的倍数
 - IP头部中的总长度与头部长度可知道分段携带的数据部分长度
- 接收端怎么知道所有的分段都已经到来？
 - **MF**:后面是否还有分段
 - 收到MF=0的分段，且之前所有的分段都已到来，可以重组
 - 重组计时器超时时丢弃所有的分段
 - **DF**: **不允许分段**，如果必须分段，则丢弃该分组并发送ICMP差错报告给发送者

用户数据==>用户数据1+用户数据2+用户数据3



IP分组格式：分段和重组

2400字节的IP分组（无IP选项）经过一个MTU为1000字节的网络，需要进行分段

前面的段：携带的数据长度为 $(1000-20) // 8 * 8 = 122 * 8 = 976$ 字节

分段	头部长度	总长度	分段标识	分段偏移	DF标志	MF标志
原始	20	2400	0xa767	0	0	0
1	20	996	0xa767	0	0	1
2	20	996	0xa767	122	0	1
3	20	448	0xa767	244	0	0

- 分段是有害的，尽量避免
 - 头部开销，带宽浪费
 - 每个分段的转发开销：独立选择路由，需要计算校验和
 - 重组开销：需要缓存，一个分段丢失导致所有分段丢弃
 - 分段ID只有16比特，限制了源和目的端允许分组的发送速度
- IPv6协议：途中不允许进行分段，仅允许发送端分段，接收方重组
- TCP：建立连接时协商MSS，目的是希望发送的TCP段通过IP协议传输时不需要分段
- PATH MTU Discovery：通过ICMP协议探测途中允许的MTU的下界

IP地址：MAC地址(4.2.2)，了解链路层如何发送和接收分组

一个局域网中有多个节点

- 节点通过网卡(Network Interface Card)连接到局域网
- 媒体访问控制(Medium Access Control)机制保证某个时刻只能有1个发送者
- 每个网卡(接口) 有一个在该局域网上唯一的MAC地址
- 帧的头部中包括了源MAC地址和目的MAC地址，通过MAC地址标识是哪个(节点的)接口发送，发送给哪个(节点的)接口
- 为了保证任何两块网卡在一个局域网中唯一的MAC地址，每个网卡有一个在世界上唯一的MAC地址。MAC地址有的时候也称为网卡地址、硬件地址等
- MAC地址用于标识某个网卡，并不需要描述是哪个局域网上的网卡，采用平坦地址空间
- MAC地址为6个字节(48比特)，也称为EUI-48地址 (Extended Unique Identifier)，其中前面24个比特分配给某个厂商或机构(<http://standards.ieee.org/regauth/oui/index.shtml>), 由其保证在同一机构名下的网卡地址的不同

MAC地址

最低位在前



- 48比特的MAC地址常以易于阅读的方式来表示：每个字节用16进制表示，之间通过连字符或冒号隔开，比如AC:DE:48:12:78:80或AC-DE-48-12-78-80
- IEEE 802标准中在描述MAC地址时采用最低位在前的顺序。第1个字节的最低两位(LSB)有特殊含义：
 - 最低位为I/G(Individual/Group)，为0时表示单播地址，为1时表示组播地址
 - 次低位为U/L(Universal/Local)，为0时为全局地址，为1时是本地管理地址
 - 比如：AC-DE-48-12-7B-80，该地址并不是组播地址!!!

- 网络字节顺序：Most Significant Byte first, Most Significant Bit first
- IEEE标准：Most Significant Byte first, Least Significant Bit first

	I/G	U/L	OUI									
IEEE802地址格式	0011	0101	0111	1011	0001	0010	0100	1000	1101	1110	0000	0001
十六进制表示	AC		DE		48		12		78		80	


- Ethernet II, Src: HuaweiDe_ee:83:69 (78:85:f4:ee:83:69), Dst: IPv4mcast_7f:ff:fa (01:00:5e:7f:ff:fa)
 - Destination: IPv4mcast_7f:ff:fa (01:00:5e:7f:ff:fa)
 - Address: IPv4mcast_7f:ff:fa (01:00:5e:7f:ff:fa)
 -0. = LG bit: Globally unique address (factory default)
 -1 = IG bit: Group address (multicast/broadcast)
 - Source: HuaweiDe_ee:83:69 (78:85:f4:ee:83:69)
 - Address: HuaweiDe_ee:83:69 (78:85:f4:ee:83:69)
 -0. = LG bit: Globally unique address (factory default)
 -0 = IG bit: Individual address (unicast)
- Type: IPv4 (0x0800)

MAC地址

- 局域网(以太网)支持三种数据传输模式：
 - 单播：发送帧给指定的接收者
 - 广播：发送帧给链路上的所有接收者，广播帧的目的地址为全1，即FF-FF-FF-FF-FF-FF
 - 组播：发送帧给链路上的某些接收者，组播地址中的第一个字节的最低位为1
- 节点(网卡)从链路上收到一个单播帧之后，进行过滤
 - 如果帧的目的地址和节点的MAC地址匹配，则其递交给高层
 - 如果不匹配，则丢弃该帧
- 网卡可设置为混杂(promiscuous)模式：所有收到的帧都交给高层
- 收到广播帧时接收并递交给高层
- 收到组播帧时：
 - 网卡想要接收某些组播帧，必须通知网卡对哪些组播地址感兴趣
 - 网卡维护少数几个感兴趣的组播地址列表
 - 网卡还可纪录多个感兴趣的组播地址的散列值
 - 首先将帧中的目的组播地址与感兴趣的组播地址列表匹配；如果不匹配时，计算目的组播地址的散列值，然后与保存的散列值比较，如果匹配，则接收并递交给高层

IP地址

- 每个节点(主机和路由器)的网络接口(网卡) 都有一个IP地址: **ID + Locator**
 - 唯一标识连接到IP网络的接口
 - 描述该接口所在的位置 (网络)


 - IP地址分为网络号和主机号, 网络号标识某个"物理"网络, 主机号标识该网络中的主机
 - 路由时只需了解如何到达接口所在的(物理)网络, 而不必了解该网络的每一台主机
 - 第二层物理网络可以直接递交给其所在物理网络中的其他节点
 - 需要将IP地址映射为物理地址→ARP协议
 - 网卡地址为48比特的整数, 描述时一般将每个字节转换为十六进制, 中间以-或:隔开
- 目前的Internet(采用IPv4协议)使用的IP地址为**32比特的整数**, 总共4,294,967,296 (2^{32})个地址
 - 采用**点十进制方法描述**: 每个字节转换为十进制数字, 中间以.隔开

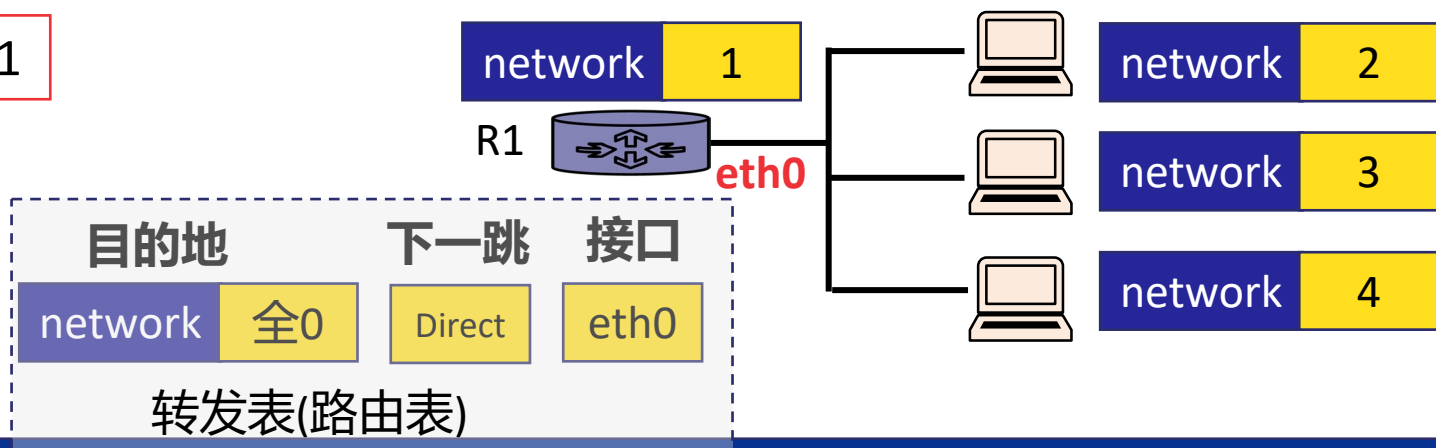
11001010 01111000 11100000 01010001

ca 78 e0 51

202 120 224 81

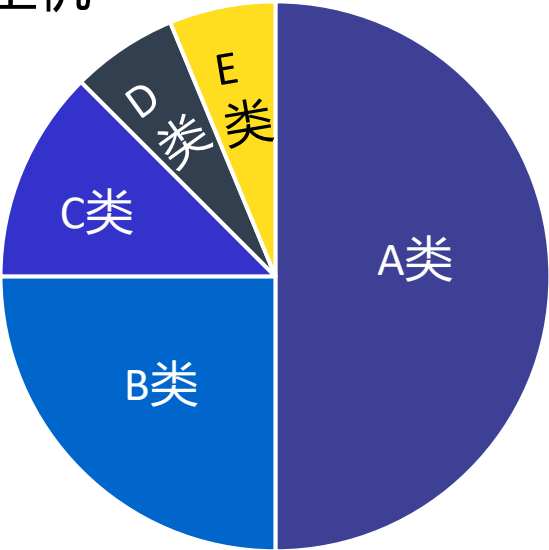
ping 0xca78e051

ping 202.120.224.81



IP地址类

- 如何知道IP地址哪些属于网络部分和主机部分?
- 最初的设计, 根据IP地址的前面几个比特的取值判断
 - 考虑到不同的网络规模, 引入A类、B类和C类地址, 主机部分可以1、2或3个字节
 - A类: 1/2地址空间, 网络号全0和127有特殊含义, 支持1千6百多万($2^{24}-2$)主机
 - B类: 1/4地址空间, 16384 (2^{14}) 个网络, 65534($2^{16}-2$)个主机
 - C类: 1/8地址空间, 200多万 (2^{21}) 个C类网络, 254台主机



单播

组播

	0	1	2	3	4	8	16	24	31	IP地址范围			
A类	0	网络号				主机号					0.0.0.0 ~ 127.255.255.255		
B类	1	0	网络号				主机号					128.0.0.0 ~ 191.255.255.255	
C类	1	1	0	网络号				主机号					192.0.0.0 ~ 223.255.255.255
D类	1	1	1	0	组播地址							224.0.0.0 ~ 239.255.255.255	
E类	1	1	1	1	保留供将来使用							240.0.0.0 ~ 255.255.255.255	

特殊的IP地址

network

host

全0表示this, 全1表示all

- 主机部分

- 全0: 表示本网络, 比如 202.120.224.0
- 全1: 可作为目的地址, 表示网络中的所有主机, 定向广播地址, 比如202.120.224.255

network

0000

network

1111

- 网络部分和主机部分:

- 全1: 255.255.255.255, 可作为目的地址, 本地/有限(limited) 广播地址, 表示接口上的所有IP主机
- 全0: 0.0.0.0
 - IP分组头部的源地址: 表示尚未知道所用的IP地址
 - 路由表中的目的网络地址: 表示缺省路由

111111

1111

000000

0000

- 网络部分:

- 全0: 表示尚未知道网络部分的地址, 很少使用

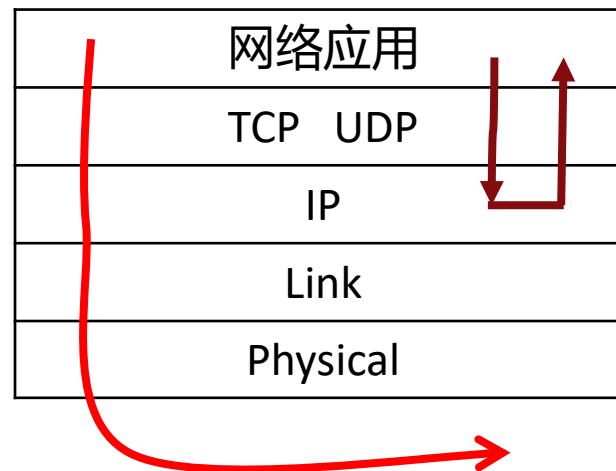
000000

host

- A类地址127.0.0.0为loopback地址

IP地址：回环地址(loopback address)

- **127.0.0.1**(实际上127.x.x.x)为回环(loopback)地址，对应名字localhost
 - 任何发送到该地址的IP分组不会发送到实际的网卡上，而是由IP模块递交给高层相应的协议(TCP或者UDP)模块
 - 可测试协议栈，可用于主机内不同进程间的通信
 - IP模块从高层收到目的地为**本机某个接口的IP地址**时通过回环接口递交给高层协议模块



```
demo@mars2:~$ ifconfig
enp0s3  Link encap:Ethernet  HWaddr 08:00:27:e9:b7:0f
        inet addr:10.0.2.15  Bcast:10.0.2.255  Mask:255.255.255.0
        inet6 addr: fe80::e6bc:e568:cc49:4197/64 Scope:Link
        UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
        RX packets:4796 errors:0 dropped:0 overruns:0 frame:0
        TX packets:2179 errors:0 dropped:0 overruns:0 carrier:0
        collisions:0 txqueuelen:1000
        RX bytes:6010324 (6.0 MB)  TX bytes:164340 (164.3 KB)

lo      Link encap:Local Loopback
        inet addr:127.0.0.1  Mask:255.0.0.0
        inet6 addr: ::1/128 Scope:Host
        UP LOOPBACK RUNNING  MTU:65536  Metric:1
        RX packets:410 errors:0 dropped:0 overruns:0 frame:0
        TX packets:410 errors:0 dropped:0 overruns:0 carrier:0
        collisions:0 txqueuelen:1000
        RX bytes:39299 (39.2 KB)  TX bytes:39299 (39.2 KB)
```

- Windows命令：**ipconfig**/route/netstat/arp
- Linux的ip命令提供了查看和修改网络配置的功能，用于替代ifconfig/arp/route等
 - ip link: 网络接口
 - **ip address: 网络接口IP地址**
 - ip route: 路由表，替代route
 - ip neighbor: 邻居，替代arp

IP地址：子网地址

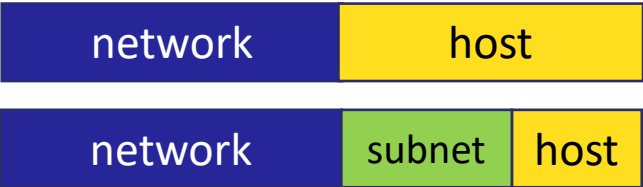
A类/B类/C类网络内部进一步划分子网

- 三层层次结构：网络ID + 子网ID + 主机ID
- 外部路由器知道怎么到达网络ID，而本地路由器知道子网怎么划分，知道怎么到达子网中的主机
- 引入子网掩码知道子网的网络部分和主机部分
 - 32比特的整数，可采用点十进制方法描述
 - 如果为主机部分，则对应的比特为0，否则为1
 - 可采用**按位与**运算来截取网络部分
 - 实践中一般主机部分为最后的一些比特，用/n描述，n表示前面n个比特（前缀）都为网络部分

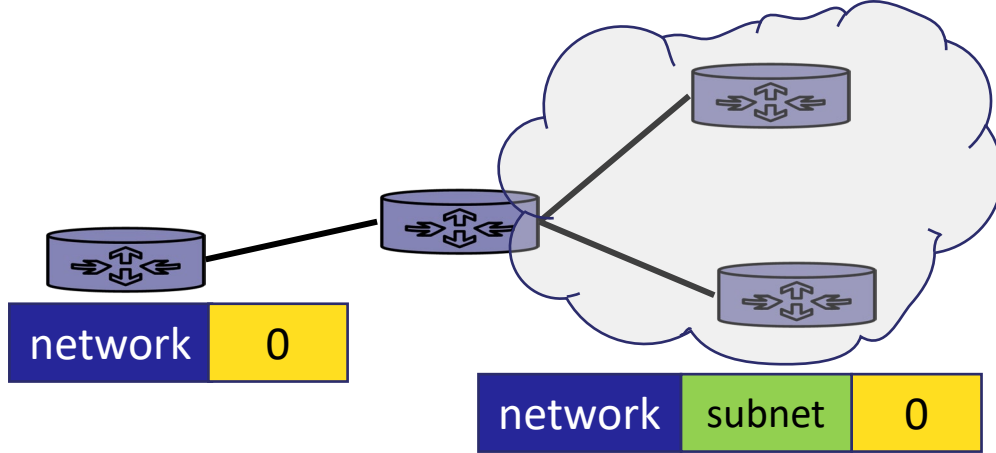
目的IP地址是A/B/C类地址，确定网络部分



目的IP地址 + 网络掩码，确定网络部分



11001010.01111000.11100000.10100000	202.120.224.160	IP地址
11111111.11111111.11111111.11000000	255.255.255.192	子网掩码
<hr/>		
11001010.01111000.11100000.10000000	202.120.224.128	子网地址
11001010.01111000.11100000.10111111	202.120.224.191	子网广播地址



