Department of
Information Sciences and Telecommunications

Computer Networks – TELCOM 2310

Lecture 5

Network Layer and Internetworking

Routing, Switching and Multicasting

Dr. Znati Lecture Notes

Network Layer Design Issues and Protocols

Goals:

- Understand principles behind network layer services:
  - Routing (path selection)
  - Dealing with scale
  - How a router works
  - Advanced topics: IPv6, multicasting
- Instantiation and implementation in the Internet

Overview:

- Network layer services
- Routing principles: path selection
- Hierarchical routing
- IP
- Internet routing protocols reliable transfer
  - Intra-domain
  - Inter-domain
- What's inside a router?
- IPv6
- Multicasting

Network Design Issues and Protocols Internetworking

1 Introduction and Network Service Models

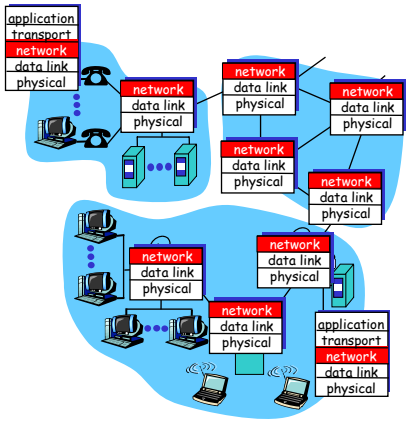
2. Routing Principles
3. Hierarchical Routing
4. The Internet (IP) Protocol
5. Routing in the Internet
6. What's Inside a Router
7. IPv6
8. Multicast Routing

Network Layer Functions

- Transport packet from sending to receiving hosts
- Network layer protocols in *every* host, router

Three important functions:

- *Path determination:* route taken by packets from source to dest. *Routing algorithms*
- *Forwarding:* move packets from router's input to appropriate router output
- *Call setup:* some network architectures require router call setup along path before data flows



Network Service Model

Q: What *service model* for "channel" transporting packets from sender to receiver?

service abstraction

- Guaranteed bandwidth?
- Preservation of inter-packet timing (no jitter)?
- Loss-free delivery?
- In-order delivery?
- Congestion feedback to sender?

The most important abstraction provided by network layer:

virtual circuit or datagram?

Virtual circuits

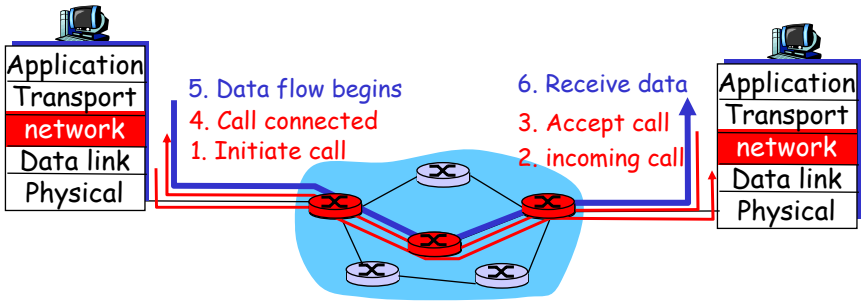
"Source-to-destination path behaves much like telephone circuit"

- Performance-wise
- Network actions along source-to-dest path are required

- Call setup, teardown for each call *before* data can flow
- Each packet carries VC identifier (not destination host ID)
- *Every* router on source-dest path maintains "state" for each passing connection
  - Transport-layer connection only involved two end systems
- Link, router resources (bandwidth, buffers) may be *allocated* to VC
  - To get circuit-like performance

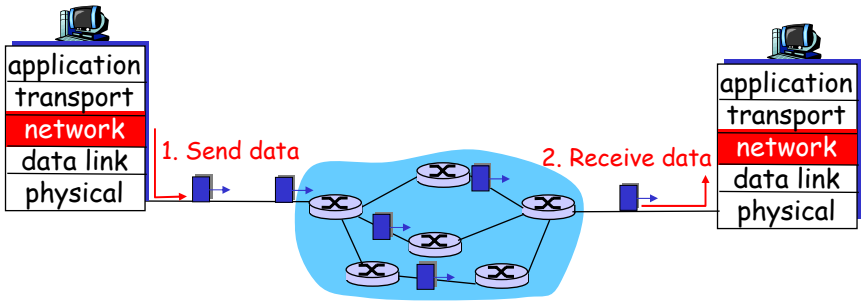
Virtual Circuits: Signaling Protocols

- Used to setup, maintain teardown VC
- Used in Asynchronous Transfer Mode (ATM), frame-relay, X.25
- Not used in today's Internet



