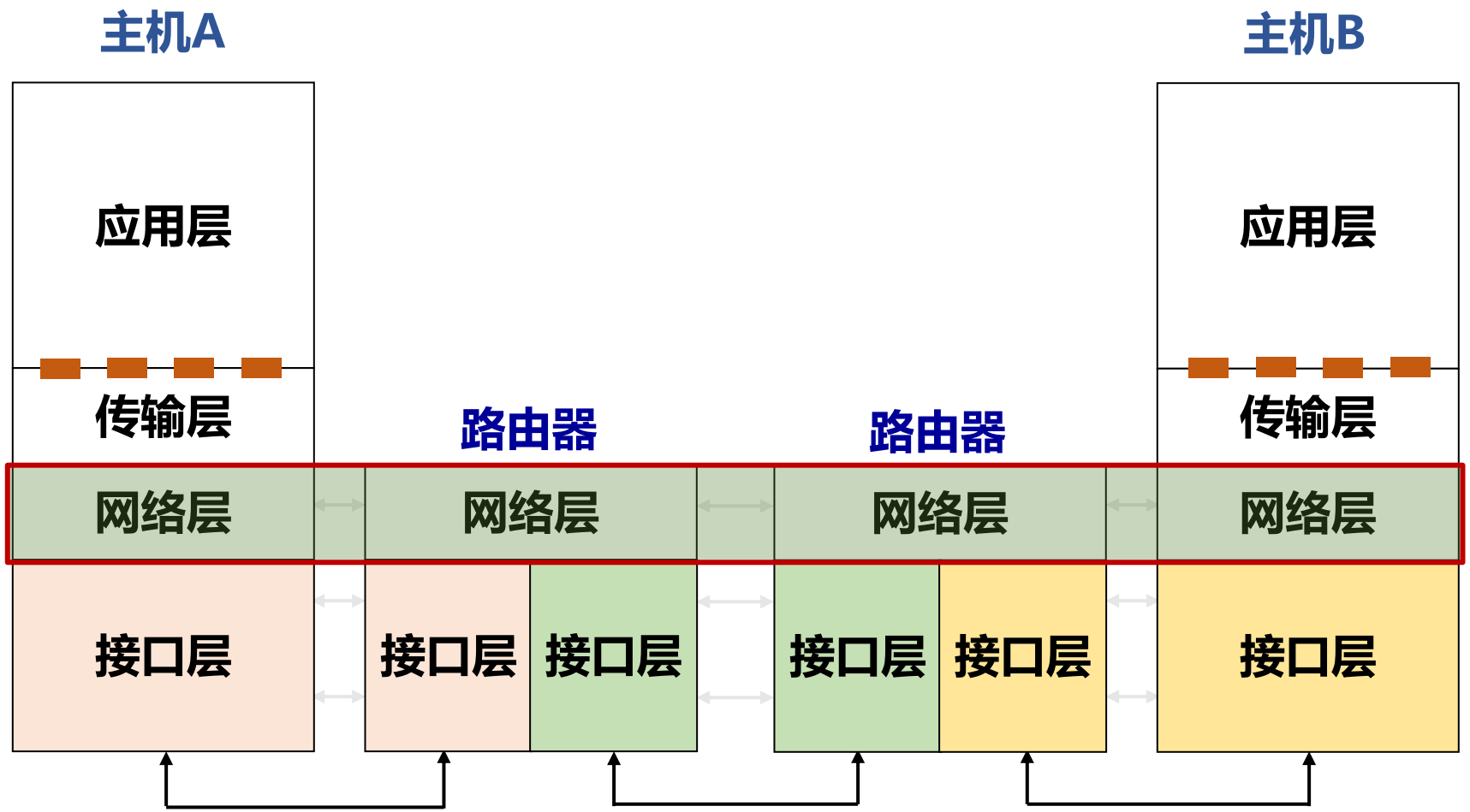
第五章 路由选择和网络层

“101计划”
计算机网络

网络层在哪里？



接口层通常包括数据链路层和物理层

本章内容

5.1 网络层概述

1. 网络层服务概述
2. 无连接服务的实现
3. 面向连接服务的实现
4. 虚电路与数据报网络的比较

5.2 网络层协议

5.3 路由算法

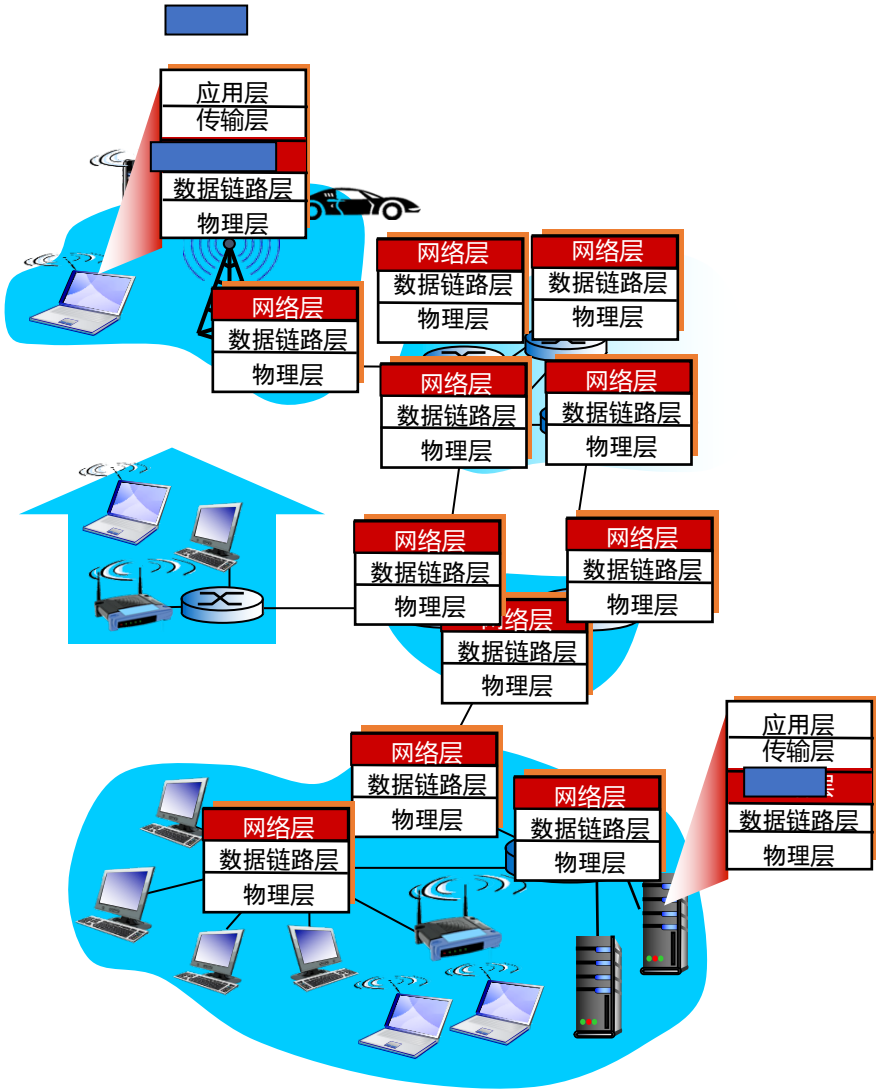
5.4 流量管理与服务质量

5.5 路由器体系结构与关键技术

5.6 软件定义网络

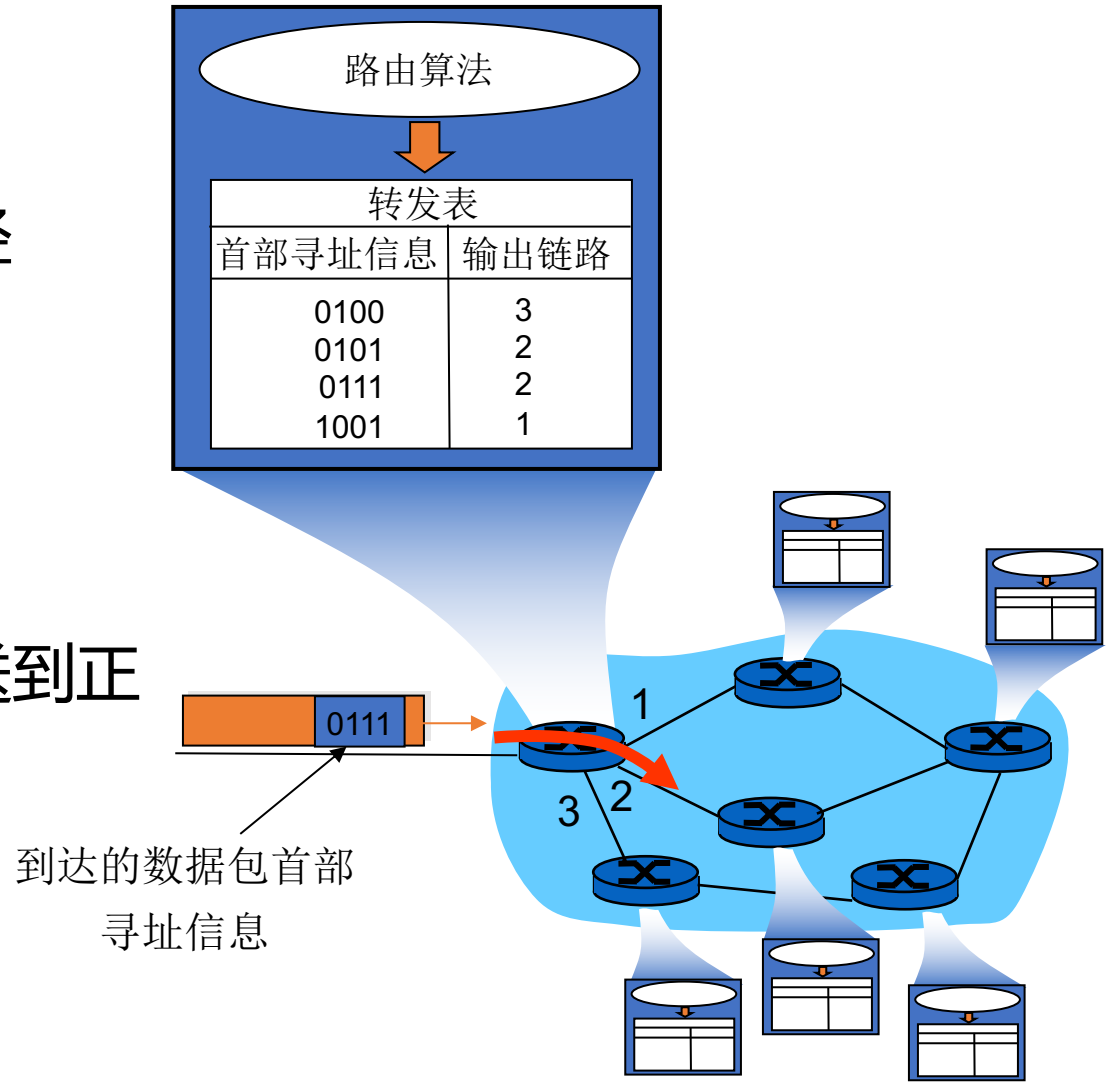
网络层服务的实现

- 网络层实现端系统间多跳传输可达
- 网络层功能存在每台主机和路由器中
  - 发送端：将传输层数据单元封装在数据包中
  - 接收端：解析接收的数据包中，取出传输层数据单元，交付给传输层
  - 路由器：检查数据包首部，转发数据包



网络层关键功能

- 路由（控制面）
  - 选择数据报从源端到目的端的路径
  - 核心：路由算法与协议
- 转发（数据面）
  - 将数据报从路由器的输入接口传送到正确的输出接口



提供给传输层的服务

- 网络通信的可靠交付服务，谁来负责？

  “网络” OR **“端系统”**

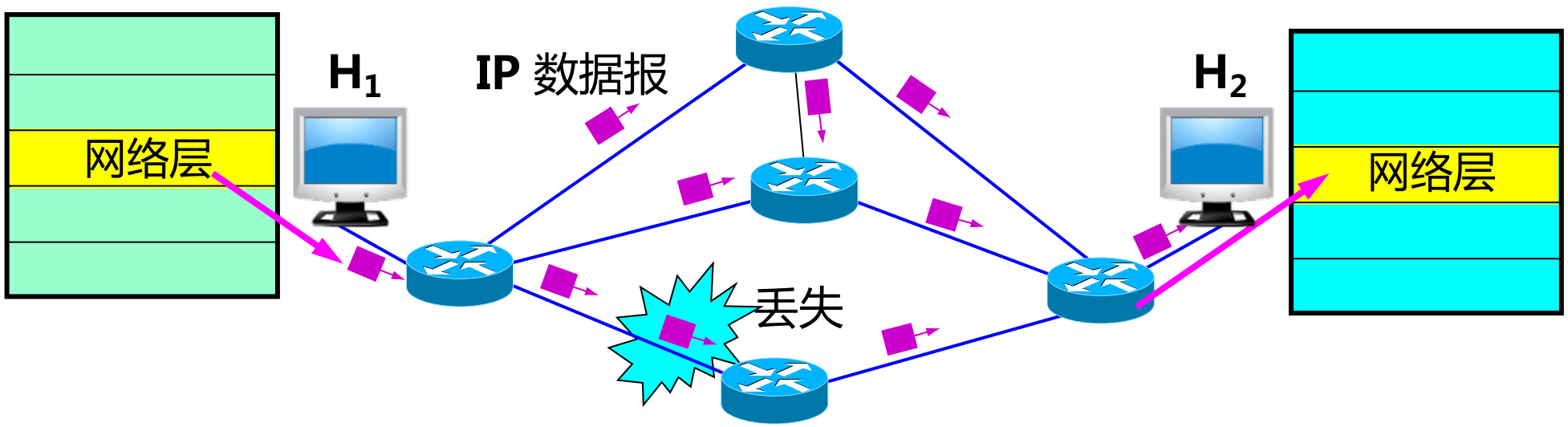
- 网络层应该向运输层提供怎样的服务？

  “面向连接”（虚电路）OR **“无连接“**（数据报）

无连接服务的实现

- 无连接服务：如寄信
  - 不需要提前建立连接
- 数据报服务
  - 网络层向上只提供简单灵活无连接的、尽最大努力交付的数据报服务
  - 发送分组时不需要先建立连接，每个分组独立发送
  - 数据报独立转发，相同源-目的的数据报可能经过不同的路径
  - 网络层不提供服务质量的承诺
- 尽力而为交付
  - 传输网络不提供端到端的可靠传输服务：丢包、乱序、错误
  - 优点：网络的造价大大降低，运行方式灵活，能够适应多种应用

无连接服务的实现

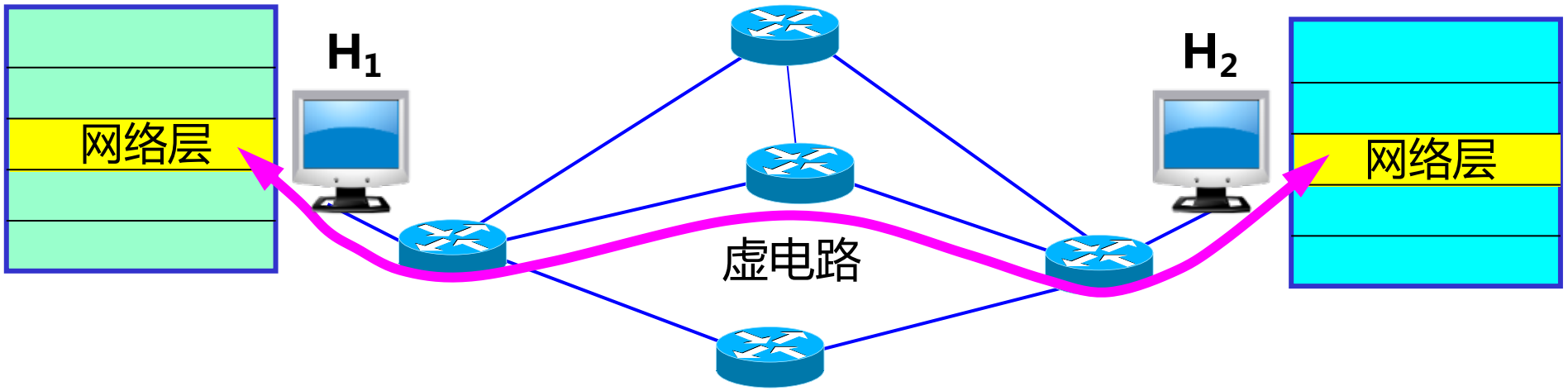


H1 发送给 H2 的分组可能沿着不同路径传送
在数据包分片的情况下，尽量还是沿相同路径

面向连接服务的实现

- 面向连接服务：如打电话
  - 通信之间先建立逻辑连接：在此过程中，如有需要，可以预留网络资源
  - 结合使用可靠传输的网络协议，保证所发送的分组无差错按序到达终点
- 虚电路是逻辑连接
  - 虚电路表示这只是一条逻辑上的连接，分组都沿着这条逻辑连接按照存储转发方式传送，而并不是真正建立了一条物理连接
  - 注意，电路交换的电话通信是先建立了一条真正的连接
  - 因此分组交换的虚连接和电路交换的连接只是类似，但并不完全相同

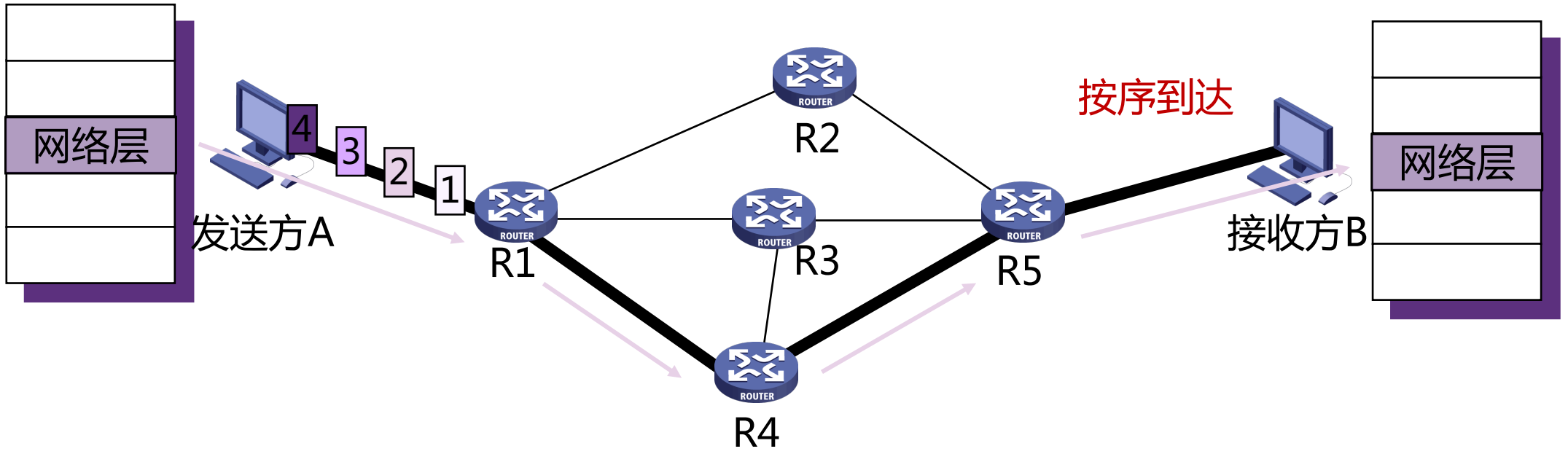
面向连接服务的实现



H_1 发送给 H_2 的所有分组都沿着同一条虚电路传送

面向连接的虚电路

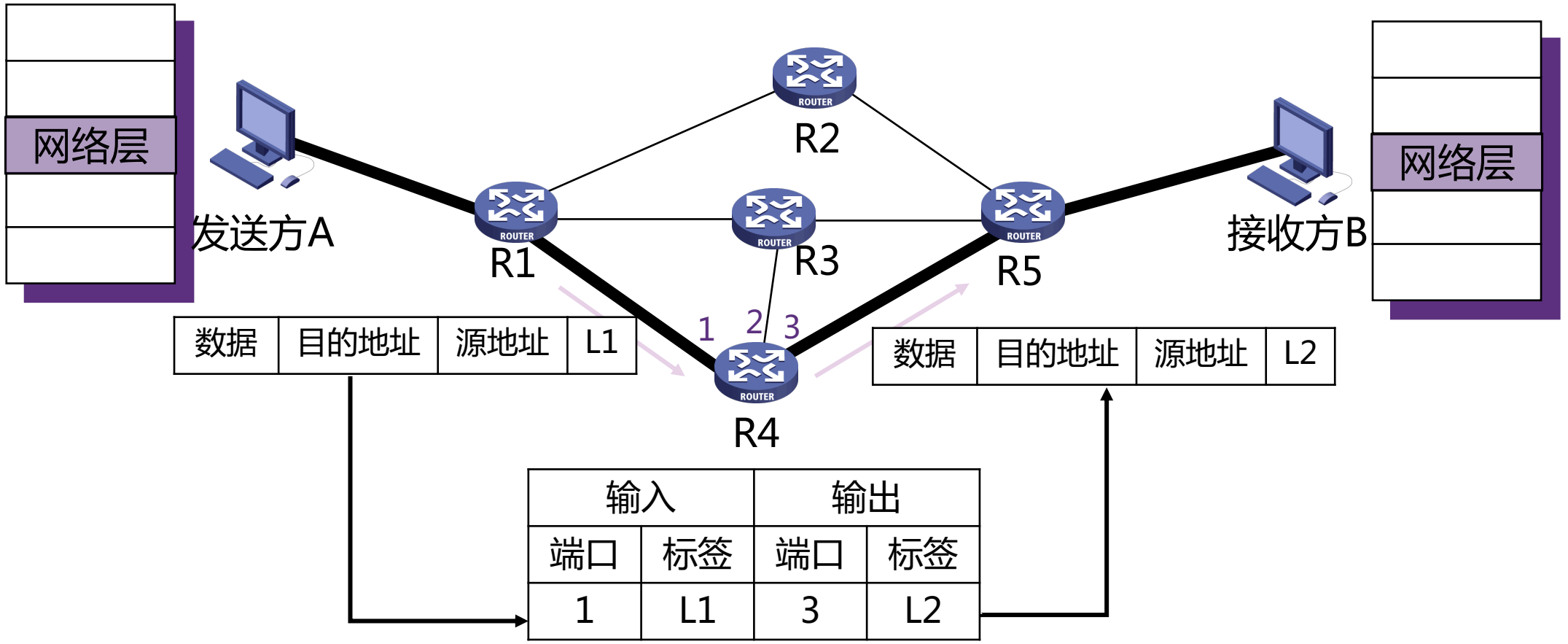
- 虚电路（virtual circuit）：面向连接的方法



面向连接的方法也不一定能完全保证数据的可靠传输，链路中的任何一个组成环节仍有可能失效，而这种失效是严重的，可能导致所有数据丢失

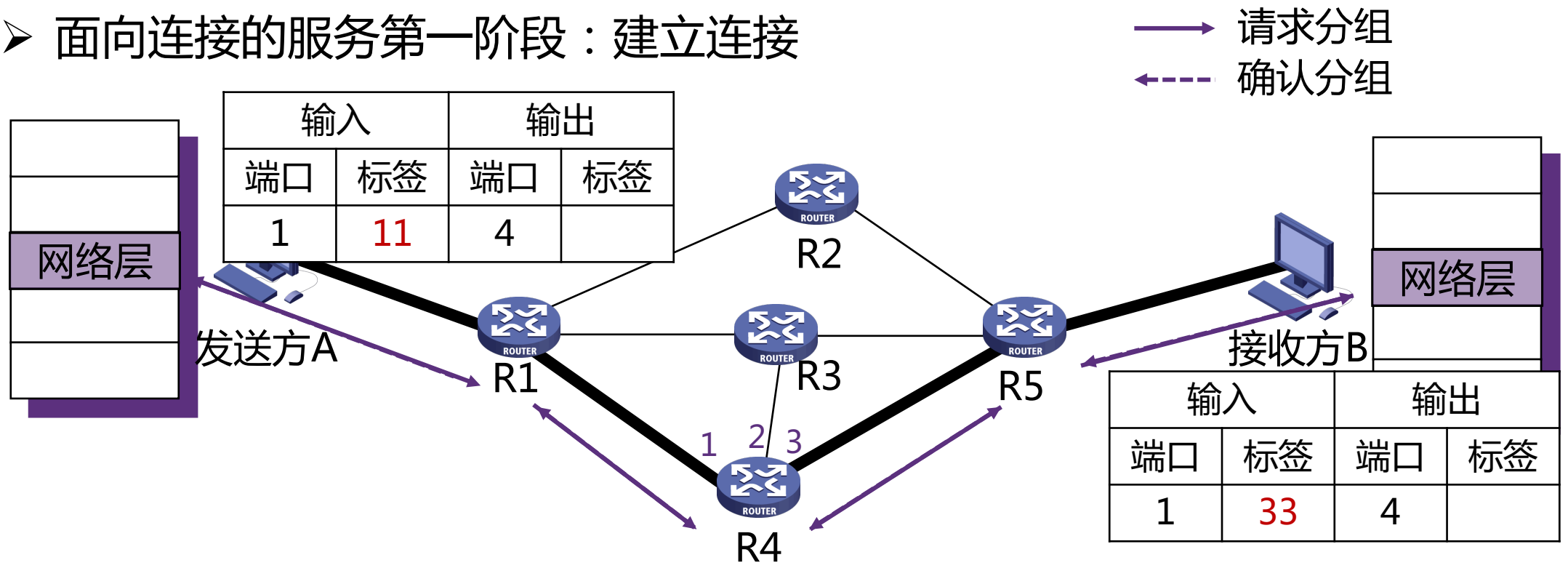
虚电路转发策略

- 虚电路的转发策略：虚电路转发决策基于分组标签，即虚电路号



面向连接的虚电路

- 面向连接的服务第一阶段：建立连接



输入		输出	
端口	标签	端口	标签
1	11	4	

输入		输出	
端口	标签	端口	标签
1	33	4	

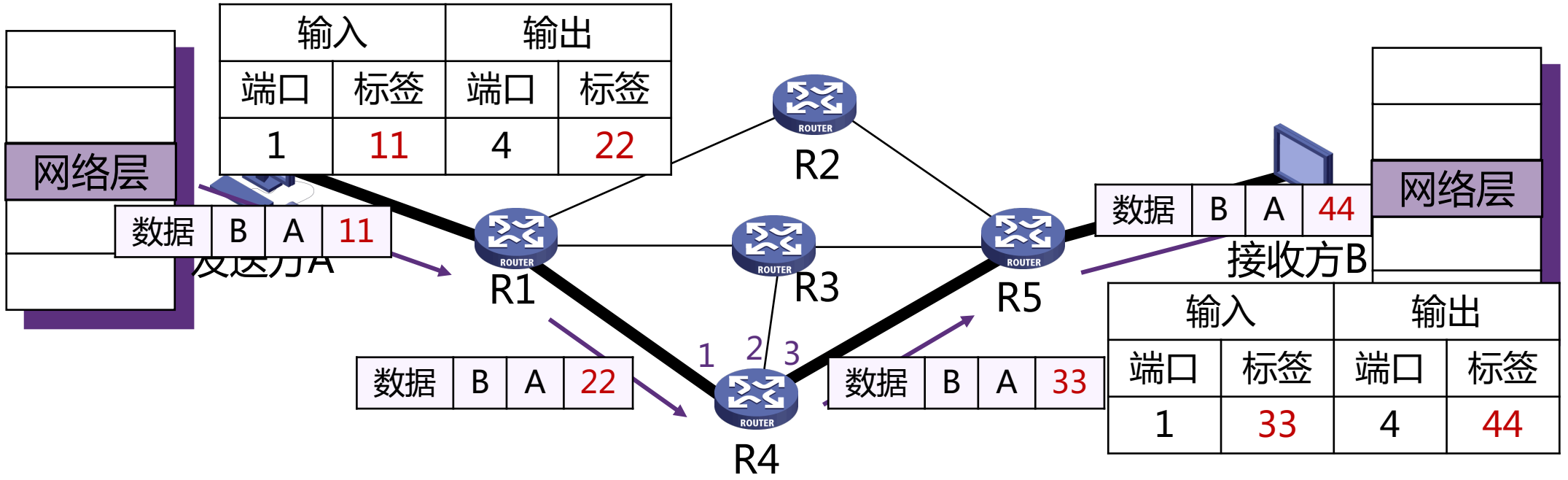
输入		输出	
端口	标签	端口	标签
1	22	3	

面向连接的虚电路

- 面向连接的服务第二阶段：发送数据

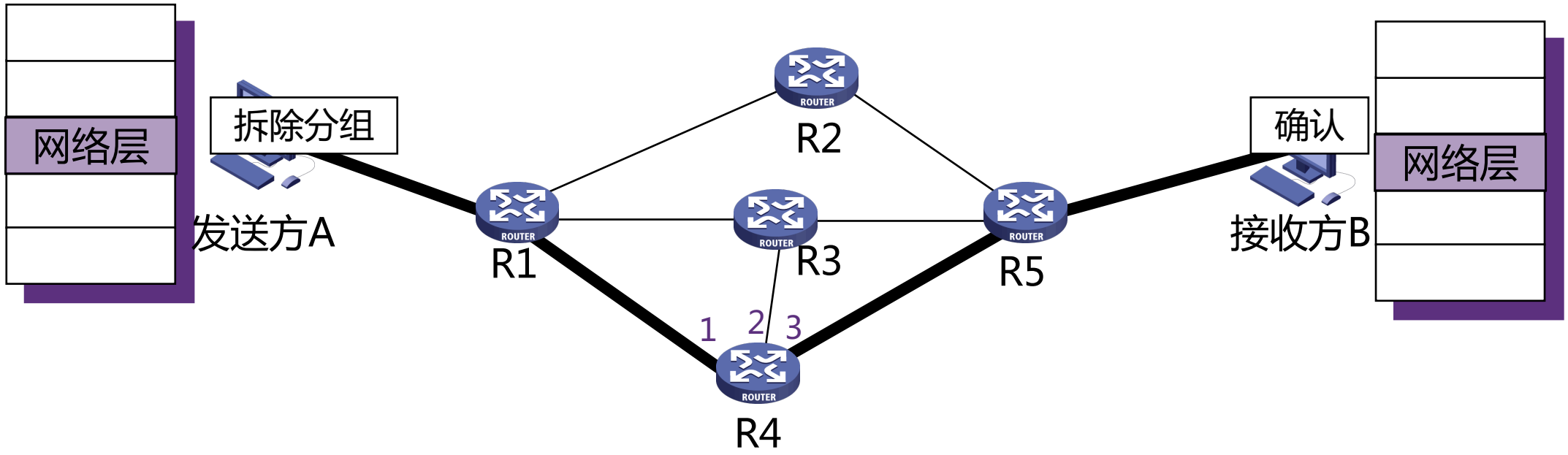
数据	目的地址	源地址	标签

分组



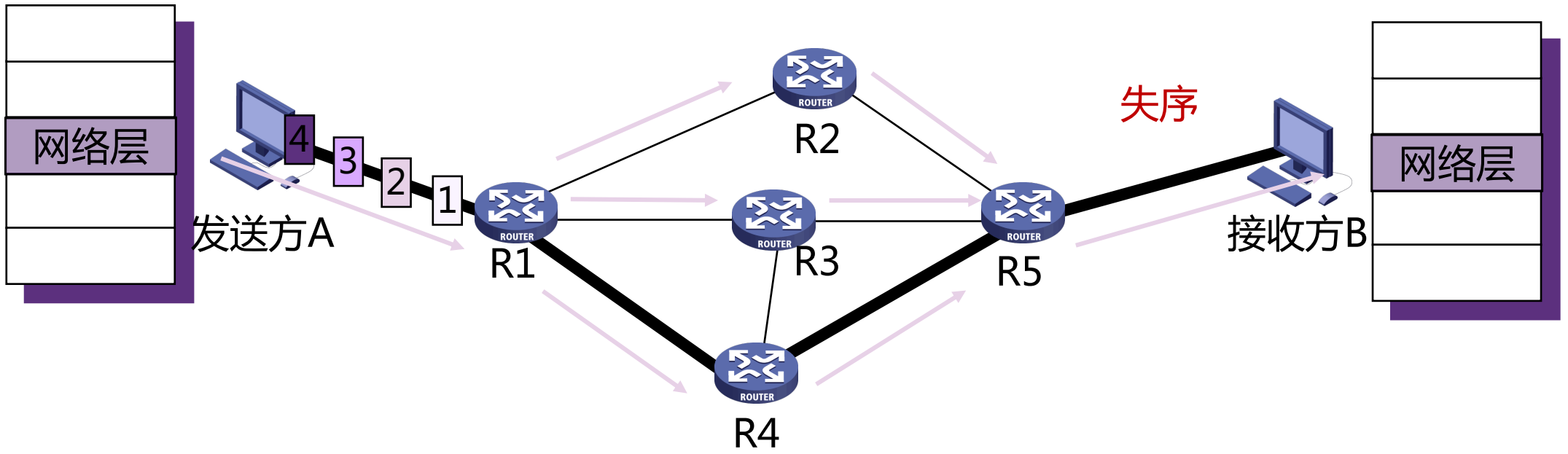
面向连接的虚电路

- 面向连接的服务第三阶段：释放连接



无连接的数据报

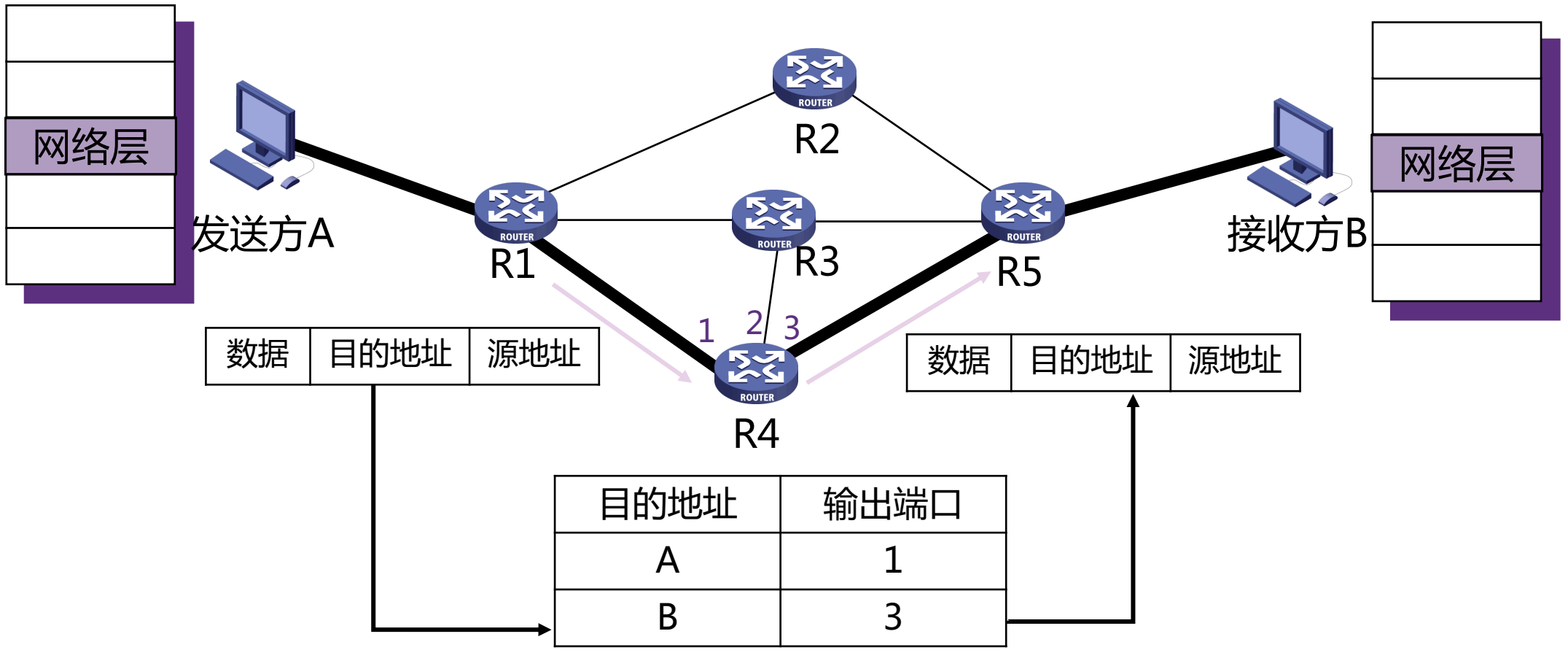
➢数据报（datagram）：无连接的方法



无连接的方法允许分组有选择不同路径的可能性，但这样可能会导致接收数据的失序；需要说明的是，为避免增加额外的开销进行数据排序，网络并不会完全随意地发送数据，在大多数情况下，仍然是会尽量沿着某一条路径发送。

数据报的转发策略

- 数据报转发策略：数据报转发决策基于分组的目的地址



虚电路与数据报网络的比较

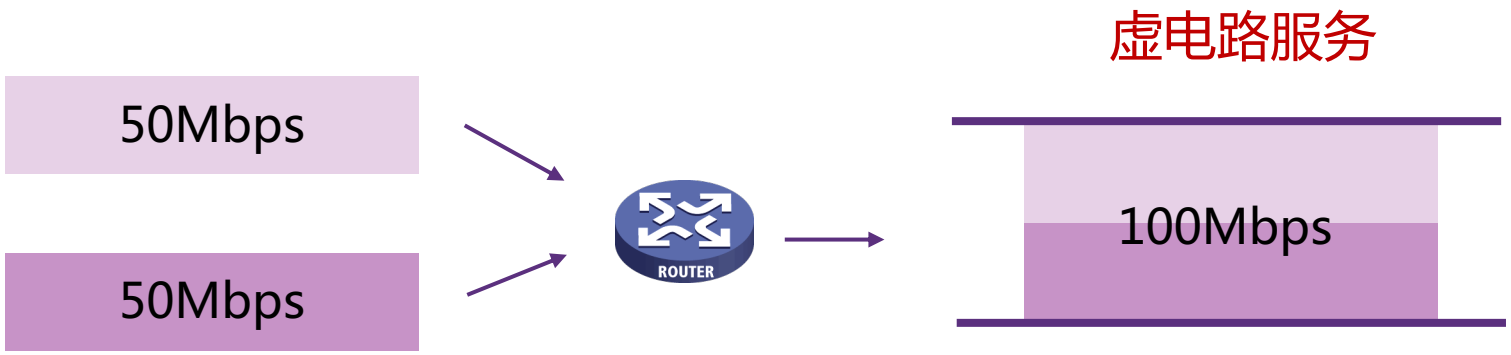
对比内容	虚电路服务	数据报服务
可靠传输的保证	可靠通信由网络保证	可靠通信由主机保证
连接的建立	必须要	不需要
地址	每个分组含有一个短的虚电路号	每个分组需要有源地址和目的地址
状态信息	建立好的虚电路要占用子网表空间	子网不存储状态信息
路由选择	分组必须经过建立好的路由发送	每个分组独立选择路由
分组顺序	总是按序到达	可能乱序
路由器失效	所有经过失效路由器的虚电路都要终止	失效结点可能丢失分组
差错处理和流量控制	网络或用户主机负责	用户主机负责
拥塞控制	容易控制	难控制

虚电路与数据报网络的比较

- 虚电路 *vs.* 数据报：哪种方法更好？
- 取决于从什么角度进行比较
  - 性能
  - 效率
  - 对失败的控制
  - 实现的复杂性
  - ……

虚电路与数据报网络的性能比较

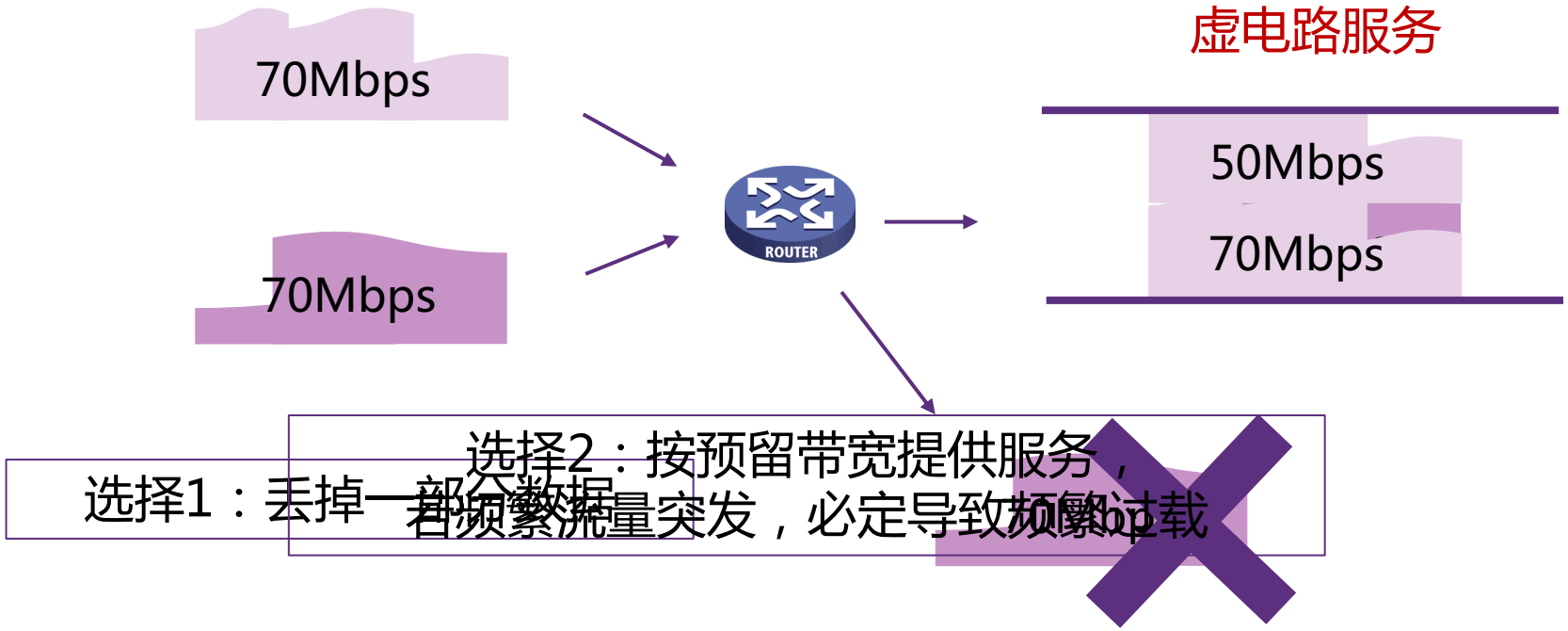
- 例1：从性能角度比较
  - 假设总带宽100Mbps，有2个数据源共享带宽
  - 如果每个数据源按50Mbps的**恒定速率**发送数据，使用虚电路服务，结果如何？



带宽不浪费
每个数据源发送数据的带宽都可被保证

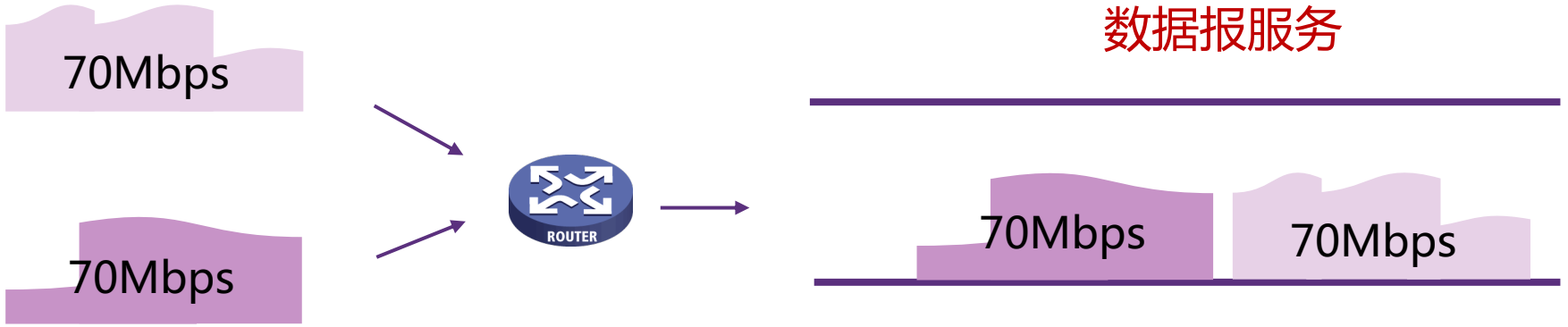
虚电路与数据报网络的性能比较

- 例1：从性能角度比较
  - 假设总带宽100Mbps，有2个数据源共享带宽
  - 如果每个数据源都是**突发流量**，且最高可达70Mbps，使用虚电路服务，结果如何？



虚电路与数据报网络的性能比较

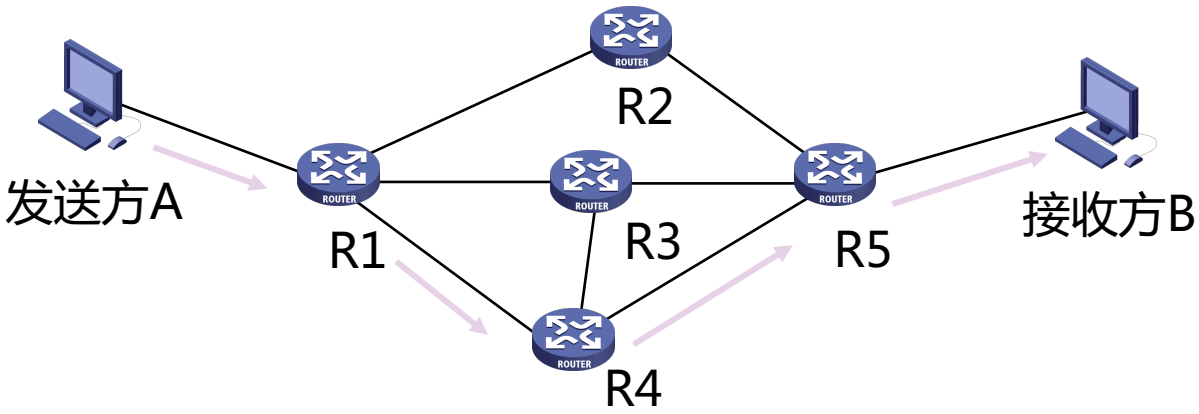
- 例1：从性能角度比较
  - 假设总带宽100Mbps，有2个数据源共享带宽
  - 如果每个数据源都是**突发流量**，且最高可达70Mbps，使用数据报服务，结果如何？



性能完全不受影响
也不会过载

虚电路与数据报网络的效率比较

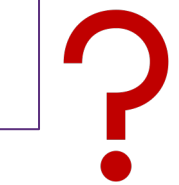
- 例2：从效率角度比较
  - 假设不考虑过载，发送同样多的数据，消耗的时间比较



- 假设不考虑A的发送时延和链路传播时延，在上图3个转接节点的情况下，链路上的数据传输速率*B* bps，每个分组的长度*P* bit，每个分组的开销*H* bit，虚电路分组交换的呼叫建立时间*S* s，每个转接点的转接延迟时间*D* s，则：
  - 虚电路分组交换总时延T=S+3[D+（P+H）/B]
  - 数据报分组交换总时延 T=3[D+（P+H）/B]

虚电路与数据报网络的发展历史

思考：
到底哪种方法更好？



- 70-80年代：分组交换
  - X.25，帧中继
- 80年代末-90年代：研究人员和工业应用认为电路交换更好
  - 认为语音/电视直播将成为互联网真正的杀手级应用
- 分组交换已经成为互联网的实际服务方式，电路交换最终没有广泛应用于互联网...Why？
  - 人们重新编写应用程序以适应网络（应用程序并不需要保证带宽)
  - Email和Web广泛应用（突发流量）
- 虚电路仍有使用（MPLS，租用专线等）
  - 企业分支结构之间，昂贵，通常是静态设置（而非最初所希望的动态预留资源）

本章内容

5.1 网络层概述

5.2 网络层协议

1. IPv4协议&数据报分片
2. 编址问题
3. 特殊的IP地址

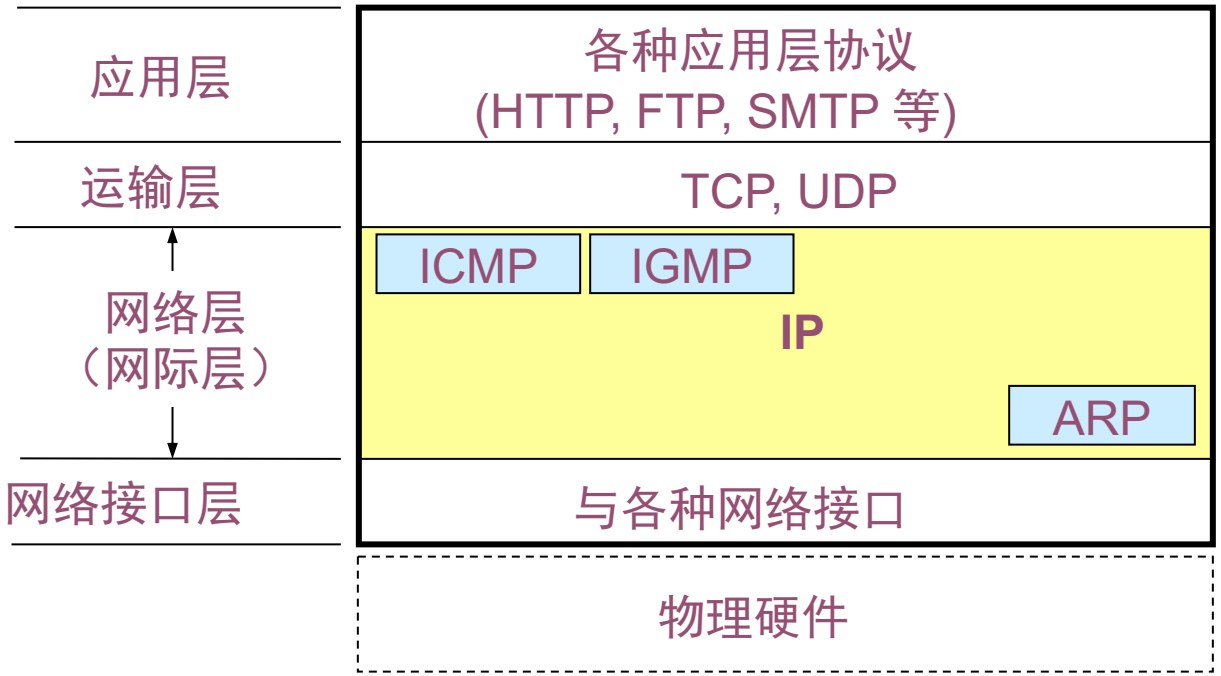
5.3 路由算法

5.4 流量管理与服务质量

5.5 路由器体系结构与关键技术

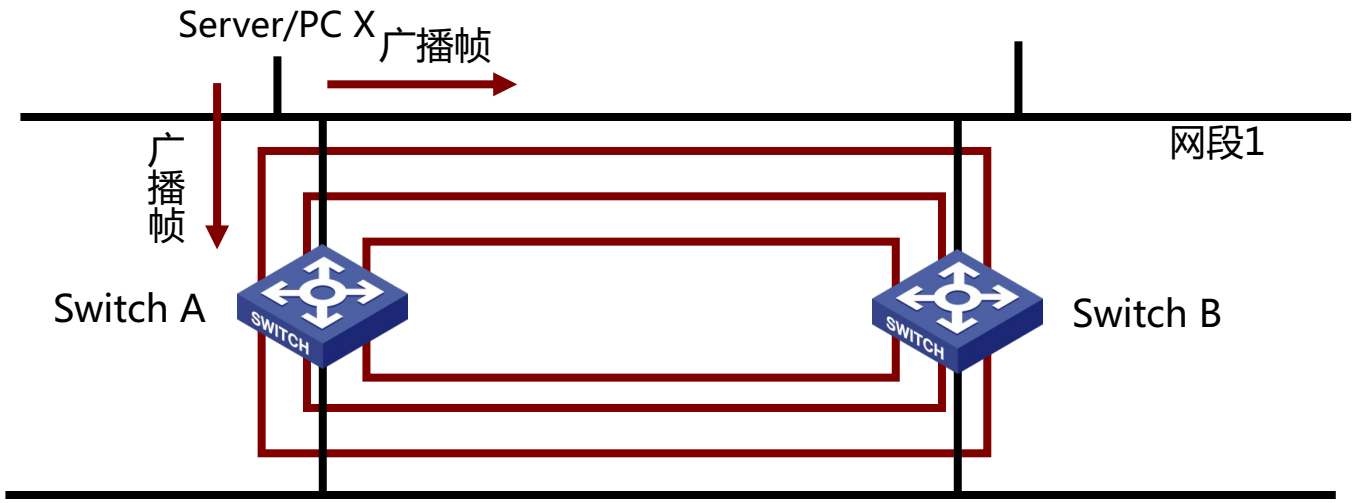
5.6 软件定义网络

网络层的 IP 协议及配套协议



IP协议功能需求

- 路由回路问题
  - 路由环路使数据包不断循环，耗尽网络资源
  - 链路层的生成树协议很好，但是……
  - 网络层的路由器负责消除环路？路由协议？
- 数据转发过程的终极解决方案？计数器TTL



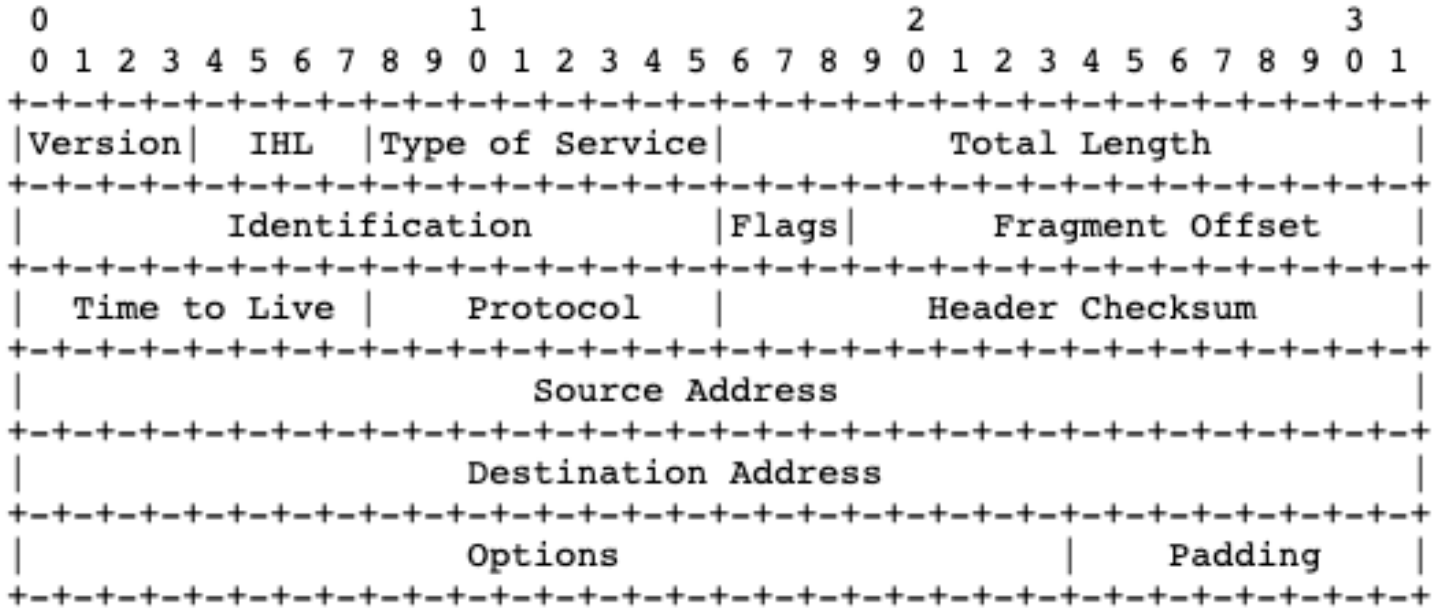
IP协议功能需求

- 网络层基本功能
  - 支持多跳寻路将IP数据报送达目的端：*目的IP地址*
  - 表明发送端身份：*源IP地址*
  - 根据IP头部协议类型，提交给不同上层协议处理：*协议*
- 其它相关问题
  - 数据报长度大于传输链路的MTU的问题，通过分片机制解决：*标识、标志、片偏移*
  - IP报头错误导致无效传输，通过头部校验解决：*首部校验和*
  - 防止循环转发浪费网络资源（路由错误、设备故障...），通过跳数限制解决：*生存时间TTL*

IPv4协议

- IPv4协议
  - Internet Protocol，网际协议版本4，一种无连接的协议，是互联网的核心，也是使用最广泛的网际协议版本，其后继版本为IPv6

- IP协议两个基本功能
  - 编址(addressing)
  - 分片(fragmentation)



RFC 791

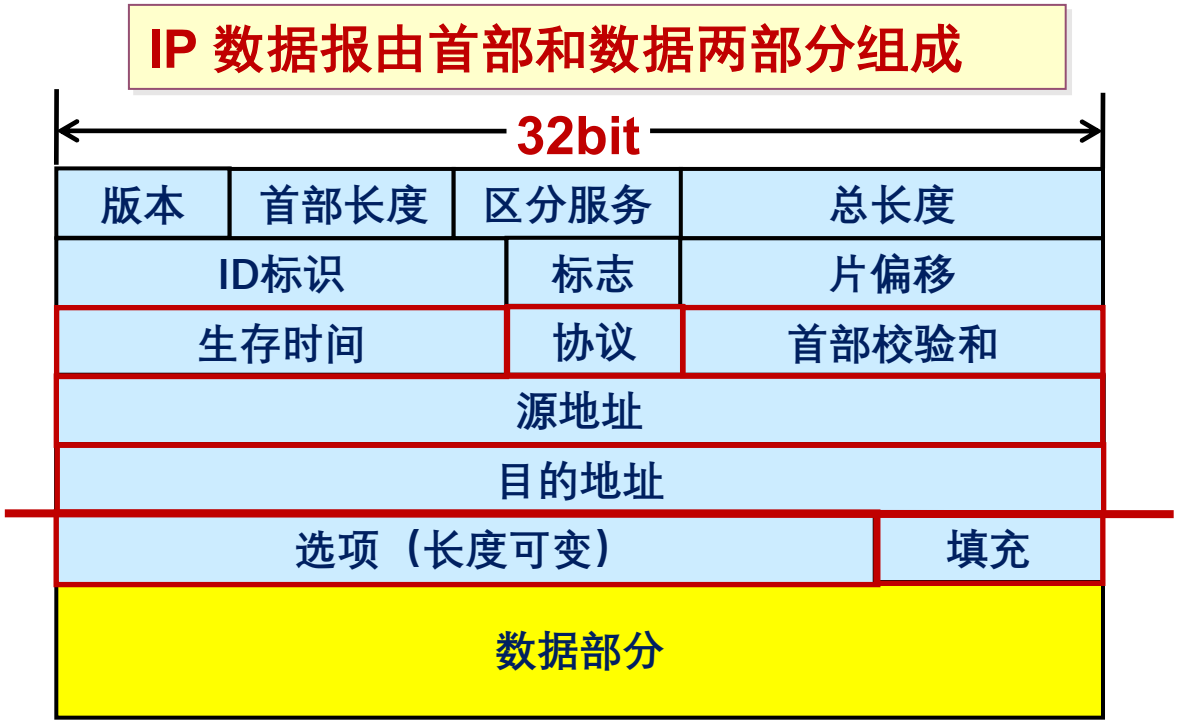
IPv4数据报格式

- 版本： 4bit ，表示采用的IP协议版本
- 首部长度：4bit，表示整个IP数据报首部的长度，以**4B**为单位，最大表示范围(2^4-1)×4=60字节
- 服务类型ToS： 8bit，该字段一般情况下不使用
- 总长度： 16bit，表示整个IP报文的长度,能表示的最大字节为2^16-1=65535字节，单位为**1B**
- 标识ID： 16bit，IP软件通过计数器自动产生，每产生1个数据报计数器加1，在ip分片后用来标识同一片的分片
- 标志： 3bit，目前只有两位有意义； MF，置1表示后面还有分片，置0表示这是数据报片的最后1个；DF，不能分片标志，置0时表示允许分片
- 片偏移：13bit，表示IP分片后，相应的IP片在总的IP片的相对位置(8 octets)，以**8B**为单位