IP地址：变长子网掩码VLSM

- 固定长度的子网划分的子网规模差不多，实践中容易造成IP地址空间的浪费
- 变长子网掩码（Variable Length Subnet Mask）：

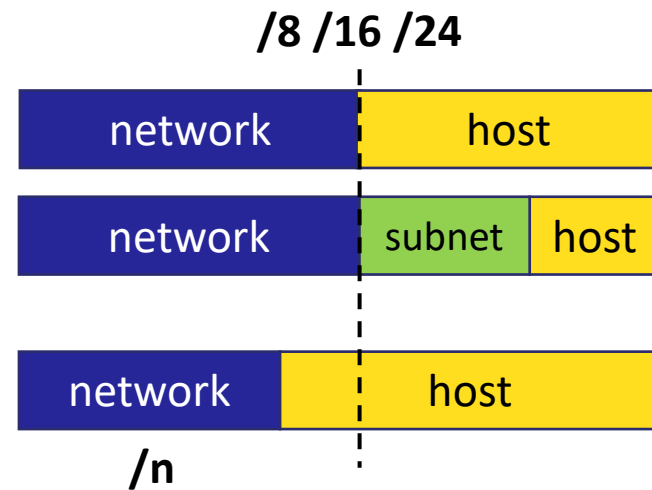
RFC 980建议不使用全0和全1的子网，是因为早期的路由消息不包含掩码

 - 不同子网可采用不同长度的子网掩码。
 - 假设202.120.224.0/24要分配给5个子网，其中3个子网要支持50台，另外2个子网支持30台主机

网络	网络（子网）地址	网络掩码	广播地址	主机可用地址范围
N	202.120.224.0/24	255.255.255.0	202.120.224.255	202.120.224.1 ~ 202.120.224.254
A	202.120.224.0/26	255.255.255.192	202.120.224.63	202.120.224.1 ~ 202.120.224.62
B	202.120.224.64/26	255.255.255.192	202.120.224.127	202.120.224.65 ~ 202.120.224.126
C	202.120.224.128/26	255.255.255.192	202.120.224.191	202.120.224.129 ~ 202.120.224.190
D	202.120.224.192/27	255.255.255.224	202.120.224.223	202.120.224.193 ~ 202.120.224.222
E	202.120.224.224/27	255.255.255.224	202.120.224.255	202.120.224.225 ~ 202.120.224.254

IP地址：CIDR

- 原有的IP地址类引入了子网，但IP地址仍然有很大一部分被浪费
 - A类网络主机数太多（地址浪费）
 - B类网络主机64K，但只有16k个B类地址
 - C类网络主机数太少（增加路由负担，需要合并）
- 主干路由器的路由表越来越庞大，占用空间，增加查找开销
- 引入CIDR，抛弃IP地址类的边界，通过**地址汇集**的方法来描述连续地址块(某个网络)
 - 用于描述网络部分的**网络掩码可以是任意长度**，从0到32
 - /32只有一个地址，表示主机
 - 长度为31的掩码有两个地址，常用于路由器之间的点到点链路上。**主机部分为0和1**不被解释为网络或广播地址
 - 长度为0的掩码相当于任意IP地址，表示缺省路由
- 子网(subnet)和超网(supernet)
 - 决定网络部分和主机部分的分界线从原来的IP地址类确定的界限往右移动，称为子网，往左边移动称为超网



根据转发表确定网络部分以进行匹配：
目的IP地址 + 网络掩码 = 目的网络 + 网络掩码

IP地址： 常用CIDR地址块

CIDR前缀长度	点十进制	地址个数	类地址个数
/13	255.248.0.0	512 K	8个B类或2048个C类
/14	255.252.0.0	256 K	4个B类或1024个C类
/15	255.254.0.0	128 K	2个B类或512个C类
/16	255.255.0.0	64 K	1个B类或256个C类
/17	255.255.128.0	32 K	128个C类
/18	255.255.192.0	16 K	64个C类
/19	255.255.224.0	8 K	32个C类
/20	255.255.240.0	4 K	16个C类
/21	255.255.248.0	2 K	8个C类
/22	255.255.252.0	1 K	4个C类
/23	255.255.254.0	512	2个C类
/24	255.255.255.0	256	1个C类
/25	255.255.255.128	128	1/2个C类
/26	255.255.255.192	64	1/4个C类
/27	255.255.255.224	32	1/8个C类

IP地址：CIDR示例

ISP拥有16个C类地址，192.60.128.0/20。假设某个用户需要1000个左右的IP地址，可以分配4个连续的C类地址

4个C类地址被汇集成为：192.60.128.0/22

192.60.128.0/24 11000000.00111100.10000000.00000000 C类地址

192.60.129.0/24 11000000.00111100.10000001.00000000

192.60.130.0/24 11000000.00111100.10000010.00000000

192.60.131.0/24 11000000.00111100.10000011.00000000

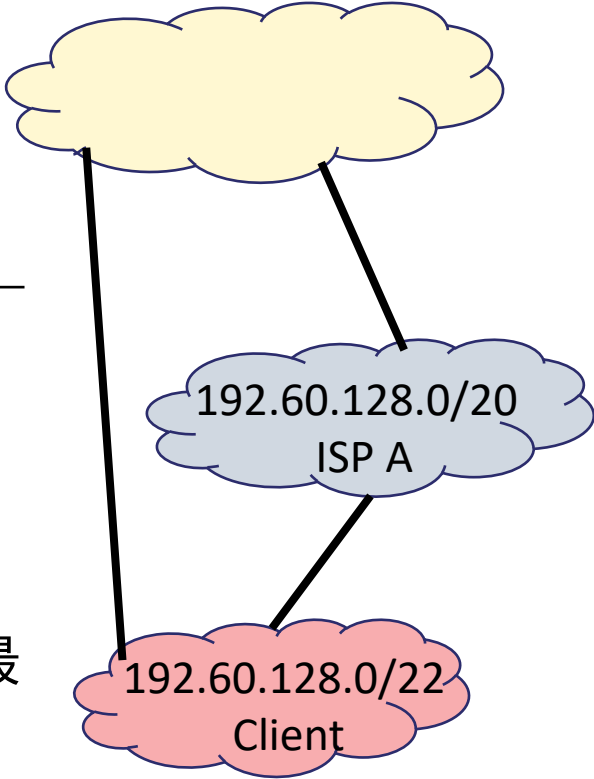
192.60.128.0/22 11000000.00111100.10000000.00000000 超网地址

255.255.252.0 11111111.11111111.11111100.00000000 网络掩码

192.60.131.255 11000000.00111100.10000011.11111111 广播地址

- 转发表中可能会有多个表项匹配，需要采取最长前缀匹配原则：网络掩码最长的匹配

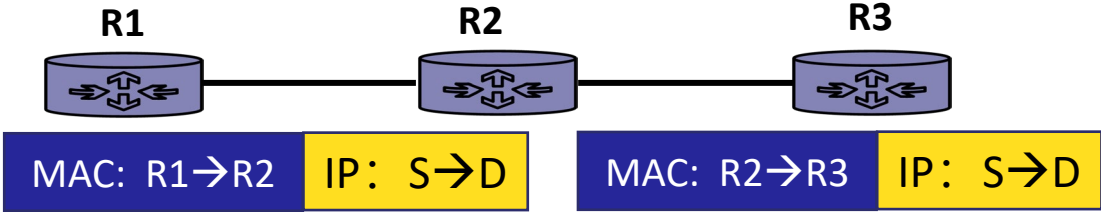
Dest	Next Hop
192.60.128.0/20	ISP A
192.60.128.0/22	Client



IP转发

- 接收分组，查找转发表将分组转发到相应的外出接口
 - 路由：节点之间交换路由信息建立路由表
 - 转发表根据路由表来确定

- 转发表一般包含：**目的网络/网络掩码**/下一跳/网络接口
 - 网络掩码为255.255.255.255(/32)的路由为**主机路由**
 - 网络掩码为0.0.0.0(/0)的路由为**缺省路由**，当转发表中找不到路由时采用
 - 直接路由**：目的节点和当前主机在同一网络，可直接通过对应的接口递交
 - 间接路由**：需要通过下一跳路由器转发
- 转发过程中不改变源和目的IP地址，改变的是MAC地址



R2: 根据D查找转发表，下一跳为R3，通过ARP获得R3的MAC地址，转发给R3

- 收到IP分组后如何匹配转发表中的表项？
 - 目的IP地址 & 表项的网络掩码 是否等于 目的网络&网络掩码
- 最长匹配原则：网络掩码最长的匹配
 - 10.11.12.8 通过eth0直接递交
 - 10.11.4.5 下一跳 10.11.12.2
 - 10.12.2.2 下一跳 10.11.12.1

	目的网络	网络掩码	下一跳路由器	网络接口
直接路由	10.11.12.0	255.255.255.0	0.0.0.0	eth0
	127.0.0.0	255.0.0.0	0.0.0.0	lo
	10.11.0.0	255.255.0.0	10.11.12.2	eth0
	202.120.224.4	255.255.255.255	10.11.12.2	eth0
缺省路由	0.0.0.0	0.0.0.0	10.11.12.1	eth0

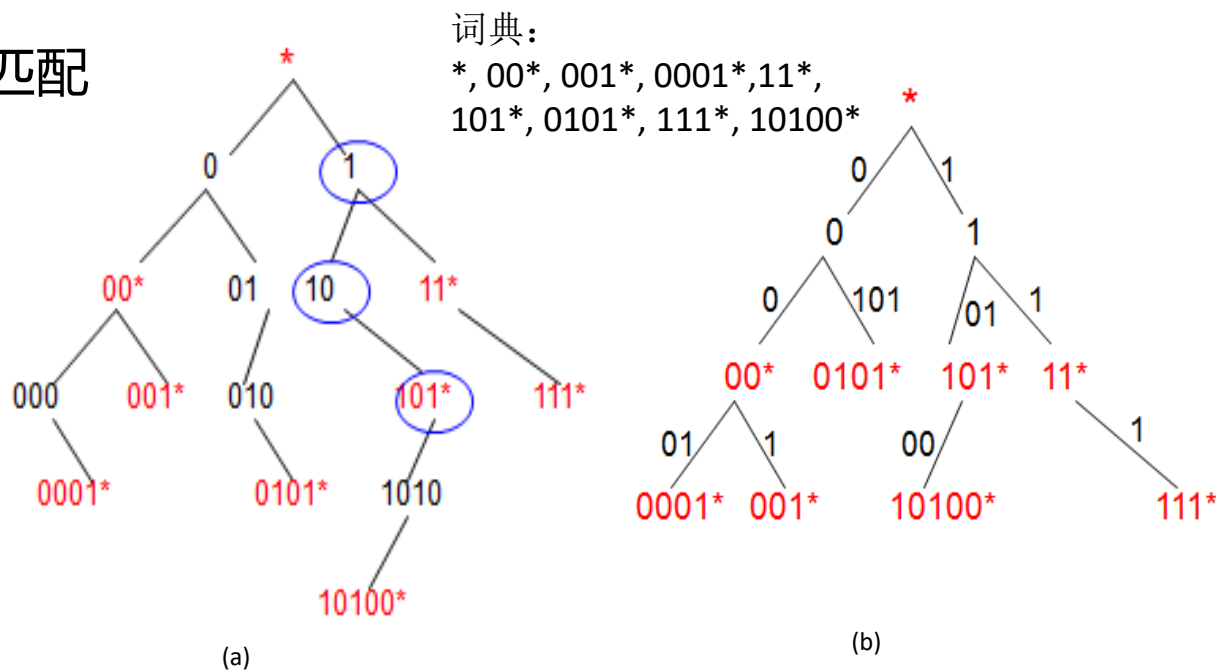
IP转发：最长前缀匹配

- 传统的BSD Unix系统采用一个称为Trie的前缀树的数据结构
- 网络地址/掩码用前缀*表示，所有前缀构成词典
- 相当于二叉树，从根开始根据前缀的每个比特取值来决定所在的分支
- 匹配时根据目的地址的每个比特决定从根开始经过的节点，直到所有比特匹配完或者无法进一步匹配时结束
 - 如果没有经过带*的节点，无法匹配
 - 否则**最后一个带*节点对应前缀**就是最长前缀匹配

目的：10101101

二叉树可以压缩：

- 如果Trie中的某个节点下面只有一条路径，而且途中没有带*的节点，那么这段路径可以压缩为该节点的子节点
- 在匹配时可以一次匹配多个比特而不是一定要逐个逐个比特地匹配



IP地址分配

- 公网IP地址由互联网号码分配机构IANA(Internet Assigned Number Authority)分配, 保证其在Internet上唯一
- IANA进一步把IP地址的分配分别授权给多个区域互联网注册管理机构RIR(regional Internet registries), 包括APNIC、ARIN等
- RIR进一步授权给更低等级的注册管理机构(National Internet Registry或Local Internet Registry), 比如中国互联网络信息中心 <http://www.cnnic.cn/>

REGISTRY	AREA COVERED
AFRINIC	Africa Region
APNIC	Asia/Pacific Region
ARIN	Canada, USA, and some Caribbean Islands
LACNIC	Latin America and some Caribbean Islands
RIPE NCC	Europe, the Middle East, and Central Asia



内部IP地址

- 内网IP地址给内部网络使用，在内部网络中唯一
- RFC 1918 Address Allocation for Private Internets给出了可供内部网络使用的地址
 - 1个A类地址：10.0.0.0~10.255.255.255，可用10.0.0.0/8表示
 - 16个B类地址：172.16.0.0~172.31.255.255，可用172.16.0.0/12表示
 - 256个C类地址：192.168.0.0~192.168.255.255，可用192.168.0.0/16表示
- 内部网络中的主机可采用内部IP地址，甚至可采用其他合法的IP地址，但是这些特殊的地址：
 - 在出口路由器处会被过滤掉，不允许出现在Internet上
 - 或者经过NAT设备映射为公网IP地址
- **拓展：**Link-local地址：169.254.0.0/16，RFC3927定义，用于IP地址的自动配置 (www.zeroconf.org)，仅仅在链路上使用，不允许转发到其他链路
- **拓展：**Shared地址：100.64.0.0/10，RFC 6598定义，为运营商级(Carrier-grade)的NAT设备使用，运营商应该确保其不会出现在Internet上

主要内容

网络层提供的服务：为高层提供节点到节点的传输，经过多跳传输最终到达目的地

5.1 交换和路由：internet的工作方式（虚电路和数据报）

5.1 交换和路由：路由方式

- 源路由和逐跳路由
 - ~~扩散法~~
 - ~~逆向学习法~~
- IP协议
 - **ARP**
 - ICMP
 - DHCP
 - NAT
 - IP隧道
 - IP组播
 - IPv6

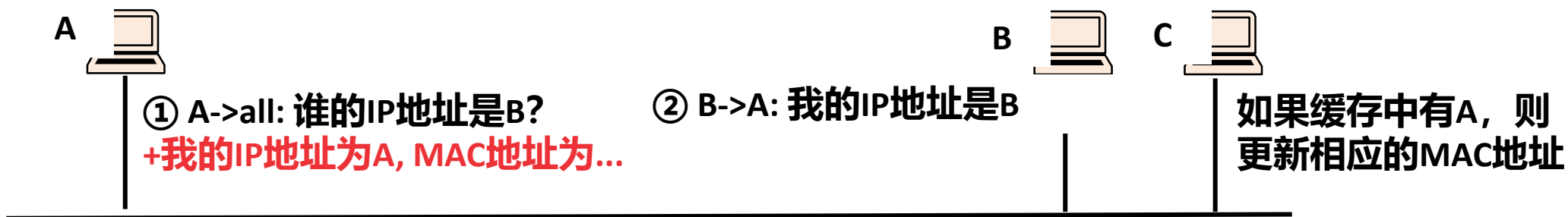
~~5.2 网桥~~

5.3 Internet网络层

地址解析协议ARP (Address Resolution Protocol)

用于以太网等广播网络中，**将IP地址映射为MAC地址**。RFC 826

- 如果不知道IP地址对应的MAC地址，发送ARP请求，暂时保留IP分组在缓冲区
 - ARP请求采用链路层广播发送：Sender IP、MAC address; Target IP、MAC address=?(全0)
- Target IP节点单播发送响应给发送者，源和目的字段对调，并且填写源MAC地址: Sender IP、MAC address; Target IP、MAC address
- IP地址和MAC地址映射保存在ARP缓存中，一定时间（60秒或30秒）后移走
 - ARP请求的目的站点B (A→B) 缓存A的映射
 - **ARP响应的接收者A缓存B的映射**，ARP是**无状态协议**，并不记录是否发送过请求
 - 其他接收到ARP请求的站点在缓存中已包含A的映射时更新映射
 - **拓展：**刷新ARP缓存时，可以**单播发送ARP请求**，确认以前的缓存是否仍然活跃。



