# The Network Layer

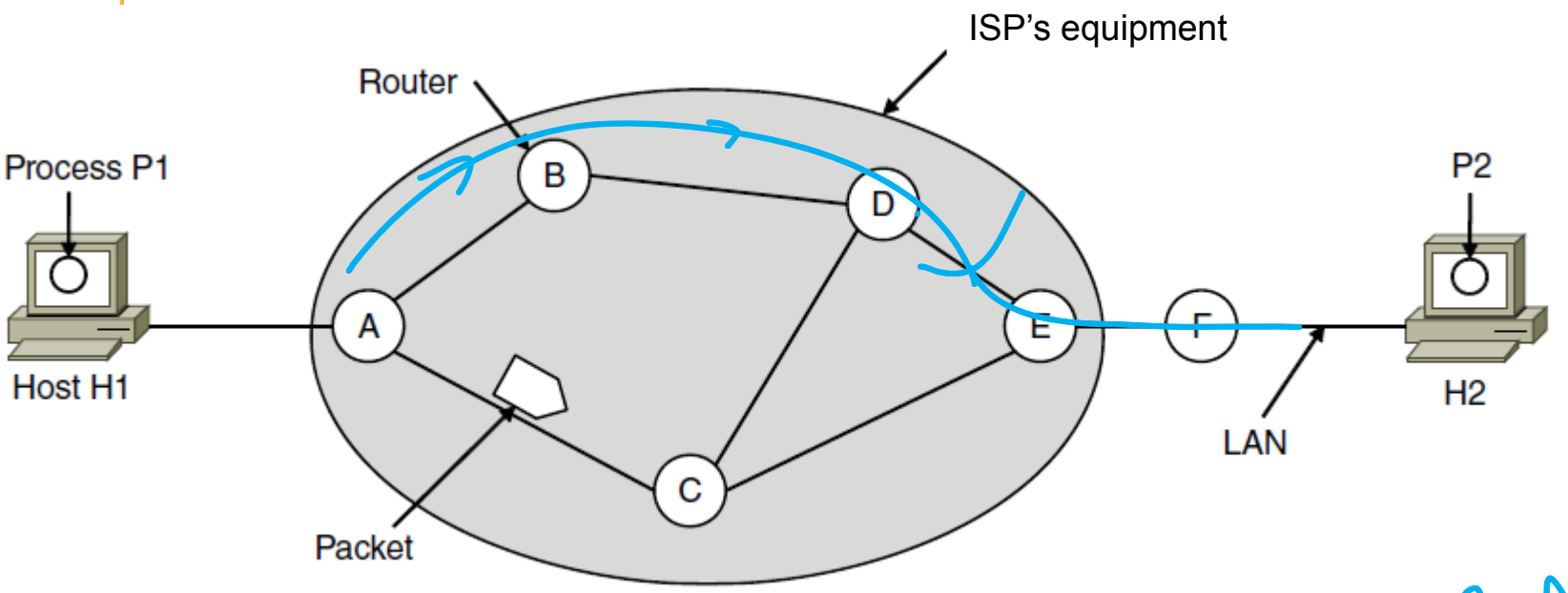
Chapter 5

# Network Layer Design Issues

- Store-and-forward packet switching
- Services provided to transport layer
- Implementation of connectionless service
- Implementation of connection-oriented service
- Comparison of virtual-circuit and datagram networks

# Store-and-Forward Packet Switching

Network layer uses ip
Transport layer uses
port address



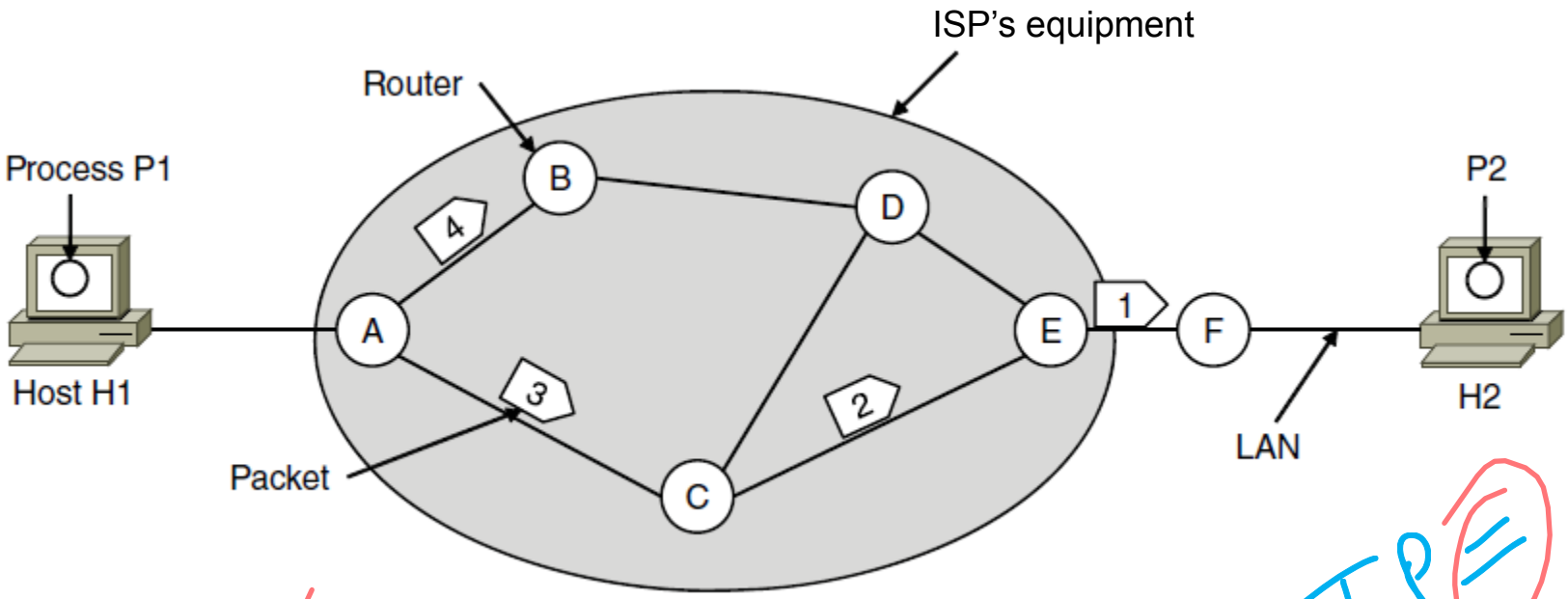

The environment of the network layer protocols.

# Services Provided to the Transport Layer

network to ?

- Services independent of router technology.
- Transport layer shielded from number, type, topology of routers.
- Network addresses available to transport layer use uniform numbering plan even across LANs and WANs

# Implementation of Connectionless Service




A's table (initially)

| Dest. | Line |
|---|---|
| A | |
| B | B |
| C | C |
| D | B |
| E | C |
| F | C |

A's table (later)

| | |
|---|---|
| A | |
| B | B |
| C | C |
| D | B |
| E | D |
| F | D |

C's Table

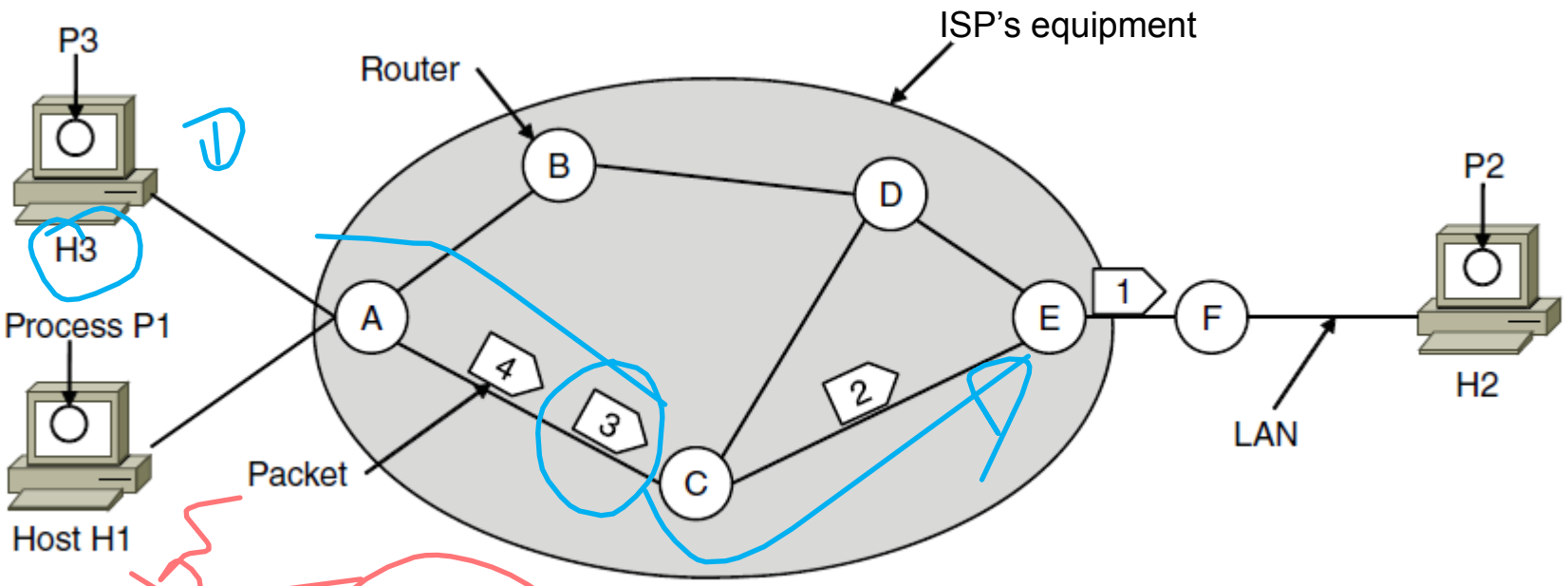
| | |
|---|---|
| A | A |
| B | A |
| C | |
| D | E |
| E | E |
| F | E |

E's Table

| | |
|---|---|
| A | C |
| B | D |
| C | C |
| D | D |
| E | |
| F | F |

Routing within a datagram network

# Implementation of Connection-Oriented Service




Routing within a virtual-circuit network

# Comparison of Virtual-Circuit and Datagram Networks

| Issue | Datagram network | Virtual-circuit network |
|---|---|---|
| Circuit setup | Not needed | Required |
| Addressing | Each packet contains the full source and destination address | Each packet contains a short VC number |
| State information | Routers do not hold state information about connections | Each VC requires router table space per connection |
| Routing | Each packet is routed independently | Route chosen when VC is set up; all packets follow it |
| Effect of router failures | None, except for packets lost during the crash | All VCs that passed through the failed router are terminated |
| Quality of service | Difficult | Easy if enough resources can be allocated in advance for each VC |
| Congestion control | Difficult | Easy if enough resources can be allocated in advance for each VC |

Comparison of datagram and virtual-circuit networks

# Routing Algorithms (1)

- Optimality principle
- Shortest path algorithm
- Flooding
- Distance vector routing
- Link state routing
- Routing in ad hoc networks

# Routing Algorithms (2)

- Broadcast routing
- Multicast routing
- Anycast routing
- Routing for mobile hosts
- Routing in ad hoc networks

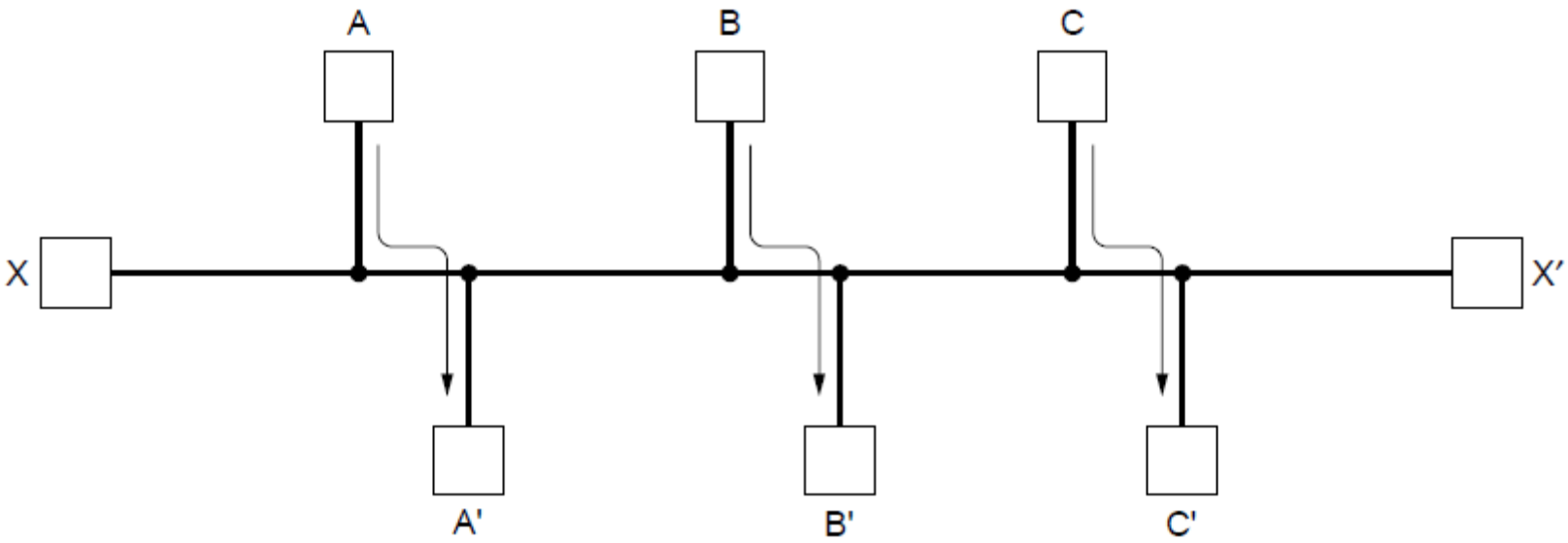
# Routing Vs Forwarding

- Routing is making the decision which routes to use, and forwarding, is what happens when a packet arrives
- Router has two processes inside it. One of them handles each packet as it arrives, looking up the outgoing line to use for it in the routing tables. This process is **forwarding**. The other process is responsible for filling in and updating the routing tables.

# Routing Algorithms

- Desirable properties of routing algorithm:
  - Correctness – Right path is chosen
  - Simplicity – Ease of use and implementation
  - Robustness – reliability in event of failure
  - Stability - converge to fixed set of paths
  - Fairness – Fair to all hosts
  - Efficiency – Utilization, delay

# Fairness vs. Efficiency



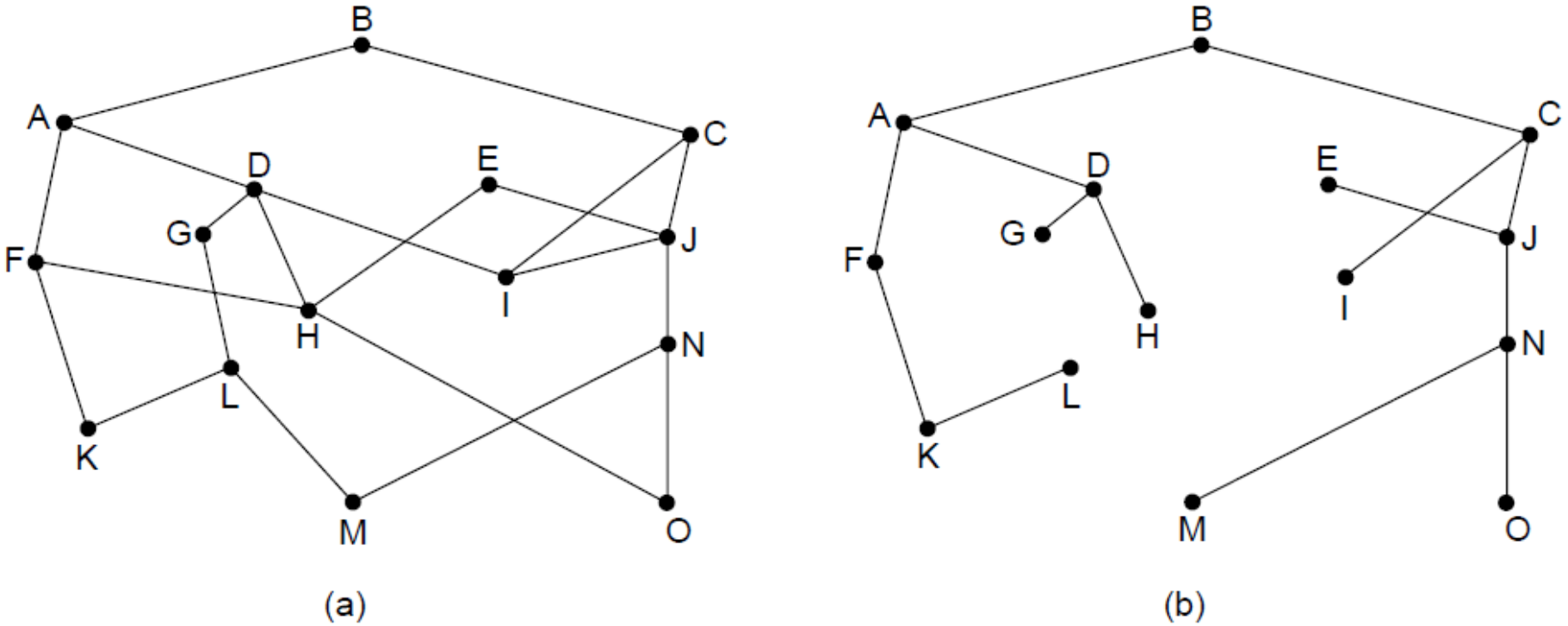

Network with a conflict between fairness and efficiency.

# Routing Algorithms

- Types of routing algorithm:
  - Nonadaptive RA – do not base their routing decisions on any measurements or estimates of the current topology and traffic. Instead the routes are computed in advance, offline and downloaded to routers when network boots. Eg Static Routing
  - Adaptive RA - change their routing decisions to reflect changes in the topology, and sometimes changes in the traffic as well. They differ in where they get their information, when they change the routes, and what metric is used for optimization. Eg Distance Vector Routing, Link State Routing

# The Optimality Principle

**Statement about optimal routes without regard to network topology or traffic**
*It states that if router J is on the optimal path from router I to router K, then the optimal path from J to K also falls along the same route.*




(a) A network. (b) A sink tree for router *B.*

# Flooding

A simple method to send a packet to all network nodes

Each node floods a new packet received on an incoming link by sending it out all of the other links

Flooding generates vast numbers of duplicate packets, in fact, an infinite number unless some measures are taken to damp the process.



CN5EbyTanenbaum&WetherallPearsonEducation-PrenticeHall

One such measure is to have a hop counter contained in the header of each packet that is decremented at each hop.

The packet being discarded when the counter reaches zero.

Ideally, the hop counter should be initialized to the length of the path from source to destination.

If the sender does not know how long the path is, it can initialize the counter to the worst case i.e. the full diameter of the network.

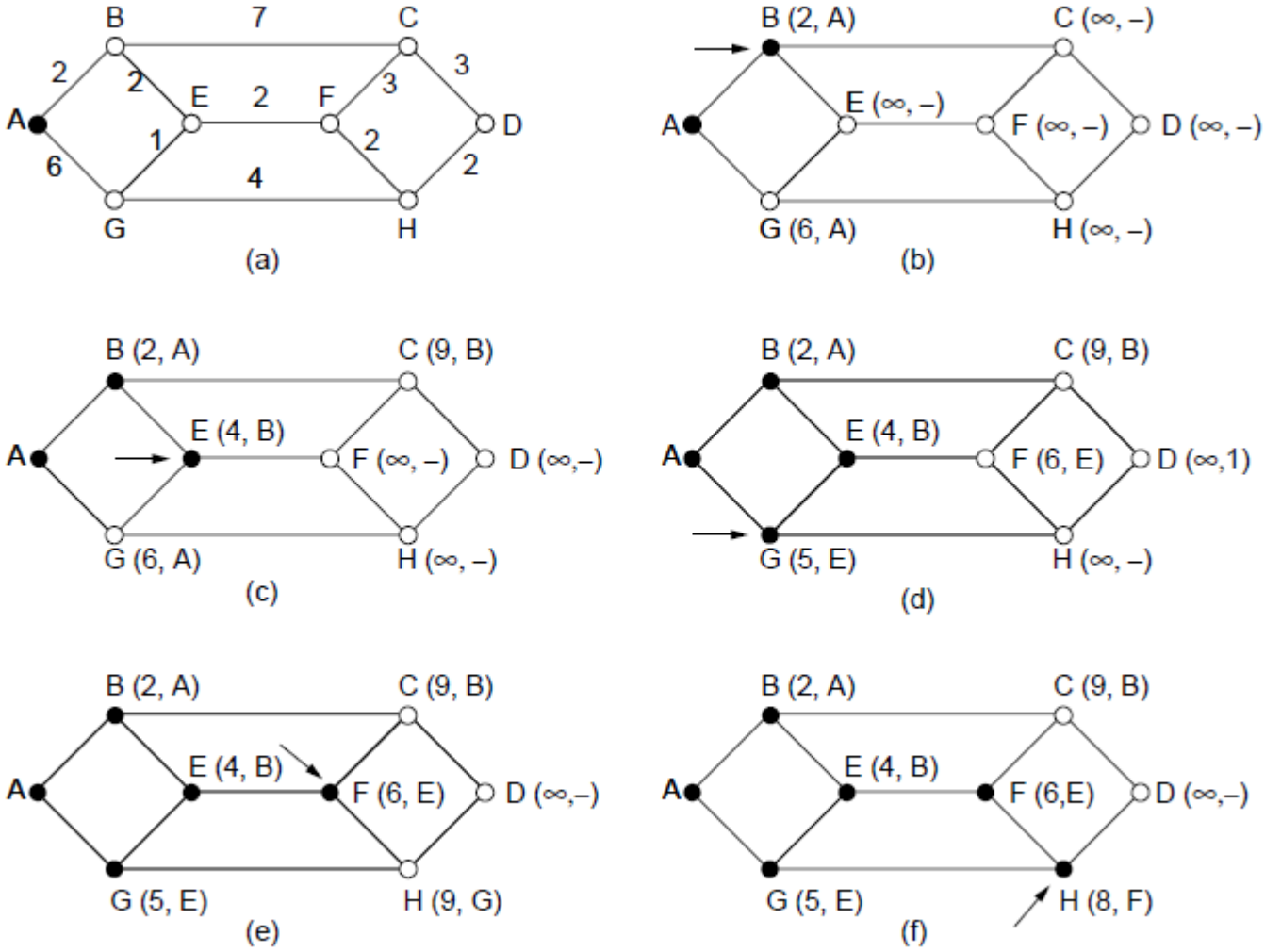
First, it ensures that a packet is delivered to every node in the network.

This may be wasteful if there is a single destination that needs the packet,

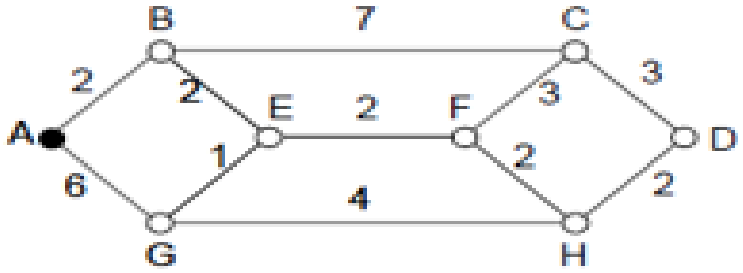
but it is effective for broadcasting information.

flooding will find a path if one exists, to get a packet to its destination(e.g. in a military network located in a war zone).

# Shortest Path Algorithm (1)

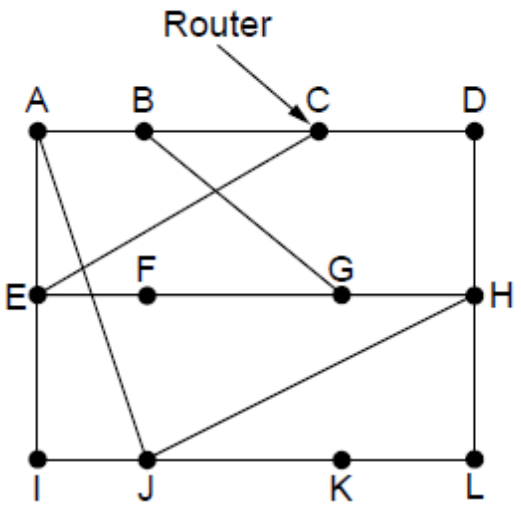



The first five steps used in computing the shortest path from *A to D.* The arrows indicate the working node




| | A | B | C | D | E | F | G | H |
|---|---|---|---|---|---|---|---|---|
| | | 2 A | ∞ | ∞ | ∞ | ∞ | 6 A | ∞ |
| +2 | B | ✓ | 9 B | ∞ | 4 B | ∞ | 6 A | ∞ |
| +4 | E | ✓ | 9 B | ∞ | ✓ | 6 A | 5 E | ∞ |
| +5 | G | ✓ | 9 B | ∞ | ✓ | 6 A | ✓ | 9 G |
| +6 | F | ✓ | 9 B | ∞ | ✓ | ✓ | ✓ | 8 F |
| +8 | H | ✓ | 9 B | 10 H | ✓ | ✓ | ✓ | ✓ |
| +9 | C | ✓ | ✓ | 10 H | ✓ | ✓ | ✓ | ✓ |

# Distance Vector Routing




(a)

New estimated delay from J

| To | A | I | H | K | Line |
|---|---|---|---|---|---|
| A | 0 | 24 | 20 | 21 | 8 A |
| B | 12 | 36 | 31 | 28 | 20 A |
| C | 25 | 18 | 19 | 36 | 28 I |
| D | 40 | 27 | 8 | 24 | 20 H |
| E | 14 | 7 | 30 | 22 | 17 I |
| F | 23 | 20 | 19 | 40 | 30 I |
| G | 18 | 31 | 6 | 31 | 18 H |
| H | 17 | 20 | 0 | 19 | 12 H |
| I | 21 | 0 | 14 | 22 | 10 I |
| J | 9 | 11 | 7 | 10 | 0 – |
| K | 24 | 22 | 22 | 0 | 6 K |
| L | 29 | 33 | 9 | 9 | 15 K |

JA delay is 8 | JI delay is 10 | JH delay is 12 | JK delay is 6

Vectors received from J's four neighbors

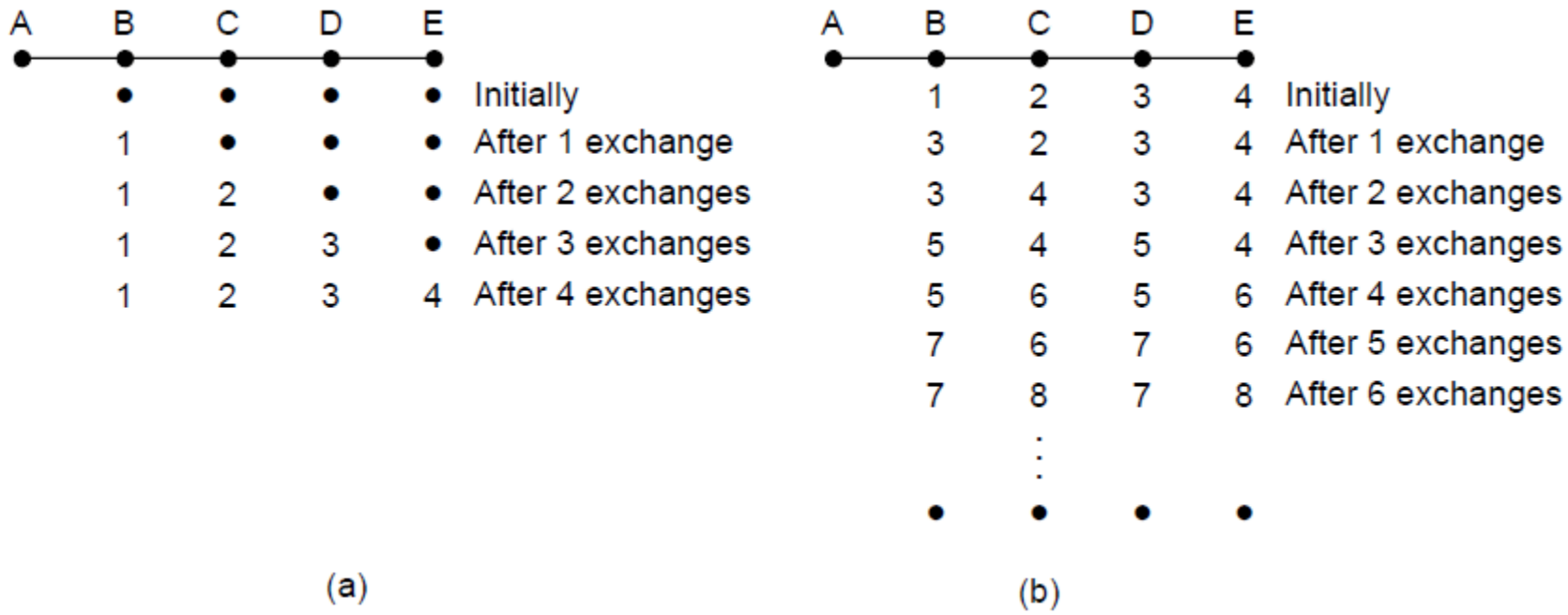
New routing table for J

(b)

(a) A network.

(b) Input from *A, I, H, K, and the new routing table* for *J.*

# The Count-to-Infinity Problem

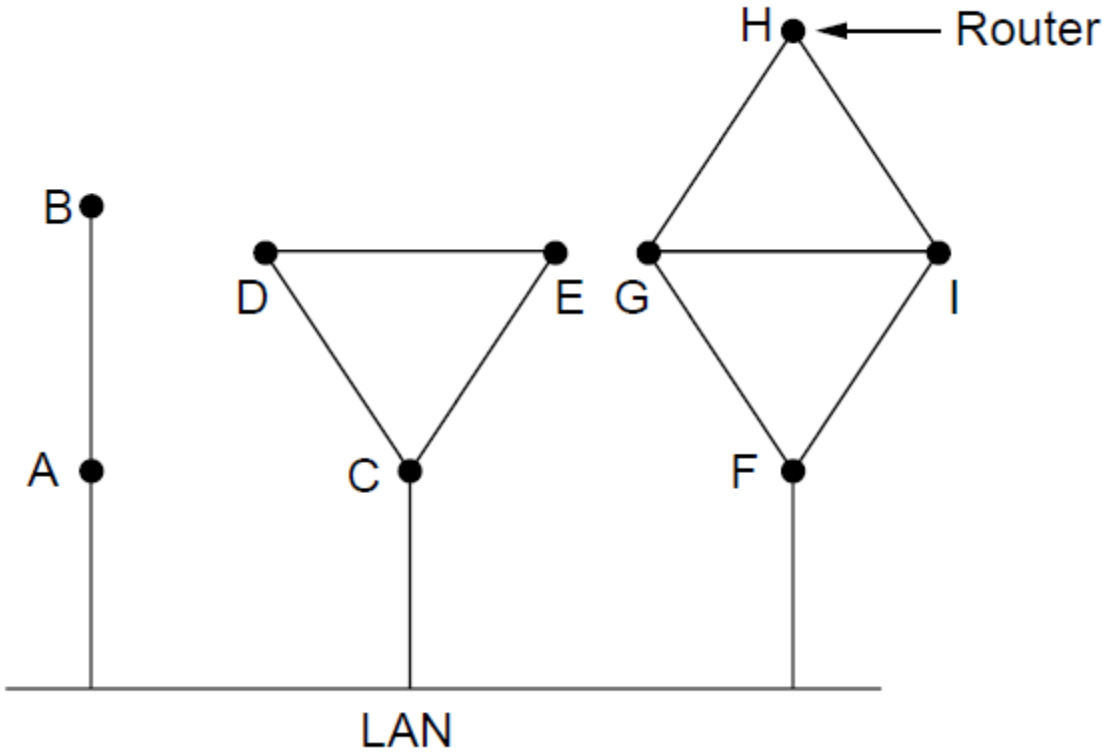



The count-to-infinity problem

# Link State Routing

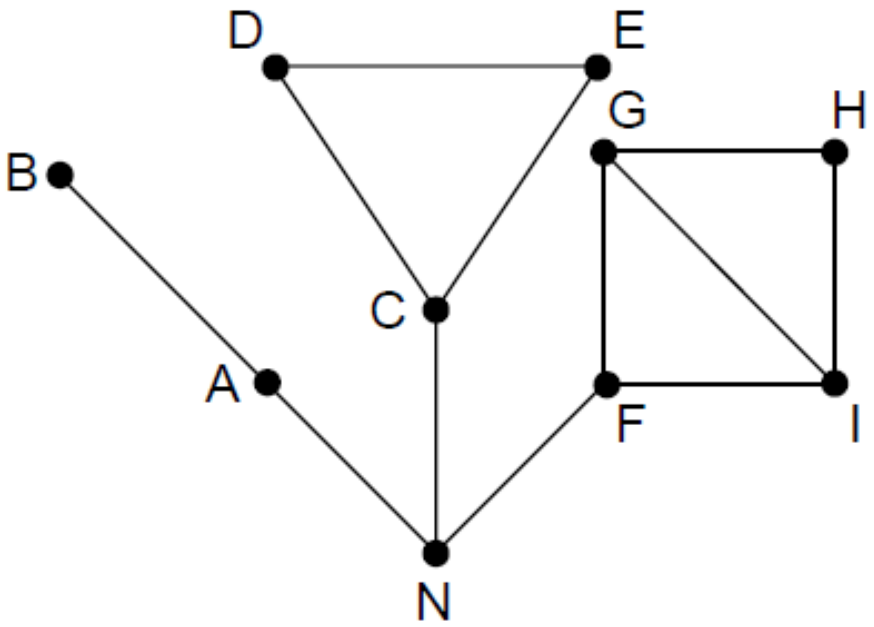
- Discover neighbors, learn network addresses.
- Set distance/cost metric to each neighbor.
- Construct packet telling all learned.
- Send packet to, receive packets from other routers.
- Compute shortest path to every other router.