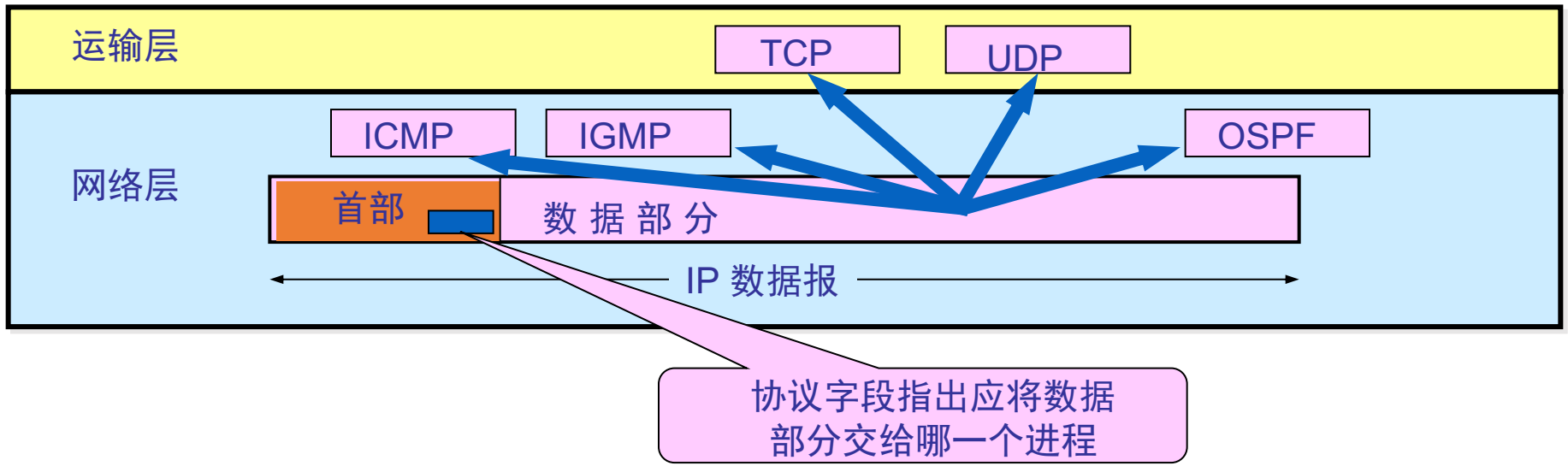
IP 数据报由首部和数据两部分组成

IPv4数据报格式

- 生存时间TTL(Time To Live) ：8bit,表示数据报在网络中的生命周期，用通过路由器的数量来计量，即跳数（每经过一个路由器会减1）
- 协议：8bit，标识上层协议（TCP/UDP/ICMP...）
- 首部校验和：16bit，对数据报首部进行校验，不包括数据部分
- 源地址：32bit，标识IP片的发送源IP地址
- 目的地址：32bit，标识IP片的目的地IP地址
- 选项：可扩充部分，具有可变长度，定义了安全性、严格源路由、松散源路由、记录路由、时间戳等选项
- 填充Padding：用全0的填充字段补齐为4字节的整数倍



附：协议字段



协议	ICMP	IGMP	TCP	UDP	OSPF	…
协议字段值	1	2	6	17	89	…

单选题 1分

设置

以下关于IP分组结构的描述中，错误的是（ ）

A IPv4 分组头的长度是可变的

B 协议字段表示 IP 的版本，值为 4 表示 IPv4

C 分组首部长度字段以 4B 为单位，总长度字段以字节为单位

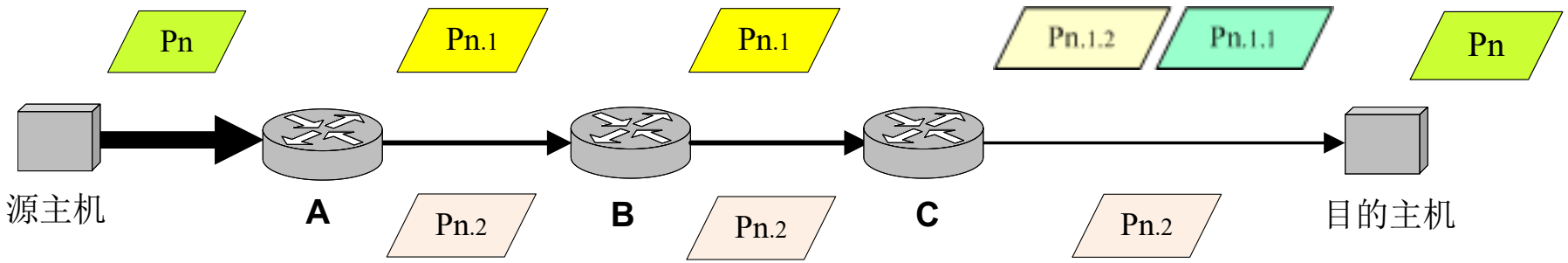
D 生存时间（TTL）字段值表示分组可以转发的跳数

提交

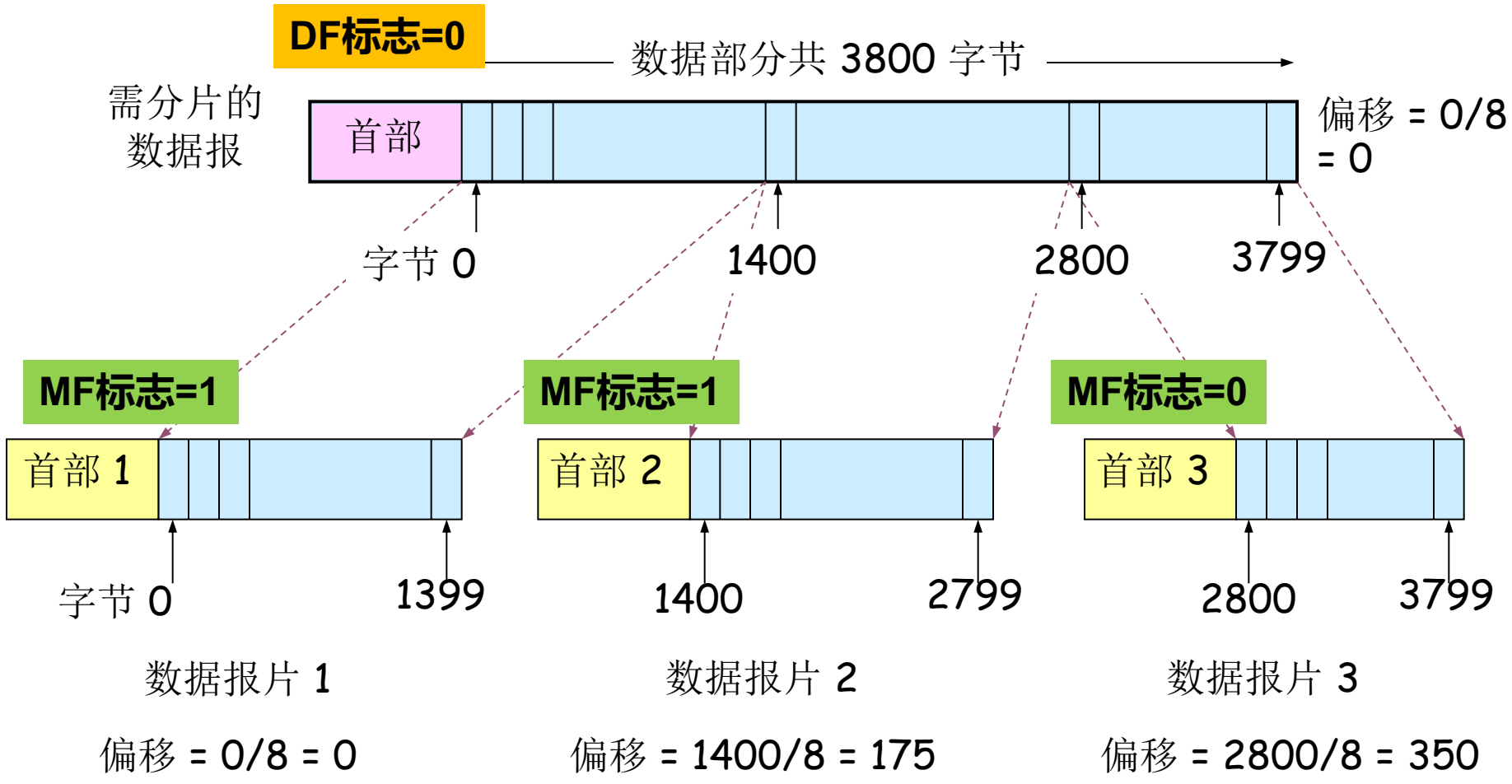
数据报分片

网络名	MTU（字节）	备注
以太网 II	1500	如果是 Jumbo 帧，此值可以达到 9000 字节，甚至更高。
PPPoEv2	1492	用 PPP 携带 Ethernet II 数据时，扣除 PPP 的 8 字节头部。
802.11	2304	2304 是 MAC 服务单元数据（MAC），如果加上 WEP（早期认证方法）的 8 字节，应为 2312 字节，如果加上 WPA2 的 16 字节，应为 2320 字节。
FDDI	4352	曾经非常引人注目的一种网络，可靠的快速（100M）传输
令牌环	4464	采用确定性介质访问控制，无冲突，现在已经很少见
X.25	576	一种古老的网络，提供面向连接（虚电路）的传输

- 回顾：以太网最大传输单元MTU
  - 最小帧长 = 46+18 = 64B
  - 最大帧长 = 1500+18 = 1518B （最大传输单元MTU：1500B）
- 每一跳承载IP数据包的能力都可能不同
- IP数据包需要能支持被切分（IP数据包分片机制）



数据报分片



原始报文和分片报文具有相同的IP标识（IP头部字段）

数据报分片

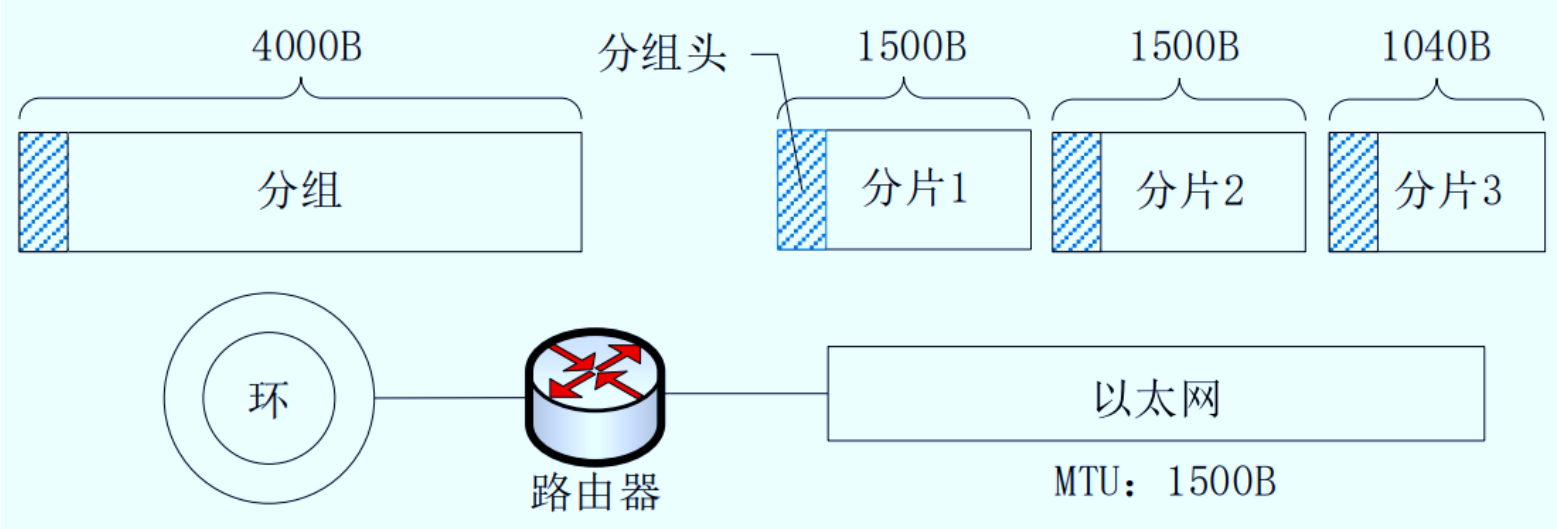
- 路由器收到分组后，如果分片，需要满足下面两个条件：
  - 分组的长度L大于转出网络的 MTU,即：L>M。
  - 分组中的标记位 DF没有被置位，即：DF=0
- 分片方法
  - 假设需要分片的总片数为 n，分组头部长度是 20 字节，分片载荷的长度上限是 d，则 $d = \lfloor \frac{M-20}{8} \rfloor \times 8$ $n = \lceil \frac{L-20}{d} \rceil$
  - 除了最后一个分片，其余的分片必须满载

$$F_i = \frac{\mathrm{d}}{8} \times (\mathrm{i}-1)\text{，其中 } 1 \le i \le n$$

$$L_i = \begin{cases} d+20, & 1 \le \mathrm{i} < \mathrm{n} \\ L - d \times (n-1), & i = n \end{cases}$$

例题

- 一个分组长 4000 字节，其中整个头部长度为 20 字节，头部标识字段值为 8580，标记位 DF=0，当分组到达路由器时，发现转出网络是以太网，其 MTU 是 1500 字节。路由器是否需要进行分片？如果进行分片，应分为多少片？每个分片的总长度、头部标识、标记位 DF 和 MF 以及片偏移的值分别是多少



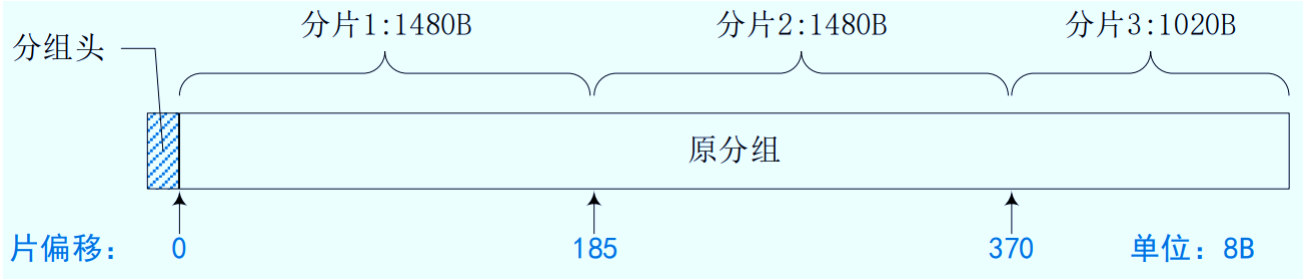
例题

- 因为4000B>1500B，DF=0，满足条件
- 分片的最大载荷 d=|(1500-20)/8|×8=1480B，则分片总片数为

$$n = \left\lceil \frac{L-20}{d} \right\rceil = \left\lceil \frac{4000-20}{1480} \right\rceil = 3$$

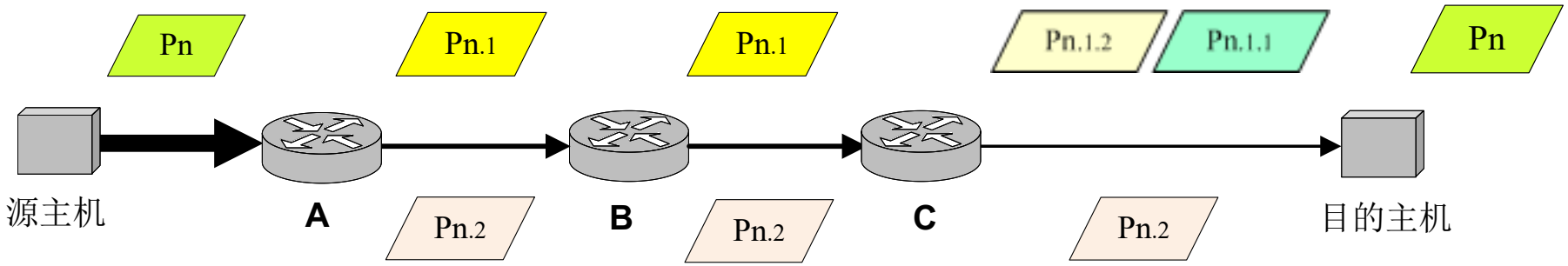
- 前两个分片满载，第三个分片的总长度应该是 4000B-1480B×2=1040B，其中的净载荷为 1020B

字段名 / 分组	总长度(字节)	标识	DF	MF	片偏移
原分组	4000	8580	0	0	0
分片 1	1500	8580	0	1	0
分片 2	1500	8580	0	1	185
分片 3	1040	8580	0	0	370



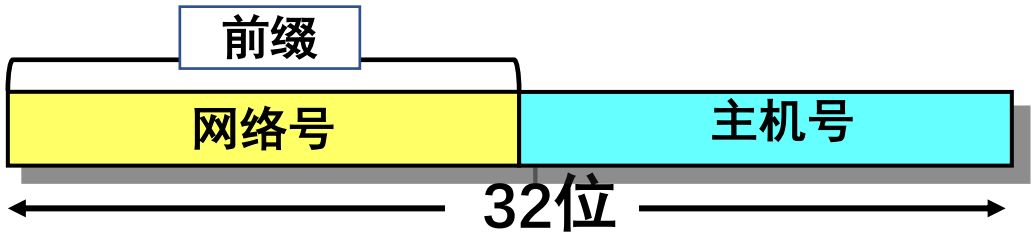
数据报分片

- IPv4分组在传输途中可以多次分片
  - 源端系统、中间路由器等，可通过标志位设定是否允许分片
- 在哪里重组？
  - 重组所需信息：原始数据报编号、分片偏移量、是否收集所有分片
  - 互联网设计原则：途中重组，实施难度大，还有安全问题？
  - IPv4分片只在目的IP对应的目的端系统进行重组
- 分片机制的优缺点探讨
  - 避免分片的IPv6：发出的数据报长度小于路径MTU（路径MTU发现机制）



回顾：IP地址

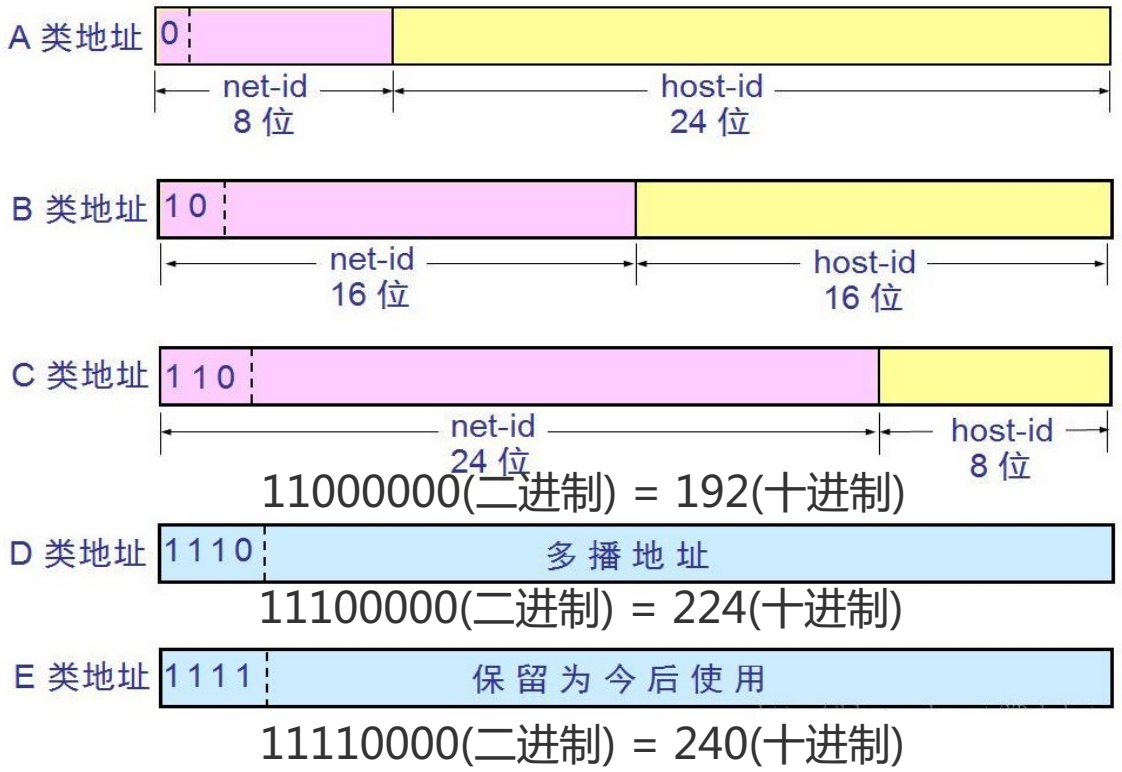
- IP地址
  - 又称逻辑地址，网络上的每一台主机（或路由器）的每一个接口都会分配一个全球唯一的32位的标识符
  - 例如：166.111.4.100/21）
- 将IP地址划分为：网络地址和网络中的主机地址
- 每个网络中的主机数量不同，地址数量不同



网络号又称为网络前缀，如何给全球设备分配地址？

分类的IP地址

- IP地址的书写采用点分十进制记法，其中每一段取值范围为0到255
- IP地址分为五类：A、B、C、D、E，其中A类、B类、C类为单播地址



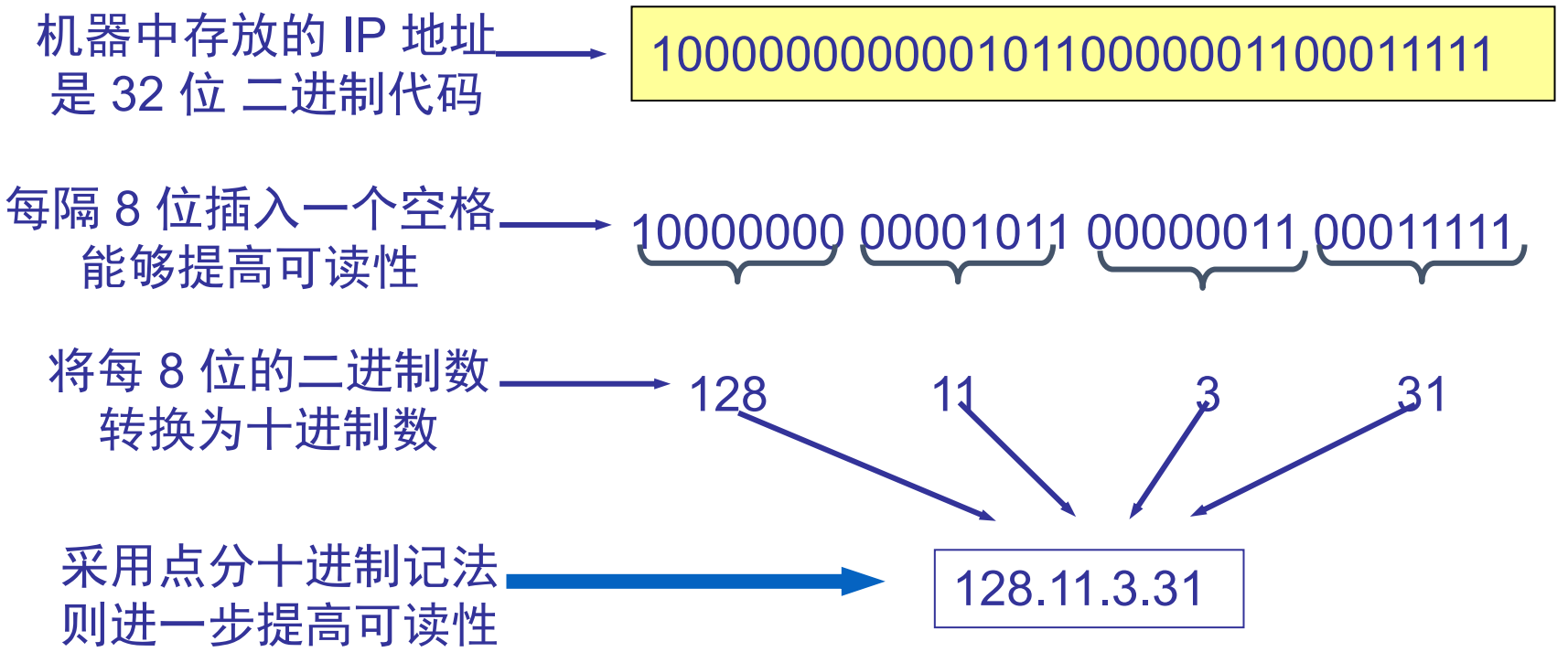
请判断下列地址的类型

10.2.1.1	**A类**
128.63.2.100	**B类**
201.222.5.64	**C类**
256.241.201.10	**不存在，超出范围**

每类网络容纳主机数不同

300台主机需要B类前缀，
但浪费的部分怎么办？

附：点分十进制记法

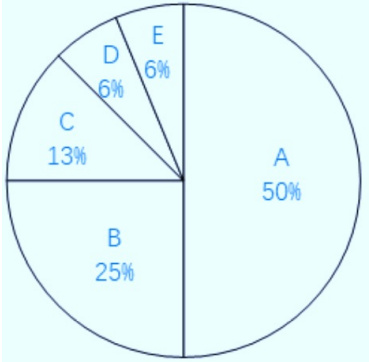

机器中存放的 IP 地址
是 32 位 二进制代码
10000000000001011000000100011111
每隔 8 位插入一个空格
能够提高可读性
10000000 00001011 00000011 00011111
将每 8 位的二进制数
转换为十进制数
128
11
3
31
采用点分十进制记法
则进一步提高可读性
128.11.3.31

分类的IP地址

- 全球地址总数约43亿，2^{32}=4 294 967 296
- 其中ABC 3类常用

类别	最高固定位	第 1 个 8 位组的值	网络位数/个数		主机位数/个数	
A	0	0-127	8b	128	24b	约 1600 万
B	10	128-191	16b	约 1.6 万	16b	约 6.6 万
C	110	192-223	24b	约 210 万	8b	256

- A类网络地址数最少（128个），但总数最多，126 x （2^{24}-2）
- 分类导致极大的浪费



分类的IP地址

- 中国人均约0.2个，而美国一个人拥有4、5个IP

#	国家/地区	IP地址数[3]	%	人口[4][a]	每1000人IP地址数
	(全球已分配)	3,686,475,740	100	8,091,734,930	456
1	美国	1,611,297,420	43.71	343,477,335	4,691
2	中华人民共和国	343,125,576	9.31	1,422,584,933	241
3	日本	189,145,768	5.13	124,370,947	1,521
4	英国	134,054,832	3.64	68,682,962	1,952
5	德国	124,185,676	3.37	84,548,231	1,469
6	韩国	112,495,296	3.05	51,748,739	2,174
7	巴西	87,096,200	2.36	211,140,729	413
8	法国	82,053,600	2.23	66,438,822	1,235
9	加拿大	67,921,556	1.84	39,299,105	1,728
10	意大利	54,020,088	1.47	59,499,453	908
11	荷兰	48,112,552	1.31	18,092,524	2,659

分类的IP地址

2022年6月，中国网民人数10.5亿！
地址缺口巨大！

#	Country or Region	Internet Users 2020 Q1	Internet Users 2000 Q4	Population, 2020 Est.	Population 2000 Est.	Internet Growth 2000 - 2020
1	China	**854,000,000**	22,500,000	**1,439,062,022**	1,283,198,970	3,796 %
2	India	**560,000,000**	5,000,000	**1,368,737,513**	1,053,050,912	11,200 %
3	United States	**313,322,868**	95,354,000	**331,002,651**	281,982,778	328 %
4	Indonesia	**171,260,000**	2,000,000	**273,523,615**	211,540,429	8,560 %
5	Brazil	**149,057,635**	5,000,000	**212,392,717**	175,287,587	2,980 %
6	Nigeria	**126,078,999**	200,000	**206,139,589**	123,486,615	63,000 %
7	Japan	**118,626,672**	47,080,000	**126,854,745**	127,533,934	252 %
8	Russia	**116,353,942**	3,100,000	**145,934,462**	146,396,514	3,751 %
9	Bangladesh	**94,199,000**	100,000	**164,689,383**	131,581,243	94,199 %
10	Mexico	**88,000,000**	2,712,400	**132,328,035**	2,712,400	3,144 %

IP特殊地址

地址	用途
网络127.0.0.0	指本地，用于测试（浪费了1700万个地址☹）
全0主机地址	用于指定网络本身，称之为网络地址或者网络号
全1主机地址	用于广播，也称定向广播，需要指定目标网络
0.0.0.0/0	匹配任意地址
255.255.255.255	用于本地广播，也称有限/受限广播，无须知道本地网络地址

主机上有路由表吗？

```
dgdeMacBook-Pro:~ yongcui$ netstat -nr
Routing tables
Internet:
Destination        Gateway            Flags        Netif
default            58.206.196.1       UGSc           en0
58.206.196/22      link#6             UCS            en0
127                127.0.0.1          UCS            lo0
```

```
dgdeMacBook-Pro:~ yongcui$ ipconfig getifaddr en0
58.206.198.135
```

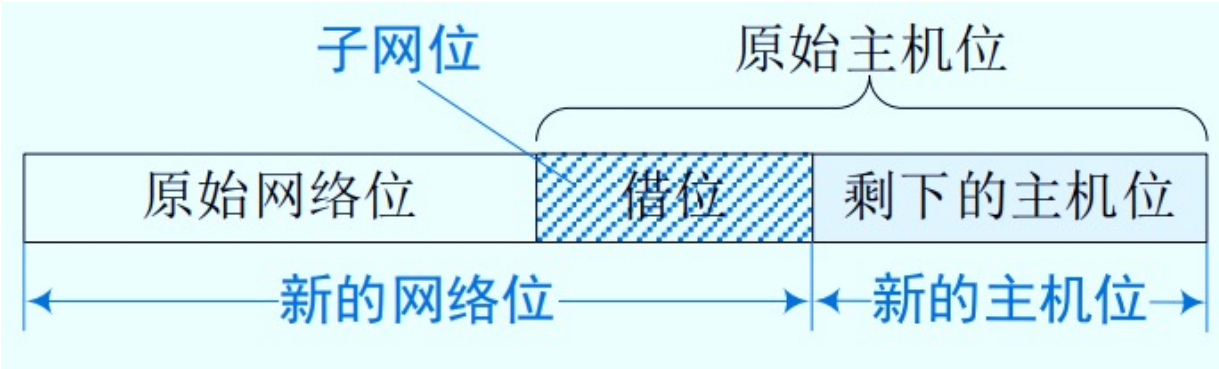
如何提升IP地址利用率？

- 在 ARPANET 的早期，IP 地址的设计确实不够合理
  - IP 地址空间的利用率有时很低
  - 给每一个物理网络分配一个网络号会使路由表变得太大,更新频繁，因而使网络性能变坏
  - 两级的 IP 地址不够灵活

如何解决？子网划分，无类域间路由CIDR

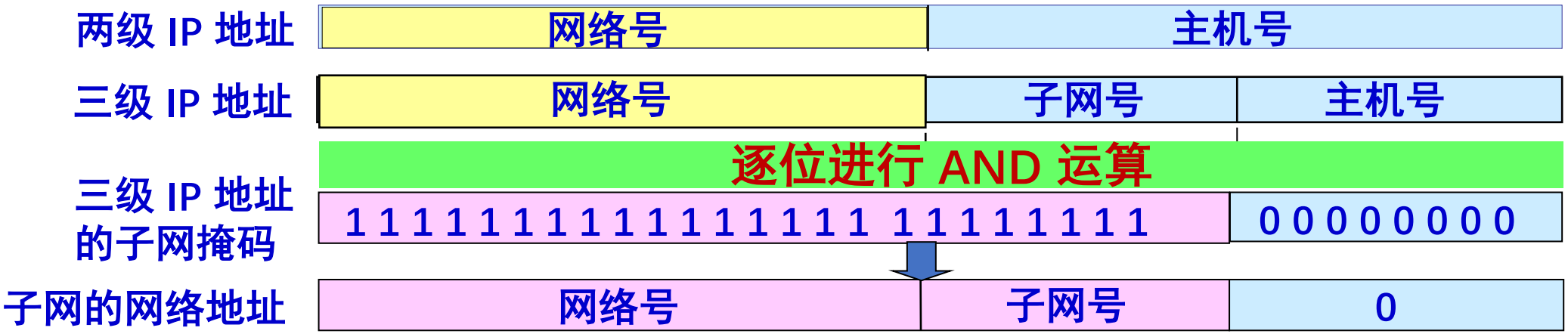
子网划分

- 子网划分：把原有 IP 地块址划分为跟物理网络（或 VLAN）大小相适应的 IP 地址块
  - 机构/单位内部的事情
- 划分方法：借位
- 借位原则
  - 从主机域的高位开始借位。从主机域的高位开始借位，可以保证所借的位紧紧挨着原网络位，剩下的主机位仍然在是一整块、连续的主机号
  - 主机域至少要保留 2 位



子网划分

- 子网划分(subnetting)，在网络内部将一个网络块进行划分，以供多个内部网络使用，对外仍是一个网络
- 子网(subnet)，一个网络进行子网划分后得到的一系列结果网络称为子网
- 子网掩码(subnet mask)，与 IP 地址一一对应，是32 bit 的二进制数，置1表示网络位，置0表示主机位



子网划分

172.16.2.160/26

地址	172	16	2	160
掩码	255	255	255	192
	10101100	00010000	00000010	10100000
	11111111	11111111	11111111	11000000
前缀	←			→ 主机位
	10101100	00010000	00000010	10000000
	10101100	00010000	00000010	10111111

逐位进行 AND 运算

主机位全0，子网地址
172.16.2.128

主机位全1，广播地址
172.16.2.191

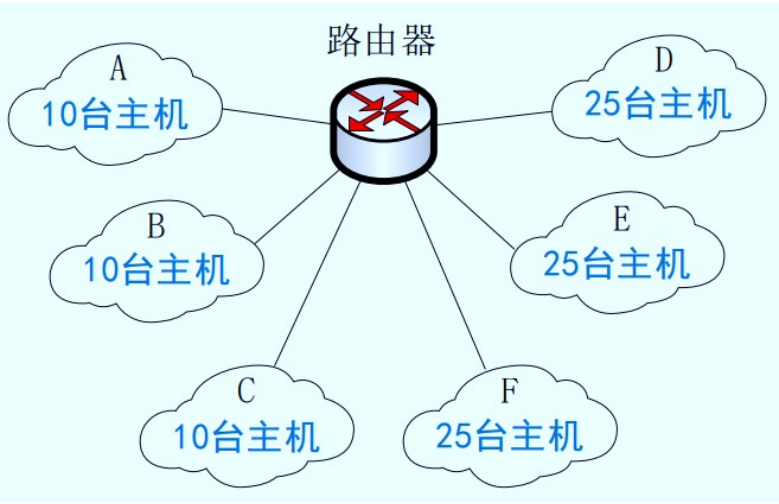
可分配IP地址范围
172.16.2.128+1~
172.16.2.191-1

子网拥有主机数量
$2^n-2=62$（n=6）

路由转发分组时基于路由匹配选择出接口

例5-3

- 一个机构部署了一个网络，包含 6 个物理网络，即 6 个子网，其拓扑和各子网挂接的主机数，如所示，有一个网络地址块 192.168.1.0/24 可用于该网络，应该如何规划这个地址块？



解答：

- 需求分析：6个子网，最多主机数25
- 借3 位才能满足子网数量的需求；剩余 5 位主机位，可以提供 30个可用 IP 地址

子网序号	子网掩码	子网络地址	子网络广播地址	可用的地址范围	分配
1	255.255.255.224（或/27）	192.168.1.0	192.168.1.31	192.168.1.1~30	A
2		192.168.1.32	192.168.1.63	192.168.1.33~62	B
3		192.168.1.64	192.168.1.95	192.168.1.65~94	C
4		192.168.1.96	192.168.1.127	192.168.1.97~126	D
5		192.168.1.128	192.168.1.159	192.168.1.129~158	E
6		192.168.1.160	192.168.1.191	192.168.1.161~190	F
7		192.168.1.192	192.168.1.223	192.168.1.193~222	备用
8		192.168.1.224	192.168.1.255	192.168.1.226-254	备用

例5-3，子网划分技巧

例：一个主机的IP地址是202.112.14.37，掩码是255.255.255.240，要求计算这个主机所在网络的网络地址和广播地址。

解答：

1）从掩码推算子网可容纳的IP地址数量：容纳的IP地址有256 - 240 = 16个（包括网络地址和广播地址）；

2）子网网络地址是可容纳IP数量的整数倍，如202.112.14.0、16、32、48……

3）网络地址 < 子网IP地址 < 广播地址，而广播地址是下一个网络地址减1，如32 < 37 < 47（48-1），所以202.112.14.37所在的网络地址和广播地址分别是202.112.14.14.32和202.112.14.47。

例5-3，子网划分技巧

例：一个C类地址，需要具有10台主机的子网，请进行子网地址的规划和计算子网掩码。

解答：

1）计算子网需要的IP地址数量，如10+1+1=12；

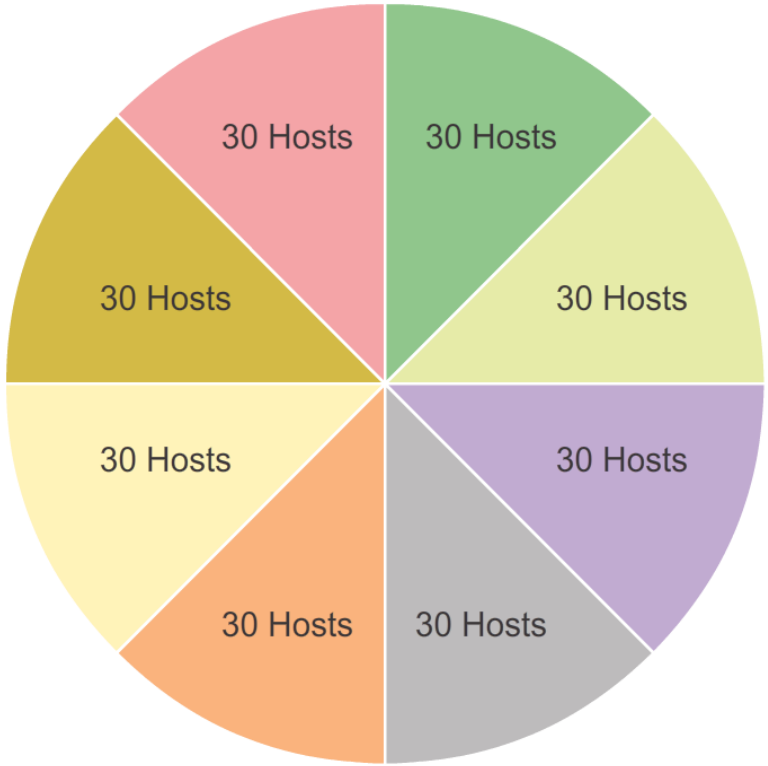
2）找到满足大于所需IP数量的最小2^n，如$12<16=2^4$；

3）n就是主机位数，确定借位，如4位主机位，借位8-4=4；

4）写出子网掩码，如255.255.255.240，其中240=256-16；或8*3+4=28，“/28”

固定长度子网划分浪费了地址

- 传统子网划分——为每个子网分配相同数量的地址。
- 需要较少地址的子网中存在未使用（浪费）的地址。例如，链路只需要2个地址。
- 可变长子网掩码（VLSM）或细分子网可以提供更有效的地址使用。

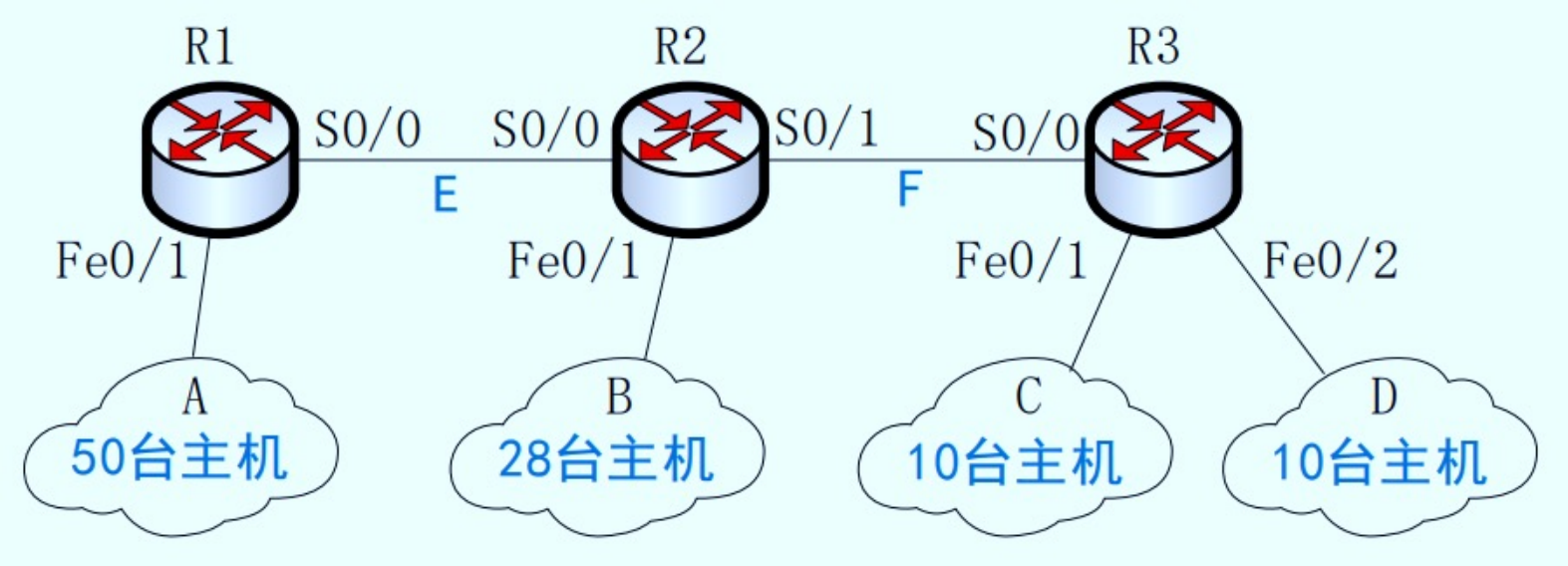


可变长子网掩码的子网划分

- 可变长子网掩码的子网划分：可以对子网进一步划分，继续借位，创建更小的子网，精打细算，尽量不浪费地址；这样子网位数不再是定长的，而是变化的
  - 子网的规模不同
- 可变长子网掩码子网划分方法
  - 满足子网个数的需求
  - 先满足大规模子网的需求
  - 最后满足小规模子网的需求

例5-4

- 一个机构部署了一个企业网络，由三台路由器连接了 A、B、C、D、E、F 共 6 个物理网络， 每个子网的规模其拓扑如图 5-17 所示；有一个网络地址块 192.168.6.0/24 可用于该网络，应该如何规划这个地址块？



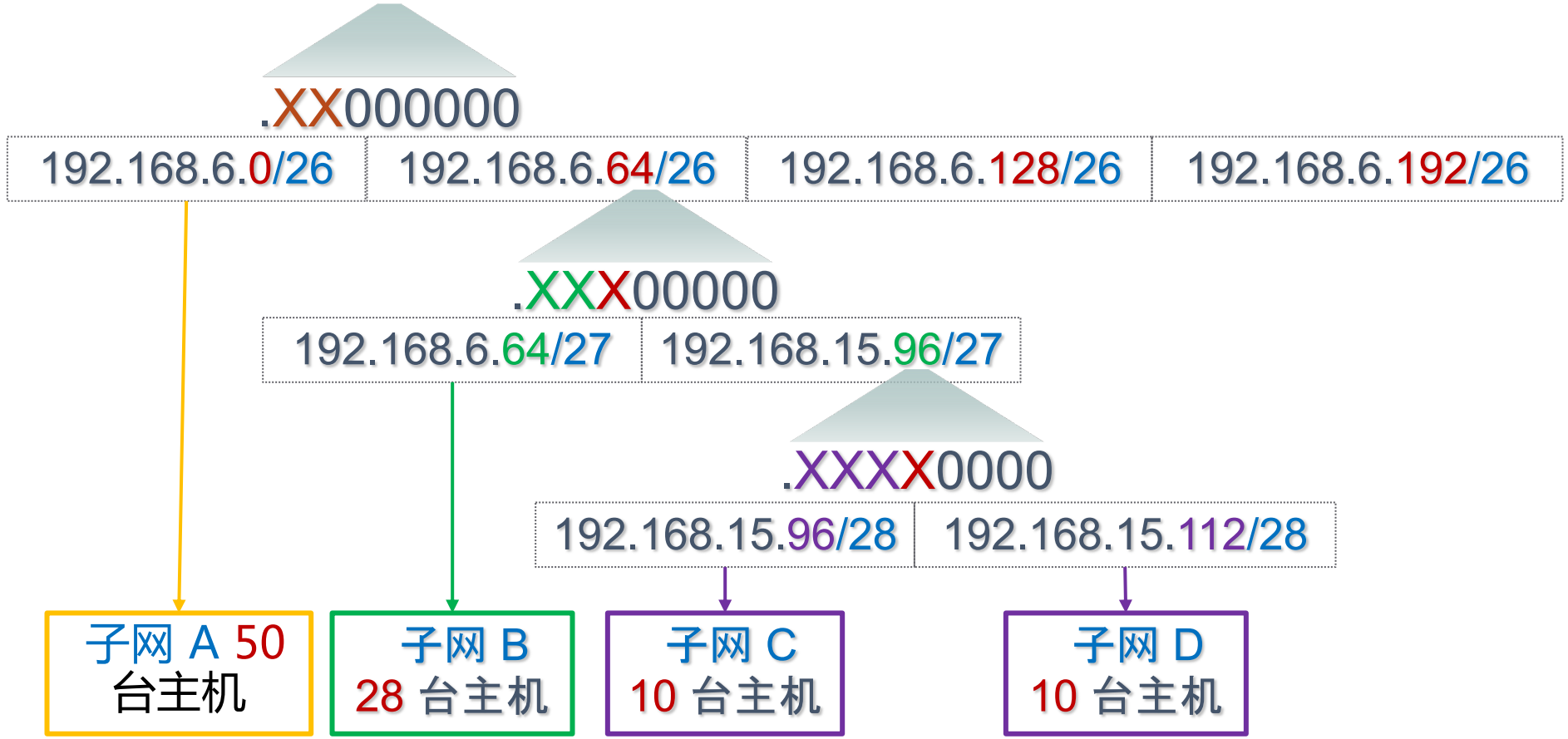
定长子网掩码无法满足需求！

例5-4

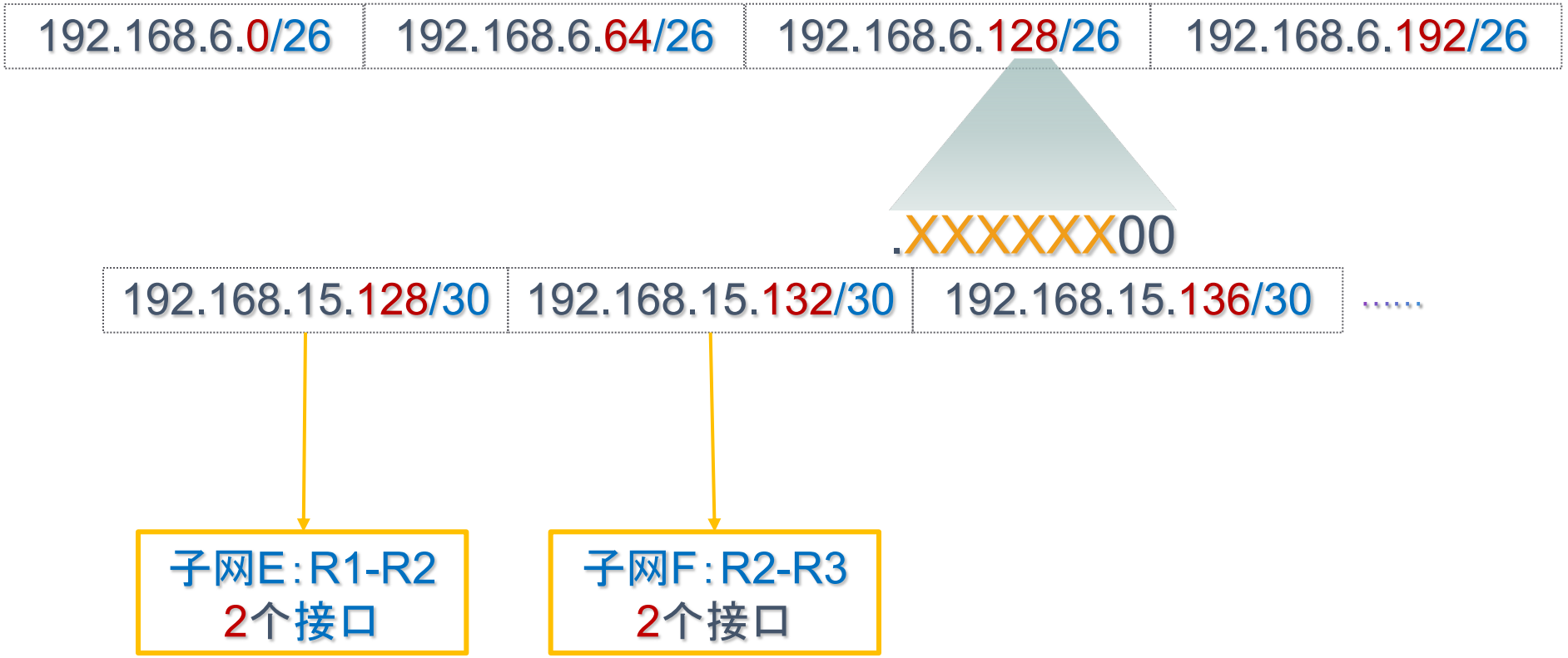
- 需求分析：网络 A、B、C 和 D 分别需要 51 个、29 个、11 个和 11 个可用的 IP 地址
- 最大的子网A需要主机位6位才能满足需求，所以，首先借2位，创建4个子网，第1个子网分配给A
- 将第2个子网继续划分，借1位，生成2个子网，其中一个分配给B
  - 剩下的那个子网继续划分，再借1位，生成2个子网，正好满足C和D的需求
- 将第3个子网192.168.6.128/26继续划分……

例5-4

192.168.6.0/24

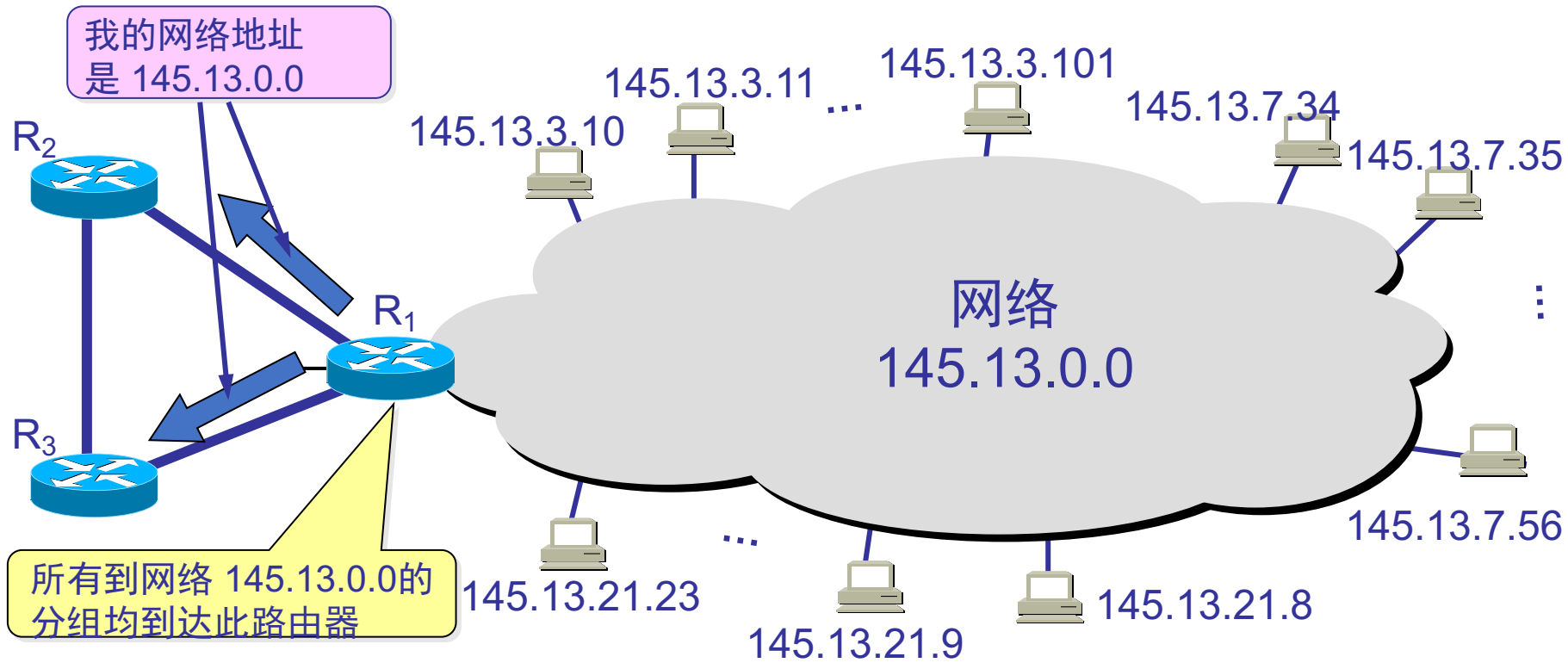


例5-4（续）

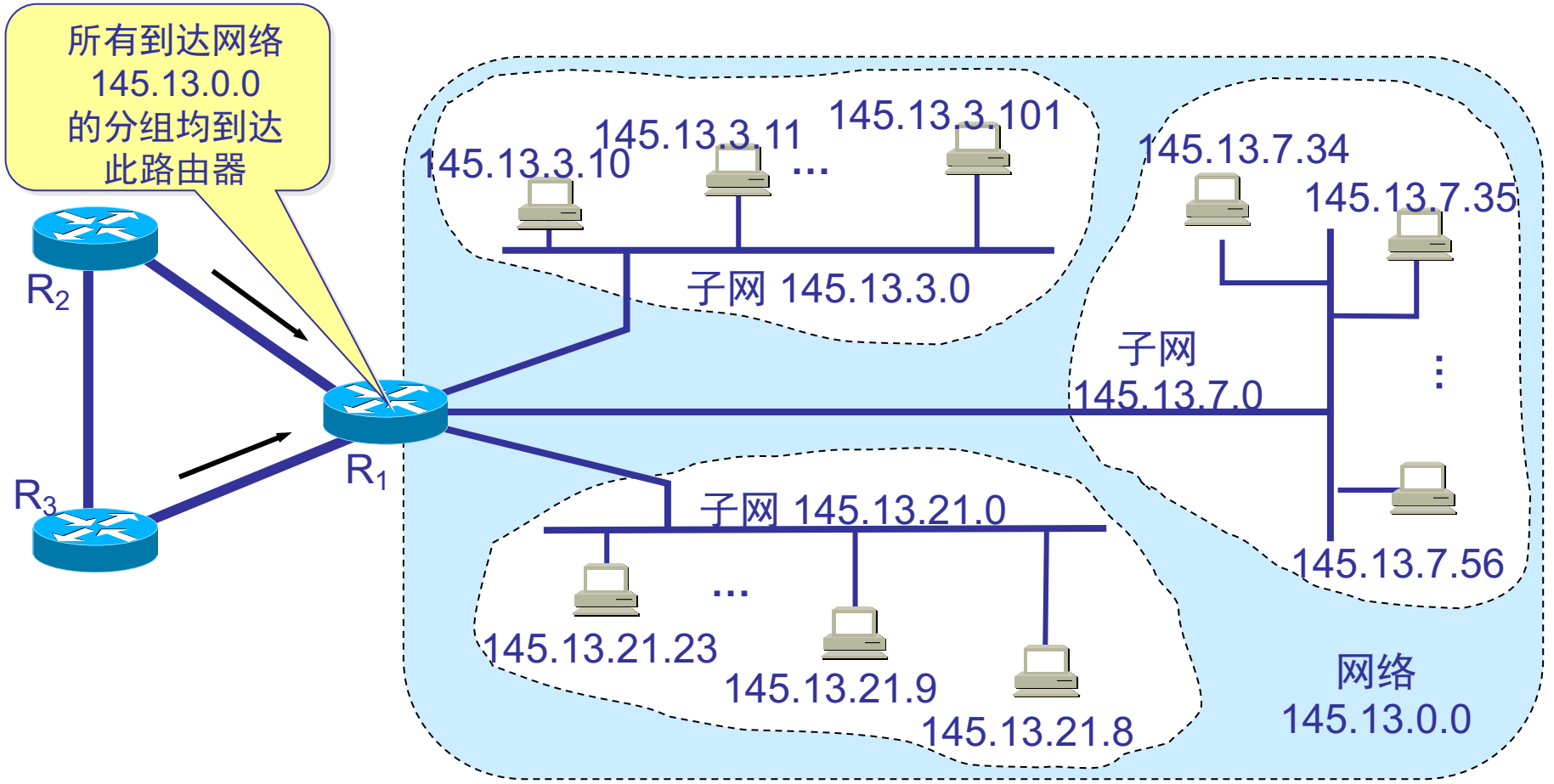

192.168.6.0/26
192.168.6.64/26
192.168.6.128/26
192.168.6.192/26
.XXXXXX00
192.168.15.128/30
192.168.15.132/30
192.168.15.136/30
.....
子网E：R1-R2
2个接口
子网F：R2-R3
2个接口