地址解析协议ARP： ARP分组格式

- ARP请求：
 - 要解析地址时：在广播网络(Ethernet)中广播发送，目的MAC地址为全1
- ARP响应：单播发送，**源MAC地址为答案**，目的MAC地址为收到ARP请求中的发送者硬件地址

48比特	48比特	16比特	28字节	18字节	32比特
目的MAC地址	源MAC地址	协议 (0x0806)	ARP分组	填充	CRC

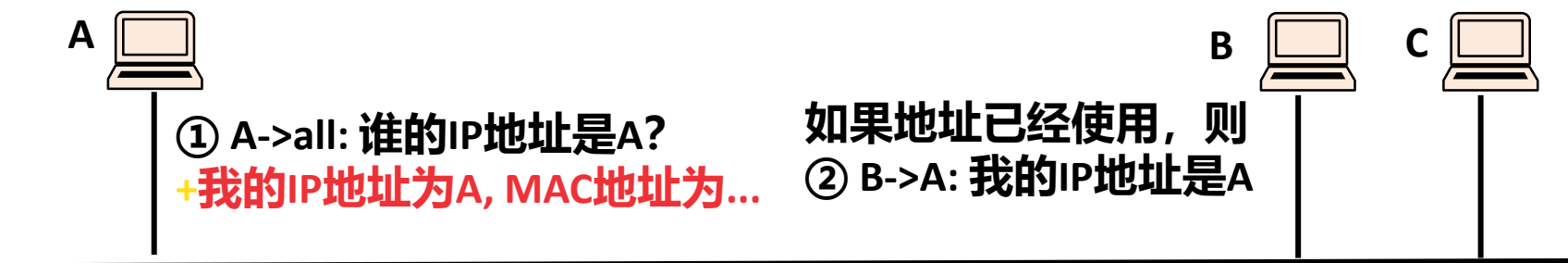
硬件类型(=1, Ethernet)		协议类型(=0x0800, IP)	
硬件地址长度	协议地址长度	操作（请求为1，响应为2）	
发送者硬件地址（Ethernet： 6个字节）			
发送者协议地址（IP： 4个字节）			
目标硬件地址（Ethernet： 6个字节）			
目标协议地址（IP： 4个字节）			

- 以太网帧携带用户数据：28字节
ARP+18字节填充=46字节
- 46 + 18字节
Ethernet头部 = 64字节=512比特

地址解析协议ARP

- 拓展：无故（Gratuitous）ARP

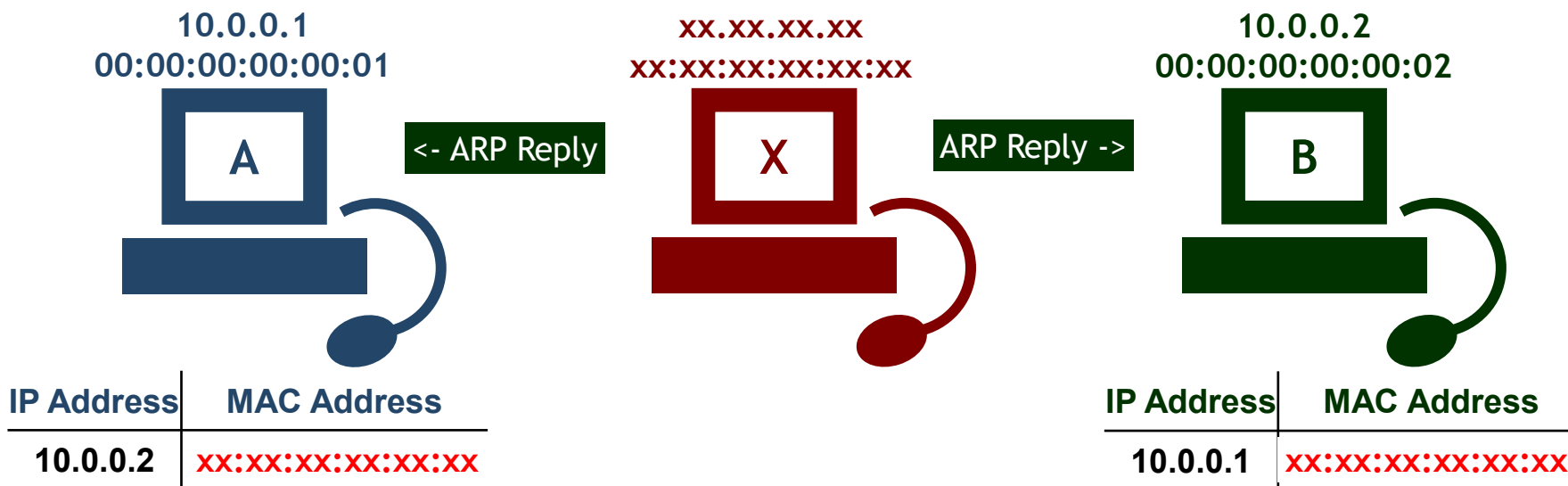
- 广播一个要求解析自己IP地址的ARP请求: sender IP, MAC address; sender IP, MAC address=?
- 可以让其他节点更新发送者的MAC地址。如在Mobile IP中MH主机回到Home Network时发送
- 在启动或者改变IP地址时，如果要检查IP地址是否已经使用： RFC5227
 - 广播ARP Probe请求: sender IP=0.0.0.0, MAC address; sender IP, MAC address=?
 - 如果连续3个probe请求都没有响应，接下来广播ARP请求（称为ARP announcement): sender IP, MAC address; sender IP, MAC address=?
 - 也可广播ARP响应: sender IP, MAC address; sender IP, MAC address=全0或全1，但考虑到历史因素以及与早期ARP实现的兼容，不建议
- 静态ARP映射可避免解析过程带来响应延迟，也可避免ARP攻击带来的安全威胁
 - arp或ip neighbor命令：显示和修改ARP缓存
 - arping命令：通过ARP协议判断网段上对应IP地址的主机是否存在



如果缓存中有A，则更新相应的MAC地址

ARP Spoofing Attack(Man-in-the-Middle)

- A要和B通信，首先A广播ARP请求，B发送响应给A
- 攻击者伪造一个ARP响应，单播发送给被攻击者
- 节点收到伪造的响应后会更新缓存，即poisoning ARP缓存



X将收到的来自于A（或B）的分组转发给B（或A）
A和B并没有觉察到中间人X的存在

主要内容

网络层提供的服务：为高层提供节点到节点的传输，经过多跳传输最终到达目的地

5.1 交换和路由：internet的工作方式（虚电路和数据报）

5.1 交换和路由：路由方式

- 源路由和逐跳路由
 - ~~扩散法~~
 - ~~逆向学习法~~
- IP协议
 - ARP
 - **ICMP**

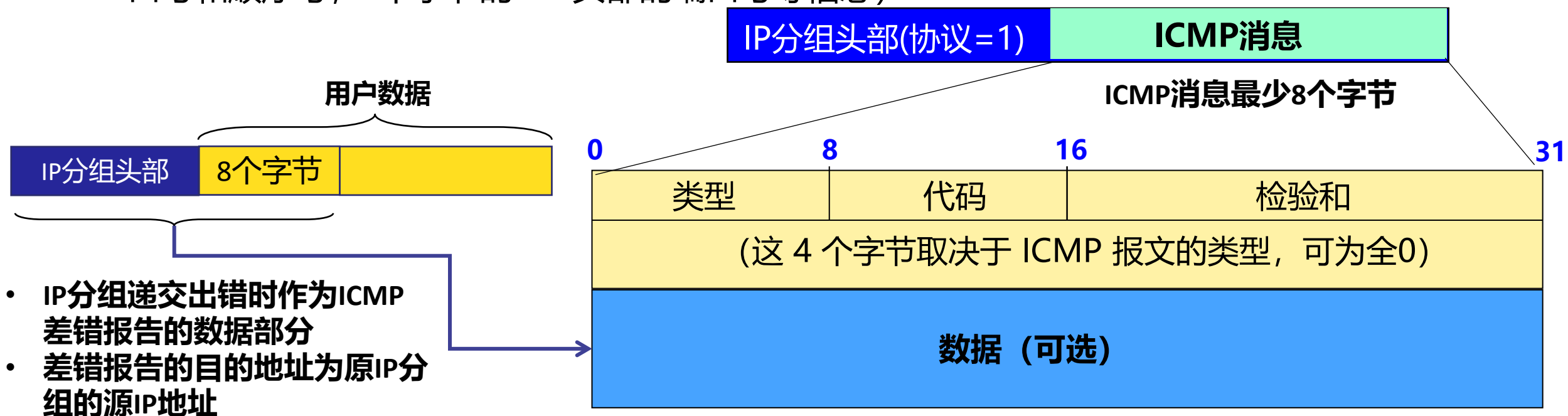
~~5.2 网桥~~

5.3 Internet网络层

- DHCP
- NAT
- IP隧道
- IP组播
- IPv6

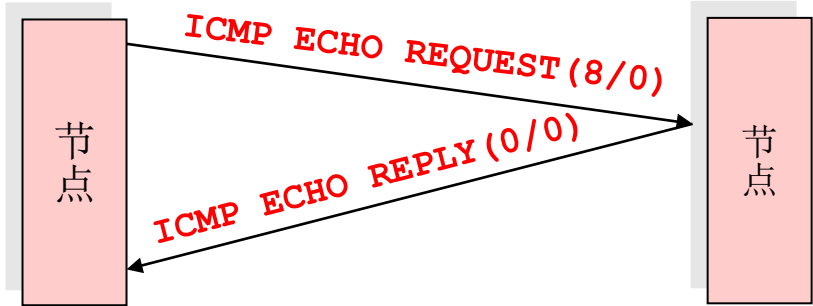
ICMP (Internet Control Message Protocol)

- ICMP是Internet网络层协议的一部分，必须实现, RFC 792
 - **差错报告**：在IP分组由于TTL超时，目的不可达，不能分段等无法递交时报告差错
 - 网络管理：判断目的地的连通情况以及途中经过的路由器 (ping/traceroute)
 - 端系统配置：路由器和端系统之间通过ICMP消息为端系统配置某些协议参数(如缺省路由器)
- ICMP消息应该是可路由的，封装在IP分组(**协议号为1**)传递
 - 查询类消息：源发送ICMP请求到目的端，目的端回应以响应
 - 差错报告类：收到IP分组后触发，应该包含原IP分组头部+数据部分的前8字节(TCP头部中的端口号和顺序号，8个字节的UDP头部的端口号等信息)



ICMP消息

类型	代码	描述
0	0	Echo请求
8	0	Echo响应
13	0	Timestamp请求
14	0	Timestamp响应
10	0	路由请求
9	0	路由通告
3	0	目的地不可达/网络不可达
3	1	目的地不可达/主机不可达
3	2	目的地不可达/协议不可达
3	3	目的地不可达/端口不可达
3	4	目的地不可达/需要分段但DF置位
5	0-3	路由重定向
11	0	TTL为0超时
11	1	重组超时
12	0,1	IP头部参数错误

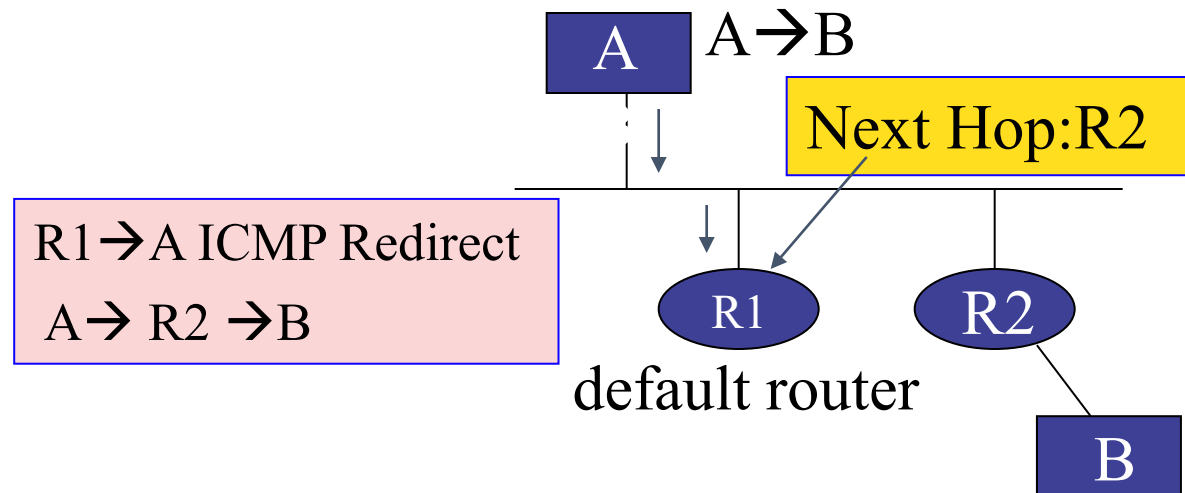


- ping host
 - 消息包括ID、顺序号和可选的数据部分
 - 测试节点之间是否可达
 - 测试RTT
 - 测试丢包情况
 - R选项使用IP选项纪录路由，可以记录途中经过的路由器，但最多9个路由器
- ping -R -c <count> -t <ttl> -s <packet size>
-R 记录路由选项
-c <count> 发送几个icmp echo请求后结束
-t <ttl> 发送的ICMP消息的TTL设置为<ttl>
-s <packet size> ICMP请求中的数据部分长度，缺省56字节

ICMP 差错报告

- 不是所有的IP分组在无法递交时都会触发ICMP差错报告
 - ICMP不报告ICMP消息本身的错误
 - ICMP不报告IP分组头部检验和错误
 - ICMP只报告IP分组的第一个分段的错误
 - ICMP不报告广播和组播分组的错误
- Type 3: 目的地不可达
 - Code 0: 网络不可达, 找不到对应路由表项
 - Code 1: 主机不可达, ARP请求没有响应
 - Code 2: 协议不可达, 目的地不支持IP头部中的protocol
 - Code 3: 端口不可达, 没有应用程序绑定在该端口上
 - Code 4: 需要分段但DF置位
- Type 5: 重定向, 通知到某个目的地有一条更好的路由

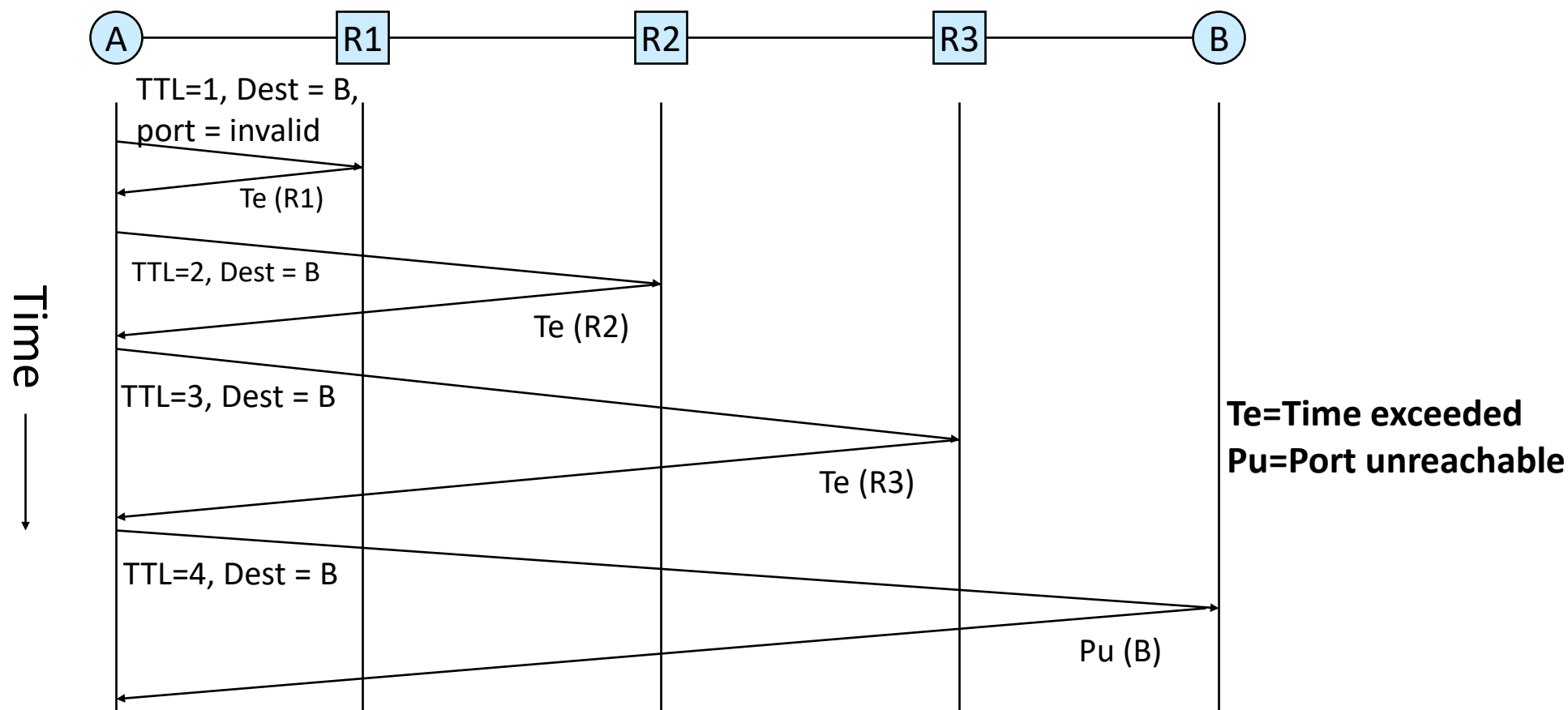
Dest	Next Hop
default	R1
B	R2



- A设置缺省路由器为R1
- A到B的分组被R1接收, R1查找路由表, 下一跳路由器为R2
- R1发现下一跳R2与源A在同一个接口上, 发送重定向分组给A, 通知其可以直接发送给R1**
- A收到后更新转发表, 下次直接发送给R2