# Learning about the Neighbors (1)



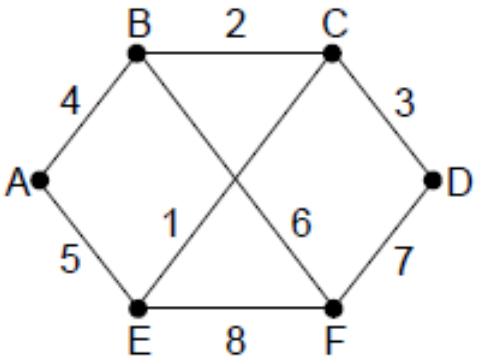

Nine routers and a broadcast LAN.

# Learning about the Neighbors (2)




A graph model of previous slide.

# Building Link State Packets




(a)

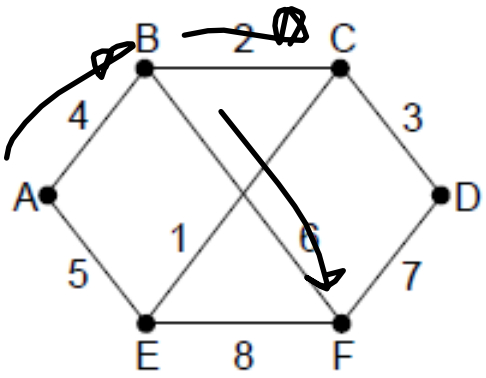
Link State Packets

| A | B | C | D | E | F |
|---|---|---|---|---|---|
| Seq. | Seq. | Seq. | Seq. | Seq. | Seq. |
| Age | Age | Age | Age | Age | Age |
| B 4 | A 4 | B 2 | C 3 | A 5 | B 6 |
| E 5 | C 2 | D 3 | F 7 | C 1 | D 7 |
| | F 6 | E 1 | | F 8 | E 8 |

(b)

(a) A network. (b) The link state packets for this network.

# Distributing the Link State Packets




(a)

Link State Packets

| A | | B | | C | | D | | E | | F | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Seq. | | Seq. | | Seq. | | Seq. | | Seq. | | Seq. | |
| Age | | Age | | Age | | Age | | Age | | Age | |
| B | 4 | A | 4 | B | 2 | C | 3 | A | 5 | B | 6 |
| E | 5 | C | 2 | D | 3 | F | 7 | C | 1 | D | 7 |
| | | F | 6 | E | 1 | | | F | 8 | E | 8 |

(b)

Send to: A C F — ACK to: A C F

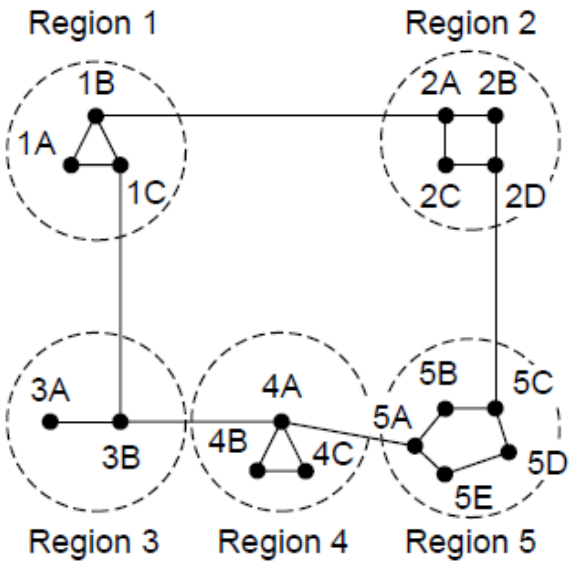
| Source | Seq. | Age | Send to A | Send to C | Send to F | ACK to A | ACK to C | ACK to F | Data |
|---|---|---|---|---|---|---|---|---|---|
| A | 21 | 60 | 0 | 1 | 1 | 1 | 0 | 0 | |
| F | 21 | 60 | 1 | 1 | 0 | 0 | 0 | 1 | |
| E | 21 | 59 | 0 | 1 | 0 | 1 | 0 | 1 | Received from A & F Both |
| C | 20 | 60 | 1 | 0 | 1 | 0 | 1 | 0 | |
| D | 21 | 59 | 1 | 0 | 0 | 0 | 1 | 1 | Received from C & F Both |

The packet buffer for router *B* in previous slide

# Hierarchical Routing

- As networks grow in size, the routing tables grow proportionally
- Not only is router memory consumed by ever-increasing tables, but more CPU time is needed to scan them and more bandwidth is needed to send status reports about them.
- The routers are divided into what we will call **regions**.
- Two levels may not be sufficient for huge networks
- It may be necessary to group the regions into clusters, the clusters into zones, the zones into groups
- How many levels should the hierarchy have?
- For example, consider a network with 720 routers. If there is no hierarchy, each router needs 720 routing table entries. If the network is partitioned into 24 regions of 30 routers each, each router needs 30 local entries plus 23 remote entries for a total of 53 entries. If a three-level hierarchy is chosen, with 8 clusters each containing 9 regions of 10 routers, each router needs 10 entries for local routers, 8 entries for routing to other regions within its own cluster, and 7 entries for distant clusters, for a total of 25 entries.

# Hierarchical Routing




(a)

Full table for 1A

| Dest. | Line | Hops |
|---|---|---|
| 1A | – | – |
| 1B | 1B | 1 |
| 1C | 1C | 1 |
| 2A | 1B | 2 |
| 2B | 1B | 3 |
| 2C | 1B | 3 |
| 2D | 1B | 4 |
| 3A | 1C | 3 |
| 3B | 1C | 2 |
| 4A | 1C | 3 |
| 4B | 1C | 4 |
| 4C | 1C | 4 |
| 5A | 1C | 4 |
| 5B | 1C | 5 |
| 5C | 1B | 5 |
| 5D | 1C | 6 |
| 5E | 1C | 5 |

(b)

Hierarchical table for 1A

| Dest. | Line | Hops |
|---|---|---|
| 1A | – | – |
| 1B | 1B | 1 |
| 1C | 1C | 1 |
| 2 | 1B | 2 |
| 3 | 1C | 2 |
| 4 | 1C | 3 |
| 5 | 1C | 4 |

(c)

Hierarchical routing.

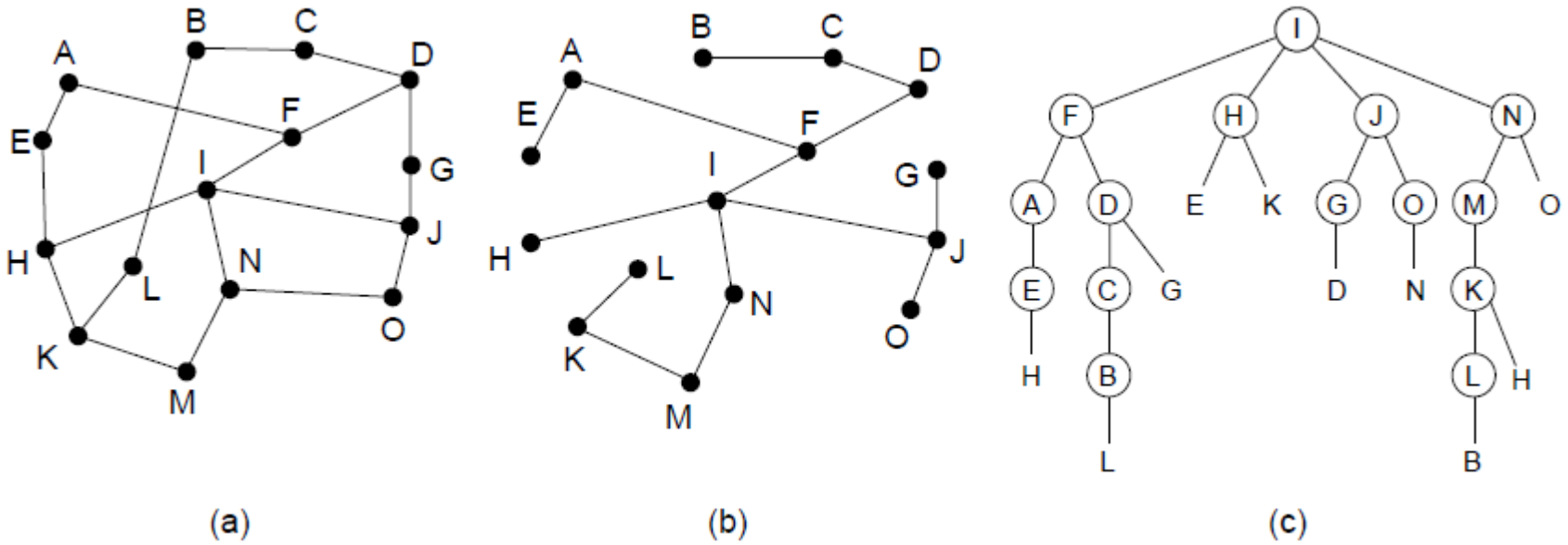
# Broadcast Routing

- Sending a packet to all destinations simultaneously is called **broadcasting**
- One broadcasting method that requires no special features from the network is for the source to simply send a distinct packet to each destination.
- An improvement is **multidestination routing**, in which each packet contains either a list of destinations or a bit map indicating the desired destinations.
- When a packet arrives at a router, the router checks all the destinations to determine the set of output lines that will be needed. The router generates a new copy of the packet for each output line to be used and includes in each packet only those destinations that are to use the line
- A better broadcast routing technique is **flooding**

# Broadcast Routing – Reverse Path Forwarding

- When a broadcast packet arrives at a router, the router checks to see if the packet arrived on the link that is normally used for sending packets *toward* the source of the broadcast.
- If so, there is an excellent chance that the broadcast packet itself followed the best route from the router and is therefore the first copy to arrive at the router.
- This being the case, the router forwards copies of it onto all links except the one it arrived on.
- If, however, the broadcast packet arrived on a link other than the preferred one for reaching the source, the packet is discarded as a likely duplicate.

# Broadcast Routing

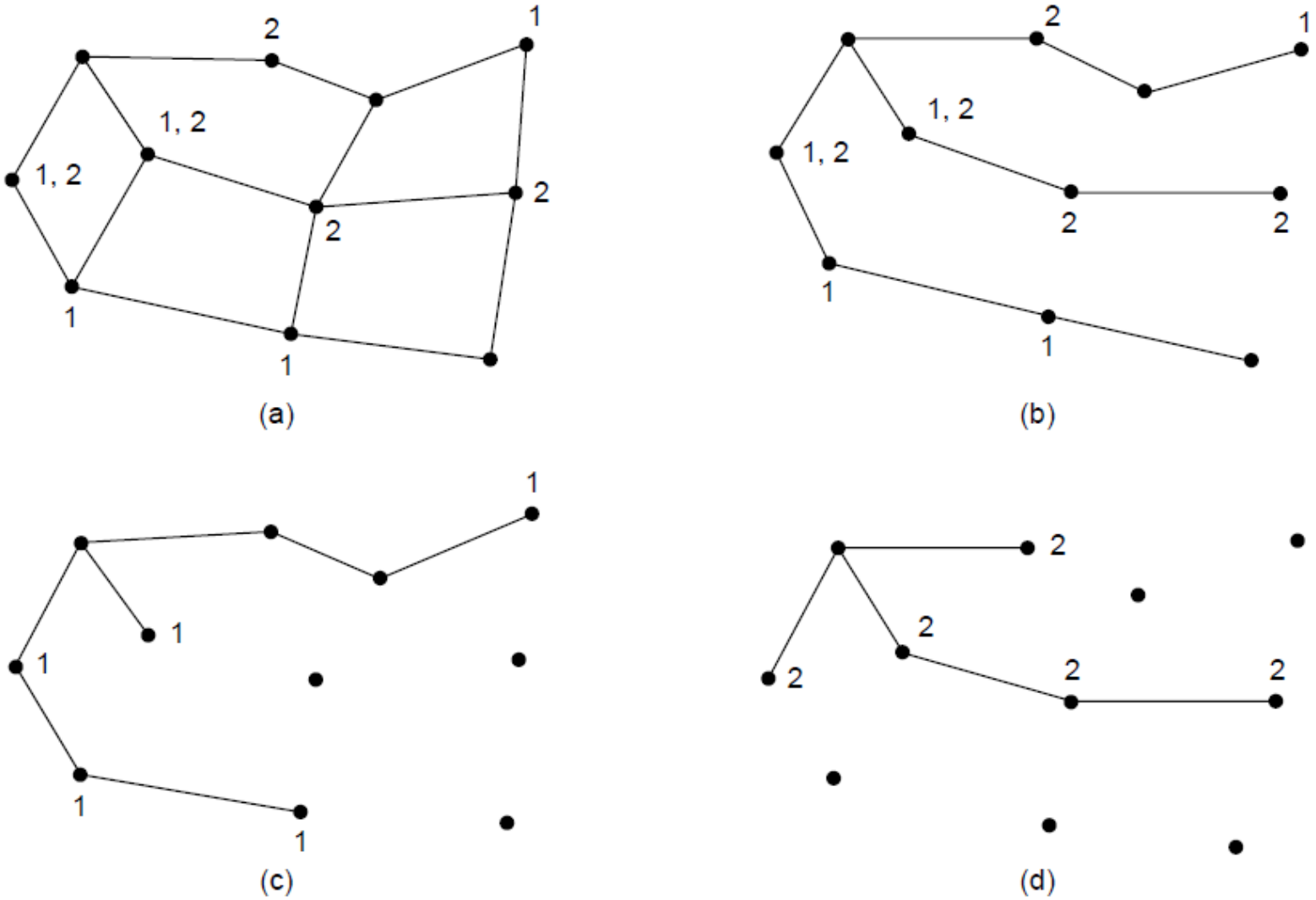



Reverse path forwarding. (a) A network. (b) A sink tree for I. (c) The tree built by reverse path forwarding.

# Multicast Routing

- Multicasting is a special case of broadcasting
- All multicasting schemes require some way to create and destroy groups and to identify which routers are members of a group
- Assume that each group is identified by a multicast address and that routers know the groups to which they belong
- If the group is dense, broadcast is a good start but it reaches all routers. The solution is pruning the broadcast tree by removing links that do not lead to members
- If OSPF is used, each routers knows complete topology, they can construct its own pruned spanning tree for each sender to the group. Eg. **MOSPF** (**Multicast OSPF**)

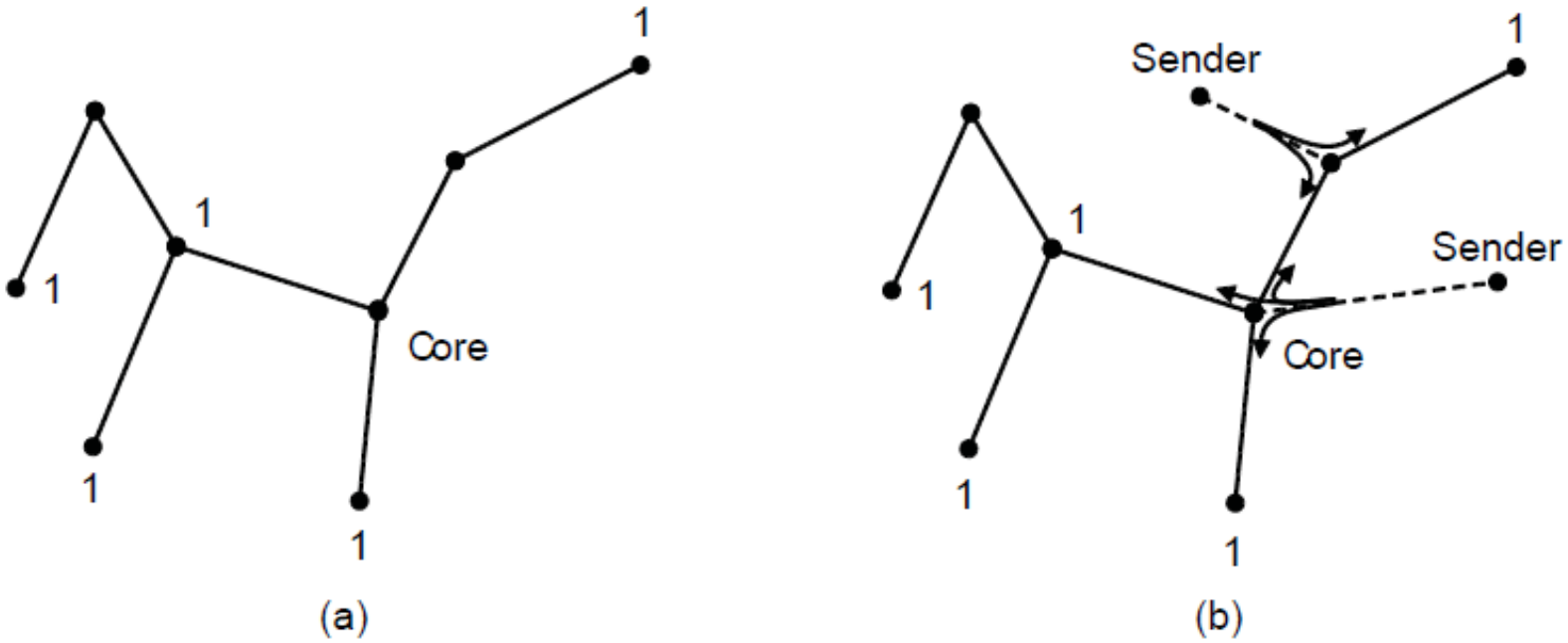
# Multicast Routing (1)




(a) A network. (b) A spanning tree for the leftmost router. (c) A multicast tree for group 1. (d) A multicast tree for group 2.

# Multicast Routing (2)

- All of the routers agree on a root (called the **core** or **rendezvous point**)
- Build the tree by sending a packet from each member to the root
- The tree is the union of the paths traced by these packets
- To send to this group, a sender sends a packet to the core

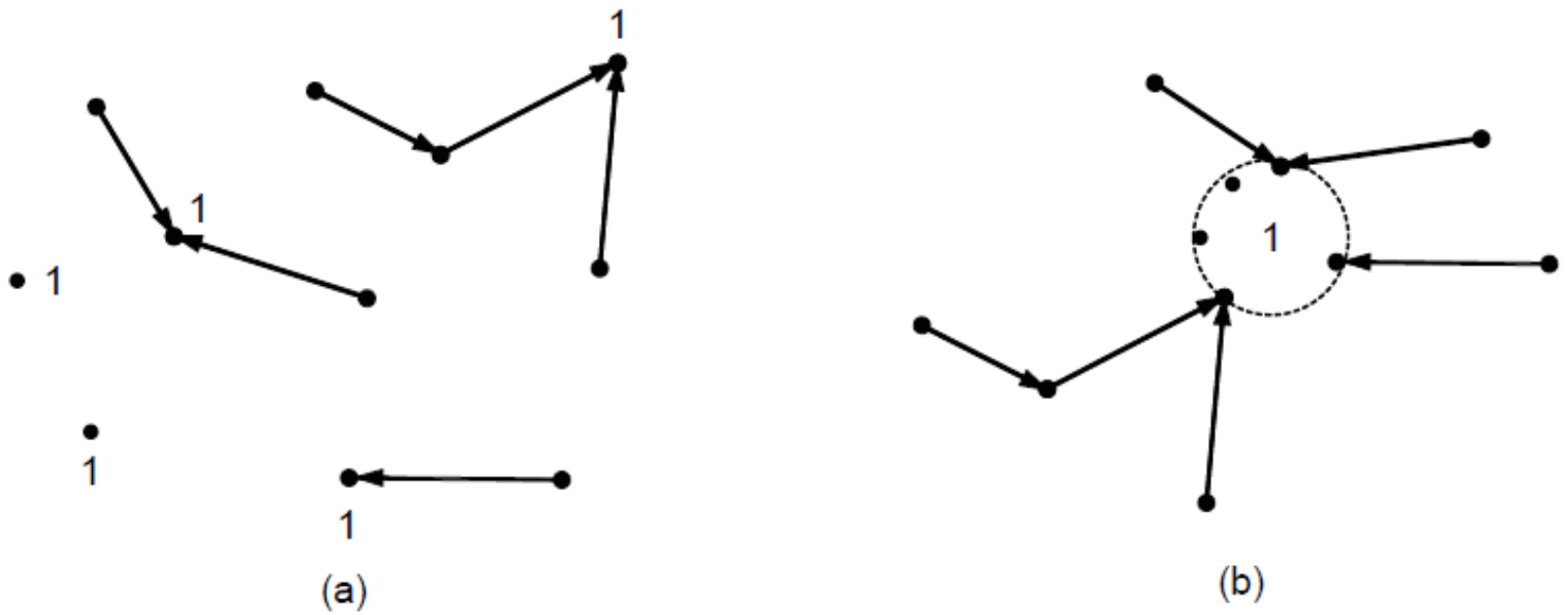



a) Core-based tree for group 1.

b) Sending to group 1.

# Anycast Routing

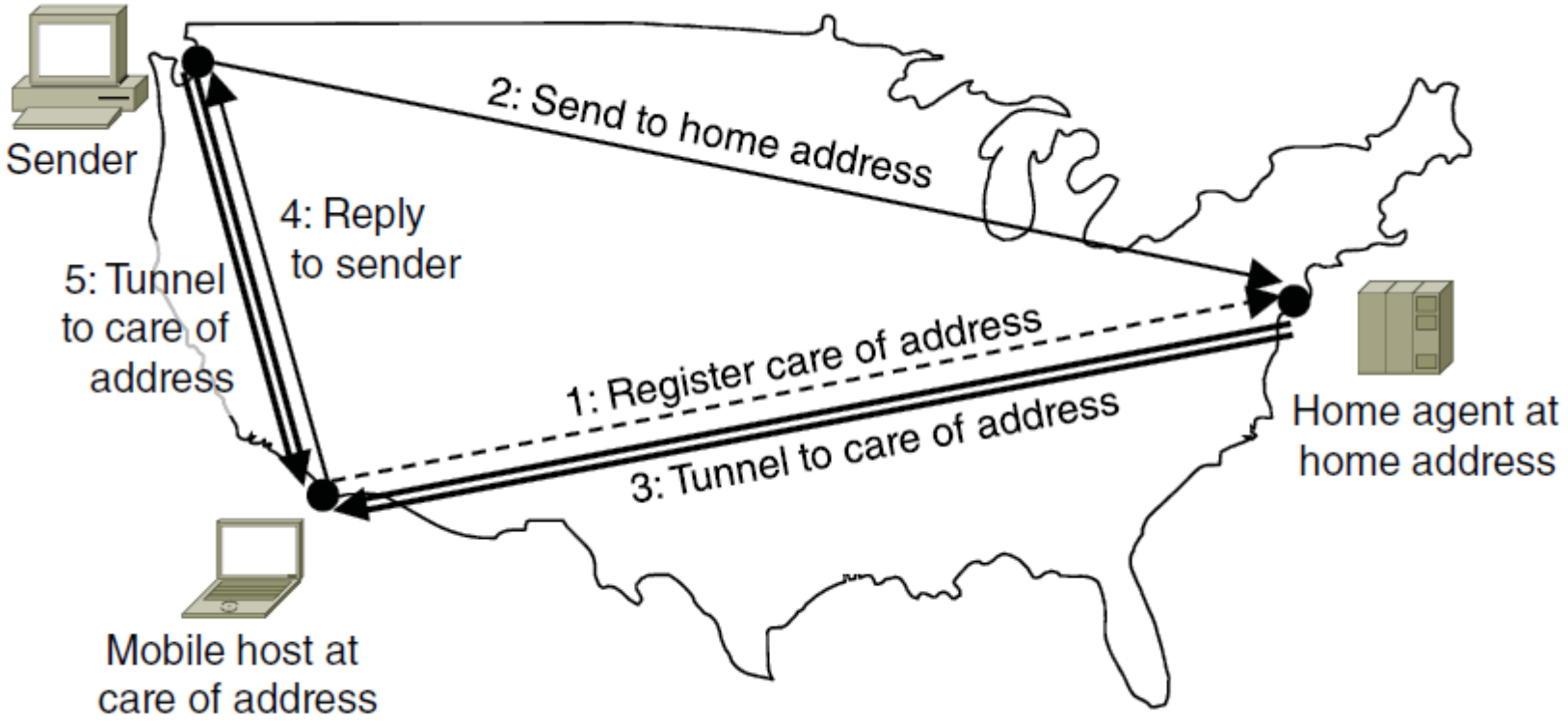
- A packet is delivered to the nearest member of a group
- Can be achieved using regular DV or LSR
- All members of the group will be given same address, say 1
- This procedure works because the routing protocol does not realize that there are multiple instances of destination 1. That is, it believes that all the instances of node 1 are the same node




a) Anycast routes to group 1.

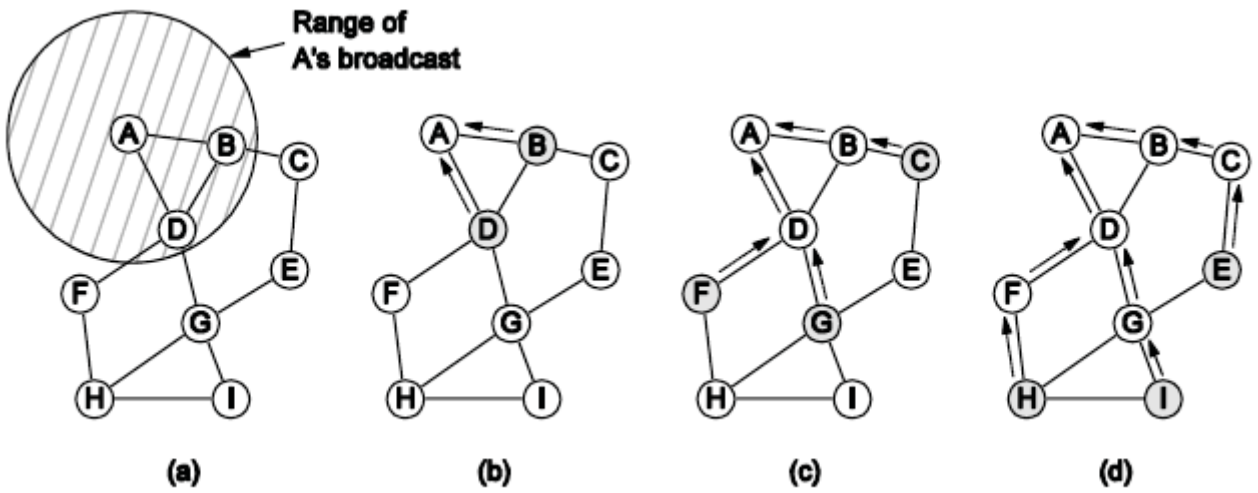
b) Topology seen by the routing protocol.

# Routing for Mobile Hosts




Packet routing for mobile hosts

# Routing in Ad Hoc Networks




a) Range of A's broadcast.
b) After B and D receive it.
c) After C, F, and G receive it.
d) After E, H, and I receive it.

The shaded nodes are new recipients. The dashed lines show possible reverse routes. The solid lines show the discovered route.

# Congestion Control Algorithms (1)

- Approaches to congestion control
- Traffic-aware routing
- Admission control
- Traffic throttling
- Load shedding

# Congestion Control

Too many packets present in the network causes packet delay and loss that degrades performance. This situation is called **congestion**.

The network and transport layers share the responsibility for handling congestion.

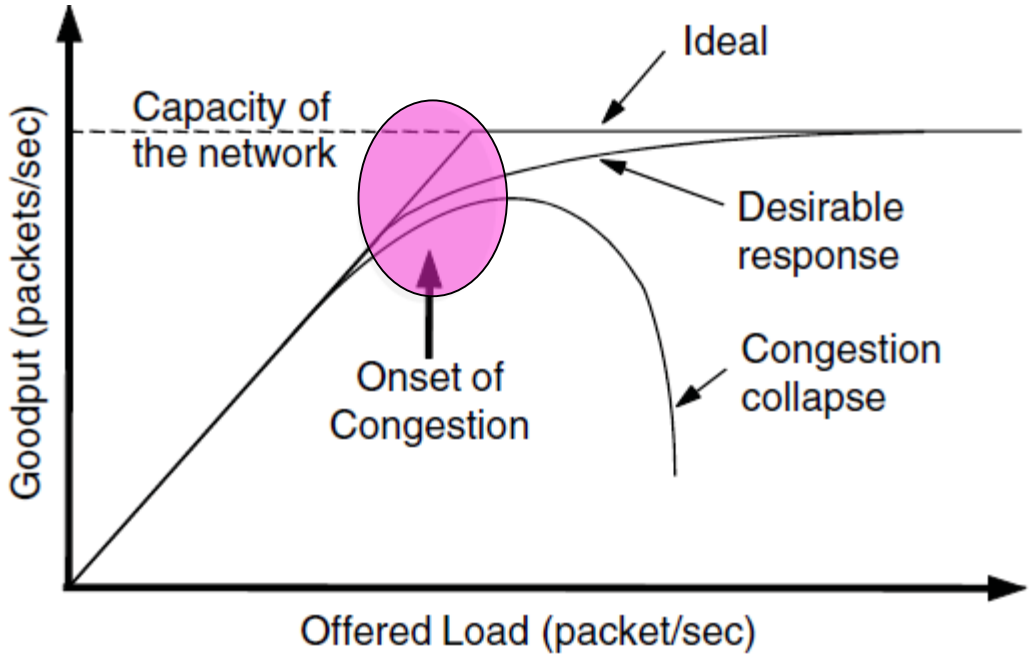
Since congestion occurs within the network, it is the network layer that directly experiences it and must ultimately determine what to do with the excess packets.

However, the most effective way to control congestion is to reduce the load that the transport layer is placing on the network. This requires the network and transport layers to work together.

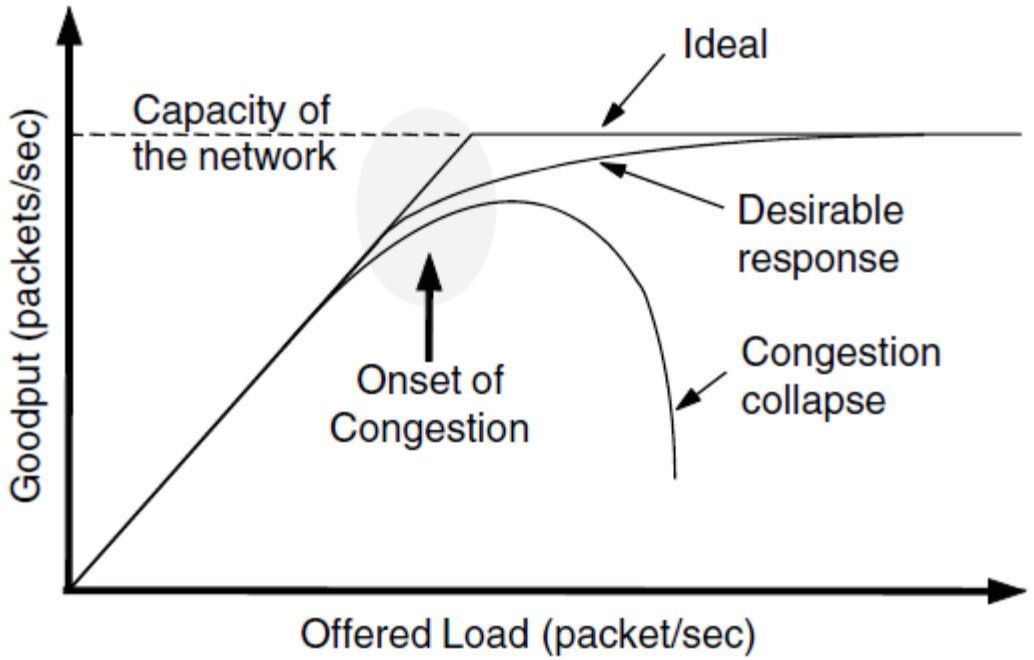
# Congestion Control (2)

Congestion results when too much traffic is offered; performance degrades due to loss/retransmissions

- Goodput (useful packets) trails offered load



# Congestion Control Algorithms (2)




When too much traffic is offered, congestion sets in and performance degrades sharply.