子网划分是网络内部行为，对外仍是一个网络

一个未划分子网的 B 类网络145.13.0.0



子网划分是网络内部行为，对外仍是一个网络

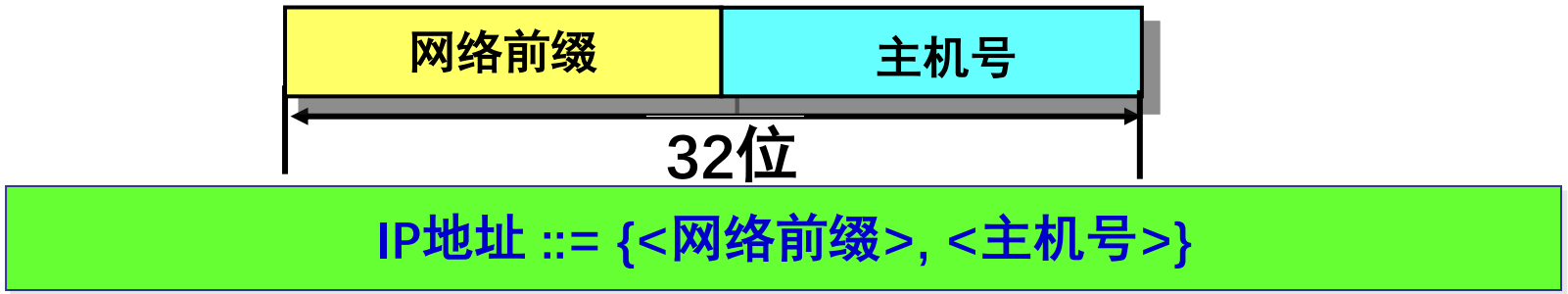


子网划分一定程度解决了IP地址使用率的问题，然而

- 1990s年代，互联网面临的问题
  - B 类地址空间耗尽
  - 互联网路由器的路由表规模增长，超出了当时的软件、硬件和有效管理能力
  - 32 位的 IPv4 地址空间也终将全部耗尽（实际上，2011 年 2 月，总地址池已经枯竭
- 无类域间路由 CIDR 并不试图解决第三个问题，而是通过提供一种机制去降低 IPv4 地址的耗尽速度以及路由表的增长速度

无类域间路由

- CIDR（ Classless Inter-Domain Routing ）
  - 按需分配地址块：不再按照A\B\C地址类别划分
  - 减少 IP 地址浪费：使得网络的组织更加灵活、便于维护和管理
  - 将32位的IP地址划分为前后两个部分，并采用斜线记法，即在IP地址后加上“/”，然后再写上网络前缀所占位数，如172.16.2.160/26



  - 一个 CIDR 地址块可以表示很多地址，这种地址的聚合常称为路由聚合（route aggregation），也称为构成超网 (supernet)
  - 聚合技术在Internet中大量使用，导致大量前缀重叠

例5-5

- 按需分配地址块：不再按照A\B\C地址类别划分
- 【例 5-5】一个 1000 人的公司，申请 1000 个主机 IP 地址。

  按照 CIDR 的无类编址机制为该公司提供地址分配

  - 解答：首先确定主机位、找到大于 1000 且最靠近 1000 的 2^n。此时 n=10，表示用 10 位主机位的 IP 地址，即可满足该公司的地址数需求，只有少量余量，可供后续扩展之用。
  - 网络位数=总位数-主机位数=32-10=22，所以只需要分配“/22”地址块，就满足了公司的 IP 地址数需求。

例5-6

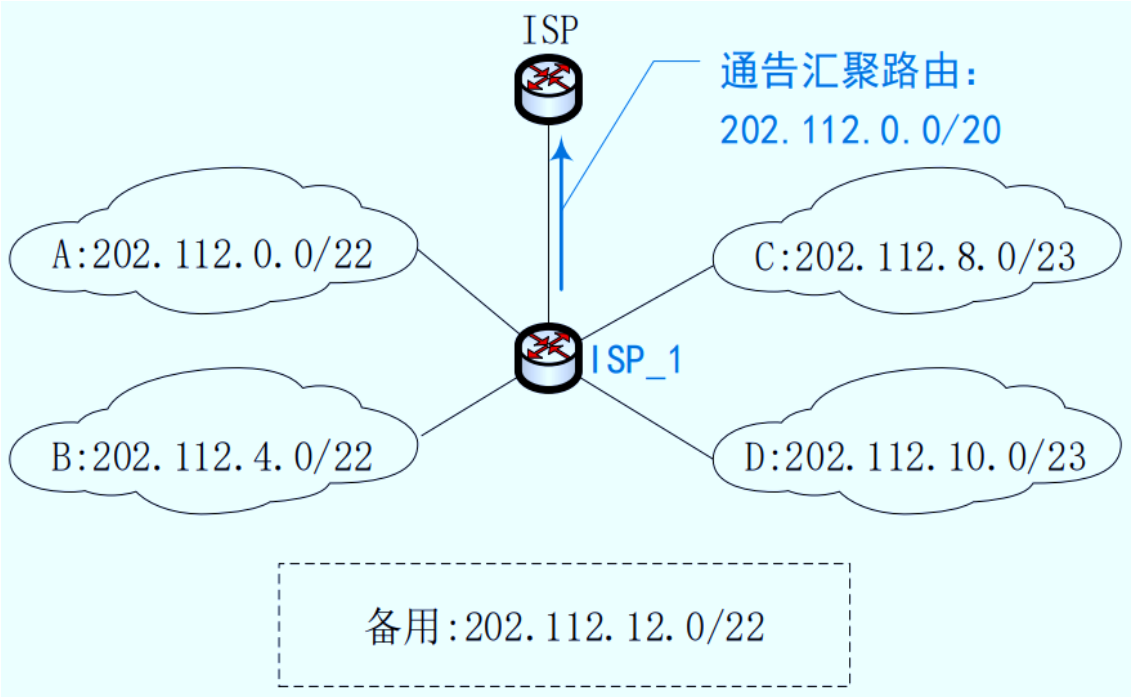
- 一个互联网服务提供商 ISP_1 获得了一个 IP 地址块202.112.0.0/20，现在有 4 个学校 A、B、C 和 D 向 ISP_1 分别申请 IP 地址数 1000、1000、500 和 500，ISP_1 为 4 个学校分配满足需求的 IP 地址

解答：

- 地址块 202.112.0.0/20 的网络位是 20 位，含 4096 个 IP 地址
- A 学校分配了网络位为 22 的 IP 地址块 202.112.0.0/22，其中，第一个 IP 地址是网络地址202.112.0.0，最后一个 IP 地址是广播地址 202.112.3.255，共提供 1024 个 IP 地址，满足需求。
- B 学校分配了网络位为 22 的 IP 地址块 202.112.4.0/22
- C 学校分配了网络位为 23 的 IP 地址块 202.112.8.0/23
- D 学校分配了网络位为 23 的 IP 地址块 202.112.10.0/23
- 余下的地址块为 202.112.12.0/22，备用。

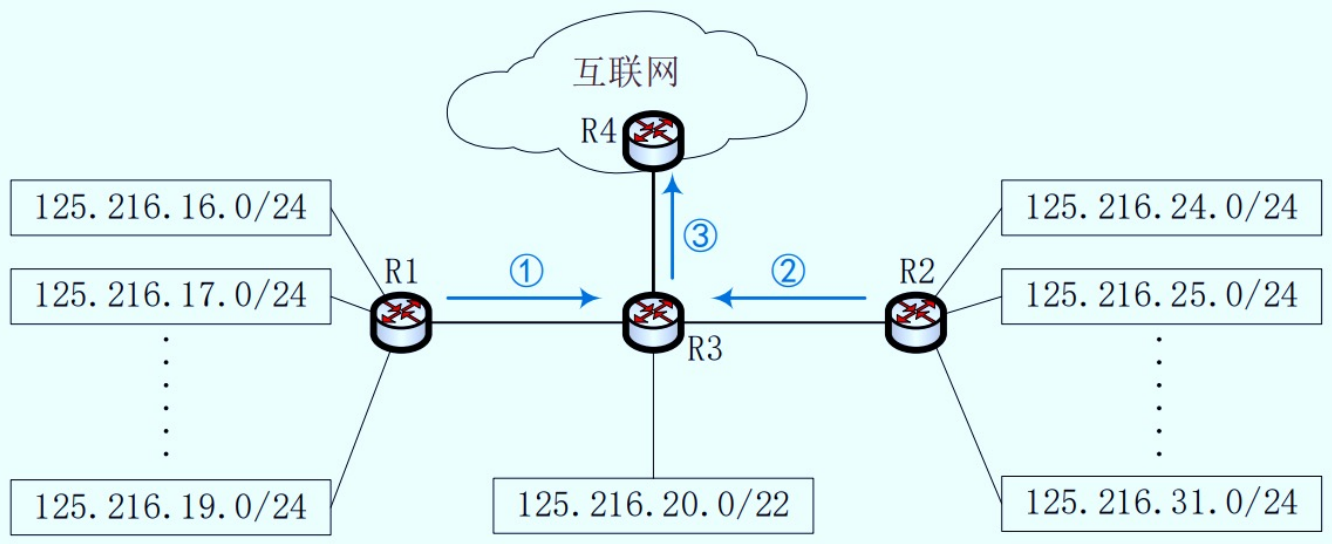
CIDR: 路由汇聚

- 路由汇聚/汇总
- 汇聚结果：超网，采用“基地址/网络位数”表示
  - 其中的基地址，指的是形成的超网地址空间中的第一个地址
- 汇聚方法
  - 网络位：地址空间中不变的位



例5-7

- 一个网络的拓扑如图所示，路由器 R1 直接连接了 4 个“/24”的网络，路由器 R2 直接连接了 8 个“/24”的网络，路由器 R3 连接了 1 个“/22”的直连网络，还连接了路由器 R1 和 R2；R1 向 R3通告时发生了路由汇聚，用①表示；R2 向 R3 通告时发生了路由汇聚，用②表示；R3 向 R4 通告时，也发生了路由汇聚，用③表示。请分别写出汇聚①、②、③发生后产生的超网



- ①4个子网汇聚：25.216.16.0/22
- ②8个子网汇聚：125.216.24.0/21
- ③2个超网和1个直连子网汇聚：125.216.16.0/20

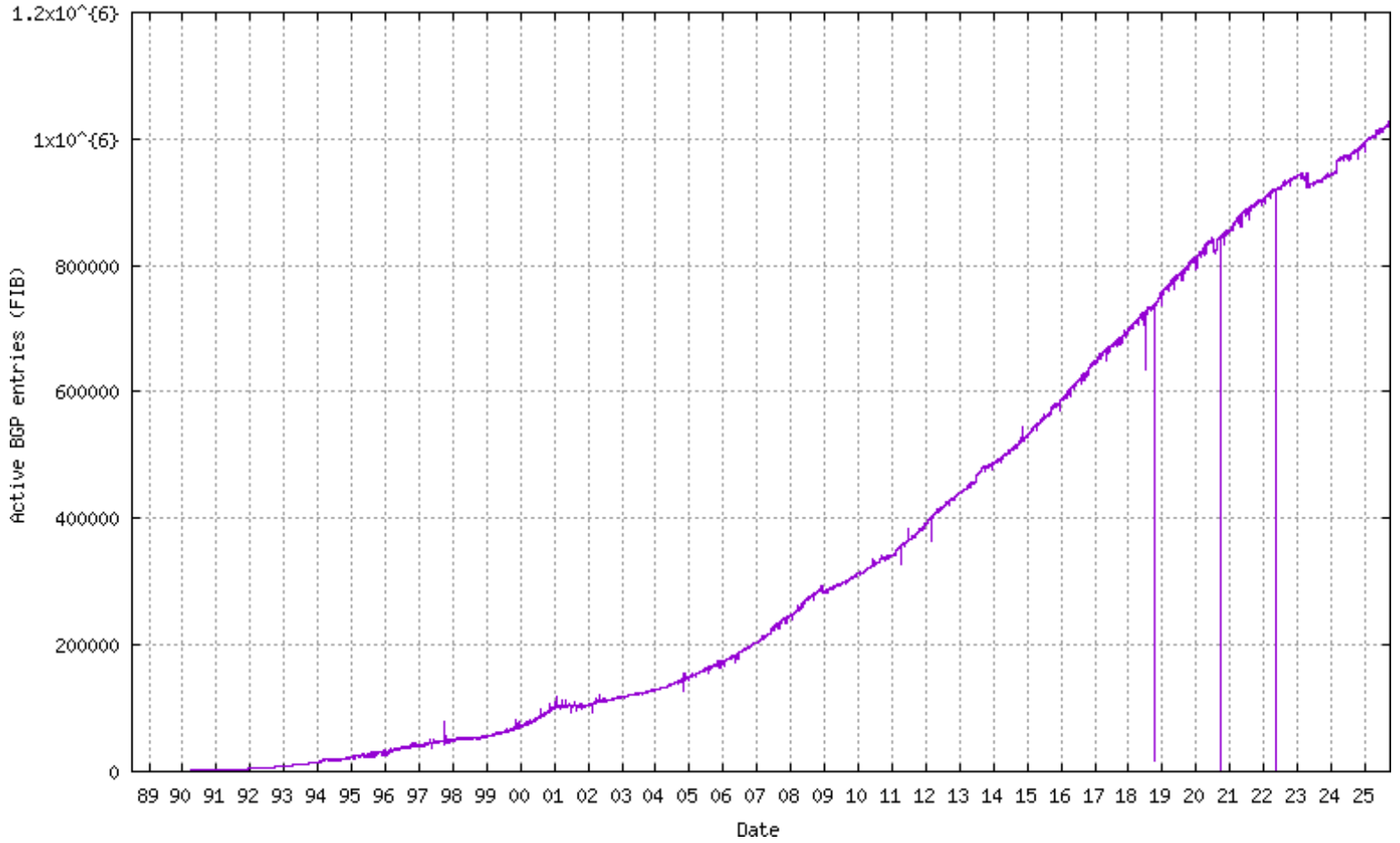
不变的位	
16：000100	00
17：000100	01
18：000100	10
19：000100	11

不变的位	
24：00011	000
25：00011	001
：	：
31：00011	111

不变的位	
16：0001	0000
17：0001	0001
：	：
20：0001	0100
：	：
31：0001	1111

CIDR的部署及效果

- 采用 CIDR 地址分配机制之后，B 类地址枯竭的趋势得到了遏制（虽然没有改变地址耗尽的命运），同时，CIDR 也改写了路由表的指数增长趋势



最长前缀匹配

最长前缀匹配（Longest prefix match）

- CIDR可变长子网掩码以及路由聚合，需要最长前缀匹配来实现最精确匹配
- IP地址与IP前缀匹配时，总是选取子网掩码最长的匹配项
- 主要用于路由器转发表项的匹配

IP前缀	匹配地址范围	出接口号
200.23.16.0/21	11001000 00010111 00010*** ********	0
200.23.24.0/23	11001000 00010111 0001100* ********	1
200.23.24.0/21	11001000 00010111 00011*** ********	2
Otherwise 0.0.0.0/0	******** ******** ******** ********	3

IP地址：200.23.22.161（11001000 00010111 00010110 10100001），接口0

IP地址：200.23.24.170（11001000 00010111 00011000 10101010），接口1

最长前缀匹配

2.128.0.0/9	00000010 10000000	interface 1
2.192.0.0/10	00000010 11000000	interface 2
2.0.0.0/8	00000010 00000000	interface 3
2.2.3.0/24	00000010 00000010 00000011	interface 4
0.0.0.0/0		interface 5

根据最长前缀匹配，下述目的IP将匹配哪个表项（出接口）？

2.5.1.2	00000010 00000101 00000001 00000010	Interface 3
2.150.1.2	00000010 10010110 00000001 00000010	Interface 1

最长前缀匹配

2.128.0.0/9	00000010 10000000	interface 1
2.192.0.0/10	00000010 11000000	interface 2
2.0.0.0/8	00000010 00000000	interface 3
2.2.3.0/24	00000010 00000010 00000011	interface 4
0.0.0.0/0		interface 5

根据最长前缀匹配，下述目的IP将匹配哪个表项（出接口）？

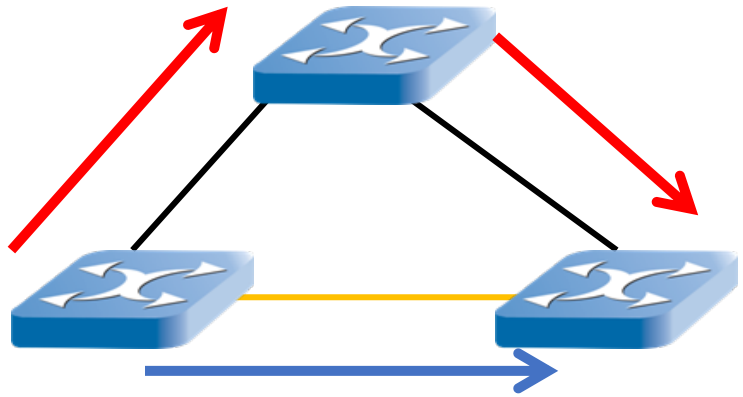
2.200.1.2 00000010 11001000 00000001 00000010 Interface 2

3.150.1.2 00000011 10010110 00000001 00000010 Interface 5

最长前缀匹配

- CIDR + 子网划分
  - 细粒度分配地址前缀
  - 相同前缀的网段地址可以统一路由
  - 地址结构可以无冲突地按照树形逻辑管理
- 树形？任意拓扑？
  - 子网间互联，有必要绕道上层吗？
  - 如果都经过上层，方案扩展性？
  - ” 抄近道 “显然更优

路由转发表	
首部寻址信息	输出接口
0100 0101 0111 1001	3 2 2 1



如何用某种机制描述这种“抄近道”的路由策略？

路由转发分组时
基于路由匹配选择出接口

回顾：IP特殊地址

地址	用途
网络127.0.0.0	指本地，用于测试（浪费了1700万个地址☹）
全0主机地址	用于指定网络本身，称之为网络地址或者网络号
全1主机地址	用于广播，也称定向广播，需要指定目标网络
0.0.0.0/0	匹配任意地址
255.255.255.255	用于本地广播，也称有限/受限广播，无须知道本地网络地址

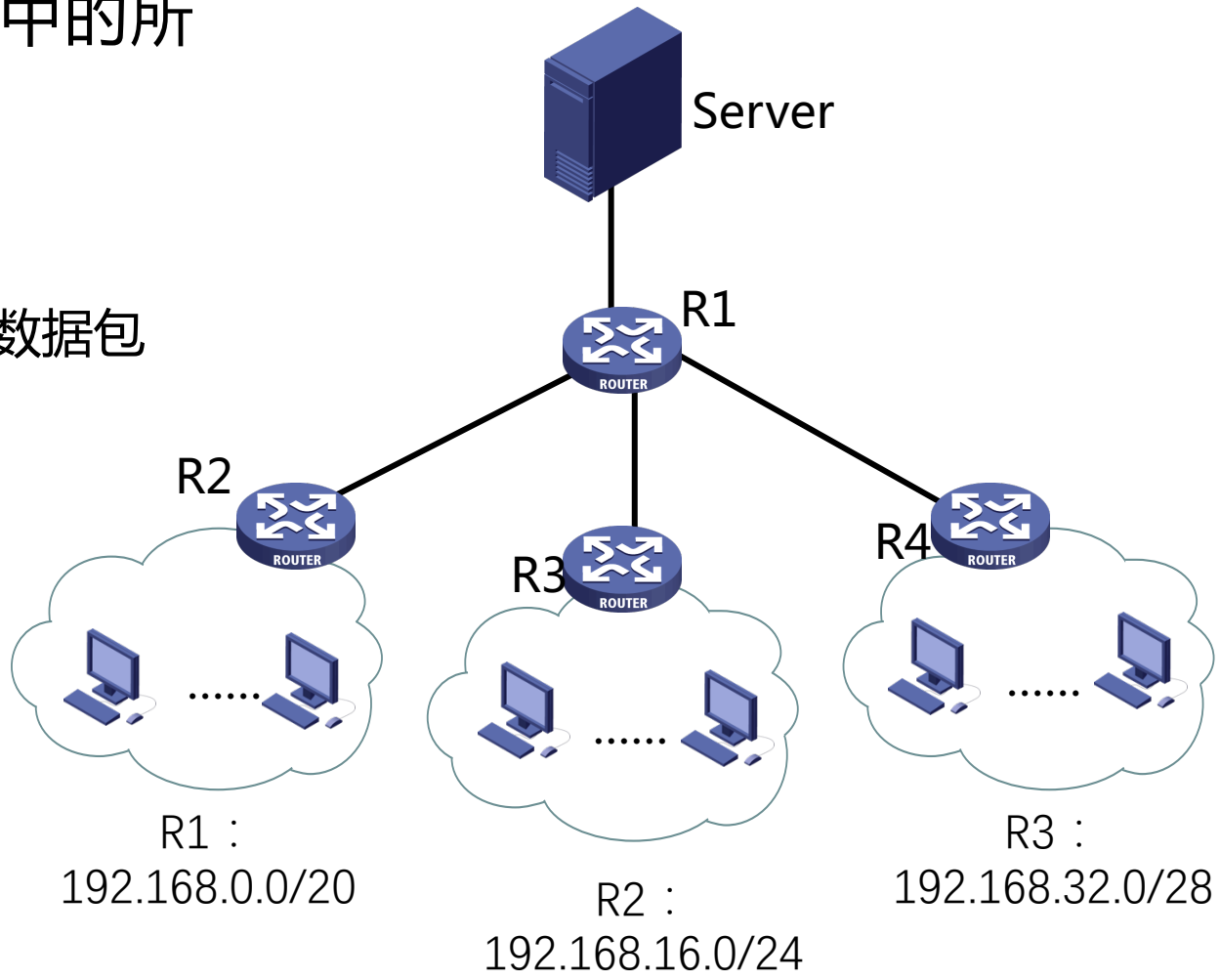
主机上有路由表吗？

```
dgdeMacBook-Pro:~ yongcui$ netstat -nr
Routing tables
Internet:
Destination        Gateway            Flags        Netif
default            58.206.196.1       UGSc           en0
58.206.196/22      link#6             UCS            en0
127                127.0.0.1          UCS            lo0
```

```
dgdeMacBook-Pro:~ yongcui$ ipconfig getifaddr en0
58.206.198.135
```

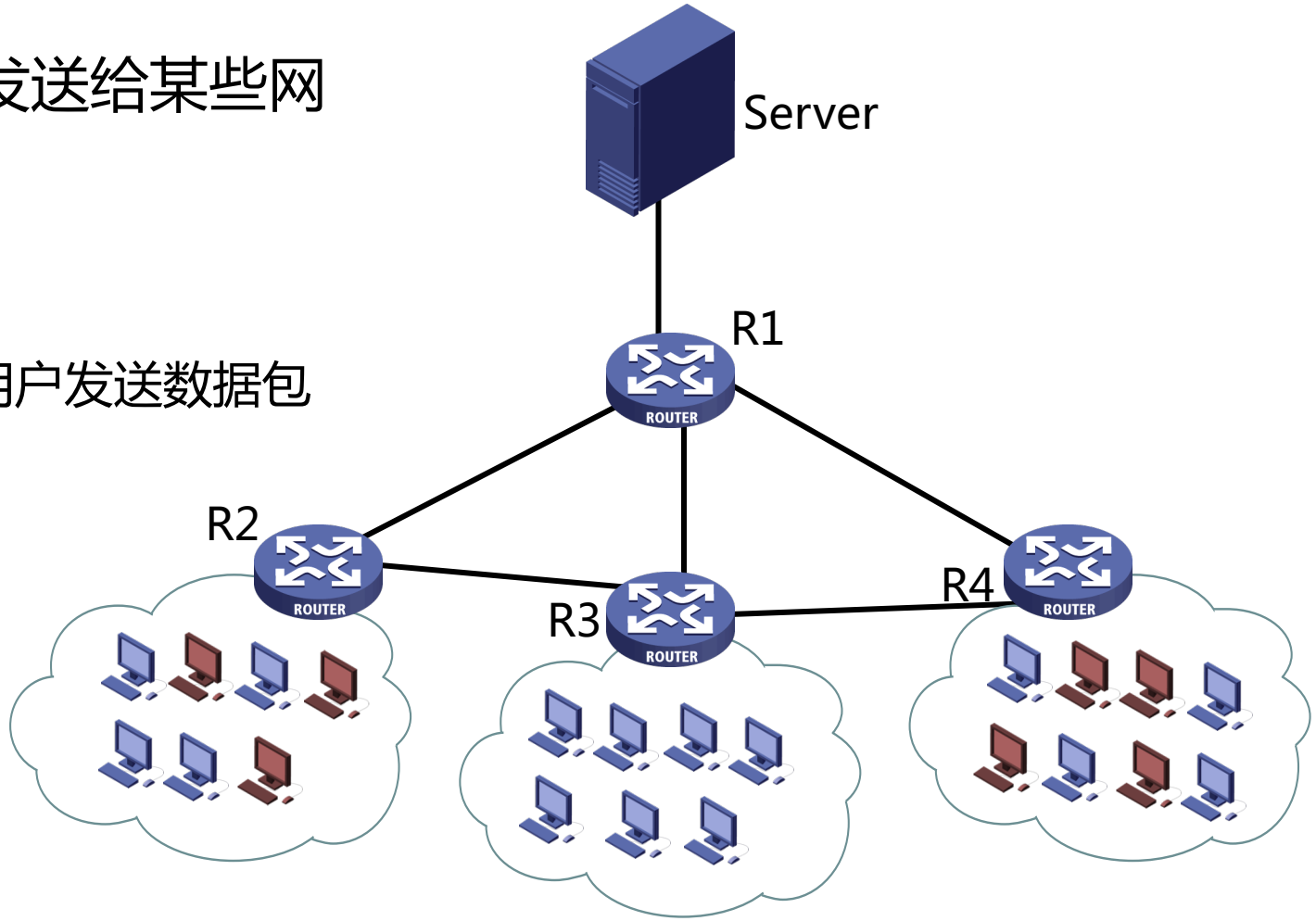
广播路由

- 服务器希望将视频广播给3个网络中的所有30个用户
- 广播（Broadcast）
  - 源主机同时给全部目标地址发送同一个数据包
- 广播地址
  - 网段内全1主机地址为广播地址
  - R1广播地址：192.168.15.255
  - R2广播地址：192.168.16.255
  - R3广播地址：192.168.32.15



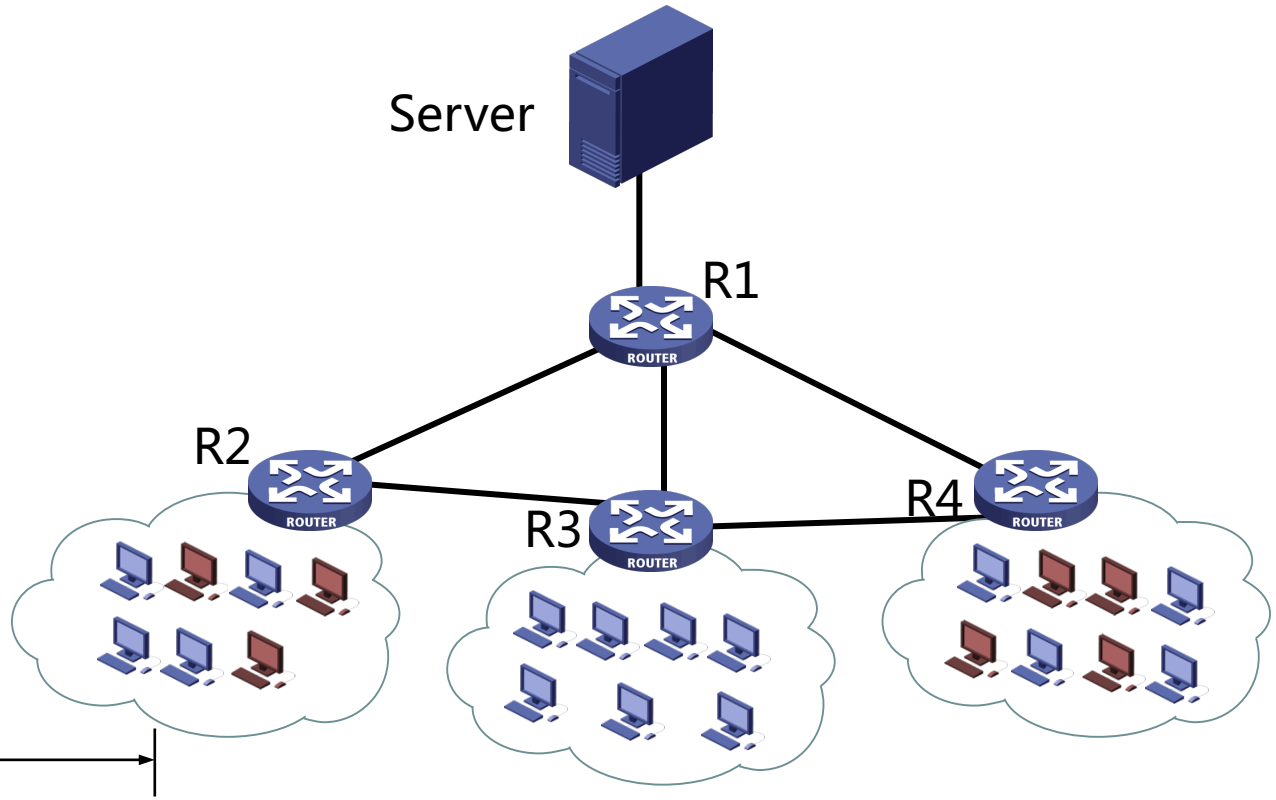
组播路由

- 服务器希望将体育直播视频发送给某些网络中的个别用户
- 组播（Multicast）
  - 源主机给网络中的一部分目标用户发送数据包



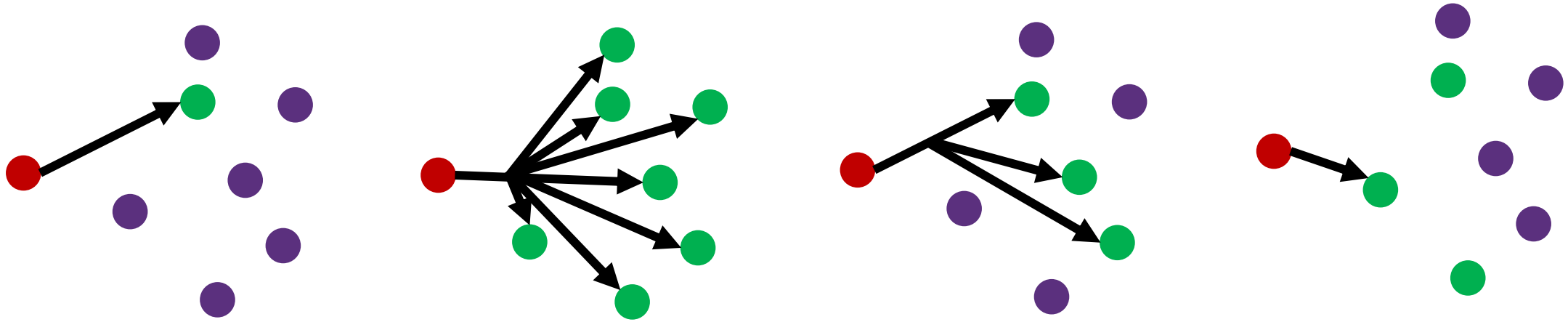
组播路由

- 常用组播地址段：224.0.0.0/24
- 局域网组播地址(一跳子网内使用)
  - 224.0.0.1 LAN上所有设备
  - 224.0.0.2 LAN上所有路由器
  - 224.0.0.5 LAN上所有OSPF路由器
  - 224.0.0.9 LAN上所有RIP路由器
  - 224.0.0.251 LAN上所有DNS服务器



单播-广播-组播-任播

- 不同的需求，不同的设计
  - 单播Unicast：一对一通信，传给特定成员
  - 广播Broadcast ：一对多通信，传给子网全部成员
  - 组播Multicast：一对多通信，传给组内的部分成员
  - 任播Anycast：一对一通信，传给最近的组员

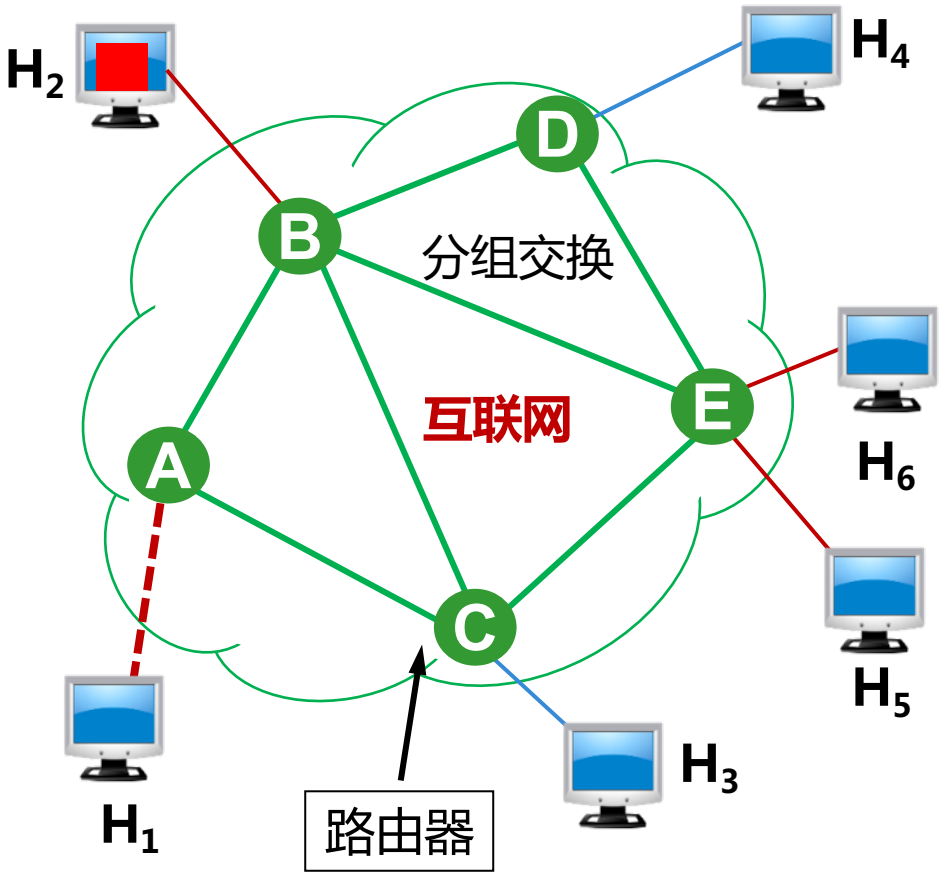


IPv4与编址-小结

- IPv4协议 & 数据报分片
  - IPv4报文格式：核心功能（编址）+ 其他功能（最大帧长 & 分片 Fragmentation、包头检错 & 校验和、环路避免 & TTL）
- 编址问题
  - 定长而灵活的编址？
  - 分类地址：ABCDE类，前缀固定长度，不够灵活
  - CIDR：任意长度前缀（前缀长度 || 子网掩码）
  - CIDR地址聚合，怎么转发？怎么“抄近道“？最长前缀匹配
- 特殊的IP地址
  - 子网大小计算时的保留地址
  - 不同的需求构成了不同的设计(单播、组播、广播、任播)

IPv4地址如何获取

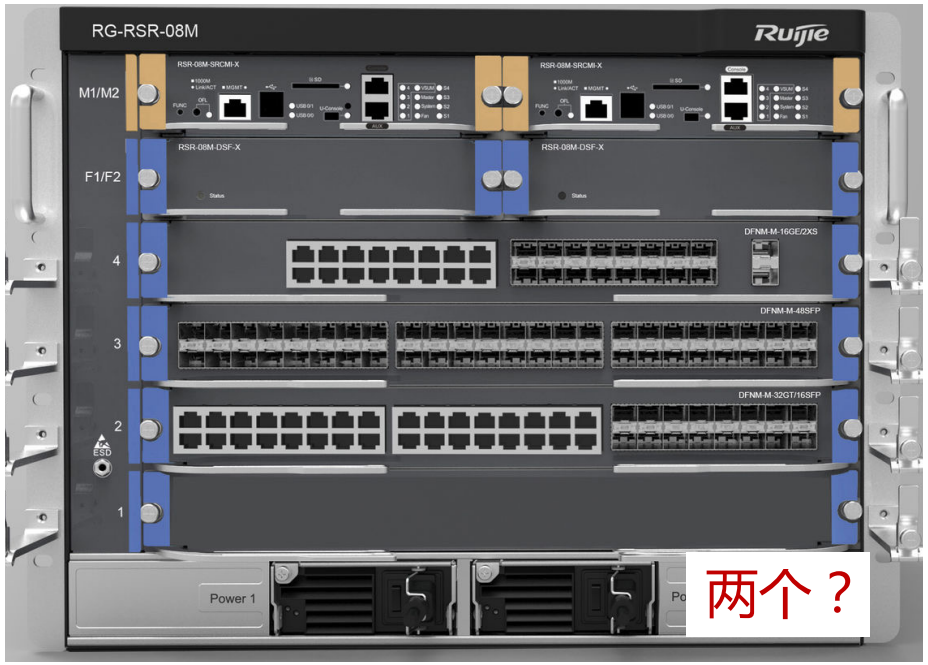
- 配置IP地址的需求是什么？
  - 全球唯一，便于使用
- 公有IP地址要求全球唯一
  - 互联网名字与编号分配机构向ISP分配，ISP再向所属机构或组织逐级分配
  - ICANN（Internet Corporation for Assigned Names and Numbers）
- 公有IP地址逐级分配
  - ICANN、CNNIC、中国教育科研网CERNET、清华校园网、计算机系网管
  - CNNIC: 中国互联网络信息中心

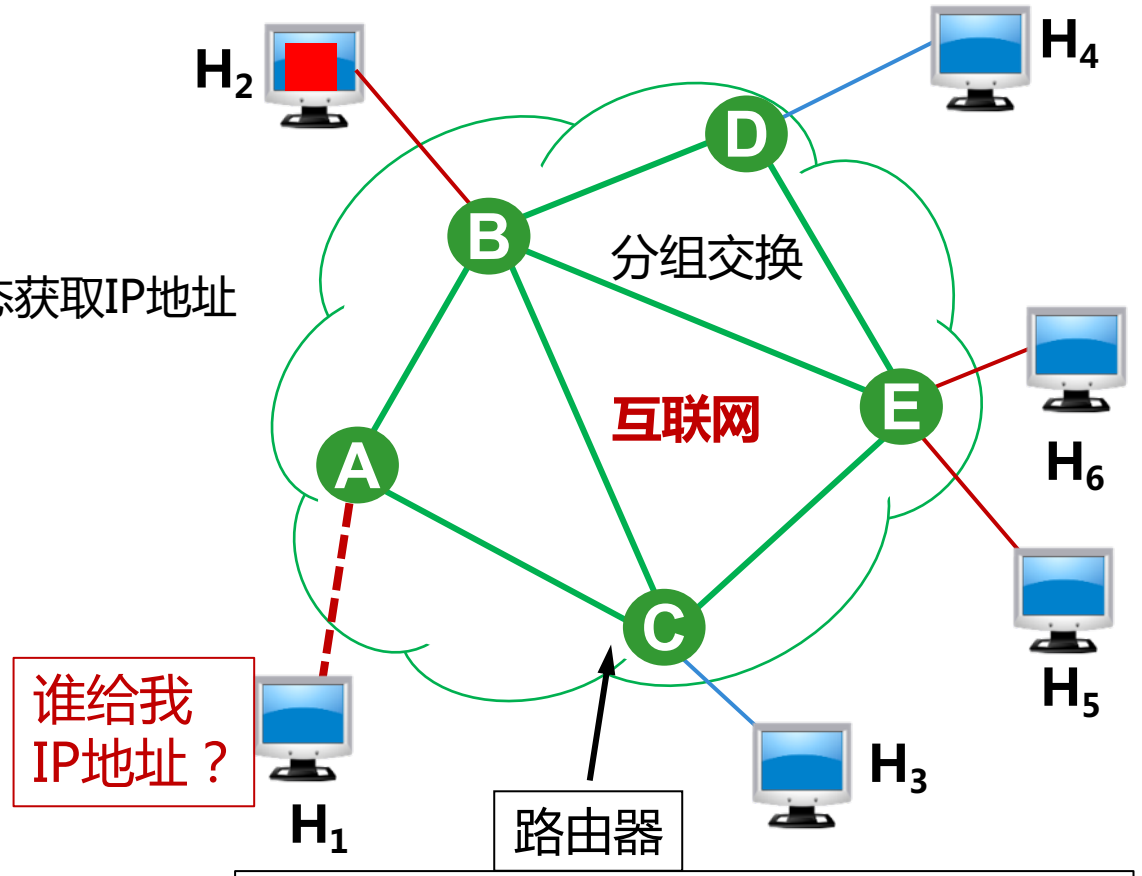


便于使用的需求分解？

IPv4地址如何获取

- 静态设定
  - 申请固定IP地址，手工设定，如路由器、服务器
- 动态获取
  - 使用DHCP协议或其他动态配置协议
  - 当主机加入IP网络，允许主机从DHCP服务器动态获取IP地址
  - 可以有效利用IP地址，方便移动主机的地址获取

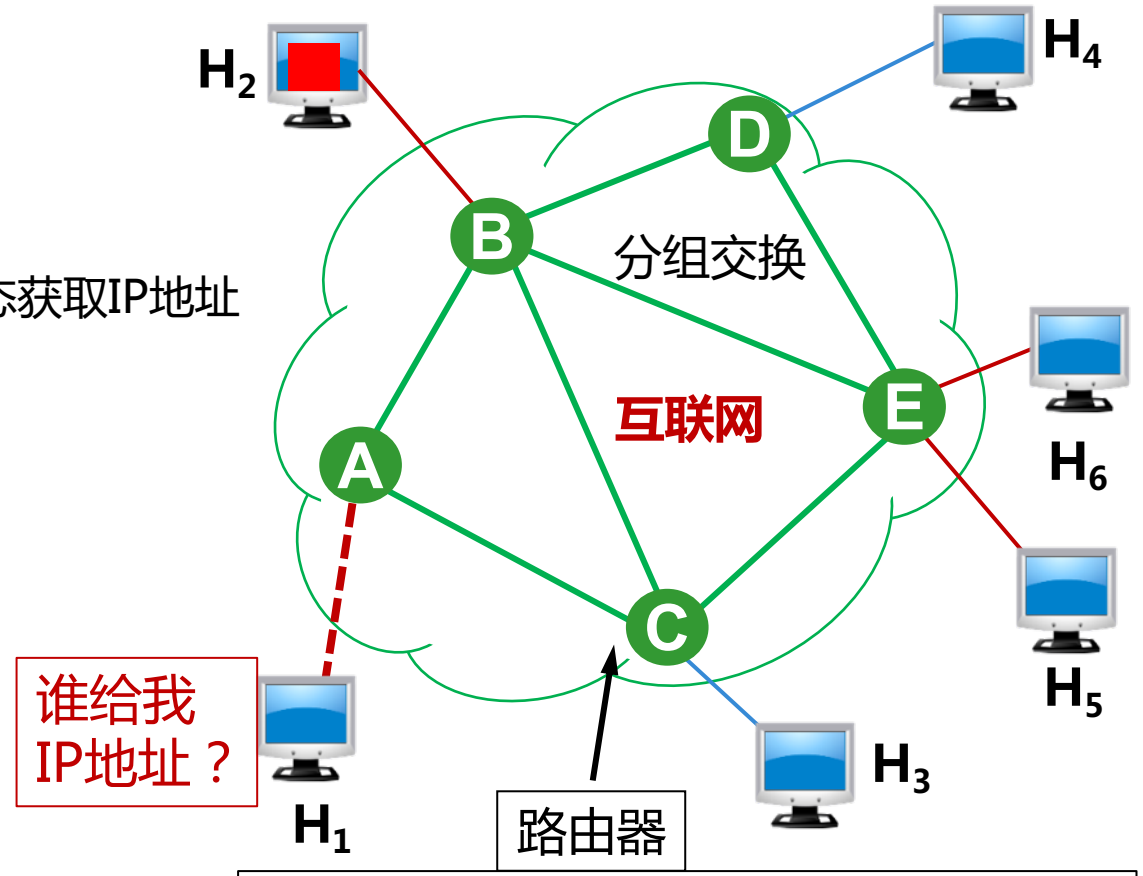





```
# interface FastEthernet 0/0”
# ip address 166.111.8.9 255.255.255.252
# no shutdown
```

IPv4地址如何获取

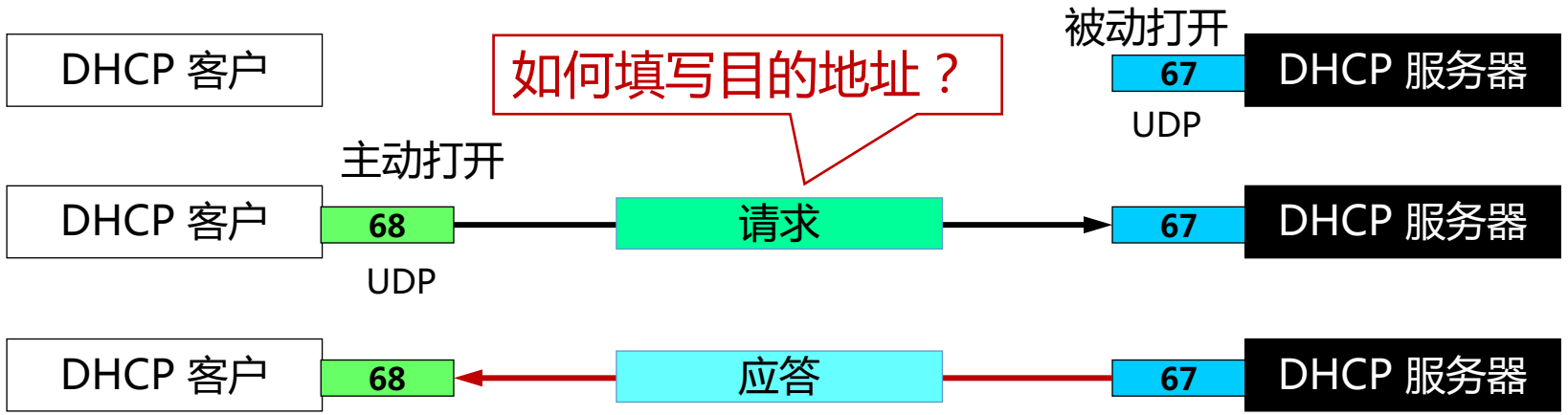
- 静态设定
  - 申请固定IP地址，手工设定，如路由器、服务器
- 动态获取
  - 使用DHCP协议或其他动态配置协议
  - 当主机加入IP网络，允许主机从DHCP服务器动态获取IP地址
  - 可以有效利用IP地址，方便移动主机的地址获取




```
# interface FastEthernet 0/0”
# ip address 166.111.8.9 255.255.255.252
# no shutdown
```

DHCP动态主机配置协议

- DHCP ：动态主机配置协议
  - 当主机加入IP网络，允许主机从DHCP服务器动态获取IP地址
  - 可以有效利用IP地址，方便移动主机的地址获取
- 工作模式：客服/服务器模式（ C/S ）
  - 基于 UDP 工作，服务器运行在 67 号端口， 客户端运行在 68 号端口



DHCP 工作过程

DHCP服务器:
223.1.2.5/24

DHCP discover

src : 0.0.0.0, 68
dest.: 255.255.255.255,67 MAC
yiaddr: 0.0.0.0
transaction ID: 654

到达的
客户机

DHCP offer

src: 223.1.2.5, 67
dest: 255.255.255.255, 68 MAC
yiaddr: 223.1.2.4
transaction ID: 654
lifetime: 3600 secs

DHCP request

src: 0.0.0.0, 68
dest:: 255.255.255.255, 67
yiaddr: 223.1.2.4
transaction ID: 655
lifetime: 3600 secs

DHCP ACK

src: 223.1.2.5, 67
dest: 255.255.255.255, 68
yiaddr: 223.1.2.4
transaction ID: 655
lifetime: 3600 secs

DHCP服务不只返回客户机所需的IP地址，还包括:

- 缺省路由器IP地址
- DNS服务器IP地址
- 网络掩码

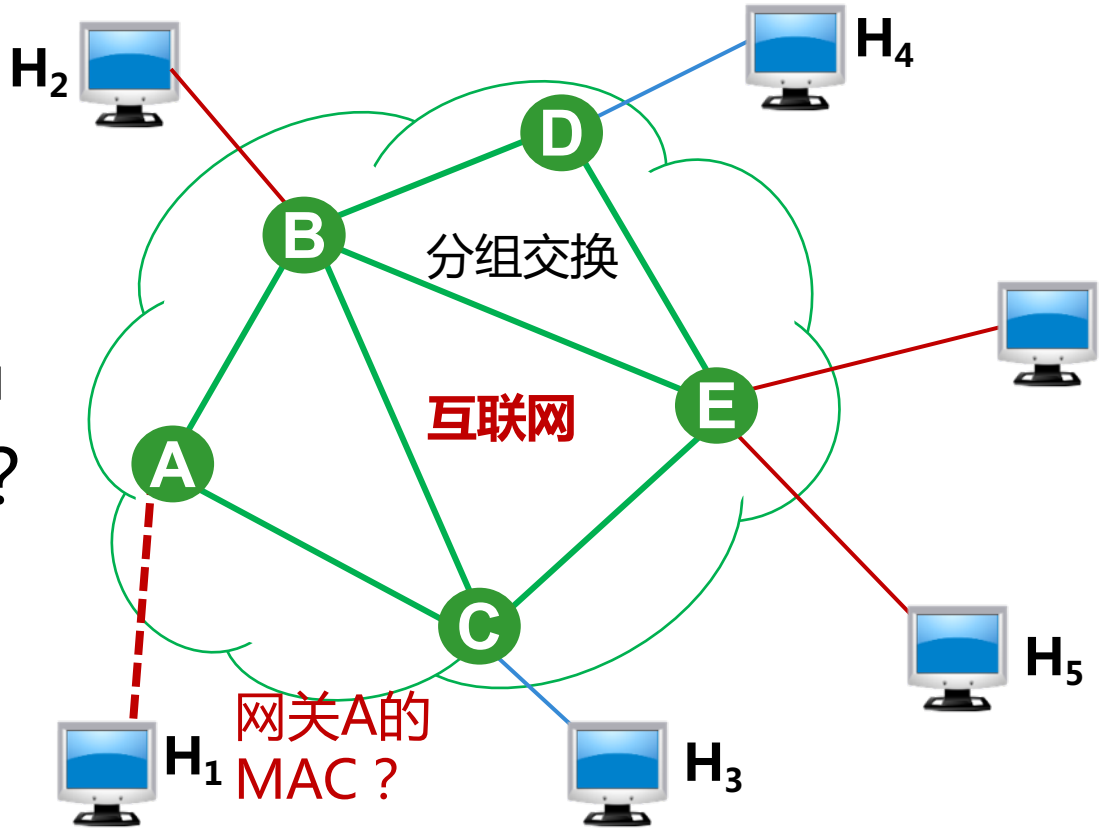
思考
为什么需要四次交互？

DHCP 工作过程

阶段	源MAC	目标MAC	源IP	目标IP	链路层
Discover	PC机的MAC	全FF	0.0.0.0	255.255.255.255	广播
Offer	DHCP服务器（如路由器）的MAC	DHCP客户机的MAC	DHCP服务器（如路由器）的IP地址	255.255.255.255	单播
Request	PC机的MAC	全FF	0.0.0.0	255.255.255.255	广播
Ack	DHCP服务器（如路由器）的MAC	DHCP客户机的MAC	DHCP服务器（如路由器）的IP地址	255.255.255.255	单播

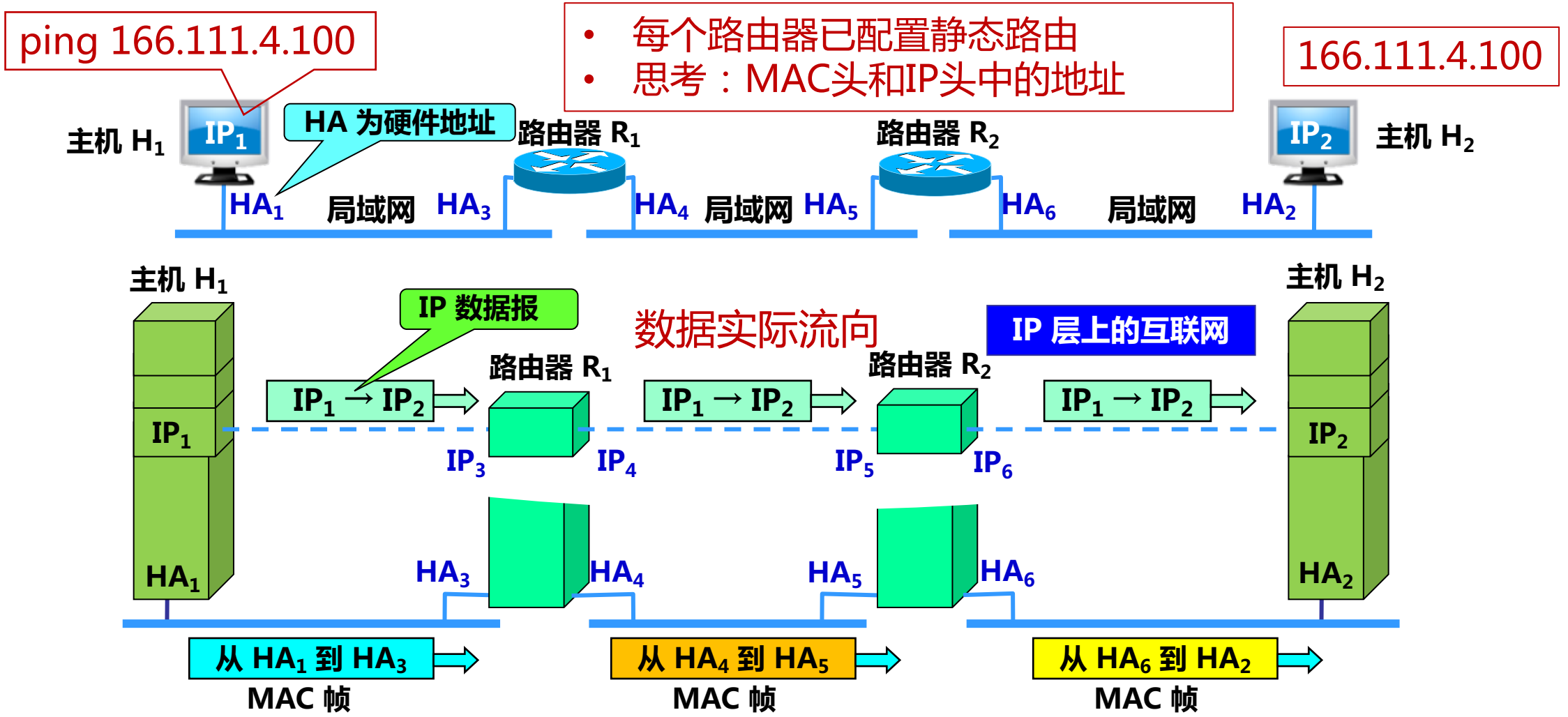
思考与发明

- 如何获取对方MAC地址？
  - 手动绑定映射表？不现实
  - ARP协议自动IP->MAC转换
- 网络出现故障怎么办？
  - 访问www.baidu.com失败，中间有多少设备和链路需要检查？
  - 手动查错不现实
  - 设计协议辅助排查：ICMP

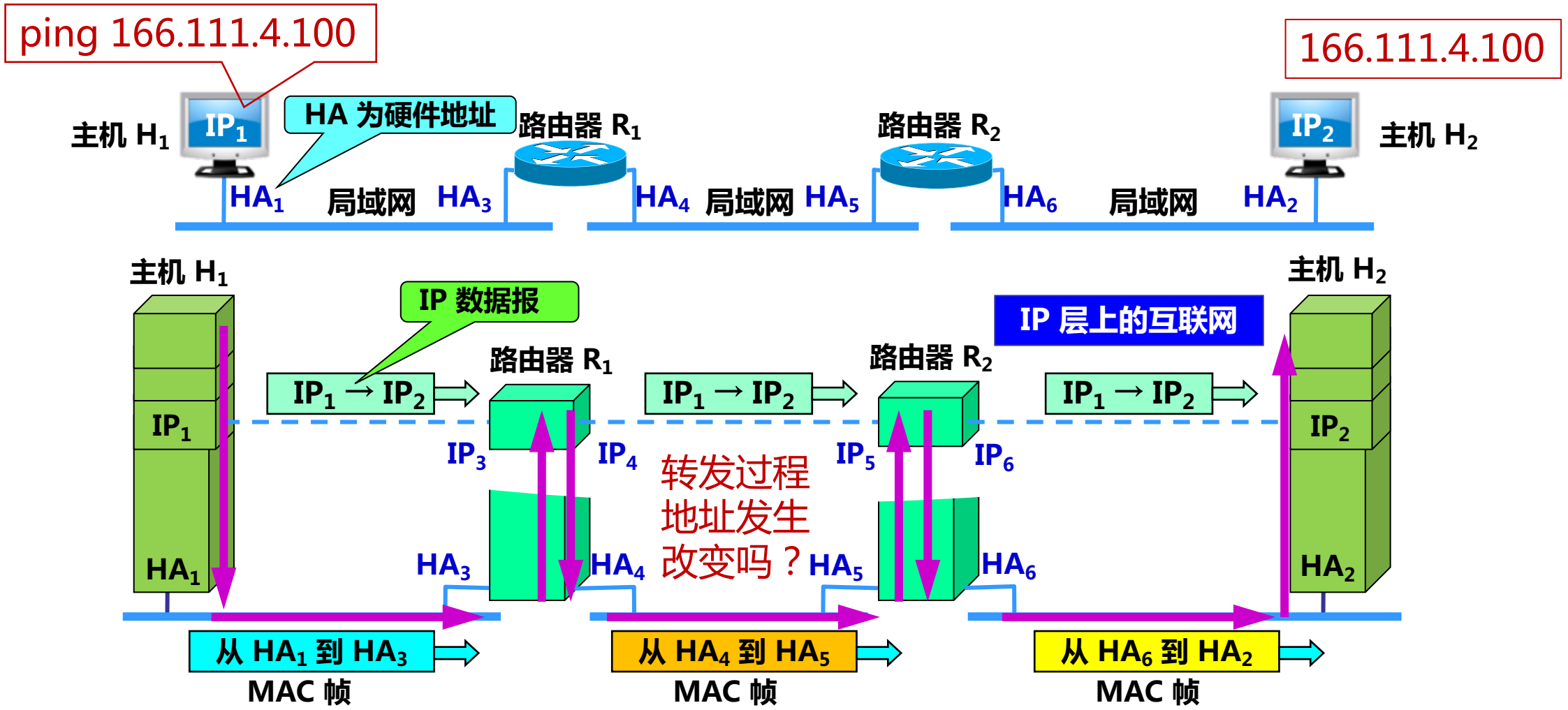


网段IP: 10.1.1.0/24
本机IP：10.1.1.200
MAC:00:20:AF:00:00:01

IP 与 MAC地址


ping 166.111.4.100
• 每个路由器已配置静态路由
• 思考：MAC头和IP头中的地址
166.111.4.100
主机 H1
IP1
HA 为硬件地址
路由器 R1
路由器 R2
IP2
主机 H2
HA1
局域网
HA3
HA4
局域网
HA5
HA6
局域网
HA2
主机 H1
IP 数据报
数据实际流向
IP 层上的互联网
路由器 R1
路由器 R2
主机 H2
IP1 → IP2
IP1 → IP2
IP1 → IP2
IP1
IP2
IP3
IP4
IP5
IP6
HA1
HA3
HA4
HA5
HA6
HA2
从 HA1 到 HA3
从 HA4 到 HA5
从 HA6 到 HA2
MAC 帧
MAC 帧
MAC 帧