ICMP: Traceroute

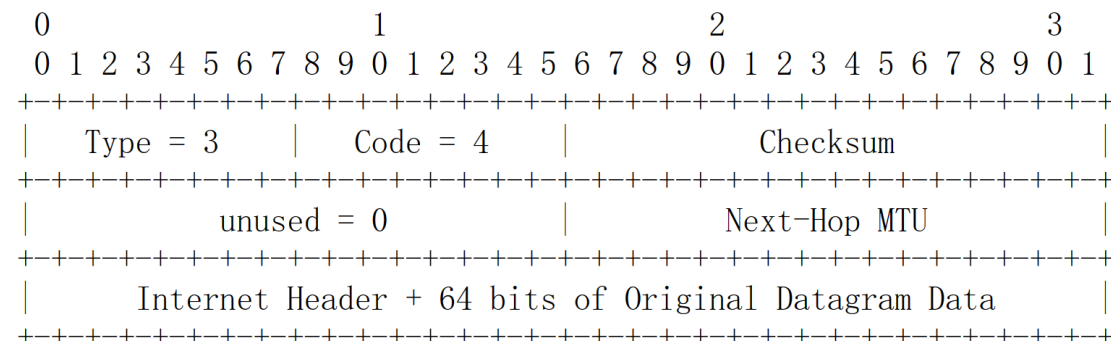
- Linux发送一个到某个UDP端口(超过33434的随机端口) 的消息, Windows发送ICMP Echo Request
- TTL从1逐步增加来探测第1、2...跳路由器
- 收到端口不可达信息(Linux实现)或者ICMP Echo reply结束



ICMP: Path MTU Discovery

分段是有害的，避免分段

- 发送较小的IP分组（MTU最少576字节）
- PATH MTU Discovery: 通过ICMP协议探测途中允许的MTU的下界，以后发送的IP分组都不超过该MTU。 **RFC 1191, Path MTU Discovery**
- Type3, Code 4: 需要分段但DF置位的不可达信息，也称为PTB(Packet Too Big)
 - Next-Hop MTU: 在无法进行分段时给出该链路上MTU
- 发送一个足够大的分组（DF标志为1）：可以取本地MTU和对方的MSS所计算的MTU的最小值
- 如果途中网络的MTU不够大，源节点将收到ICMP不能分段差错报告(PTB)
- 发送一个小一些的分组（DF标志为1），进一步探测是否可以到达目的地
 - 如果是新的PTB消息，则分组长度为其中给出的下一跳的MTU
 - 如果是早期的PTB消息，即对应的字段为0，则基于各种链路的MTU取值情况，尝试更小的分组长度
- 重复直到没有收到ICMP不能分段差错报告(PTB)为止



主要内容

网络层提供的服务：为高层提供节点到节点的传输，经过多跳传输最终到达目的地

5.1 交换和路由：internet的工作方式（虚电路和数据报）

5.1 交换和路由：路由方式

- 源路由和逐跳路由
- ~~扩散法~~
- ~~逆向学习法~~

~~5.2 网桥~~

5.3 Internet网络层

- IP协议
- ARP
- ICMP
- **DHCP**
- NAT
- IP隧道
- IP组播
- IPv6

动态主机配置协议DHCP (Dynamic Host Configuration Protocol)

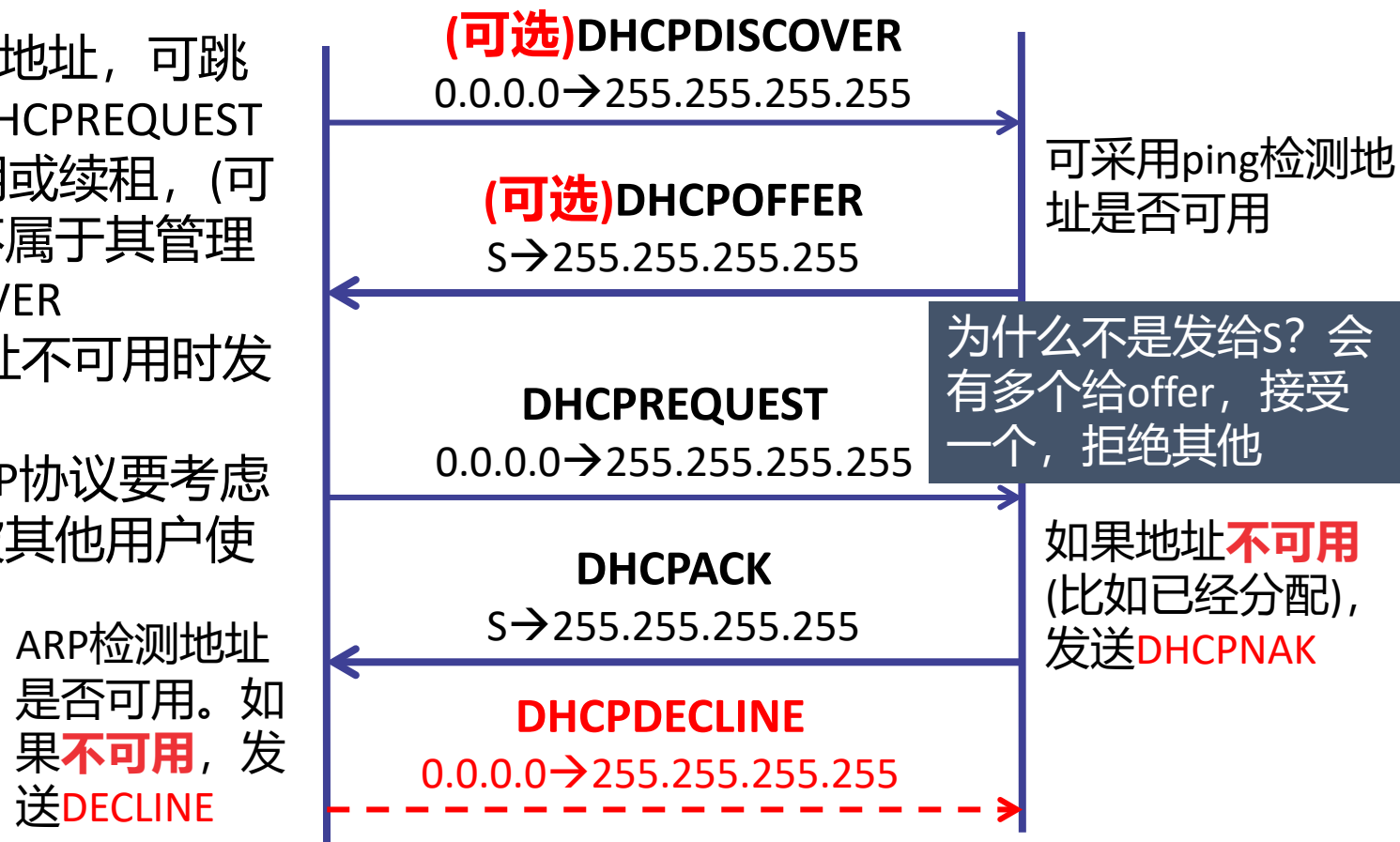
- RARP协议：
 - 节点知道自己的网卡地址，广播RARP请求询问其IP地址，RARP服务器回应以配置的IP地址
 - RARP采用**以太网广播**，无法跨越IP子网，仅能返回IP地址
- BOOTP协议(Bootstrap protocol)：为无盘工作站引导提供支持
 - **采用UDP协议**，允许跨越IP子网
 - 除了**分配IP地址**外，还可传递**引导所必须的信息**：操作系统映像所在的主机和位置
- DHCP：RFC 2131/2132定义，在**BOOTP基础上扩展**而来，采用C/S架构，采用UDP
 - 服务方端口号为67，客户方端口号为68
 - DHCP服务器可提供IP地址、掩码、缺省路由器、DNS服务器、缺省TTL等参数配置
 - 三种IP地址分配方法
 - 自动方法：根据唯一客户标识（比如网卡地址）**固定分配**
 - **动态方法：从地址池中租用一个**
 - 手工方法：IP地址通过第三方方式分配，DHCP用于配置其他参数

动态主机配置协议DHCP:DHCP工作过程

- DHCP client要从DHCP服务器动态获得IP地址
 - DHCP服务器维护了一个地址池
 - 收到请求后从DHCP地址池获得一个可用的地址分配给用户使用一段时间(lease time)
 - 在租期到来之前, DHCP Client可以进行续租
- DHCP Client希望能够再次使用上次的地址, 可跳过发送DISCOVER/OFFER, 直接发送DHCPREQUEST
- DHCPNAK: 拒绝DHCPREQUEST的租用或续租, (可能是地址已经分配, 可能是该地址不属于其管理的IP子网), 收到后广播DHCPDISCOVER
- DHCPDECLINE: 收到ACK但是发现地址不可用时发送, 重新DISCOVER
- 考虑到有些主机可能手工配置, DHCP协议要考虑地址池中的地址在试图租用时可能被其他用户使用的情形

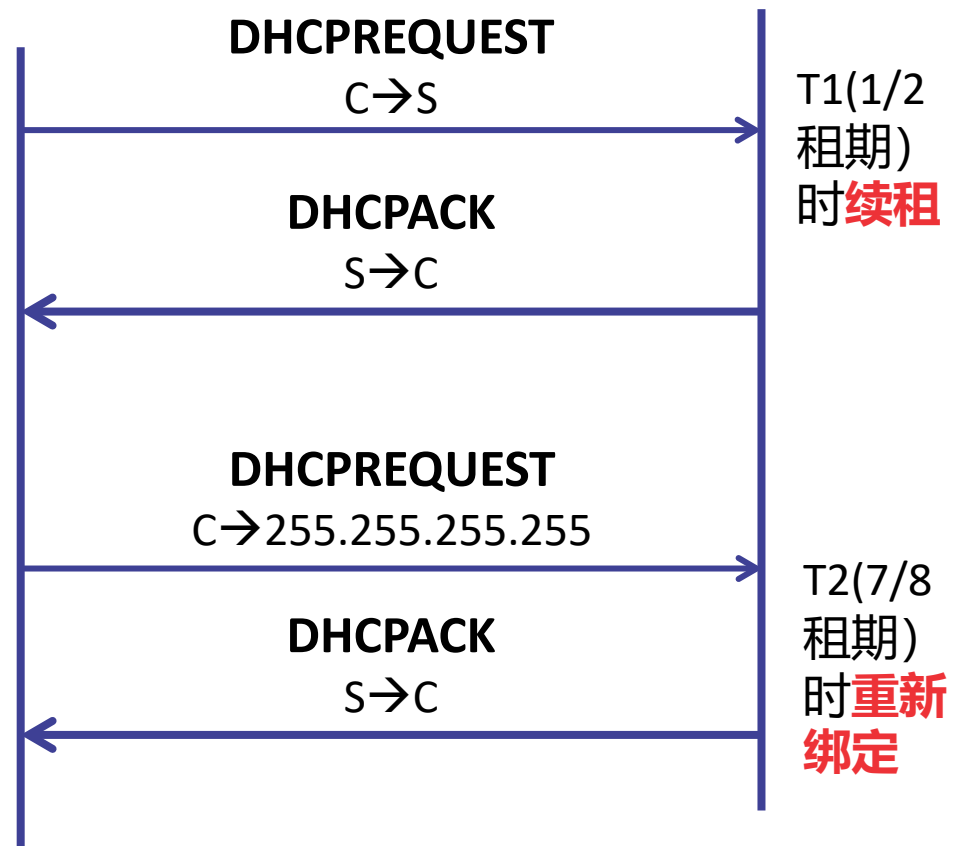
拓展: DHCPDISCOVER已包含了Client的MAC地址, 因此**DHCPOFFER/ACK/NAK也可采用单播方式**

- 如果DHCP Client在没有获得IP地址之前不能接收单播IP分组, 则可在其发送的DHCPDISCOVER消息中将flags设置为仅支持广播方式递交



动态主机配置协议DHCP:DHCP工作过程

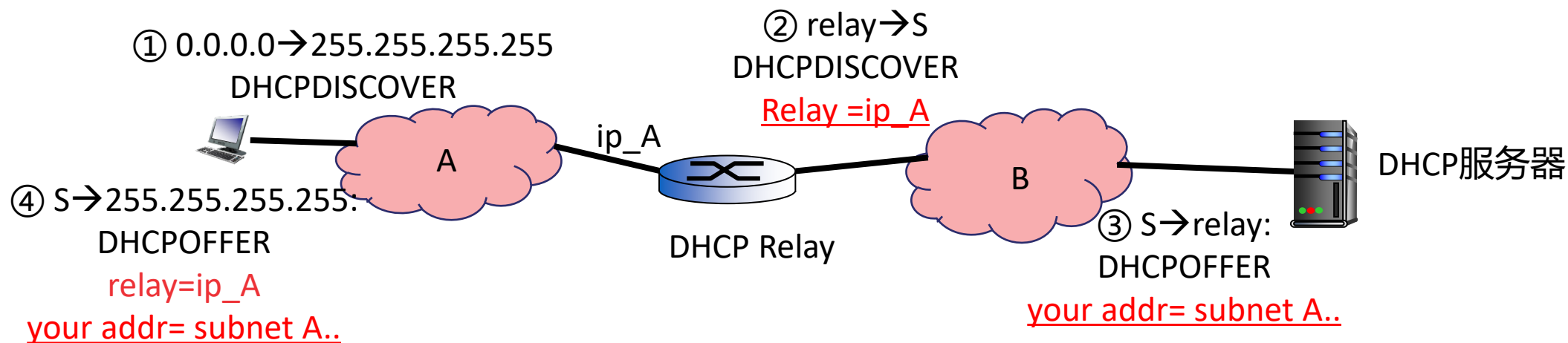
- DHCPACK: 确认租用(广播)/确认续租和确认重新绑定(单播)
- DHCPRELEASE: **不再租用时发送**
- DHCPINFORM: 用于获取IP地址外的配置信息



DHCP 中继代理

多个物理网络时，DHCPDISCOVER目的地址为本地广播(255.255.255.255)，缺省不会跨越路由器

- 一种实现：每个物理网络部署一个DHCP服务器
- 可引入**Relay Agent**，负责转发广播方式发送的DHCP消息到某个DHCP服务器，将DHCP服务器发送的响应通过广播方式转发给DHCP Client
 - 收到广播的DHCP消息时，如果中继IP地址(giaddr)字段为空，填充为当前Relay Agent的地址，并转发给DHCP服务器
 - DHCP服务器在收到giaddr字段不为空的DHCP消息时，对该DHCP消息的响应将发送给giaddr
 - DHCP服务器根据中继IP地址字段知道在哪个物理网络，从而分配合适的IP地址
 - Relay Agent收到服务器返回的DHCP消息之后通过广播方式转发给DHCP Client



DHCP 消息格式

- 操作码为0或1：请求和响应
- 跳段数：初始为0，中继代理转发时加1，检测回路
- 交易ID：请求和响应的匹配
- 中继IP地址：中继代理转发且该字段没有设置时填充
- 你的IP地址：服务器所分配的地址
- 客户方硬件地址：可用于标识客户方
- DHCP消息在BOOTP消息格式的基础上通过采用TLV描述的选项部分进行了扩展。其中类型为53的选项是必须的，描述了DHCP消息的类型

类型	长度	描述
53	1	消息类型(必需)
1	4	网络掩码
3	4,8,..	路由器地址
6	4,8,...	DNS服务器地址
50	4	DHCP请求地址
51	8	DHCP租期

操作码(请求/响应)	硬件类型	硬件地址长度	跳段数
交易ID			
时间(秒) client开始地址获取的时间		标志(最高位为广播bit)	
ciaddr客户方IP地址(已经合法拥有地址时填充)			
yiaddr你的IP地址(服务方为你分配的地址)			
(TFTP)服务方IP地址，下一步引导使用的IP地址(如OS映像所在地址)			
中继IP地址			
客户方硬件地址(16字节) 有时也可作为Client标识			
服务方主机名(可选， NULL字符结束， 最长64字节)			
引导文件路径名(可选， NULL字符结束， 最长128字节)			
选项(TLV)			

0.0.0.0 --> 255.255.255.255 DHCP Discover

Bootstrap Protocol (**Discover**)

Message type: Boot Request (1)

Hardware type: Ethernet (0x01)

Hardware address length: 6

Hops: 0

Transaction ID: 0xffa54c48

Seconds elapsed: 0

Bootp flags: 0x0000 (**Unicast**)

Client IP address: 0.0.0.0

Your (client) IP address: 0.0.0.0

Next server IP address: 0.0.0.0

Relay agent IP address: 0.0.0.0

Client MAC address: (74:e5:f9:df:e5:26)

Server host name not given

Boot file name not given

Magic cookie: DHCP

Option: (53) DHCP Message Type (**Discover**)

Option: (61) **Client identifier**

Client MAC address: (74:e5:f9:df:e5:26)

Option: (50) **Requested IP Address: 192.168.50.88**

Option: (12) Host Name: moon

Option: (60) Vendor class identifier

Option: (55) **Parameter Request List**

(1) Subnet Mask

(3) Router

(6) Domain Name Server

(15) Domain Name

...