# Congestion Control (1)

Handling congestion is the responsibility of the Network and Transport layers working together

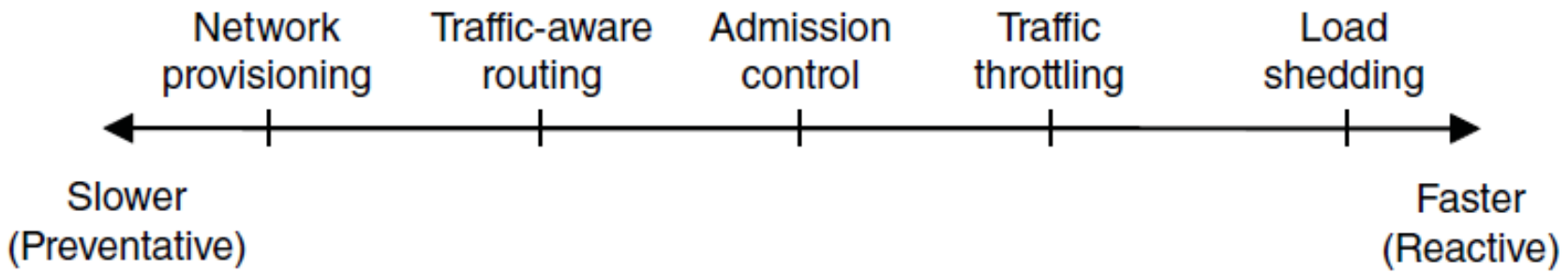
- We look at the Network portion here

– Network provisioning »
– Traffic-aware routing »
– Admission control »
– Traffic throttling »
– Load shedding »

# Congestion Control (3) – Approaches

Network must do its best with the offered load

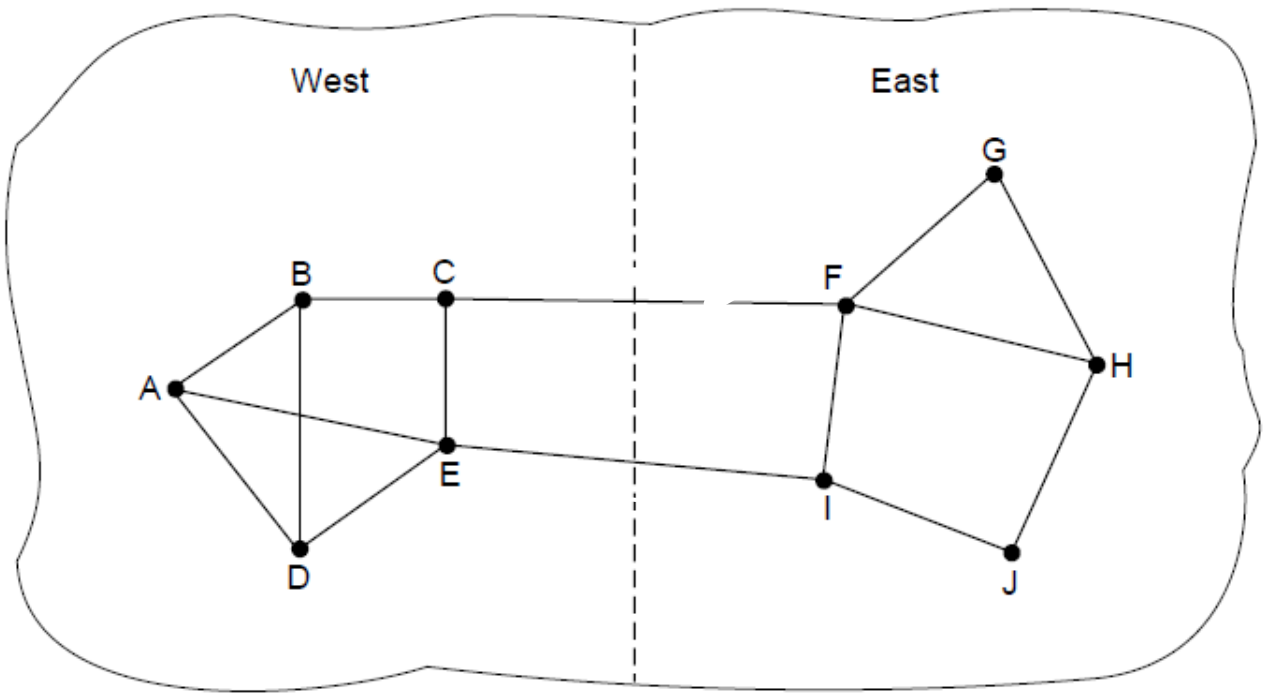
- Different approaches at different timescales
- Nodes should also reduce offered load (Transport)



# Traffic-Aware Routing

Choose routes depending on traffic, not just topology

- E.g., use *EI* for West-to-East traffic if *CF* is loaded

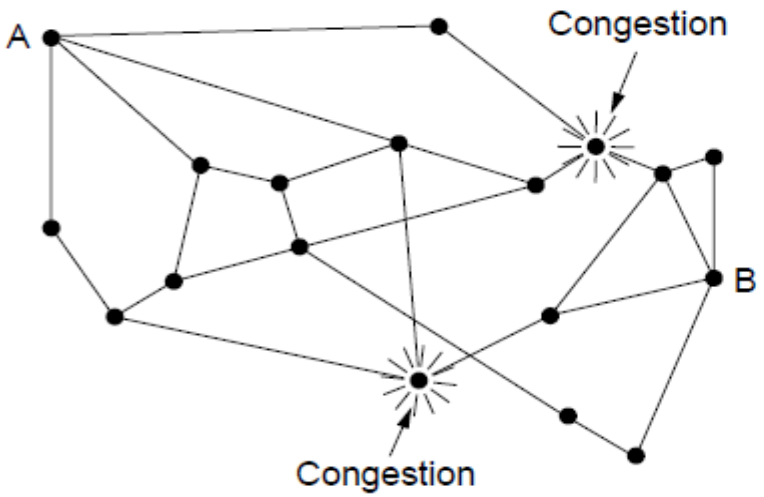


# Admission Control

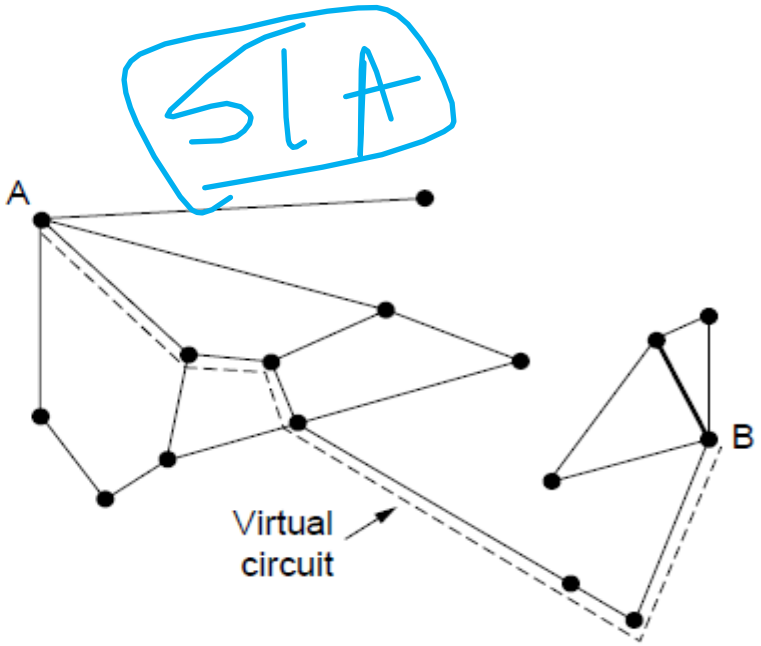
a) Admission control allows a new traffic load only if the network has sufficient capacity.

b) new connections can be refused if they would cause the network to become

c) congested.




Network with some congested nodes



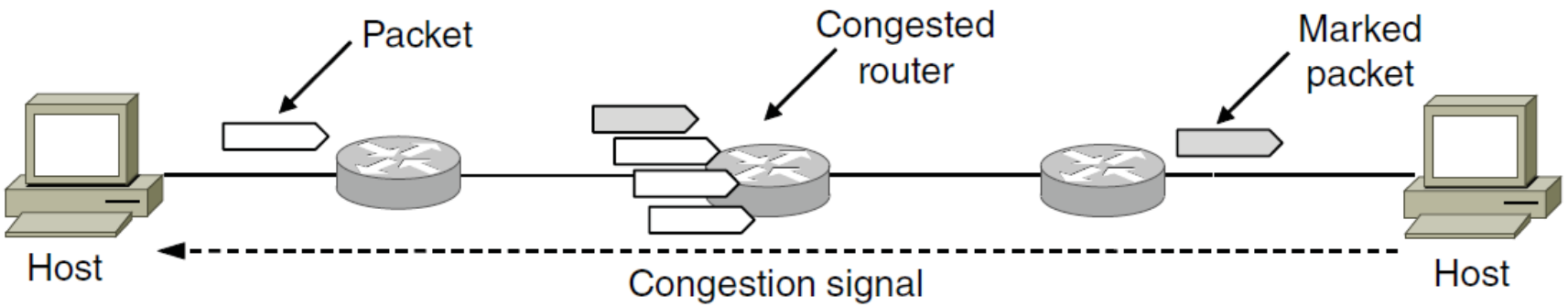

Uncongested portion and route *AB* around congestion

# Traffic Throttling

When the network delivers the packet, the destination can note that there is congestion and inform the sender when it sends a reply packet.

The sender can then throttle its transmissions as before.

ECN (Explicit Congestion Notification) marks packets and receiver returns signal to sender



# Choke Packet Technique

Choke packet scheme is mechanism where each link is monitored to examine how much utilization is taking place.

If the utilization goes beyond a certain threshold limit, the link goes to a warning and a special packet, called choke packet is sent to the source.

On receiving the choke packet, the source reduced the traffic in order to avoid congestion.

# Choke Packet Technique

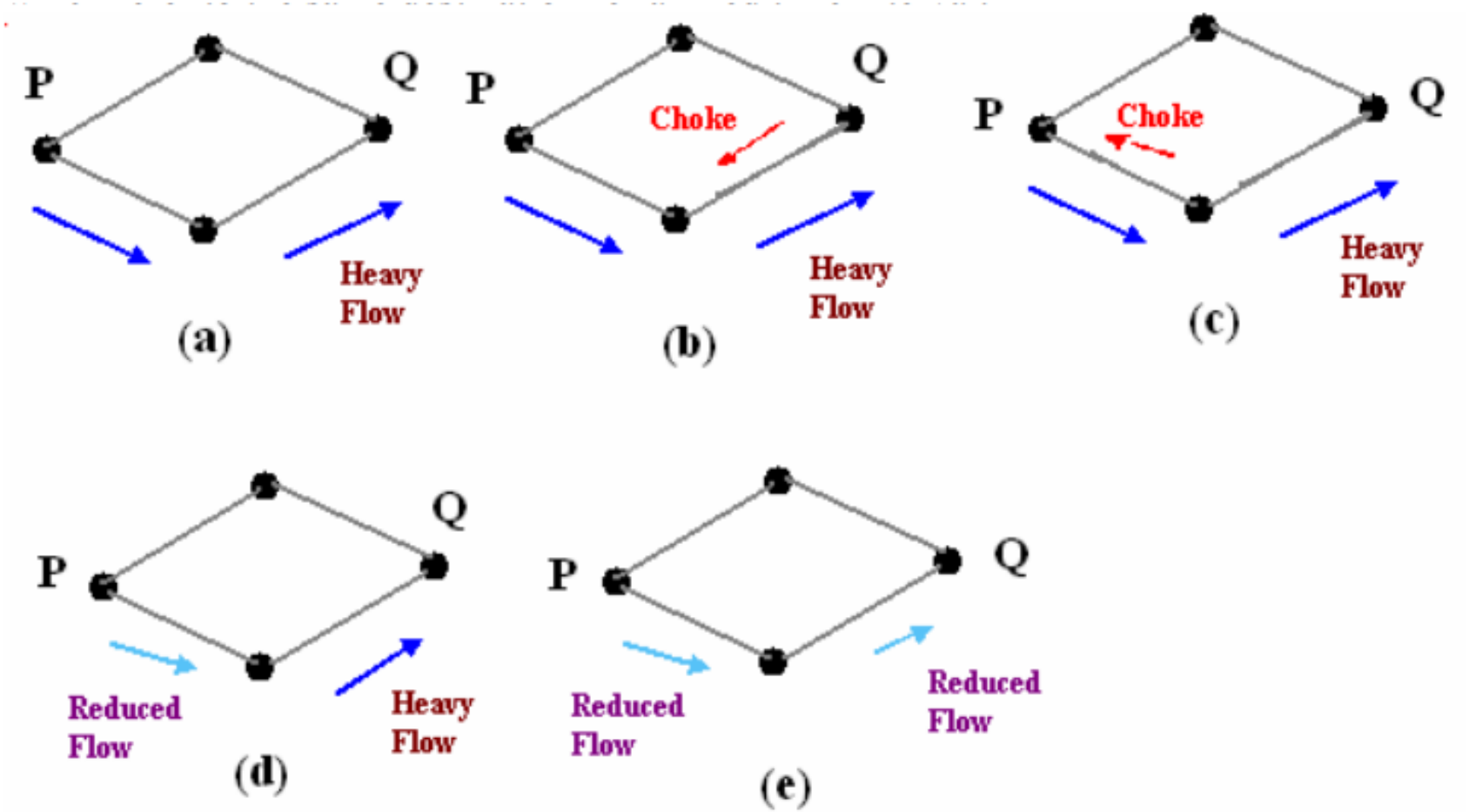



Figure 7.5.6 Depicts the functioning of choke packets, (a) Heavy traffic between nodes P and Q, (b) Node Q sends the Choke packet to P, (c) Choke packet reaches P, (d) P reduces the flow and send a reduced flow out, (e) Reduced flow reaches node Q

# Hop-by Hop Choke Packets

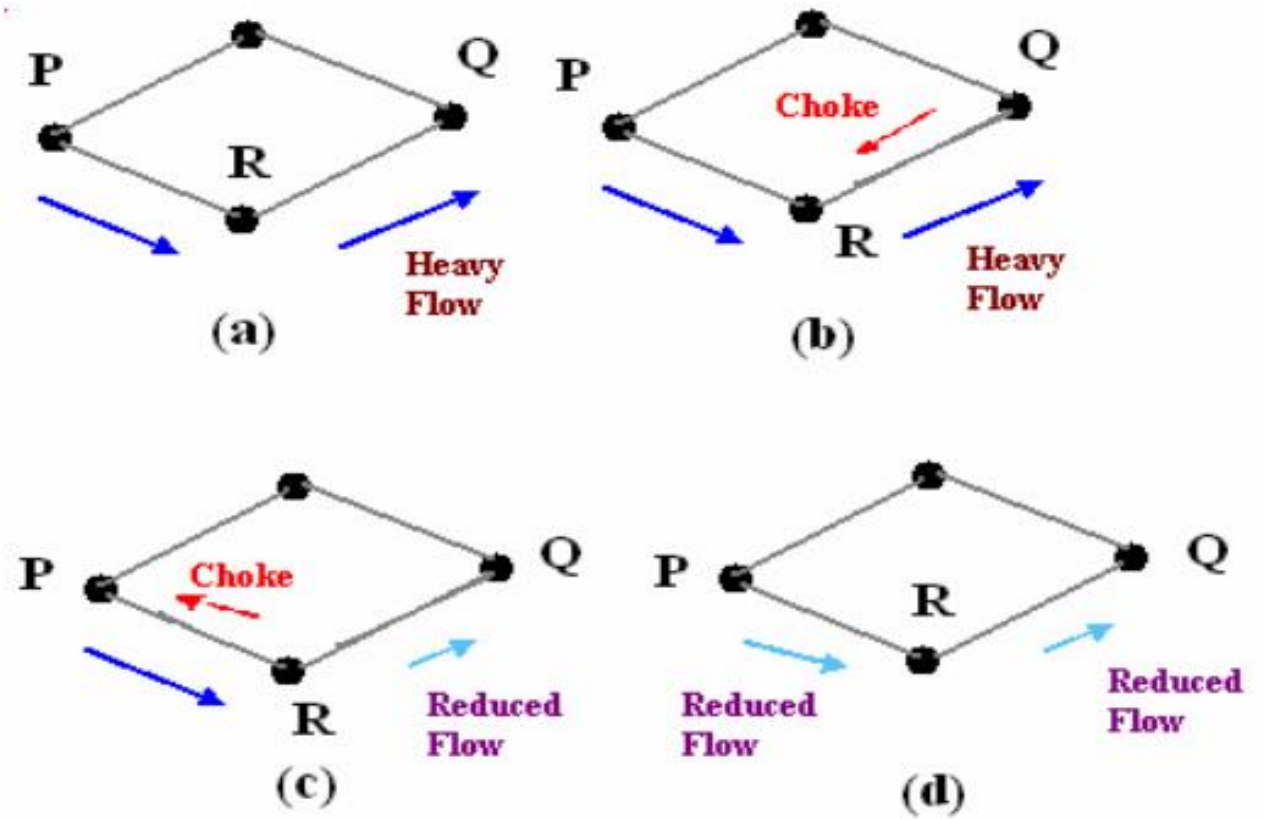



Figure 7.5.7 Depicts the functioning of Hop-by-Hop choke packets, (a) Heavy traffic between nodes P and Q, (b) Node Q sends the Choke packet to P, (c) Choke packet reaches R, and the flow between R and Q is curtail down, Choke packer reaches P, and P reduces the flow out

# Quality of Service (QoS)

- Application requirements
- Traffic shaping
- Packet scheduling
- Admission control
- Integrated services
- Differentiated services

# Application Requirements (1)

| Application | Bandwidth | Delay | Jitter | Loss |
|---|---|---|---|---|
| Email | Low | Low | Low | Medium |
| File sharing | High | Low | Low | Medium |
| Web access | Medium | Medium | Low | Medium |
| Remote login | Low | Medium | Medium | Medium |
| Audio on demand | Low | Low | High | Low |
| Video on demand | High | Low | High | Low |
| Telephony | Low | High | High | Low |
| Videoconferencing | High | High | High | Low |

How stringent the quality-of-service requirements are.

# Categories of QoS and Examples

a) Constant bit rate

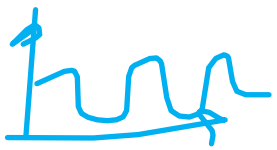
- Telephony

b) Variable bit rate

1. Real-time variable bit rate
   - Compressed videoconferencing
2. Non-real-time variable bit rate
   - Watching a movie on demand

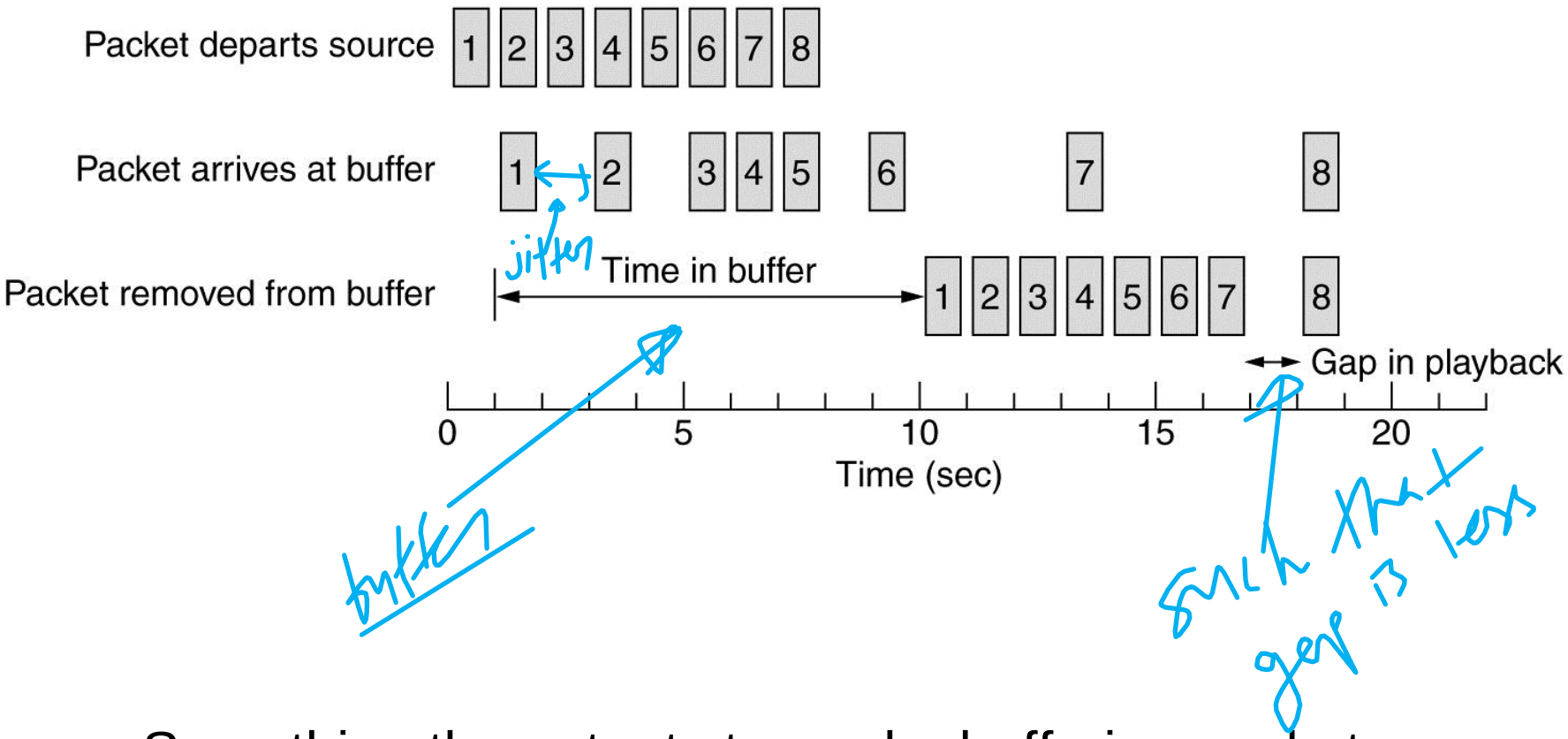
c) Available bit rate

- File transfer



# Buffering

jitter is variable delay b/w bits




Smoothing the output stream by buffering packets.

Real time application buffering

buffering is to nullify jitter

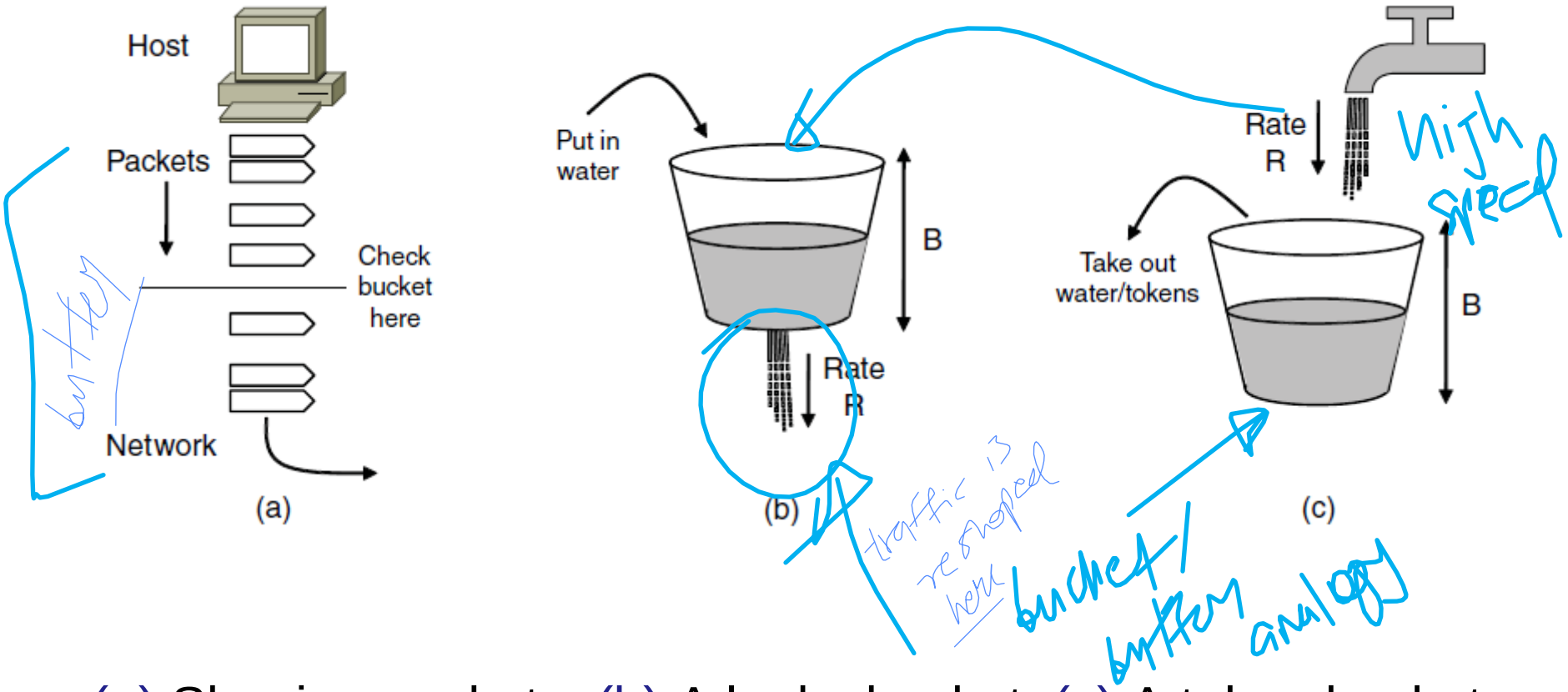
# Traffic Shaping

- Before the network can make QoS guarantees, it must know what traffic is being guaranteed
- In CBR, like telephone it is quite easy eg. needs 64 kbps
- **Traffic shaping** is a technique for regulating the average rate and burstiness of a flow of data that enters the network
- It allow applications to transmit a wide variety of traffic that suits their needs, including some bursts, yet have a simple and useful way to describe the possible traffic patterns to the network

bursts are also controlled

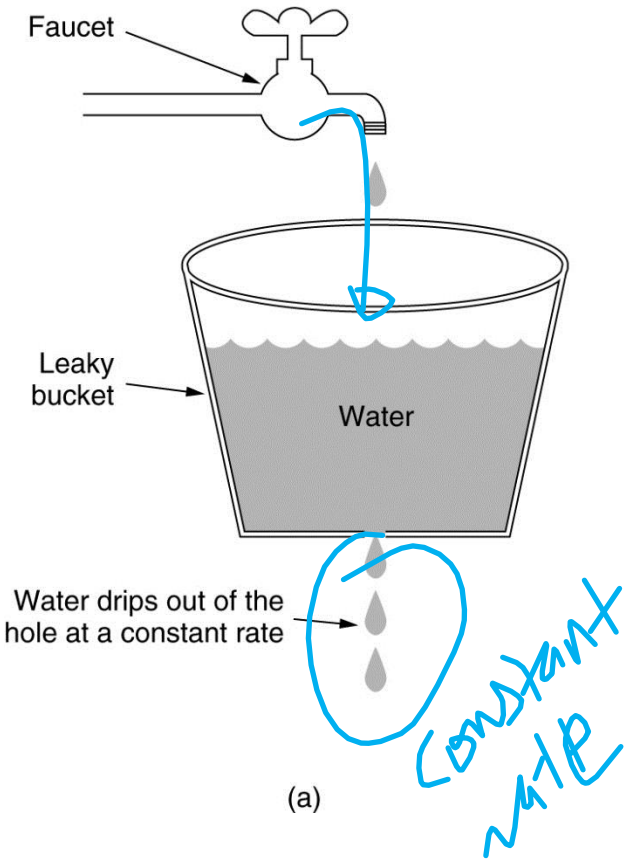
- Traffic shaping reduces congestion and thus helps the network live up to its promise
- **SLA** (**Service Level Agreement**), is made over aggregate flows and long periods of time
- Monitoring a traffic flow is called **traffic policing**

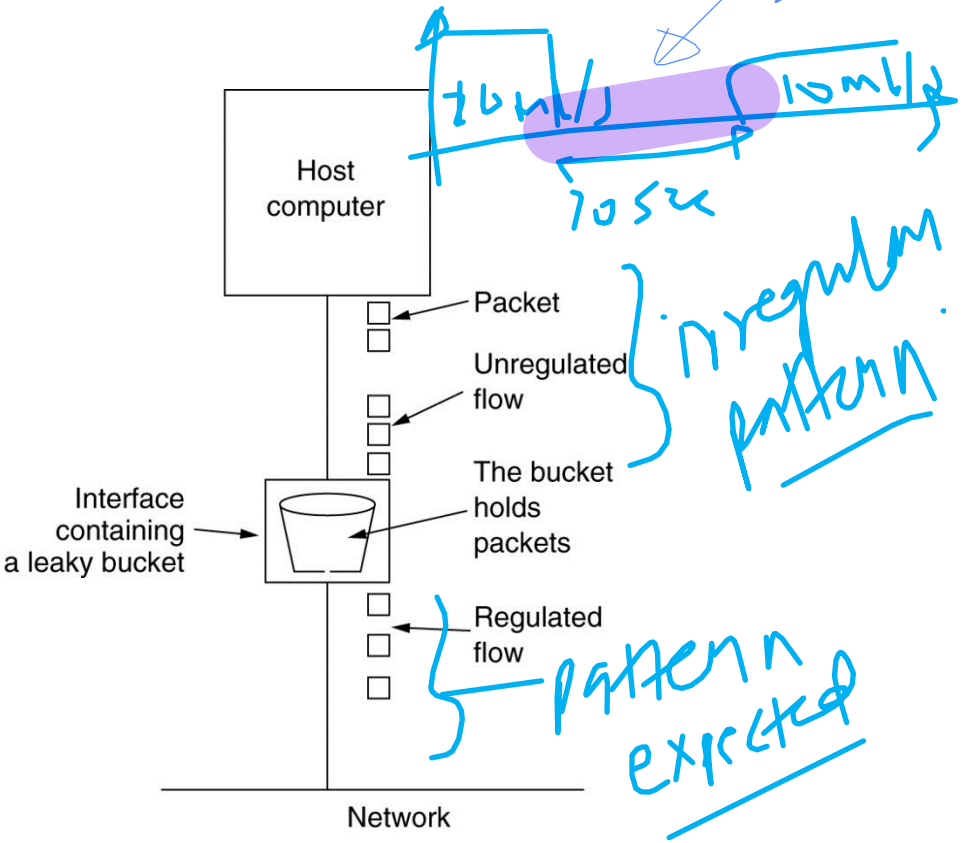
# Traffic Shaping (1)




(a) Shaping packets. (b) A leaky bucket. (c) A token bucket

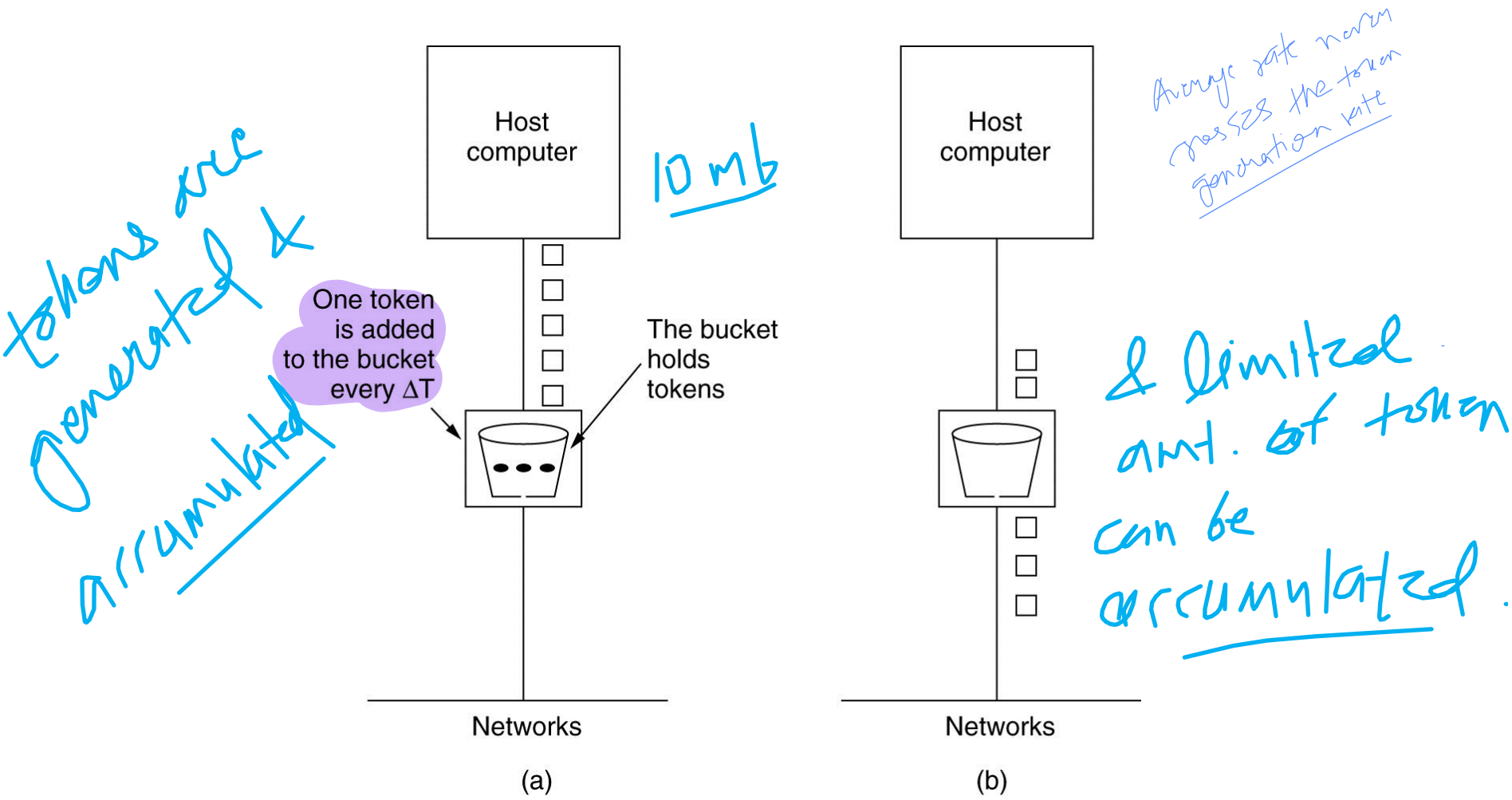
# The Leaky Bucket Algorithm



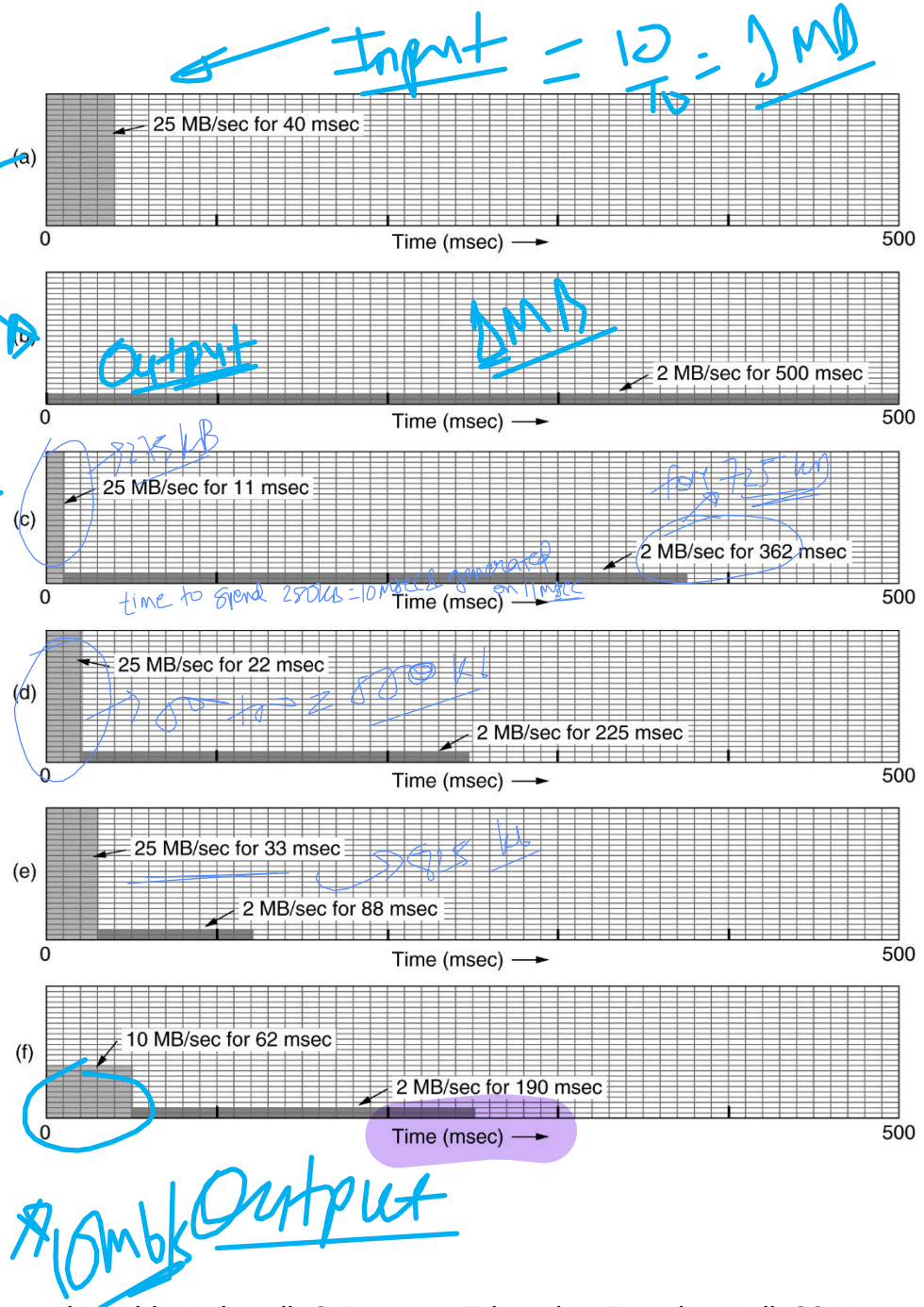

(a) A leaky bucket with water. (b) a leaky bucket with packets.