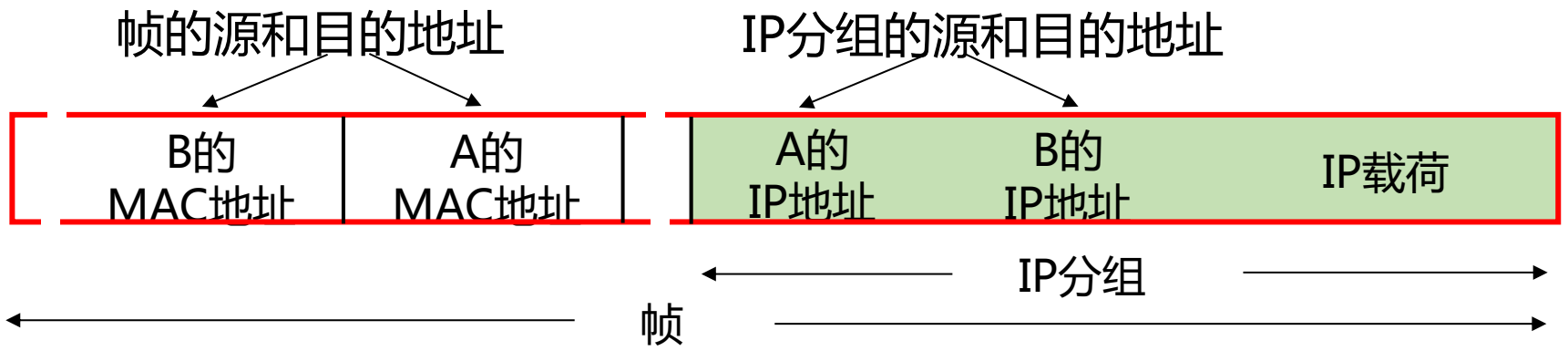
IP 与 MAC地址


ping 166.111.4.100
166.111.4.100
主机 H1
IP1
HA 为硬件地址
路由器 R1
路由器 R2
IP2
主机 H2
HA1
局域网
HA3
HA4
局域网
HA5
HA6
局域网
HA2
主机 H1
IP 数据报
路由器 R1
路由器 R2
IP 层上的互联网
主机 H2
IP1 → IP2
IP1 → IP2
IP1 → IP2
IP1
IP3
IP4
IP5
IP6
IP2
转发过程
地址发生
改变吗？
HA1
HA3
HA4
HA5
HA6
HA2
从 HA1 到 HA3
从 HA4 到 HA5
从 HA6 到 HA2
MAC 帧
MAC 帧
MAC 帧

ARP地址解析协议

ping 166.111.4.100

166.111.4.100

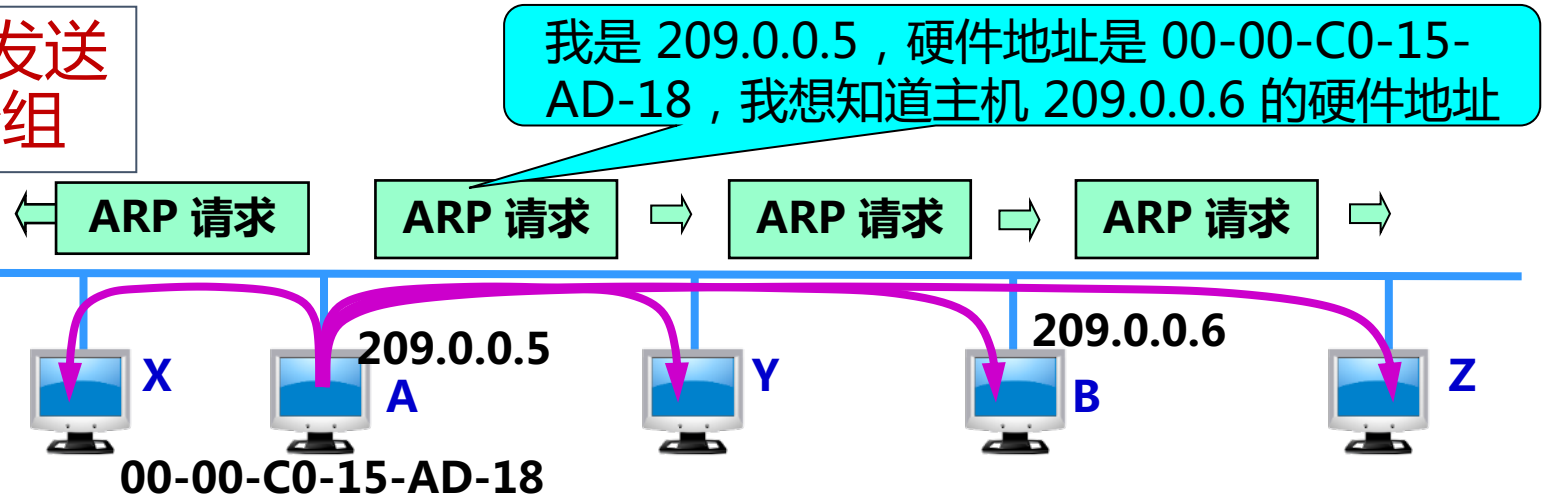


如何获取下一跳的MAC地址(已知IP地址)？

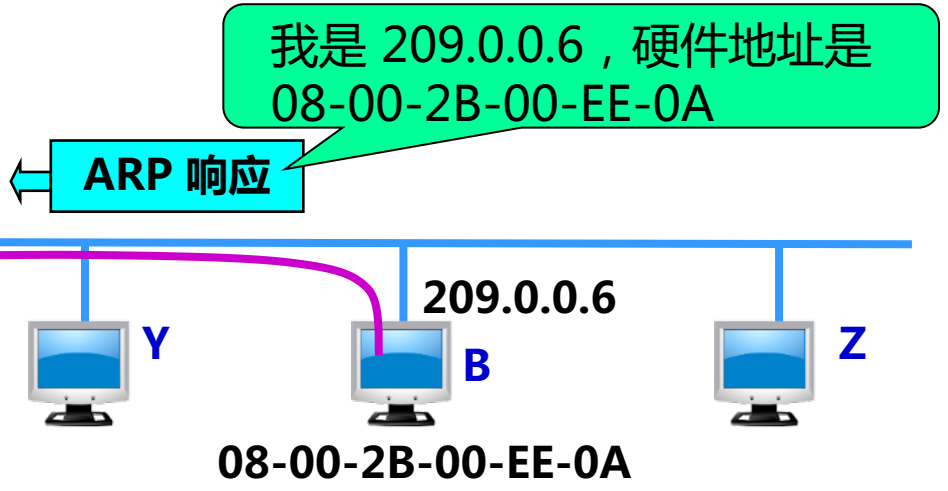
设计ARP协议？

ARP协议工作过程

主机 A 广播发送 ARP 请求分组



主机B向A单播发送 ARP 响应分组



有安全隐患吗？

ARP地址解析协议

- 在ARP表中缓存IP地址和MAC地址的映射关系（即ARP高速缓存）

```
dgdeMacBook-Pro:~ yongcui$ arp -all
Neighbor                Linklayer Address  Expire(0)
192.168.3.1             90:17:c8:0:99:99   2m18s
192.168.3.255           ff:ff:ff:ff:ff:ff  (none)
224.0.0.251             1:0:5e:0:0:fb      (none)
239.255.255.250         1:0:5e:7f:ff:fa    (none)
```

何时查询ARP表？不命中怎么办？

思考：添加、删除的优化策略？

单选题 1分

设置

ARP工作过程中，ARP请求(Request)在链路层采用（ ）发送，ARP确认(Ack)在链路层采用（ ）发送。

A 单播 组播

B 组播 单播

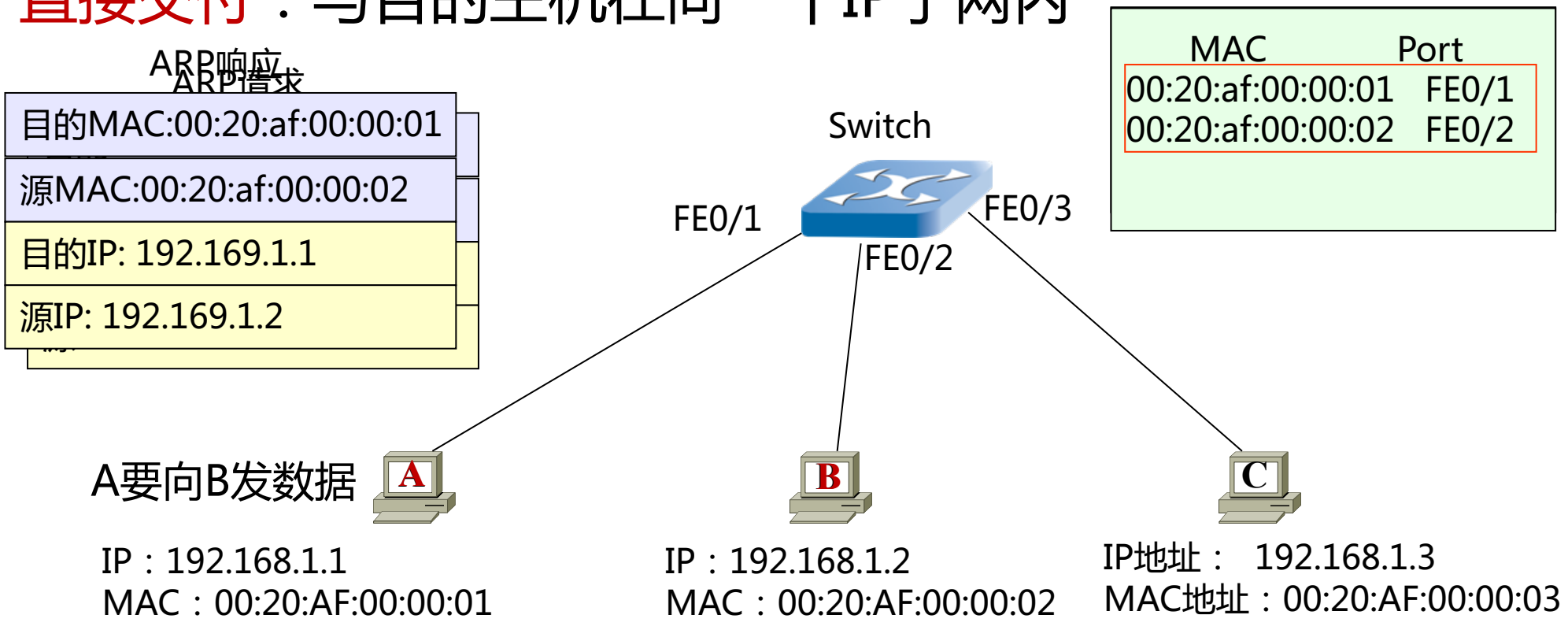
C 广播 单播

D 选播 组播

提交

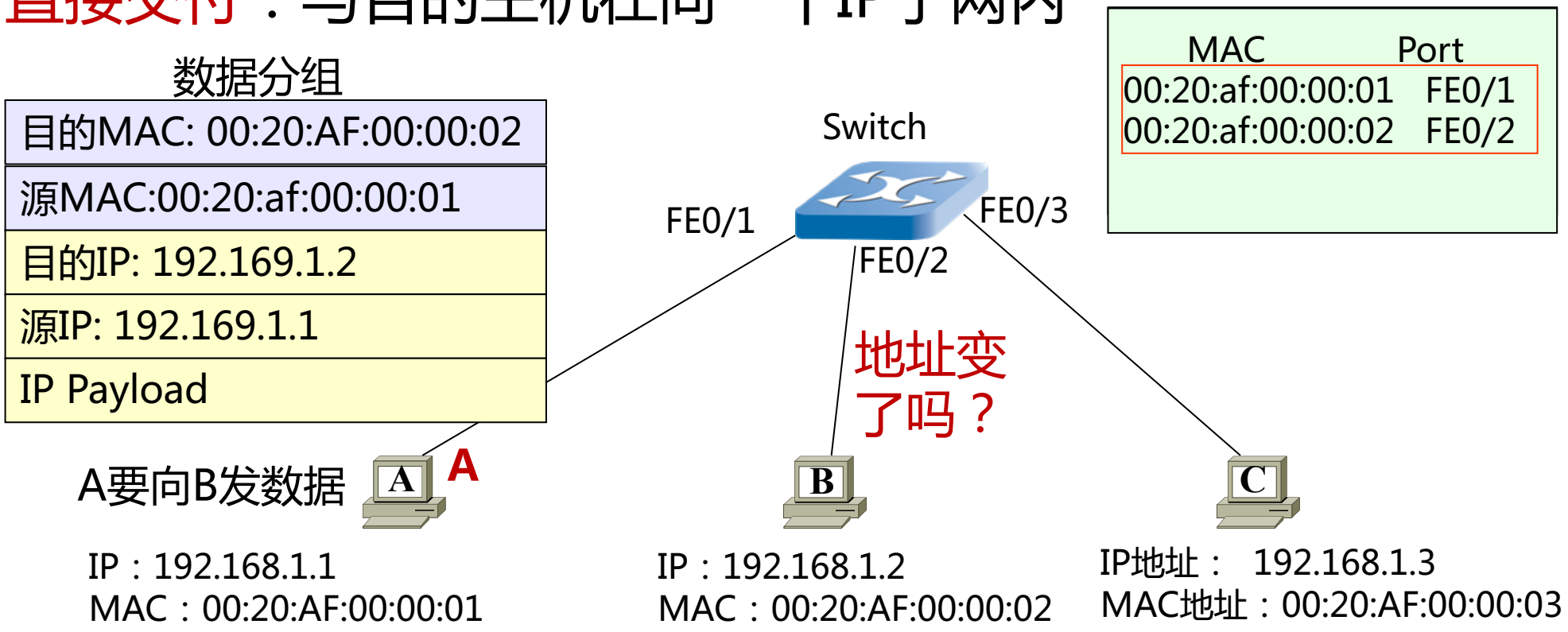
IP包转发

- 如何获取填写目的MAC地址？
- 直接交付：与目的主机在同一个IP子网内



IP包转发

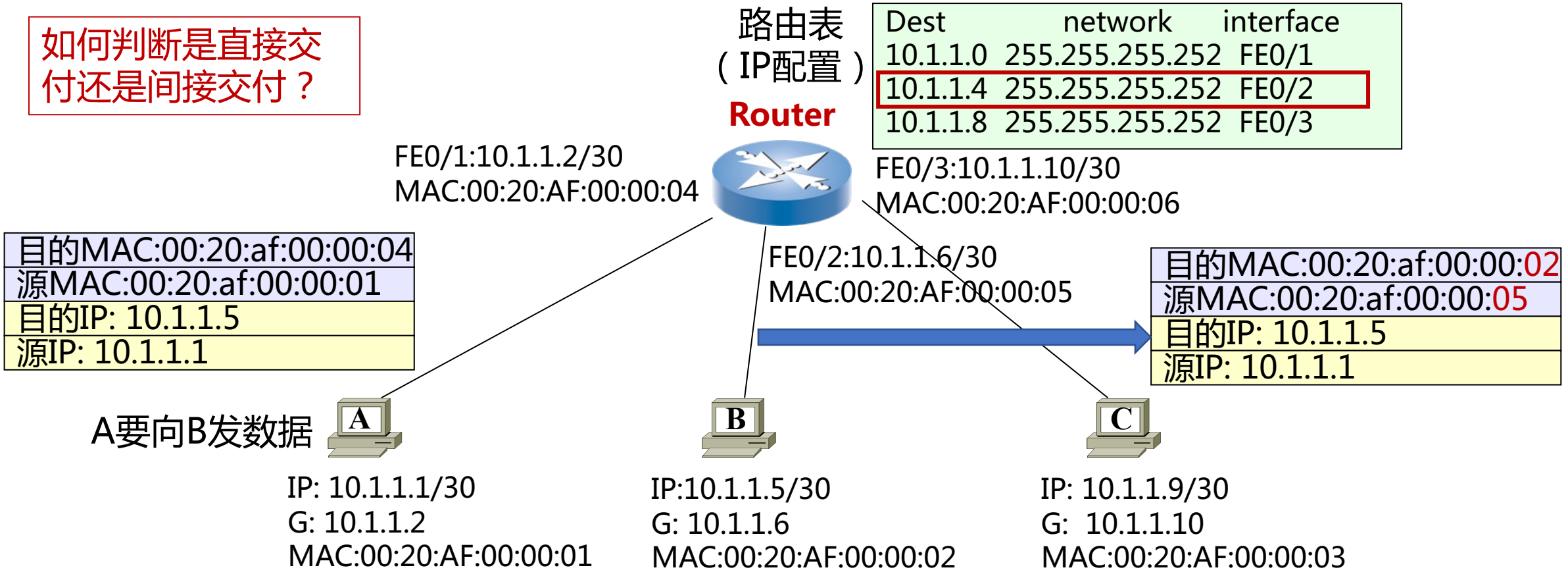
- 如何获取填写目的MAC地址？
- 直接交付：与目的主机在同一个IP子网内



直接交付如何填写目的地址？

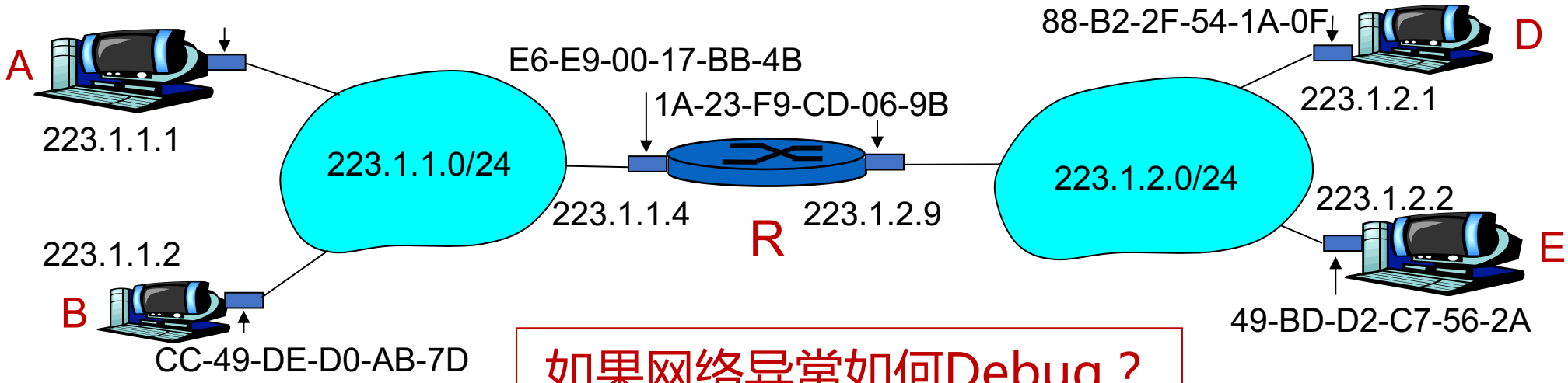
IP包转发

- 间接交付：与目的主机不在同一个IP子网内



路由到另一个局域网

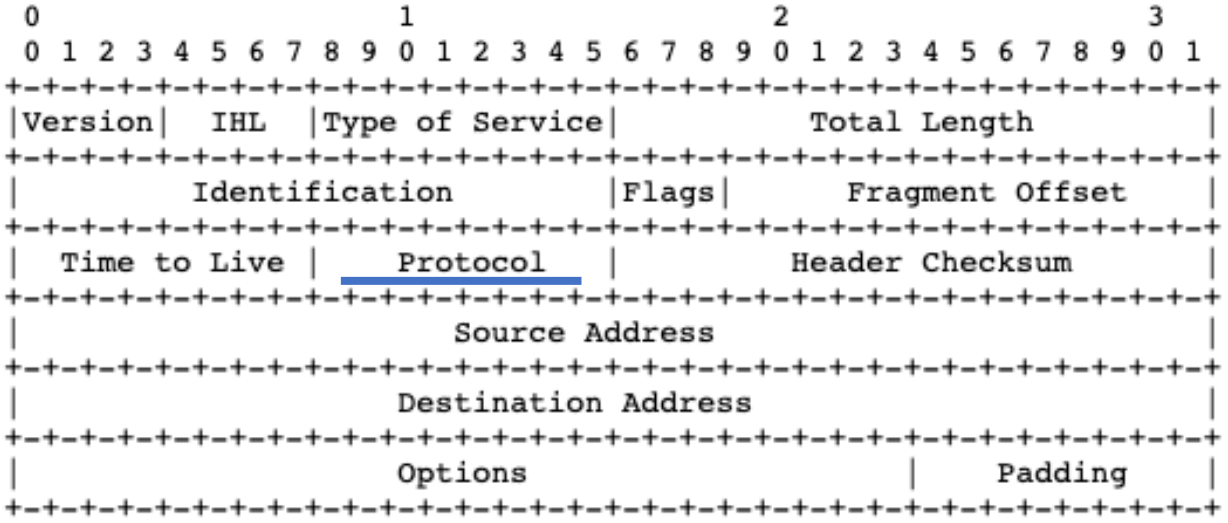
1. 主机A接入网络（希望发数据给E）
2. 申请IP地址：使用DHCP获取动态IP地址
3. A创建IP数据包：源为A、目的为E
4. 主机A查找路由表：找到路由器R的IP地址223.1.1.4
5. A获得R的MAC地址：根据R的IP地址223.1.1.4，使用ARP协议获得MAC地址
6. A创建数据帧：目的地址为R的MAC地址
7. A封装数据帧：封装A到E的IP数据包
8. A发送数据帧：A发送后，R接收数据帧



如果网络异常如何Debug？

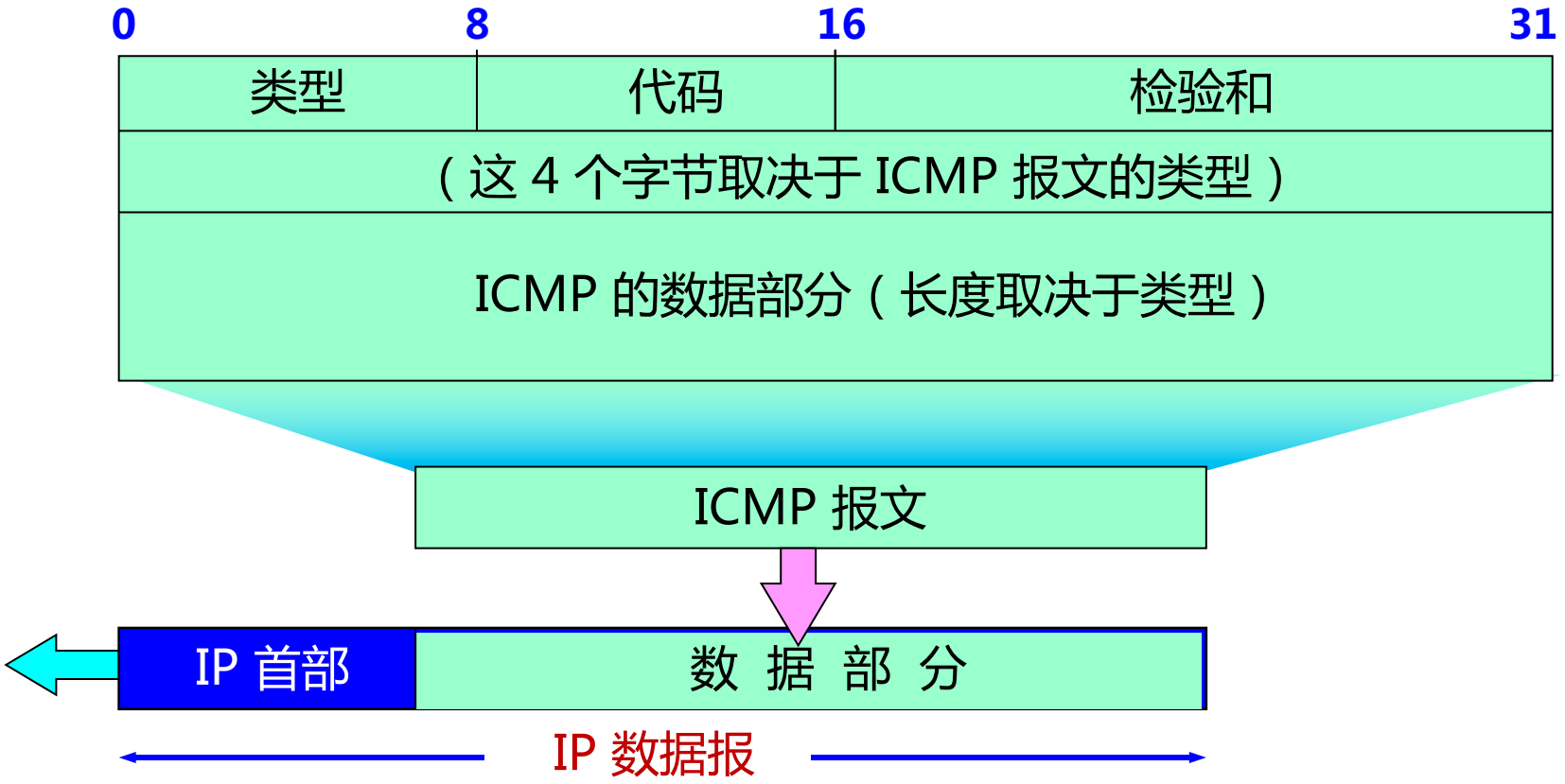
ICMP协议

- ICMP: 互联网控制报文协议
  - ICMP 允许主机或路由器报告差错情况和提供有关异常情况的报告
  - 由主机和路由器用于网络层信息的通信
  - ICMP 报文携带在IP 数据报中： IP上层协议号为1
- ICMP报文类型
  - ICMP 差错报告报文
    - 终点不可达：不可达主机、不可达网络，无效端口、协议
  - ICMP 询问报文
    - 回送请求/回答 (ping使用)



IP头的上层协议号为1

ICMP 报文格式



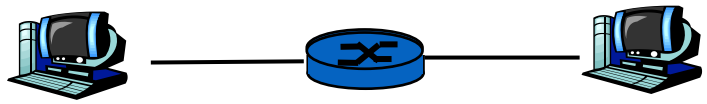
- ICMP报文的前 4 个字节包含格式统一的三个字段：类型、代码、检验和

ICMP报文类型及功能

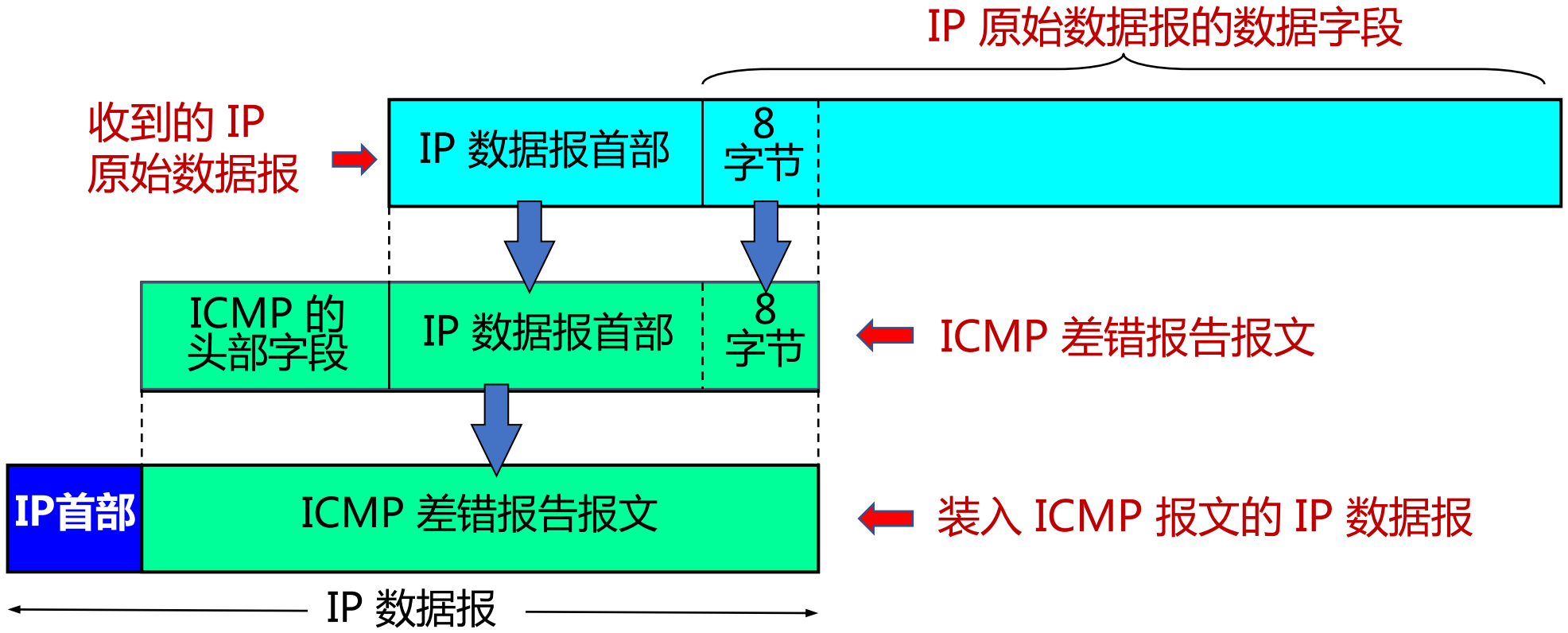
ICMP报文类型	类型值	功能描述
差错报告报文	3	终点不可达
	5	改变路由(Redirect)
	11	时间超时
	12	参数问题
询问报文	8或0	回送(Echo)请求或应答
	13或14	时间戳(Timestamp)请求或回答

类型	代码	功能描述
0	0	回送应答 (ping)
3	0	目的网络不可达
3	1	目的主机不可达
3	2	目的协议不可达
3	3	目的端口不可达
3	6	目的网络未知
3	7	目的主机未知
8	0	回送请求 (ping)
9	0	路由通告
10	0	路由发现
11	0	TTL过期
12	0	坏的IP首部

差错报告报文



ICMP报错，但是什么错误、哪个报文的错误呢？



Ping和ICMP

- PING (Packet InterNet Groper)
  - PING 用来测试两个主机之间的连通性
  - PING 使用了 ICMP 回送请求与回送回答报文

```
C:\>ping www.baidu.com

正在 Ping www.a.shifen.com [110.242.68.4] 具有 32 字节的数据:
来自 110.242.68.4 的回复: 字节=32 时间=32ms TTL=53
来自 110.242.68.4 的回复: 字节=32 时间=29ms TTL=53
来自 110.242.68.4 的回复: 字节=32 时间=29ms TTL=53
来自 110.242.68.4 的回复: 字节=32 时间=31ms TTL=53

110.242.68.4 的 Ping 统计信息:
    数据包: 已发送 = 4, 已接收 = 4, 丢失 = 0 (0% 丢失),
往返行程的估计时间(以毫秒为单位):
    最短 = 29ms, 最长 = 32ms, 平均 = 30ms
```

连通性　　往返时延　　单向转发跳数

思考：

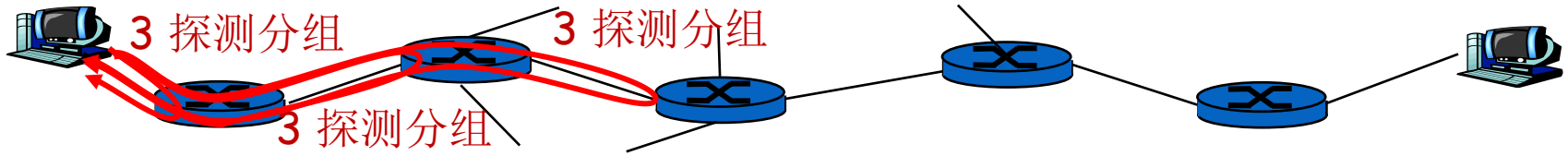
如何利用Ping命令返回的TTL值(报文剩余跳数)，来判断对方主机操作系统的类型？

默认操作系统的TTL值：

1、WINDOWS NT/2000　TTL：128
2、WINDOWS 95/98　TTL：32
3、UNIX　TTL：255
4、LINUX　TTL：64
5、WIN7　TTL：64

Traceroute和ICMP

- 如何知道整个路径上路由器的地址？使用TraceRT/Traceroute命令



- 源向目的地发送一系列UDP段
  - 第一个 TTL=1
  - 第二个 TTL=2
  - ……
- 当第n个数据报到达第n个路由器
  - 路由器丢弃数据报
  - 向源发送ICMP报文 (类型 11, TTL过期)
  - 报文的源IP地址是该路由器的IP地址

Traceroute和ICMP

C:\>tracert www.taobao.com

通过最多 30 个跃点跟踪

到 [27.211.197.171] 的路由:

1	3 ms	2 ms	4 ms	192.168.3.1
2	3 ms	6 ms	3 ms	SMBSHARE [192.168.1.1]
3	6 ms	9 ms	5 ms	27.215.136.1
4	6 ms	11 ms	7 ms	61.162.199.89
5	7 ms	18 ms	21 ms	61.162.199.9
6	23 ms	29 ms	22 ms	61.156.223.69
7	28 ms	31 ms	20 ms	112.230.160.54
8	19 ms	27 ms	36 ms	60.208.64.230
9	15 ms	15 ms	16 ms	119.164.254.86
10	19 ms	13 ms	13 ms	27.211.197.171

- 当源收到ICMP报文，计算RTT
- Tracert针对同一RTT值执行上述过程3次

停止条件

- UDP段最终到达目的地主机
- 目的地返回ICMP 分组
- 当源得到该ICMP，停止

路由器会偷懒吗？

网络层典型协议和技术-小结

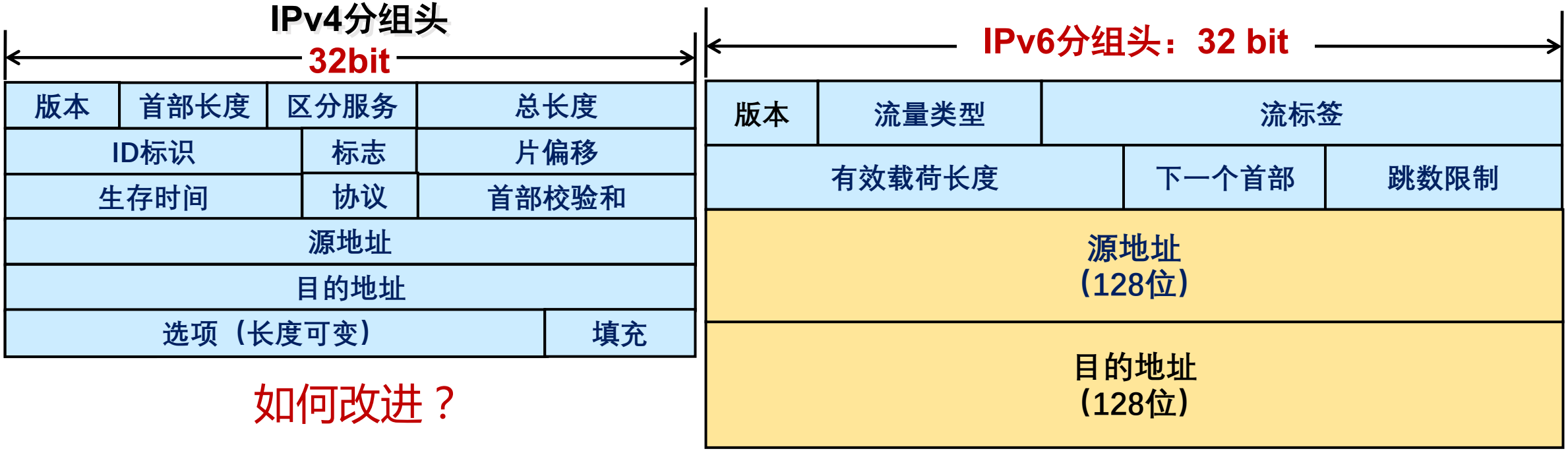
- DHCP协议
  - 动态主机配置协议，C/S模式动态分配IP地址（广播 | 单播）
- ARP协议
  - ARP：给定IP地址，如何获得对应的MAC？
  - 多跳网络中，IP 数据报中IP地址不改变，Mac帧中的硬件地址逐跳改变
  - 直接交付和间接交付的区别？
- ICMP协议
  - IP层报告差错情况
  - 两种检错工具：Ping、Traceroute

家用路由器有这些功能吗？

IPv6协议

- IPv6（Internet Protocol version 6）
  - 千年大计：32-bit地址空间耗尽
  - CIDR和NAT都无法从根本上解决地址短缺问题
  - 互联网工程任务组（IETF）设计的用于替代IPv4的下一代协议
- IPv6 地址
  - 地址长度为128bit，是IPv4地址长度的4倍
  - IPv6地址空间数量约为$3*10^{38}$
  - IPv6地址表示法，冒分十六进制，x:x:x:x:x:x:x:x
    - 简化方法：每个x前面的0可省略，可把连续的值为0的x表示为“::”，且“::”只能出现1次
    - 简化前地址，2001:0DA8:0000:0000:200C:0000:0000:00A5
    - 简化后地址，2001:DA8:0000:0000:200C::A5

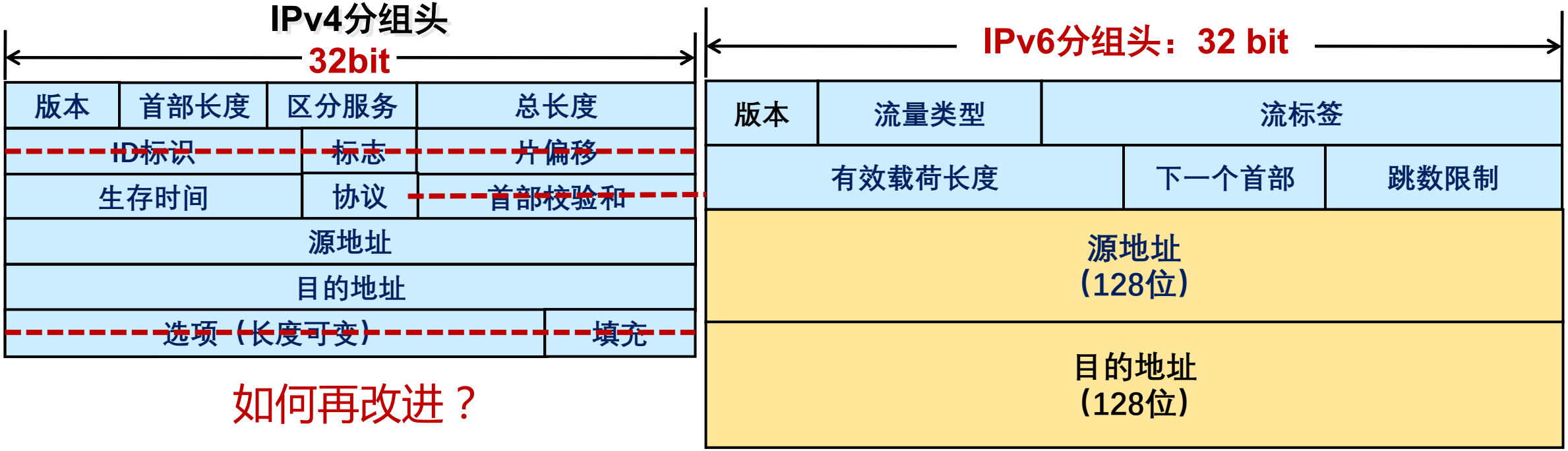
IPv6头部



如何改进？

- 版本：4bit，协议版本号，值为6
- 流量类型：8bit，区分数据包的服务类别或优先级
- 流标签：20bit，标识同一个数据流
- 有效载荷长度：16bit ，IPv6报头之后载荷的字节数（含扩展头），最大值64K

IPv6头部的优化



如何再改进？

- 下一个首部：8bit ，下一个协议类型，如TCP/UDP/ICMP等，也可是扩展头
- 跳数限制：8bit ，类似IPv4的TTL，每次转发跳数减1，值为0时包将会被丢弃
- 源地址： 128bit ，标识该报文的源地址
- 目的地址： 128bit ，标识该报文的目的地址

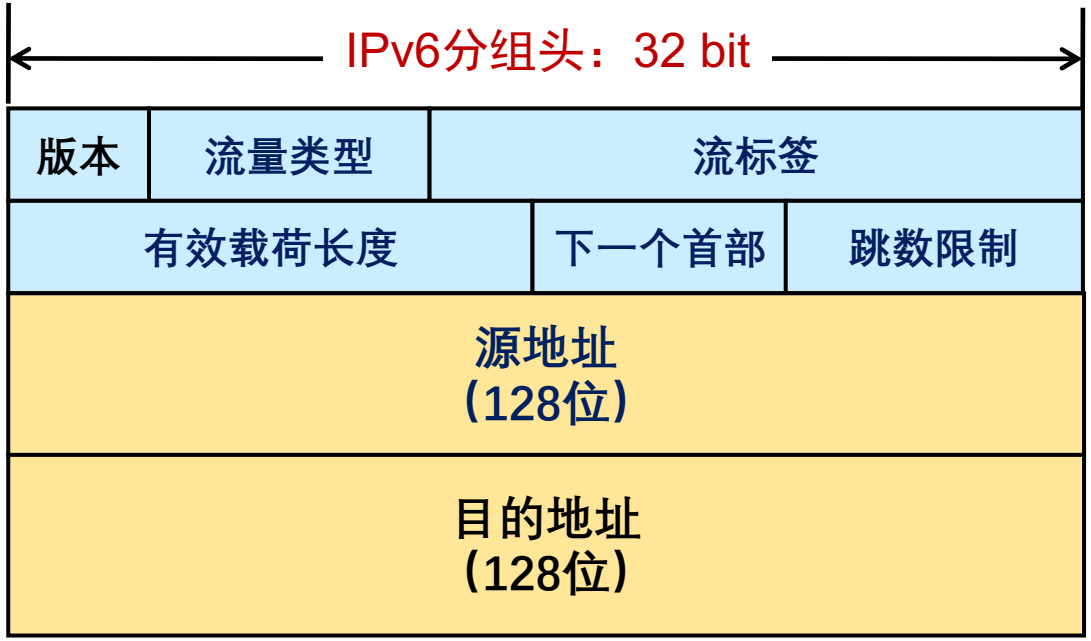
IPv6头部字段分析

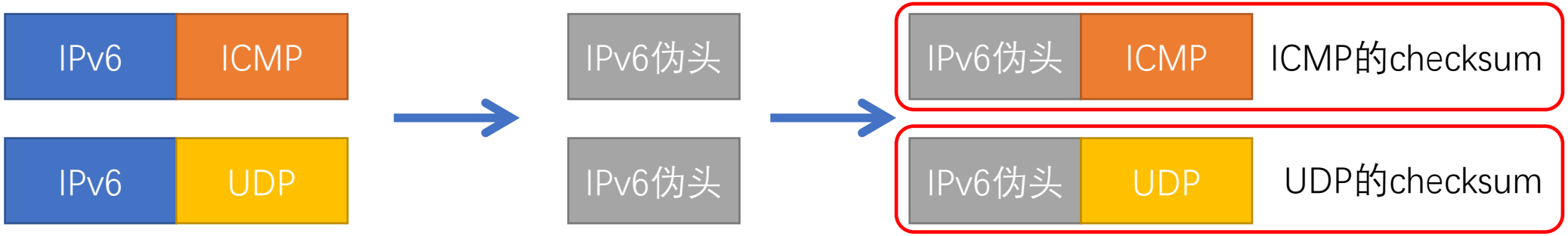
如何在IPv4基础上提升分组转发处理速度？

- IPv6头部长度固定40字节，所有“选项”字段都在IPv6扩展头部分
- IPv6分片机制
  - IPv6分组不能在传输途中分片，只在源端进行分片
  - IPv6支持Path MTU发现机制
  - 去除分片字段：“标识” “标志” “片偏移” ，移至扩展头（分段头）
- 与IPv4头部的比较
  - 去除“首部长度”（首部长度固定为40字节）
  - 去除“首部校验和” ，提升转发速度

路由器转发的校验和优化

- IPv6基本报头无需逐跳校验
  - IPv6路由器转发时不进行逐跳校验
  - 数据链路层和传输层（如TCP/UDP校验和）已提供校验机制，网络层校验冗余
- 网络层与上层合作进行校验和检查
  - 跳数限制（Hop Limit）递减无需校验
  - 需构造 IPv6 Pseudo Header（伪头）
- 实验实现：ICMPv6和UDP下的checksum

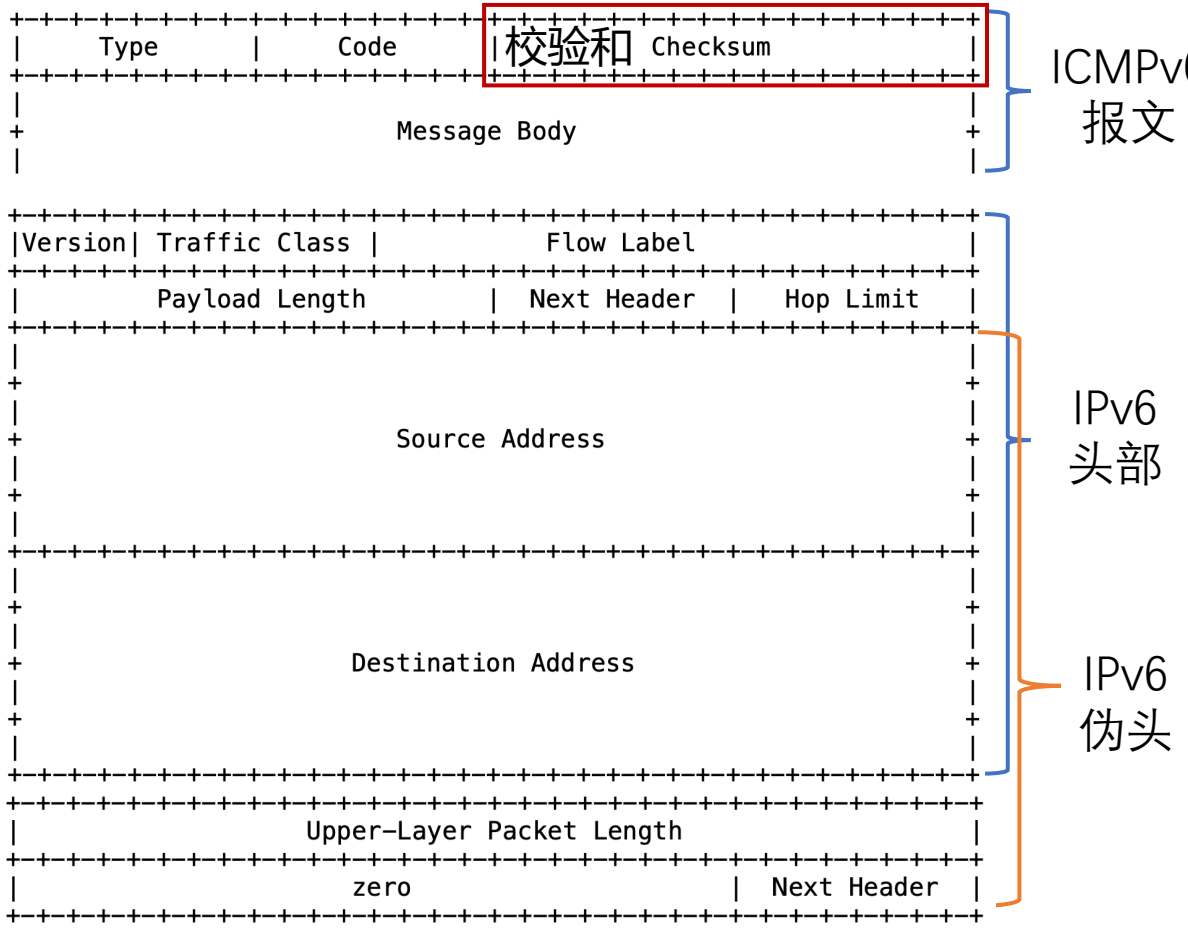




举例：ICMPv6校验和的计算

- 如何计算 ICMPv6 的校验和
  1. 构造IPv6伪头：
     - 填入源、目的地址
     - 计算 ICMPv6 报文长度（不含伪头）
     - Next Header填58
  2. 构造 ICMPv6 报文
     - Checksum字段临时置零
     - 保留其他字段
  3. 拼接伪头 + ICMPv6 报文
  4. 计算并填充校验和
     - 拼接后的数据每16位补码求和取反

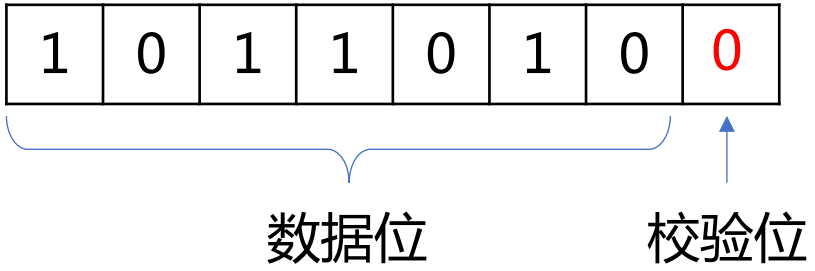
参考资料：RFC 4443 2.3; RFC 8200 8.1



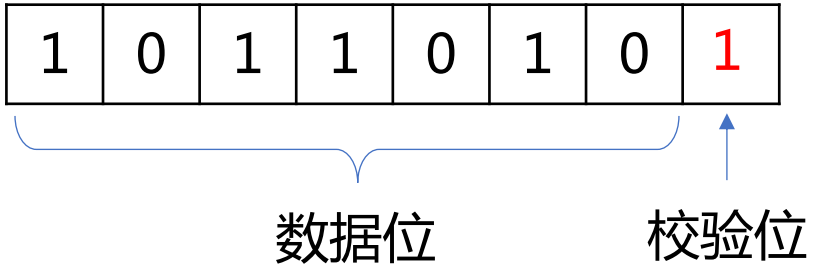
ICMPv6报文、IPv6头部及伪头结构示意图

典型检错码—奇偶校验

- 1位奇偶校验：增加1位校验位，可以检查奇数位错误
  - 偶校验：保证1的个数为偶数个



  - 奇校验：保证1的个数为奇数个



1	0	1	1	0	1	0	0
1	0	1	0	0	1	0	1
1	0	1	1	0	1	0	0
1	0	1	0	0	1	0	1

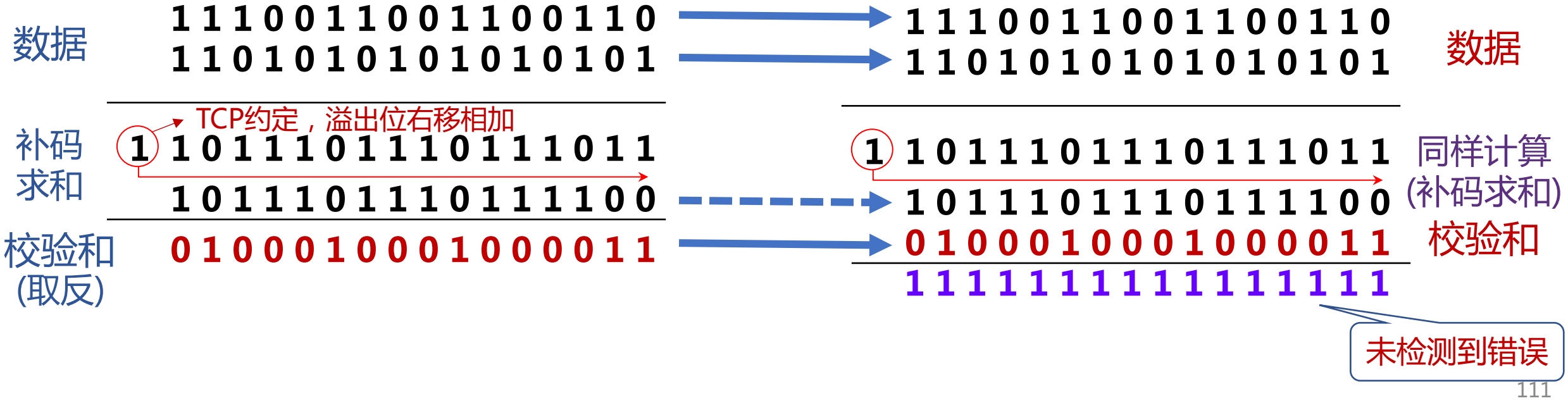
会有什么问题？
连续突发错误？

典型检错码—校验和

- 以TCP/IP体系中主要采用的校验方法为例

发送方：进行 16 位二进制补码求和运算，计算结果取反，随数据一同发送

接收方：进行 16 位二进制补码求和运算（包含校验和）。若结果非全1，则检测到错误



IPv6扩展头

- IPv6报文可承载多个扩展头
- 每个扩展头都包含“下一个首部”字段（IPv6首部固定字段也有）
  - 可指向下一个扩展头类型
  - 或指明传统上层协议类型（最后一个扩展头）：TCP/UDP/ICMP . . .
- 如有多个扩展头，需按规定顺序出现
  - 逐跳选项头，转发路径上每个节点都需检查该扩展头的信息
  - 路由头，指明转发途中需经过哪些节点，类似于IPv4的源路由机制
  - 分段头，包含类似IPv4分片处理信息：片偏移、“更多段”标志、标识符
  - 目的地选项头，目的端系统需要确认的信息
  - . . .