Option: (255) End

192.168.50.1 --> 192.168.50.88 DHCP Offer

Bootstrap Protocol (**Offer**)

Message type: Boot Reply (2)

Hardware type: Ethernet (0x01)

Hardware address length: 6

Hops: 0

Transaction ID: 0xffa54c48

Seconds elapsed: 0

Bootp flags: 0x0000 (Unicast)

Client IP address: 0.0.0.0

Your (client) IP address: 192.168.50.88

Next server IP address: 192.168.50.1

Relay agent IP address: 0.0.0.0

Client MAC address: 74:e5:f9:df:e5:26

Server host name not given

Boot file name not given

Magic cookie: DHCP

Option: (53) DHCP Message Type (**Offer**)

Option: (54) **DHCP Server Identifier: 192.168.50.1**

Option: (51) **IP Address Lease Time: (86400s) 1 day**

Option: (58) Renewal Time Value: (43200s) 12 hours

Option: (59) Rebinding Time Value: (75600s) 21 hours

Option: (1) **Subnet Mask: 255.255.255.0**

Option: (28) Broadcast Address: 192.168.50.255

Option: (6) **Domain Name Server: 192.168.50.1**

Option: (3) **Router: 192.168.50.1**

Option: (255) End

0.0.0.0 --> 255.255.255.255 DHCP Request

Bootstrap Protocol (**Request**)

Message type: Boot Request (1)

Hardware type: Ethernet (0x01)

Hardware address length: 6

Hops: 0

Transaction ID: 0xffa54c48

Seconds elapsed: 0

Bootp flags: 0x0000 (**Unicast**)

Client IP address: 0.0.0.0

Your (client) IP address: 0.0.0.0

Next server IP address: 0.0.0.0

Relay agent IP address: 0.0.0.0

Client MAC address: (74:e5:f9:df:e5:26)

Server host name not given

Boot file name not given

Magic cookie: DHCP

Option: (53) DHCP Message Type (**Request**)

Option: (61) **Client identifier**

Client MAC address: (74:e5:f9:df:e5:26)

Option: (50) **Requested IP Address: 192.168.50.88**

Option: (54) **DHCP Server Identifier: 192.168.50.1**

Option: (12) Host Name: moon

Option: (60) Vendor class identifier

Option: (55) **Parameter Request List**

(1) Subnet Mask

(3) Router

(6) Domain Name Server

(15) Domain Name

...

Option: (255) End

192.168.50.1 --> 192.168.50.88 DHCP ACK

Bootstrap Protocol (**ACK**)

Message type: Boot Reply (2)

Hardware type: Ethernet (0x01)

Hardware address length: 6

Hops: 0

Transaction ID: 0xffa54c48

Seconds elapsed: 0

Bootp flags: 0x0000 (Unicast)

Client IP address: 0.0.0.0

Your (client) IP address: 192.168.50.88

Next server IP address: 192.168.50.1

Relay agent IP address: 0.0.0.0

Client MAC address: 74:e5:f9:df:e5:26

Server host name not given

Boot file name not given

Magic cookie: DHCP

Option: (53) DHCP Message Type (**ACK**)

Option: (54) **DHCP Server Identifier: 192.168.50.1**

Option: (51) **IP Address Lease Time: (86400s) 1 day**

Option: (58) Renewal Time Value: (43200s) 12 hours

Option: (59) Rebinding Time Value: (75600s) 21 hours

Option: (1) **Subnet Mask: 255.255.255.0**

Option: (28) Broadcast Address: 192.168.50.255

Option: (6) **Domain Name Server: 192.168.50.1**

Option: (3) **Router: 192.168.50.1**

Option: (255) End

主要内容

网络层提供的服务：为高层提供节点到节点的传输，经过多跳传输最终到达目的地

5.1 交换和路由：internet的工作方式（虚电路和数据报）

5.1 交换和路由：路由方式

- 源路由和逐跳路由
 - ~~扩散法~~
 - ~~逆向学习法~~
- IP协议
 - ARP
 - ICMP
 - DHCP
 - **NAT**
 - IP隧道
 - IP组播
 - IPv6

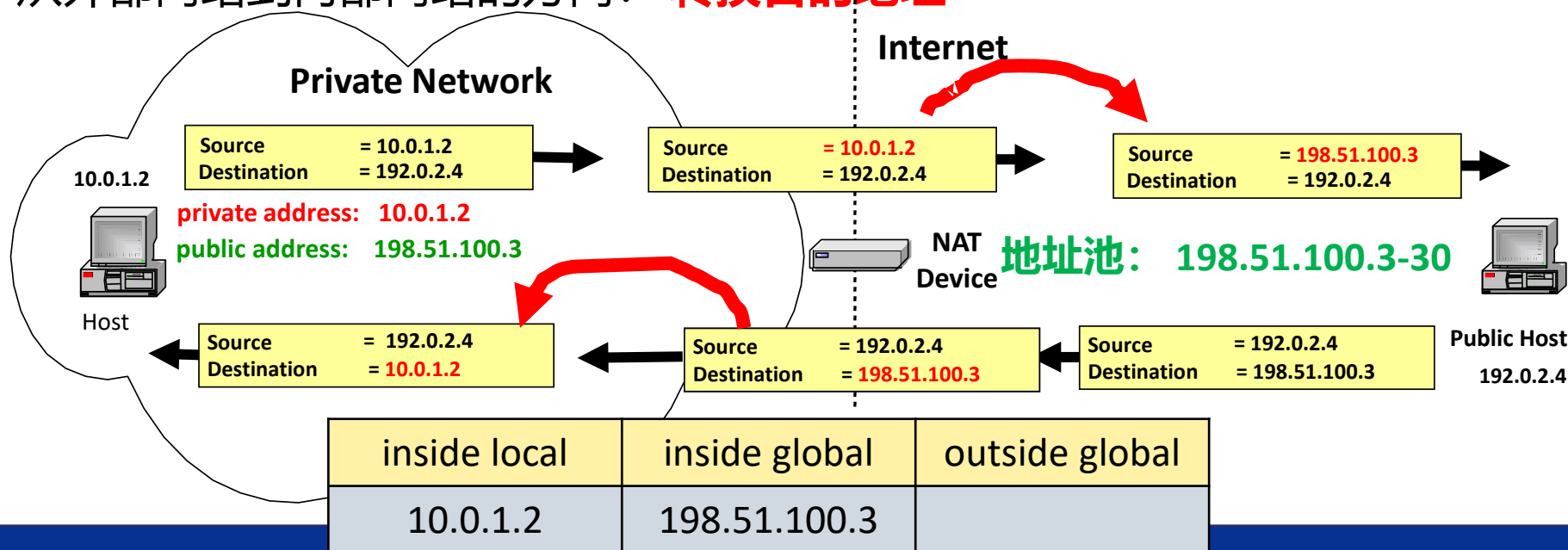
5.2 网桥

5.3 Internet网络层

网络地址转换NAT (Network Address Translation)

内部网络中的主机不会要求在同一时刻全部连接到Internet

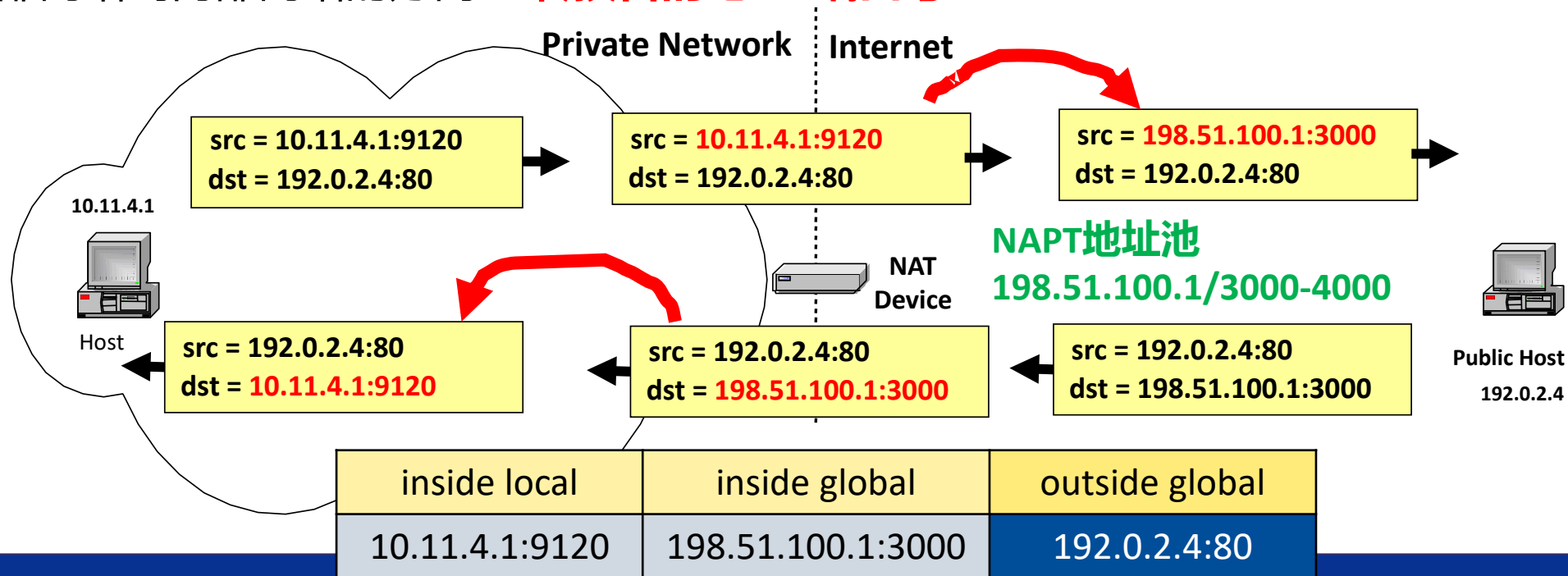
- 内部网络中主机可采用内部IP地址，甚至可采用其他合法的IP地址
 - RFC1918 给出了A、B、C类内部IP地址，分别为10.0.0.0/8、172.16.0.0/12、192.168.0.0/16
 - RFC6598分配shared address 100.64.0.0/10，供carrier-grade NAT使用
- 基本NAT：负责在内部网络用到的**内部IP地址**与外部用于连接到Internet的**公共IP地址**(从地址池中选择)之间进行地址转换
 - 从内部网络到外部网络的方向：**转换源地址**
 - 从外部网络到内部网络的方向：**转换目的地址**



网络地址转换NAT

基于端口的NAT (NAPT) :

- 针对主机的每个会话 (TCP连接、UDP或者ICMP会话) 进行映射
- 一个公共IP地址可以有65535个端口号, 即6万多条TCP连接
- 内部主机上的应用 (某个端口) 被映射为某(几)个外部IP地址+端口号
- 从内部网络到外部网络的方向: **转换源地址+端口号**
- 从外部网络到内部网络的方向: **转换目的地址+端口号**



网络地址转换NAT

- 自动添加：内部网络中的主机发送的分组到达NAT设备时，从地址池中选择一个可用地址或者可用地址加上端口号
- 手工添加静态映射：
 - 端口重定向：比如外部网络的主机访问198.51.100.1上的HTTP服务时，定向到10.11.4.1的8080端口
 - 支持负载均衡：外部网络的主机访问某个地址(198.51.100.2)的某个服务时可定向到多个地址+端口之一

		inside local	inside global	outside global
端口重定向		10.11.4.1:8080	198.51.100.1:80	*
负载均衡	{	10.11.5.1:80	198.51.100.2:80	192.4.4.4
		10.11.5.2:80	198.51.100.2:80	192.5.5.5

通过与NAT设备的交互更新NAT转换表

- UPnP (Universal Plug and Play) ， 基于SSDP/SOAP和HTTP协议， 端口号1900
 - 位于内部网络的主机通过该协议与NAT设备通信来添加NAT映射（内部IP地址+端口号到公共IP地址和端口号）
 - 将映射后的公共IP地址和端口号广而告之其他主机
- 苹果公司提出的NAT-PMP协议(RFC 6886 NAT Port Mapping Protocol); RFC 6887 Port Control Protocol

网络地址转换NAT

下面两种NAT较少使用，不进行介绍

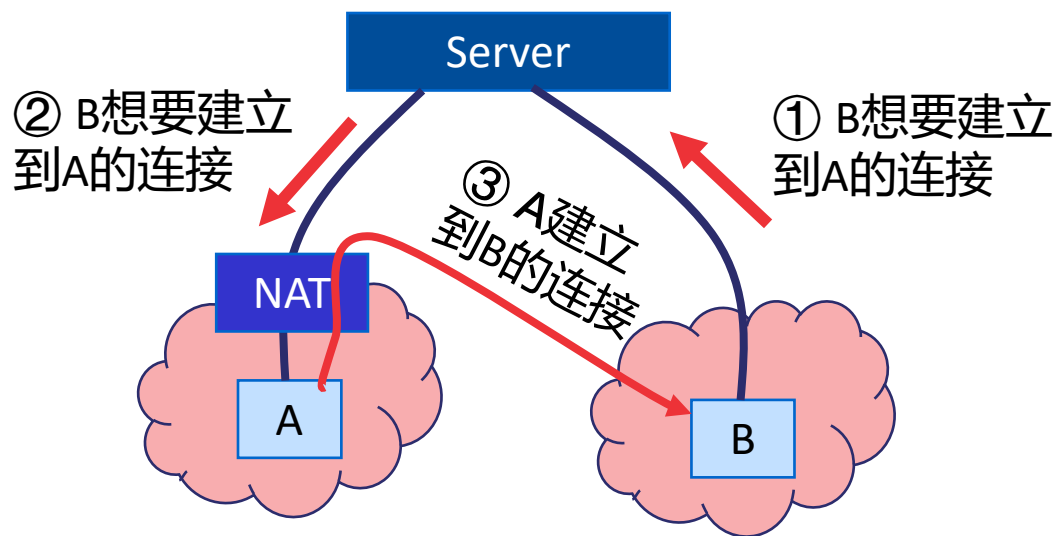
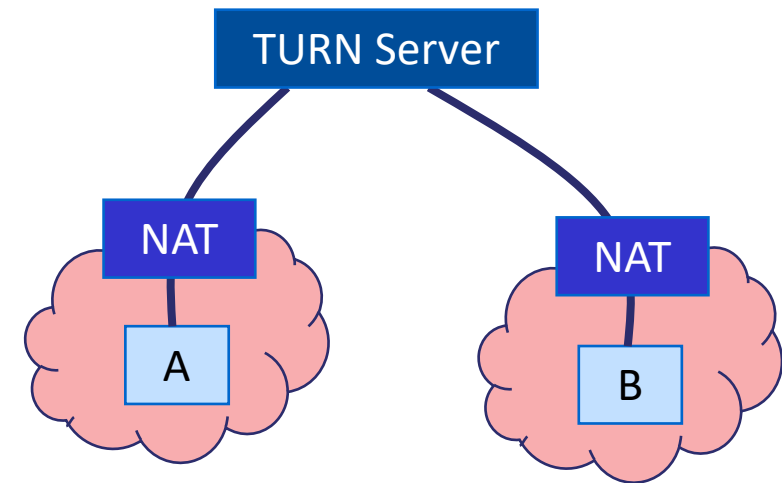
- 双向(Bi-Directional) NAT: 域名服务器也在内部网络，DNS响应给出的是服务器的内部地址。引入应用层网关DNS-ALG，更改DNS响应中的地址
- 两次NAT(Twice-NAT): **内部网络和外部网络中的IP地址空间有重复**时，这些具有重复IP地址的主机之间的分组的源地址和目的地址可能都需要进行转换

网络地址转换NAT：优缺点

- NAT的优点：
 - 只需要少量的Internet唯一的公网地址，缓解IP地址空间不足
 - 大部分情况对于主机透明，也不用关心NAT所使用的公网地址是什么
 - 额外的安全保障，除非NAT表有相应的记录，内部网络的主机无法被外网访问
 - 可支持端口重定向、负载均衡等
- NAT的问题
 - 违背了端到端和无状态路由的设计原则，采用跨层设计
 - NAT设备成为性能的瓶颈和可能的攻击目标
 - 有些应用和协议需要应用层网关的帮助才能穿越NAT
 - 许多应用依赖于IP服务模型的一个关键假设：所有节点都有一个唯一的全球IP地址
 - 应用层协议（比如FTP）消息里面包含了IP地址和端口号时有问题
 - 应用层使用的多条TCP连接之间有关联时有问题
 - 如果通信的双方都位于NAT设备之后，要采用相应的NAT穿越技术