# The Token Bucket Algorithm



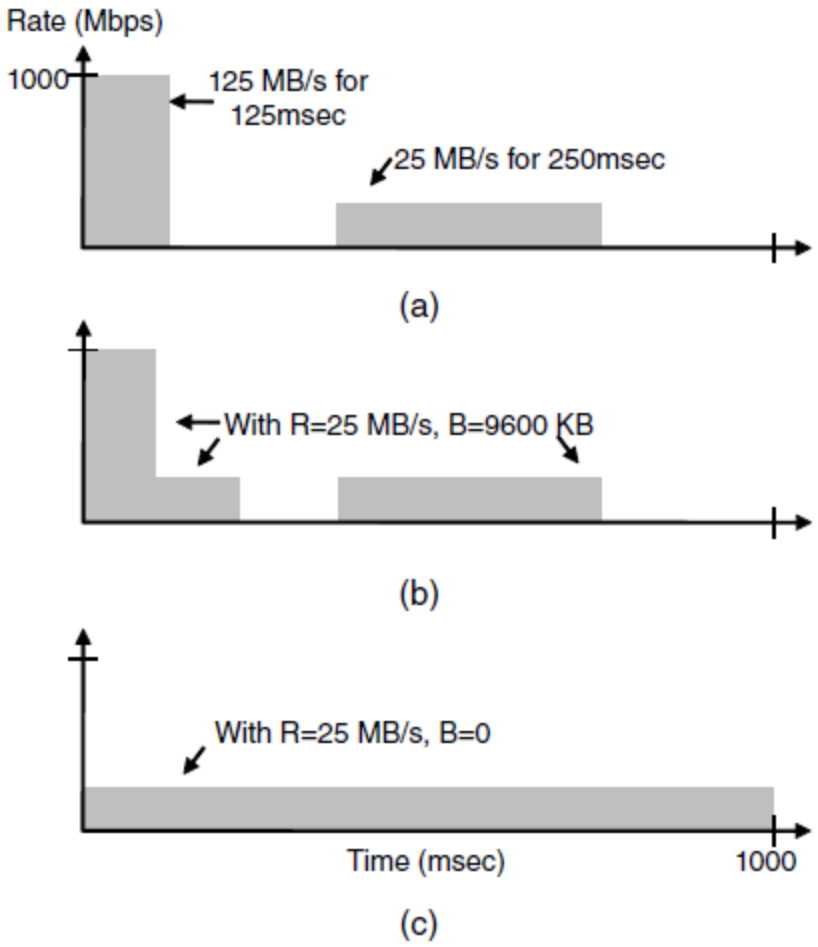

(a) Before. (b) After.

# The Leaky Bucket Algorithm



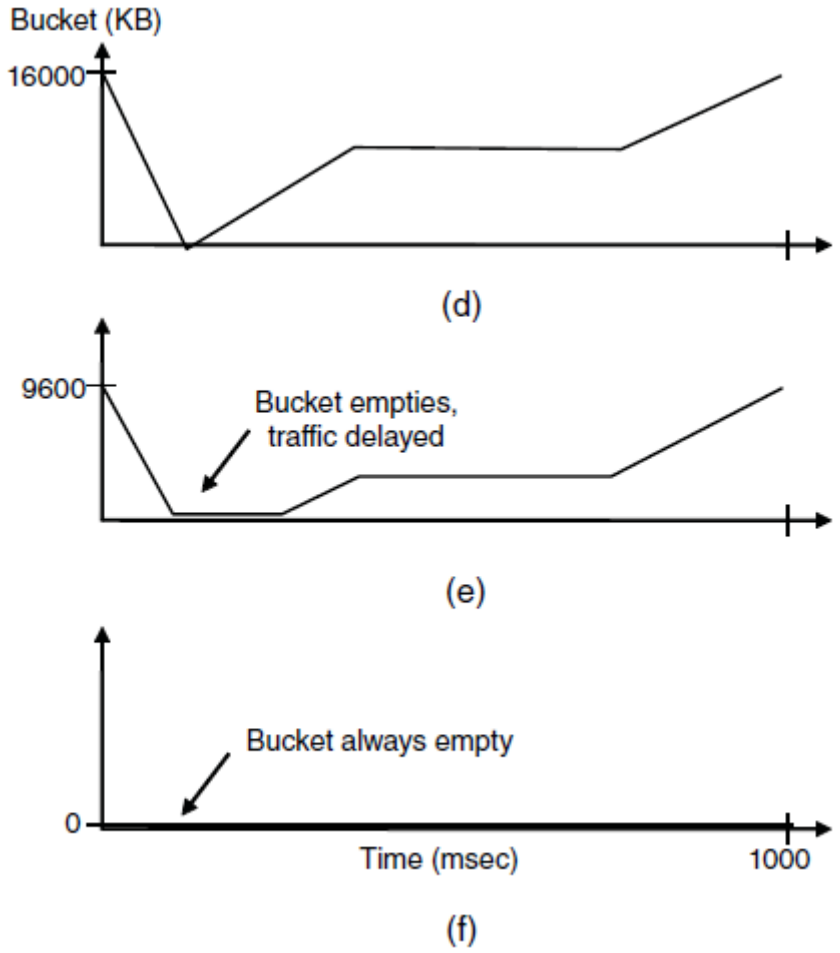

(a) Input to a leaky bucket. (b) Output from a leaky bucket. Output from a token bucket with capacities of (c) 250 KB, (d) 500 KB, (e) 750 KB, (f) Output from a 500KB token bucket feeding a 10-MB/sec leaky bucket.

# Traffic Shaping (2)




(a) Traffic from a host. Output shaped by a token bucket of rate 200 Mbps and capacity (b) 9600 KB, (c) 0 KB.

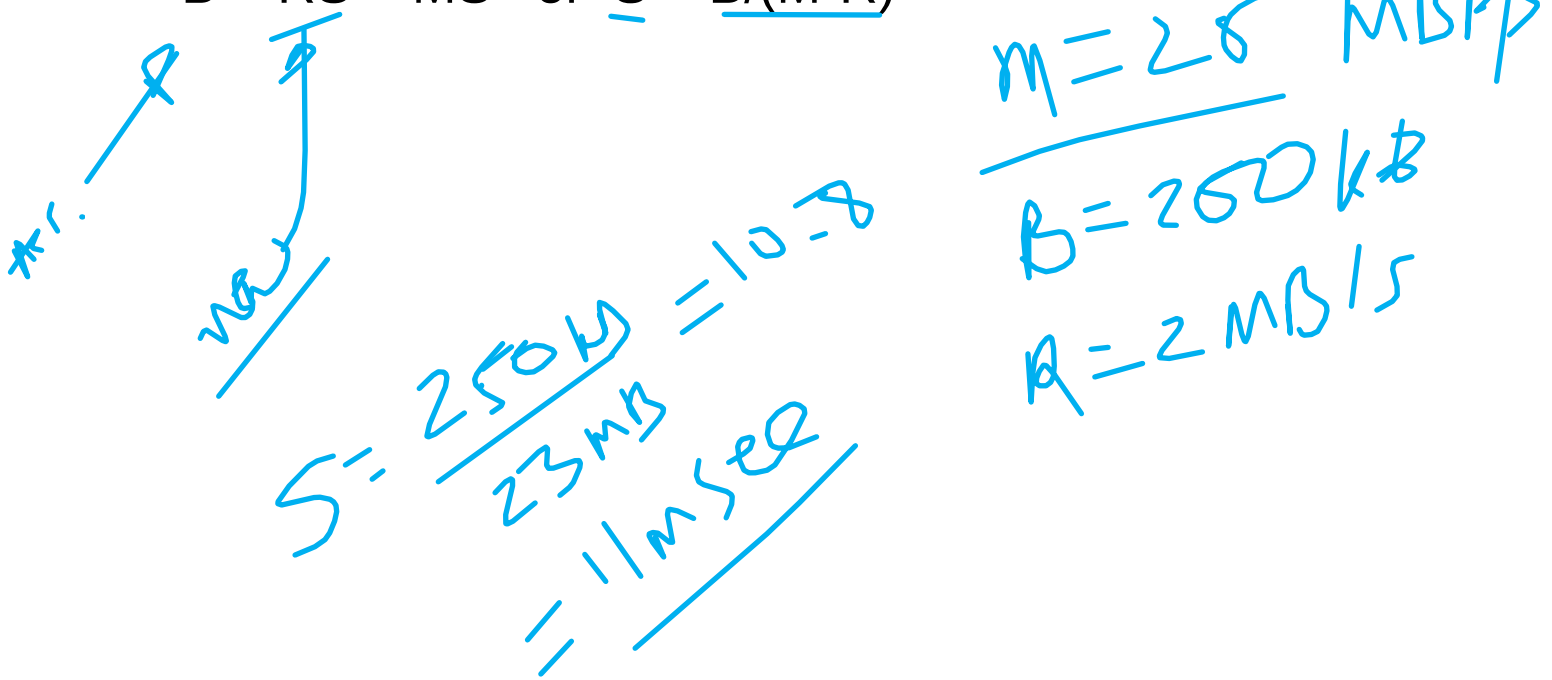
# Traffic Shaping (3)




Token bucket level for shaping with rate 200 Mbps and capacity (d) 16000 KB, (e) 9600 KB, and (f) 0KB..

# Traffic Shaping

- Calculate the length of the maximum burst
- If burst length is *S* sec., the maximum output rate *M* bytes/sec, the token bucket capacity *B* bytes, and the token arrival rate *R* bytes/sec, the output burst contains a maximum of *B* + *RS* bytes which gets transferred at M bytes/sec for *S* sec
- B + RS = MS or S = B/(M-R)

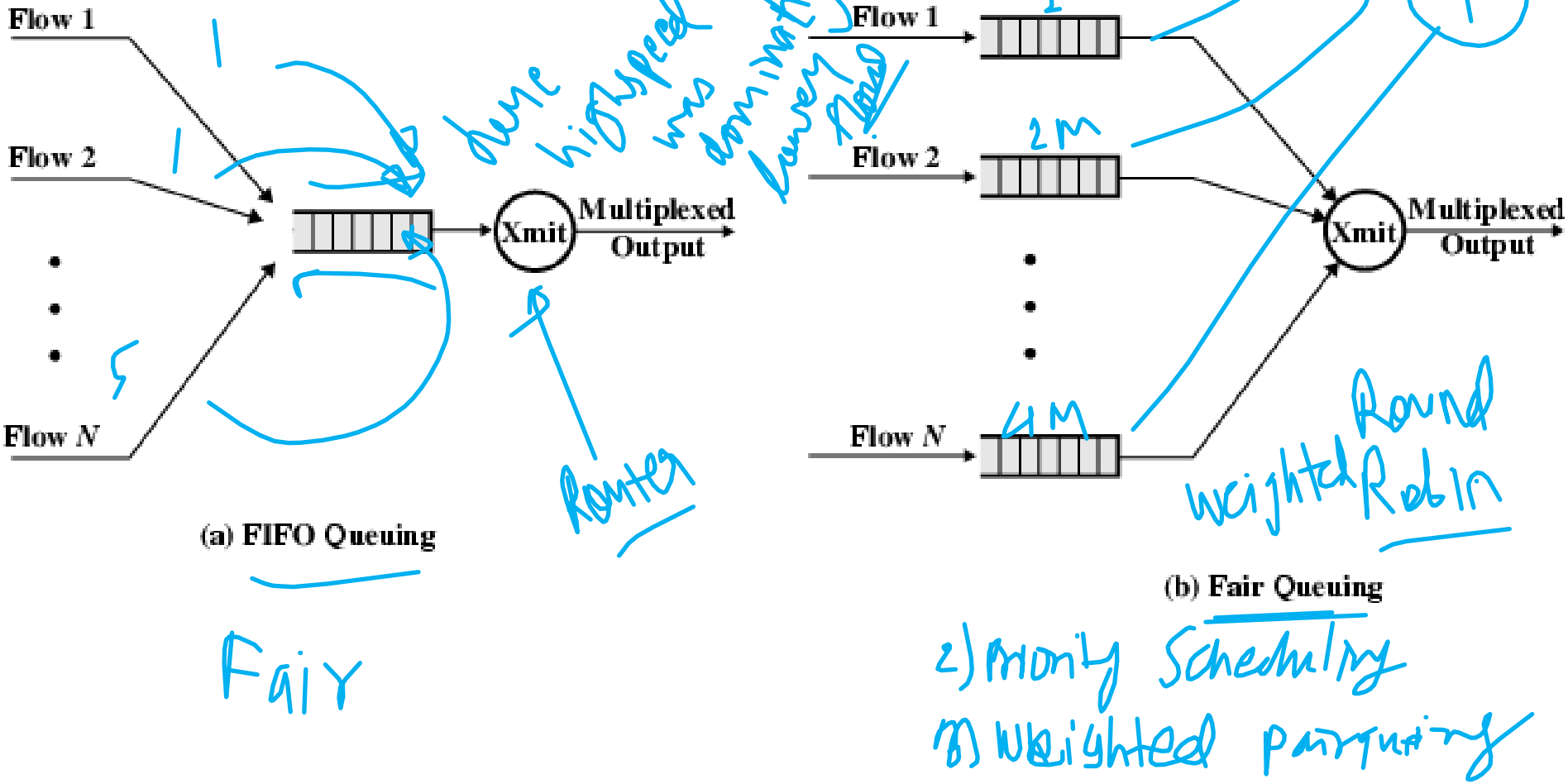


# Packet Scheduling

Kinds of resources can potentially be reserved for different flows:

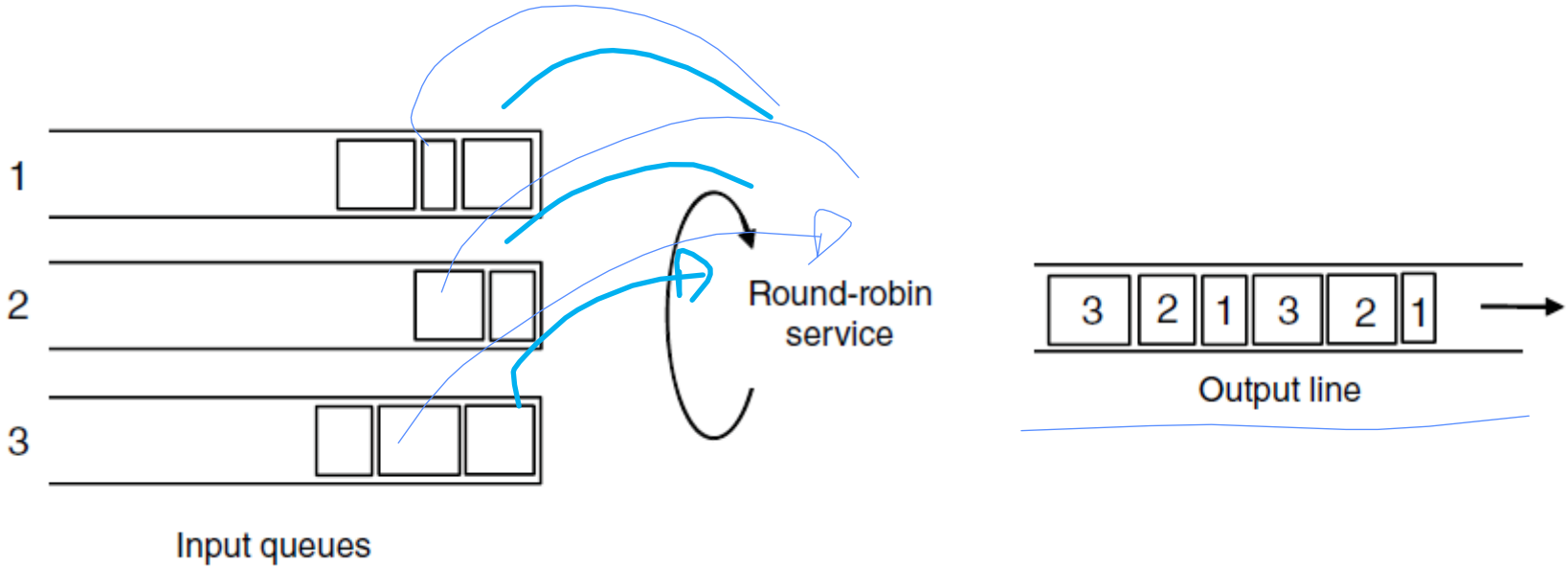
- Bandwidth. } reserved b/w
- Buffer space.
- CPU cycles. } processing power.

# Packet Scheduling



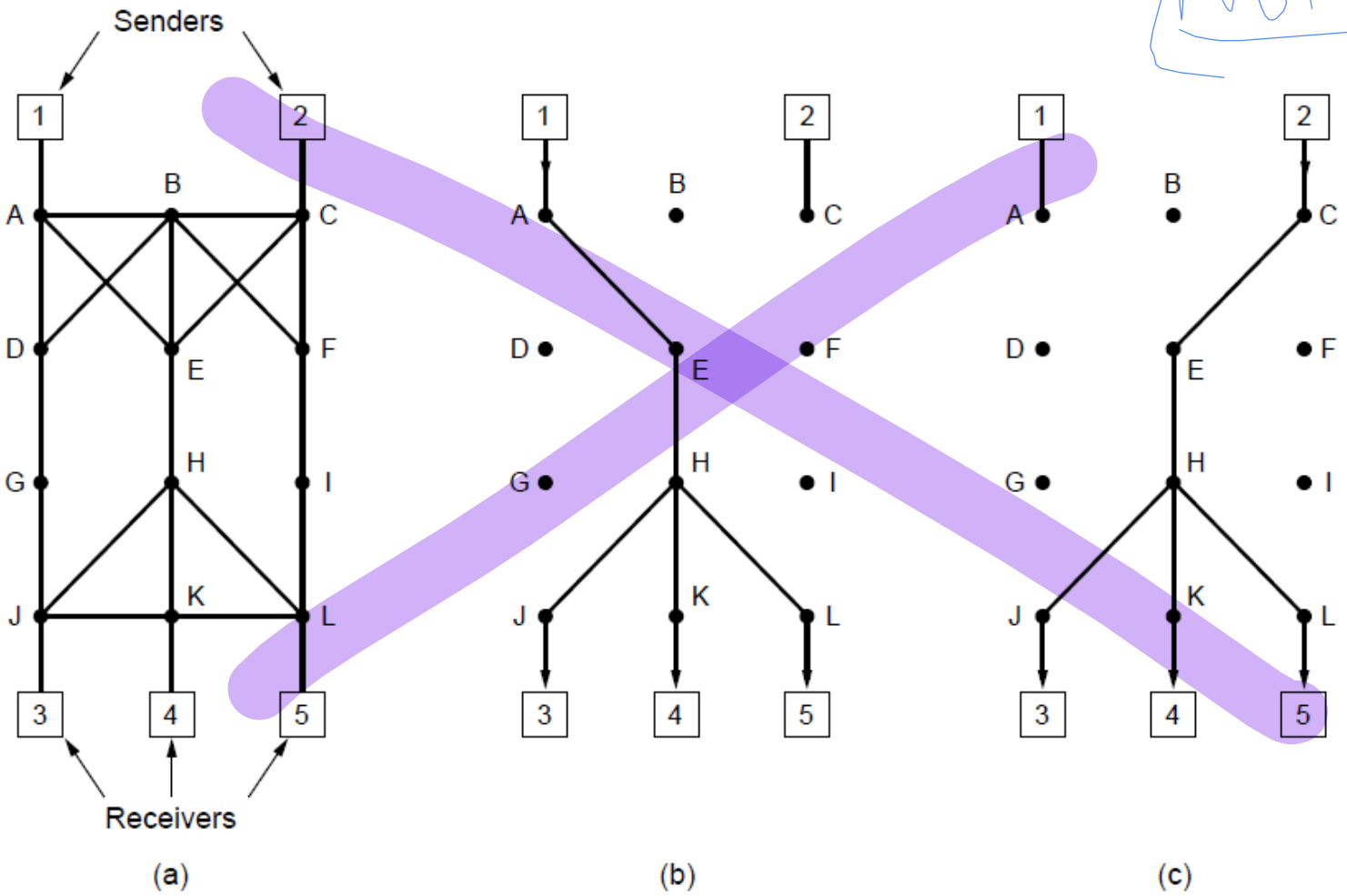

*Computer Networks*, Fifth Edition by Andrew Tanenbaum and David Wetherall, © Pearson Education-Prentice Hall, 2011

# Packet Scheduling



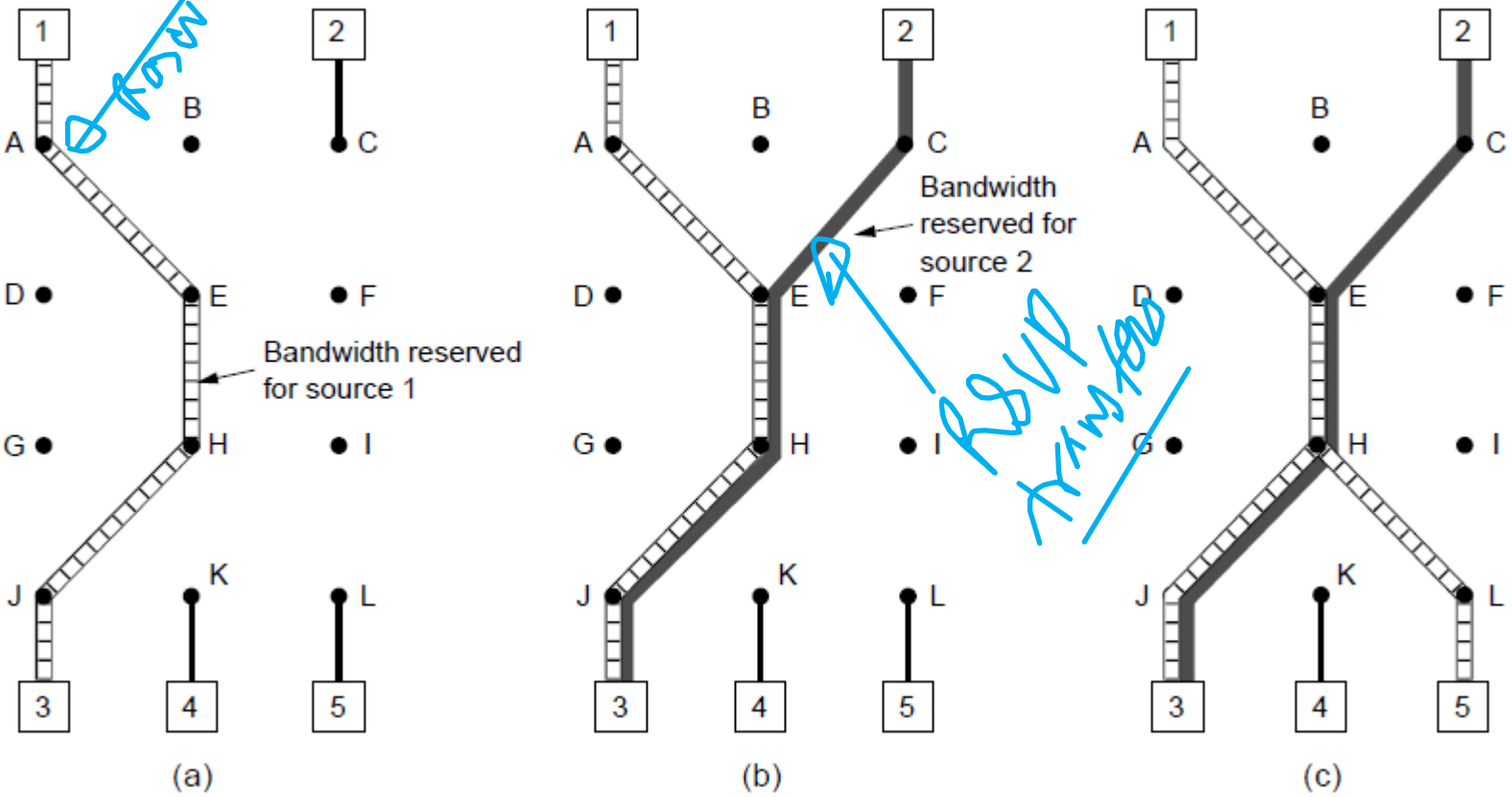

Round-robin Fair Queuing

# Integrated Services (1)



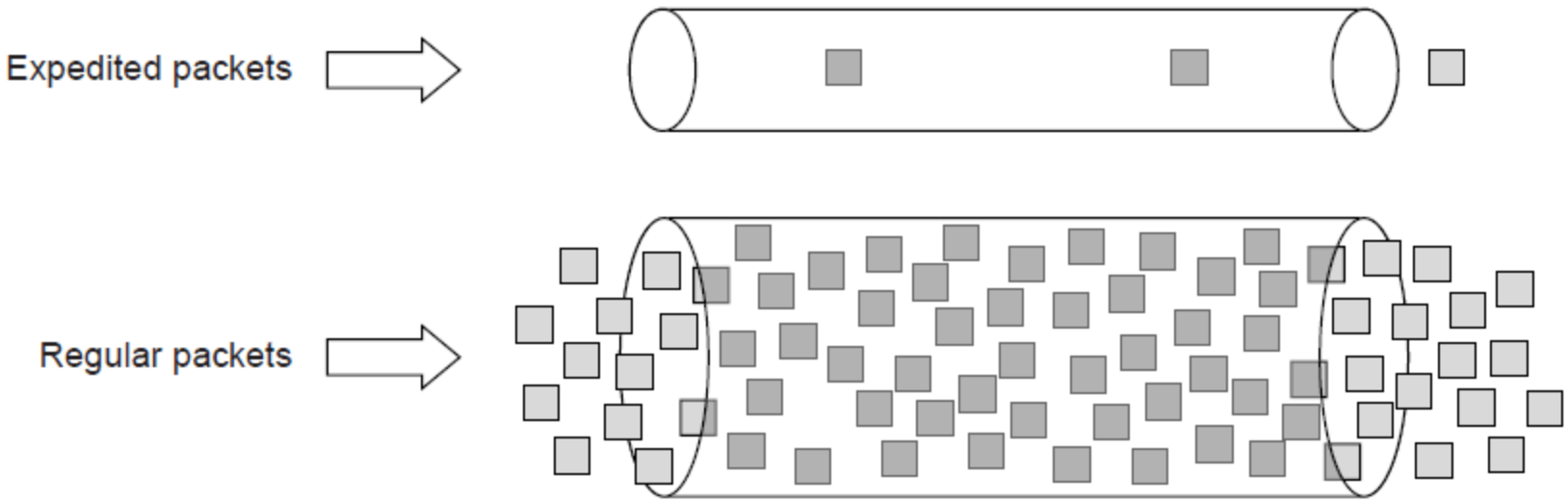

(a) A network. (b) The multicast spanning tree for host 1. (c) The multicast spanning tree for host 2.

# Integrated Services (2)




(a) Host 3 requests a channel to host 1. (b) Host 3 then requests a second channel, to host 2. (c) Host 5 requests a channel to host 1.

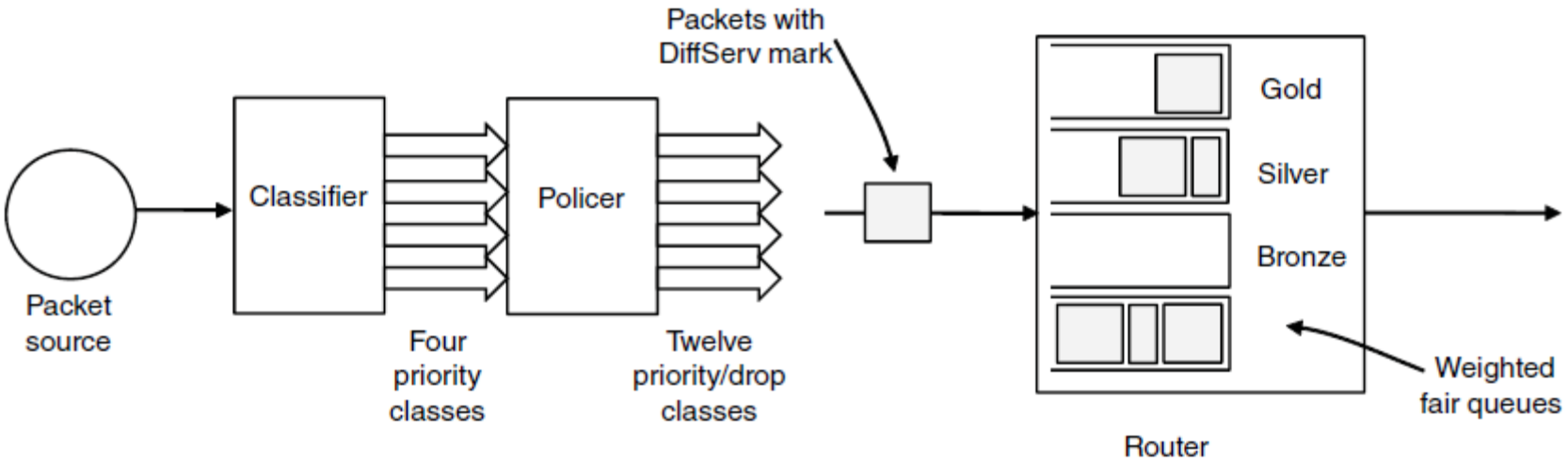
# Differentiated Services (1)




classify the packets & forward.