IPv6地址及配置

- IPv6地址分类
  - 未指定地址（::/128），不能分配给任何节点
  - 回环地址（::1/128），表示节点自己，不分配，类似IPv4中的127.0.0.1
  - 组播地址（FF00::/8）
  - 链路本地地址（FE80::/10），也称为Link-local地址，仅在本地链路上使用，网络设备根据接口MAC地址自动生成
  - 全局单播地址，其它地址
- IPv6地址配置方式
  - 手动配置
  - DHCPv6（IPv6动态主机配置协议）
  - 无状态地址自动配置，基于ND协议的RS报文的IPv6前缀信息，结合自己的链路层地址生成IPv6地址

邻居发现ND协议

- 邻居发现协议（Neighbor Discovery Protocol，ND）
  - 邻居发现基于ICMPv6实现，不同的Type值和Code值表示不同的ND消息
- 消息类型1：邻居请求(Neighbor Solicitation，NS)
  - 类似于IPv4中的ARP请求报文，获取邻居的链路层地址，验证邻居可达，重复地址检测
- 消息类型2 ：邻居通告(Neighbor Adivertisment，NA)
  - 类似于IPv4中的ARP应答报文，对NS消息进行响应
- 消息类型3 ：路由器请求(Router Solicitation，RS)
  - 端系统通过RS消息向路由器发出请求，请求地址前缀和其他信息，用于节点的自动配置
- 消息类型4 ：路由器通告(Router Advertisement，RA)
  - 路由器通过RA消息向端系统发布地址前缀（IPv6地址自动配置）和其他配置信息
- 消息类型5 ：重定向(Redirect)
  - 通知主机重新选择正确的下一跳地址（针对某个目的IPv6地址）

IPv6路由协议

- RIPng for IPv6，RFC 2080，对RIP修改以适应IPv6环境
  - 使用路由器的链路本地IPv6地址作为源地址，发送路由更新信息
- OSPFv3 for IPv6，RFC 5340，适应IPv6网络
  - 使用路由器的链路本地IPv6地址作为源地址，并作为下一跳地址
  - OSPFv3有7种类型的LSA，新增Link LSA和Intra Area Prefix LSA
- MP-BGP(Multi-Protocol BGP)，RFC 4760，支持多种网络层协议（IPv6和IPX）
  - MP-BGP向前兼容，并支持组播
  - BGP连接可以是IPv4或IPv6，报文内可传递其它网络协议的路由信息
  - 多协议可达NLRI描述了到达目的地的信息：地址属于哪个网络层协议，下一跳地址

IPv4 -> IPv6

网络层基础：IPv4
动态地址分配：DHCP
单跳网络处理：ARP
基础管理和控制：ICMP
距离向量路由：RIP
链路状态路由：OSPF
外部网关路由：BGP

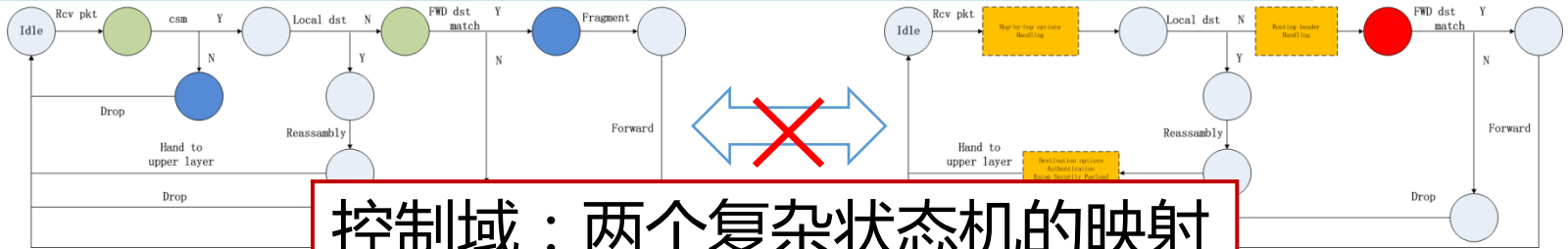
网络层基础：IPv6
动态地址分配：DHCPv6
单跳网络处理：ND
基础管理和控制：ICMPv6
距离向量路由：RIPng
链路状态路由：OSPFv3
外部网关路由：MP-BGP

- 双协议栈技术
  - 每个设备同时运行IPv4和IPv6两个协议栈
  - 无法解决IPv4地址不足的问题，不能促进IPv6部署

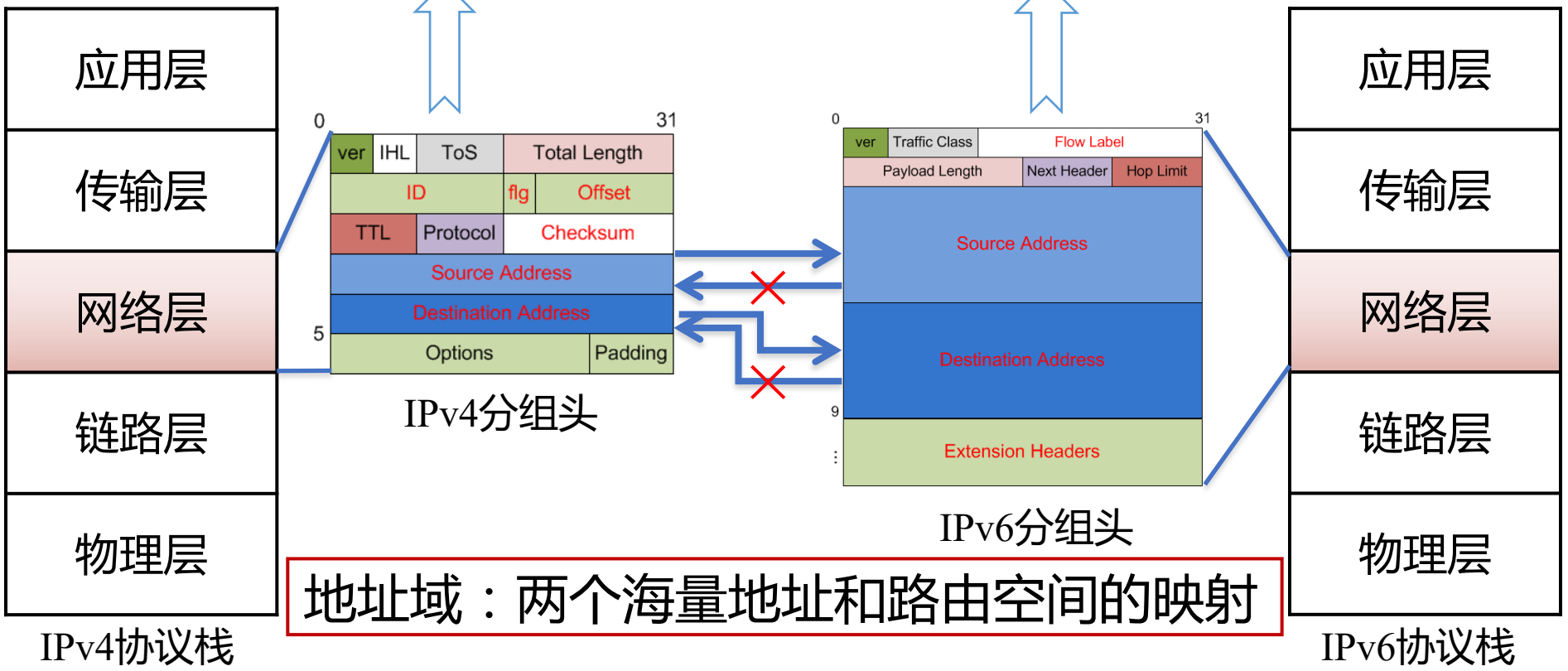
目标：保证端到端透明原则
（用户无感知）

- IPv4和IPv6技术并不能兼容，怎么办？
  - 完全抛弃IPv4，建立新的IPv6网络？
  - 设计过渡技术，同步发展，逐步切换到IPv6网络？更合理的选择

IPv4到IPv6过渡技术难点



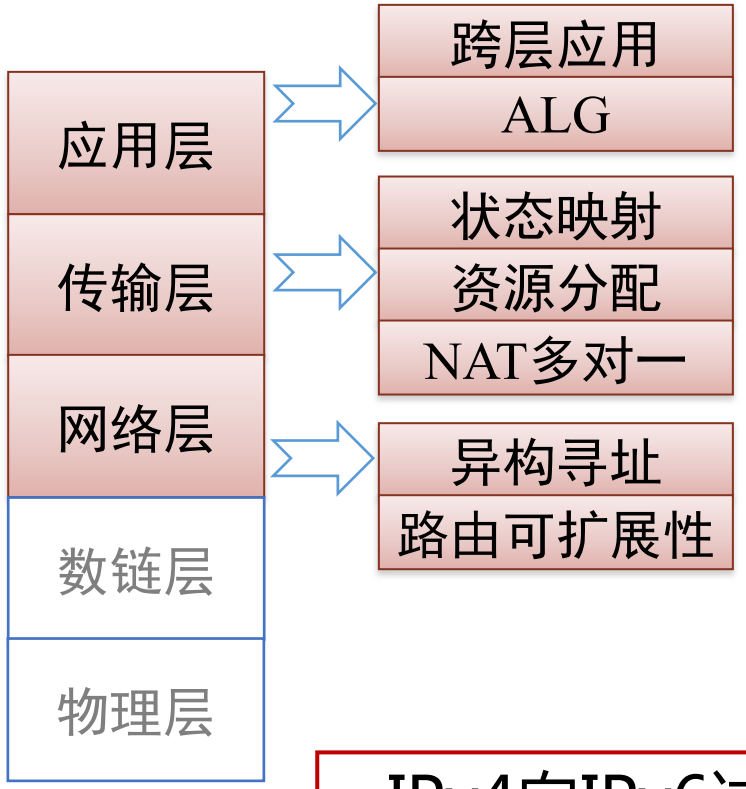
控制域：两个复杂状态机的映射



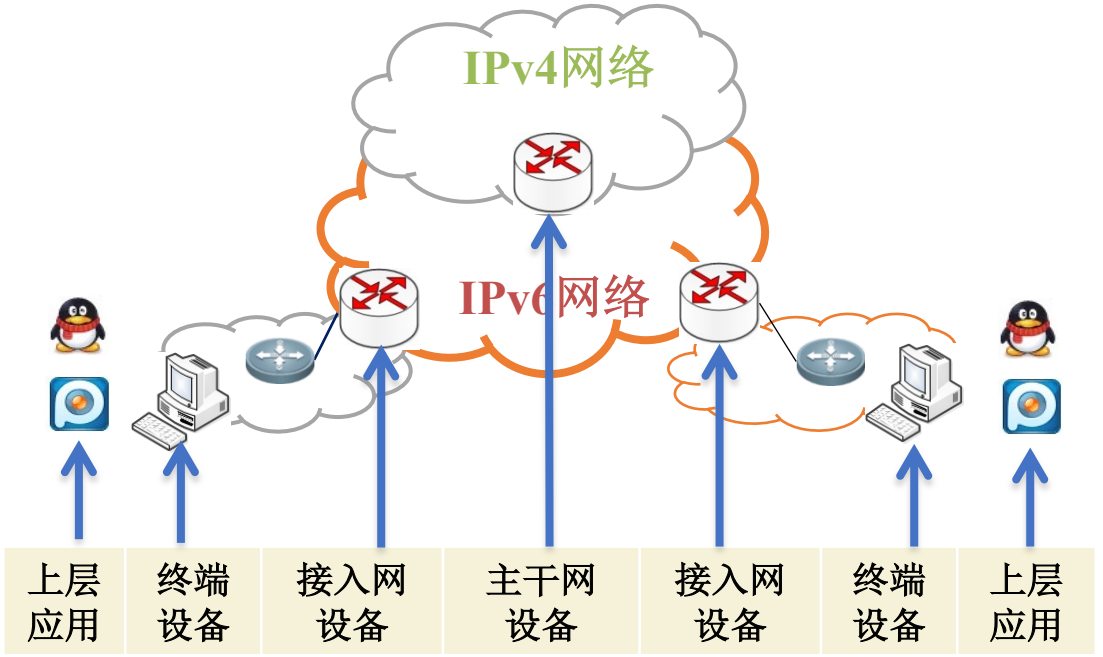
地址域：两个海量地址和路由空间的映射

IPv4到IPv6迁移及过渡技术难点

纵向贯穿体系结构各层次



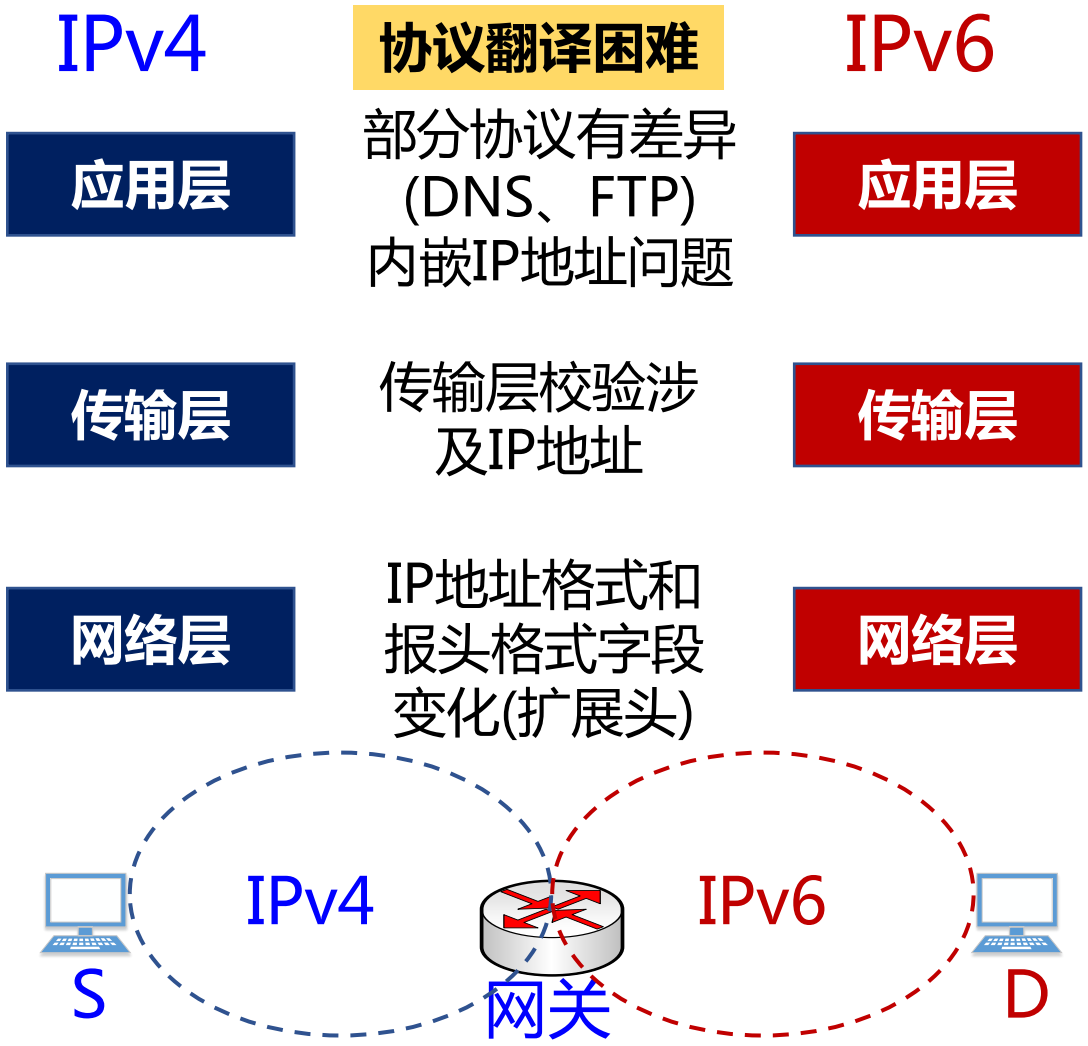
横向涉及网络结构各网元



IPv4向IPv6过渡是发展下一代互联网的重大技术难题

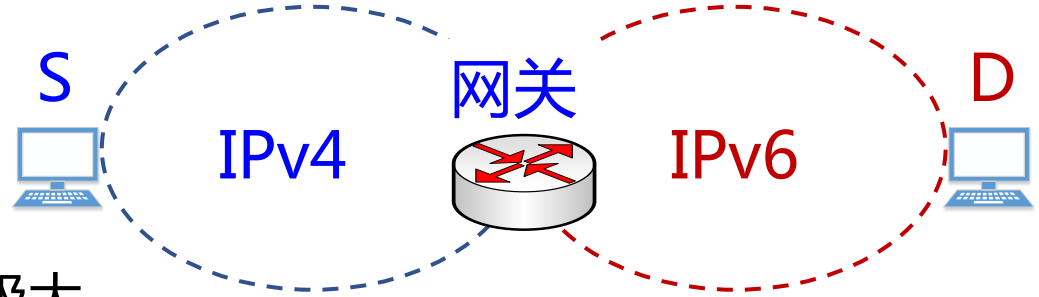
IPv4/IPv6翻译

- IPv4-IPv6地址往返转换
- IPv4报头和IPv6报头翻译
  - 各字段对应，包括IP选项翻译
- 传输层校验和转换，涉及IPv6地址
- ICMP翻译
  - ICMPv6重新设计了类型值和代码值
- 其它应用层协议翻译
  - FTP（命令名称变化，内嵌IP地址）
  - 内嵌IP地址的其它应用层协议
- DNS处理
  - 实现域名查询中A记录和4A记录的双向翻译



IPv4/IPv6翻译的问题

- 路由可扩展性问题
  - 96位前缀？给路由聚合带来巨大困难
- 应用层内嵌IP地址
  - 部分应用层协议内嵌IP地址，应用层翻译难度极大
- 翻译导致报文变长可能导致分片问题，影响转发性能
- 破坏互联网端到端原则
  - 通信双方只看到通信对端在本网络对应的地址
  - 翻译网关处理传输层、应用层（上层数据无法透明传输）
- 异构地址寻址问题
  - 域名访问需DNS服务支撑

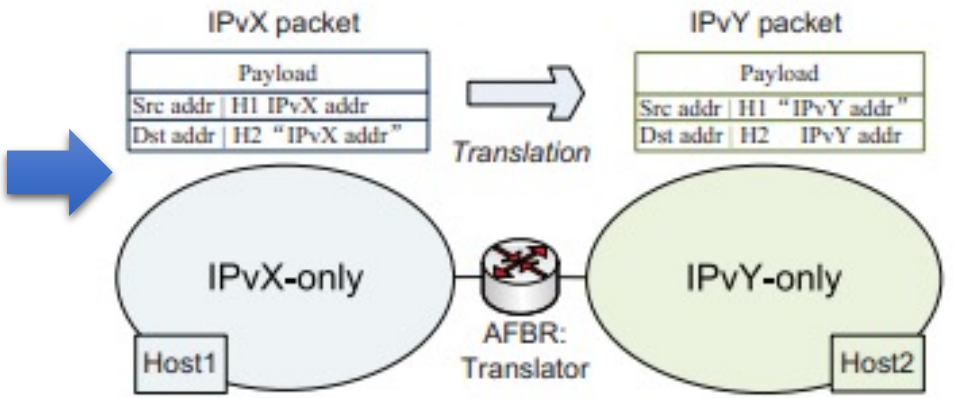


端到端加密呢？

两种过渡技术路线

翻译技术

形式化系统
相互转换

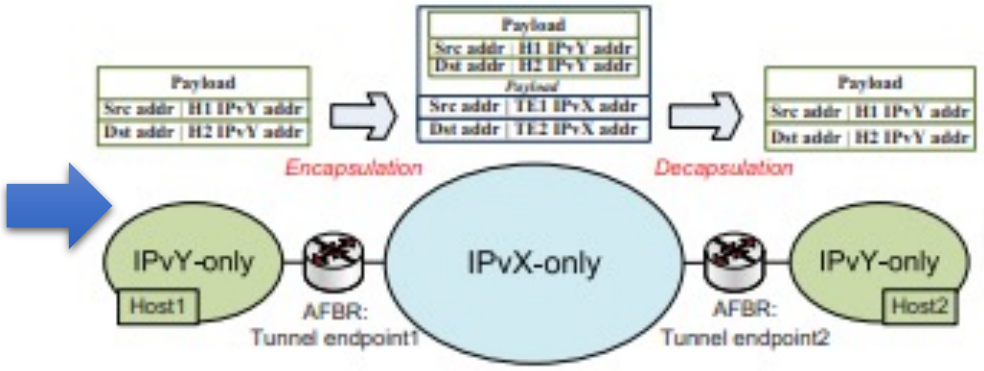


难实现完全转换

适用于特定
过渡场景
(RFC6144)

隧道技术

形式化系统
相互承载



可实现完全承载

广泛适用于
各种场景

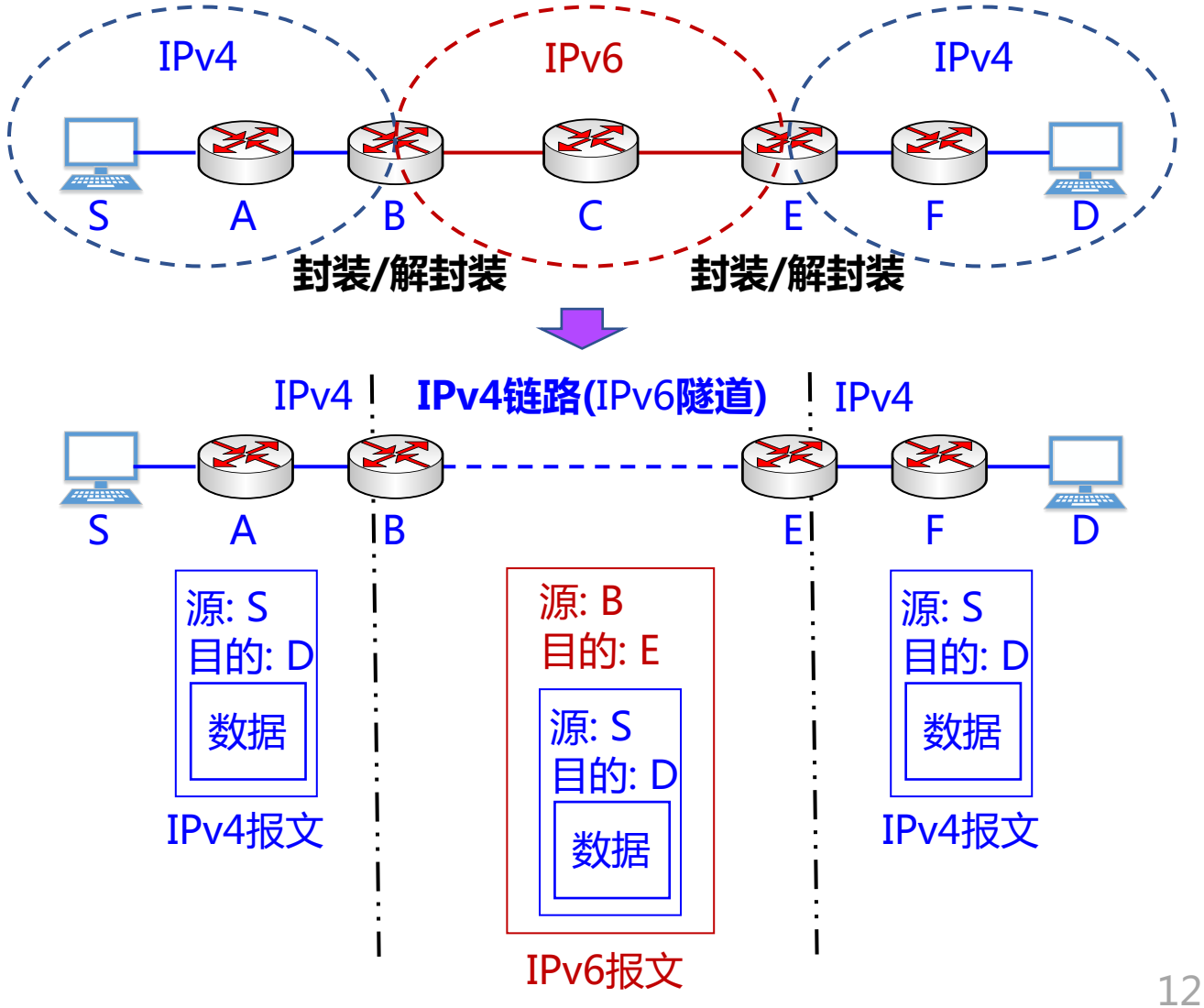
隧道技术

隧道技术

- 同构网络跨越异构网络进行通信
- 将A协议数据包封装在B协议中传输

隧道类型

- 应用层隧道
  - SSH隧道，HTTPS隧道
- 传输层隧道
  - TCP隧道，UDP隧道
- 网络层隧道
  - 4 in 4，4 in 6，6 in 4
  - GRE，通用路由封装隧道
- 链路层隧道
  - L2TP协议，链路层隧道
  - PPTP协议，点对点隧道



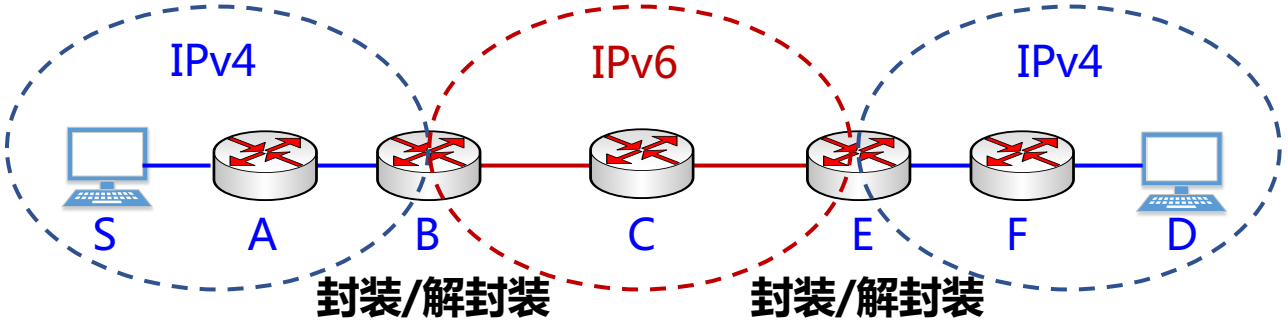
隧道技术

隧道技术

- 同构网络跨越异构网络进行通信
- 将A协议数据包封装在B协议中传输

隧道类型

- 应用层隧道
  - SSH隧道，HTTPS隧道
- 传输层隧道
  - TCP隧道，UDP隧道
- 网络层隧道
  - 4 in 4，4 in 6，6 in 4
  - GRE，通用路由封装隧道
- 链路层隧道
  - L2TP协议，链路层隧道
  - PPTP协议，点对点隧道

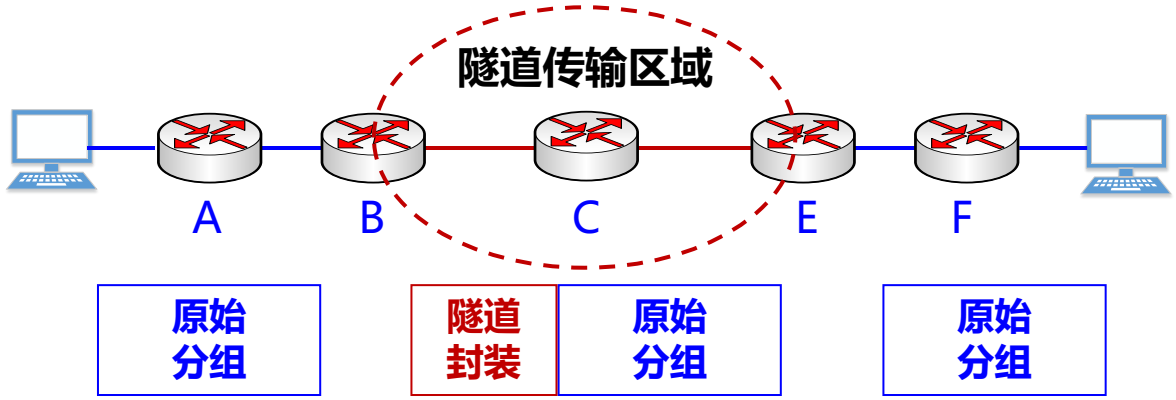


4over6隧道

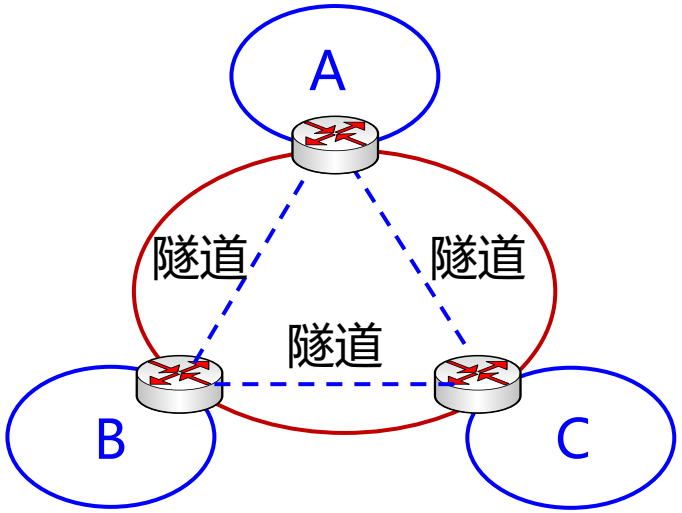
链路层隧道

隧道技术

- 报文长度增大导致的分片问题
  - 途中分片与重组对传输性能影响较大
- 解决方法
  - 隧道网关提前分片
  - Path MTU发现机制
- 隧道链路的选择问题
  - A发往F的报文，一路上如何路由？



- 静态配置
  - ip route 30.0.0.0/8 tunnel 1
- 4over6：与路由结合的动态学习
  - 主干网：扩展核心路由协议BGP
  - 接入网：DHCP协同NAT，实现轻量级
  - 从嘲笑、怀疑，到争吵追随
  - OpenWRT，中国电信、德电、法电



单选题 1分

设置

以下关于IPv6地址
1A22:120D:0000:0000:72A2:0000:0000:00C0的表示中，错误的是

A 1A22:120D::72A2:0000:0000:00C0

B 1A22:120D::72A2:0:0:C0

C 1A22::120D::72A2::00C0

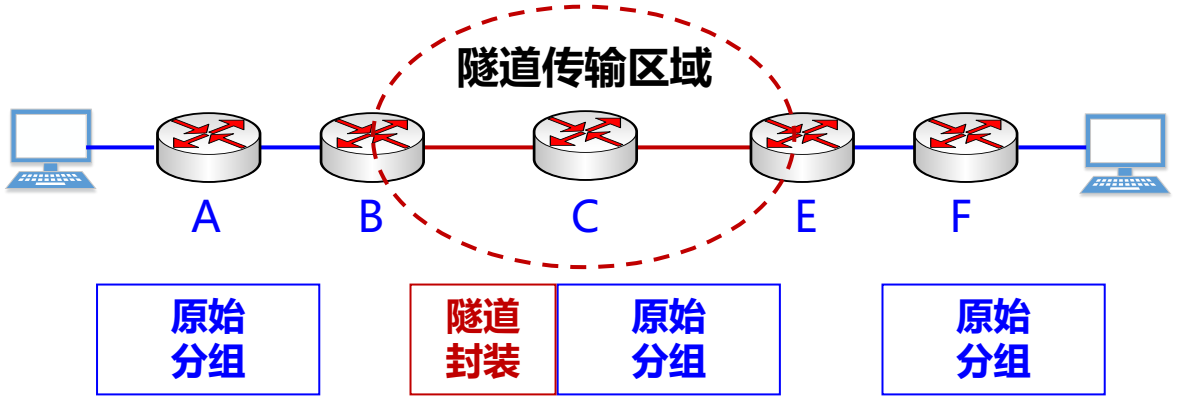
D 1A22:120D:0:0:72A2:0:0:C0

提交

NAT、IPv6技术小结

- 相似的出发点：IPv4地址不够用？
  - 在IPv4的框架中优化：NAT技术
  - 拓展地址位数，新设计：IPv6技术
- NAT技术
  - NAT转换表：内网(IP，端口)<->外网(IP，端口)
- IPv6技术
  - 地址空间：32 -> 128位
  - IP协议族和路由协议的IPv6设计
  - IPv4/IPv6共存：翻译、隧道

网络层基础：IPv6
动态地址分配：DHCPv6
单跳网络处理：邻居发现
基础管理和控制：ICMPv6
距离向量路由：RIPng
链路状态路由：OSPFv3
外部网关路由：MP-BGP



本章内容

5.1 网络层概述

5.2 网络层协议

5.3 路由算法

5.4 流量管理与服务质量

5.5 路由器体系结构与关键技术

5.6 软件定义网络

1. 优化原则
2. 最短路径算法
3. 距离向量路由
4. 链路状态路由
5. 组播路由
6. 路由协议

路由算法

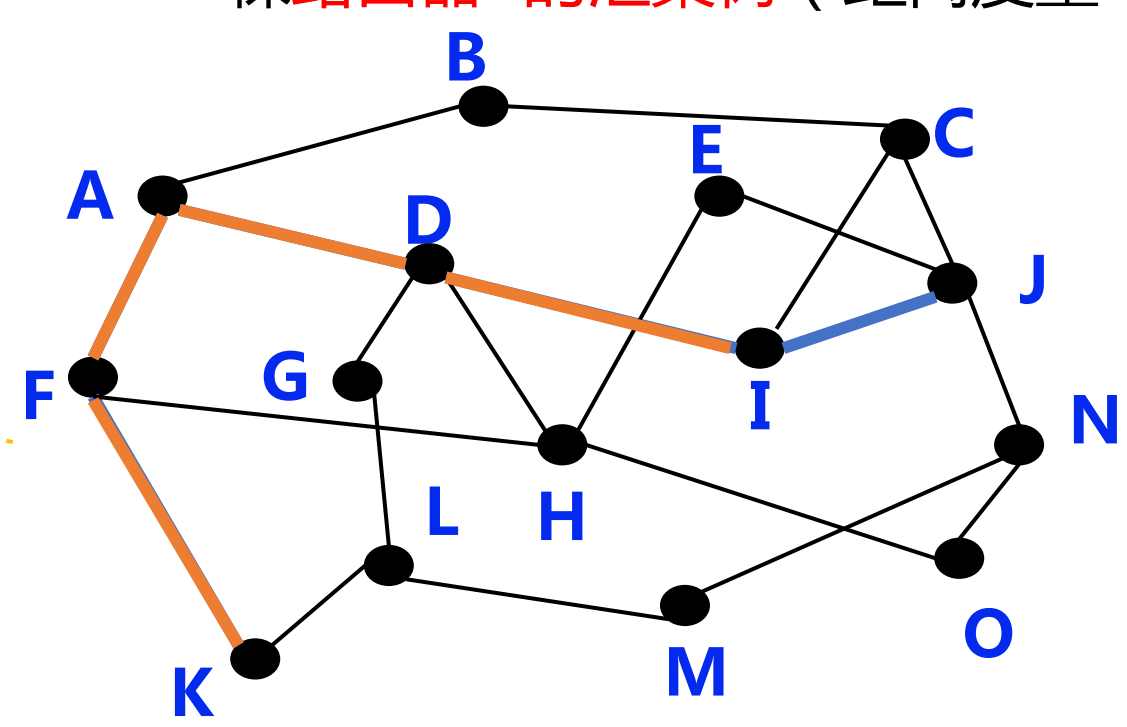
- 路由算法须满足的特性：
  - 正确性
  - 简单性
  - 鲁棒性
  - 稳定性
  - 公平性
  - 有效性
- 根据路由算法是否随网络的通信量或拓扑自适应划分
  - 静态路由选择策略（非自适应路由选择）
  - 动态路由选择策略（自适应路由选择）

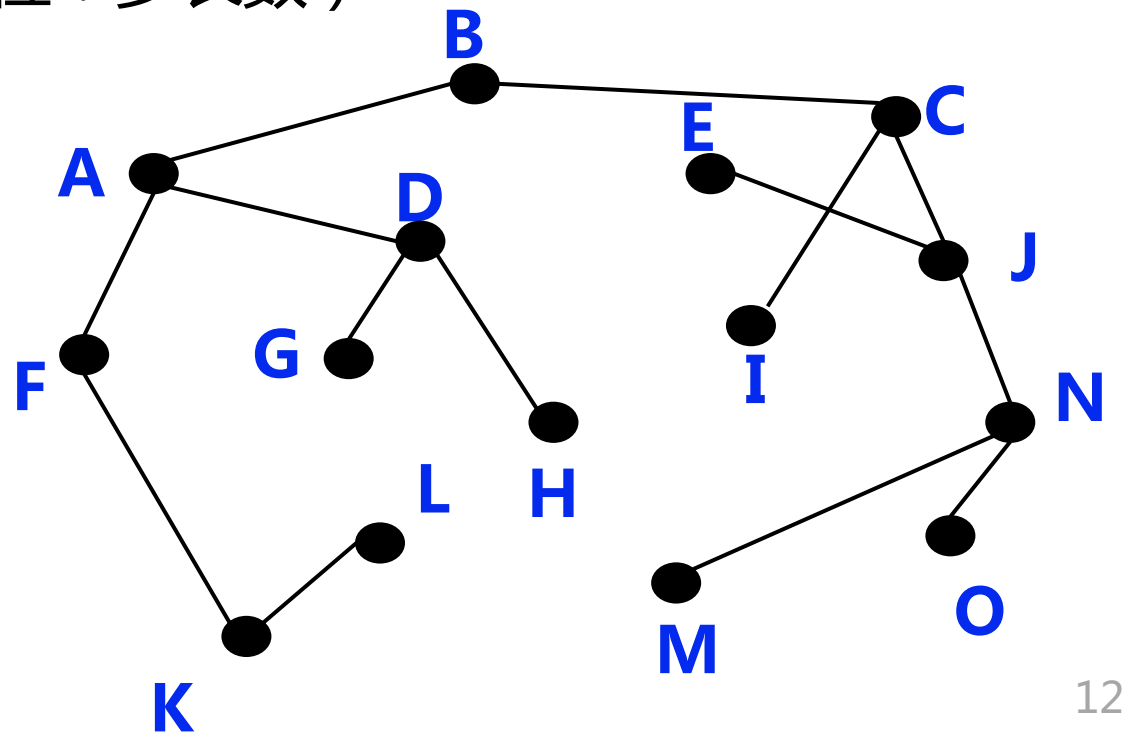
优化原则

- 汇集树(Sink Tree)

汇集树不是唯一的

  - 所有的源节点到一个指定目标节点的最优路径的集合构成一棵以目标节点为根的树
  - 一棵路由器B的汇集树（距离度量单位：步长数）



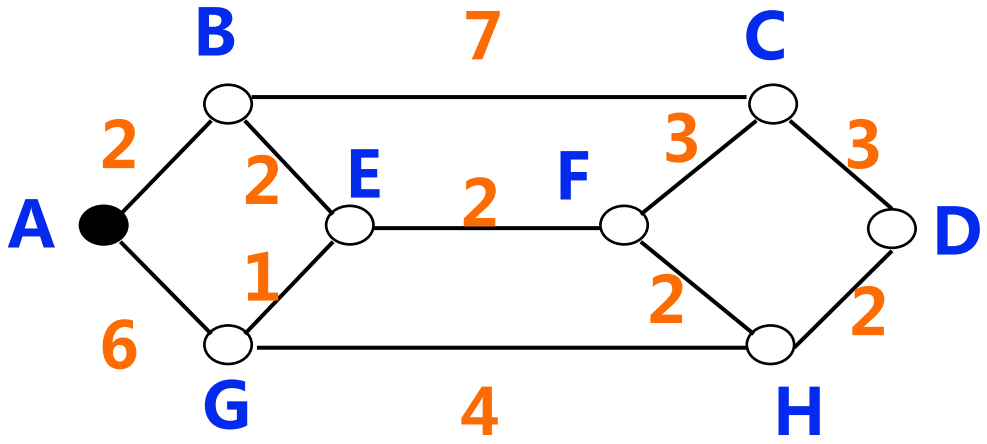


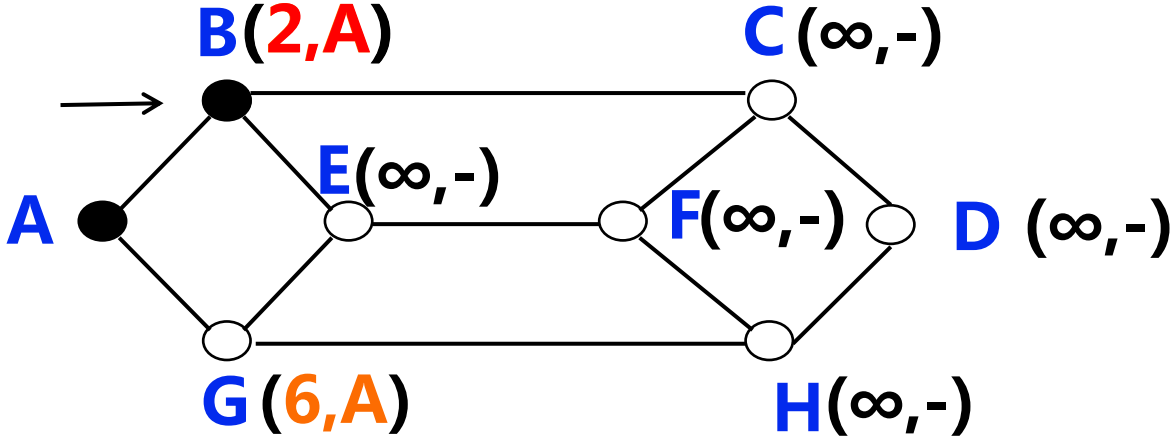
最短路径算法

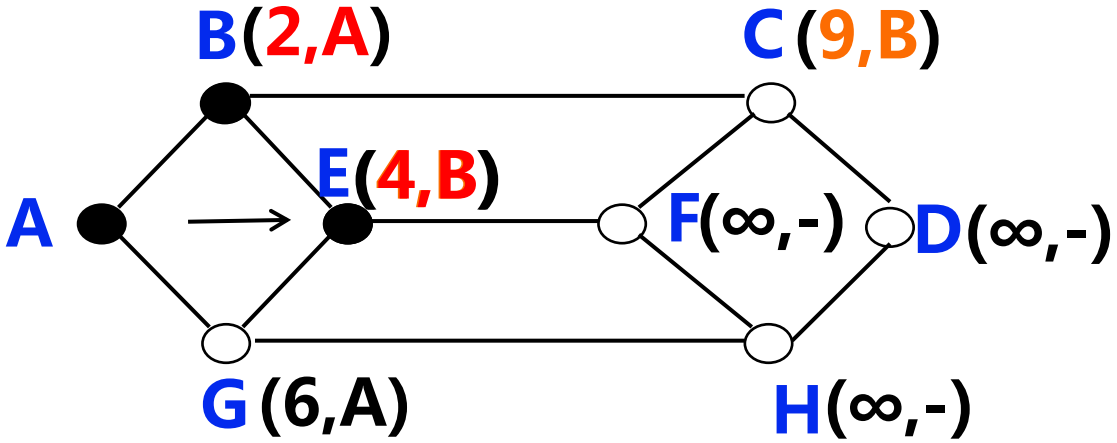
- 定义
  - 用于计算一个节点到其他所有节点的最短路径，主要特点是以起始点为中心向外逐层扩展，直到扩展到终点为止
- Dijkstra算法思想
  - 建立网络图
    - 节点表示路由器
    - 边表示通信线路/链路
    - 链路代价表示链路上的距离、信道宽度或通信开销等参数
  - 根据算法在网络图上为某一对路由器找之间的最短路径

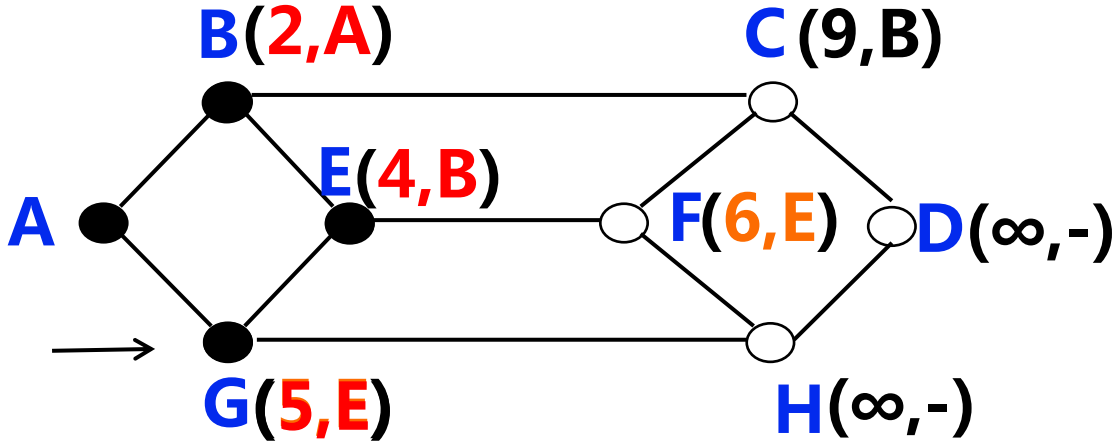
最短路径算法

- 具体例子（A到D的最短路径过程）



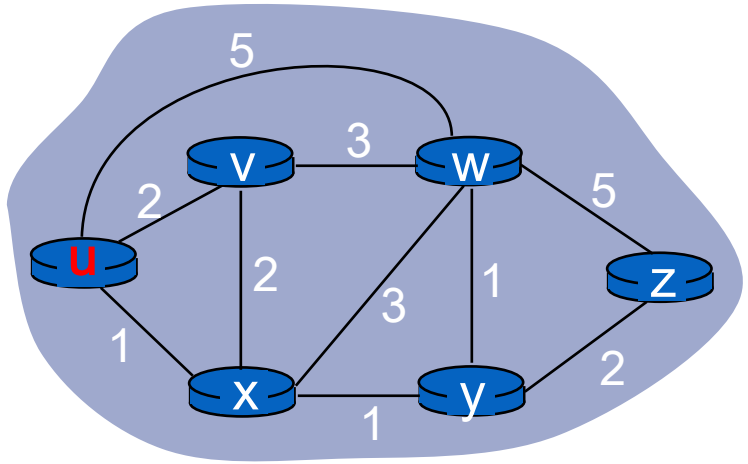






距离向量路由

- Bellman-Ford 方程
  - 假设$D_x(y)$ 是 从x到y最小代价路径的代价值
  - 则：$D_x(y) = \min \{c(x,m) + D_m(y)\}$
    其中m为x的邻居, c(x,m)为m到X的距离

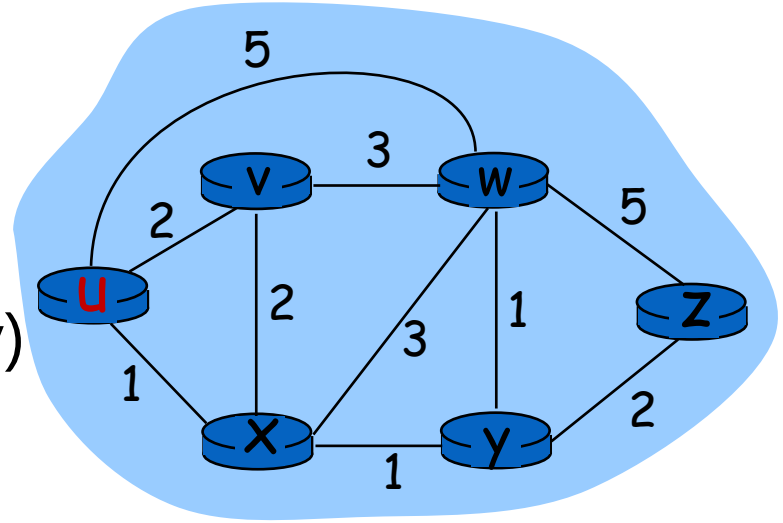


  - 示例：
    - 已获知$D_v(z) = 5$, $D_x(z) = 3$, $D_w(z) = 3$
    - $D_u(z) = \min \{ c(u,v) + D_v(z), c(u,x) + D_x(z), c(u,w) + D_w(z) \}$
      $= \min \{2 + 5, 1 + 3, 5 + 3\}$
      $= 4$
  - 计算出代价最小的节点，也就得到了对应转发项从u去往z，应从x转发

距离向量路由

- 距离向量（Distance Vector）算法基本思想:
  - 每个节点周期性地向邻居发送它自己到某些节点的距离向量；
  - 当节点x接收到来自邻居的新DV估计，它使用B-F方程更新其自己的DV：

$$D_x(y) \leftarrow min_v\{c(x,v) + D_v(y)\} \quad for\ each\ node\ y \in N$$

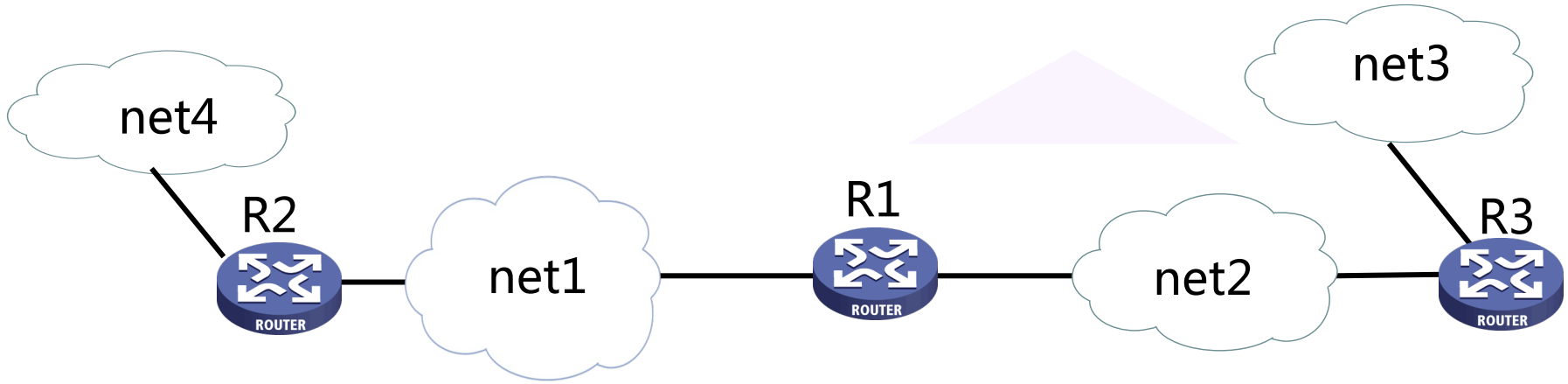


  - 上述过程迭代执行，$D_x(y)$*收敛为实际最小费用* $d_x(y)$

- 距离向量算法特点：迭代的、分布式的
  - 每次本地迭代由下列引起: 本地链路费用改变、邻居更新报文
  - 分布式:各节点依次计算，相互依赖

距离向量路由

- 路由器启动时初始化自己的路由表
  - 初始路由表包含所有直接相连的网络路径，距离均为0



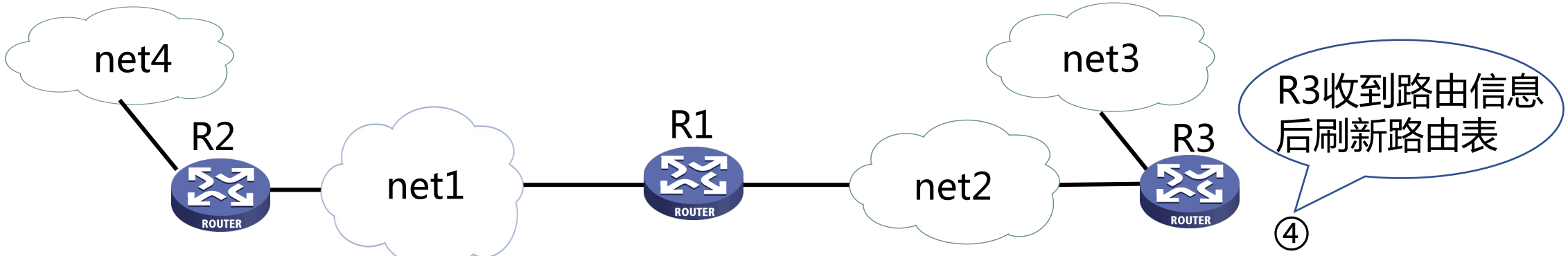
目的网络	距离	下一跳
net1	0	--
net4	0	--

目的网络	距离	下一跳
net1	0	--
net2	0	--

目的网络	距离	下一跳
net2	0	--
net3	0	--

距离向量路由

- 路由器周期性地向其相邻路由器广播自己知道的路由信息
- 相邻路由器可以根据收到的路由信息修改和刷新自己的路由表



目的网络	距离	下一跳
net1	0	--
net4	0	--

①

R2将路由信息发送给R1

目的网络	距离	下一跳
net1	0	--
net2	0	--
net4	1	R2

②

R1收到路由信息后刷新路由表

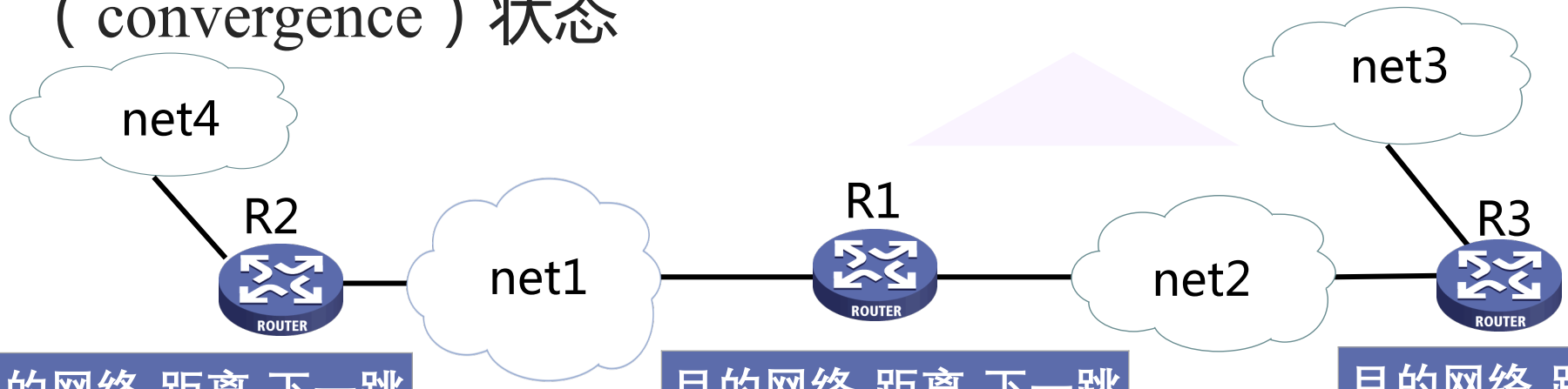
③

R1将路由信息发送给R3

目的网络	距离	下一跳
net2	0	--
net3	0	--
net1	1	R1
net4	2	R1

距离向量路由

- 路由器经过若干次更新后，最终都会知道到达所有网络的最短距离
- 所有的路由器都得到正确的路由选择信息时网络进入“收敛”（convergence）状态



目的网络	距离	下一跳
net1	0	--
net4	0	--
net2	1	R1
net3	2	R1

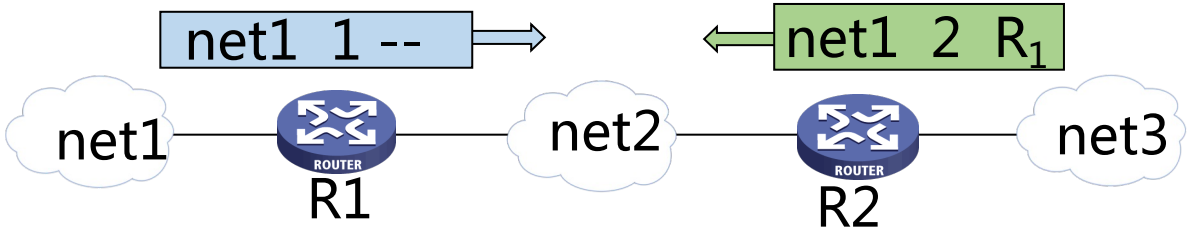
目的网络	距离	下一跳
net1	0	--
net2	0	--
net4	1	R2
net3	1	R3

目的网络	距离	下一跳
net2	0	--
net3	0	--
net1	1	R1
net4	2	R1

距离向量路由

- 计数到无穷问题（The Count-to-Infinity Problem）

正常情况

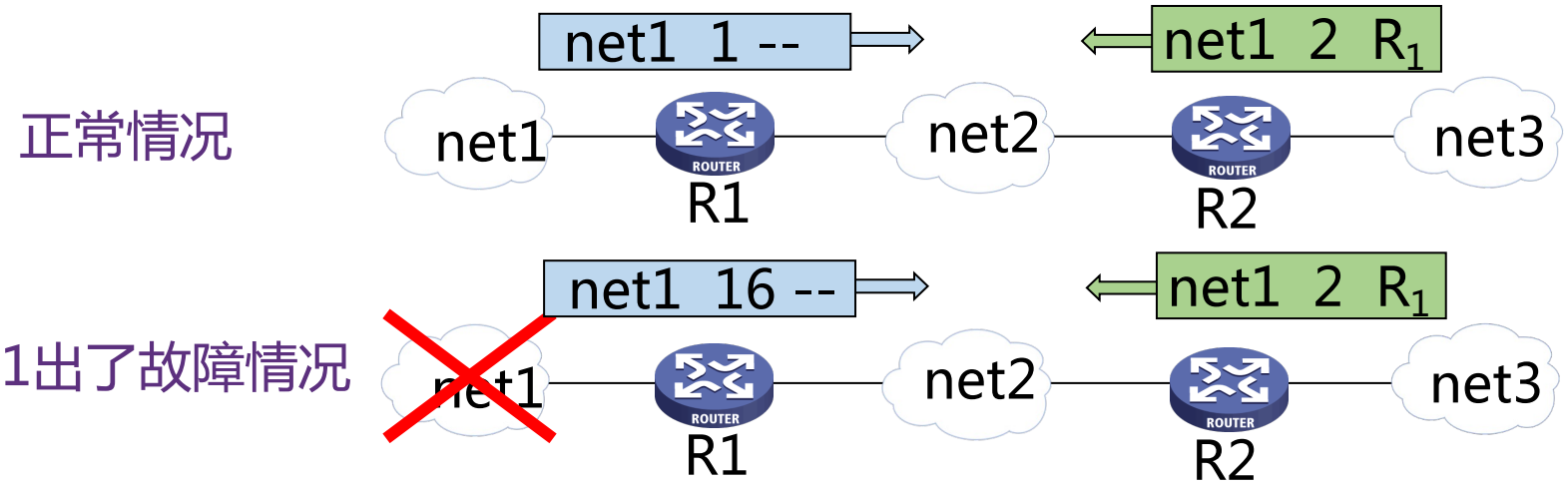


R_1 说：“我到net1 的距离是 1，是直接交付。”

R_2 说：“我到net1 的距离是 2，是经过 R_1。”

距离向量路由

- 计数到无穷问题（The Count-to-Infinity Problem）



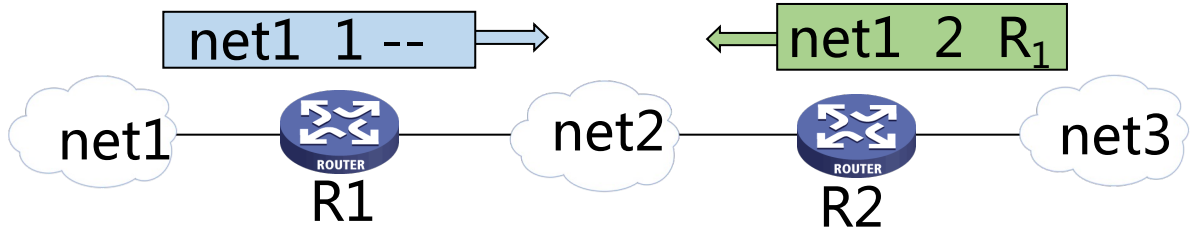
R_1 说：“我到net1 的距离是 16（表示无法到达），是直接交付。”