Datagram networks: The Internet Model

- No call setup at network layer
- Routers: no state about end-to-end connections
  - No network-level concept of "connection"
- Packets forwarded using destination host address
  - Packets between same source-dest pair may take different paths

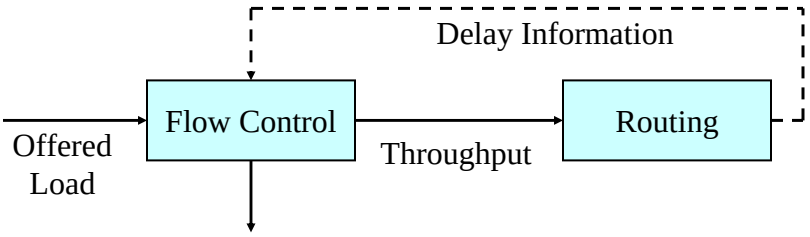


Network Design Issues and Protocols Internetworking

1. Introduction and Network Service Models
2. Routing Principles
   - Routing design issues
   - Routing Algorithms
   - Link state routing
   - Distance vector routing
3. Hierarchical Routing
4. The Internet (IP) Protocol
5. Routing in the Internet
6. What's Inside a Router
7. IPv6
8. Multicast Routing

Routing and Flow Control

- Routing determines paths for data flowing between source and destination pairs
- Flow control regulates traffic to avoid congestion
  - Explicit Congestion Indication
  - Implicit Congestion Indication



Routing

- The principal task of the network layer is to accept packets from the source node and deliver them to the destination node
- Generally, more than one route is possible
  - A routing function must be performed
  - A number of desirable attributes for routing function
    - Correctness, simplicity, robustness, stability, fairness, and optimality

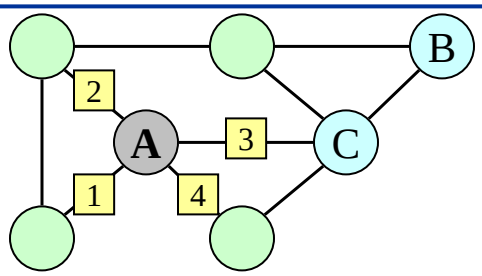
Dimensions of a Routing Task

- In connection-oriented service, routing decisions are only made when a new virtual circuit is being set up
- In datagram service, the decision is made anew at the arrival of every new packet

Routing with Datagrams

- Each node maintains a routing table
  - Entries in the routing table may be based on estimates of congestion in the network
  - Entries may contain alternate links for routing
    - Alternate links can be selected randomly, in a round robin fashion, or used to balance the traffic load

Datagram Routing Tables

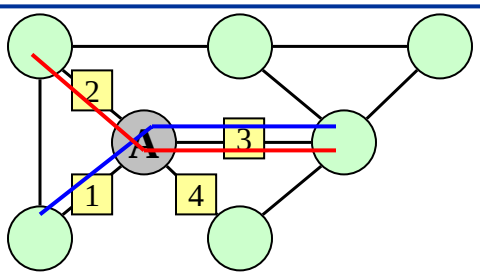


Destination	Link	Fraction
B	2	0.6
	3	0.4
C	3	0.8
	4	0.2

Routing with Virtual Circuits

- Routing tables for virtual circuits indentify circuit numbers instead of destinations
  - Entries in the routing table specify for each incoming link and each incoming virtual circuit number, the corresponding outgoing link and outgoing virtual circuit number
    - Nodes assign numbers independently, so numbers may change along the path

Virtual Circuit Routing Tables



IN		OUT	
Link	Circuit	Link	Circuit
2	1	3	2
3	2	2	1
1	1	3	1
3	1	1	1

Routing Algorithms Desgin Goals

- The main desing goals of a routing algorithm are:
  - Simplicity and low overhead
  - Robustness and stability
  - Rapid convergence (for dynamic routing algorithms)
  - Flexibility
  - Optimality

Simplicity

- Routing algorithms must offer their functionalities Efficiency
  - Minimum of software and utilization overhead
- Efficiency is important, particularly when the software is implemented to run over a computer with limited resource capabilities