拓展: NAT Traversal

- TURN(Traversal Using Relays around NAT) protocol
 - A和B之间的通信通过TURN服务器中转
 - A和B都可位于NAT之后, 但是TURN服务器成为性能瓶颈
- Connection Reversal (逆向连接)
 - 通信的A和B中只有一个主机在NAT之后使用, 假设A在NAT设备之后, B在Internet
 - A和B都连接到某个服务器S
 - B无法主动发起到A的会话, 但可通过S告知A, 由内部网络中的A主动发起到位于Internet的B的通信



拓展: NAT Traversal

不同类型的NAT设备为内部网络提供不同类型的洞(hole)

inside local	inside global	outside global
10.11.4.1:4444	198.1.1.1:2222	200.5.6.7:80

- **Full cone NAT(不检查outside global)**
 - 内网中的主机上的应用通过socket访问外网时打了一个洞inside local/port \leftrightarrow inside global/port。同一个socket来的到不同目的地的分组都是通过同一个洞
 - 外网中的任何主机都可以通过打开的洞(inside global/port)进入(inside local/port)
- **restricted cone NAT: (检查outside global的IP地址, 但是不需要匹配outside global的port)**
 - 和Full cone NAT类似, 只是只有内网主机上的socket前面打了洞到达该外部主机时, 才允许该外部主机(的另一个端口)通过该洞进入
- **port restricted cone NAT: (检查outside global ip/port)**
 - 和restricted cone NAT类似, 但只有内网主机上的socket前面打了洞通向该外部主机上的socket时, 才允许该外部主机的socket通过该洞进入
- **symmetric NAT: 无法打洞, 只能通过TURN服务器中转**
 - 来自于同一个内部主机和端口的到不同的外部主机或者端口的分组映射为其他源地址或者端口



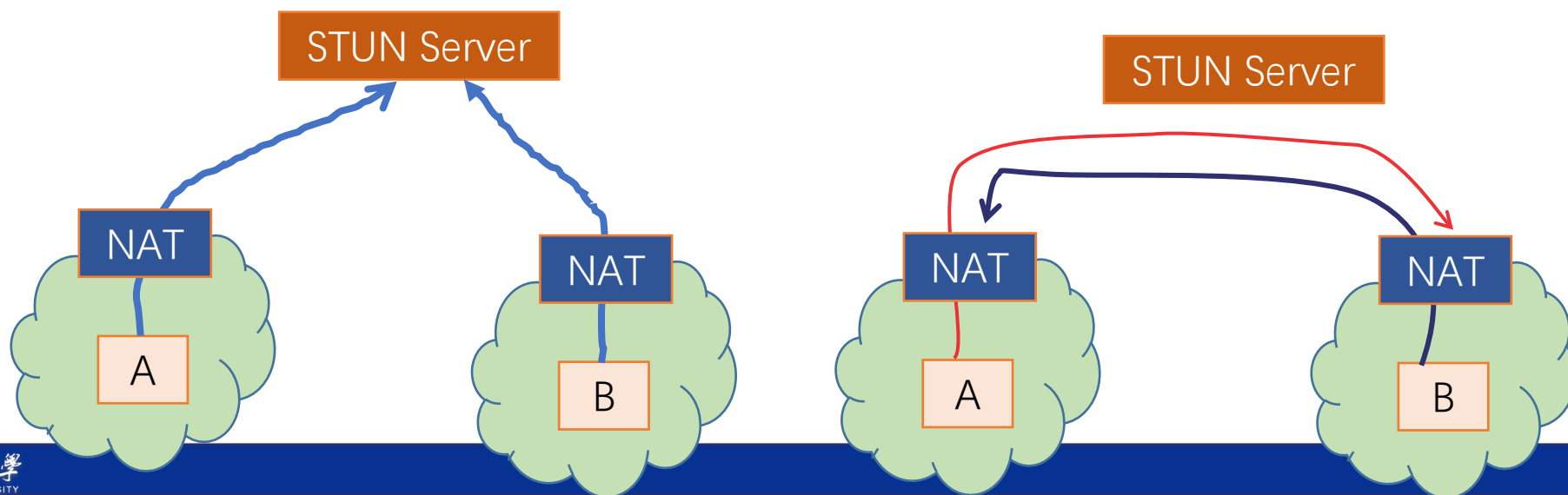
拓展: NAT Traversal

RFC 5389 STUN(Session Traversal Utilities for NAT)

- A和B发送消息给外网的STUN服务器(比如stun.stunprotocol.org, stun.sipgate.net), STUN服务器可以看到最后一个NAT设备的洞(inside global/port)
- A和B定期发送keepalive消息给STUN服务器, 保证打的洞一直存在
- A和B之间尝试通过前面打的洞看是否可以进入相应的NAT

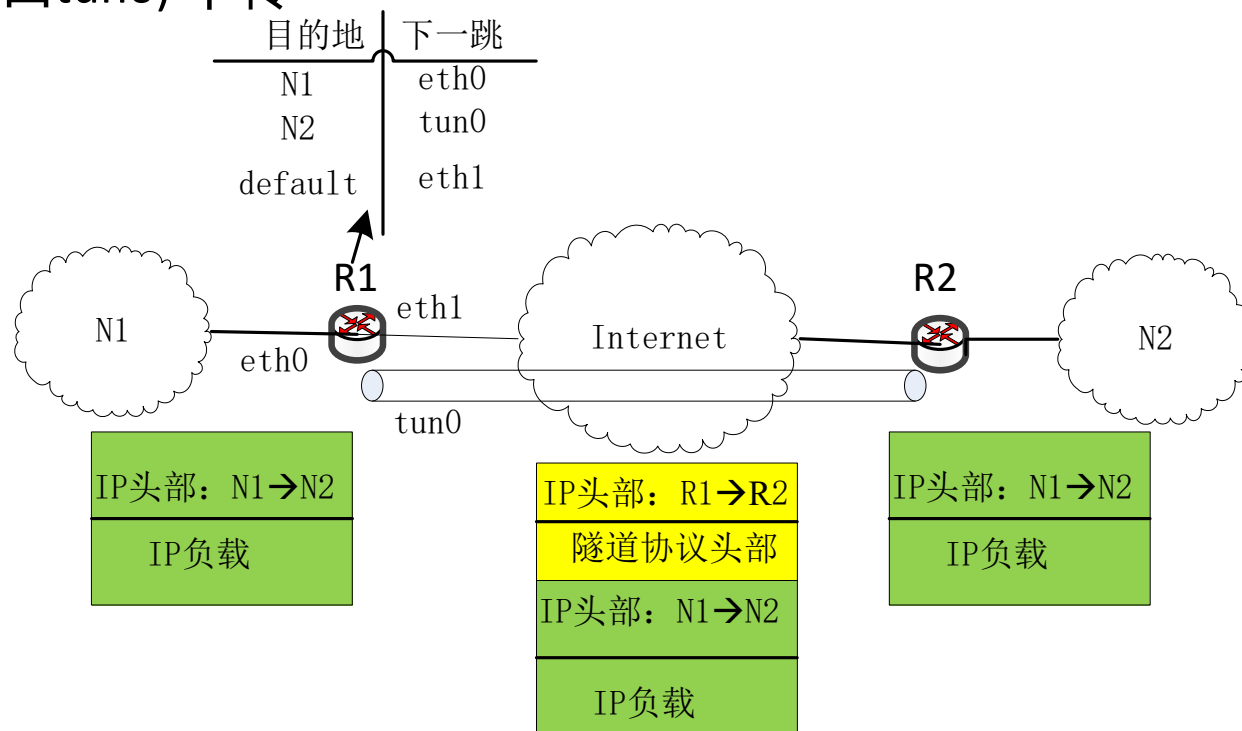
即便是**port restricted cone NAT**

- $A \rightarrow B$'s Hole: 打开A的门, 允许来自于B's Hole的分组进入NAT A
- $B \rightarrow A$'s Hole: 打开B的门, 允许来自于A's Hole的分组进入NAT B



IP隧道(tunnel)

- 采用IP协议的网络上的任意两个节点之间建立的一条**虚拟链路**（虚拟接口tun）
- 通过这些虚拟链路连接的节点构成了一个**覆盖网**（Overlay Network）
- 隧道参数：隧道的两端以及隧道所采用的封装方式
- 隧道可由管理人员手工配置，也可通过一个隧道建立协议自动建立
- N1和N2之间的负载通过R1与R2之间的隧道(虚拟接口tun0) 中转
- N1→N2的分组通过eth0到达R1时，查找路由表知道其应该通过tun0隧道转发
- tun0隧道入口进行封装，创建R1→R2的隧道分组，查找路由表，通过eth1发送到Internet
- R2收到隧道分组，解包得到N1→N2的分组，转发给N2



IP隧道：隧道封装协议

S→D, protocol=4(IP-in-IP)
原有IP分组

- IP-in-IP：RFC2003定义
 - 原有的IP分组作为IP分组的数据部分，源和目的地址分别为隧道的两端，协议号为4
 - 从隧道出来之后：原有IP分组头部除了其TTL减一和必要的校验和的重新计算外，其他头部字段都不变
- GRE（Generic Routing Encapsulation）在RFC2890定义
 - 为任何网络层协议之上传输任何协议的分组提供封装支持
 - 用于IP网络时，IP分组头部协议字段为0x47表示GRE分组
- 版本为0、1、2，表示GRE、PPTP和L2TP
- 协议类型：封装什么分组，等同Ethernet帧的协议字段，如0x0800表示IP分组，而0x880b表示PPP帧等
- C/K/S表示后面可选字段是否采用
- 校验和(checksum)：保护GRE头部和封装分组
- 关键(key)：可用于标识隧道上的分组流
- 顺序号(seq)：
 - 第一个分组顺序号为0
 - 检测丢失和失序到达

S→D, protocol=0x47(GRE)
GRE头部 (protocol=0x0800, IP协议)
原有IP分组

0	3	12	15	31
C	K	S	保留0	版本
校验和（可选）				协议类型
保留1（可选）				
关键（可选，4字节）				
顺序号（可选，4字节）				

主要内容

网络层提供的服务：为高层提供节点到节点的传输，经过多跳传输最终到达目的地

5.1 交换和路由：internet的工作方式（虚电路和数据报）

5.1 交换和路由：路由方式

- 源路由和逐跳路由
- ~~扩散法~~
- ~~逆向学习法~~

• IP协议

• ARP

• ICMP

• DHCP

• NAT

• IP隧道

• **IP组播(5.5)**

• IPv6

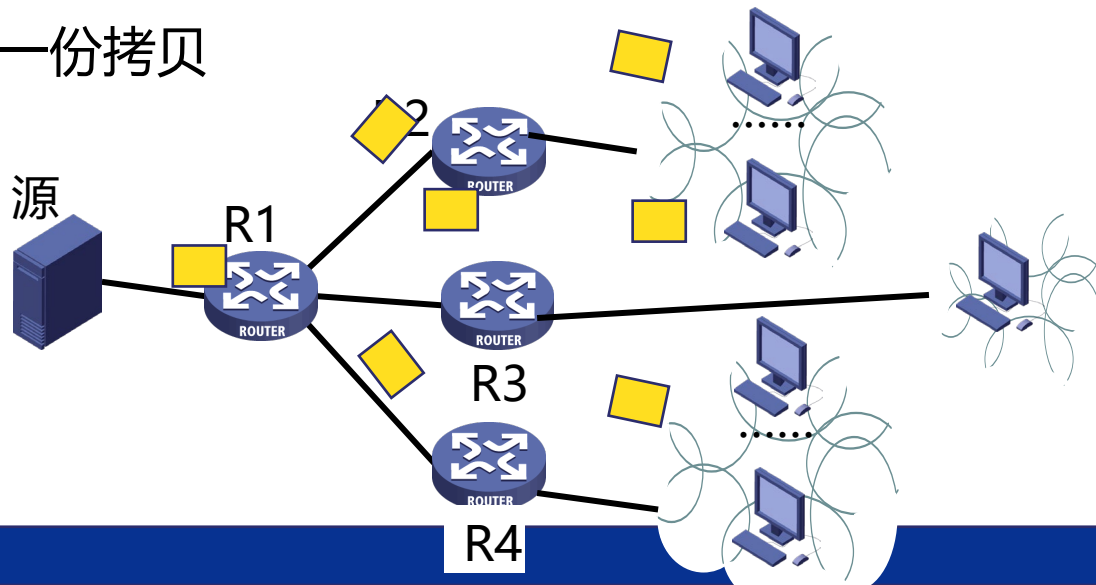
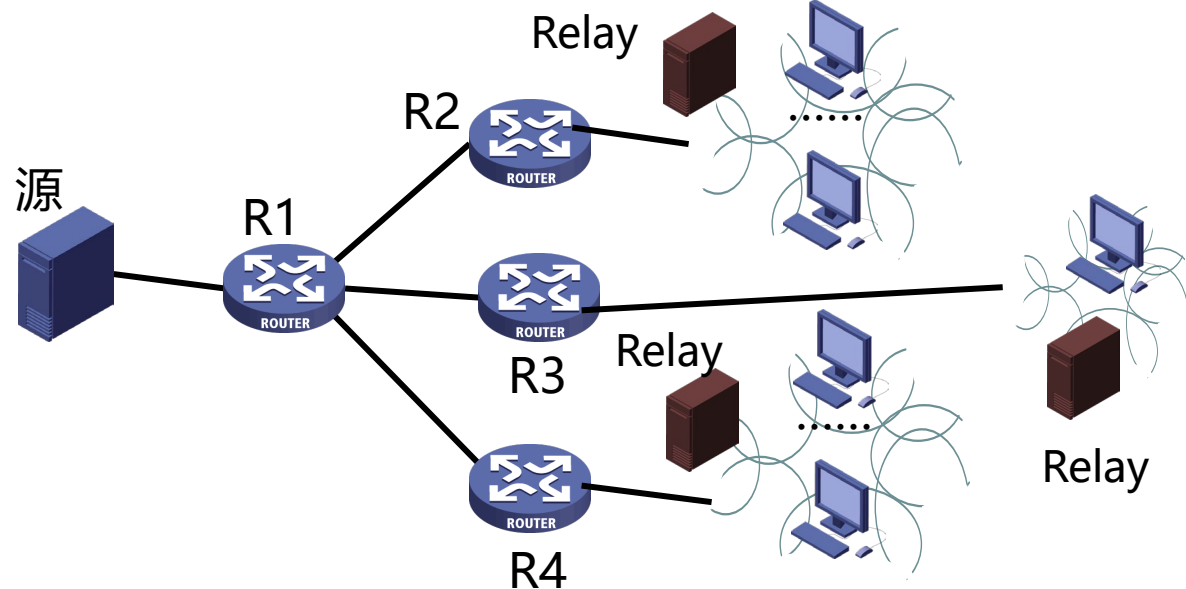
5.2 ~~网桥~~

5.3 Internet网络层

IP组播

组播应用：一对多或者多对多

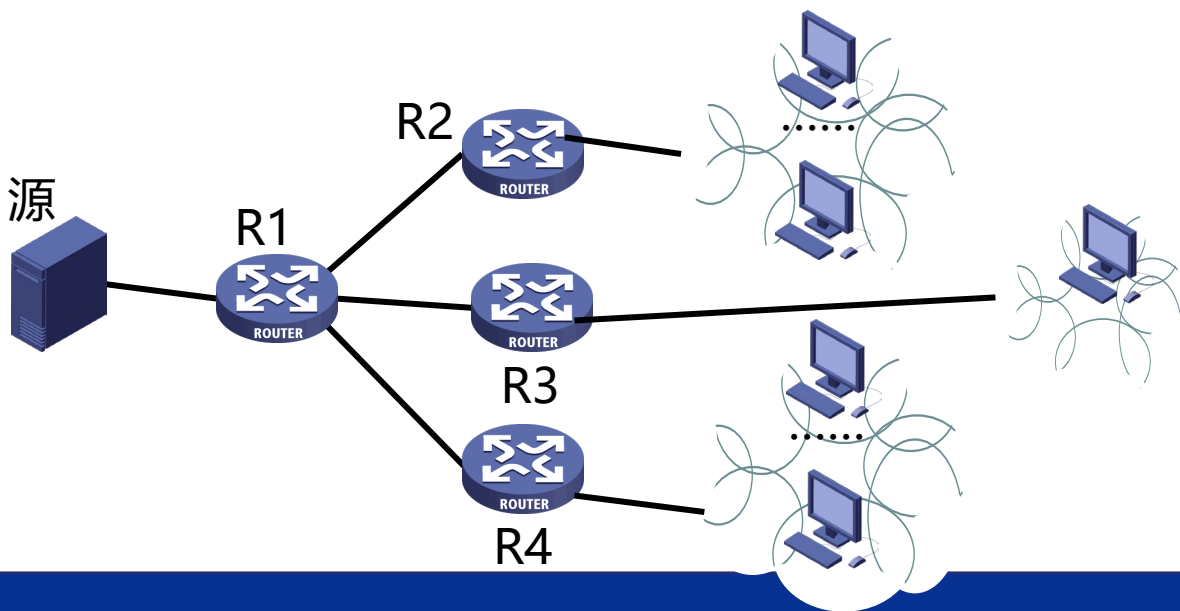
- 采用广播或由发送者给每个接收者单播的方式
- **应用层组播**：
 - 不是由源直接单播给所有接收者
 - 引入一个或者多个中继节点
 - 源单播发送给中继节点。每个中继节点再采用单播方式发送数据给其服务的那部分接收者
- **IP组播**（IP Multicast）：
 - **网络层提供组播支持**，允许1个或多个发送者发送单一的IP分组到特定的多个接收者
 - 组播路由器保证每个链路上最多只会有分组的一份拷贝