但 R_2 在收到 R_1 的更新报文之前，还发送原来的报文，因为这时 R_2 并不知道 R_1 出了故障。

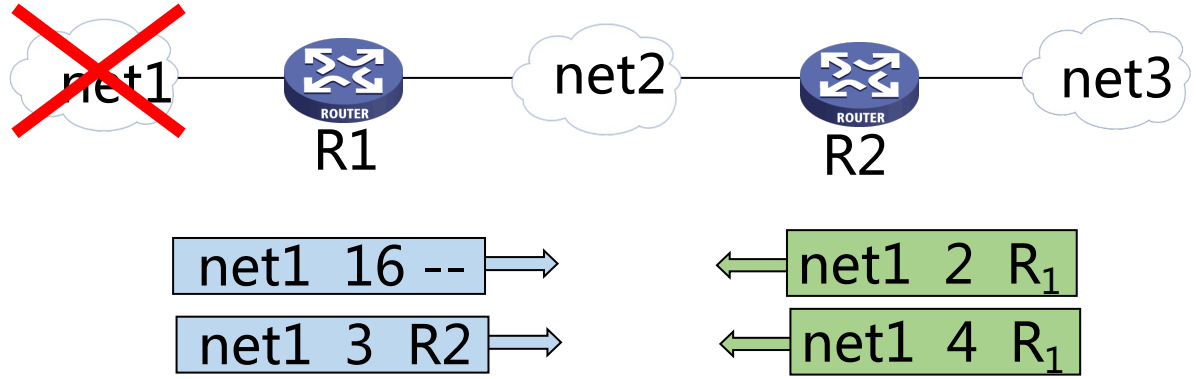
距离向量路由

- 计数到无穷问题（The Count-to-Infinity Problem）

正常情况



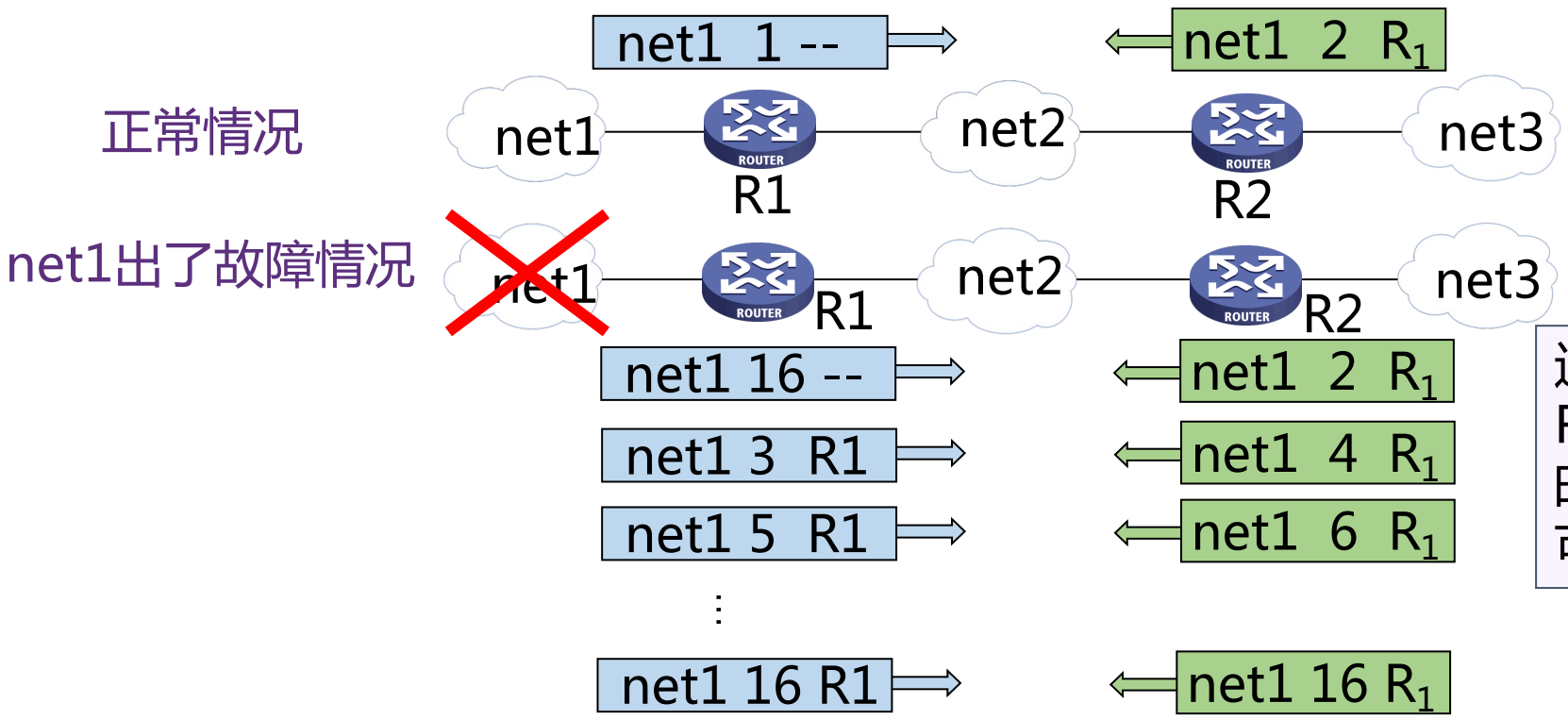
net1出了故障情况



R_2 以后又更新自己的路由表为“net1, 4, R_1”，表明 “我到net 1 距离是 4，下一跳经过 R_1”。

距离向量路由

- 计数到无穷问题（The Count-to-Infinity Problem）



这样不断更新下去，直到 R_1 和 R_2 到net 1 的距离都增大到16时， R_1 和 R_2才知道net 1 是不可达的。

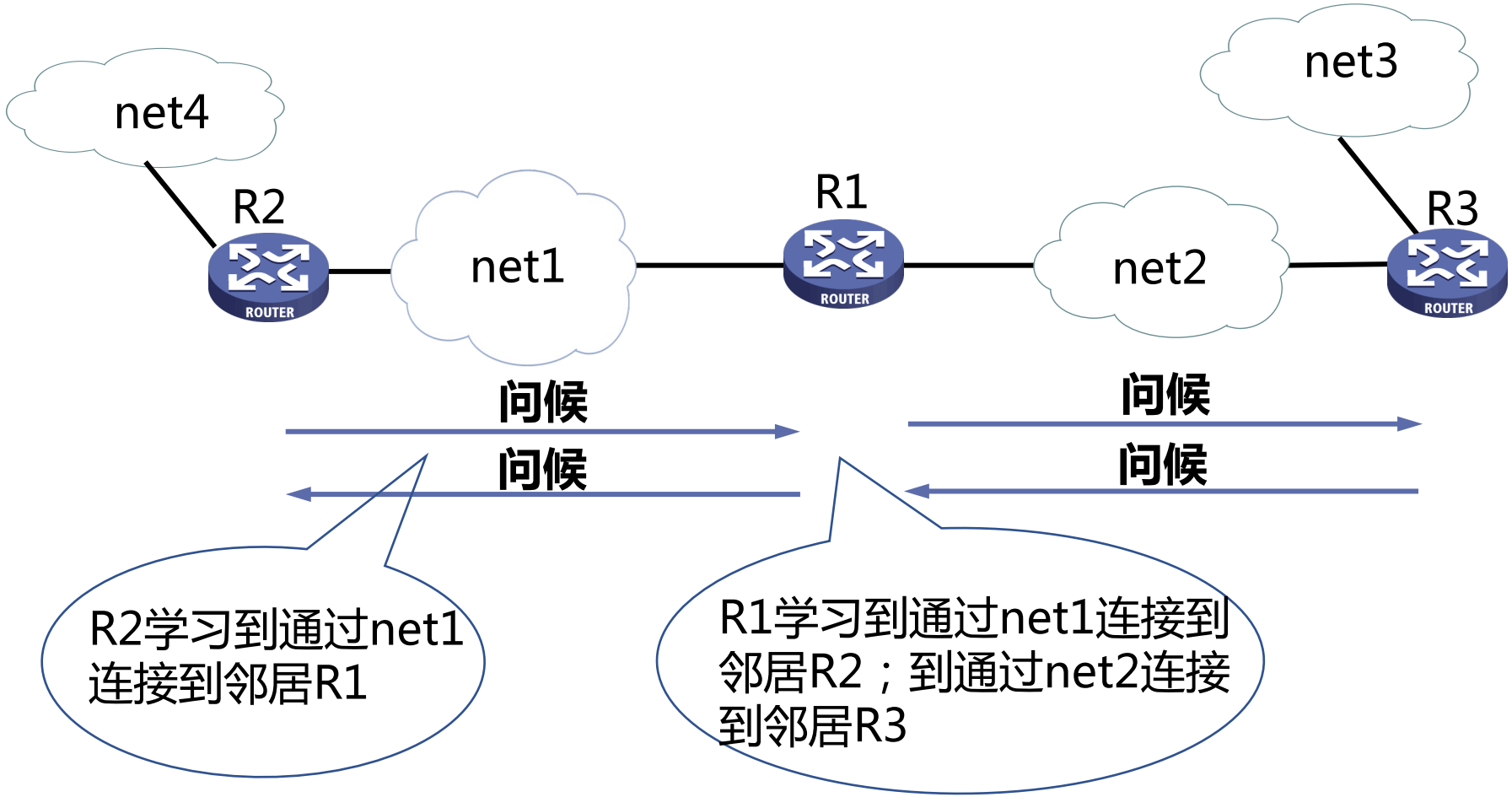
- 好消息传播快，坏消息传播慢，是距离向量路由的一个主要缺点

链路状态路由

- 链路状态（Link State）路由可分为五个部分：
  - 1. 发现邻居，了解他们的网络地址；
  - 2. 设置到每个邻居的成本度量；
  - 3. 构造一个分组，分组中包含刚收到的所有信息；
  - 4. 将此分组发送给其他的路由器；
  - 5. 计算到其他路由器的最短路径。

链路状态路由

- 1.发现邻居，了解他们的网络地址

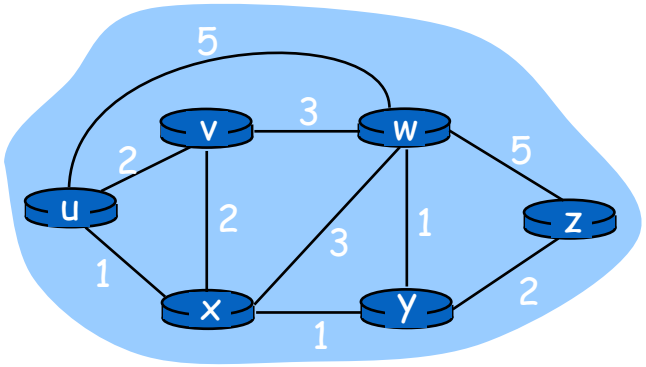


链路状态路由

- 2.设置到每个邻居的成本度量
  - 开销/度量/代价：
    - 自动发现设置或人工配置
    - 度量：带宽、跳数、延迟、负载、可靠性等
  - 常用度量：链路带宽（反比）
    - 例如：1-Gbps以太网的代价为1，100-Mbps以太网的代价为10
  - 可选度量：延迟
    - 发送一个echo包，另一端立即回送一个应答
    - 通过测量往返时间RTT，可以获得一个合理的延迟估计值

链路状态路由

- 3.构造一个分组，分组中包含刚收到的所有信息
  - 构造链路状态分组（link state packet，LSP）
    - 发送方标识
    - 序列号
    - 年龄
    - 邻居列表



u	
Seq.	
Age	
v	2
x	1
w	5

v	
Seq.	
Age	
u	2
x	2
w	3

x	
Seq.	
Age	
u	1
v	2
w	3
y	1

y	
Seq.	
Age	
x	1
w	1
z	2

w	
Seq.	
Age	
u	5
v	3
x	3
y	1
z	5

z	
Seq.	
Age	
w	5
y	2

链路状态路由

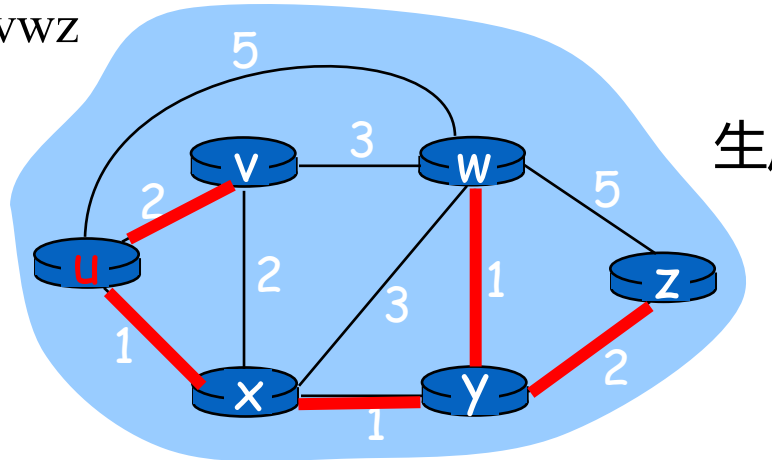
- 4.将LSP分组发送给其他的路由器
  - 每个LSP分组包含一个序列号，且递增
  - 路由器记录所收到的所有（源路由器、序列号）对
  - 当一个新分组到达时，路由器根据记录判断：
    - 如果是新分组，洪泛广播
    - 如果是重复分组，丢弃
    - 如果是过时分组，拒绝

链路状态路由

- 5.计算到其他路由器的最短路径：Dijkstra算法示例
  - D(k)：从计算节点到目的节点k当前路径代价
  - p(k)：从计算节点到目的节点k的路径中k节点的前继节点

步骤	集合N	D(v),p(v)	D(w),p(w)	D(x),p(x)	D(y),p(y)	D(z),p(z)
0	u	2,u	5,u	1,u	∞	∞
1	ux	2,u	4,x		2,x	∞
2	uxy	2,u	3,y			4,y
3	uxyv		3,y			4,y
4	uxyvw					4,y
5	uxyvwz					

生成最短路径树



生成路由表

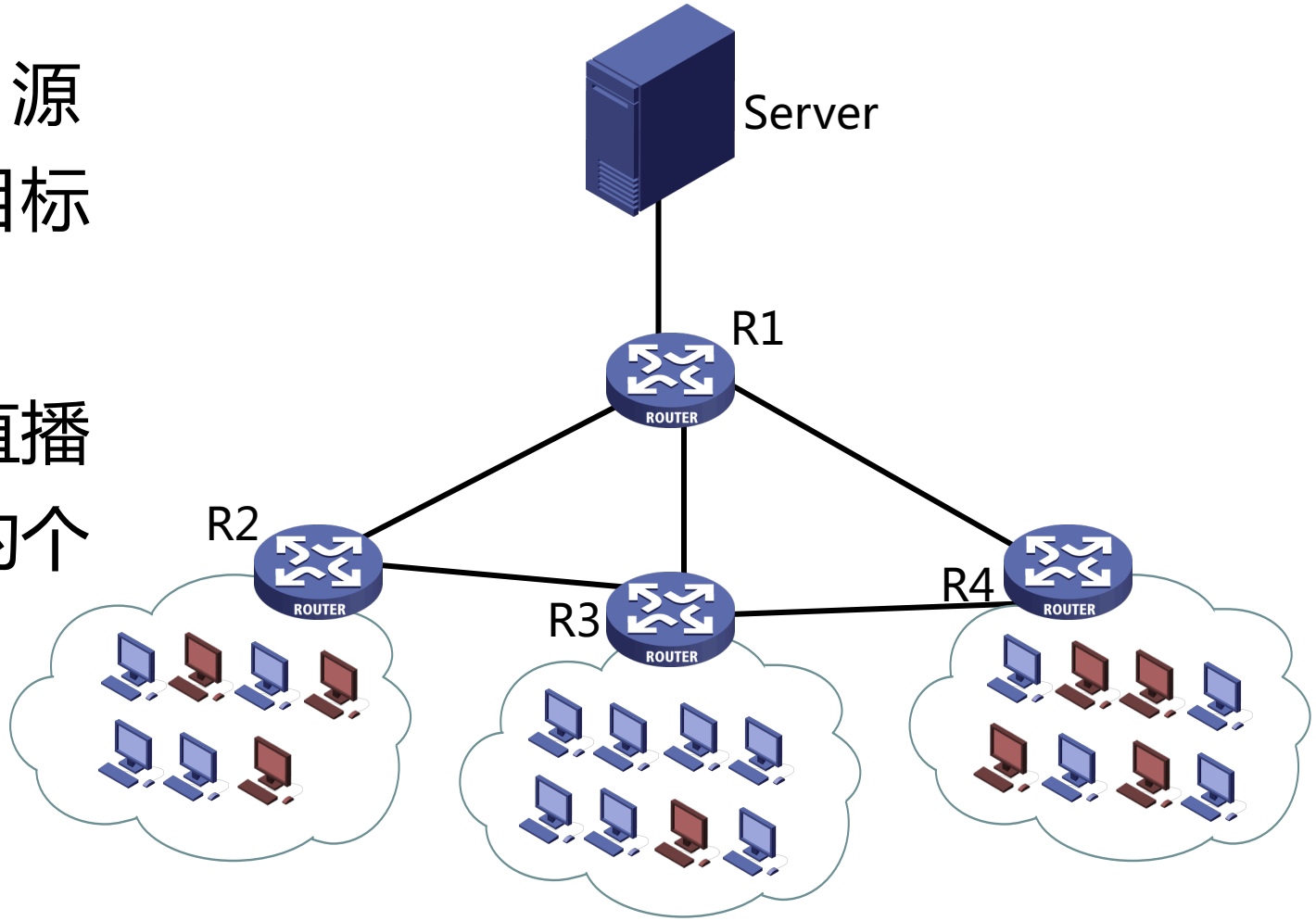
目的	下一跳	代价
v	v	2
w	x	3
x	x	1
y	x	2
z	x	4

链路状态路由

- 距离向量和链路状态算法比较：
  - 网络状态信息交换的范围
    - DV:邻居间交换
    - LS:全网扩散
  - 网络状态信息的可靠性
    - DV:部分道听途说
    - LS:自己测量
  - 健壮性:
    - DV:计算结果传递，健壮性差
    - LS: 各自计算，健壮性好
  - 收敛速度：
    - DV: 慢,可能有计数到无穷问题
    - LS: 快

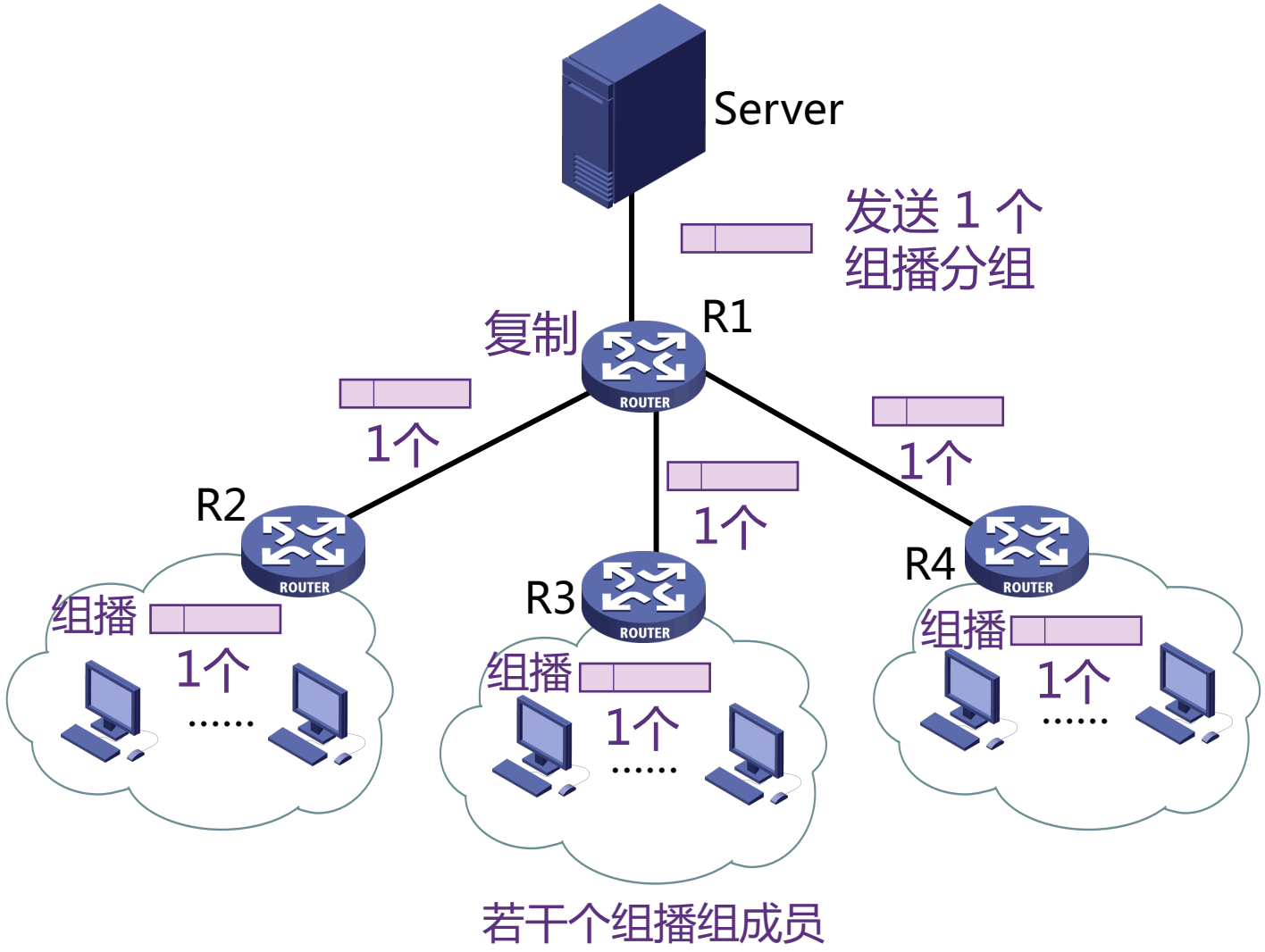
组播路由

- 组播（multicasting）：源主机给网络中的一部分目标用户发送数据包
- 例：服务器希望将体育直播视频发送给某些网络中的个别用户，怎么办？



组播路由

- 组播（multicasting）路由算法的目标：
  - 为每个组建立多播转发树（到达该组所有成员的路径树）
  - 每个组成员应当只收到多播分组的一个拷贝
  - 非本组成员不应收到多播分组
  - 从源节点到每一个组成员节点的路径应当是最佳的（最短路径）

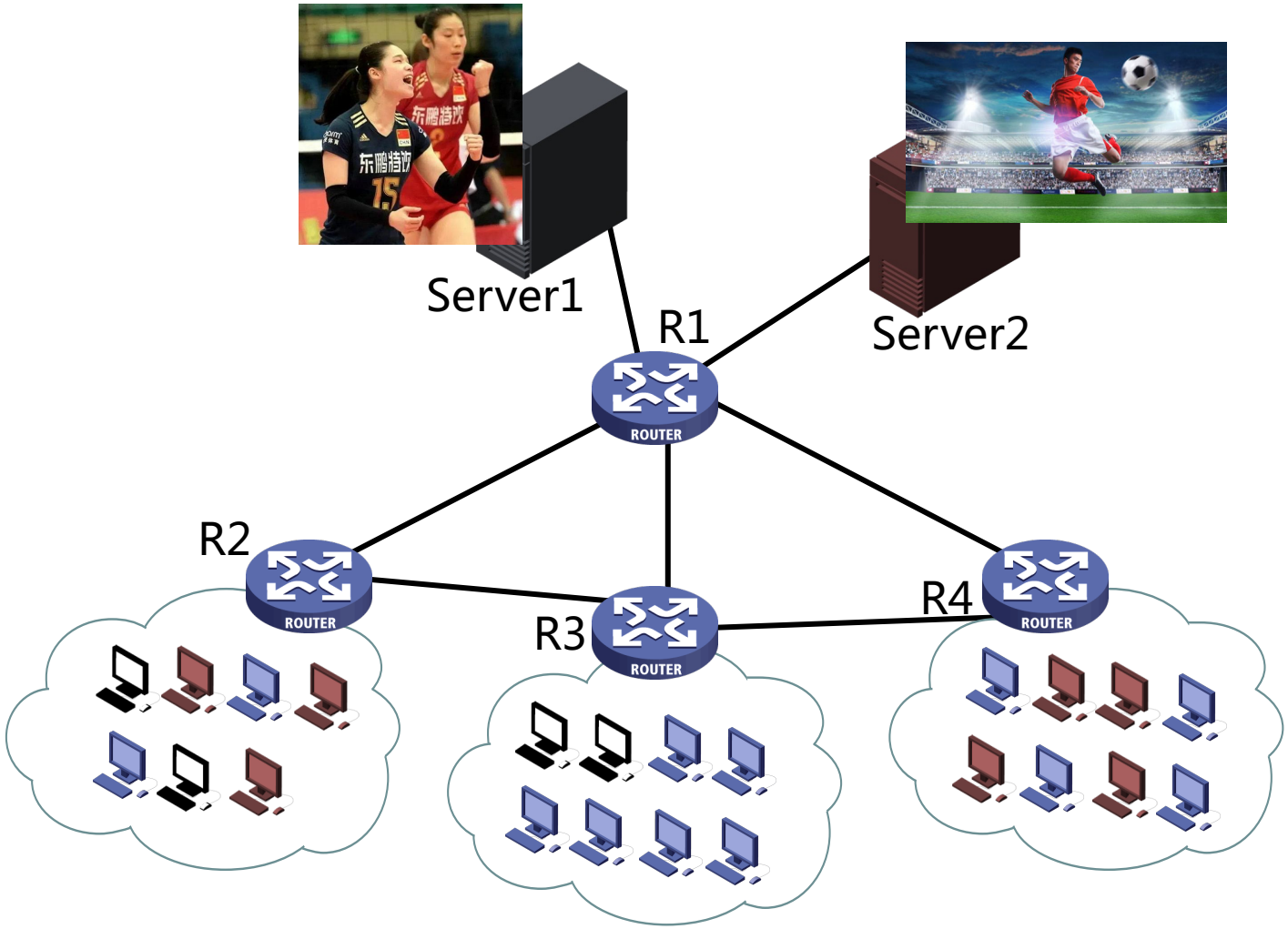


组播路由

➢组播实现的两个步骤：

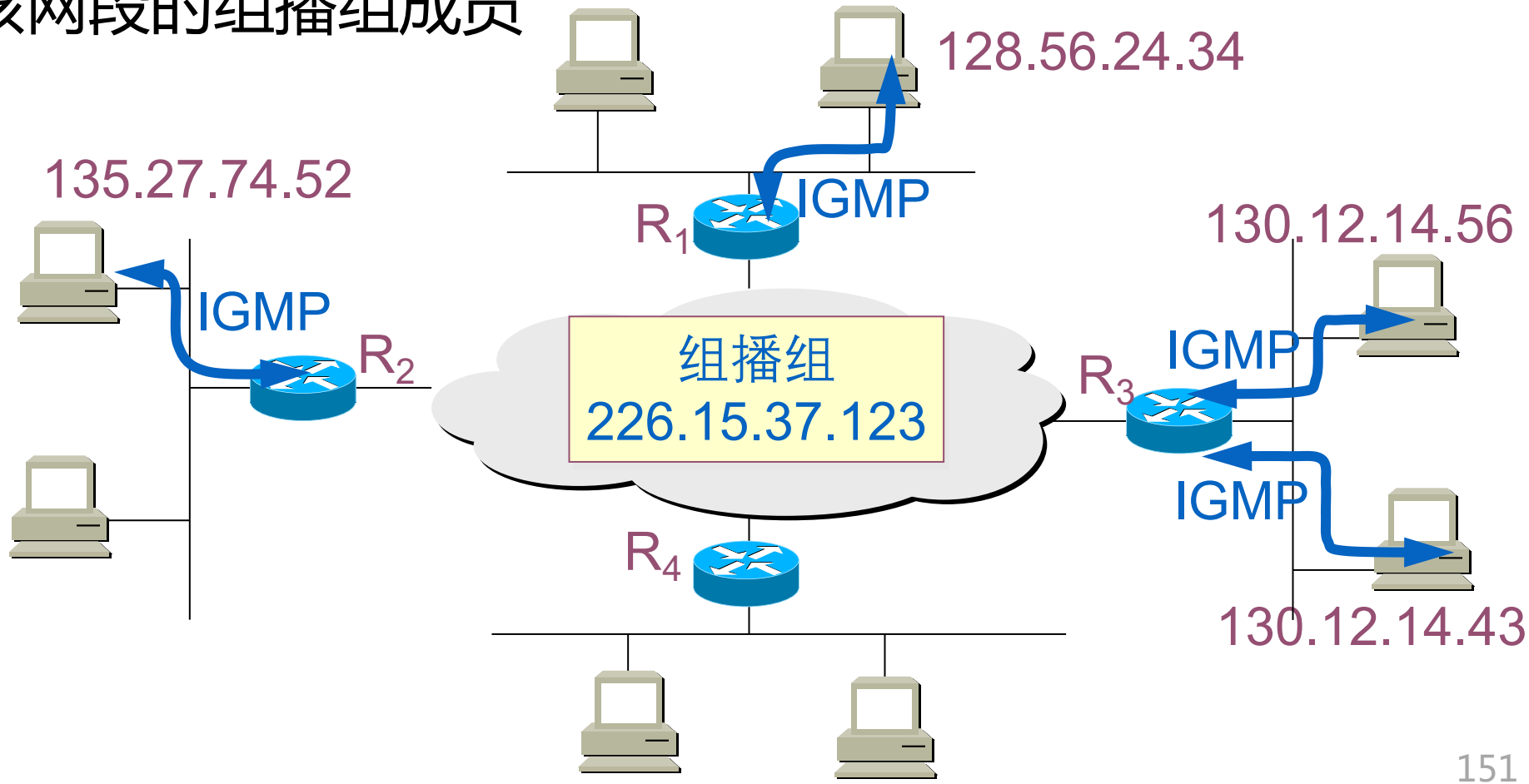
确定组成员：边缘路由器通过与主机交互，了解到从它的某个端口可以到达哪些组的成员——主机与路由器之间的组成员关系协议

- 怎样标识组成员？
- 如何编址？



组播路由

- IGMP (Internet Group Management Protocol)
  路由器获悉该网段的组播组成员

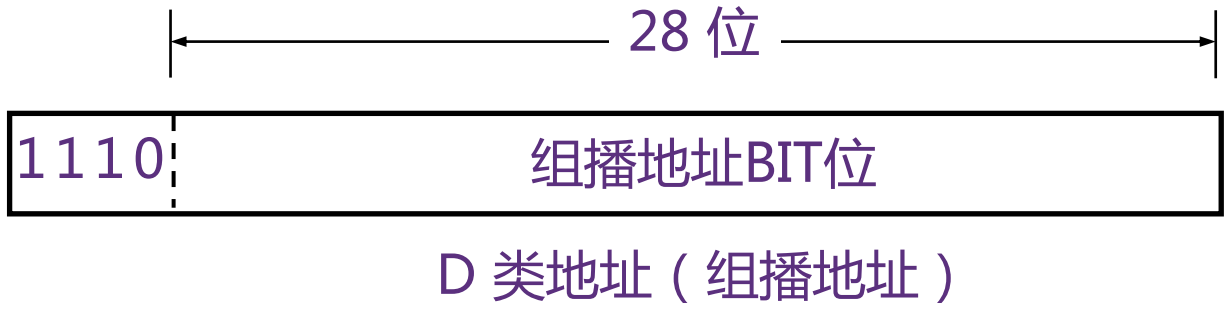


组播路由

- 常用组播地址段：224.0.0.0/24

  局域网组播地址(一跳子网内使用)

  - 224.0.0.1　　LAN上所有设备
  - 224.0.0.2　　LAN上所有路由器
  - 224.0.0.5　　LAN上所有OSPF路由器
  - 224.0.0.251 LAN上所有DNS服务器



组播路由

- 组播实现的两个步骤：

生成树——路由器与路由器之间的协议

  - 建立在我们已经学习过的广播路由方案基础之上
  - 数据包沿生成树发送
  - 最佳生成树的使用取决于组的密度分布
    - 密集分布：接受者遍布在网络的大部分区域
    - 稀疏分布：大部分网络区域都不属于组播组

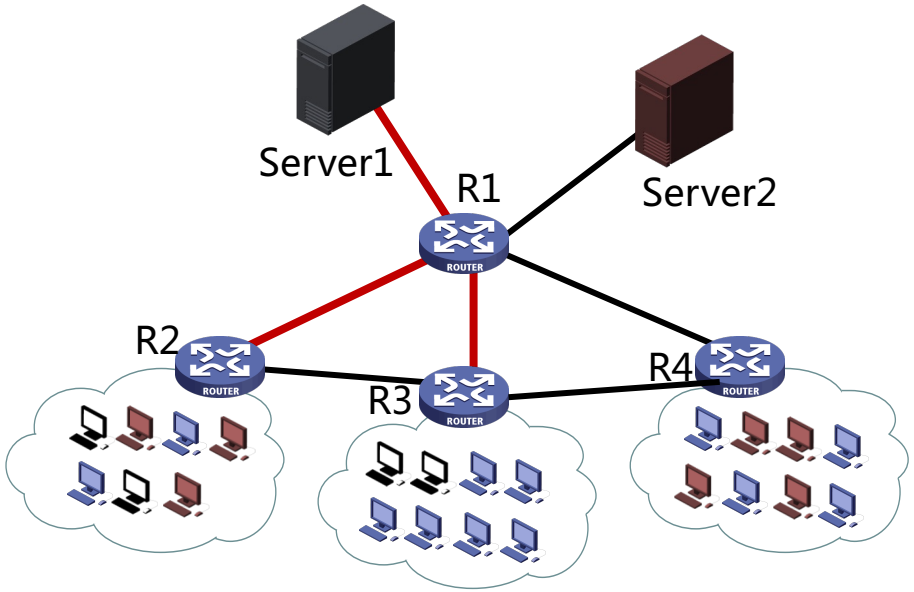
组播路由

- 密集分布，基于源点树（source-based trees）
  - 链路状态路由算法：每个路由器针对组内的每个发送者构造一颗独立树，例如多播开放最短通路优先协议（Multicast Open Shortest Path First，MOSPF）
  - 距离向量路由算法：逆向路径转发，修剪没有组成员的路由器，例如距离向量多播路由协议（Distanse Vector Multicast Routing Protocol, DVMRP）、协议无关多播-稠密模式（Protocol Independent Multicast - Dense Mode, PIM-DM）

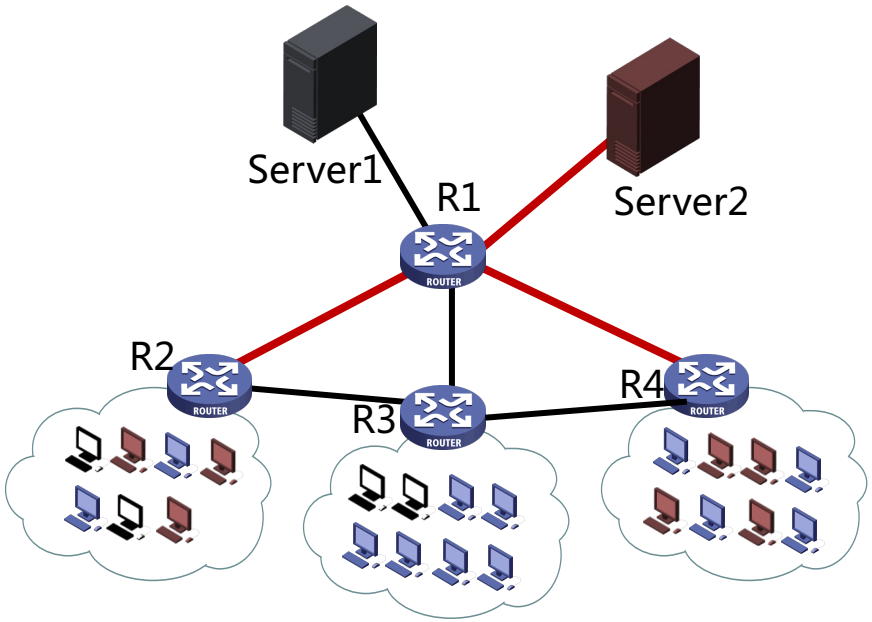
协议无关指的是与单播路由协议无关，即PIM不需要维护专门的单播路由信息

组播路由

- 基于源点树（source-based trees）存在的问题：
  - 大型网络中，组播源很多时，路由器需生成多颗棵树，工作量巨大
  - 路由器需要大量空间来存储多颗树



以Server1为源生成的树



以Server2为源生成的树

组播路由

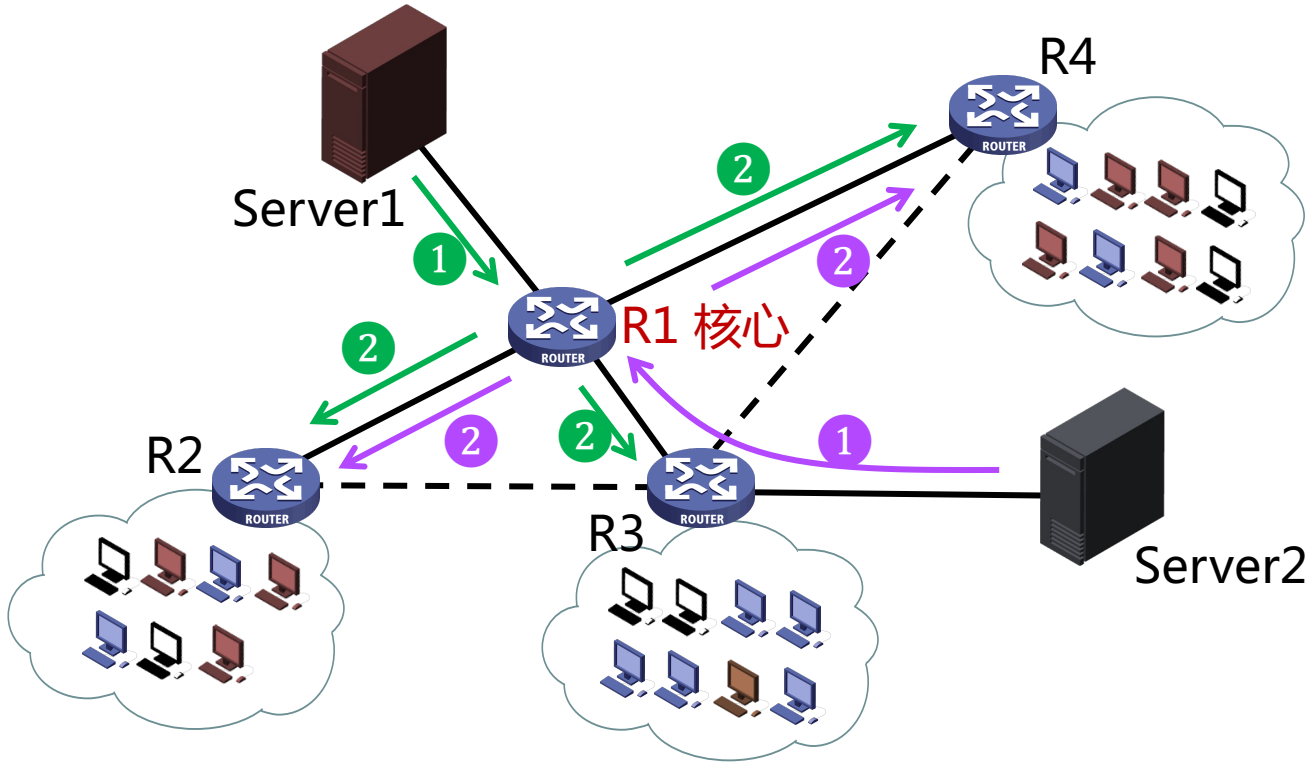
- 稀疏分布，基于核心树（core-based trees）
  - 多个组播源**共享树**
  - 大大节省存储开销、消息发送和计算
  - 每个路由器只需要保存一棵树
  - 不属于共享树的路由器不需要为组做任何工作
  - 例如协议无关多播-稀疏方式（ Protocol Independent Multicast - Sparse Mode，PIM-SM），特定源组播（ Protocol Independent Multicast - source-specific multicast, PIM-SSM）

组播路由

- 协议无关多播-稀疏方式 PIM-SM

① RP（Rendezvous Point），组播流量汇聚点，所有组播接收路由器以RP为树根构建共享树(RPT)；

② 发送端路由器将组播报文发给RP（通过最短路径），RP再通过RPT分发组接收者



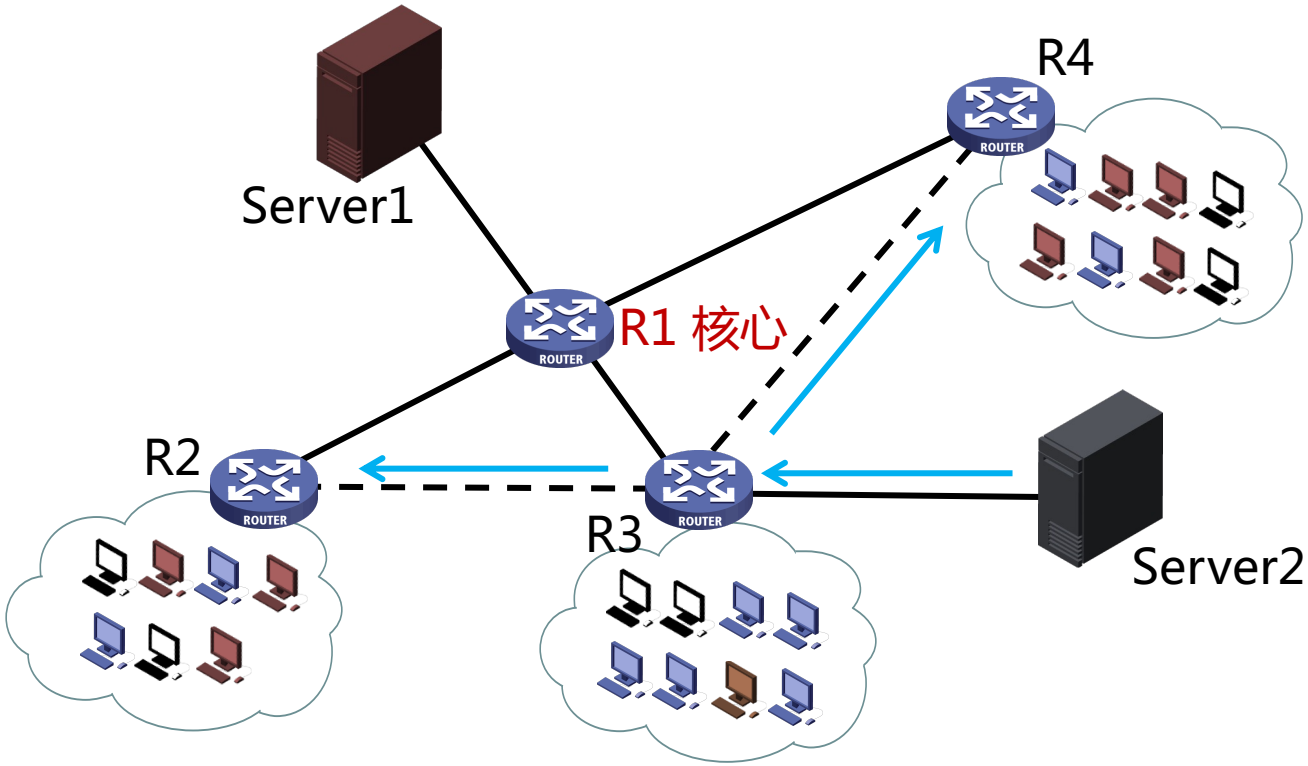
组播路由

- 协议无关多播-稀疏方式 PIM-SM
- 特定源组播 PIM-SSM
  - 组播接收者加入组播组时，即可以指定接收和拒绝来自特定组播源的组播流量
  - 组播源通过SPT（最短路径树）分发到接收路由器
- 思考：当存在多个组播源时，核心树的方式可能有什么问题？

组播路由

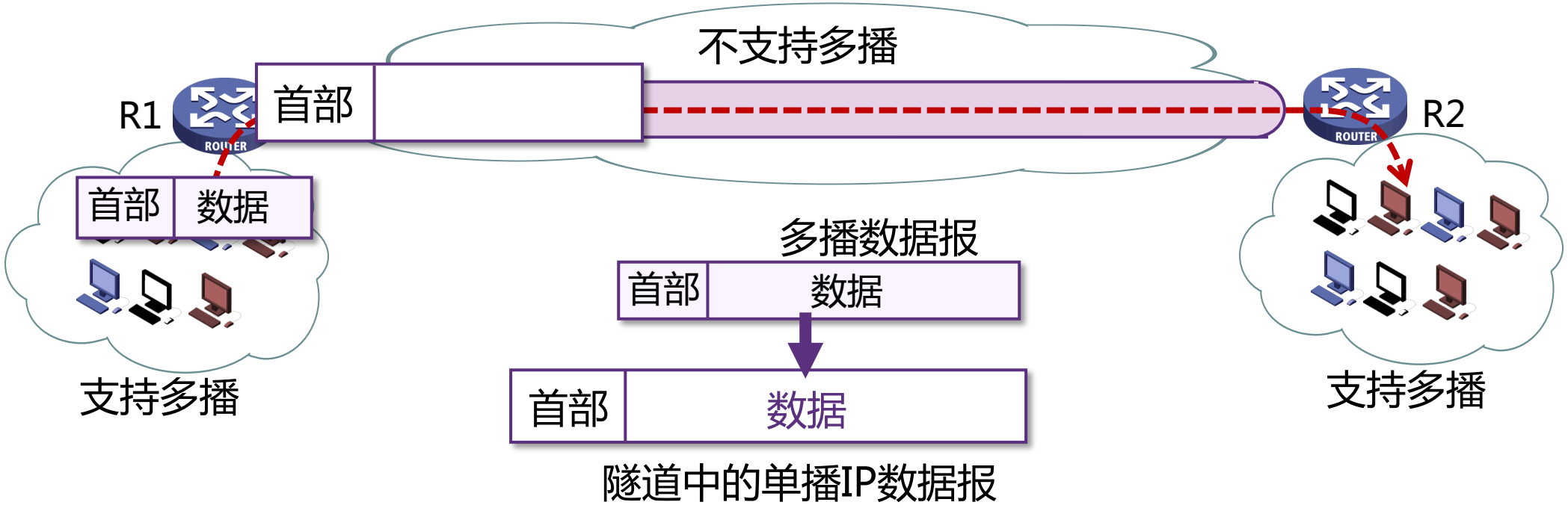
- 基于核心树（core-based trees）存在的问题：
  - 可能无法达到最优
  - 如果只有一个发送者，将发送者作为核心是最优的

例如，组播源Server2的数据，如果采用基于源点树的方法，可能到达目的主机所需跳数更少



组播路由

- Internet的组播主干（MBone）：可以转发多播流量的路由器和主机组成的网络
- 通常Internet是单播方式工作，如何使多播流量穿越单播网络？
- 隧道（tunneling）：可以使多播数据报穿越不支持多播的网络



组播的应用

- 音频/视频会议
- 共享电子白板
- 数据分发
- 实时数据组播：音频视频点播、网络收看体育比赛直播、炒股……
- 游戏与仿真：同时有大量参与者的网络游戏




OSPF-概述

- OSPF（Open Shortest Path First）开放最短路径优先协议于1989 年开发
- 采用分布式的链路状态算法
- OSPF协议的基本思想
  - 向本自治系统中所有路由器洪泛信息
  - 发送的信息就是与本路由器相邻的所有路由器的链路状态
  - 只有当链路状态发生变化时路由器才用洪泛法发送此信息

OSPF-链路状态

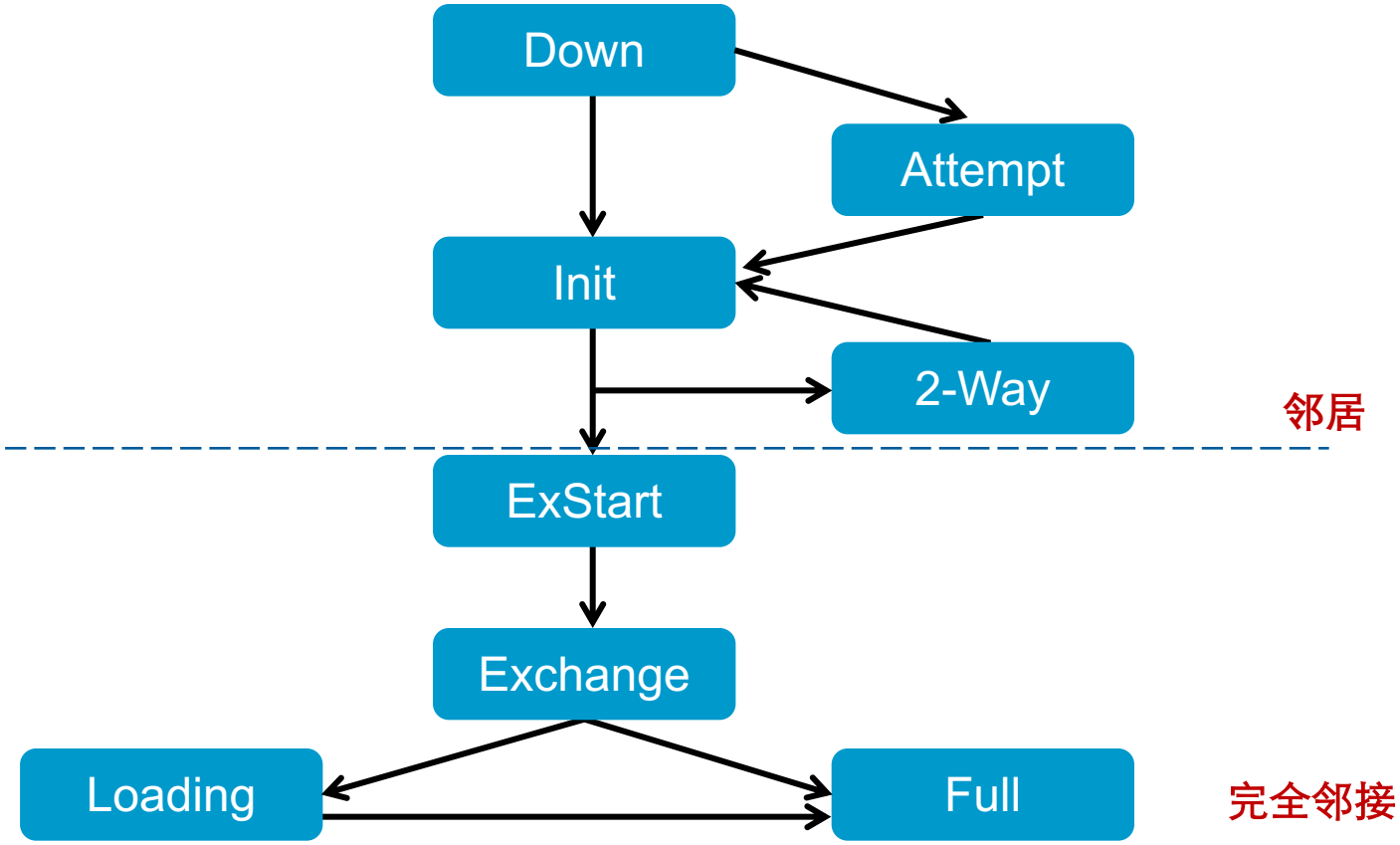
- “链路状态”就是说明本路由器都和哪些路由器相邻，以及该链路的“度量”(metric)
  - OSPF度量值一般包括费用、距离、时延、带宽等
- 由于各路由器之间频繁地交换链路状态信息，因此所有的路由器最终都能建立一个链路状态数据库LSDB
- 这个数据库实际上就是区域内的拓扑结构图，它在区域内是一致的（这称为链路状态数据库的同步）

OSPF-五种报文

- Hello 报文
  - 最常用的一种报文，用于发现、维护邻居关系
- 数据库描述（Database Description，DD）报文
  - 用于描述自己的LSDB
  - 内容包括LSDB 中每一条LSA 的Header 头部，对端路由器根据LSA Header 就可以判断出是否已有这条LSA
- 链路状态请求（LSA Request，LSR）报文
  - 用于请求缺少的LSA，内容包括所需要的LSA 的摘要
- 链路状态更新（LSA Update，LSU）报文
  - 用于向对端路由器发送所需要的LSA，内容是多条LSA（全部内容）的集合
- 链路状态确认（Link State Acknowledgment，LSACK）报文
  - 用来对接收到的LSU 报文进行确认

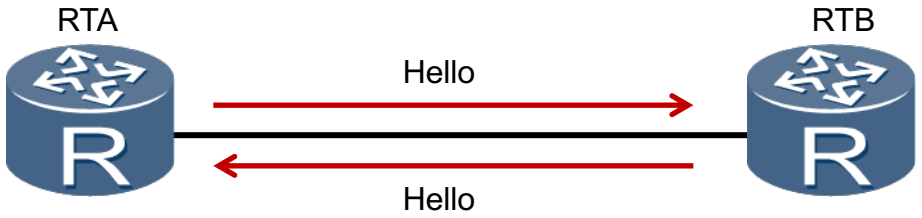
OSPF-邻居状态机

- 邻居
  - 端口连接到同一个网段
  - 由hello报文维护
- 完全邻接
  - 为了交换路由信息而形成的关系
  - 并非所有的邻居关系都可以成为邻接关系



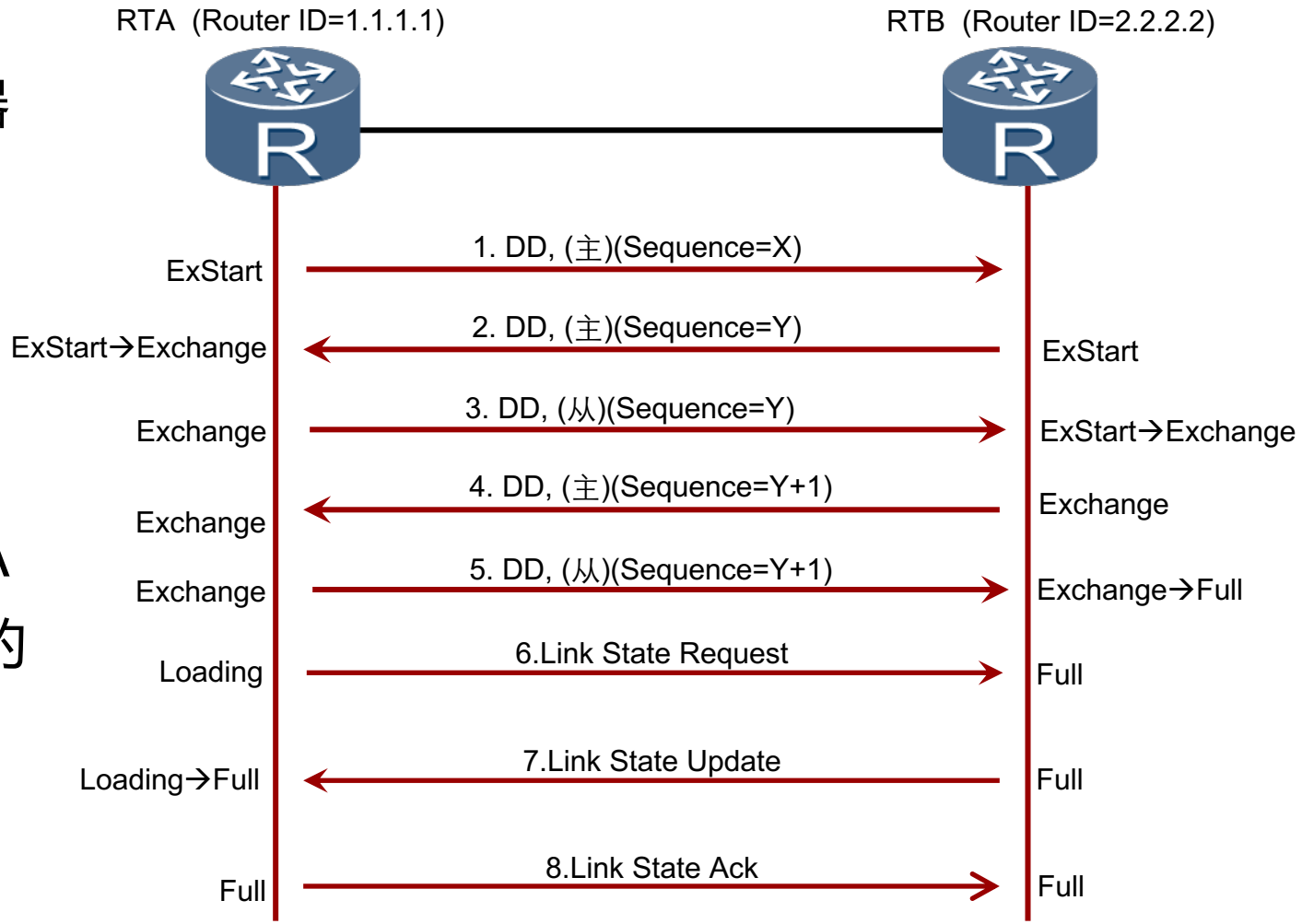
OSPF-报文交互

- 发现邻居
  - 使用HELLO报文
  - 检查HELLO报文中的参数，如果参数一致，形成邻居关系
  - 对于广播和NBMA 类型的网络，选举指定路由器（Designated Router，DR）和备份指定路由器（Backup Designated Router，BDR）



OSPF-报文交互

- 数据库同步
  - 使用DD报文来进行主从路由器的选举和数据库摘要信息的交互
  - DD报文包含LSA的头部信息，用来描述LSDB的摘要信息
- 建立完全邻接关系
  - LSR用于向对方请求所需的LSA
  - LSU用于向对方发送其所需要的LSA
  - LSACK用于向对方发送收到LSA的确认

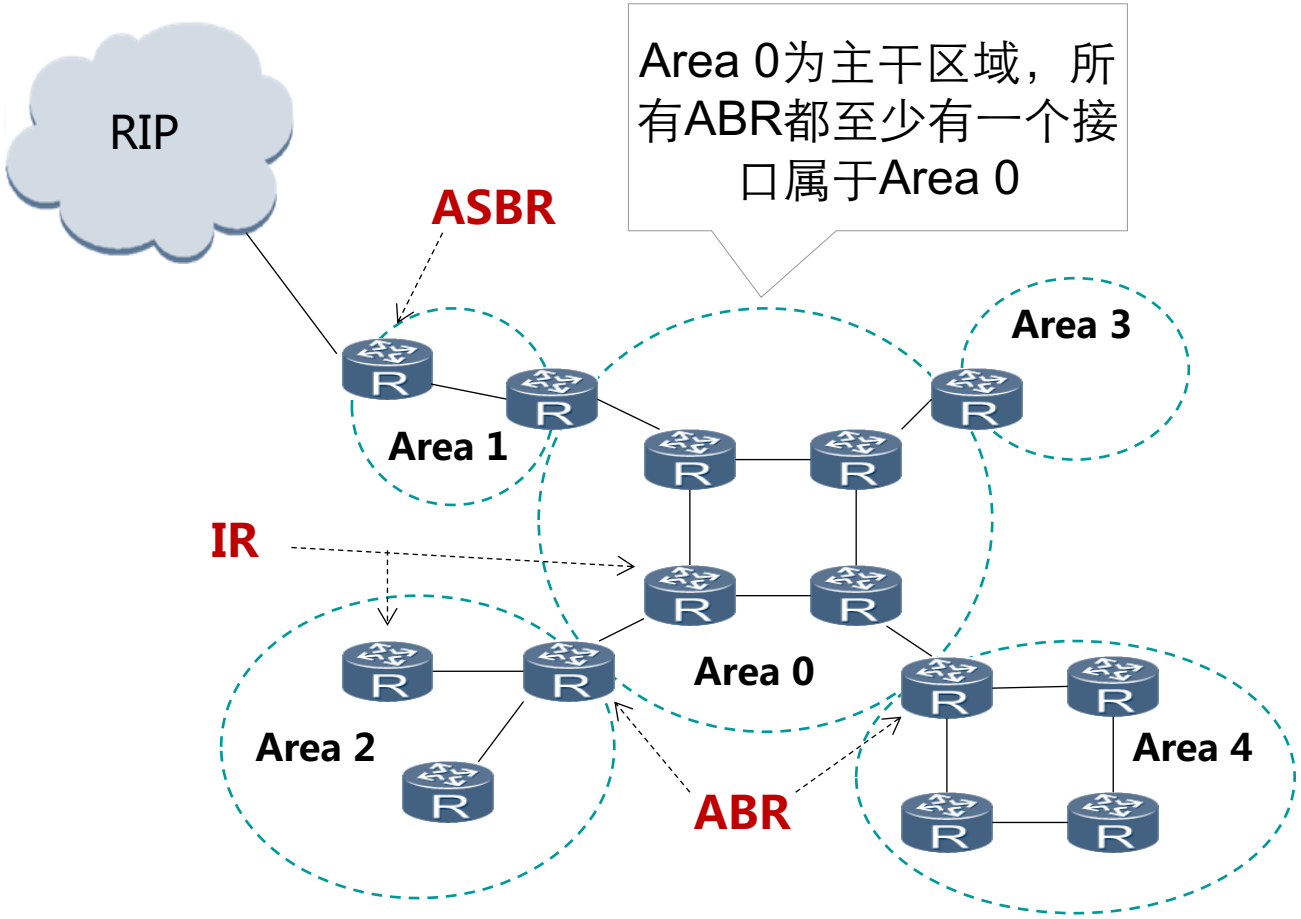


OSPF-区域的概念

- OSPF支持将一组网段组合在一起，称为一个区域
- 详细的描述拓扑结构的链路状态信息仅在区域内传递，区域间传递的是抽象的路由信息
- 使用层次结构的区域划分，上层的区域叫做主干区域(backbone area)，其他区域都必须与主干区域相连
- 非主干区域之间不允许直接发布区域间路由信息
- 区域也不能太大，在一个区域内的路由器最好不超过 200 个
- 划分区域可以缩小LSDB规模，减少网络流量