Robustness

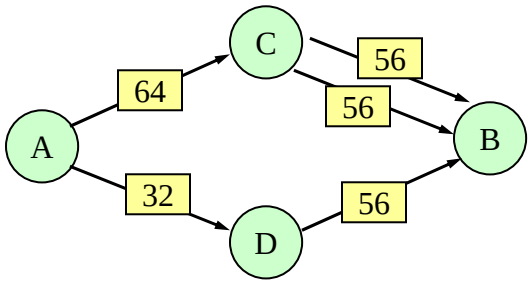
- Routing algorithm must perform correctly despite unforeseen circumstances
  - Hardware failures
  - High load conditions
  - Incorrect implementations
- Unstable routing algorithms can cause considerable problems when they fail at the network junction points

Robustness

- Robustness requires three tasks:
  - Monitoring the network status
  - Update routing decission to reflect new changes
  - Enforce new decisions to react as fast as possible to network changes

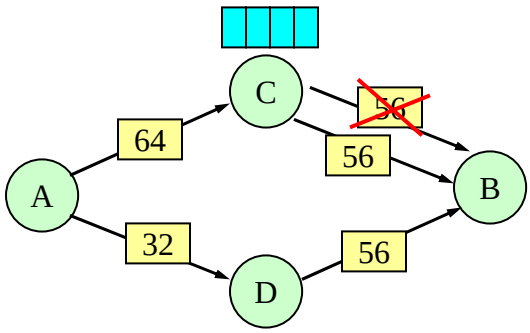
Robustness Example

- Buffers do not accumulate bits, since C and D rate of transmission is faster than input bit rates



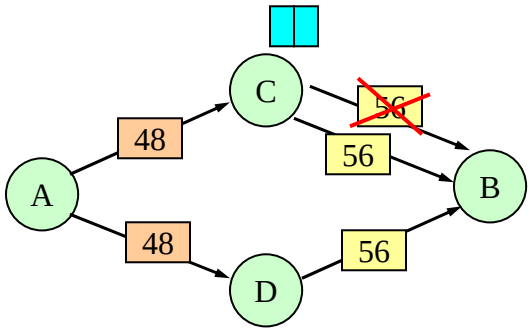
Robustness Example – Link Failure

- Buffering occurs at node C, since the arrival rate is higher than the transmission rate



Robustness Example
Routing Decision Adjustment

- After *S* units of time, A learns of the failure and updates routing decisions to take the failure into consideration
  - Buffers at C begin to empty



Robustness

- The time *S* to learn about the failure has two implications
  - Buffers at C must be able to store the traffic exceeding the output link capacity for the period *S*
  - Bits are delayed at the node C during the failure

Stability and Rapid Convergence

- Convergence is the process of arrangements by the routers on the optimal routes
  - Dialogue among routers is carried by update messages
  - Routing update stimulate recalculations of optimal routes
- Routing algorithm that converge slowly can cause loops or network outages

Slow Convergence and Routing Loops

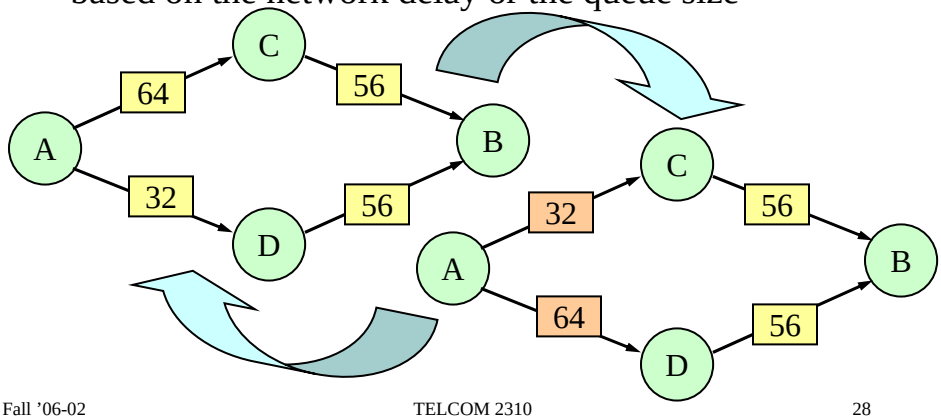
- Some algorithms exhibit slow convergence of routing information
  - Information about possible routes is computed on the basis of notification messages coming from router's neighbors
  - In a case of failure of a distant link, the decision about the best route might be based on an outdated information
  - The process of detecting that there is no valid route to a destination may take long time
  - Switching to another route may create temporary routing loops (until all routers' tables converge to a steady state)