Expedited packets experience a traffic-free network

# Differentiated Services (2)




A possible implementation of assured forwarding

# Internetworking

Internet is collection of variety of network. IP, ATM, etc....
So Interconnection b/w networks can be done by tunnelling

- How networks differ
- How networks can be connected
- Tunneling
- Internetwork routing
- Packet fragmentation

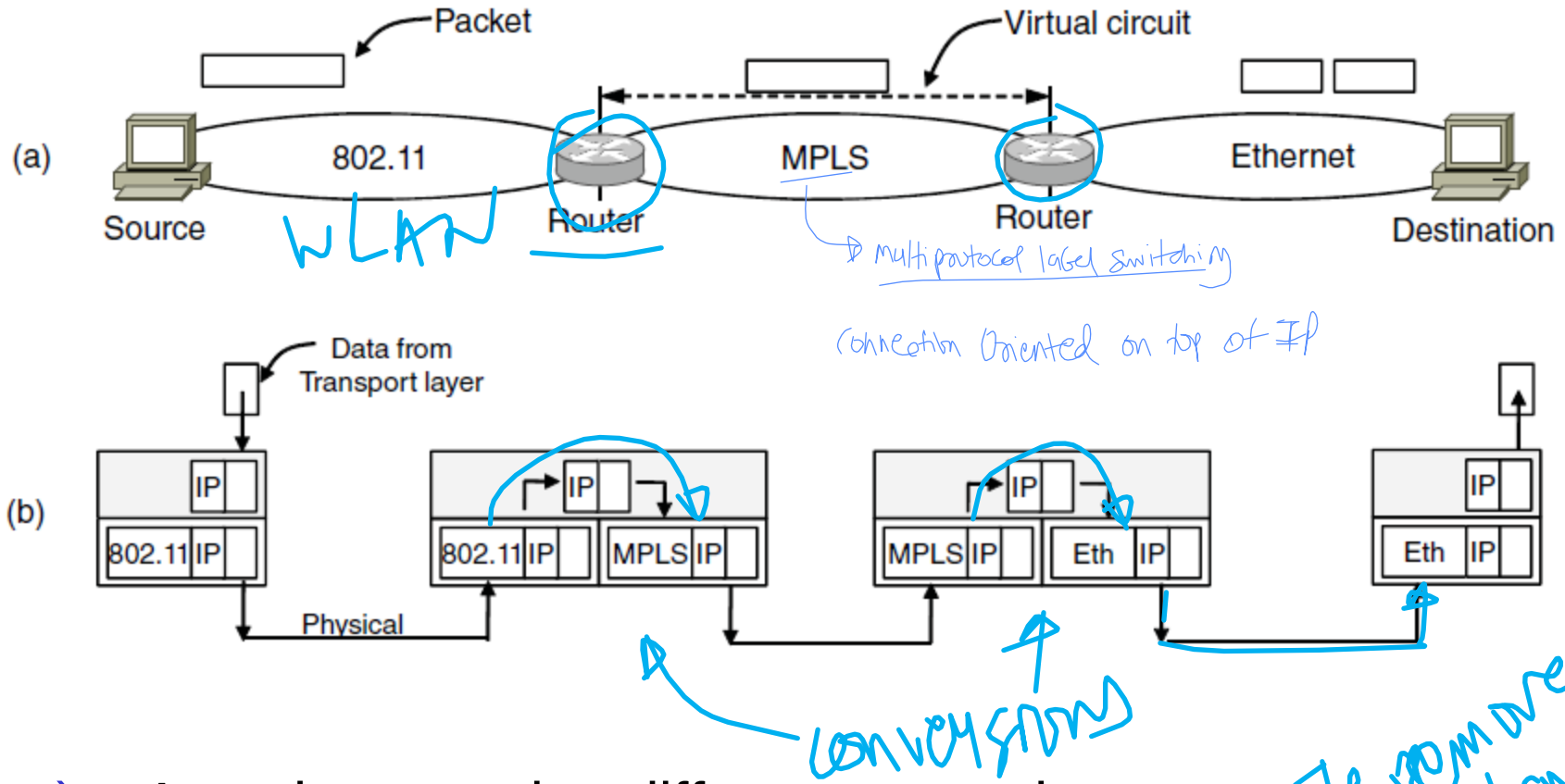
Solution

# How Networks Differ

| Item | Some Possibilities |
|---|---|
| Service offered | Connectionless versus connection oriented |
| Addressing | Different sizes, flat or hierarchical |
| Broadcasting | Present or absent (also multicast) |
| Packet size | Every network has its own maximum |
| Ordering | Ordered and unordered delivery |
| Quality of service | Present or absent; many different kinds |
| Reliability | Different levels of loss |
| Security | Privacy rules, encryption, etc. |
| Parameters | Different timeouts, flow specifications, etc. |
| Accounting | By connect time, packet, byte, or not at all |

Some of the many ways networks can differ

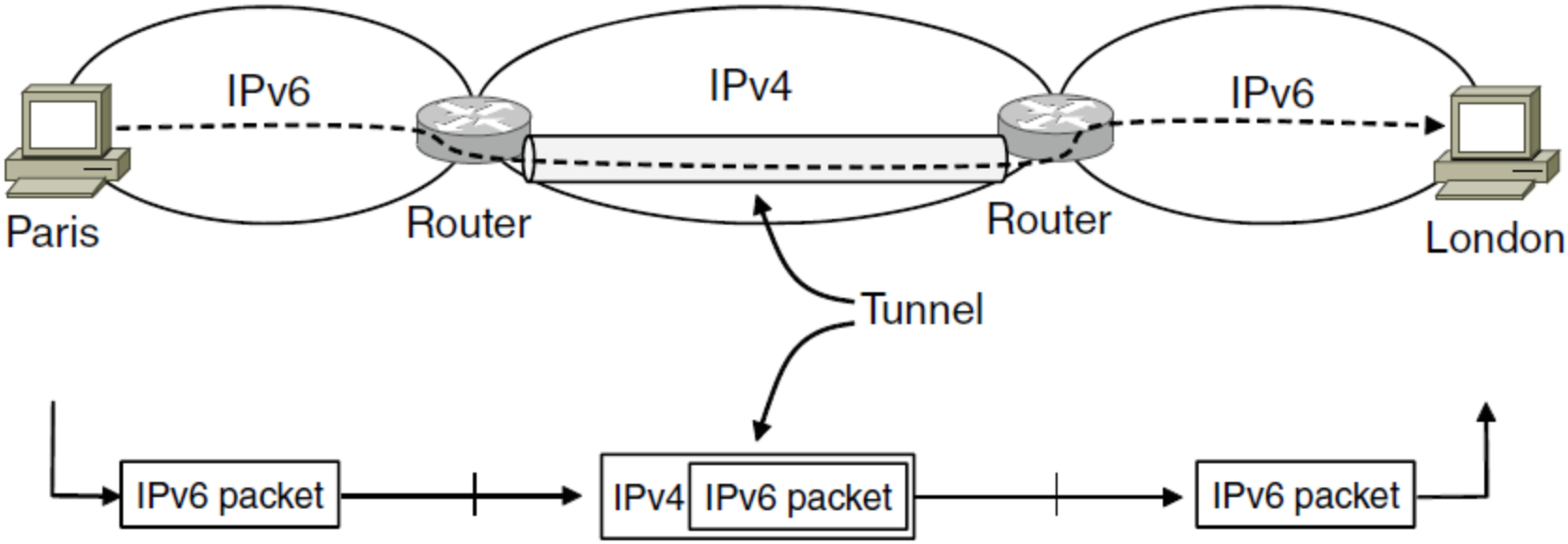
# How Networks Can Be Connected




a) A packet crossing different networks.

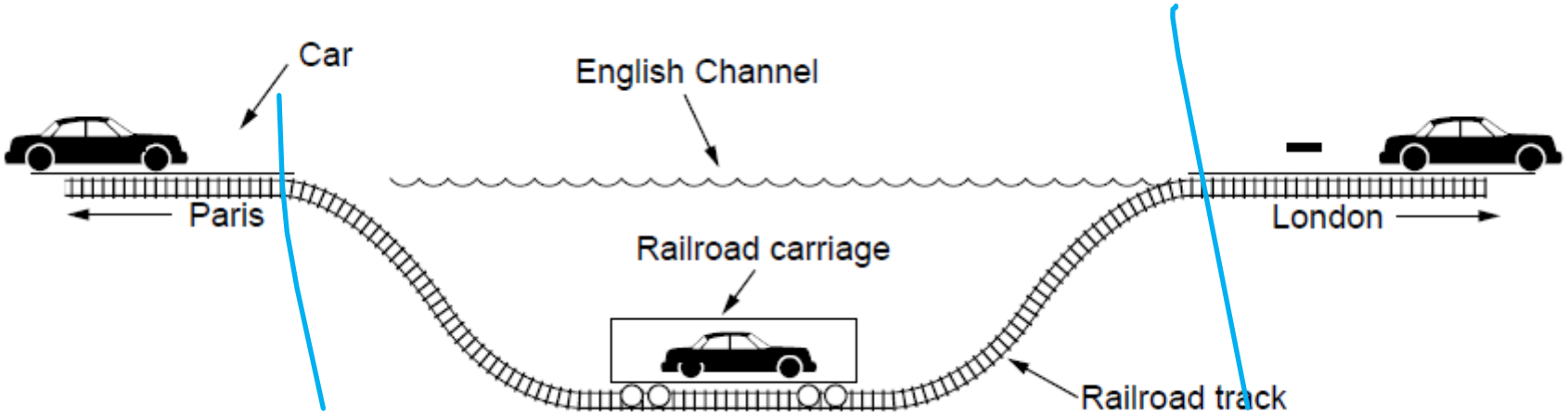
b) Network and link layer protocol processing.

# Tunneling (1)




Tunneling a packet from Paris to London.

# Tunneling (2)



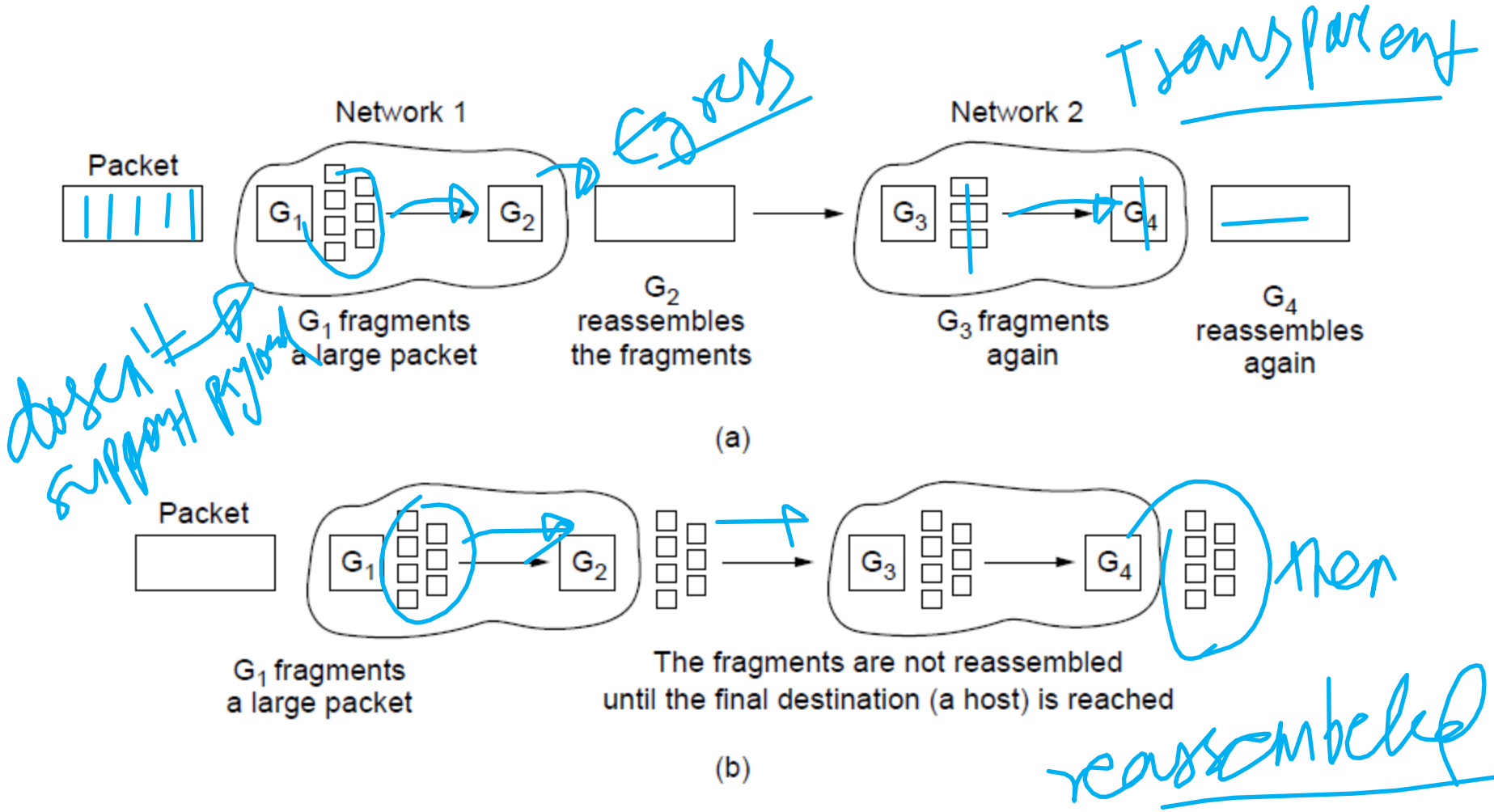

Tunneling a car from France to England

# Packet Fragmentation (1)

Packet size issues:


- Hardware
- Operating system
- Protocols
- Compliance with (inter)national standard.
- Reduce error-induced retransmissions
- Prevent packet occupying channel too long.

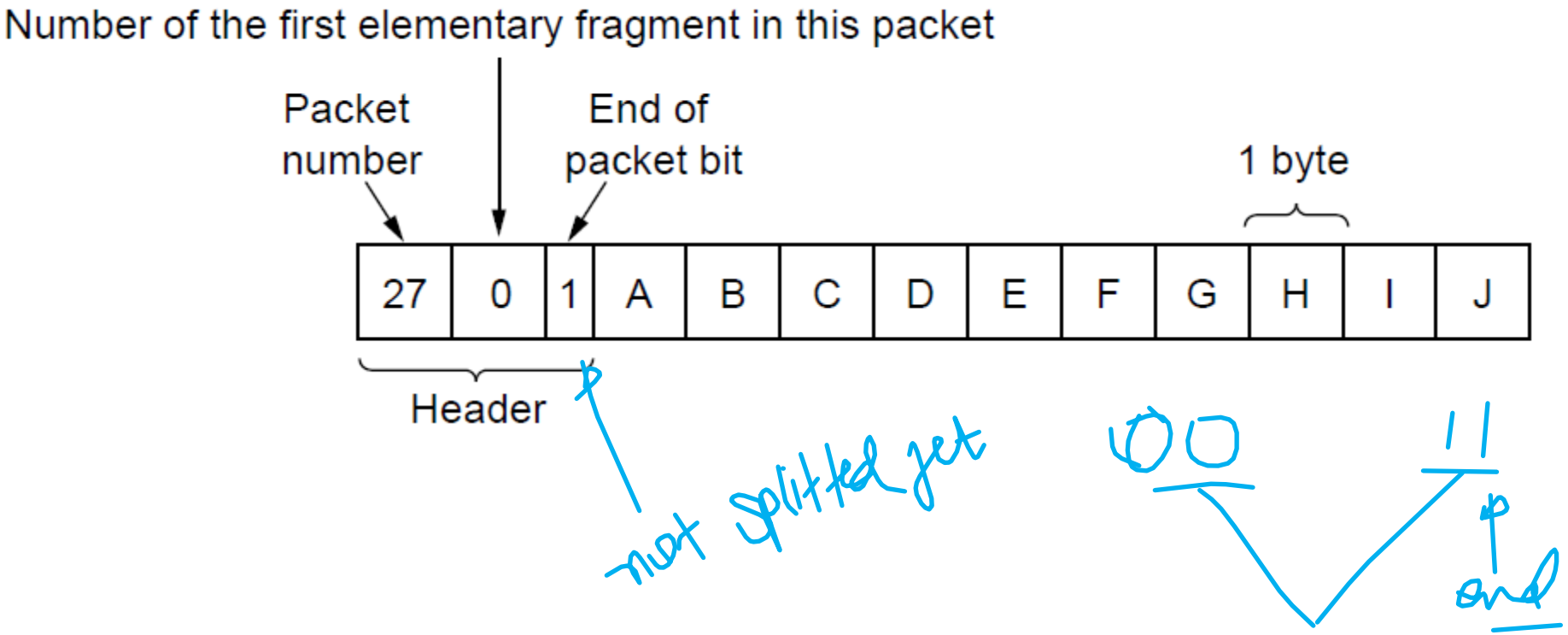
# Packet Fragmentation (2)




a) Transparent fragmentation.

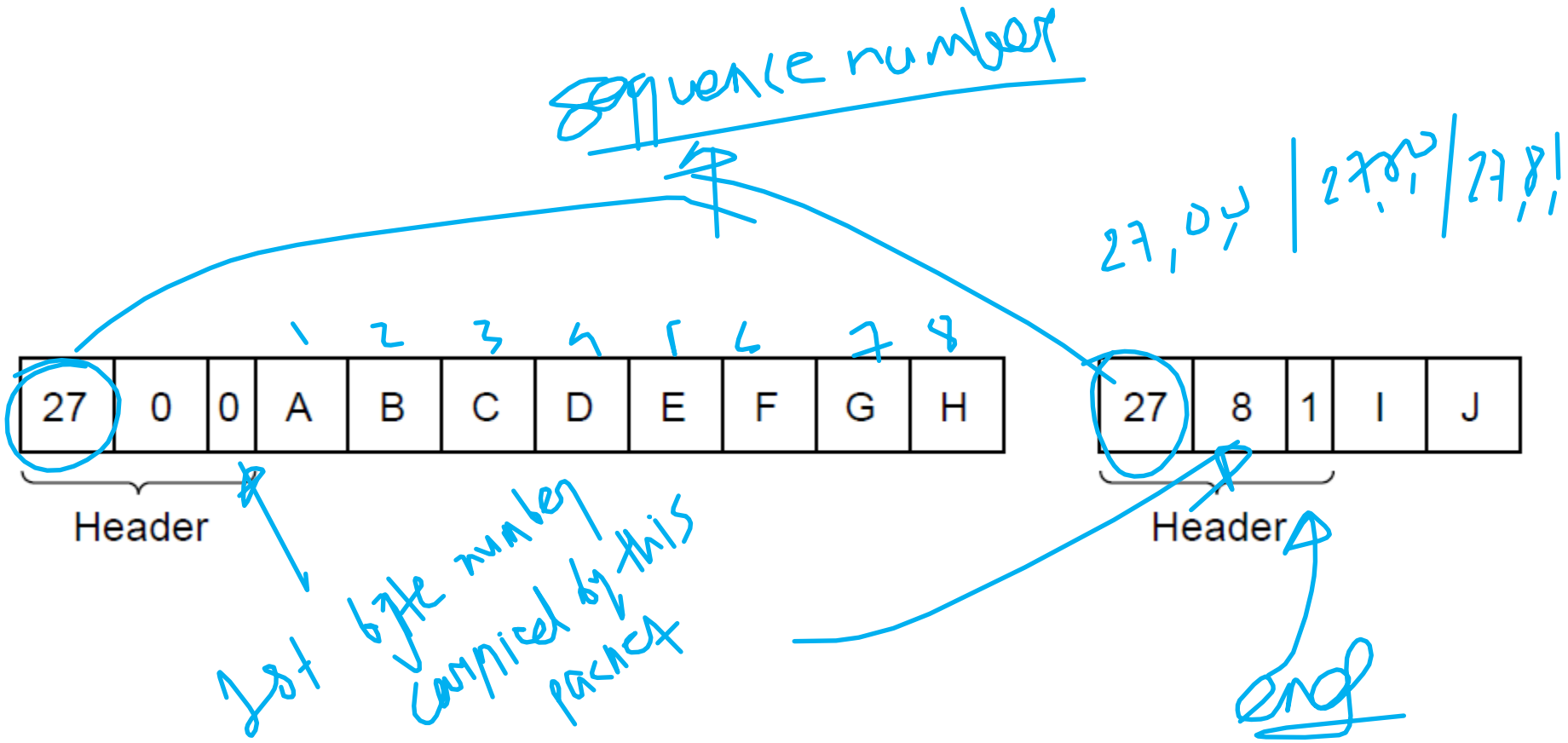
b) Nontransparent fragmentation

# Packet Fragmentation (3)




Fragmentation when the elementary data size is 1 byte.
(a) Original packet, containing 10 data bytes.

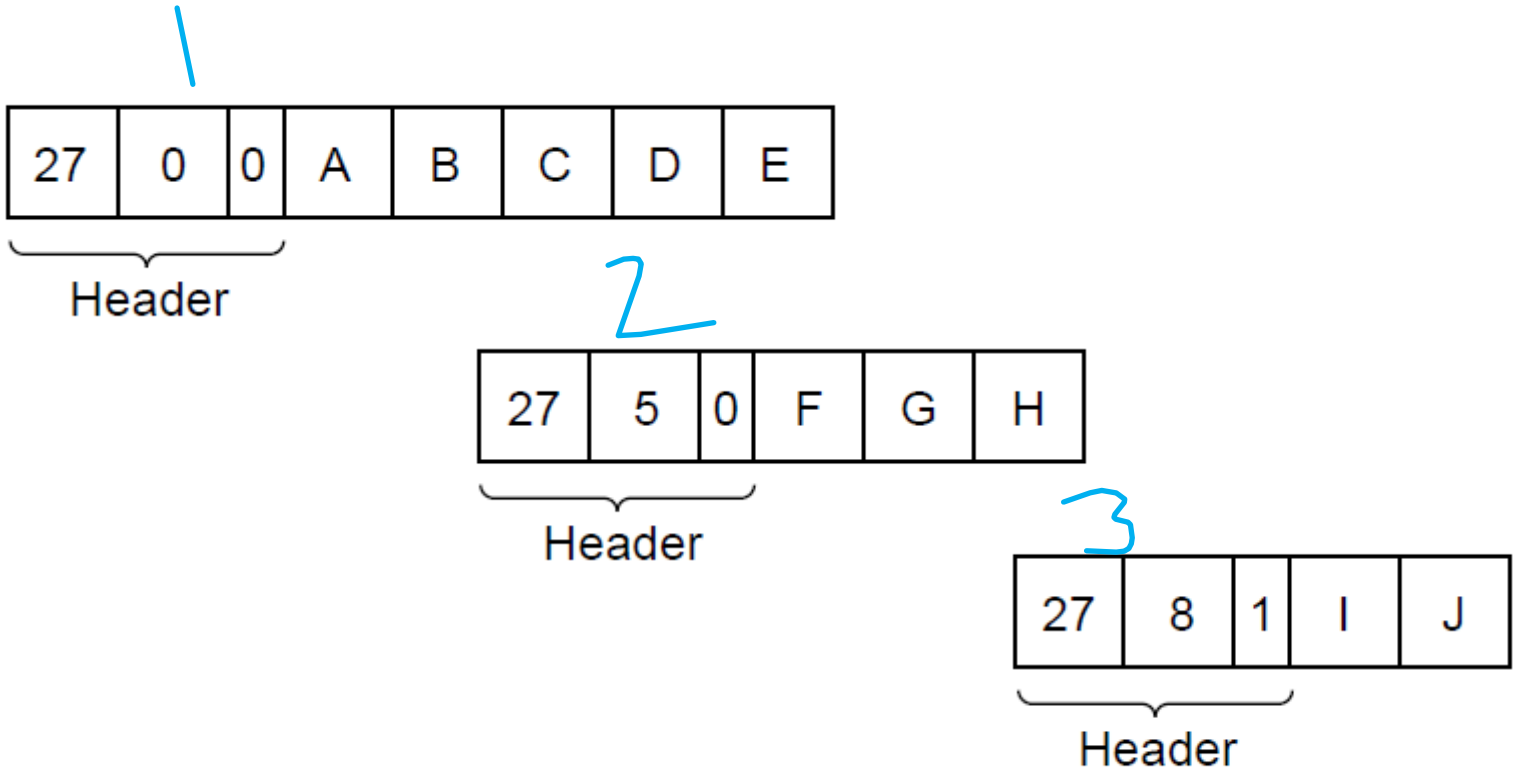
# Packet Fragmentation (4)




Fragmentation when the elementary data size is 1 byte

(b) Fragments after passing through a network with maximum packet size of 8 payload bytes plus header.

# Packet Fragmentation (5)

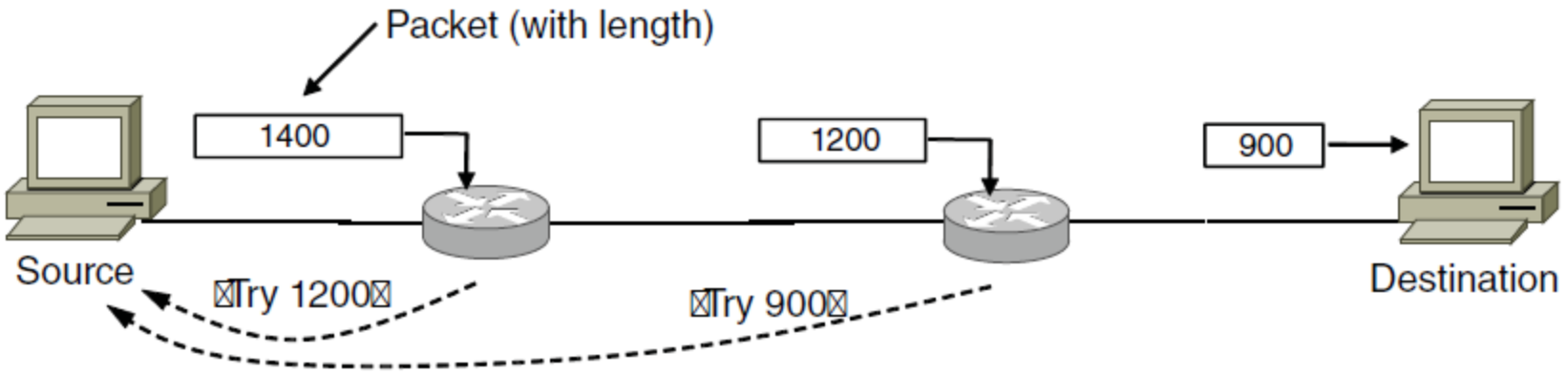



Fragmentation when the elementary data size is 1 byte

(c) Fragments after passing through a size 5 gateway.

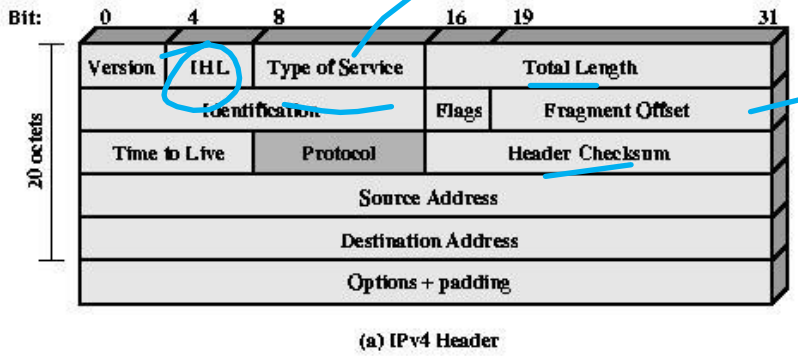
# Packet Fragmentation (6)

In IPv6 only Src router can fragment

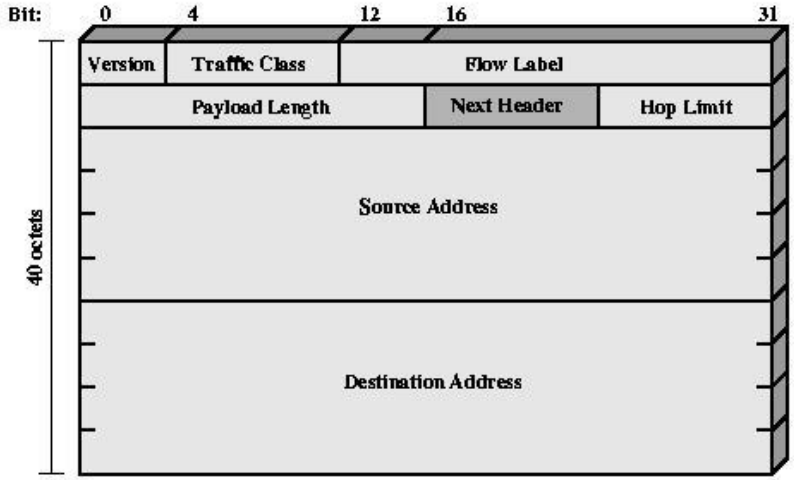



Path MTU Discovery

packet




(a) IPv4 Header




(b) IPv6 Header

**Figure 2.2 IP Headers**

# Fragmentation and Reassembly

a) Networks may have different maximum packet size

b) Router may need to fragment datagrams before sending to next network

c) Fragments may need further fragmenting in later networks

d) Reassembly done only at final destination since fragments may take different routes

-transparent not in IP

# Fragmentation and Reassembly

a) If Reassembly permitted at intermediate routers

- Large buffers needed to store partial datagram
- All fragment must pass thru same gateway

difficult

# Fragmentation Example

*MTUs for some networks*

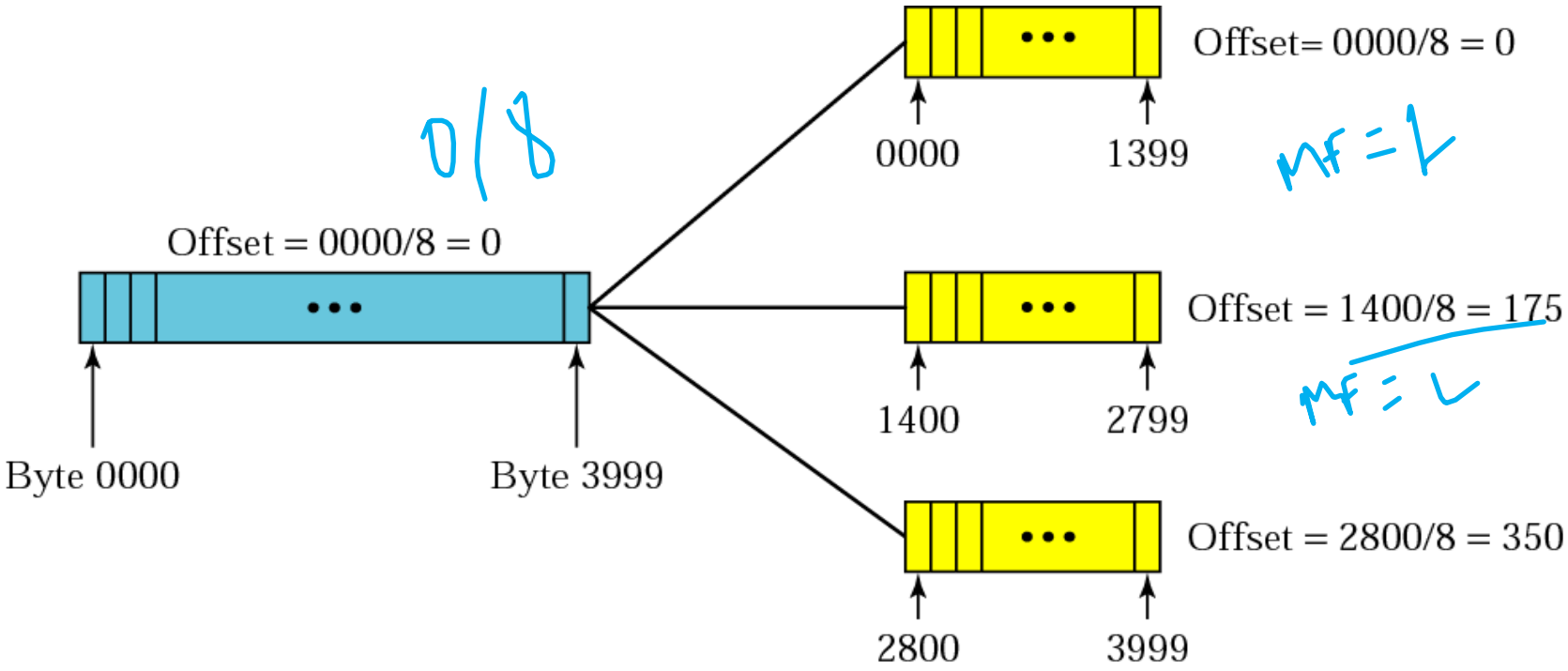
| *Protocol* | *MTU* |
|---|---|
| Hyperchannel | 65,535 |
| Token Ring (16 Mbps) | 17,914 |
| Token Ring (4 Mbps) | 4,464 |
| FDDI | 4,352 |
| Ethernet | 1,500 |
| X.25 | 576 |
| PPP | 296 |

# Fragmentation Example

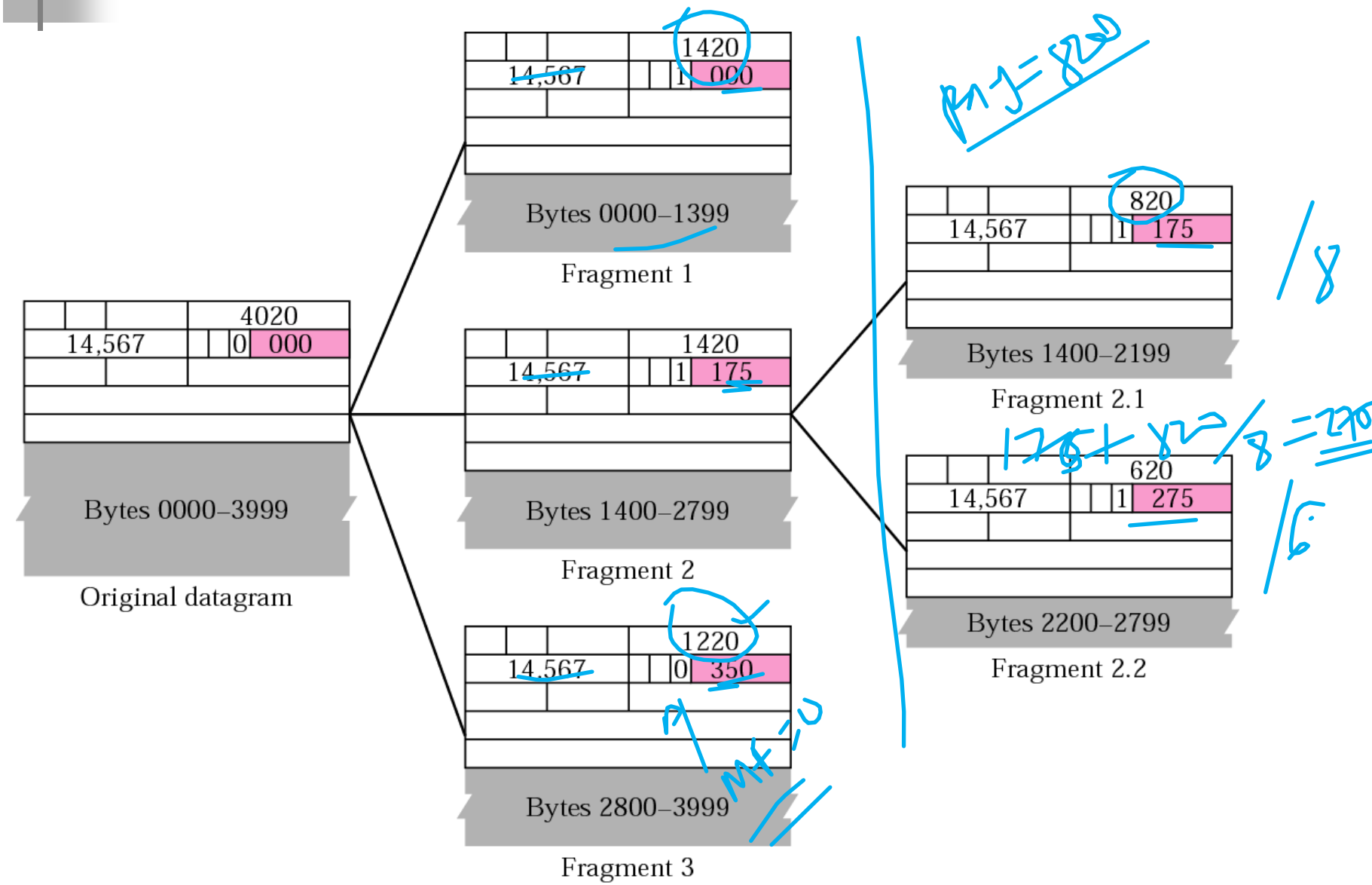
*Flags field*

D: Do not fragment
M: More fragments

# Fragmentation Example



# Fragmentation Example



# The Network Layer Principles (1)

- Make sure it works
- Keep it simple
- Make clear choices
- Exploit modularity
- Expect heterogeneity

. . .