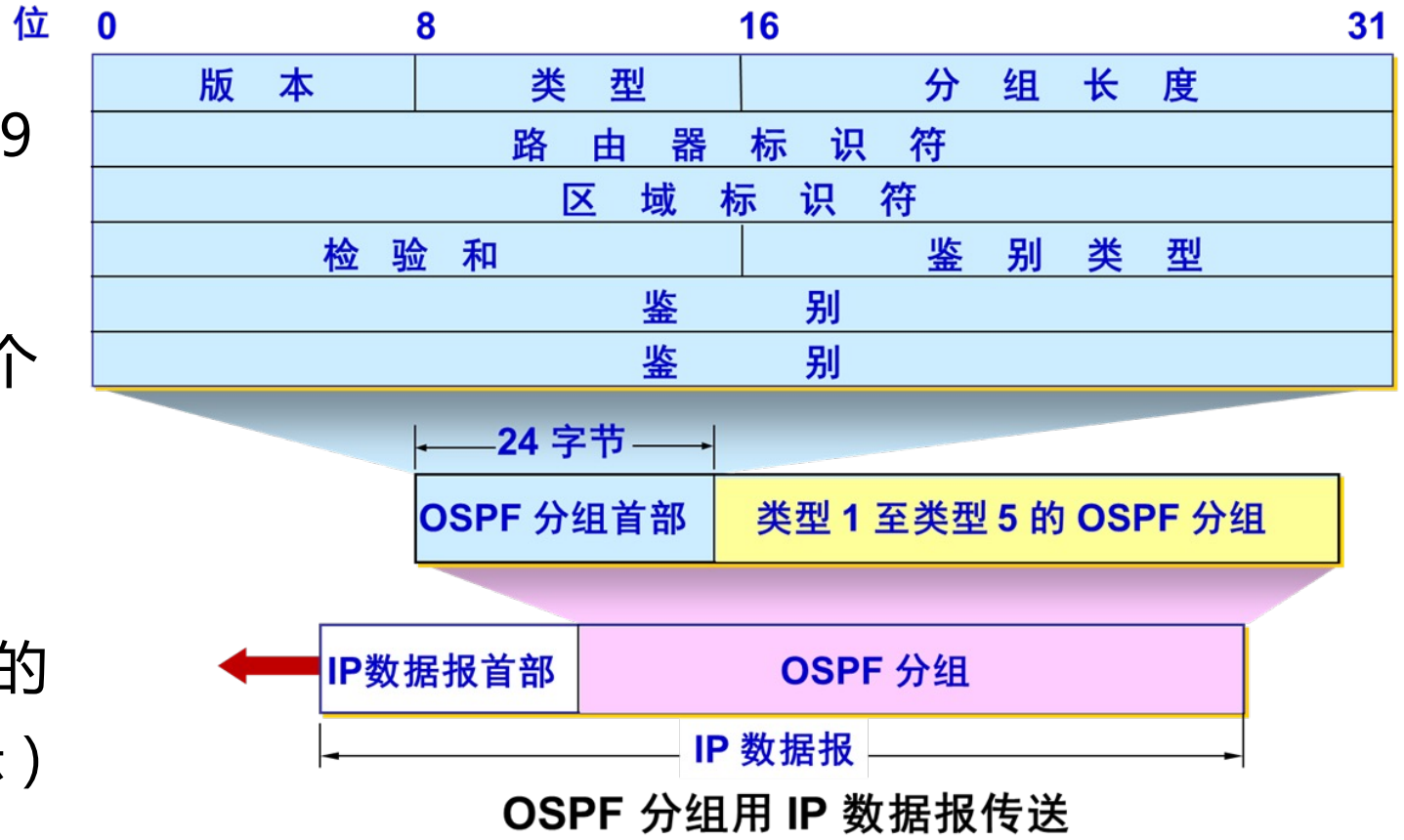
OSPF-区域的概念

- OSPF的区域
  - 主干区域
  - 非主干区域
- 路由器角色
  - 内部路由器(Internal Router，IR)
  - 区域边界路由器(Area Bounder Router，ABR)
  - 自治系统边界路由器(AS Bounder Router, ASBR)



OSPF-报文格式

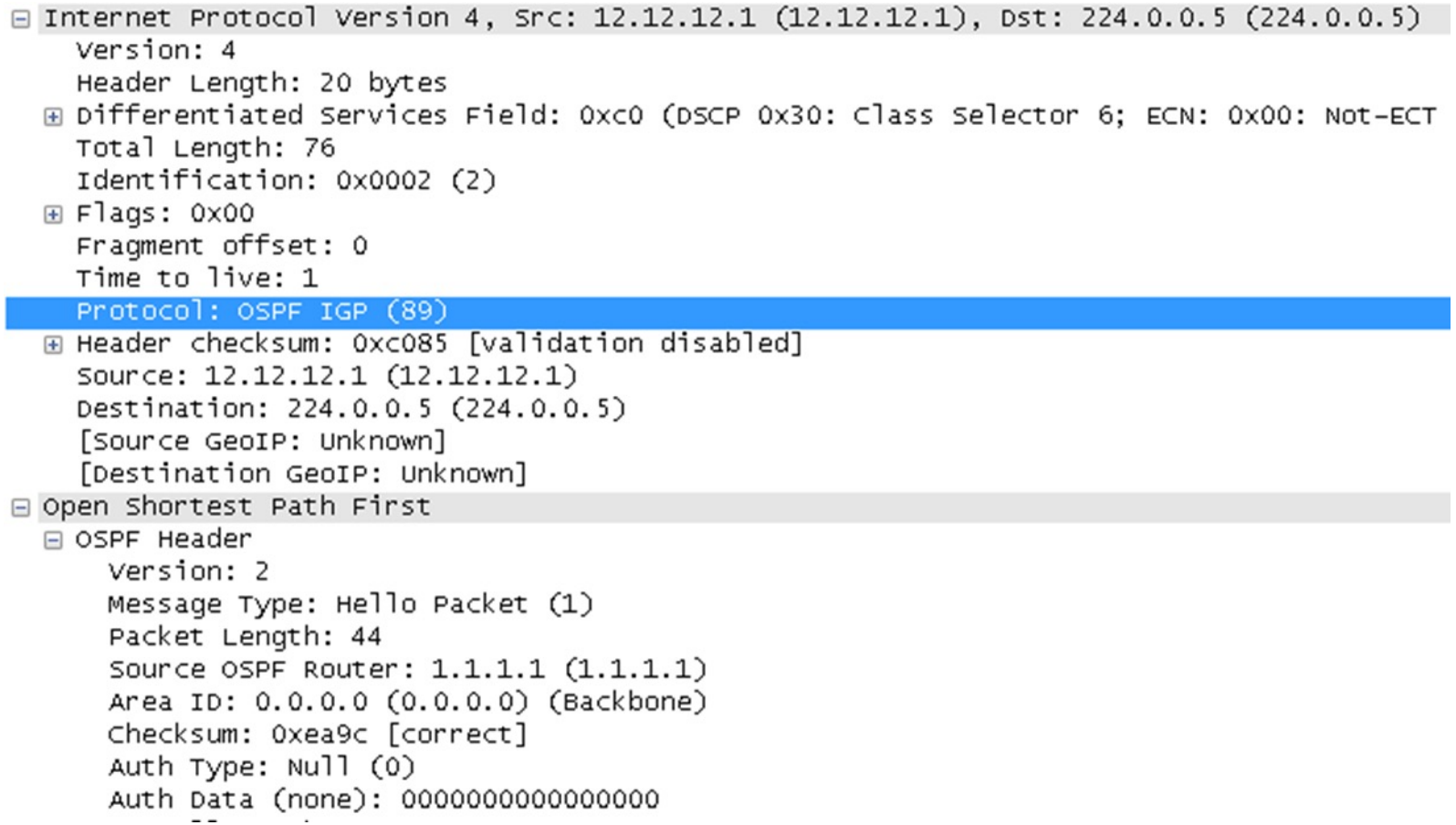
- 协议封装
  - 运行于IP协议之上，协议号89
- 路由器标识符
  - 一个32位的值，唯一标识一个自治系统内的路由器
- 区域标识符
  - 每一个区域都有一个 32 位的标识符（用点分十进制表示）
  - 主干区域为0.0.0.0



OSPF 分组用 IP 数据报传送

OSPF-报文格式

- 请观察：
  - 这是什么报文？
  - 这个报文在什么区域？
  - 采用什么认证？



OSPF-小结

- OSPF的特点
  - 支持无类域间路由（CIDR）
  - 无路由自环
  - 收敛速度快
  - 使用IP组播收发协议数据
  - 支持多条等值路由
  - 支持协议报文的认证

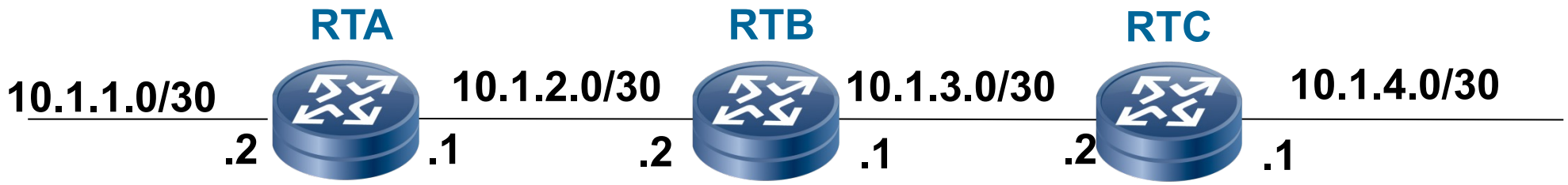
- 请思考：
  - 为什么OSPF协议适合大型网络？
  - OSPF会不会存在“坏消息传得比较慢”的问题？

RIP-概述

- 路由选择协议RIP（ Routing Information Protocol）是基于距离矢量算法的协议
- 使用跳数衡量到达目的网络的距离
  - RIP 认为一个好的路由就是它通过的路由器的数目少，即“距离短”
  - RIP 允许一条路径最多只能包含 15 个路由器
- RIP协议的基本思想
  - 仅和相邻路由器交换信息
  - 路由器交换的内容是自己的路由表
  - 周期性更新：30s

RIP-工作过程

- 初始化



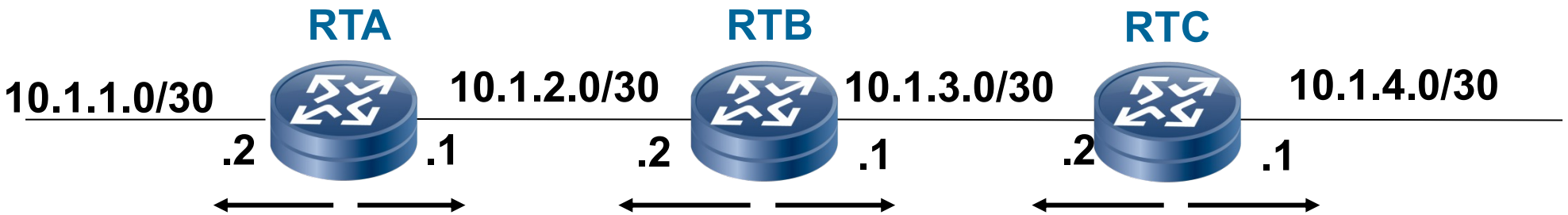
目标网络	下一跳	距离
10.1.1.0	– –	0
10.1.2.0	– –	0

目标网络	下一跳	距离
10.1.2.0	– –	0
10.1.3.0	– –	0

目标网络	下一跳	距离
10.1.3.0	– –	0
10.1.4.0	– –	0

RIP-工作过程

- 周期性更新



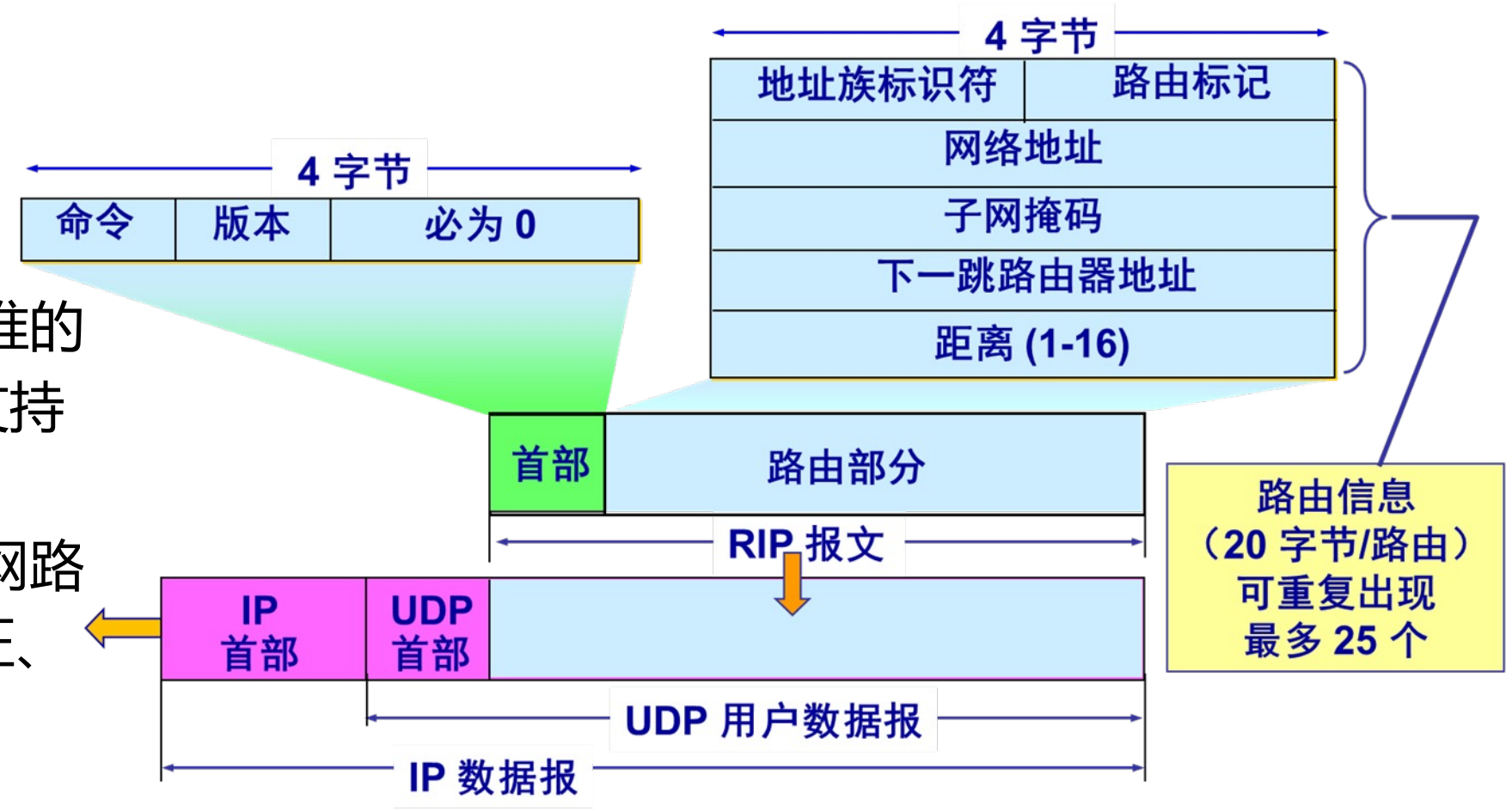
目标网络	下一跳	距离
10.1.1.0	– –	0
10.1.2.0	– –	0
10.1.3.0	10.1.2.2	1
10.1.4.0	10.1.2.2	2

目标网络	下一跳	距离
10.1.2.0	– –	0
10.1.3.0	– –	0
10.1.1.0	10.1.2.1	1
10.1.4.0	10.1.3.2	1

目标网络	下一跳	距离
10.1.3.0	– –	0
10.1.4.0	– –	0
10.1.2.0	10.1.3.1	1
10.1.1.0	10.1.2.1	2

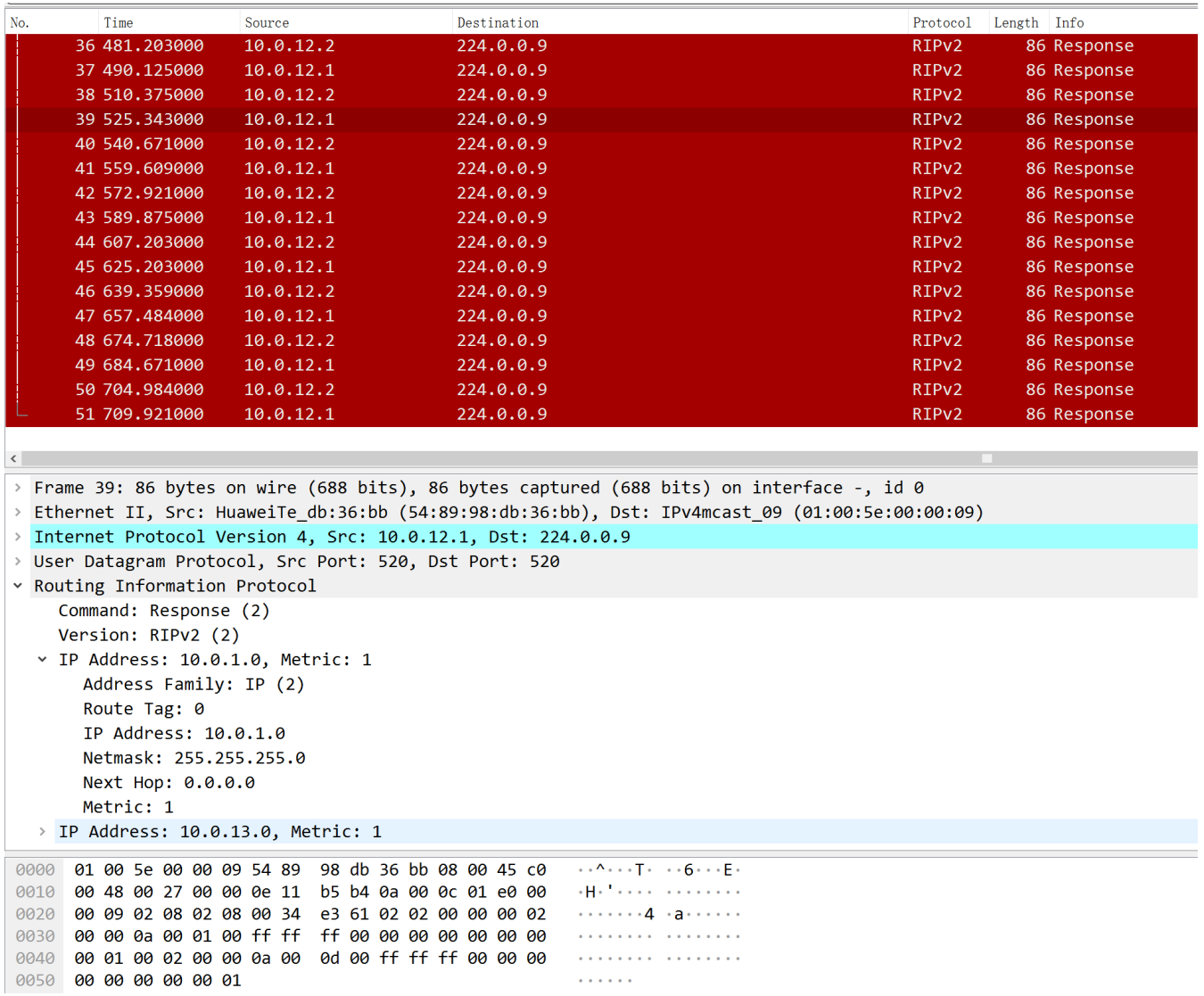
RIP-报文格式(V2)

- 协议封装
  - UDP，520
- 版本演进
  - V1：使用标准的IP地址，不支持子网路由
  - V2：支持子网路由、身份验证、多播



RIP-报文格式

- 请观察：
  - 这个报文是RIP版本多少？
  - RIP的周期更新如何实现？
  - 实现时是什么时候把跳数加1？
  - 为什么RIP没有确认机制？



No.	Time	Source	Destination	Protocol	Length	Info
36	481.203000	10.0.12.2	224.0.0.9	RIPv2	86	Response
37	490.125000	10.0.12.1	224.0.0.9	RIPv2	86	Response
38	510.375000	10.0.12.2	224.0.0.9	RIPv2	86	Response
39	525.343000	10.0.12.1	224.0.0.9	RIPv2	86	Response
40	540.671000	10.0.12.2	224.0.0.9	RIPv2	86	Response
41	559.609000	10.0.12.1	224.0.0.9	RIPv2	86	Response
42	572.921000	10.0.12.2	224.0.0.9	RIPv2	86	Response
43	589.875000	10.0.12.1	224.0.0.9	RIPv2	86	Response
44	607.203000	10.0.12.2	224.0.0.9	RIPv2	86	Response
45	625.203000	10.0.12.1	224.0.0.9	RIPv2	86	Response
46	639.359000	10.0.12.2	224.0.0.9	RIPv2	86	Response
47	657.484000	10.0.12.1	224.0.0.9	RIPv2	86	Response
48	674.718000	10.0.12.2	224.0.0.9	RIPv2	86	Response
49	684.671000	10.0.12.1	224.0.0.9	RIPv2	86	Response
50	704.984000	10.0.12.2	224.0.0.9	RIPv2	86	Response
51	709.921000	10.0.12.1	224.0.0.9	RIPv2	86	Response

```
> Frame 39: 86 bytes on wire (688 bits), 86 bytes captured (688 bits) on interface -, id 0
> Ethernet II, Src: HuaweiTe_db:36:bb (54:89:98:db:36:bb), Dst: IPv4mcast_09 (01:00:5e:00:00:09)
> Internet Protocol Version 4, Src: 10.0.12.1, Dst: 224.0.0.9
> User Datagram Protocol, Src Port: 520, Dst Port: 520
v Routing Information Protocol
    Command: Response (2)
    Version: RIPv2 (2)
  v IP Address: 10.0.1.0, Metric: 1
      Address Family: IP (2)
      Route Tag: 0
      IP Address: 10.0.1.0
      Netmask: 255.255.255.0
      Next Hop: 0.0.0.0
      Metric: 1
  > IP Address: 10.0.13.0, Metric: 1
```

```
0000  01 00 5e 00 00 09 54 89  98 db 36 bb 08 00 45 c0   ··^···T· ··6···E·
0010  00 48 00 27 00 00 0e 11  b5 b4 0a 00 0c 01 e0 00   ·H·'···· ········
0020  00 09 02 08 02 08 00 34  e3 61 02 02 00 00 00 02   ·······4 ·a······
0030  00 00 0a 00 01 00 ff ff  ff 00 00 00 00 00 00 00   ········ ········
0040  00 01 00 02 00 00 0a 00  0d 00 ff ff ff 00 00 00   ········ ········
0050  00 00 00 00 00 01                                 ·····
```

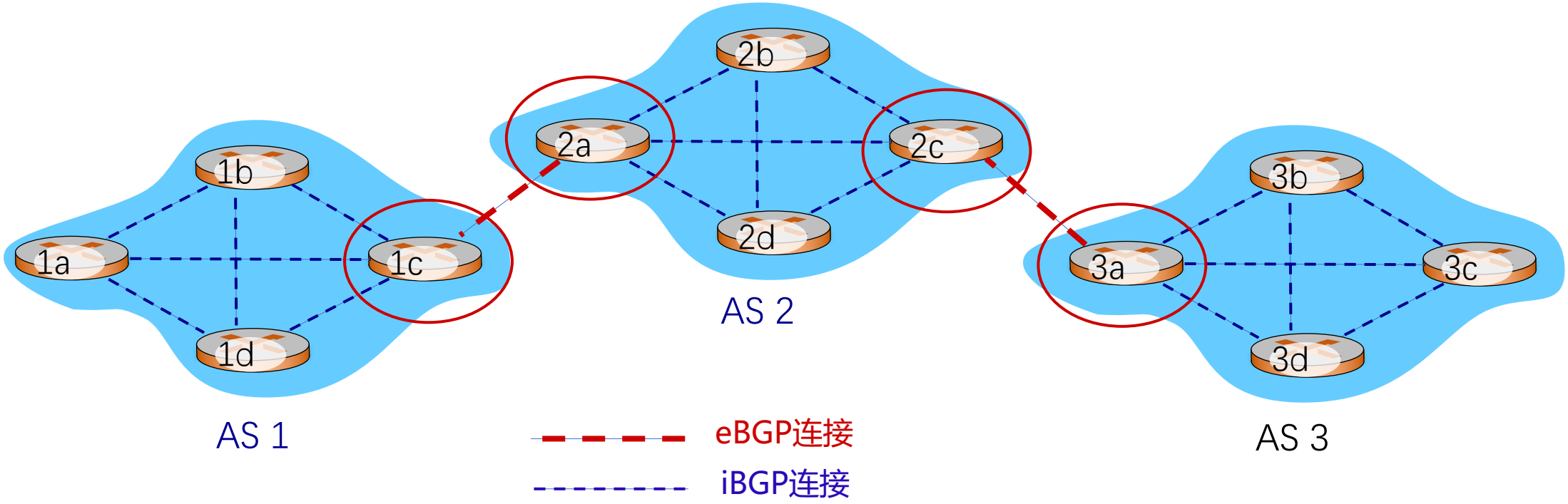
RIP-小结

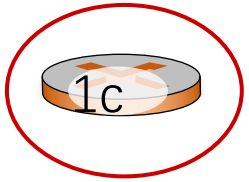
- RIP协议的特点
  - 算法简单，易于实现
  - 收敛慢
  - 需要交换的信息量较大
- RIP协议的适用场合
  - 中小型网络
- RIP协议的防环路机制
  - 触发更新
  - 毒性反转
  - 水平分割
  - 其他

BGP-外部网关路由协议

- 路由协议
  - 内部网关协议 IGP： 有 RIP 和、OSPF、ISIS 等多种具体的协议
  - 外部网关协议 EGP：目前使用的协议就是 BGP
- 边界网关协议BGP (Border Gateway Protocol)
  - 目前互联网中唯一实际运行的自治域间的路由协议
- BGP功能
  - eBGP：从相邻的AS获得网络可达信息
  - iBGP： 将网络可达信息传播给AS内的路由器
  - 基于网络可达信息和策略决定到其他网络的“最优”路由

eBGP&iBGP连接

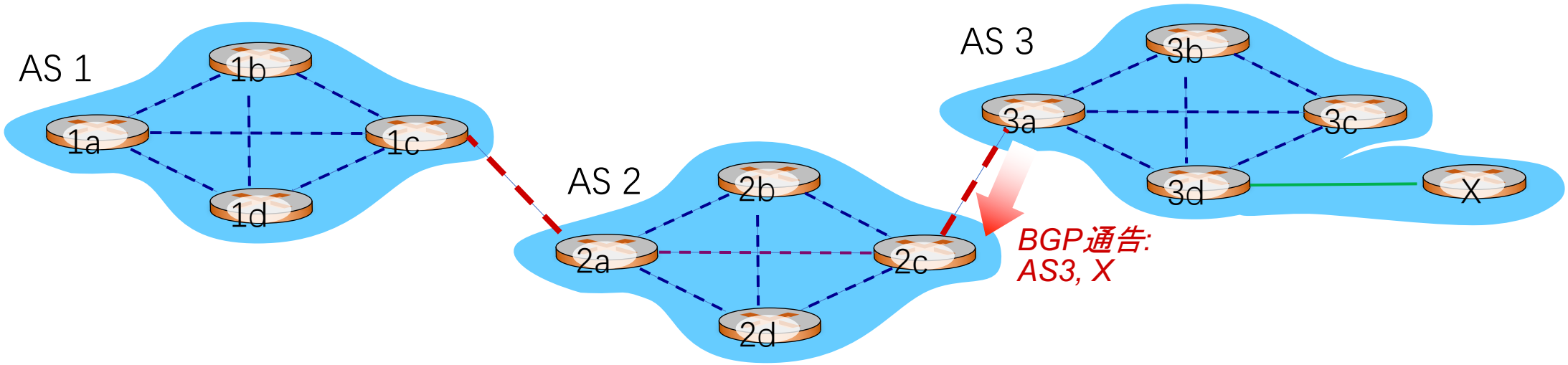




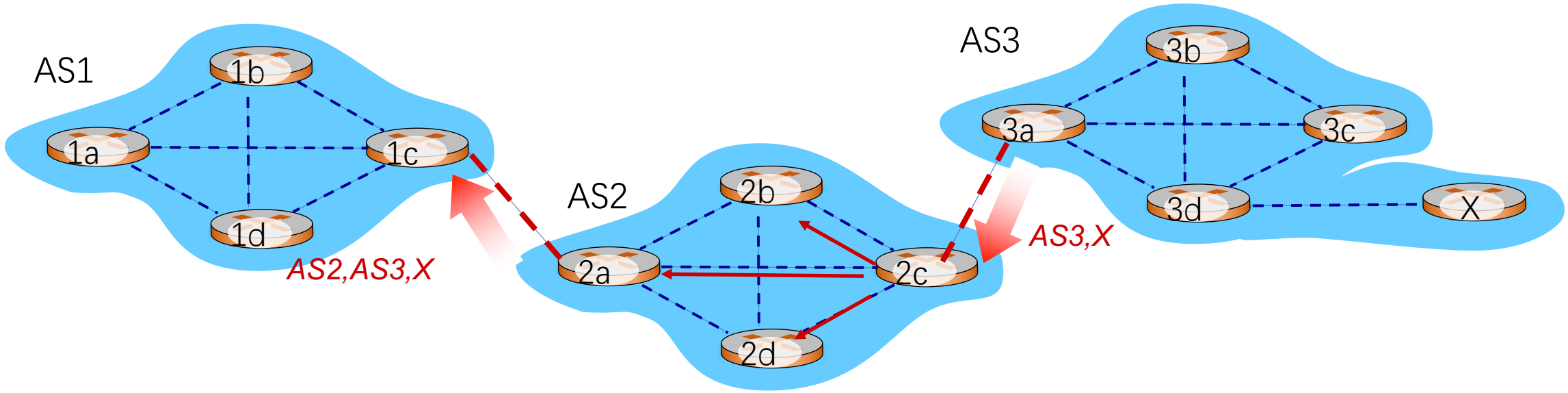
运行eBGP, iBGP协议和IGP的路由器

BGP基础

- BGP会话: 两个BGP路由器通过TCP连接交换BGP报文
  - 通告到不同网络前缀的路径，即路径向量协议
- 例：当AS3的路由器3a向AS2的路由器2c通告路径AS3, X时AS3向AS2承诺它会向X转发数据包

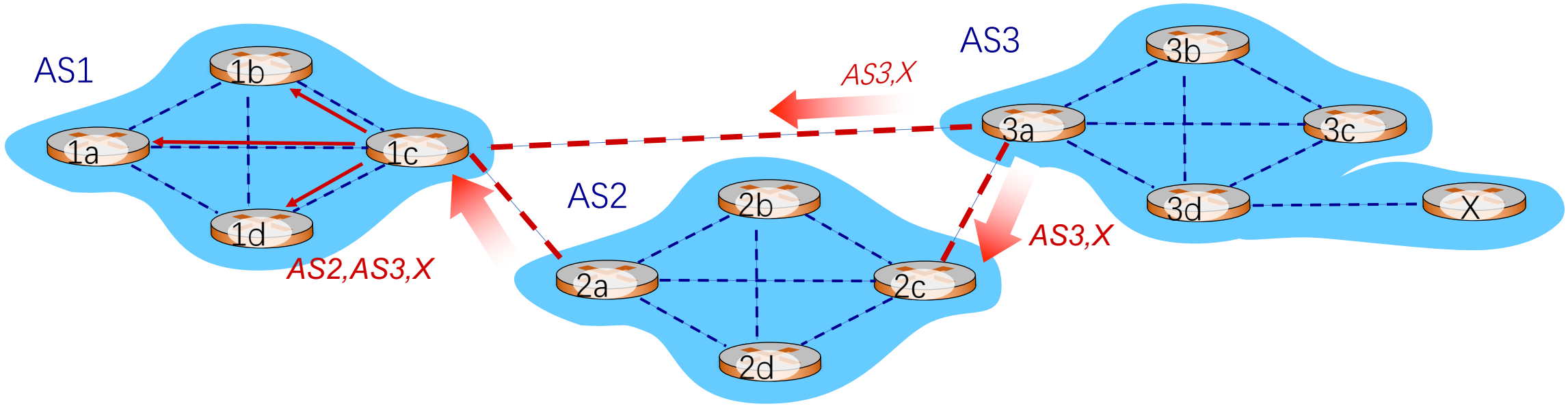


BGP路径通告



- AS2的路由器2c从AS3的路由器3a接收到路径AS3, X
- 根据AS2的策略，AS2的路由器2c接受路径AS3, X，通过iBGP传播给AS2的所有路由器
- 根据AS2策略，AS2的路由器2a通过eBGP向AS1的路由器1c通告从AS3的路由器3a接收到路径AS2, AS3, X

BGP路径通告



路由器可能会学到多条到目的网络的路径:

- AS1的路由器1c从2a学到路径*AS2, AS3, X*
- AS1的路由器1c从3a学到路径*AS3, X*
- 由策略，AS1路由器1c可能选择路径*AS3, X,* 并在AS1中通过iBGP通告路径

BGP协议的特点

- BGP 协议交换路由信息的结点数量级是自治系统数的量级
- 每一个自治系统边界路由器的数目是很少的
- 在 BGP 刚刚运行时，BGP 的邻站是交换整个的 BGP 路由表；以后只需要在发生变化时更新有变化的部分
- BGP为每个AS提供：
  - 从邻居AS获取网络可达信息（eBGP协议）
  - 传播可达信息给所有的域内路由器（iBGP协议）
  - 根据“可达信息”和“策略”决定路由

BGP报文

- BGP通过TCP的179端口交换报文
- BGP报文包括
  - Open报文：用于建立BGP对等体（peer）之间的会话连接，协商BGP参数（该过程需要认证）
  - Update报文：用于在对等体之间交换路由信息
  - Keepalive报文：用于保持BGP会话连接
  - Notification报文：用于差错报告和关闭BGP连接

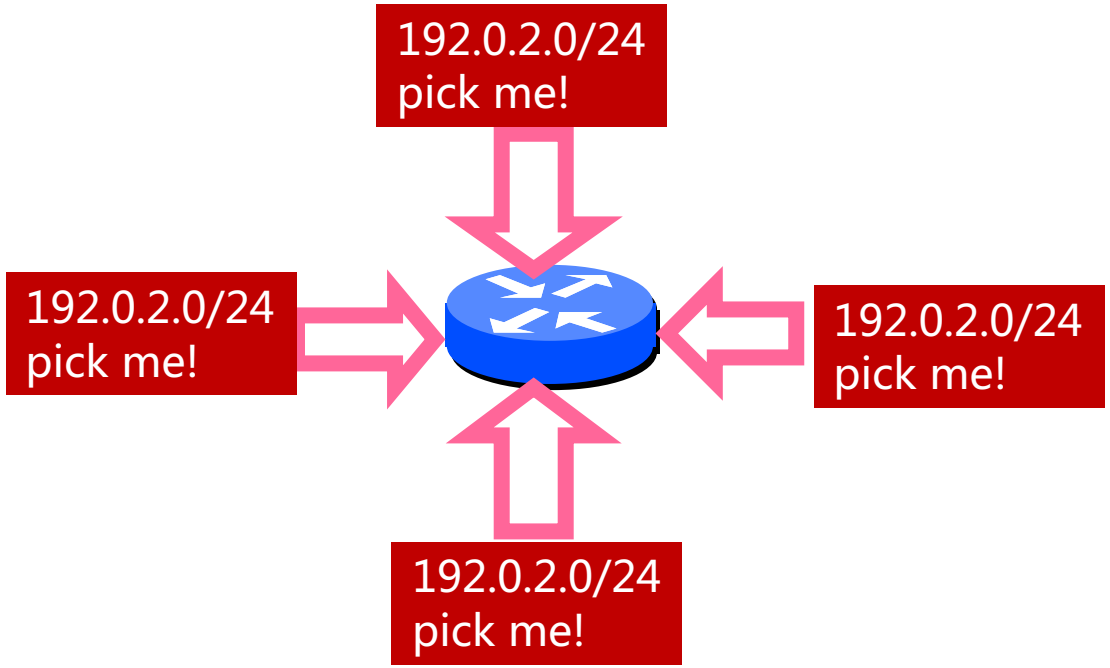
BGP路径属性

- 发布的前缀信息包括BGP属性(BGP attributes)
  - 路由“route”=prefix + attributes
- 两个重要属性
  - AS路径（AS-PATH）：IP前缀通过经过的所有AS号

    如：AS 67, AS 17
  - 下一跳（NEXT-HOP）：说明路由信息对应的下一跳IP地址
- 网关路由器接收到路由通告时，通过既定策略采纳或拒绝(accept/decline)

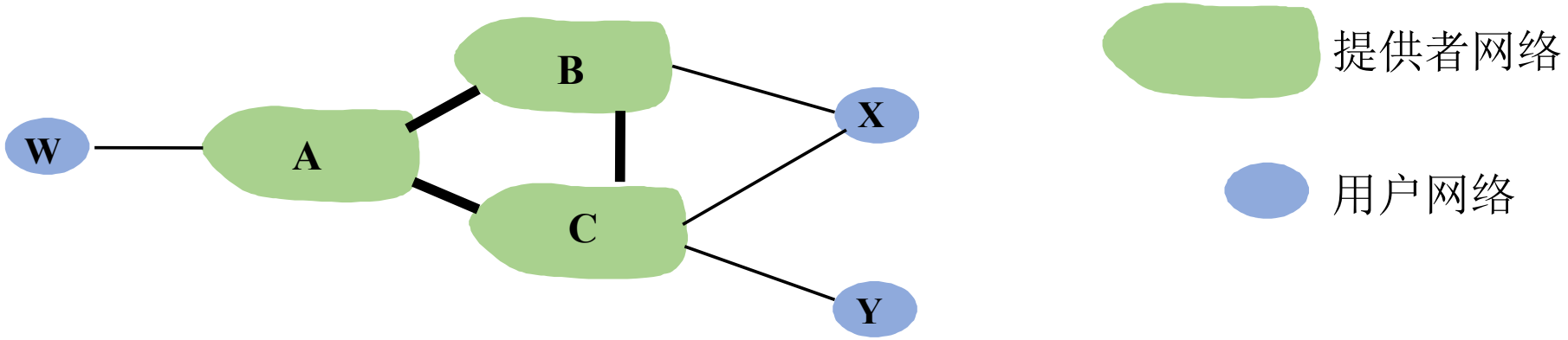
BGP 路由选择

- 路由器可能从多个对等体收到针对同一目的IP的路由
- 需要选择一条最佳路由
- 选择规则：自上向下，依次排序
  - 本地偏好值属性：政策决策
  - 最短的AS-PATH
  - 最近的NEXT-HOP路由器
  - 附加标准…
  - 最低路由器ID



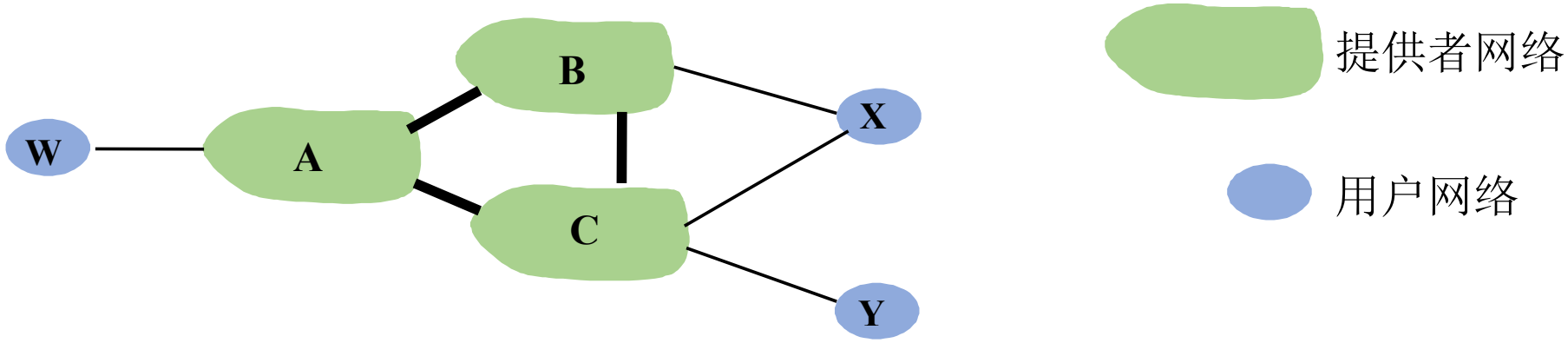
BGP路由策略

- 路由器使用策略决定接受或拒绝接收到的路由通告
- 路由器也会基于策略决定是否向其他相邻AS通告路径信息



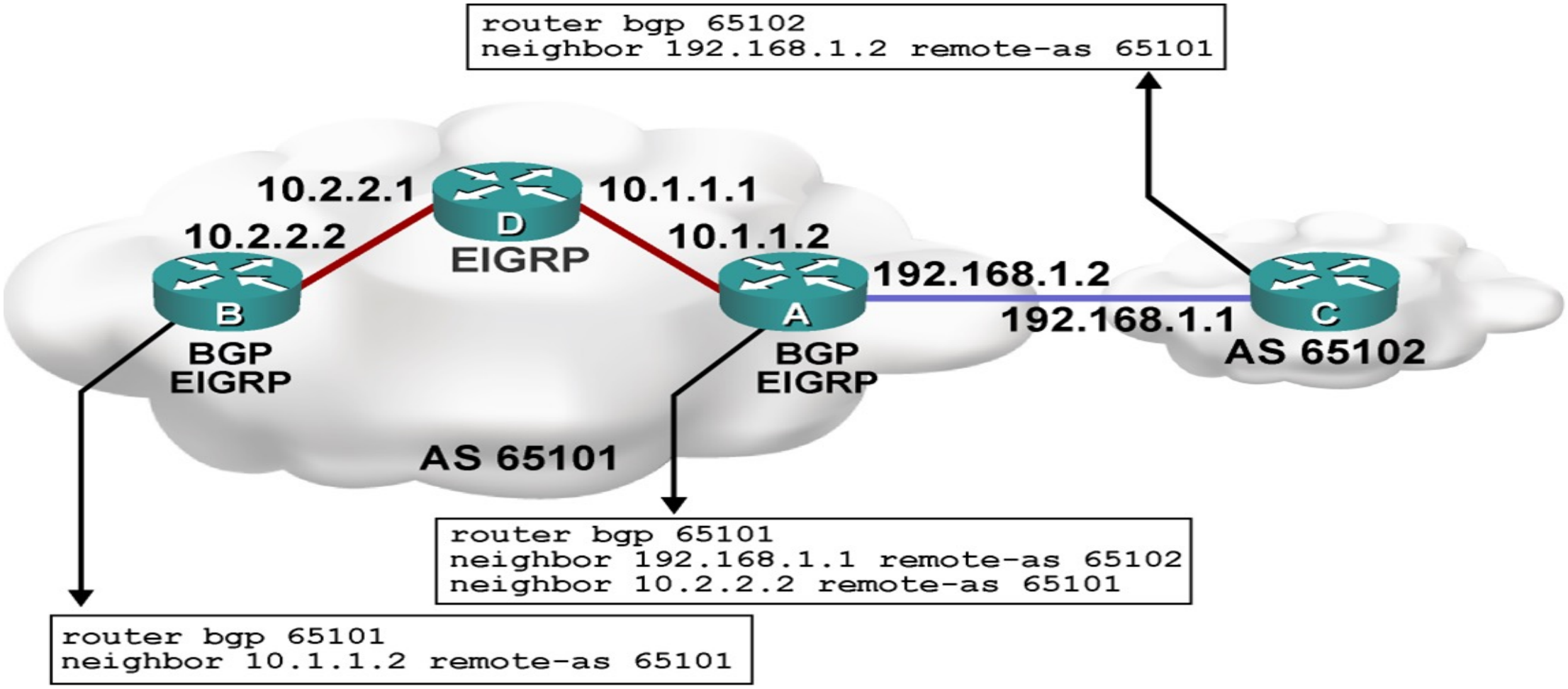
- 例如：X连接到两个提供者网络（dual-homed）
  - X为用户网络，X不希望从B到C的数据包经过X
  - X则不向B通告到C的路由

BGP路由策略



- A向B通告路径*AW*
- B通告到目的W的路径为*BAW*
- B是否向C通告路径*BAW*？
  - 由于W和C都不是B的用户，B要迫使C通过A路由到W
  - B只路由来自于或到达其用户的数据包

BGP配置实例

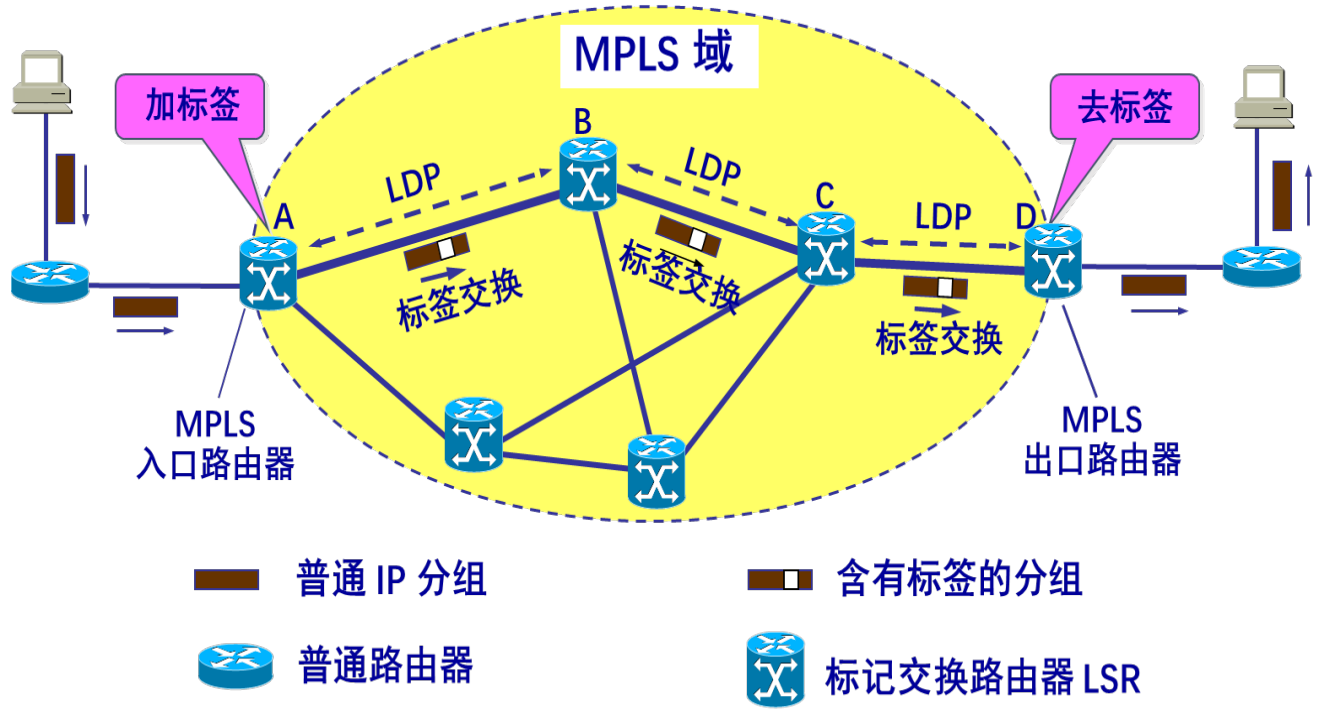

router bgp 65102
neighbor 192.168.1.2 remote-as 65101
10.2.2.1
10.1.1.1
10.2.2.2
D
EIGRP
10.1.1.2
192.168.1.2
B
A
C
192.168.1.1
BGP
EIGRP
BGP
EIGRP
AS 65102
AS 65101
router bgp 65101
neighbor 192.168.1.1 remote-as 65102
neighbor 10.2.2.2 remote-as 65101
router bgp 65101
neighbor 10.1.1.2 remote-as 65101

标签交换和MPLS-概述

- MPLS (MultiProtocol Label Switching)全称是多协议标签交换
  - 多协议表示在 MPLS 的上层可以采用多种协议，例如：IP，IPv6、IPX
  - 标签是指每个分组被分配一个标签，路由器根据该标签对分组进行转发
  - 交换是指标签的交换，MPLS 报文交换和转发是基于标签的
- MPLS 设计初衷为了提升查找速度
- MPLS 主要有以下三个方面的应用
  - 面向连接的服务质量管理
  - 流量工程，平衡网络负载
  - 虚拟专用网VPN

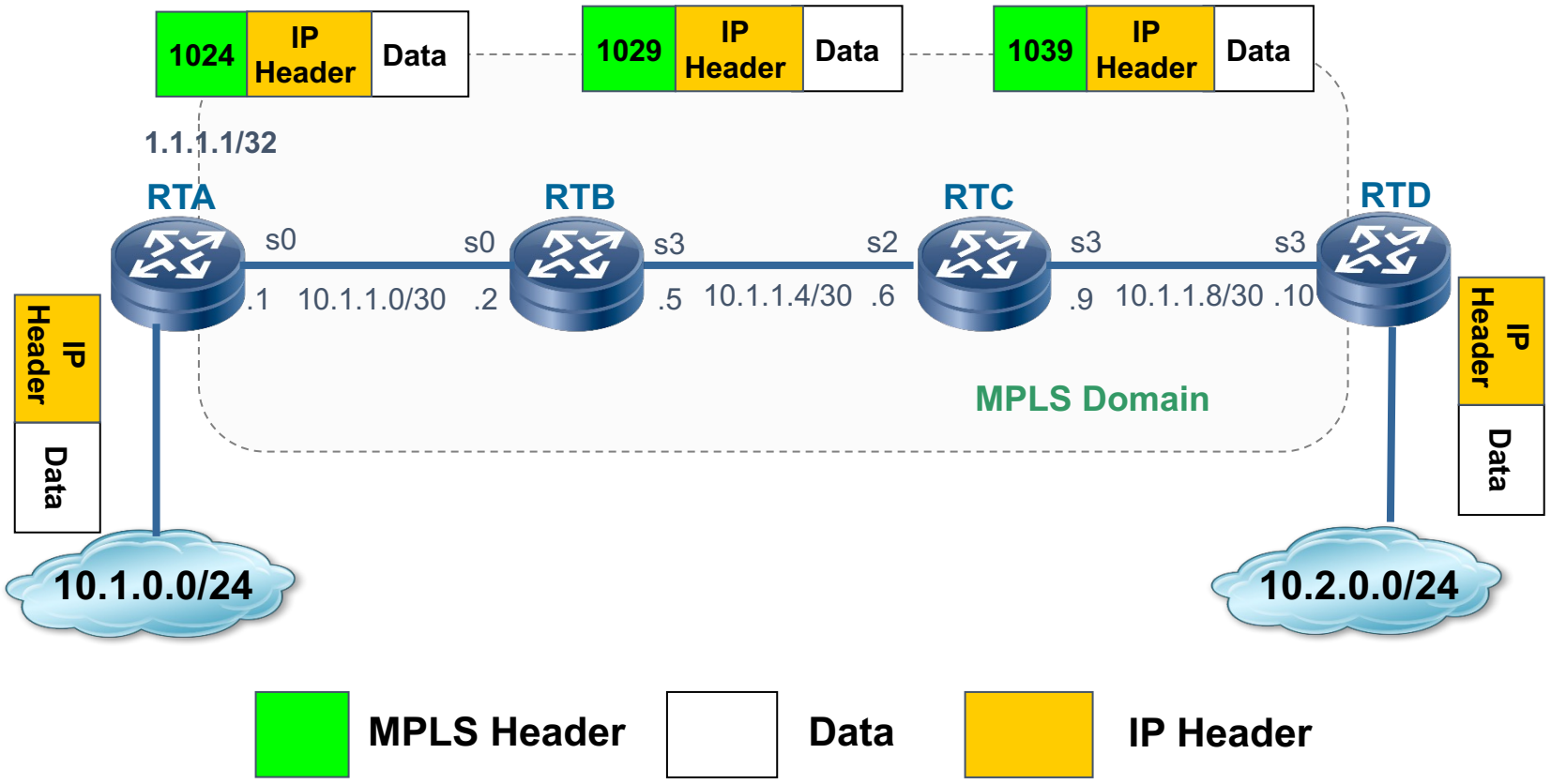
标签交换和MPLS

- 标签交换路由器LSR
  - 支持MPLS的路由器
  - 具备标签交换、路由选择两种功能
- MPLS 域
  - 所有相邻的支持MPLS技术的路由器构成的区域
- 标签分配协议LDP
  - 用来在LSR之间建立LDP 会话并交换Label/FEC映射信息



标签交换和MPLS-工作过程

- 加标签
  - 在 MPLS 域的入口处，给每一个 IP 数据报加上 标签，然后对加上标记的 IP 数据报用硬件进行转发
- 标签交换
  - 采用硬件技术对加上标记的 IP 数据报进行转发称为标签交换
- 去标签
  - 当分组离开 MPLS 域时，MPLS 出口路由器把分组的标签去除。后续按照一般IP分组的转发方法进行转发



标签交换和MPLS-标签

- 标签仅仅在两个LSR 之间才有意义
- LSR会维护一张转发表

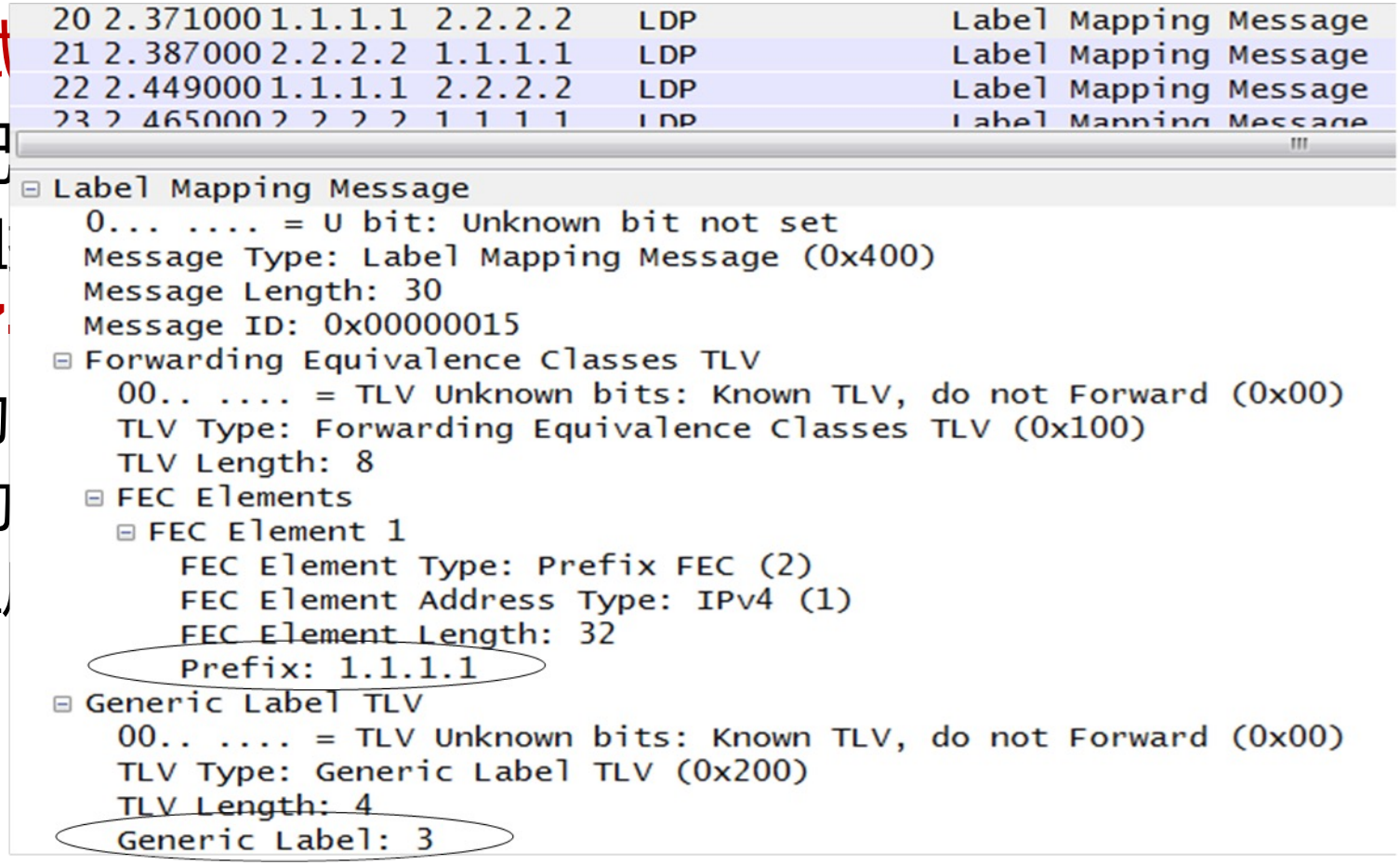
转发表

入接口	入标记	出接口	出标记
0	3	1	1

表项含义：从入接口 0 收到一个入标记为 3 的IP 数据报，转发时，应当把该IP数据报从出接口 1 转发出去，同时把标记对换为 1。

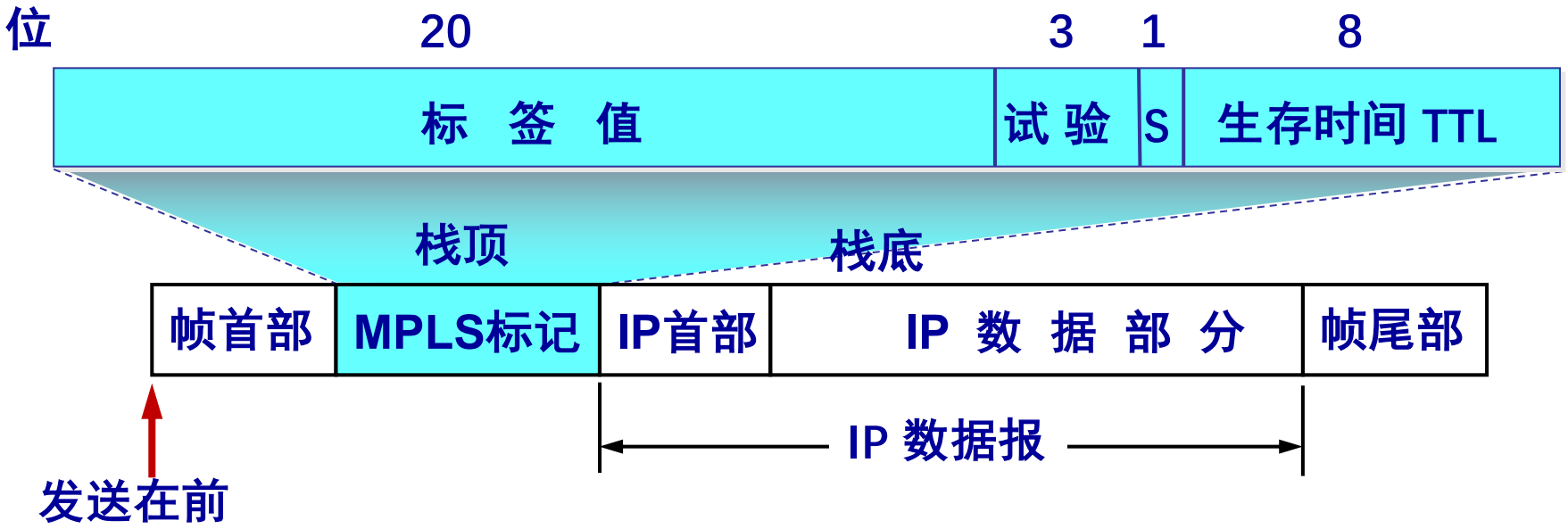
标签交换和MPLS-转发等价类

- 路由器按照同样方式
- 通常对一个FEC分配
  - 属于某特定组的组
  - **目的IP地址匹配了**
  - 有相同QoS策略的
  - 属于同一个VPN的
  - 报文的目的IP地址
  - 其他