Flexibility

- Flexibility refers to the capacity of the routing algorithm to adapt quickly and accurately to a variety of network circumstances
- Flexible routing algorithms can adapt easily to:
  - Changes in the network bandwidth
  - Router queue size
  - Network delay
  - Link failure

Stability

- The existence of different routes to a destination means possible stability problems, if routing decisions are based on the network delay or the queue size



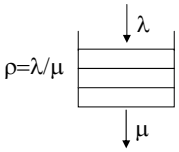
Routing Desirable Characteristics – Fairness

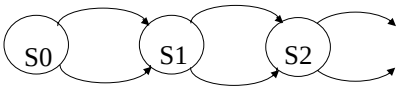
- Routing must accomodate traffic with different quality of service requirements (QoS)
- Packets within the same class should suffer similar delays

Routing Fairness

M/M/1 Queue Model

- The input and output to the queue are Poisson distributed.
- λ: input data rate (w/s, pix/BCO)
- μ: output data rate
- ρ: traffic intensity

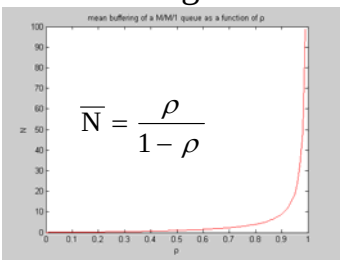




State diagram

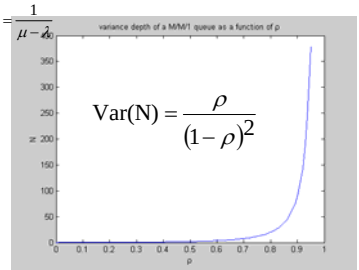
$$p_k = \rho^k p_0 \quad p_0 = 1-\rho$$

Steady State equations



$$\overline{N} = \frac{\rho}{1-\rho}$$

$$D = \frac{1}{\mu - \lambda}$$



$$\mathrm{Var}(N) = \frac{\rho}{(1-\rho)^2}$$

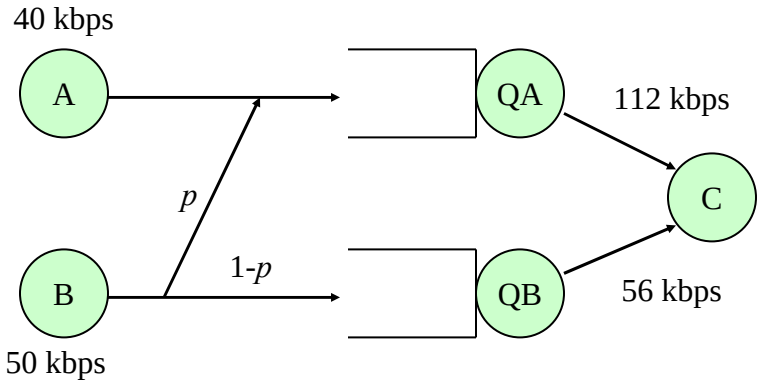
Fairness - Example

- Little's Law is commonly used law to compute the delay, *D*, a packet suffers at a node:

$$\overline{N} = \lambda D \qquad D = \frac{1}{\mu - \lambda}$$

  - μ - the transmission rate
  - λ - the arrival rate

Fairness - Example



Fairness – Example

- Using the delay formula, we can write for the stream incoming to the queue A:

$$D_A = \frac{1}{112 - \lambda_A}$$

$$\lambda_A = 40 + 50p$$

$$D_A = \frac{1}{72 - 50p}$$

Fairness – Example

- Similarly, using the delay expression, we can write for the stream incoming to the queue B:

$$D_B = \frac{1}{56 - \lambda_B}$$

$$\lambda_B = 50 - 50p$$

$$D_B = \frac{1}{6 + 50p}$$

Fairness – Example

- A fair routing algorithm must select p such that:

$$D_A = D_B$$

- Solving for p results in:

$$72 - 50p = 6 + 50p$$
$$66 = 100p$$
$$p = 0.66$$

Optimality

- Optimality refers to the capability of the routing algorithm to find the "best route"
  - It is strictly defined with respect to the routing metrics and metric weightings
- May be difficult to achieve