# The Network Layer Principles (2)

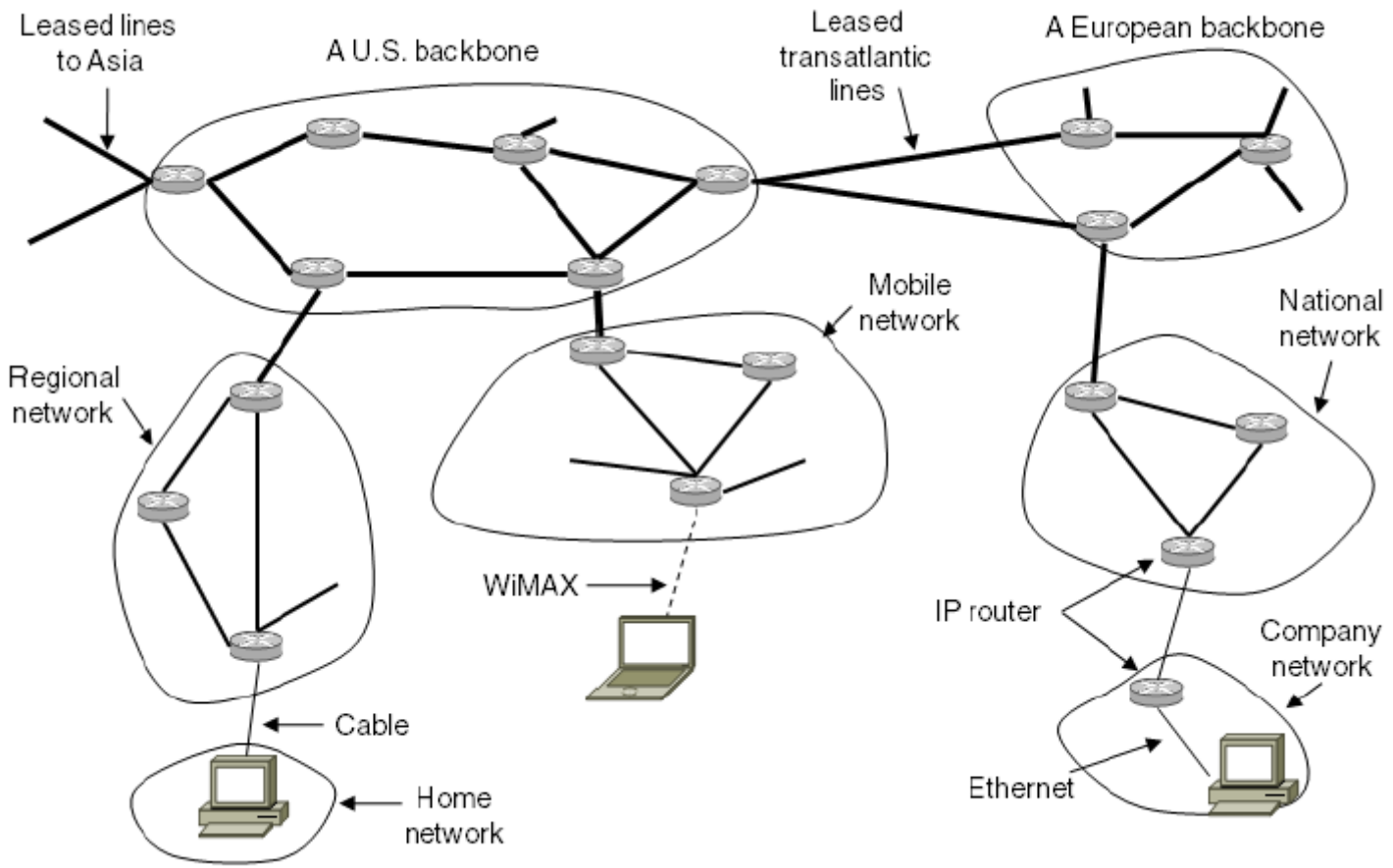
. . .

- Avoid static options and parameters
- Look for good design (not perfect)
- Strict sending, tolerant receiving
- Think about scalability
- Consider performance and cost

# The Network Layer in the Internet (1)

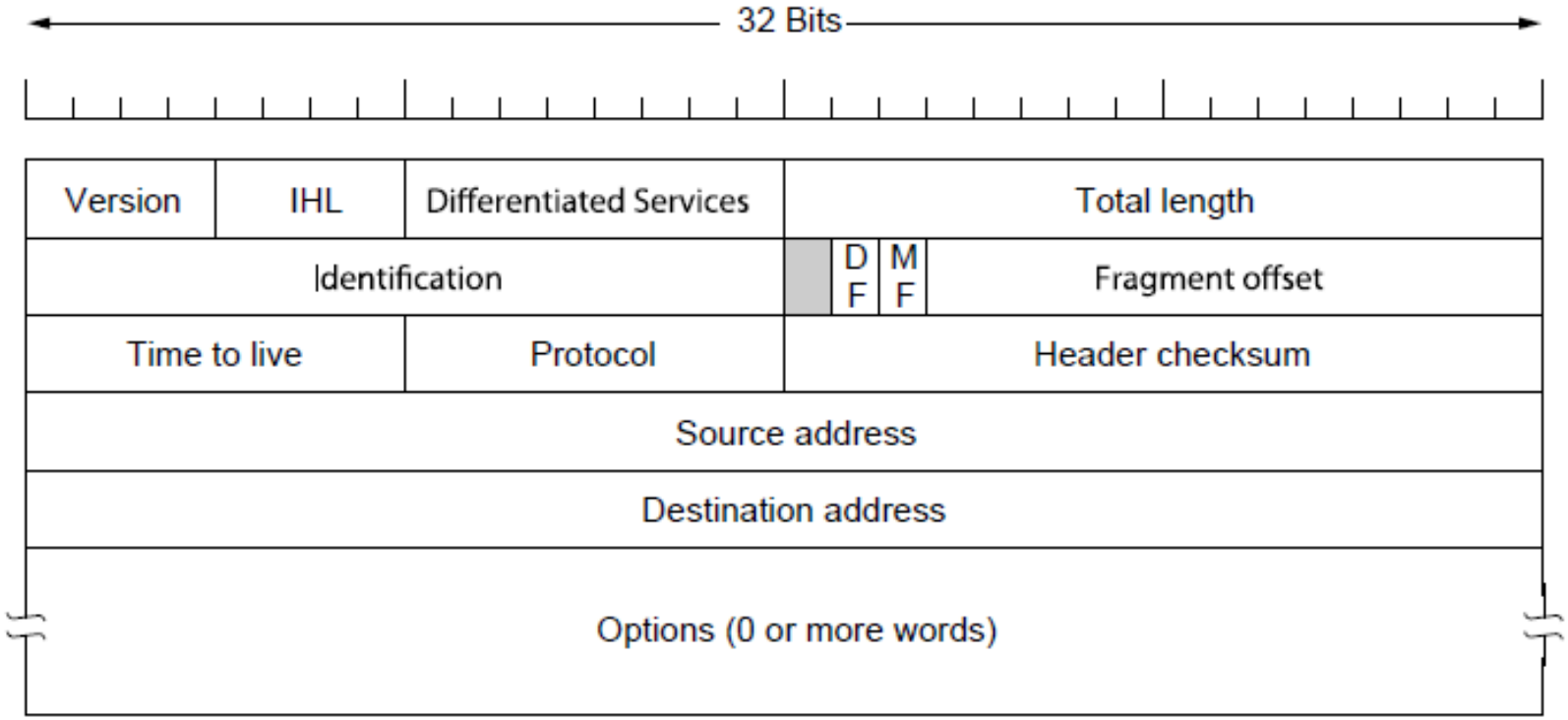
- The IP Version 4 Protocol
- IP Addresses
- IP Version 6
- Internet Control Protocols
- Label Switching and MPLS
- OSPF—An Interior Gateway Routing Protocol
- BGP—The Exterior Gateway Routing Protocol
- Internet Multicasting
- Mobile IP

# The Network Layer in the Internet (2)




The Internet is an interconnected collection of many networks.

# The IP Version 4 Protocol (1)



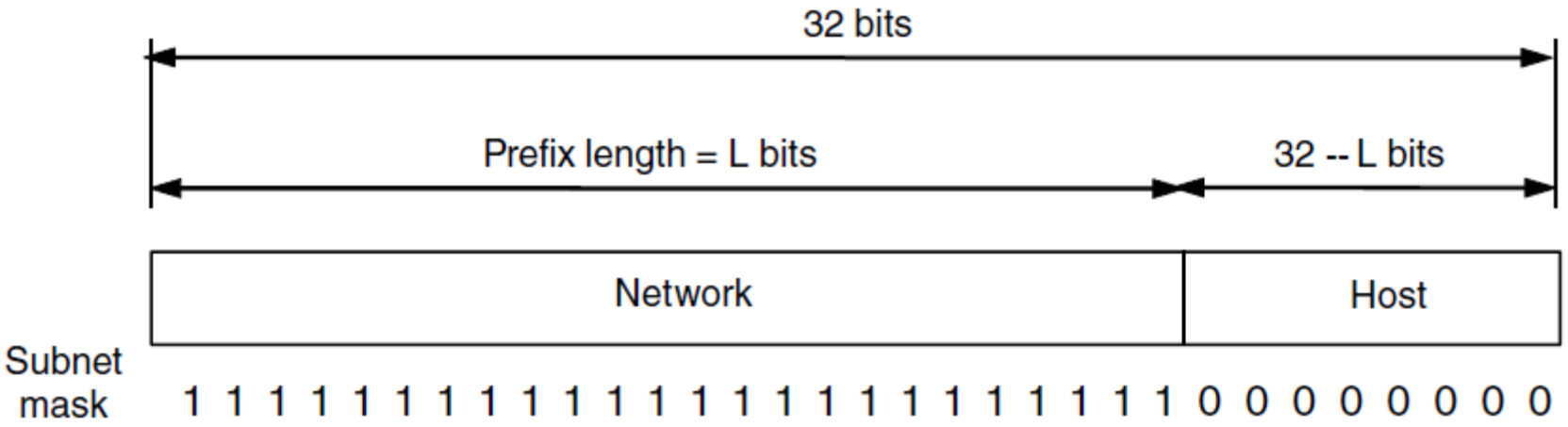

The IPv4 (Internet Protocol) header.

# The IP Version 4 Protocol (2)

| Option | Description |
|---|---|
| Security | Specifies how secret the datagram is |
| Strict source routing | Gives the complete path to be followed |
| Loose source routing | Gives a list of routers not to be missed |
| Record route | Makes each router append its IP address |
| Timestamp | Makes each router append its address and timestamp |

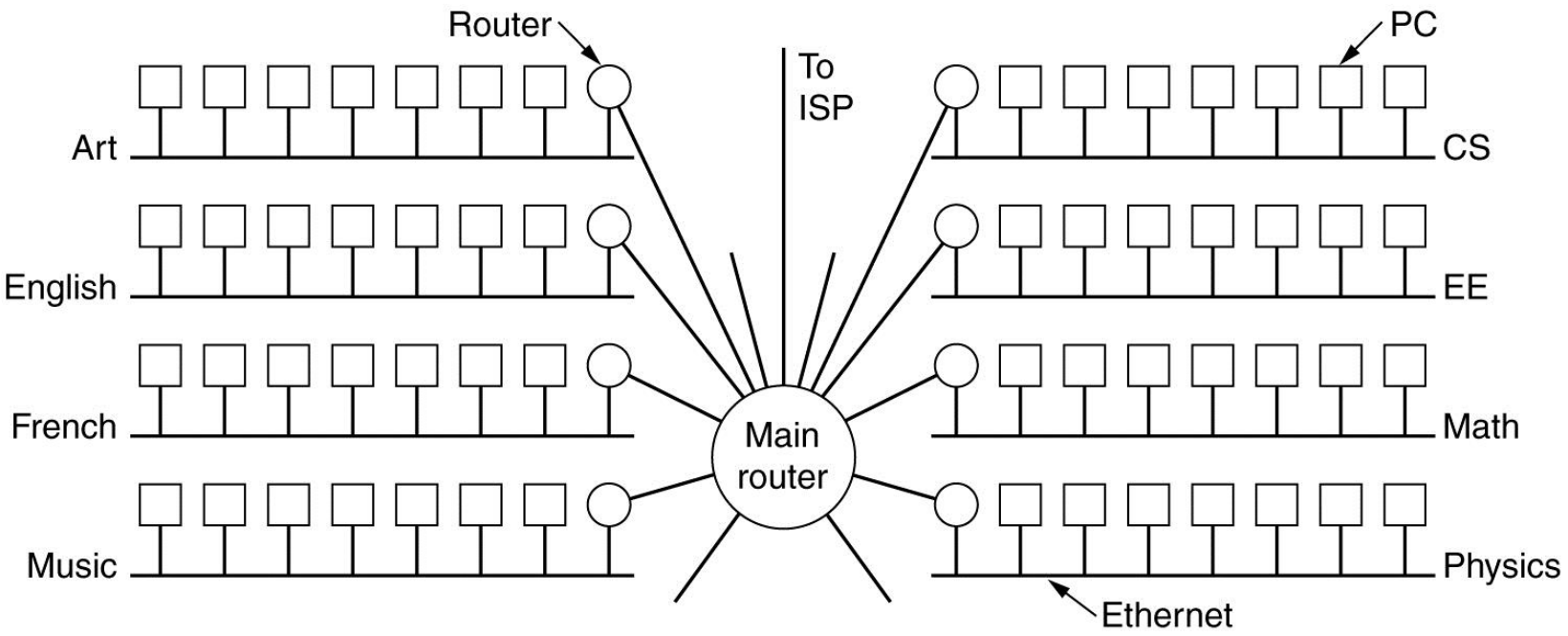
Some of the IP options.

# IP Addresses (1)



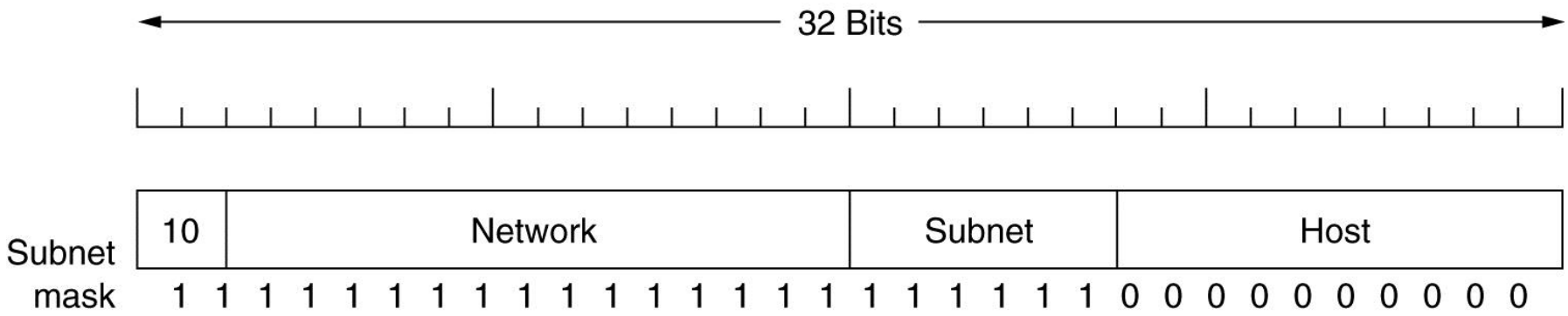

An IP prefix.

# Subnets



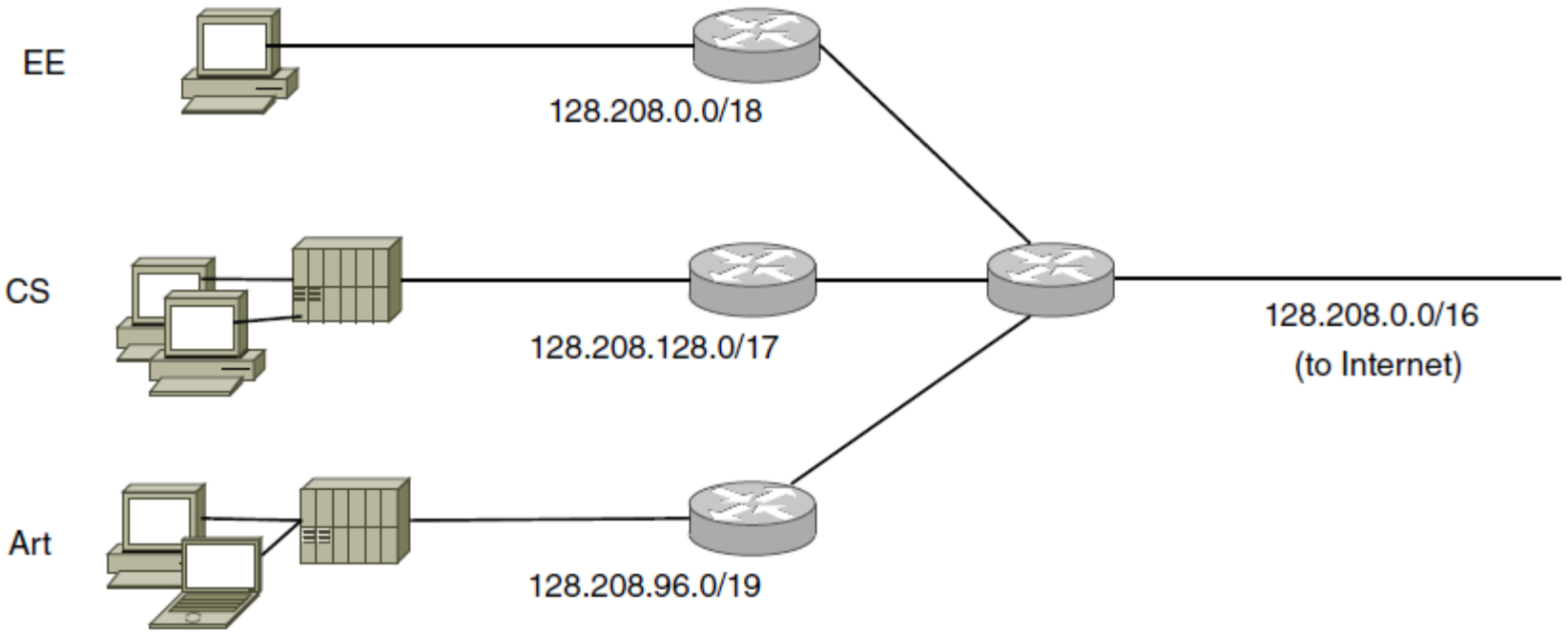

A campus network consisting of LANs for various departments.

# Subnets (2)




A class B network subnetted into 64 subnets.

# IP Addresses (2)



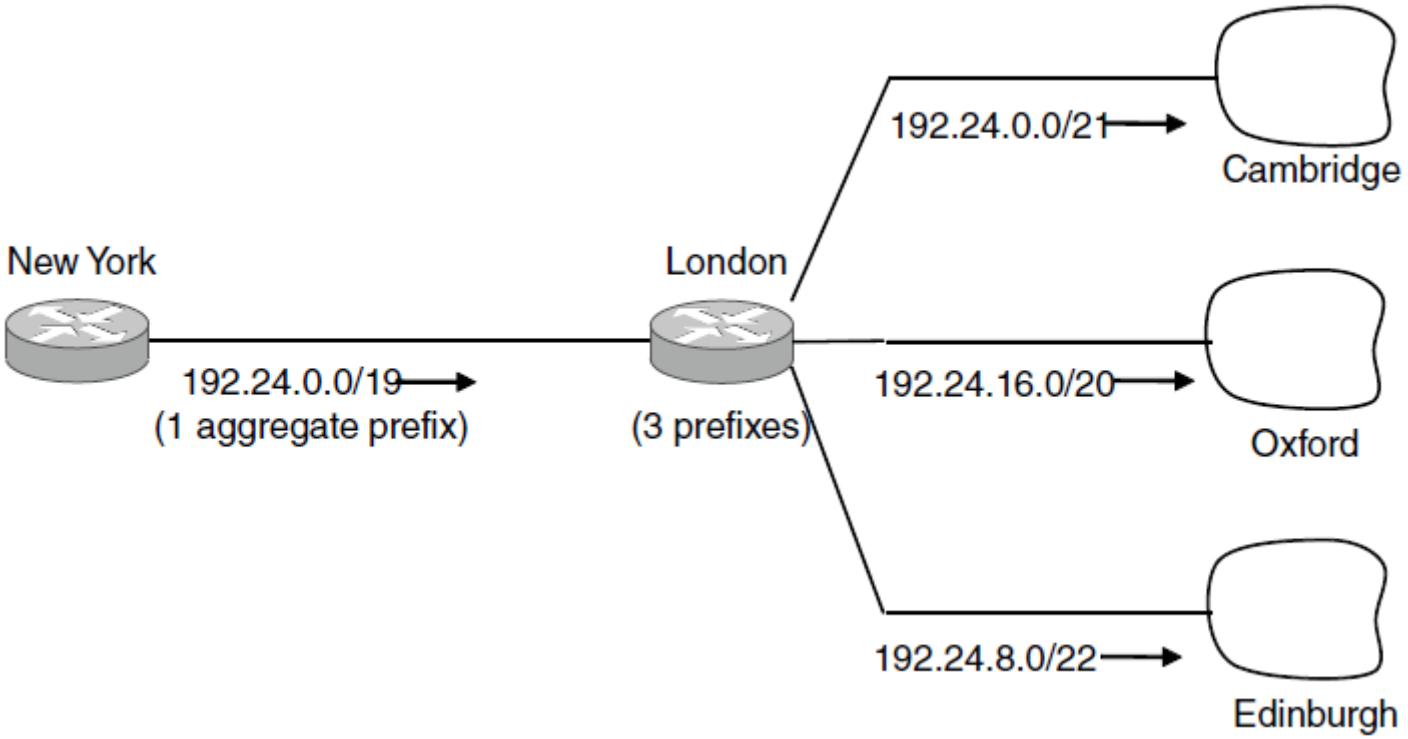

Splitting an IP prefix into separate networks with subnetting.

# IP Addresses (3)

| University | First address | Last address | How many | Prefix |
|---|---|---|---|---|
| Cambridge | 194.24.0.0 | 194.24.7.255 | 2048 | 194.24.0.0/21 |
| Edinburgh | 194.24.8.0 | 194.24.11.255 | 1024 | 194.24.8.0/22 |
| (Available) | 194.24.12.0 | 194.24.15.255 | 1024 | 194.24.12/22 |
| Oxford | 194.24.16.0 | 194.24.31.255 | 4096 | 194.24.16.0/20 |

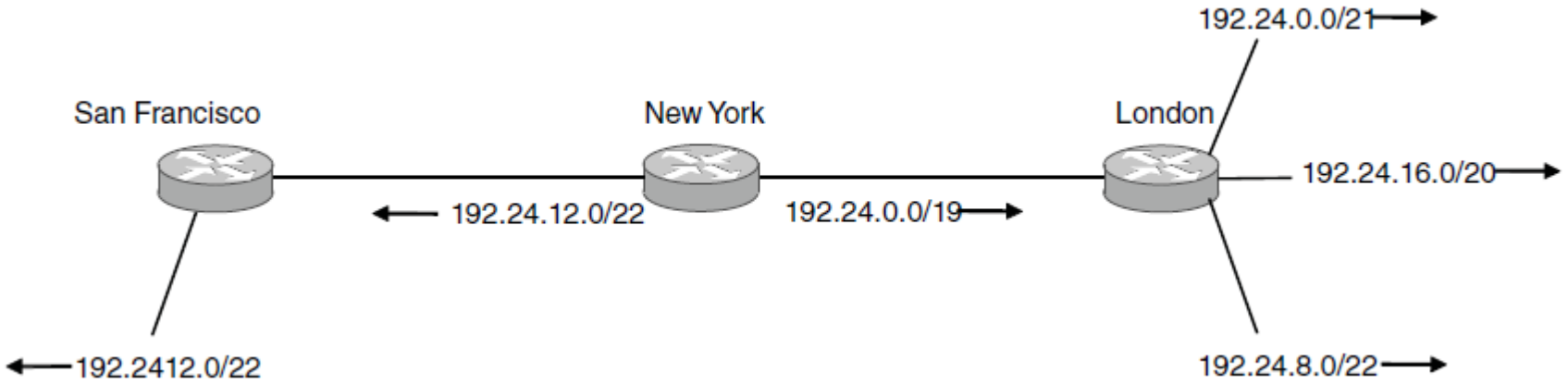
A set of IP address assignments

# IP Addresses (4)



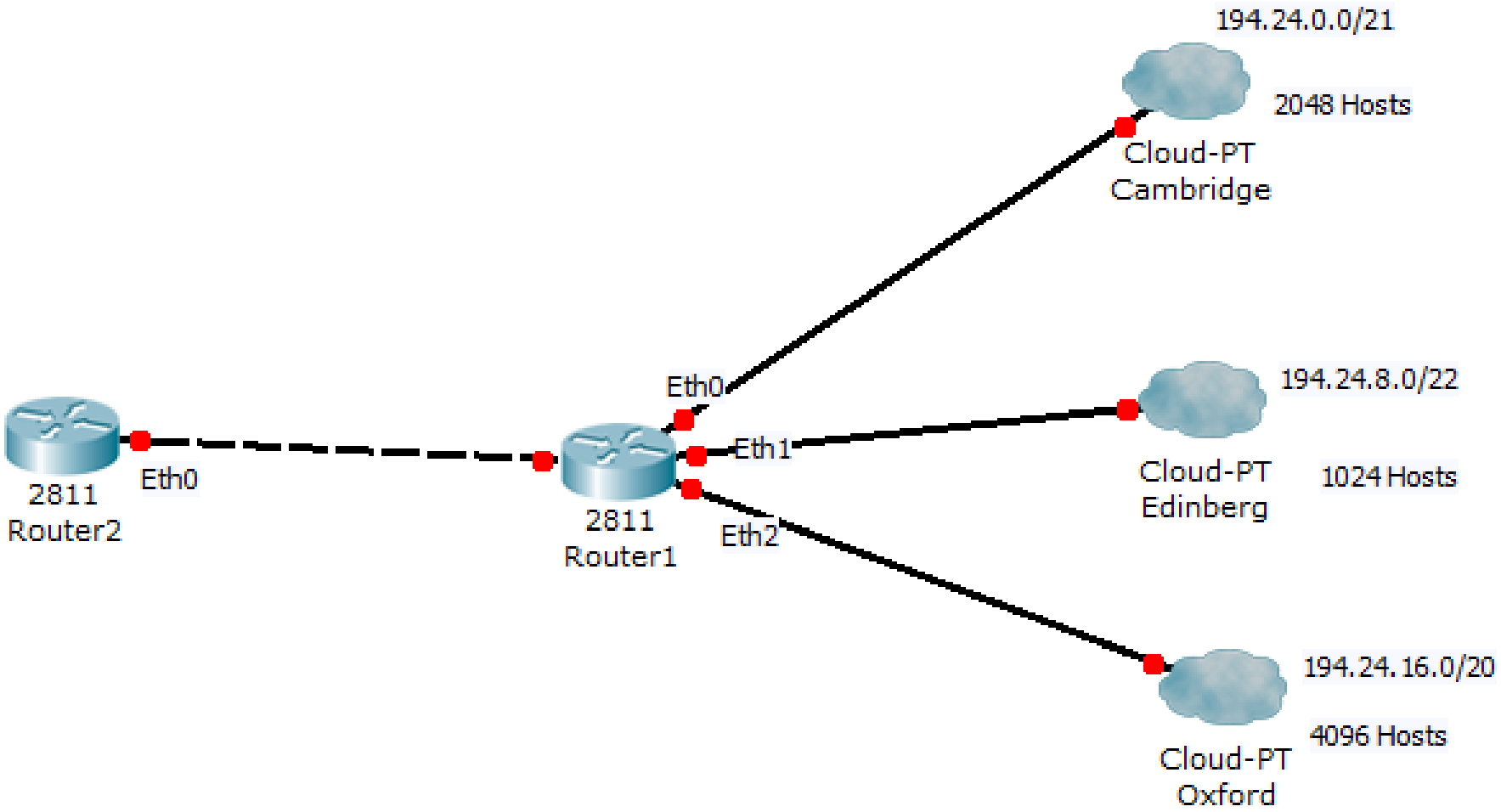

Aggregation of IP prefixes

# IP Addresses (5)




Longest matching prefix routing at the New York router.

# Network



# CDR – Classless InterDomain Routing

Routing Table at Router 1

| | Address | Mask | Interface |
|---|---|---|---|
| Cambridge | 194.24.0.0 | 21 | Eth0 |
| Edinberg | 194.24.8.0 | 22 | Eth1 |
| Oxford | 194.24.16.0 | 20 | Eth2 |

Consider a packet with 194.24.17.4. Compare to all the entries and use longest matching entry

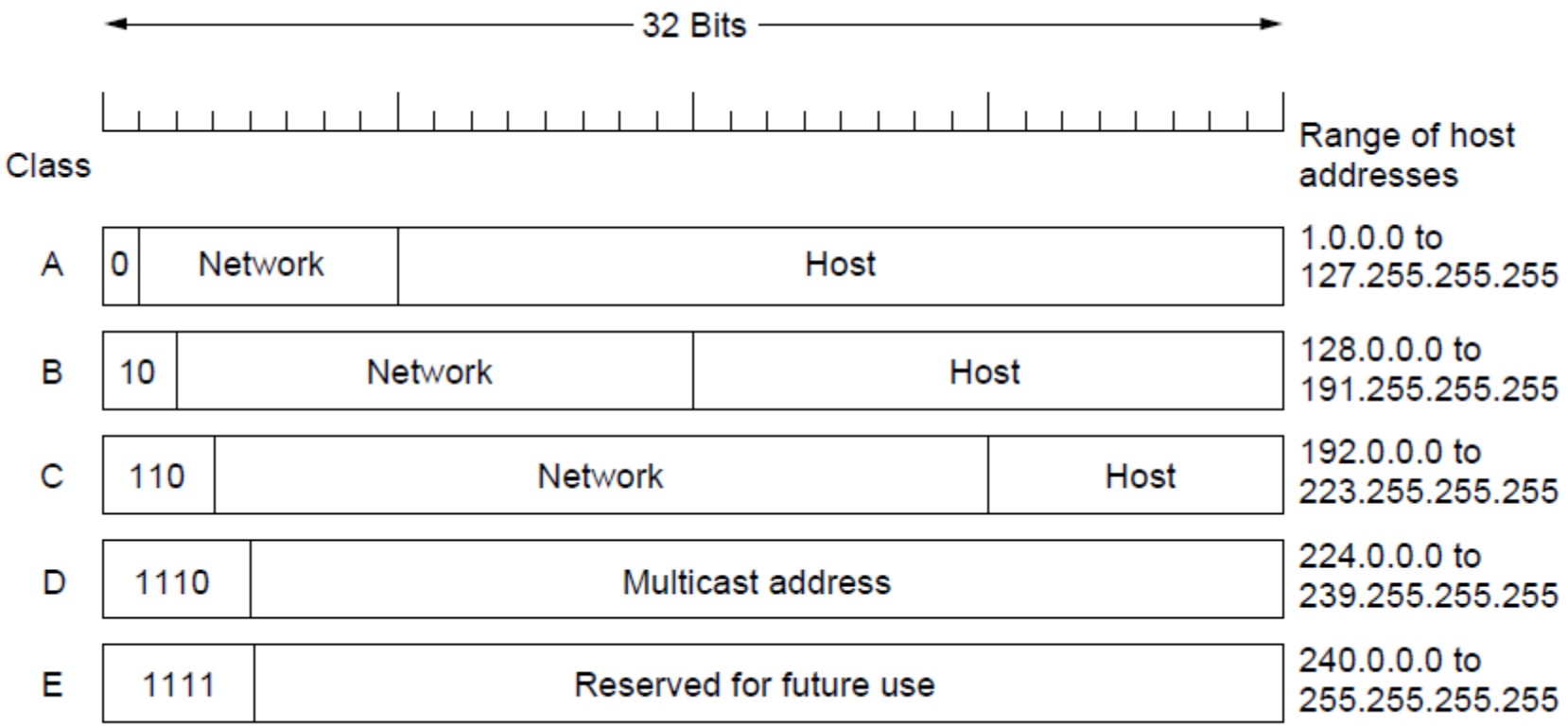
Routing Table at Router 2

| | Address | Mask | Interface |
|---|---|---|---|
| Cambridge | 194.24.0.0 | 21 | Eth0 |
| Edinberg | 194.24.8.0 | 22 | Eth0 |
| Oxford | 194.24.16.0 | 20 | Eth0 |

Aggregated Routing Table at Router 2

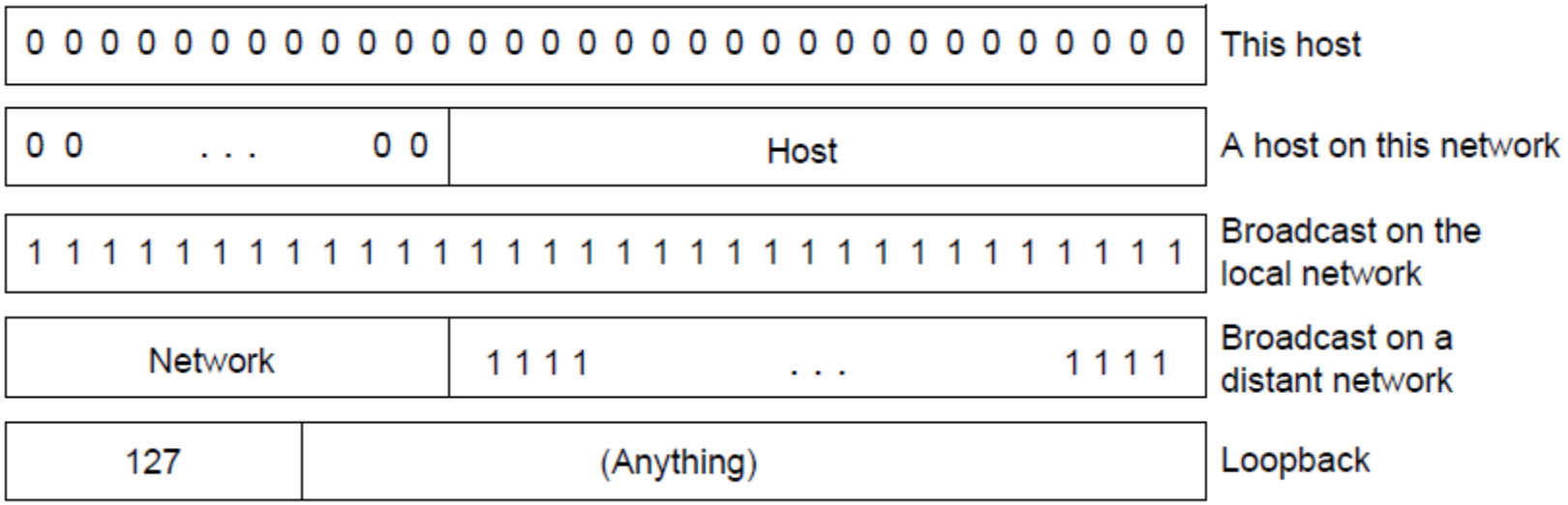
| | Address | Mask | Interface |
|---|---|---|---|
| All | 194.24.0.0 | 19 | Eth0 |

# IP Addresses (6)




IP address formats

# IP Addresses (7)




Special IP addresses

# The IPv4 Shortage

a) Problem: consumer ISPs typically only give one IP address per-household
   - Additional IPs cost extra
   - More IPs may not be available

b) Today's households have more networked devices than ever
   - Laptops and desktops
   - TV, bluray players, game consoles
   - Tablets, smartphones, eReaders

c) How to get all these devices online?