IP组播：组播服务模型

1989年斯坦福大学的S. Deering在RFC 1112定义了**任意源组播ASM** (Any Source Multicast) 服务模型

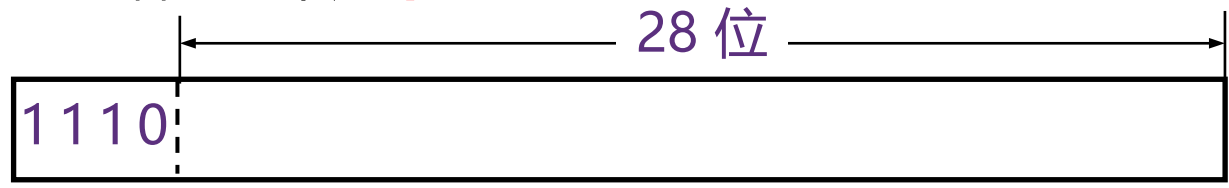
- **发送者没有限制**，可不属于组播组，组装一个组播帧（目的地址为组播地址）发送就可以了
- 接收者属于同一个组播组，**通过组播地址标识**
- 组播组可任意规模，接收者可在Internet上**任何地方，动态加入或退出**
- 组播路由器负责将组播分组转发到所有成员所在的网络上，保证在任意一个网络上最多传输一次
 - 要求组播路由器知道其所连接的接口方向是否有成员存在，这样只需要往有成员存在的方向转发就可以了
 - 组播路由器间维护一个**组播分发树**，组播分组沿着该分发树发送



IP组播地址

成员可位于Internet的任何地方，用于表示组播组的IP组播地址采用**平坦地址**结构

- 永久组播地址：路由和拓扑发现
 - 由IANA分配，范围为224.0.0.0至224.0.0.255
 - 本地唯一（Link-Local），TTL取值为1，不被转发到其他链路
- 临时组播地址：
 - 需要保证在一定的范围内没有其它的组播组使用同一地址：
 - 本地管理(私有) 组播地址，地址范围为239.0.0.0 ~ 239.255.255.255
 - 类似于单播IP地址空间的内部地址，只在内部网络中使用
 - 内部网络的出口路由器保证该组播分组不会被传播到Internet之上



D 类地址（组播地址）
224.0.0.0 ~ 239.255.255.255

如何为应用分配组播地址？

组播地址	描述	组播地址	描述
224.0.0.1	所有主机	224.0.0.6	OSPF DR路由器
224.0.0.2	所有路由器	224.0.0.9	RIP2路由器
224.0.0.4	DVMRP路由器	224.0.0.22	IGMPv3组播路由器
224.0.0.5	OSPF路由器	224.0.0.251	mDNS服务器(UDP port 5353)

IP组播地址

成员可位于Internet的任何地方，用于表示组播组的IP组播地址采用**平坦地址**结构

- 224.0.0.0/24 限定为链路
- 239.0.0.0/8 限定为内部网络
- 组播分组头部的TTL字段可用于限定组播分组的传输范围
 - TTL为0限制主机范围，TTL为1限制在链路上
 - 组播路由器可配置一个TTL阈值，如果收到的组播分组的TTL字段取值小于或者等于该阈值，则不转发
 - **扩展环搜索**（Expanding Ring Search）决定最合适的TTL取值
 - 发送一个TTL为1的组播分组，如果没有任何响应，则尝试TTL为2，一直继续下去。最远成员回应结束

IP组播分组在链路上的传输

- 点到点链路上与单播分组一样

- 广播链路：

- IP组播地址映射为链路层组播地址：**直接映射**（多对一）
 - OUI(24比特)=01:00:5E
 - 第25比特为0，低23比特=IP组播地址的低23比特→ $2^{(28-23)} = 32$ 个IP组播地址映射到同一个MAC地址

- 发送：组装成链路层组播帧后发送

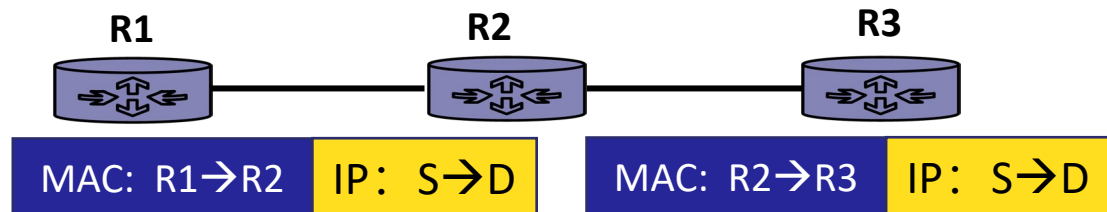
- 接收：

- 组播路由器接收所有组播分组

- 组播组的成员主机：

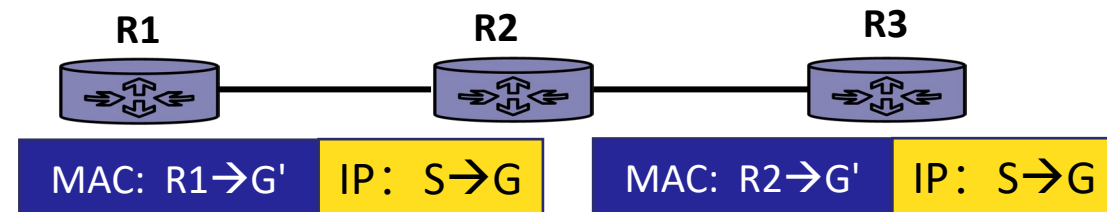
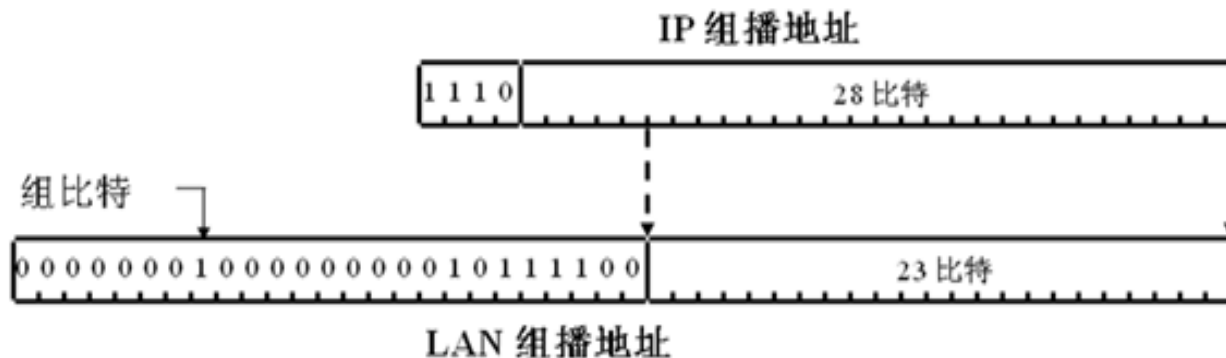
- **setsockopt**告知IP模块加入到组播组G
- 告诉网卡要求监听相应的链路层组播帧
- IGMP协议：主机与组播路由器间协议

```
group = socket.inet_aton(multicast_group)
mreq = struct.pack('4sL', group, socket.INADDR_ANY)
sock.setsockopt(socket.IPPROTO_IP, socket.IP_ADD_MEMBERSHIP, mreq)
```



单播分组的转发

R2: 根据D查找转发表, 下一跳为R3, 通过ARP获得R3的MAC地址, 转发给R3

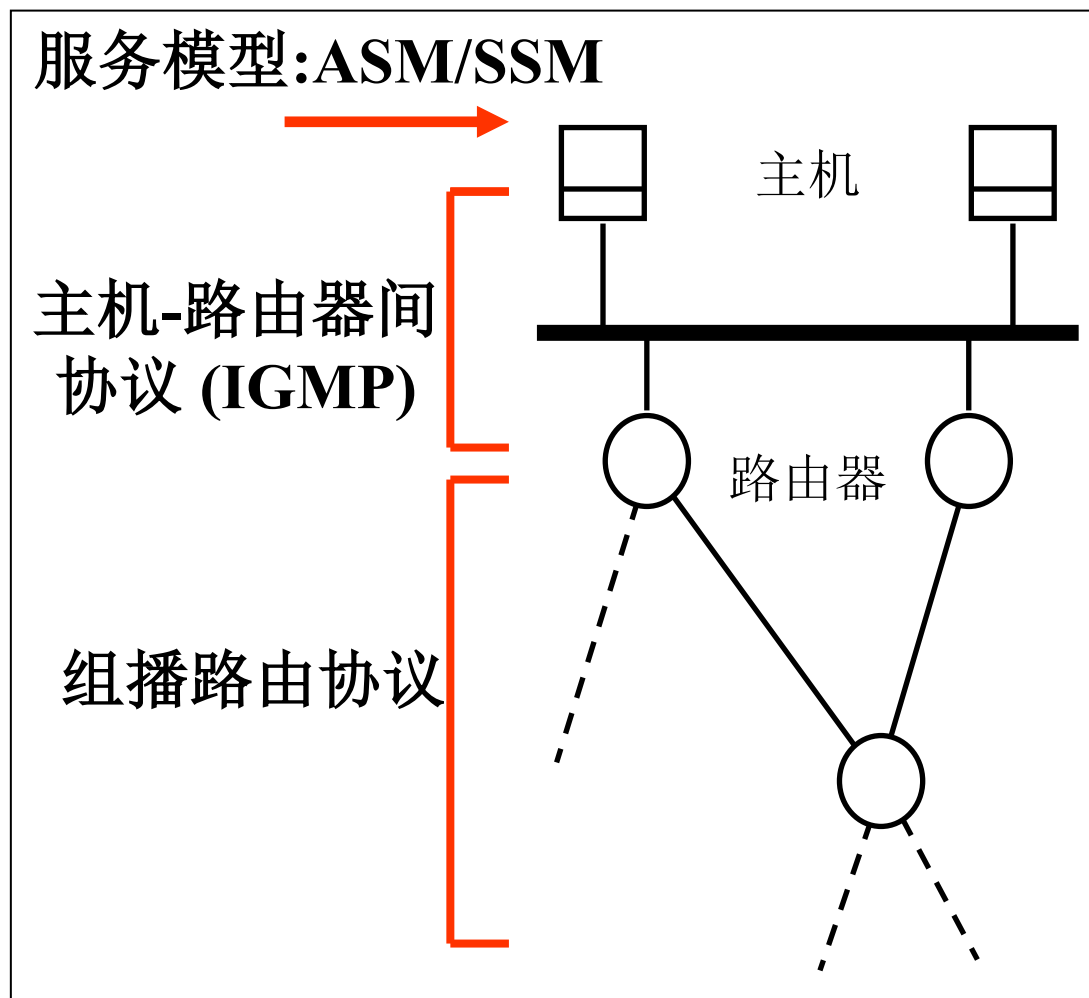


组播分组的转发

R2: 根据G查找转发表, 决定往哪些端口转发, 根据G得到其链路层组播地址G', 通过链路层组播转发到相应链路

IP组播体系结构：单一源组播(SSM)服务模型

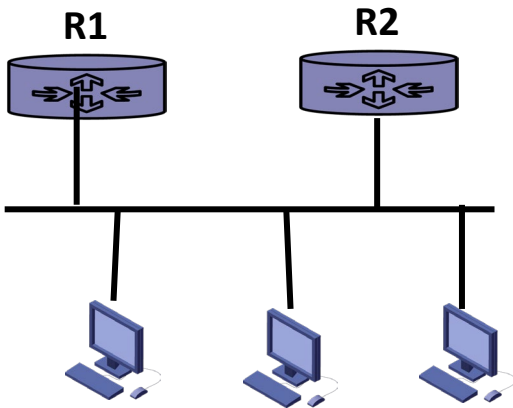
- **任意源组播ASM** (Any Source Multicast) 服务模型
 - 发送者没有限制，可不属于组播组
 - 接收者属于同一个组播组，**通过组播地址标识**
 - 组播组可任意规模，接收者可在Internet上**任何地方，动态加入或退出**
 - 组播路由器间维护**组播分发树**，组播分组沿着分发树发送
- **单一源组播SSM**(Single Source Multicast)模型：
 - 只有一个发送者
 - 一个组播会话通过<**source, group**>标识，避免了原ASM模型中组播地址分配的问题
 - 组播路由维护一个**基于源的分发树**，更加简单和高效
- 如果要限制接收分组的成员(提供付费服务)
 - 对于内容加密？
 - 不采用组播？



组成员关系协议IGMPv2 (Internet Group Management Protocol)

RFC 2236定义了IGMPv2协议，用于任意源服务模型中的IP组播

- IGMP协议工作在主机-路由器之间，建立在IP上，协议号为2
- 组播路由器通过IGMP协议了解其所连接的那些链路上是否有成员属于组播组
- 每个组播路由器定期（60到90秒）发送Query消息给所有主机 (224.0.0.1)
 - 最小IP地址的组播路由器充当询问者，其他停止发送Query
 - 收到Query后主机发送Report到其所属的组播地址G
 - 反馈抑制：Query包含最大响应时间，主机选择[0,Max Resp. Time]时刻准备发送Report
 - 主机在加入组播组时主动发送Report
 - 软状态机制：定期发送Query以刷新状态；超时时移走状态信息
- 离开时：可悄悄离开，但会浪费带宽
 - 最近一次组成员询问中发送report消息的主机发送Leave Group消息给所有组播路由器(224.0.0.2)
 - Querier发送指定了组播组的Query消息来询问对于特定的组播组G是否还有其他成员属于该组播组



General Query	Router→224.0.0.1	定期询问有成员属于哪些组
Report	Host → G	我在G，其他人不用发了
Leave Group	Host → 224.0.0.2	上次报告的我走了，你再问问有没有其他成员在G中
Specified Query	Router → G	还有成员在G吗？

组成员关系协议IGMP消息格式

RFC 3376给出了IGMPv3规范，支持单一源组播SSM (Single Source Multicast)

- 组播组通过 (源,组播地址) 标识
- Query消息：源个数 + 源IP地址列表
- IGMPv3Report：指出要过滤和允许接收的源IP地址列表，目的地址为224.0.0.22，即IGMPv3-capable组播路由器

V
P
M
P
I

3

类型	最大响应时间	检验和
组播地址		
标志	源个数	
源IP地址1		
...		
源IP地址N		

IGMP Query消息

组记录

类型	保留	检验和
保留		组记录个数(M)
记录类型	Aux数据长度	源的个数(N)
组播地址		
源地址[1]		
源地址[2]		
...		
源地址[N]		
辅助(Aux)数据		
...		
组记录[M]		

IGMP v3 Report

主要内容

网络层提供的服务：为高层提供节点到节点的传输，经过多跳传输最终到达目的地

5.1 交换和路由：internet的工作方式（虚电路和数据报）

5.1 交换和路由：路由方式

- 源路由和逐跳路由
- ~~扩散法~~
- ~~逆向学习法~~

5.2 ~~网桥~~

5.3 Internet网络层

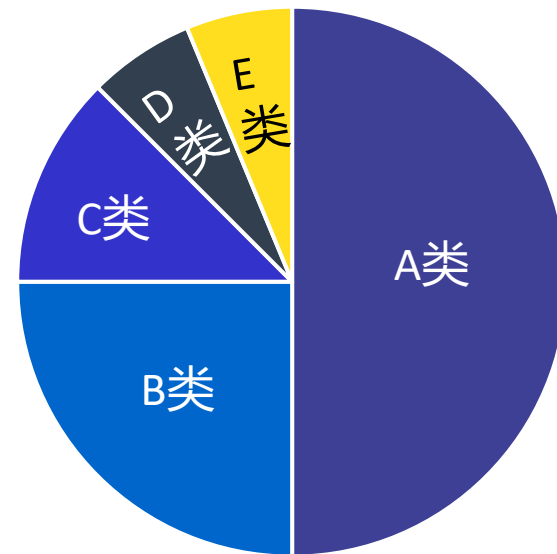
- IP协议
- ARP
- ICMP
- DHCP
- NAT
- IP隧道
- IP组播
- **IPv6(5.8)**