Routing Metrics

- Path length, sum of the costs associated with links
  - Frequently reduced to hop count
- Reliability, link bit error rate, usually assigned by a network administrator
- Delay, as the amount of time required to move packets from a source to a destination
  - Delay depends on the link bandwidth, router queues and the network load
- Communication costs to reduce expenditures

Optimality

- Consider the previous example and assume that the objective is to minimize the average packet delay:

$$D = \frac{40}{40+50} D_a + \frac{50}{40+50} D_b$$

$$D_a = D_A$$

$$D_b = pD_A + (1-p)D_B$$

Optimality

- Taking the derivative of the delay expression D with respect to p and setting it to 0 produces the value of p that minimizes the average network delay
  - Solving the equation results in:

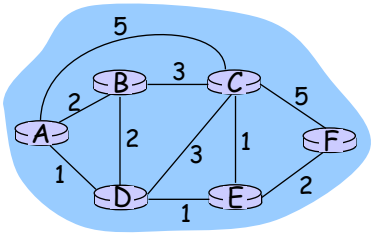
$$D = \frac{40}{90}\frac{1}{72-50p} + \frac{50}{90}\left(\frac{p}{72-50p} + \frac{1-p}{6+50p}\right)$$

$$\frac{70}{9}\left(\frac{2}{(36-25p)^2} - \frac{1}{(3+25p)^2}\right) = 0 \qquad p \approx 0.526$$

Routing Algorithms Design

Routing protocol

Goal: determine "good" network path, sequence of routers, from source to dest.



Graph abstraction for routing algorithms:

- Graph nodes are routers
- Graph edges are physical links
  - Link cost: delay, $ cost, or congestion level

- "Good" path:
  - Typically means minimum cost path
  - Other definitions are also possible

Routing Algorithm Classification

Global or decentralized information?

Global:

- All routers have complete topology, link cost info
- "Link State" Algorithms

Decentralized:

- Router knows physically-connected neighbors, link costs to neighbors
- Iterative process of computation, exchange of info with neighbors
- "Distance Vector" Algorithms

Static or dynamic?

Static:

- Routes change slowly over time

Dynamic:

- Routes change more quickly
  - Periodic update
  - In response to link cost changes

Routing Techniques
Path Optimality Principle

- "If a router J is in the optimal path from I to K, then the optimal path from J to K also falls along the same route"
- As a direct consequence, the set of optimal routes from all sources to a given destination from a tree rooted at the destination
  - The tree is referred to as "a sink tree"
- Link State and Distance routing algorithms

A Link-State Routing Algorithm

Dijkstra's Algorithm

- Network topology, link costs known to all nodes
  - Accomplished via "link state broadcast"
  - All nodes have same info
- Computes least cost paths from one node ('source") to all other nodes
  - gives routing table for that node
- Iterative: after k iterations, know least cost path to k dest.'s

Notation:

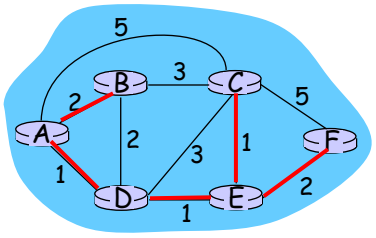
- c(i,j): link cost from node i to j. cost infinite if not direct neighbors
- D(v): current value of cost of path from source to dest. V
- p(v): predecessor node along path from source to v, that is next v
- N: set of nodes whose least cost path definitively known

Dijsktra's Algorithm

```
1  Initialization:
2    N = {A}
3    for all nodes v
4      if v adjacent to A
5        then D(v) = c(A,v)
6        else D(v) = infinity
7
8  Loop
9     find w not in N such that D(w) is a minimum
10    add w to N
11    update D(v) for all v adjacent to w and not in N:
12       D(v) = min( D(v), D(w) + c(w,v) )
13    /* new cost to v is either old cost to v or known
14     shortest path cost to w plus cost from w to v */
15 until all nodes in N
```

Dijkstra's algorithm: example

Step	start N	D(B),p(B)	D(C),p(C)	D(D),p(D)	D(E),p(E)	D(F),p(F)
0	A	2,A	5,A	1,A	infinity	infinity
1	AD	2,A	4,D		2,D	infinity
2	ADE	2,A	3,E			4,E
3	ADEB		3,E			4,E
4	ADEBC					4,E
5	ADEBCF					