# Private IP Networks

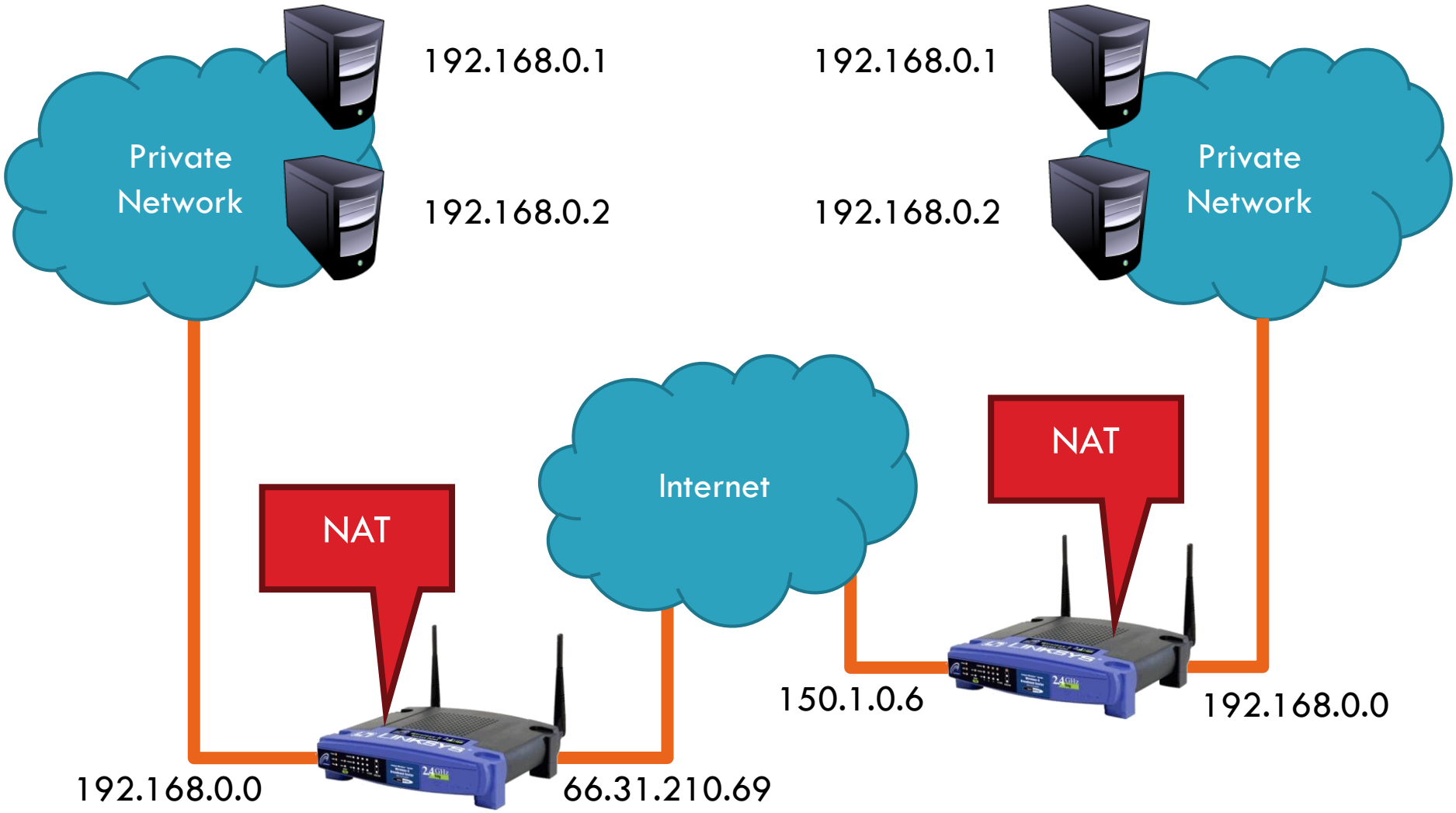
a) Idea: create a range of private IPs that are separate from the rest of the network
   - Use the private IPs for internal routing
   - Use a special router to bridge the LAN and the WAN

b) Properties of private IPs
   - Not globally unique
   - Usually taken from non-routable IP ranges (why?)

c) Typical private IP ranges
   - 10.0.0.0 – 10.255.255.255
   - 172.16.0.0 – 172.31.255.255
   - 192.168.0.0 – 192.168.255.255

# Private Networks





# Network Address Translation (NAT)

a) NAT allows hosts on a private network to communicate with the Internet
   - Warning: connectivity is not seamless

b) Special router at the boundary of a private network
   - Replaces internal IPs with external IP
     - This is "Network Address Translation"
   - May also replace TCP/UDP port numbers

c) Maintains a table of active flows
   - Outgoing packets initialize a table entry
   - Incoming packets are rewritten based on the table

# Basic NAT Operation



**Private Network**

**Internet**

Source: 192.168.0.1
Dest: 74.125.228.67

Source: 66.31.210.69
Dest: 74.125.228.67

| Private Address | Public Address |
| --- | --- |
| 192.168.0.1:2345 | 66.31.210.69:80 |

192.168.0.1

66.31.210.69

74.125.228.67

Source: 74.125.228.67
Dest: 192.168.0.1

Source: 74.125.228.67
Dest: 66.31.210.69

# Advantages of NATs

a) Allow multiple hosts to share a single public IP

b) Allow migration between ISPs
   - Even if the public IP address changes, you don't need to reconfigure the machines on the LAN

c) Load balancing
   - Forward traffic from a single public IP to multiple private hosts

# Natural Firewall



Private Network

Internet

| Private Address | Public Address |
| --- | --- |
| | |

192.168.0.1

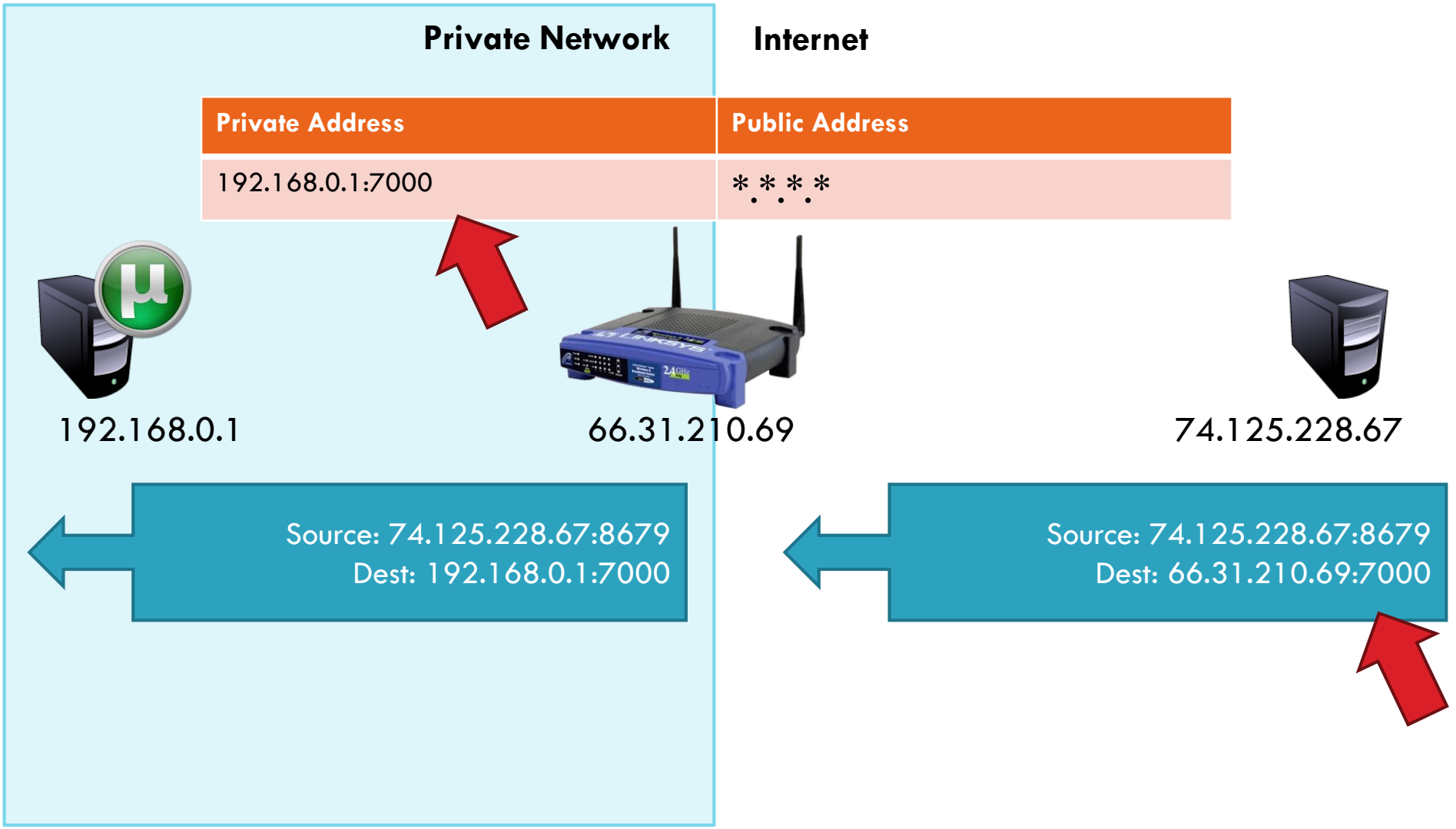
66.31.210.69

74.125.228.67

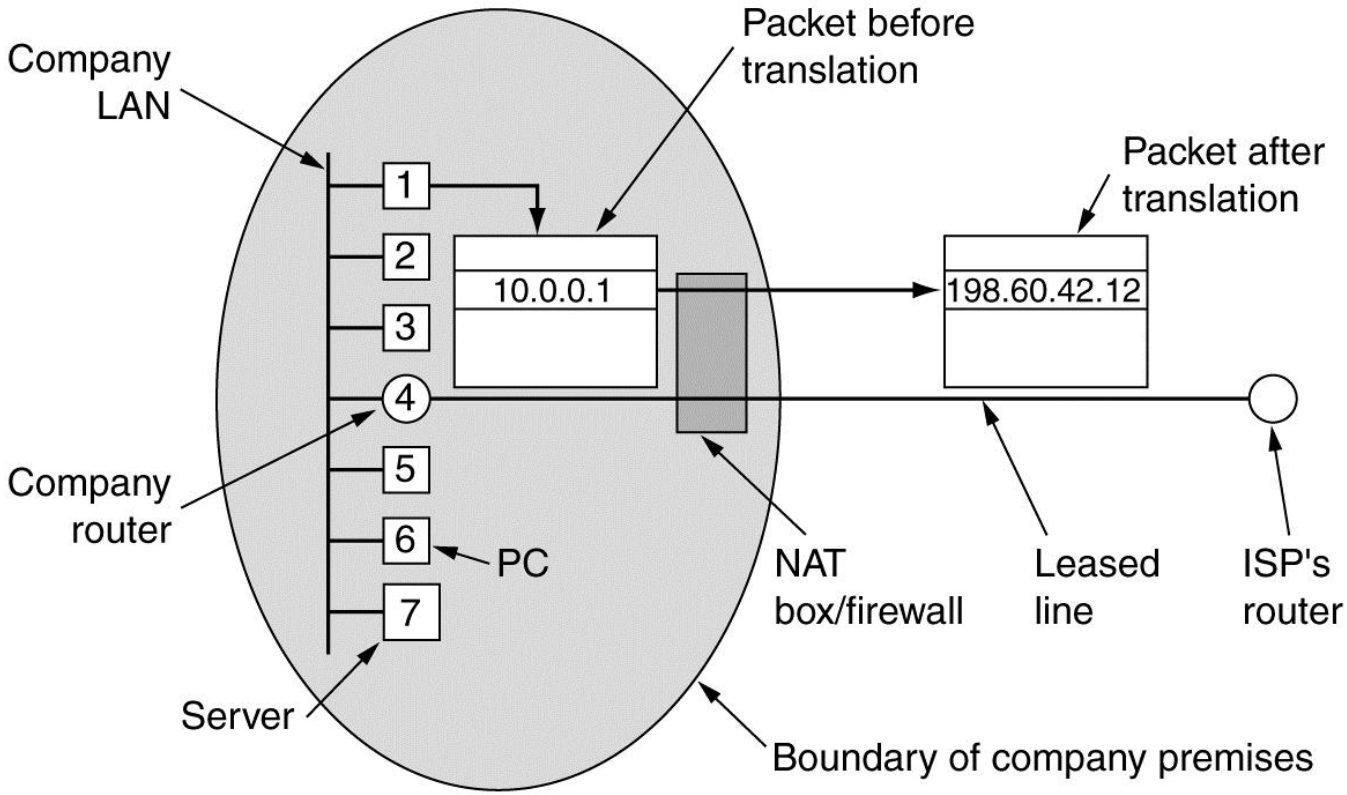
Source: 74.125.228.67
Dest: 66.31.210.69

# Port Forwarding





# NAT – Network Address Translation

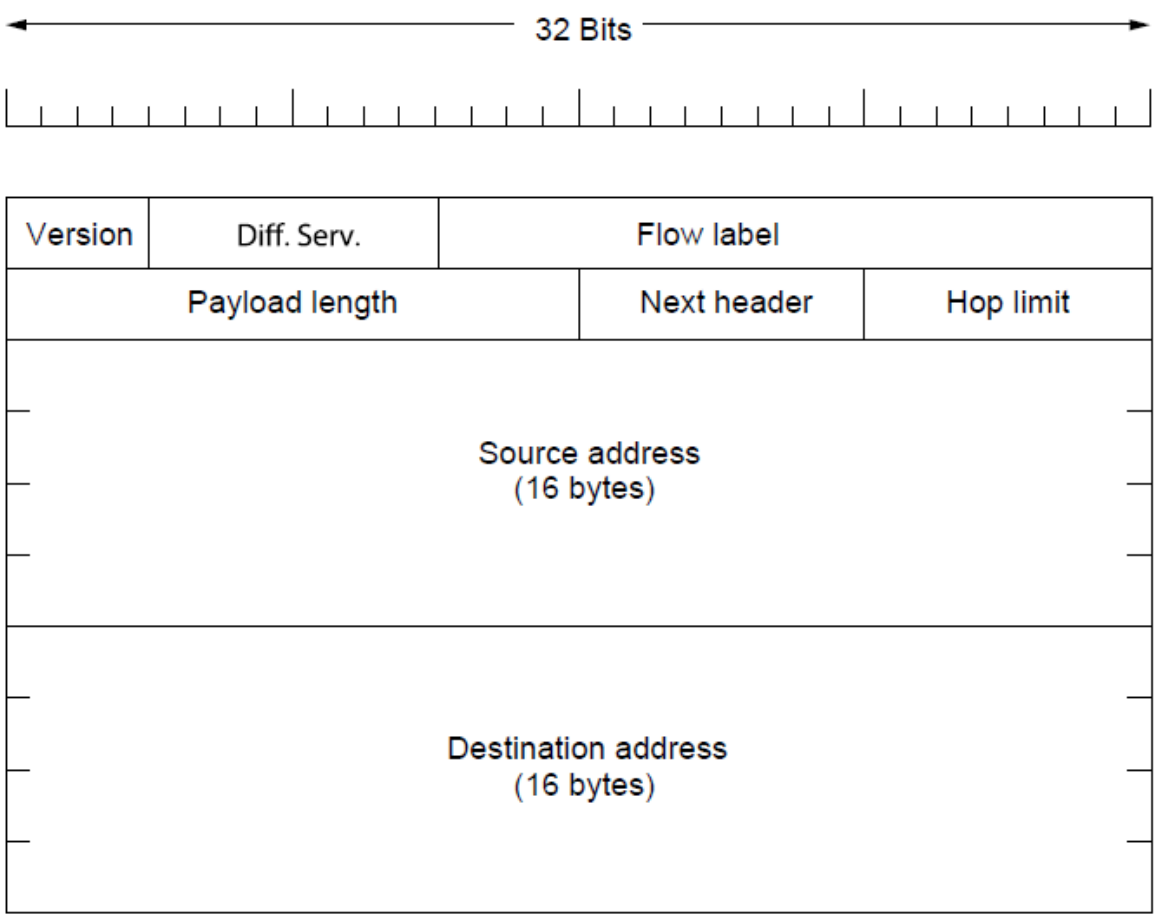



Placement and operation of a NAT box.

# IP Version 6 Goals

- Support billions of hosts
- Reduce routing table size
- Simplify protocol
- Better security
- Attention to type of service
- Aid multicasting
- Roaming host without changing address
- Allow future protocol evolution
- Permit coexistence of old, new protocols. . .

# IP Version 6 (1)



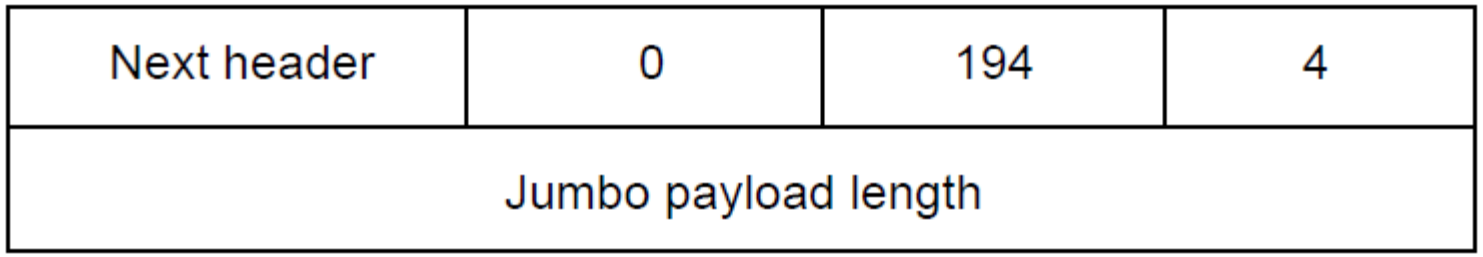

The IPv6 fixed header (required).

# IP Version 6 (2)

| Extension header | Description |
|---|---|
| Hop-by-hop options | Miscellaneous information for routers |
| Destination options | Additional information for the destination |
| Routing | Loose list of routers to visit |
| Fragmentation | Management of datagram fragments |
| Authentication | Verification of the sender's identity |
| Encrypted security payload | Information about the encrypted contents |

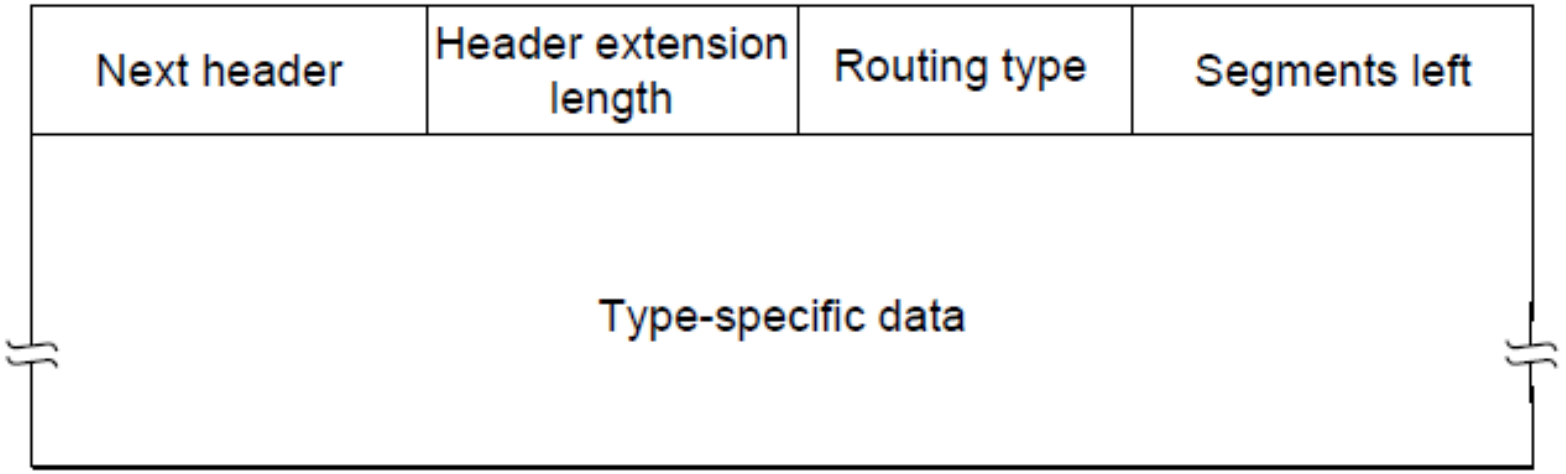
IPv6 extension headers

# IP Version 6 (3)




The hop-by-hop extension header for large datagrams (jumbograms).

# IP Version 6 (4)



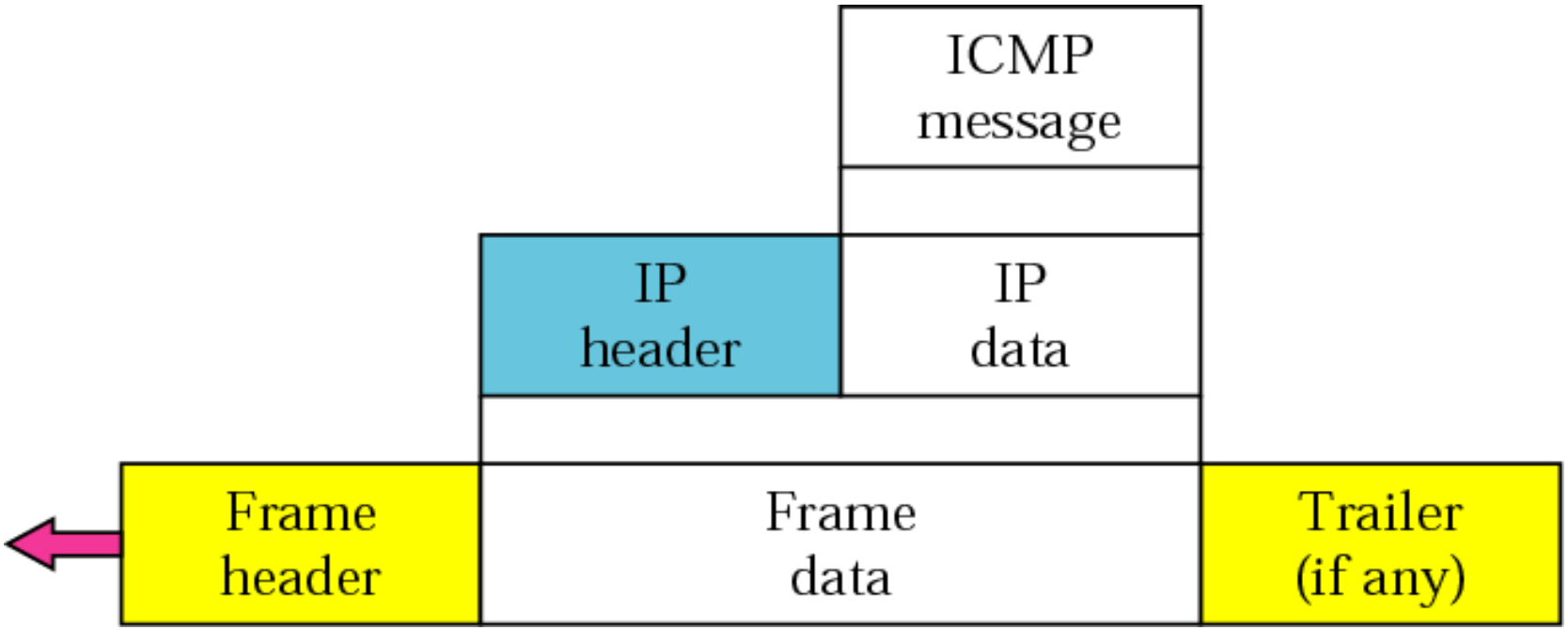

The extension header for routing.

# Internet Control Protocols (1)

| Message type | Description |
|---|---|
| Destination unreachable | Packet could not be delivered |
| Time exceeded | Time to live field hit 0 |
| Parameter problem | Invalid header field |
| Source quench | Choke packet |
| Redirect | Teach a router about geography |
| Echo and Echo reply | Check if a machine is alive |
| Timestamp request/reply | Same as Echo, but with timestamp |
| Router advertisement/solicitation | Find a nearby router |

The principal ICMP message types.

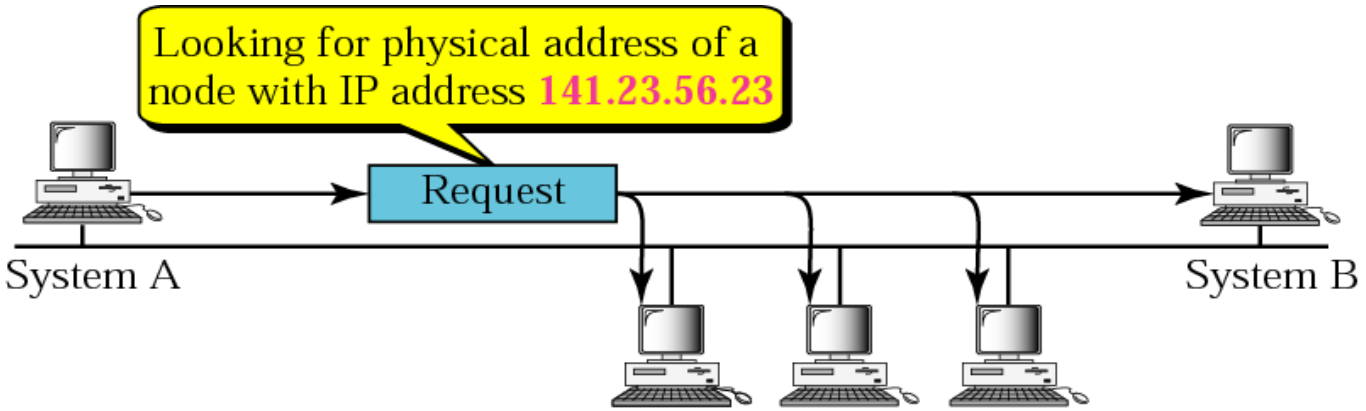
*ICMP encapsulation*



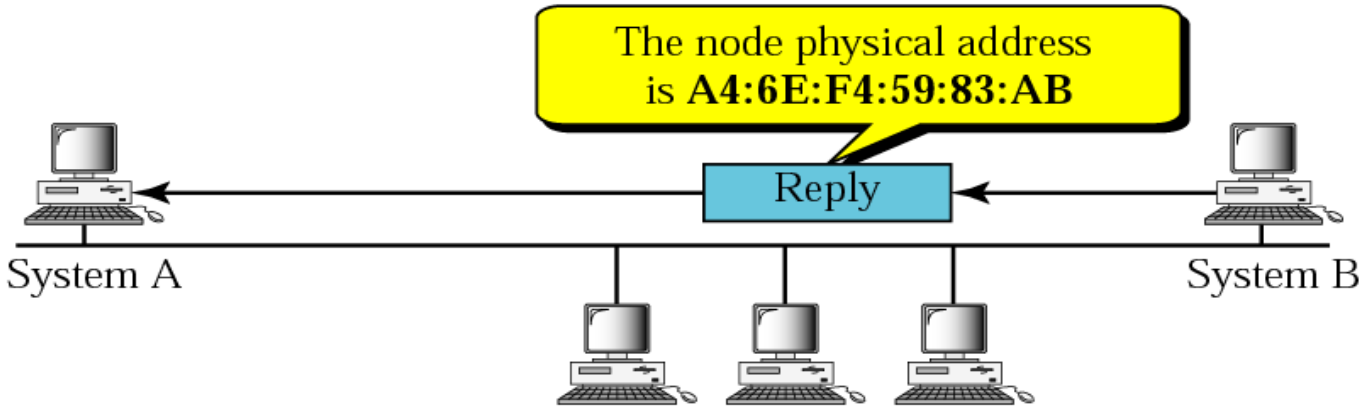
## *ICMP messages*

| *Category* | *Type* | *Message* |
|---|---|---|
| Error-reporting messages | 3 | Destination unreachable |
| | 4 | Source quench |
| | 11 | Time exceeded |
| | 12 | Parameter problem |
| | 5 | Redirection |
| Query messages | 8 or 0 | Echo request or reply |
| | 13 or 14 | Timestamp request or reply |
| | 17 or 18 | Address mask request or reply |
| | 10 or 9 | Router solicitation or advertisement |

# ARP– The Address Resolution Protocol

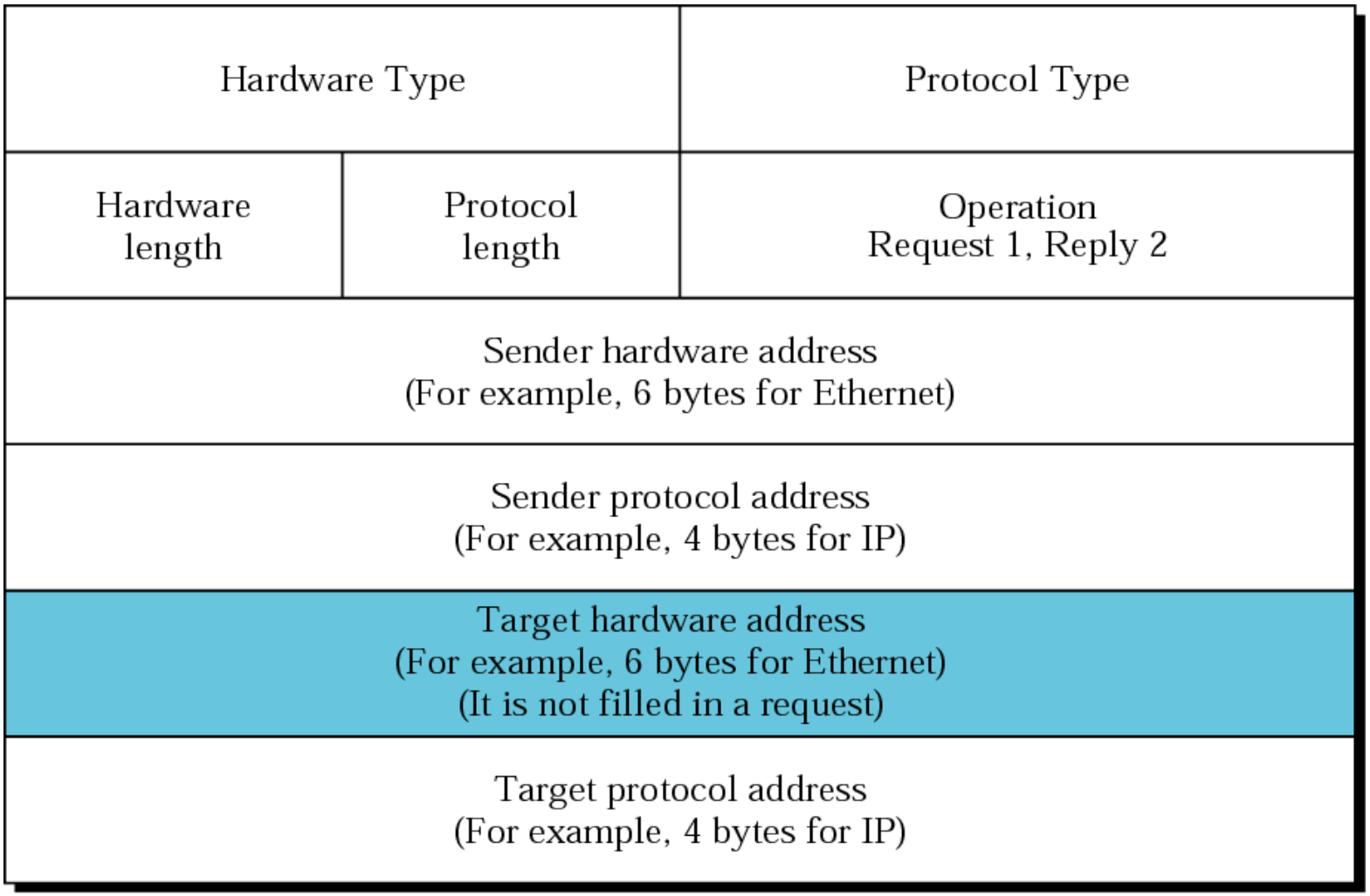



a. ARP request is broadcast

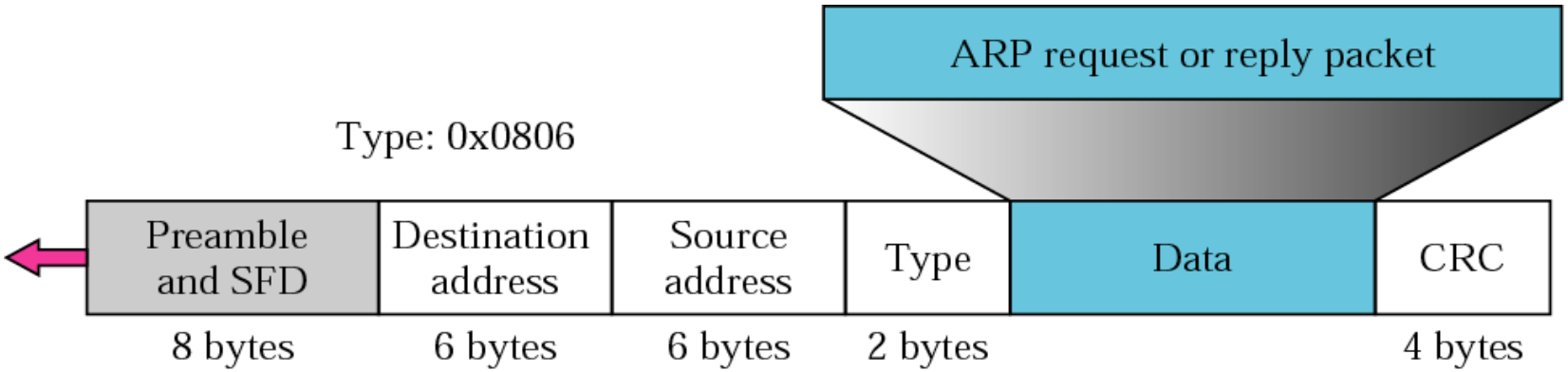



b. ARP reply is unicast

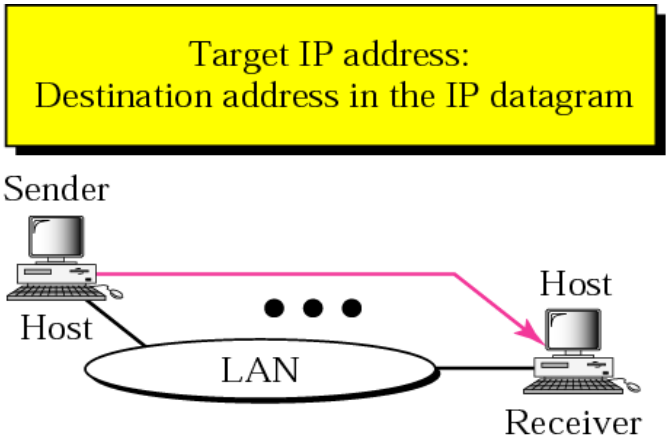
*ARP packet*



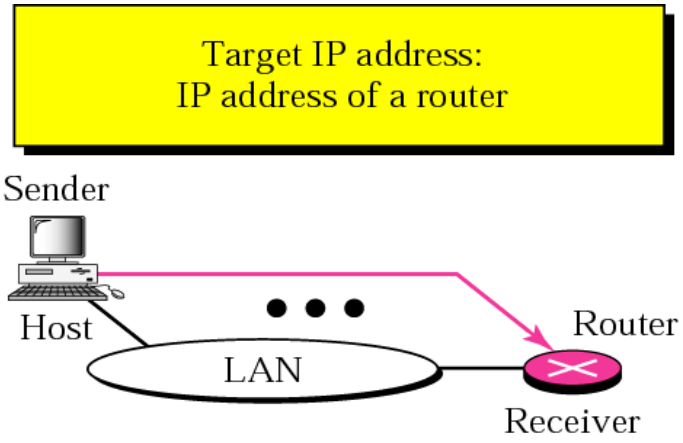
*Encapsulation of ARP packet*



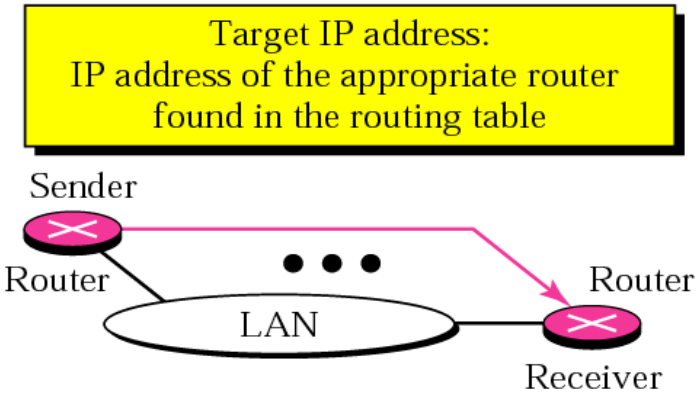
# *Four cases using ARP*



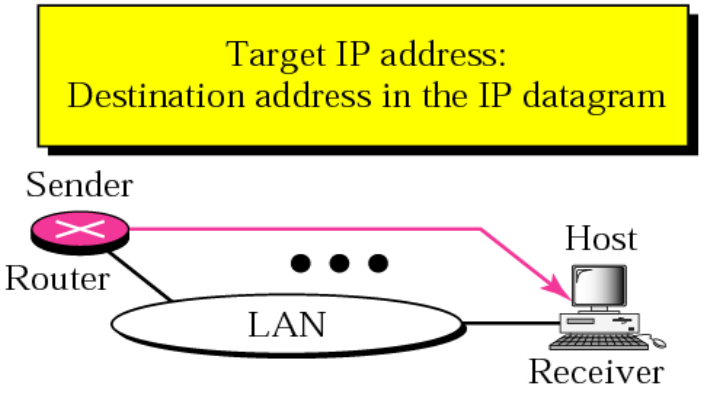

Case 1. A host has a packet to send to another host on the same network.