IPv6

- 2017年11月26日，中共中央办公厅、国务院办公厅印发了《推进互联网协议第六版（IPv6）规模部署行动计划》
 - 互联网演进升级的必然趋势: IPv4地址消耗殆尽、服务质量难以保证
 - 技术产业创新发展的重大契机: 移动互联网、物联网、工业互联网、云计算、大数据、人工智能等新兴领域
 - 网络安全能力强化的迫切需要
- 用5到10年时间，形成下一代互联网自主技术体系和产业生态，建成全球最大规模的IPv6商业应用网络，实现下一代互联网在经济社会各领域深度融合应用，成为全球下一代互联网发展的重要主导力量

IPv4地址匮乏

- 尽管有43亿个IP地址，但是有些地址保留，有些地址早期无序分配
- 2011年2月IANA将所有的/8地址分配完毕
- 2011年4月APNIC的IPv4地址块也分配完毕
- 2012年9月(欧洲)RIPE的IPv4地址块分配完
- 2014年6月(拉美)LACNIC地址池也分配完
- 2015年9月(美国)ARIN的地址块分配完
- (非洲)AfriNIC2018年上半年的地址块也分配完
- 区域注册机构(RIR)的地址分配完指的是地址块已经分配给下一级的ISP，并不意味着任何人就无法获得新的IP地址



IPv6

1995年RFC1883发布的IPv6标准，1998年RFC 2460成为draft standard，最新的IPv6标准在2017年发布的RFC8200中定义

- **地址空间(32比特)有限**

- CIDR, NAT, DHCP等缓解地址不足
- 但考虑到:
 - 新用户：发展中国家
 - 新设备：PDA/Cell Phone/家用电器, 物联网技术的发展

- 路由表越来越庞大，路由效率低

- 头部过于复杂：检验和，可变长，IP选项，无法快速处理和转发

- 没有安全性考虑
- 无法提供QoS保障
- 没有移动性支持
- 采用DHCPv4自动配置或者手工配置TCP/IP参数

- ◆ **更大的地址空间（128比特）： 3.4×10^{38}**

- ◆ **更加有效的路由**

- ◆ **简化的头部格式（固定头部+扩展头部、无检验和、途中无分段、Hop Count）**

- ◆ **内嵌的安全支持：IPSec**

- ◆ **头部包括Traffic Class, Flow Label，支持QoS**

- ◆ **考虑到IP移动，包含路由和移动头部等**

- ◆ **除了DHCP外还支持server-less自动配置：plug-and-play**

IPv6分组格式

IPv4要求MTU最少为68字节,
接收者支持至少576字节
IPv6要求MTU最少为1280字节

- 4 bit Version: 取值为6
- 不再使用的字段:
 - 头部长度的IHL: 固定头部
 - 和分段有关字段: 只支持源端分段
 - 头部检验和
- 保留但替代的字段:
 - 8 bit Next Header 代替协议字段, 支持扩展头部(选项)
 - 8 bit Hop Limit代替TTL
 - 16 bit Payload Length(固定头部后面的长度) 代替Total Length
 - 8bit Traffic Class (DSCP+ECN) 代替ToS
- **新增20 bit Flow Label**, 可标识分组流, 用于快速转发 (MPLS)

去掉的字段

保留的字段

Version	IHL	ToS	Total length	
Identification			Flags	Fragment offset
Time to live		Protocol	Header checksum	
Source address				
Destination address				
Options (padding)				

IPv4有, 但是改变

20字节+最多40字节的选项

新增的字段

Version	DSCP	ECN	Flow Label(20)	
Payload length			Next header	Hop Limit
Source address(128)				
Destination address(128)				

40字节+可扩充的选项

IPv6扩展头部

Next Header：用于链接扩展头部和数据部分，其取值与IPv4的协议号一致

- 目前定义了8个扩展头部，只有**逐跳选项才需要在途中的路由器进行处理**，其他扩展头部只有在到达目的地址字段给出的节点之后才会进行处理
- 逐跳选项：如果有，**则为第一个选项**
- 建议顺序：逐跳选项、（中间目的地的）目的选项、路由、分段、认证（AH）、封装安全负载（ESP）、（最终目的地的）**目的选项(书中漏掉了)**、移动。最后是高层（TCP/UDP）协议的头部。如果没有高层协议信息时，最后为无下一个头部(No Next Header)
- 分段头部：IPv6仅允许源端分段，假设最小的MTU从68(一般576) 增加到1280字节

IPv6头部 (Next Header = TCP)	TCP头部	TCP用户数据		
IPv6头部 (Next Header = 路由)	路由头部 (Next Header = TCP)	TCP头部	TCP用户数据	
IPv6头部 (Next Header = 路由)	路由头部 (Next Header = 分段)	分段头部 (Next Header = TCP)	TCP头部	TCP用户数据

IPv6地址

- 128比特的IPv6地址以16个比特为一组，分为8组，**每组以十六进制书写，组与组之间用冒号分割**
- IPv6地址书写格式中的0可以压缩
 - 每组中前面的0可以移走，0123→123
 - 连续的为全0的组可以省略，但仅仅缩写一次
- IPv4地址可在书写格式的最低32比特出现
- ✓ IPv6**没有广播地址**
- ✓ IPv6组播地址: 前面8个比特为全1，即**FF00::/8**
- ✓ IPv6引入Anycast地址
 - anycast地址并没有一个特殊的前缀，看起来就是一个普通的unicast地址，依赖于Internet路由协议实现anycast
 - **子网-路由器(subnet-router)任播地址**: 预定义的anycast地址，用于与子网上的任一路由器通信，其与IPv4的子网地址格式类似，前面为子网前缀，最后为连续多个0
 - 子网-路由器(subnet-router)任播地址仅可用于IPv6分组的目的地地址，不可作为源地址

8000:0000:0000:0000:0123:4567:89AB:CDEF

8000:**0:0:0:123**:4567:89AB:CDEF

8000::**123:4567:89AB:CDEF**

IPv4映射地址: ::FFFF:**202.120.224.5**

IPv6地址：组播地址（拓展）

IPv6组播地址: 前面8个比特为全1, 即FF00::/8

8bits	4bits	4bits	112bits
0xFF	Flags (ORPT)	Scope	Group ID

- 临时地址: FF10::/12
 - R: 组ID是否包含RP的信息
 - P: 组ID是否包含了网络前缀信息
 - T: 组ID是否为临时地址/众所周知
- Well-Known地址: FF00::/12
 - Scope, 限定组播分组的传播范围
 - 0x1: 接口(interface-local)
 - 0x2: 链路(link-local)
 - 0xE: 全局
 - FF02::1 All Nodes, 相当于本地广播地址
 - FF02::2 All Routers
 - 请求节点(Solicited-Node)地址: FF02:0:0:0:0:1:FF00::/104, 链路唯一(scope=2)

请求节点组播地址

FF02:0:0:0:0:1:FF	请求的IPv6地址的最后24位
-------------------	-----------------

- 用于重复地址检测和地址解析等, 类似于ARP, 但采用以太网组播而不是广播
- IPv4 ARP: A广播ARP请求到链路层广播地址 FF-FF-FF-FF-FF-FF, 询问B的MAC地址
- IPv6: A发送Neighbor Solicitation消息到B的请求节点组播地址 (静态映射为链路层组播地址), 询问B的MAC地址

01-00-5E	IPv4组播地址的最后23位
----------	----------------

IPv6组播地址==>以太网组播地址

33-33	IPv6组播地址的最后32位
-------	----------------

- IPv6将地址解析(邻居发现)、差错报告和组播成员管理(IGMP)结合在一起, 通过ICMPv6消息实现
 - RFC 4443 ICMPv6; RFC 4861 邻居发现(NDP); RFC 2710 MLD(Multicast Listener Discovery); RFC 3810 MLDv2(类似于IGMPv3)



IPv6地址

- IPv6接口会配置一个链路唯一的IPv6地址，与链路上的其他IPv6主机通信
- IPv6接口还可配置一个全局单播地址，其分配考虑到路由汇集的支持。一般来说
 - 001 + 注册ID + 提供者ID + 用户ID共48比特（ $m + n + o = 45$ ）
 - 子网ID：16比特
 - 接口ID：64比特

3	m	n	o	p	125-m-n-o-p
001	注册ID	提供者ID	用户ID	子网ID	接口ID

用途	前缀	CIDR表示
未知地址， 仅仅作为源地址	128个0	::/128
回环地址， 仅在主机内部	127个0 + 1个1	::1/128
组播地址	1111 1111	FF00::/8
链路唯一(link-local)单播地址	1111 1110 10	FE80::/10
本地唯一（Unique-local）单播地址	1111 110	FC00::/7
IPv4映射地址， 不应出现在分组头部	80个0+16个1	::FFFF:0:0/96
Teredo地址	0010 0000 0000 0001+16个0	2001::/32
6to4地址	0010 0000 0000 0010	2002::/16
(当前分配的)全局单播地址	001	2000::/3

复旦大学IPv6地址段
2001:DA8:8001::/48 目前部署
2001:DA8:B7::/48
2001:250:6001::/48
2001:251:7805::/48

如何配置IPv6地址?

- **链路唯一的IPv6地址**(FE80::/10): 1111 1110 10+54个0 + 64bit的接口地址
- IPv6路由器定期(每200秒)发送ICMPv6 Router Advertisement消息给ff02::1(所有IPv6主机组播地址)
 - 包括了**IPv6前缀和前缀长度**、缺省路由器地址(Link-local地址)、支持的IPv6地址获得方式等
 - 可以采用**DHCPv6**获得IPv6地址以及其他配置信息
 - 可以采取SLAAC(**Stateless Address Autoconfiguration**)方式获得全局唯一的IPv6地址
 - 由RA消息中的**IPv6前缀/前缀长度再加上64位的接口地址**组成

- 如何获得64位的接口地址?

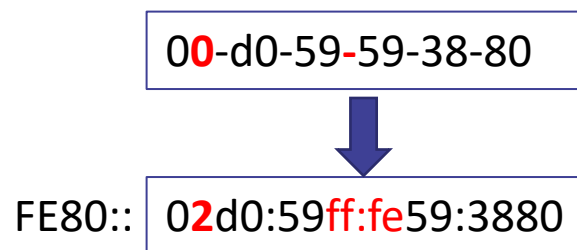
- 可以采用64位EUI-64地址 U/L比特取反

- 唯一不变

- 可以进行跟踪, 安全隐患

- RFC 8981 Temporary Address Extensions for Stateless Address Autoconfiguration in IPv6

- 引入了临时地址的概念, 该地址具有一定的生命期 (缺省2天, 1天内优先), 一个IPv6接口可以有多个临时地址



当前临时地址

之前分配的临时地址

主地址

链路唯一

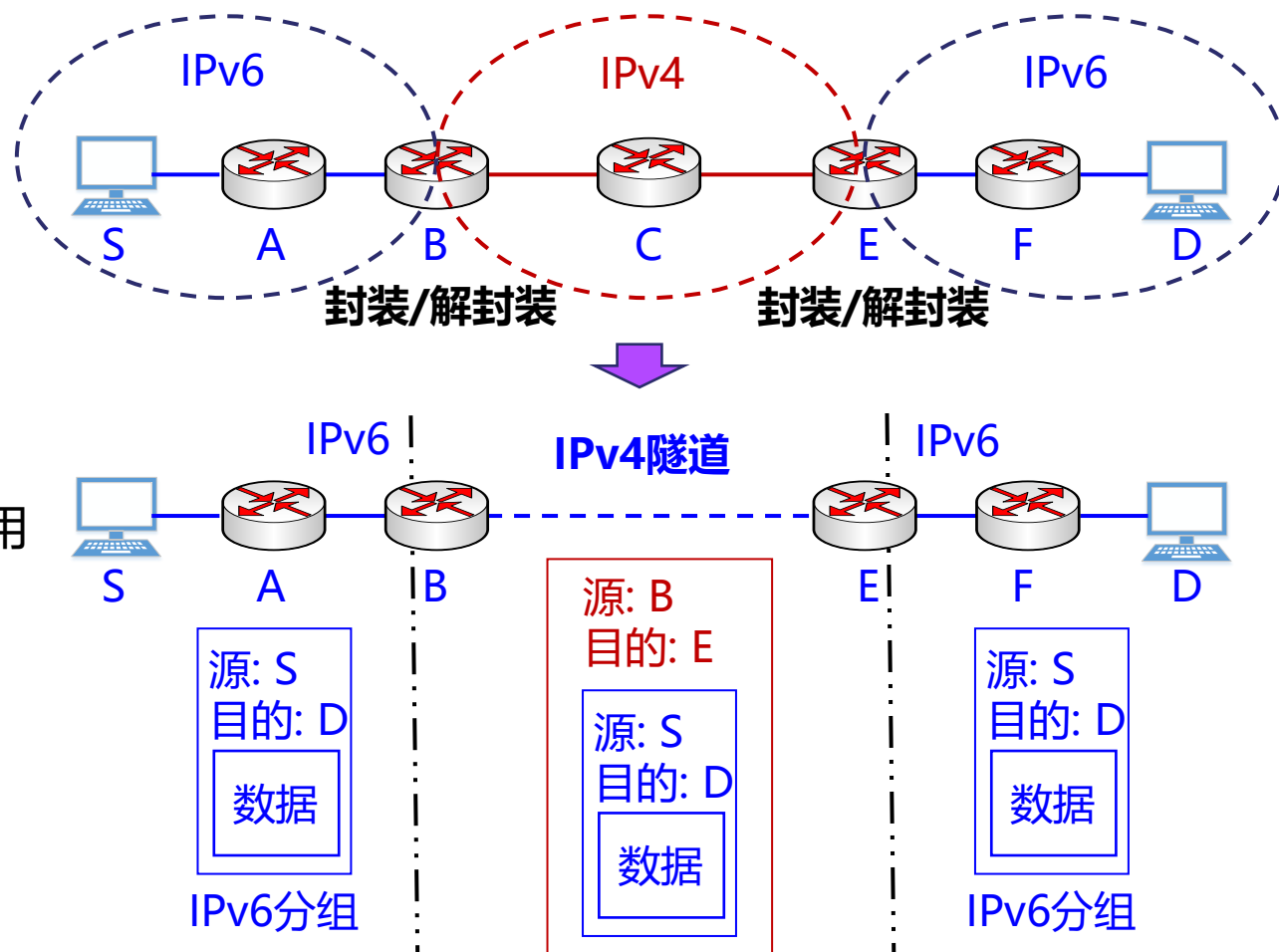
```
inet6 2001:da8:8001:7a54:f8ac:1ff1:5b1d:b5b4/64 scope global temporary dynamic
      valid_lft 524788sec preferred_lft 6194sec
inet6 2001:da8:8001:7a54:d021:30df:54c4:a227/64 scope global temporary deprecated dynamic
      valid_lft 418555sec preferred_lft 0sec
inet6 2001:da8:8001:7a54:ebe3:1fa1:5931:65e1/64 scope global dynamic mngtmpaddr noprefixroute
      valid_lft 2591984sec preferred_lft 604784sec
inet6 fe80::1945:7e63:b388:e072/64 scope link noprefixroute valid_lft forever preferred_lft forever
```


IPv4到IPv6的过渡

- IPv4协议和IPv6协议并不兼容，迁移可能经历很长时间
 - 涉及：互联网用户、互联网服务提供商（ISP）、互联网内容提供商（ICP）、网络设备厂家
- 较长的一段时间之内，IPv4和IPv6会同时并存，如何过渡？
 - 双协议栈：同时运行IPv4和IPv6协议栈
 - 协议转换：类似于NAT，负责分组格式转换
 - 隧道

如何找到隧道的另一端？

- 人工配置:RFC 4213定义6in4，即IPv6-in-IPv4封装 (protocol=41)
- 隧道代理（Tunnel Broker）：RFC3053定义，帮助用户配置包括6in4在内的IP隧道
- 自动隧道：包括6to4、**Teredo(linux实现为miredo)**、ISATAP、6rd等



6to 隧道

- RFC 3056: 引入6to4地址2002::/16。隧道一端为6to4主机，另外一端为另一个6to4主机或6to4 relay
- 6to4主机: 双协议栈, 且有一个**公网IPv4地址**

2002	IPv4地址	0	接口地址
------	--------	---	------

 - 公网IPv4地址映射为6to4地址2002::/16, 接下来32比特为(public)IP地址
 - **128.178.156.38** → 2002:**80b2:9c26**:0:EUI (2002:ipv4/48)
- 6to4主机A(或路由器A) 和另外一个6to4主机B (或路由器A) (从其16比特前缀为2002了解到) 通信时, 从中获取对方的IPv4地址, 然后建立IPv4隧道A→B
- 6to4主机A与IPv6主机C的通信则要通过6to4 Relay Router, 采用IPv4地址anycast地址192.88.99.1(2002:c058:6301::0), 首先建立隧道A→any 6to4 relay router, 然后通过IPv6到达目的地