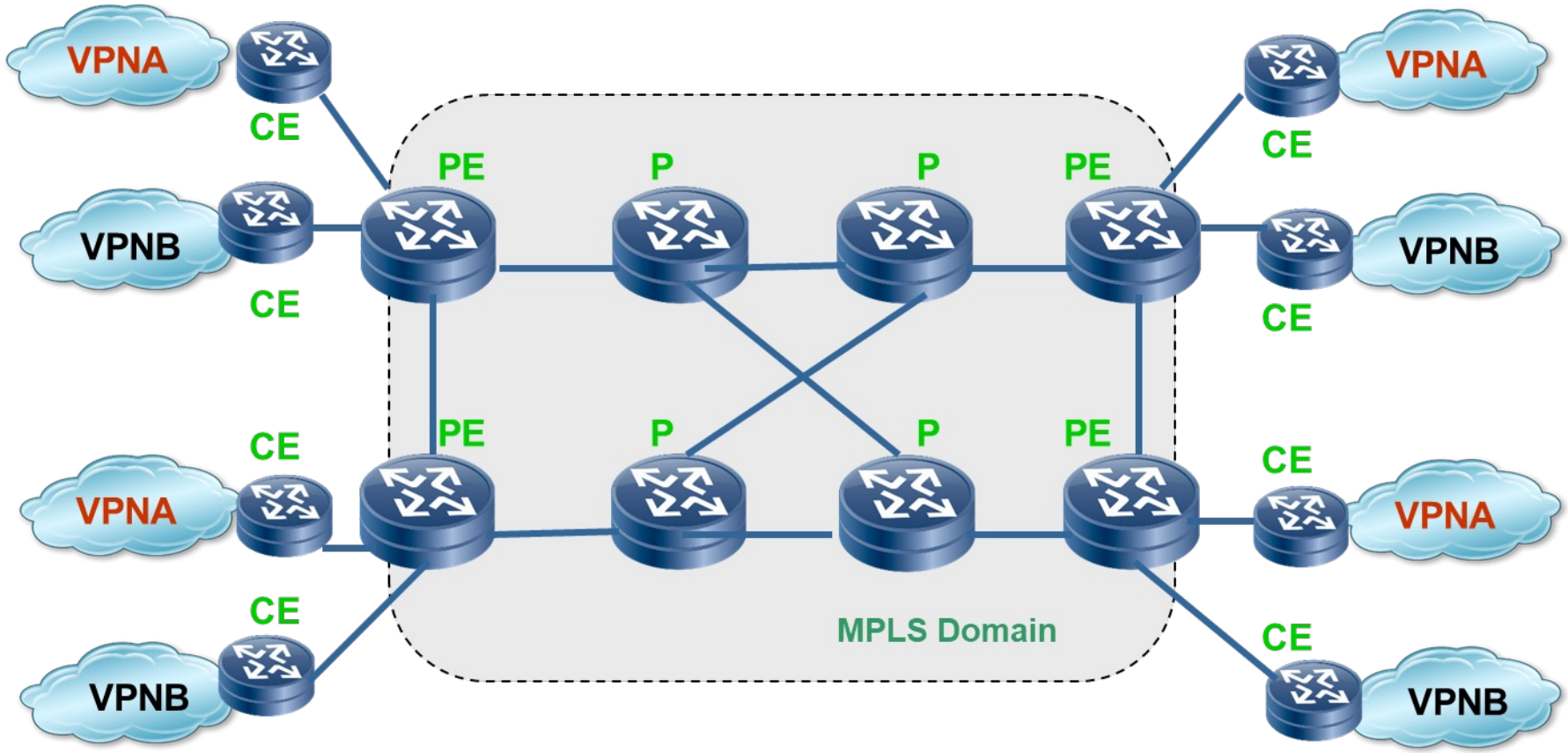


标签交换和MPLS-报文结构

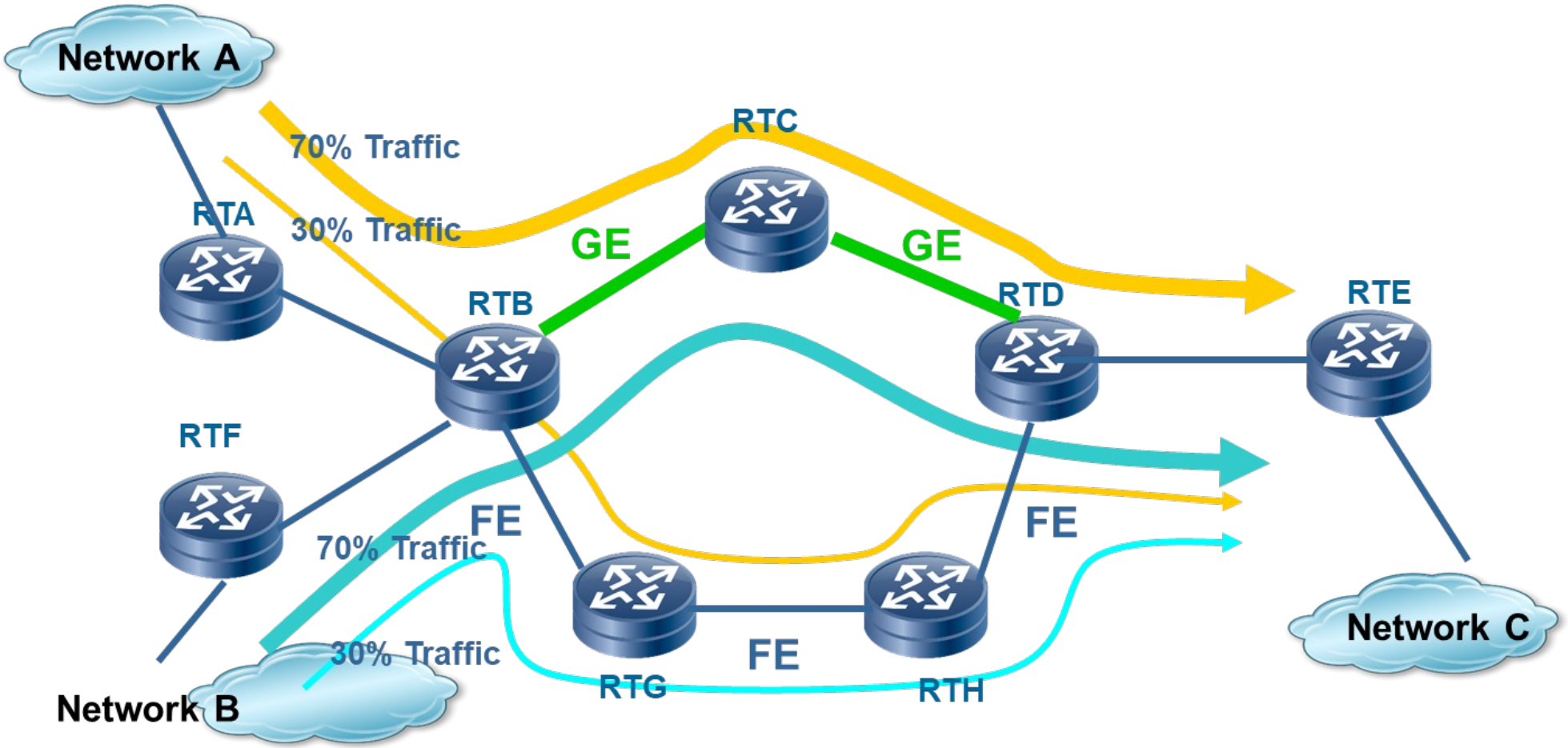
- “给 IP 数据报加标签”其实就是在以太网的帧首部和IP数据报的首部之间插入一个 4 字节的 MPLS 首部
- MPLS又称为2.5层协议



标签交换和MPLS-应用



标签交换和MPLS-应用



本章内容

5.1 网络层概述

5.2 网络层协议

5.3 路由算法

5.4 流量管理与服务质量

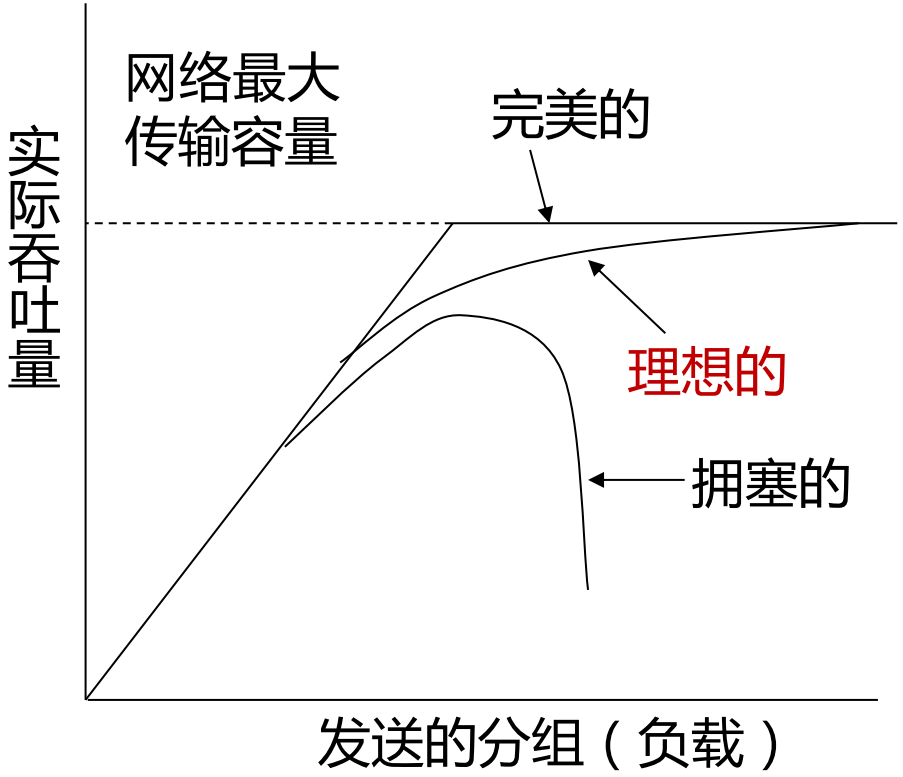
5.5 路由器体系结构与关键技术

5.6 软件定义网络

1. 拥塞控制概述
2. 流量管理
3. 网络服务质量

拥塞控制概述

- 拥塞
  - 网络中存在太多的数据包导致数据包传输延迟或丢失，从而导致网络吞吐量下降
- 拥塞控制（congestion control）
  - 需要确保通信子网能够承载用户提交的通信量，是全局性问题，涉及主机、路由器等多种因素
- 产生拥塞的原因
  - 主机发送到网络的数据包数量过多，超过了网络的承载能力
  - 突发的流量填满了路由器的缓冲区，造成某些数据包会被丢弃



拥塞控制概述

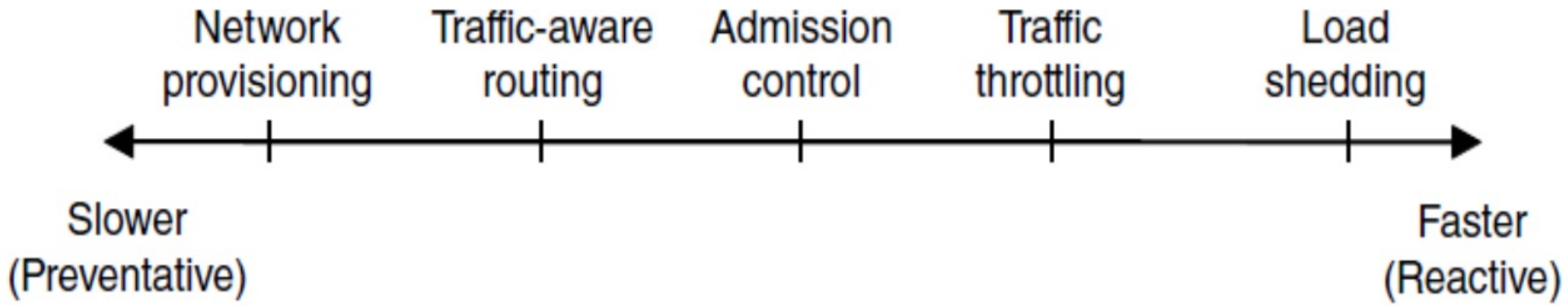
➢ 拥塞控制的基本策略

- **开环控制**----事先对通信流参数进行协商，协商后，不管网络是拥塞还是带宽充足，参数不能动态改变
  - 开环控制属于预防性拥塞控制，它竭力使网络总是处于无拥塞状态运行。开环控制的方法包括决定什么时候接受新流量，什么时候丢弃数据包和丢弃哪些数据包。其缺点是没有考虑网络的当前状态
- **闭环控制**---根据网络状态进行动态控制，包括两部分：反馈机制和控制机制。闭环控制方法分为两个子类：显式反馈与隐式反馈

拥塞控制概述

➢ 拥塞控制的途径

- 提高网络供给
- 流量感知路由
- 准入控制
- 流量调节
- 负载丢弃



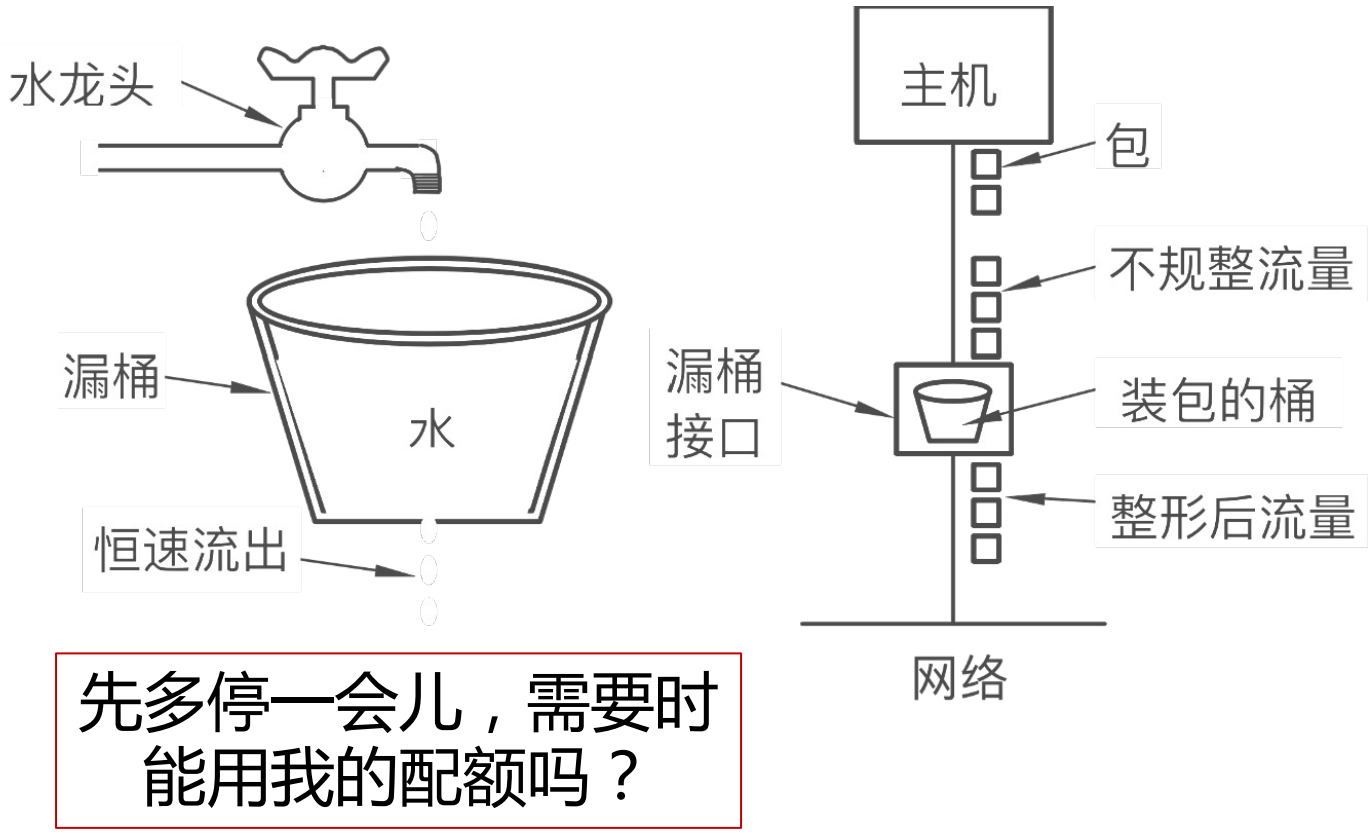
流量整形

- 流量整形(traffic shaping)
  - 其作用是限制流出某一网络的某一连接的流量与突发，使这类报文以比较均匀的速度向外发送
- 流量整形算法包括漏桶算法和令牌桶算法
  - 漏桶算法（Leaky Bucket Algorithm）：主要目的是控制数据注入到网络的速率，平滑网络上的突发流量，突发流量可以被整形以便为网络提供一个稳定的流量
  - 令牌桶算法（Token Bucket Algorithm）：用来控制发送到网络上的数据的数目，并允许突发数据的发送

流量整形

漏桶算法原理

- 到达的数据包（网络层的PDU）被放置在底部具有漏孔的桶中（数据包缓存）
- 漏桶最多可以**排队b个字节**，漏桶的这个尺寸受限于内存。如果数据包到达的时候漏桶已经满了，那么数据包应被丢弃
- 数据包从漏桶中漏出，以常量速率（**r字节/秒**）注入网络，因此平滑了突发流量

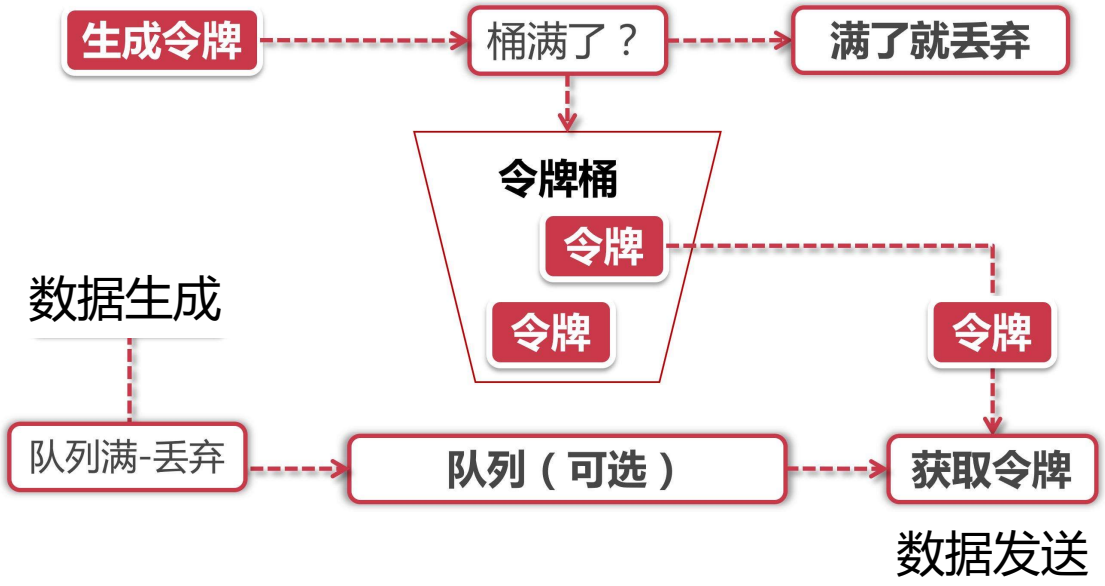


先多停一会儿，需要时能用我的配额吗？

流量整形

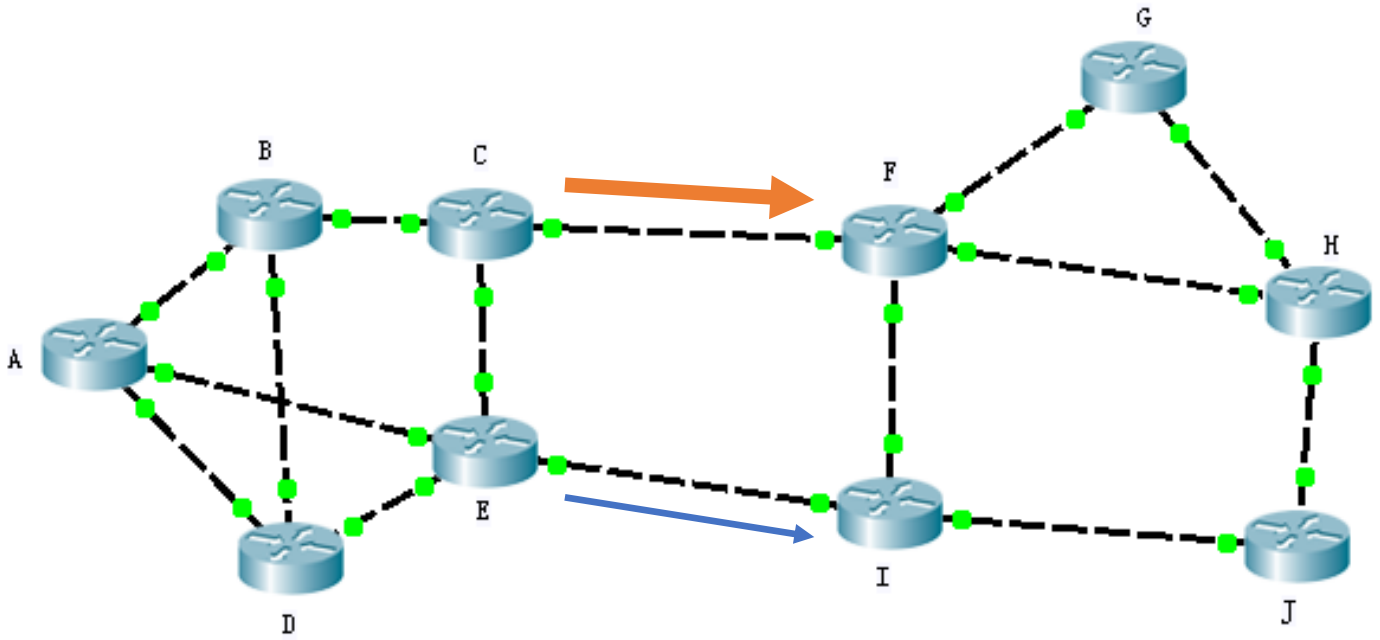
令牌桶算法工作原理

- 产生令牌：周期性的以速率r向令牌桶中增加令牌，桶中的令牌不断增多。如果桶中令牌数已到达上限，则丢弃多余令牌
- 消耗令牌：输入数据包会消耗桶中的令牌。在网络传输中，数据包的大小通常不一致。大的数据包相较于小的数据包消耗的令牌要多
- 判断是否通过：输入数据包经过令牌桶时存在两种可能：输出该数据包或者被丢弃。当桶中的令牌数量可以满足数据包对令牌的需求，则将数据包输入，否则将其丢弃



服务质量路由QoSR

- 思路：绕开热门的区域，疏散流量
- 方法：计算路径权重时包含跳数、带宽、传输延迟、负载、排队延迟等
- 问题：路由表可能会出现反复变化，从而导致不稳定的路由



发的数据实在太多怎么办？

海量终端如何互相协调？

流量调节

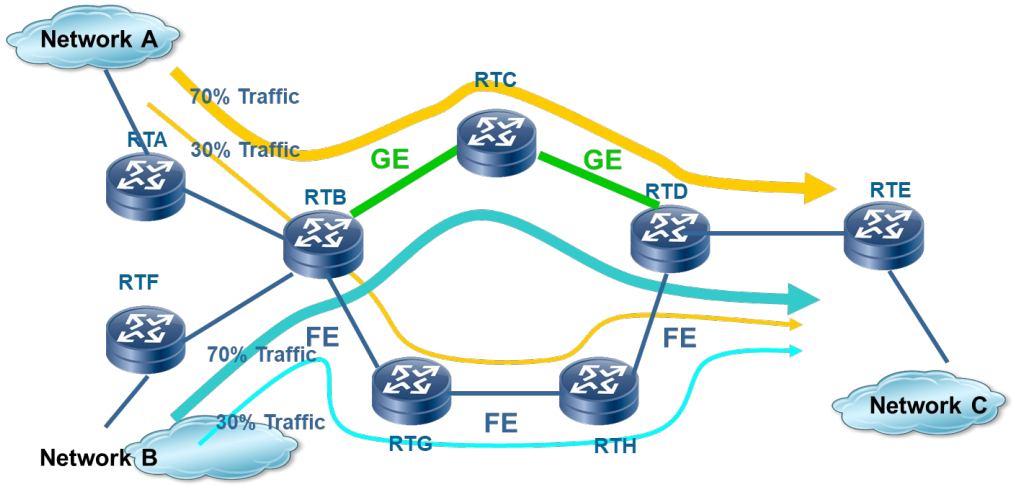
➢ 抑制包(Choke Packets)

- 用于通知发送方减小发送量，路由器选择一个被拥塞的数据包，给该数据包的源主机返回一个抑制包，抑制包中的目的地址取自该拥塞数据包
- 源主机收到抑制包后，减少发向特定目的地址的流量

如何提前预警，
让源端少发一些？

➢ 逐跳的抑制包(Hop-by-Hop Choke Packets)

- 在高速或长距离网络中，由于源主机响应太慢，抑制包算法对拥塞控制的效果并不好，可采用逐跳抑制方法
- 其核心思想是抑制包对它经过的每个路由器都起作用，能够迅速缓解发生拥塞处的拥塞，但要求上游路由器有更大的缓冲区



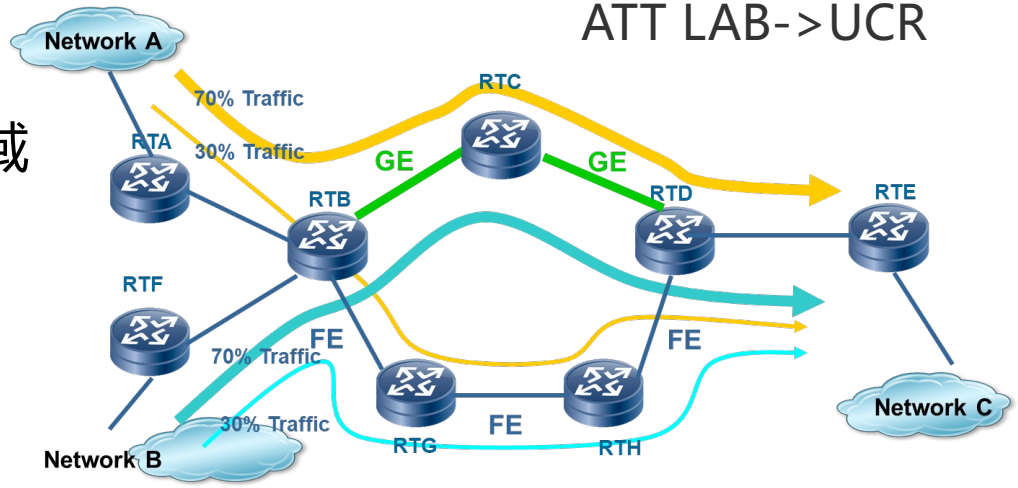
流量调节

- 显式拥塞通告ECN
  - ECN：Explicit Congestion Notification
  - 在IP包头中记录数据包是否经历了拥塞
  - 在数据包转发过程中，路由器可以在包头中标记为经历拥塞
  - 接收方在下一个应答数据包里回显该标记作为显式拥塞信号
  - 发送端降速
- 写入IETF国际标准
  - RFC3168定义IP头的TOS域未使用的两位为ECN域
  - 00：发送主机不支持ECN
  - 01或者10：发送主机支持ECN
  - 11：路由器正在经历拥塞

好技术用了吗？

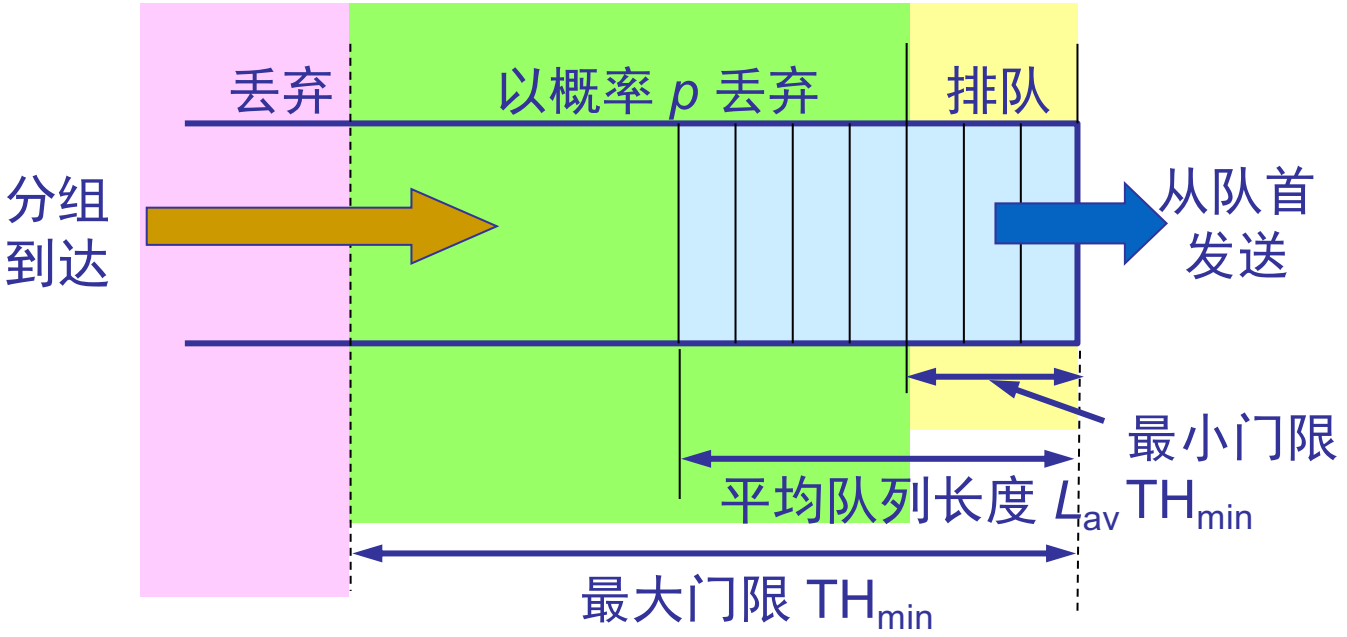
避免端网协同？

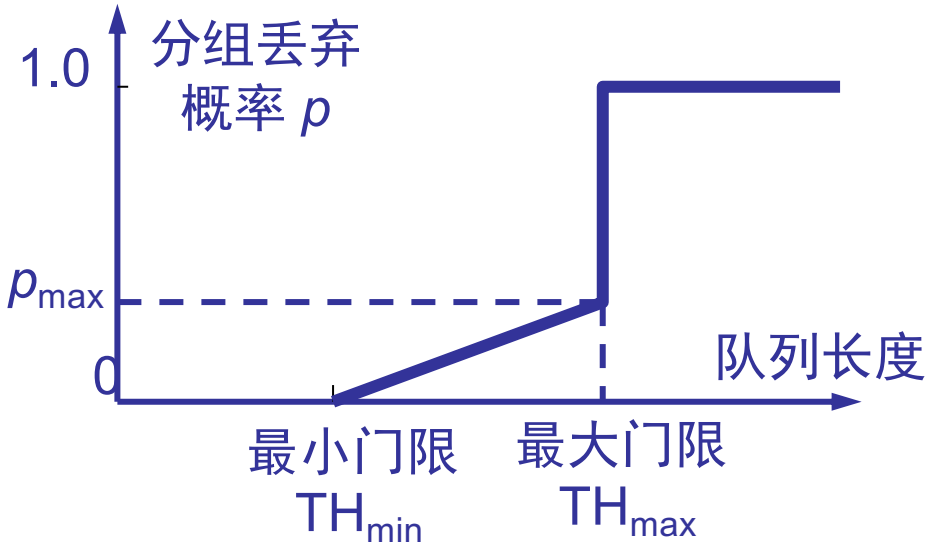
K. K. Ramakrishnan
ATT LAB->UCR



随机早期检测

- 传统方式的问题
  - Buffer满了只能全部丢掉，有什么问题？所有流量损失惨重，能否不要太生硬？
- 随机早期检测 RED (Random Early Detection)
  - RED 将路由器的到达队列划分成为三个区域





网络服务质量概述

➢ 问题的提出

- 互联网本身只能提供“尽力而为的服务”或称“尽最大努力交付的服务”
- 当互联网越来越多的用于传输多媒体信息时，由于这些实时业务对网络的传输延时、延时抖动等特性较为敏感，这样网络的传输质量就难以保障了
- IP网络不能保证特定业务的QoS要求，已经成为IP网络发展的巨大障碍
- 网络的服务质量越来越多的引起人们的关注，甚至成为网络技术研究的热点问题

➢ 什么是网络服务质量？（QoS, Quality of Service）

- QoS是网络在传输数据流时要满足一系列服务请求，具体可以量化为带宽、时延、抖动、丢包率等性能指标

网络服务质量概述

确保服务质量需要解决以下4个问题：

- （1）应用程序需要网络提供什么样的质量？
- （2）如何规范进入网络的流量？
- （3）为了保障性能如何在路由器预留资源？
- （4）网络能否安全的接受更多流量？

QoS的度量指标

- 带宽
- 时延
- 抖动
- 丢包率

QoS针对各种业务的不同需求，为其提供端到端的服务质量保证。在有限的带宽资源下，它允许不同的流量不平等地竞争网络资源，语音、视频和重要的数据应用在网络设备中可以优先得到服务

数据包调度

- 在同一个流的数据包之间以及在竞争流之间分配路由器资源的算法称为包调度算法，它负责分配带宽和其他路由器资源，负责确定把缓冲区中的哪些数据包发送到输出链路上
  - 先来先服务FCFS（First-Come First-Serve）
  - 公平队列算法（Fair Queueing）
  - 加权公平队列算法（Weighted Fair Queueing）
  - 优先级调度（Priority Scheduling）

综合服务

综合服务（IntServ：Integrated Services）

- 综合服务特点是：需要所有的路由器在控制路径上处理每个流的消息，维护每个流的路径状态和资源预留状态，在路径上执行基于流的分类、调度、管理
- 综合服务基于资源预留协议RSVP，逐节点建立或拆除流的状态和资源预留状态，根据流的状态进行QoS路由
- 综合服务的特征：资源预分配、全局流状态、传输控制

区分服务

区分服务（DiffServ：Differentiated services）

- 区分服务（DiffServ）是一种计算机网络体系结构，它指定了一种简单且可扩展的机制，用于在IP网络上分类和管理网络流量并提供服务质量（QoS）
- DiffServ可用于向诸如语音或流媒体之类的关键网络流量提供低延迟服务，同时向诸如web流量或文件传输之类的非关键服务提供简单的尽力而为服务
- DiffServ在IP报头的8位区分服务字段（DS字段）中使用6位区分服务码点（DSCP）进行分组分类。换言之，DS字段替换过时的IPv4 TOS字段

区分服务

区分服务（DiffServ：Differentiated services）

- 边界节点根据约定好的QoS规定，把将要进入网络的流量分类成不同的流。流的聚类信息用IP头部的DS field来标识。内部路由节点在转发这种包的时候，只需根据不同的DSCP选择相应的调度和转发服务即可。

Assured Forwarding behavior group

	Class 1	Class 2	Class 3	Class 4
Low drop probability	AF11 (DSCP 10) 001010	AF21 (DSCP 18) 010010	AF31 (DSCP 26) 011010	AF41 (DSCP 34) 100010
Med drop probability	AF12 (DSCP 12) 001100	AF22 (DSCP 20) 010100	AF32 (DSCP 28) 011100	AF42 (DSCP 36) 100100
High drop probability	AF13 (DSCP 14) 001110	AF23 (DSCP 22) 010110	AF33 (DSCP 30) 011110	AF43 (DSCP 38) 100110

服务质量和拥塞控制算法小结

- 如何用上更好的网络服务？
  - 更好？网络服务质量QoS（带宽、时延、抖动、丢包率）
- 网络层QoS
  - 突发流量 <-> 流量整形
    - 限制突发流量产生，避免节点缓存波动导致的丢包
  - 缺少差异化服务<->综合服务、区分服务
    - IntServ逐流维护状态，DiffServ按照每类处理优先级
  - 流量超出带宽<->拥塞控制
    - 流量感知路由：通过调度流量绕开热门区域，疏解流量
    - 端网协同的流量调节：抑制包（显示拥塞通告ECN），逐跳反压机制
    - 网络节点的随机早期检测：未雨绸缪，缓解即将到来的拥塞

本章内容

5.1 网络层概述

5.2 路由算法

5.3 流量管理与服务质量

5.4 网络层协议

5.5 路由器体系结构

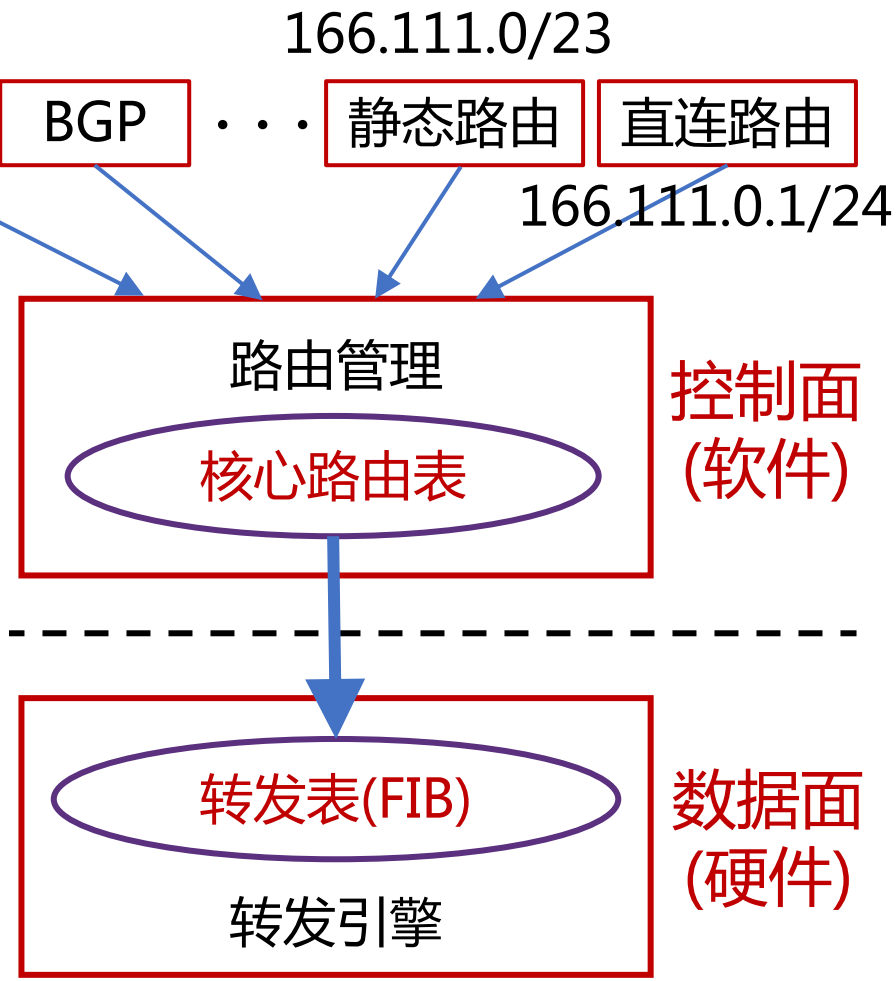
5.6 软件定义网络

1. 路由器概述
2. 路由器控制层
3. 路由器数据层
4. 路由器交换结构

路由器控制平面

- 控制面的路由管理
  - 路由器（主控板）可同时运行多个路由协议
  - 路由器也可不运行任何路由协议，只使用静态路由和直连路由
  - 路由管理根据路由优先级，选择最佳路由，形成核心路由表
  - 路由管理将核心路由表提供给各个路由协议，实现控制面闭环
- 路由表 v.s. 转发表
  - 控制面将核心路由表下发到接口板的数据面，形成转发表（FIB）

主控板 v.s. 接口板



路由器控制平面

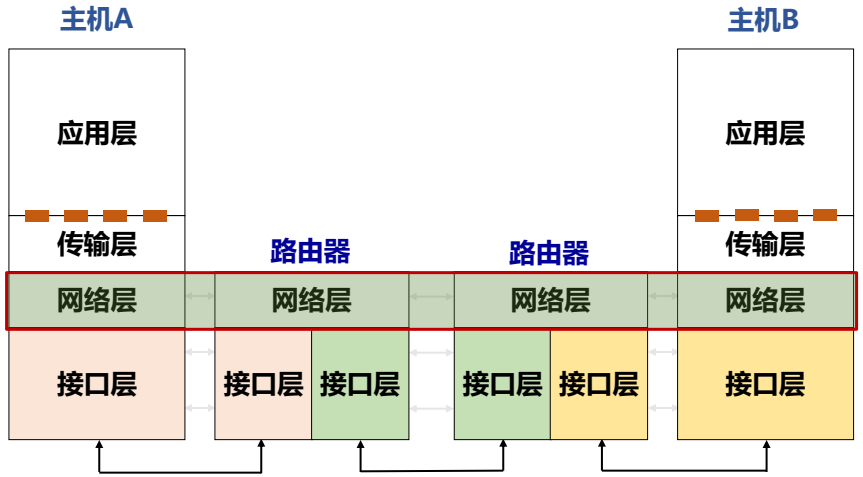
- 若存在多个“去往同一目的IP前缀”的不同类型路由，路由器根据优先级选择最佳路由（形成转发表）
- 优先级数值越小，优先级越高

路由种类	路由优先级
直连路由	0
静态路由	1
eBGP路由	20
OSPF路由	110
RIP路由	120
iBGP路由	200

热土豆在哪里？

路由器数据平面

- 路由器中IP报文转发过程
  - 链路层解封装（核对MAC地址、MAC校验）
  - 若上层为IP，进行IP头部校验，获取目的IP地址
  - 用目的IP地址和最长前缀匹配规则，查询转发表
  - 路由匹配查询失败，丢弃报文
  - 查询成功
    - 获取转发出接口和下一跳IP地址
    - IP头部“TTL”字段值减1，重新计算IP头部“校验和”
    - 重新进行链路层封装，发送报文

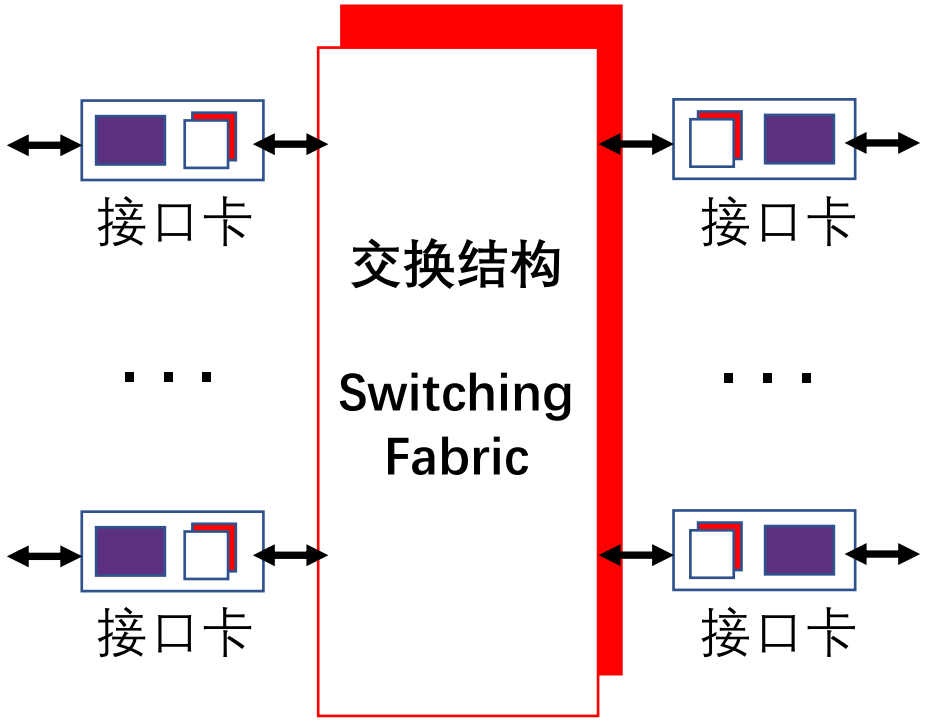


普通IP报文转发过程中，路由器不查看传输层及以上层协议的内容

- IP报文在路由器转发前后的变化
  - 链路层封装更新（新头）；IP头部“TTL”减1，IP头部“校验和”更新

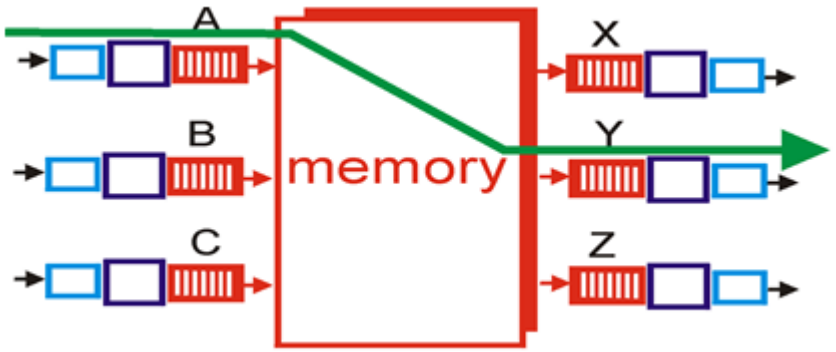
路由器数据层

- 数据报在不同硬件单元的处理
  - 报文输入的接口卡
    - 链路层解封装
    - 转发表查询
    - 通过交换结构将报文排队并发往目的接口卡
  - 不同类型的交换结构
    - 从输入接口卡发往输出接口卡
  - 报文输出的接口卡
    - 从交换结构接收报文，排队进行后续处理
    - 链路层封装
    - 从输出接口发送报文

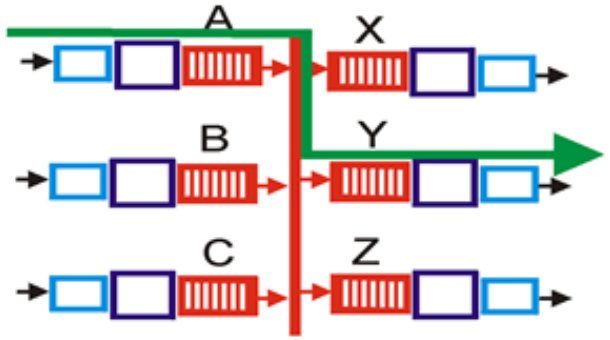


3种典型的交换结构

共享内存

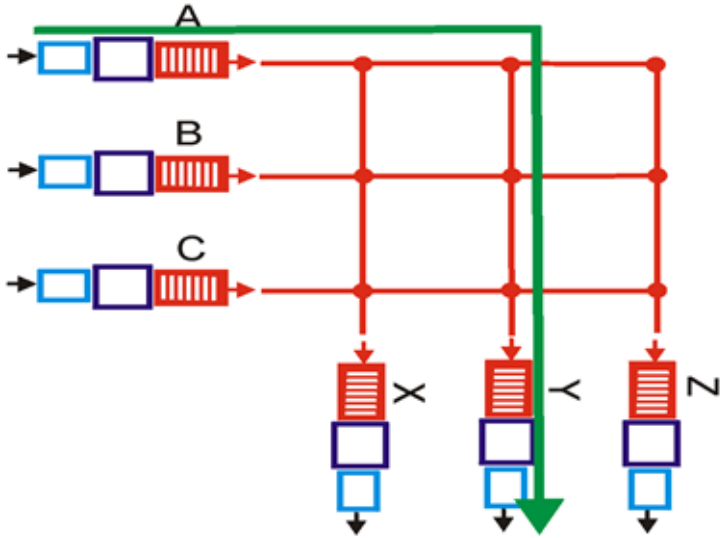


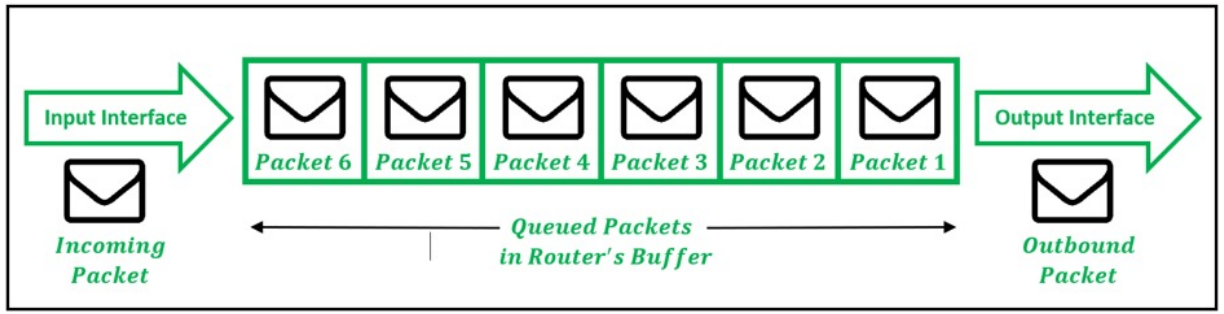
共享总线



纵横式 Crossbar

应用于高性能分布式路由器





输入排队 v.s. 输出排队
队头丢弃 v.s. 队尾丢弃

路由器扩展知识

路由器有4/5两层协议吗？

- 路由器的端系统角色
  - 远程网络管理，SNMP
  - 远程网络配置，SSH
  - 文件传输，FTP，TFTP
  - 各种路由协议交互
  - ...
  - 路由器系统包含完整TCP/IP协议栈
    - 传输层协议
    - 应用层协议

软件 v.s. 硬件

- 家用路由器
  - 不运行动态路由协议（出口唯一）
  - 运行DHCP协议，分配私有IP
  - NAT地址转换
  - 本地DNS服务
  - 用户管理及认证
  - 防火墙功能
  - 无线AP……

与典型网络路由器差异较大

本章内容

5.1 网络层概述

5.2 路由算法

5.3 流量管理与服务质量

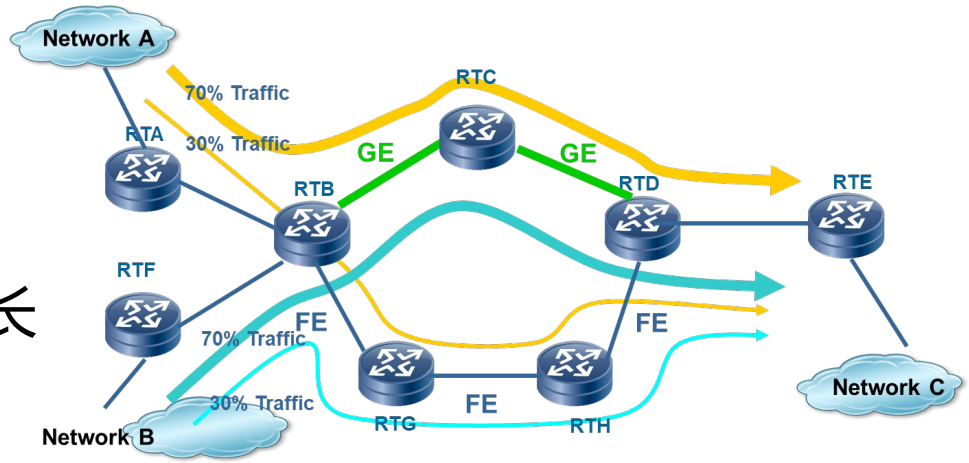
5.4 网络层协议

5.5 路由器体系结构

5.6 软件定义网络

传统网络面临的问题

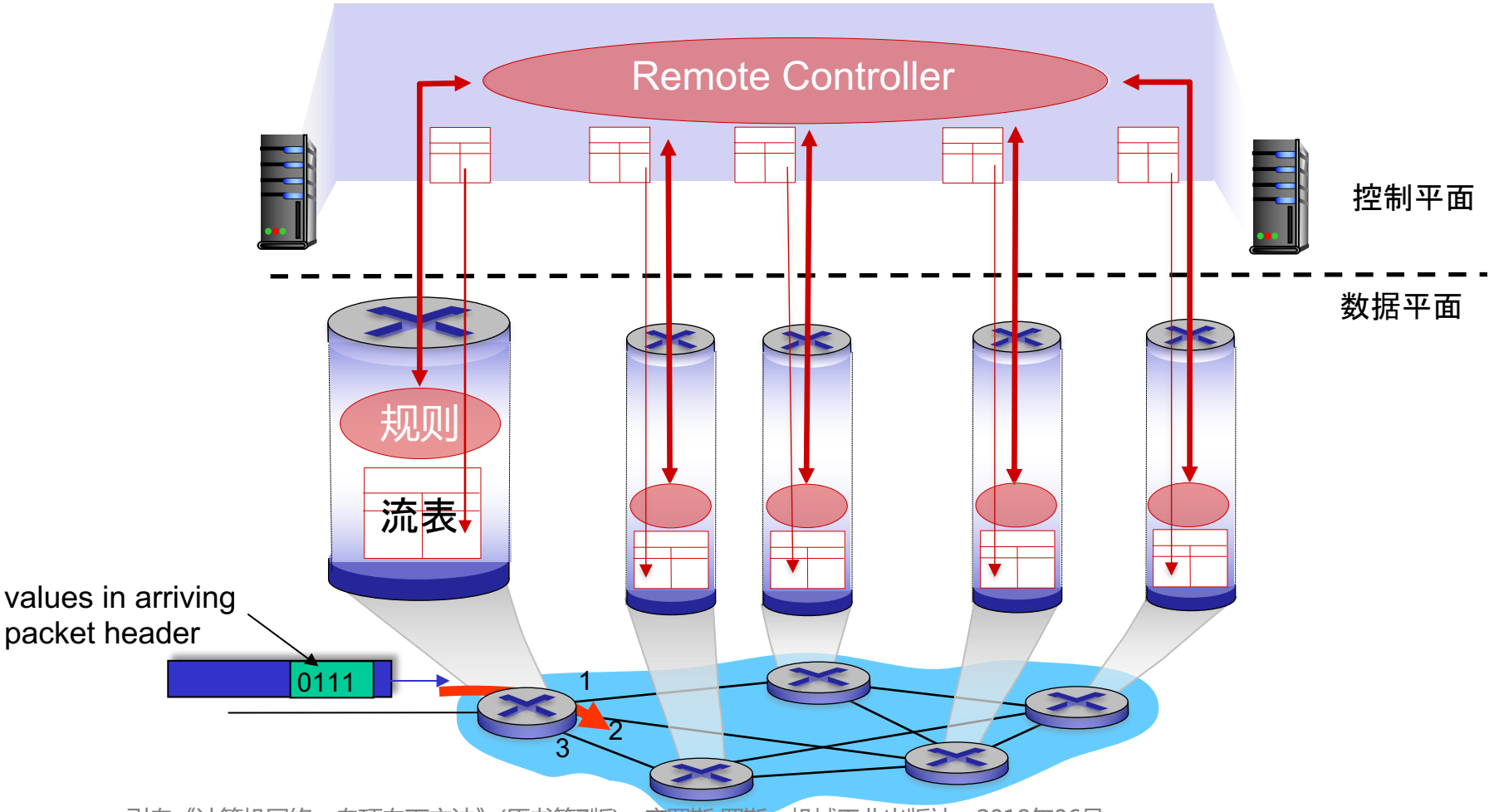
- 分布式路由的缺点
  - 网络设备以接力棒形式告诉下一跳邻居设备
  - 缺少全局控制，难以优化
  - 底层网络设备数量不断增加，路由收敛时间长



- 传统网络不可编程，运营商难以掌控
  - 互联网核心节点硬件实现价格高，门槛高
  - 运营商的网络新需求难以设备难以支持，运营商难以配置
  - 很难实现高效、按需的数据传输
- 运营商和互联网厂商的新呼唤
  - 用简单的硬件，实现灵活的组网和配置

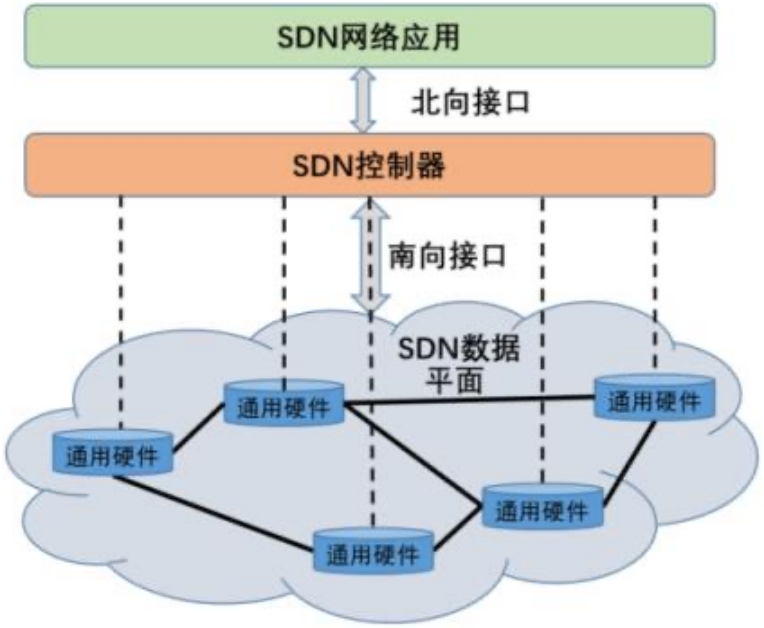
SDN：控制平面和数据平面分离

远程控制、计算、配置SDN交换机



软件定义网络SDN

- 软件定义网络（SDN）
  - 新的网络体系结构：分离转发面与控制面
- SDN控制器集中控制
  - 具有全局视野，动态控制网络
  - 计算最优路径，下发流表控制交换机转发
  - 实现高效、按需的数据传输
- 主要接口
  - 北向接口：向上提供，进行适配应用和管理网络
  - 南向接口：向下提供，管理转发面设备



网络的成本去哪里了？

谁喜欢这个方案？

集中式网络与路由服务器

话说天下大势
分久必合，合久必分