Case 2. A host wants to send a packet to another host on another network.
It must first be delivered to a router.

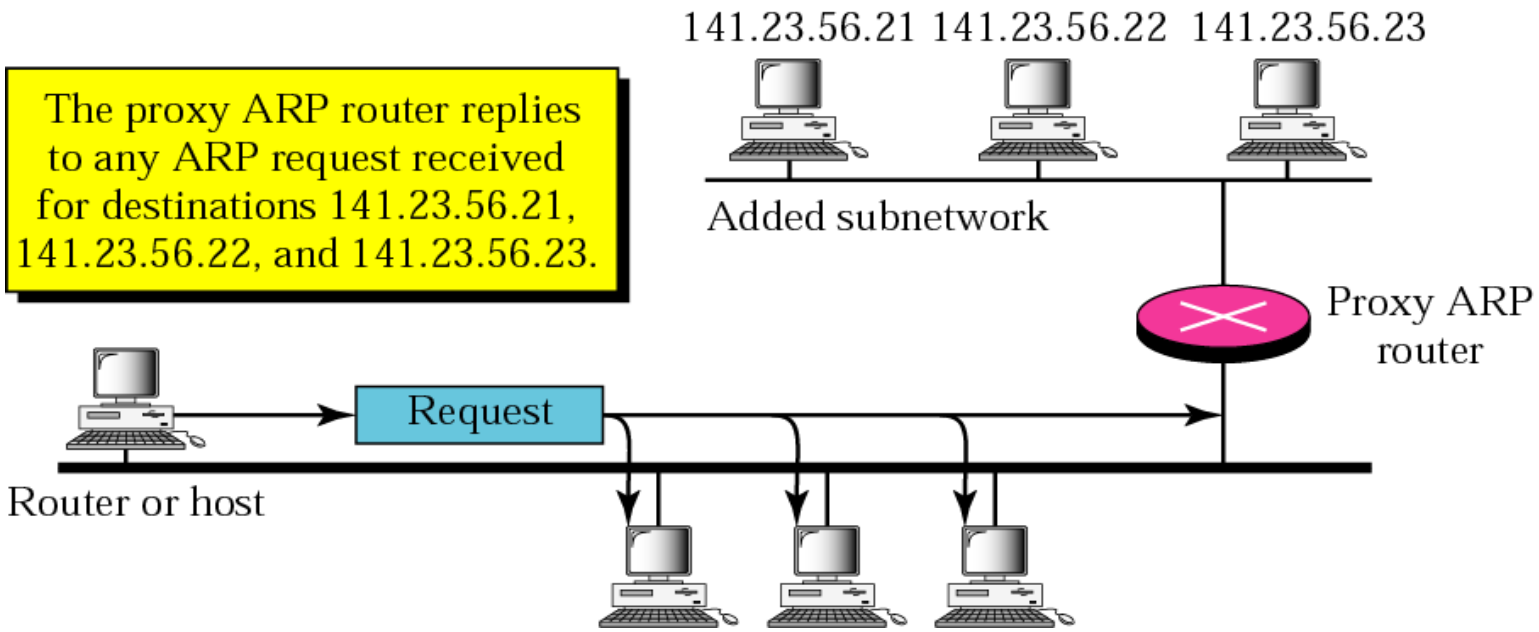



Case 3. A router receives a packet to be sent to a host on another network.
It must first be delivered to the appropriate router.

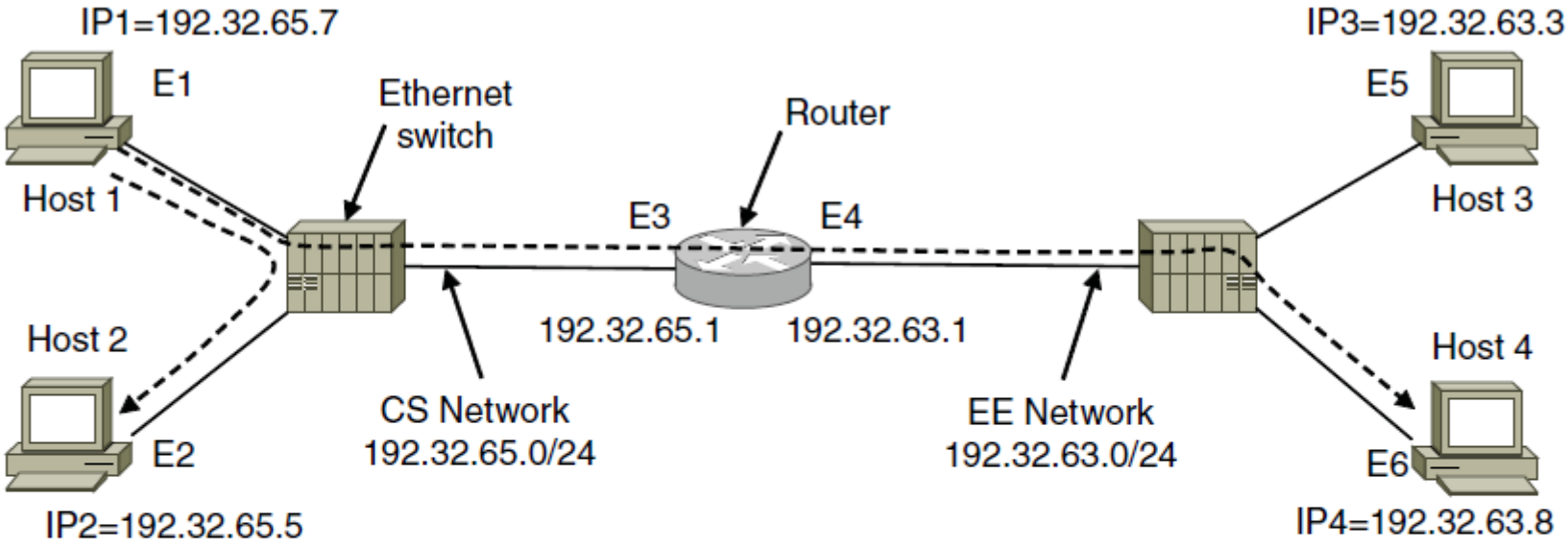



Case 4. A router receives a packet to be sent to a host on the same network.

# *Proxy ARP*



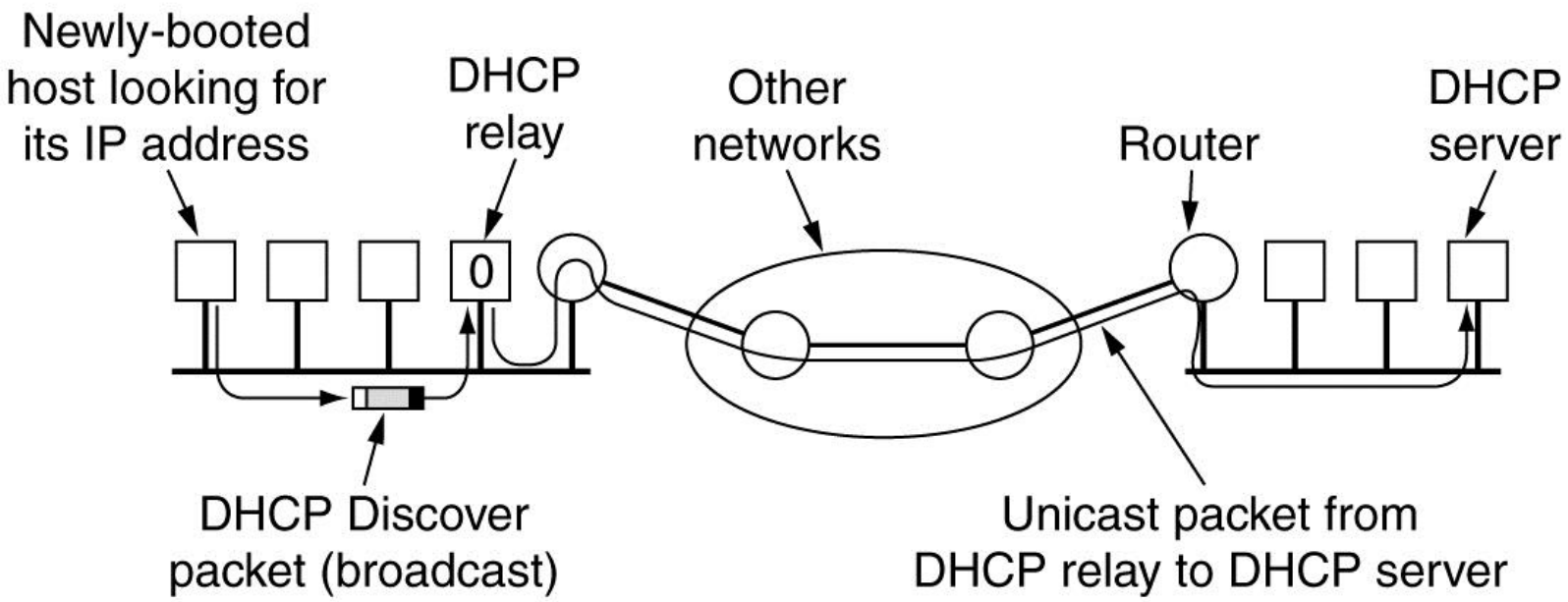
# ARP




| Frame | Source IP | Source Eth. | Destination IP | Destination Eth. |
|---|---|---|---|---|
| Host 1 to 2, on CS net | IP1 | E1 | IP2 | E2 |
| Host 1 to 4, on CS net | IP1 | E1 | IP4 | E3 |
| Host 1 to 4, on EE net | IP1 | E4 | IP4 | E6 |

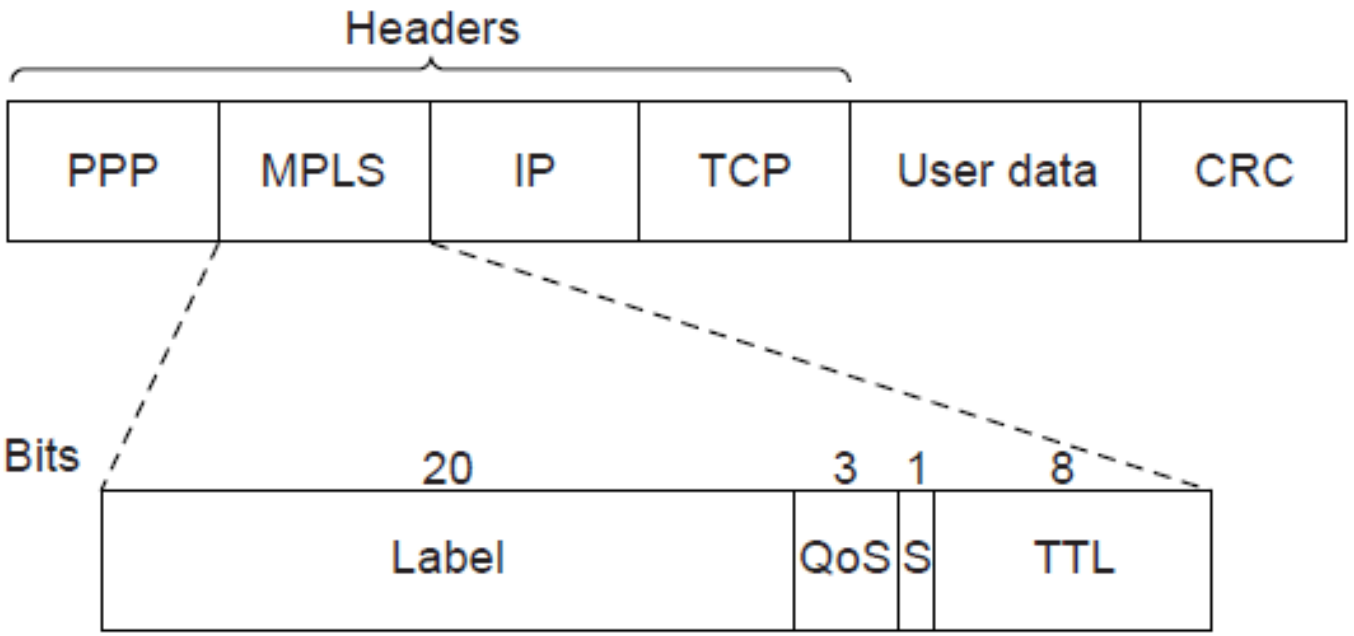
Two switched Ethernet LANs joined by a router

# Dynamic Host Configuration Protocol



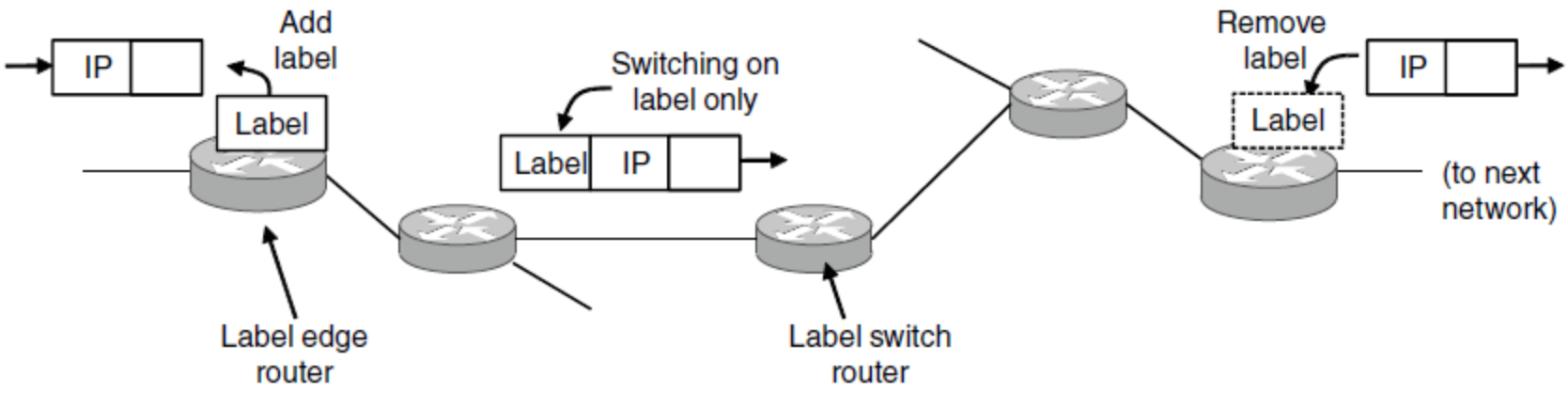

Operation of DHCP.

# Label Switching and MPLS (1)



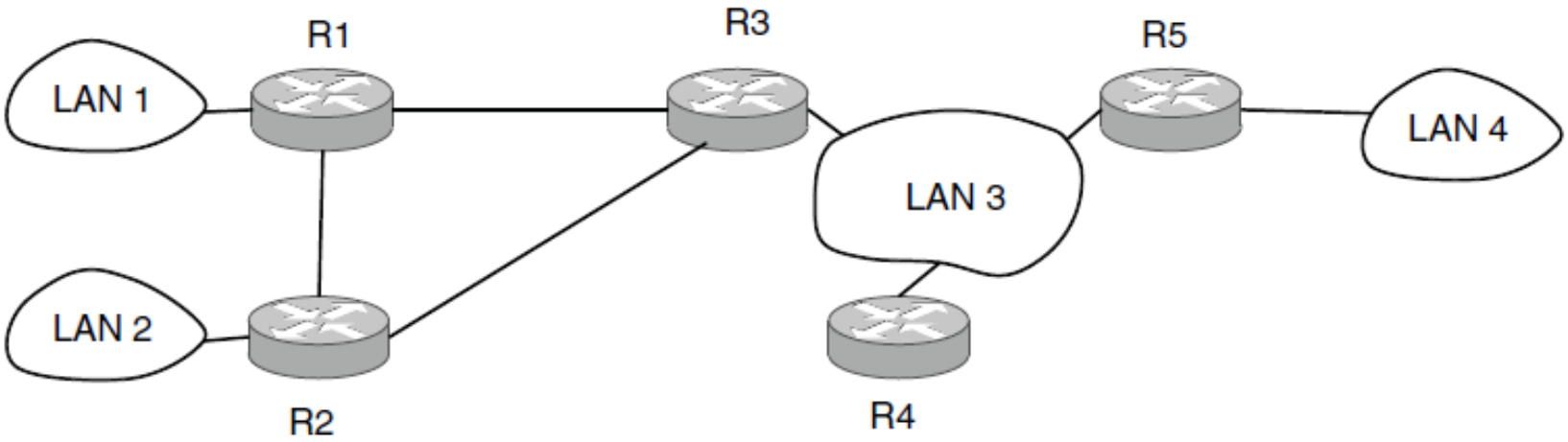

Transmitting a TCP segment using IP, MPLS, and PPP.

# Label Switching and MPLS (2)



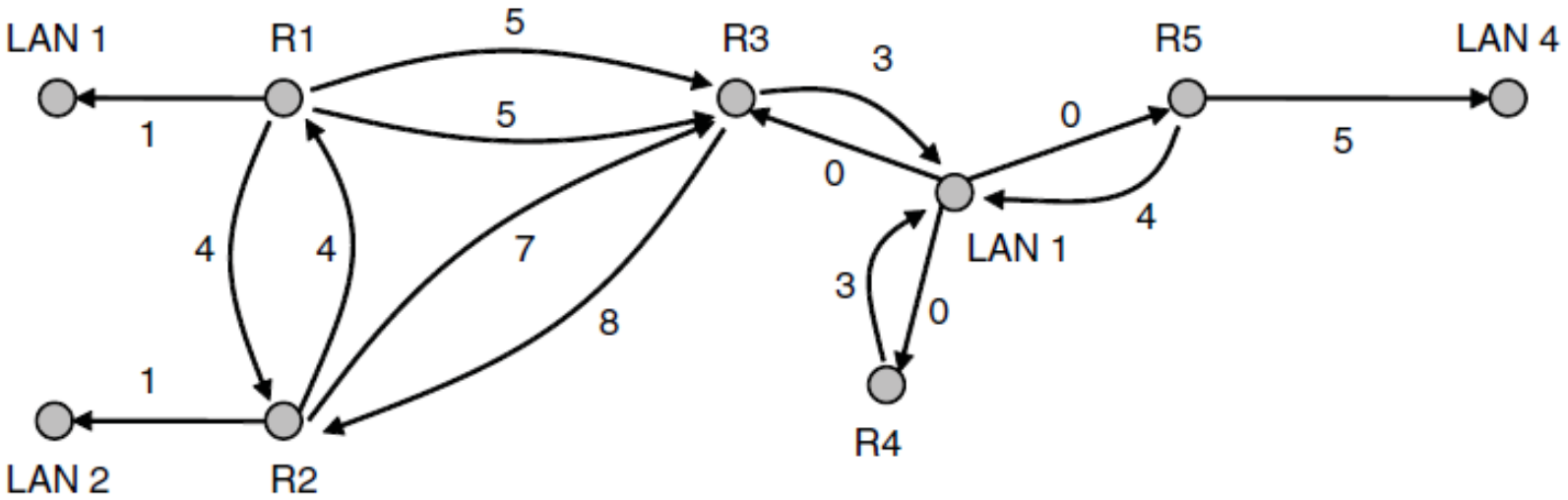

Forwarding an IP packet through an MPLS network

# OSPF—An Interior Gateway Routing Protocol (1)



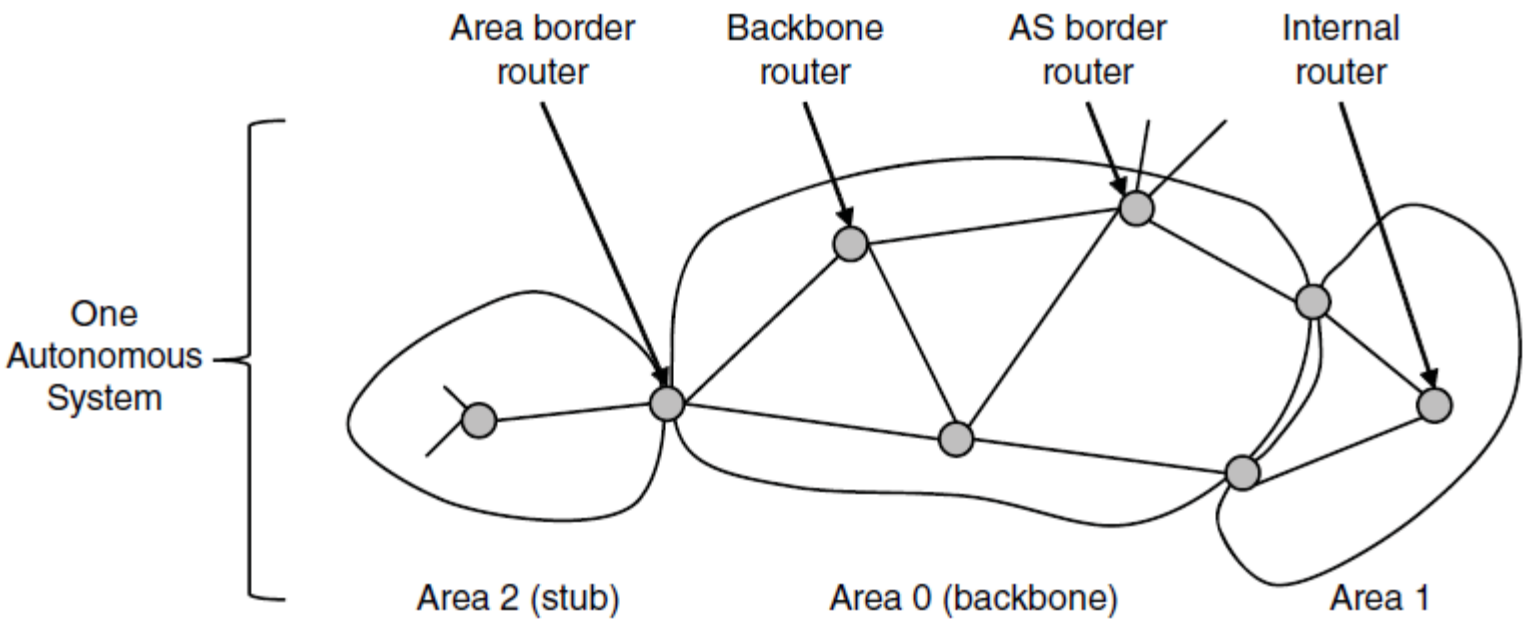

An autonomous system

# OSPF—An Interior Gateway Routing Protocol (2)




A graph representation of the previous slide.

# OSPF—An Interior Gateway Routing Protocol (3)




The relation between ASes, backbones, and areas in OSPF.

# OSPF—An Interior Gateway Routing Protocol (4)

| Message type | Description |
|---|---|
| Hello | Used to discover who the neighbors are |
| Link state update | Provides the sender's costs to its neighbors |
| Link state ack | Acknowledges link state update |
| Database description | Announces which updates the sender has |
| Link state request | Requests information from the partner |

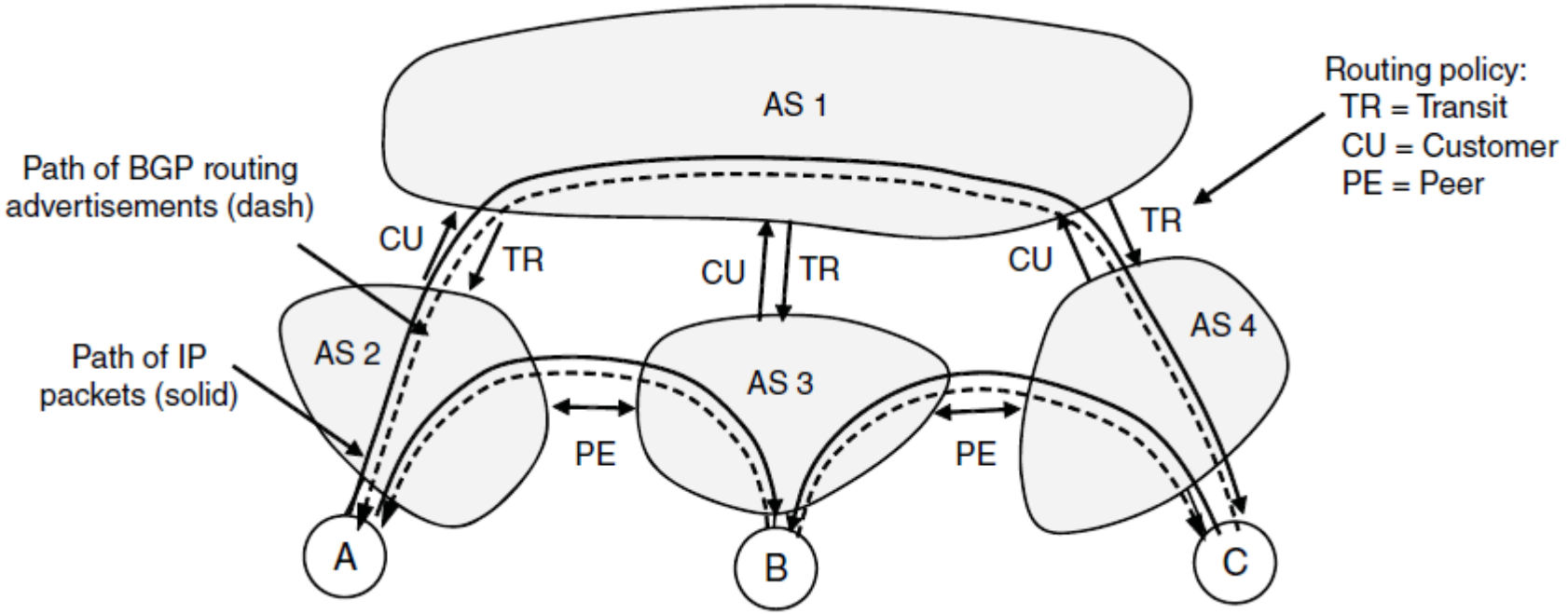
The five types of OSPF messages

# BGP—The Exterior Gateway Routing Protocol (1)

Examples of routing constraints:

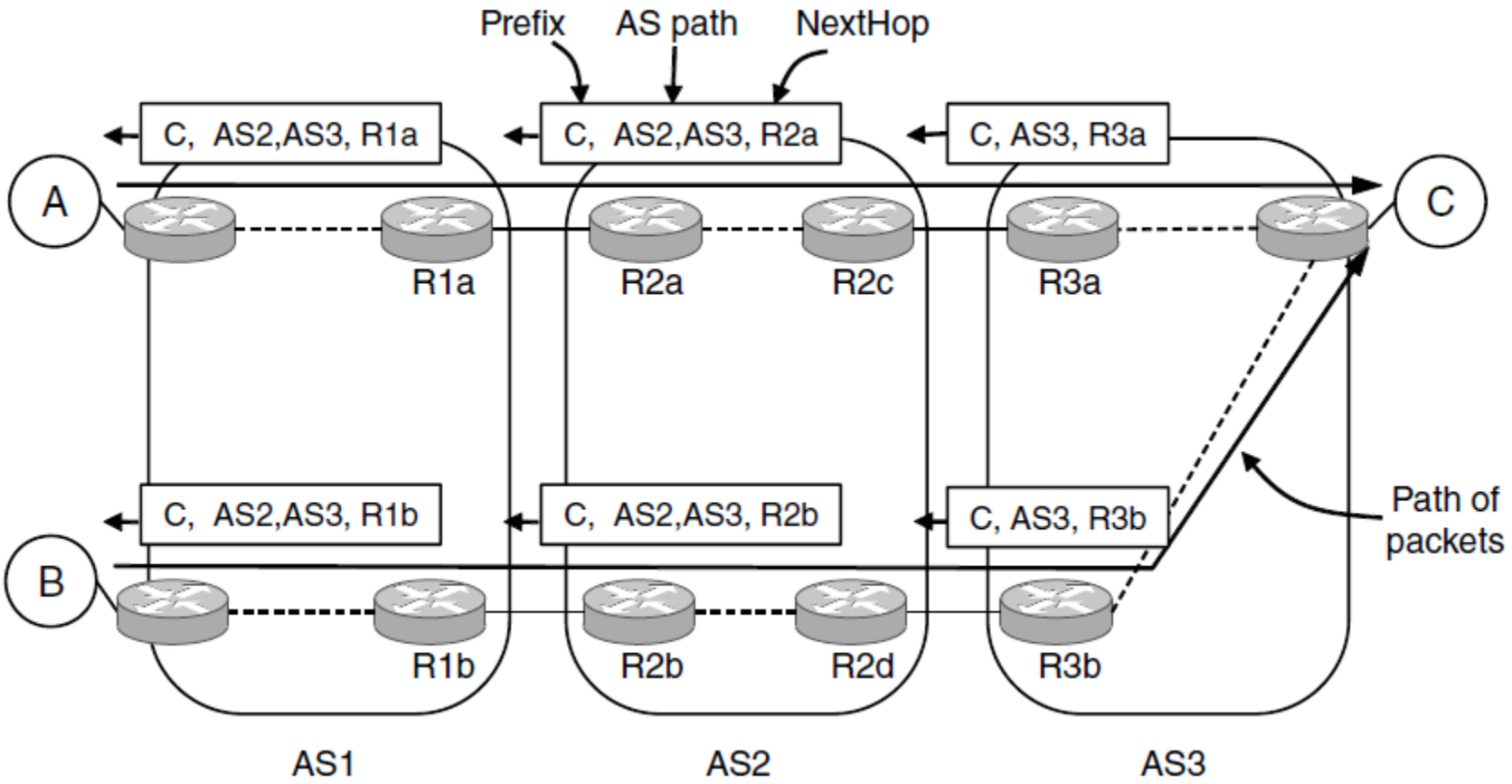
- No commercial traffic for educat. network
- Never put Iraq on route starting at Pentagon
- Choose cheaper network
- Choose better performing network
- Don’t go from Apple to Google to Apple

# BGP—The Exterior Gateway Routing Protocol (2)




Routing policies between four Autonomous Systems

# BGP—The Exterior Gateway Routing Protocol (3)




Propagation of BGP route advertisements

# Mobile IP

Goals

- Mobile host use home IP address anywhere.
- No software changes to fixed hosts
- No changes to router software, tables
- Packets for mobile hosts – restrict detours
- No overhead for mobile host at home.

# End

Chapter 5