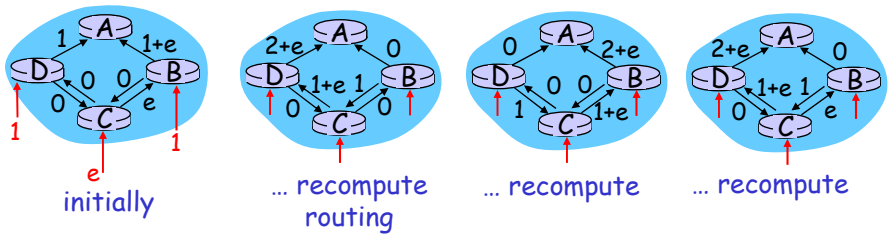
Dijkstra's algorithm, discussion

Algorithm complexity: n nodes

- Each iteration: need to check all nodes, w, not in N
  - n*(n+1)/2 comparisons: O(n**2)
- More efficient implementations possible: O(nlogn)

Oscillations possible:

- e.g., link cost = amount of carried traffic



Distance Vector Routing Algorithm

Iterative:

- continues until no nodes exchange info.
- *self-terminating*: no "signal" to stop

Asynchronous:

- nodes need *not* exchange info/iterate in lock step!

Distributed:

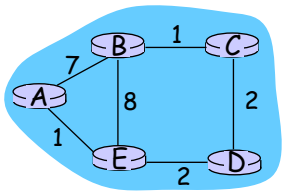
- each node communicates *only* with directly-attached neighbors

Distance Table data structure

- Each node has its own
- Row for each possible destination
- Column for each directly-attached neighbor to node
- Example: in node X, for dest. Y via neighbor Z:

$$D^X(Y,Z) = \text{distance } \textit{from } X \textit{ to } Y, \textit{ via } Z \text{ as next hop}$$
$$= c(X,Z) + \min_w\{D^Z(Y,w)\}$$

Distance Table: example

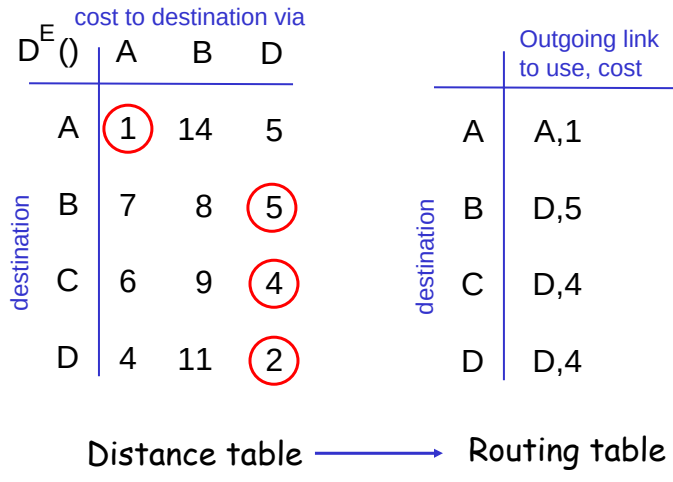


$$D^E(C,D) = c(E,D) + \min_w\{D^D(C,w)\}$$
$$= 2+2 = 4$$
$$D^E(A,D) = c(E,D) + \min_w\{D^D(A,w)\}$$
$$= 2+3 = 5 \quad \text{loop!}$$
$$D^E(A,B) = c(E,B) + \min_w\{D^B(A,w)\}$$
$$= 8+6 = 14 \quad \text{loop!}$$

$D^E()$ — cost to destination via

destination	A	B	D
A	1	14	5
B	7	8	5
C	6	9	4
D	4	11	2

Distance table gives routing table



Distance Vector Routing: overview

Iterative, asynchronous:

- Each local iteration caused by:
  - Local link cost change
  - Message from neighbor: its least cost path change from neighbor

Distributed:

- Each node notifies neighbors *only* when its least cost path to any destination changes
  - Neighbors then notify their neighbors if necessary

Each node:

wait for (change in local link cost of msg from neighbor)

↓

recompute distance table

↓

if least cost path to any dest has changed, *notify* neighbors

Distance Vector Algorithm:

At all nodes, X:

1 Initialization:

2 For all adjacent nodes v:

3 $D^X(*,v) = \text{infinity}$ /* The * operator means "for all rows". */

4 $D^X(v,v) = c(X,v)$

5 For all destinations, y

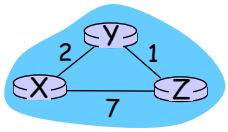
6 Send $\min_w D^X(y,w)$ to each neighbor /* w over all X's neighbors */

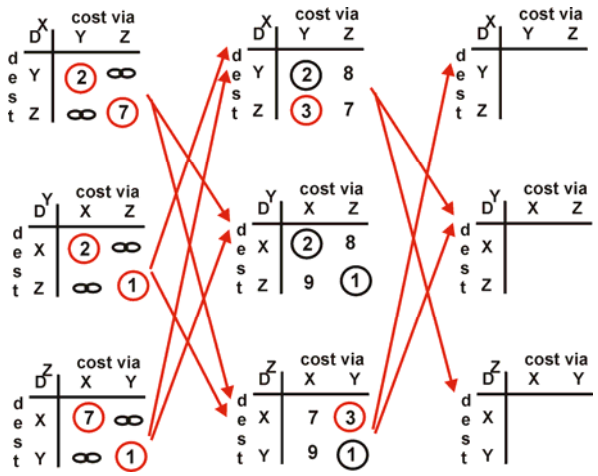
Distance Vector Algorithm

8 **Loop**

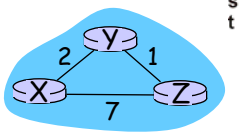
9 **Wait** (until I see a link cost change to neighbor V
10 or until I receive update from neighbor V)
11 **if** (c(X,V) changes by d)
12 /* change cost to all dest's via neighbor v by d */
13 /* note: d could be positive or negative */
14 for all destinations y: $D^X(y,V) = D^X(y,V) + d$
15
16 **else if** (update received from V wrt destination Y)
17 /* shortest path from V to some Y has changed */
18 /* V has sent a new value for its $\min_w DV(Y,w)$ */
19 /* call this received new value is "newval" */
20 for the single destination y: $D^X(Y,V) = c(X,V) + newval$
21
22 **if** we have a new $\min_w D^X(Y,w)$ for any destination Y
23 send new value of $\min_w D^X(Y,w)$ to all neighbors
24 **Forever**

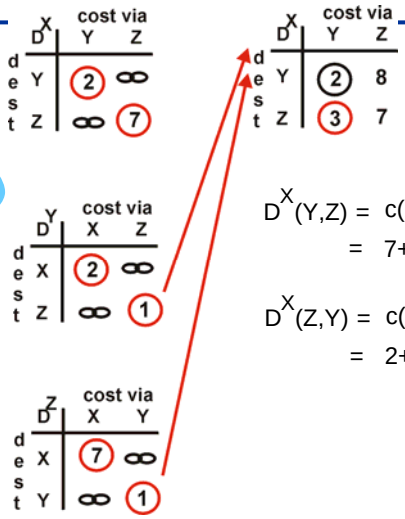
Distance Vector Algorithm: Example





Distance Vector Algorithm: example





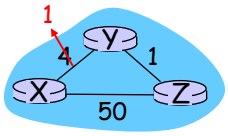
$D^X(Y,Z) = c(X,Z) + \min_w\{D^Z(Y,w)\}$
$= 7+1 = 8$

$D^X(Z,Y) = c(X,Y) + \min_w\{D^Y(Z,w)\}$
$= 2+1 = 3$

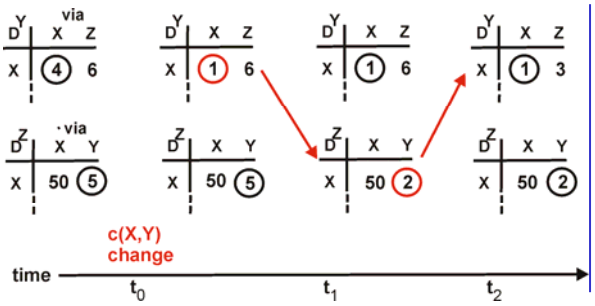
Distance Vector: Link Cost Changes

Link Cost Changes:

- Node detects local link cost change
- Updates distance table: line 15.
- Notify neighbors, if cost change in least cost path : lines 22 and 23.



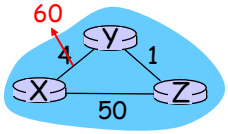
"Good news travels fast"

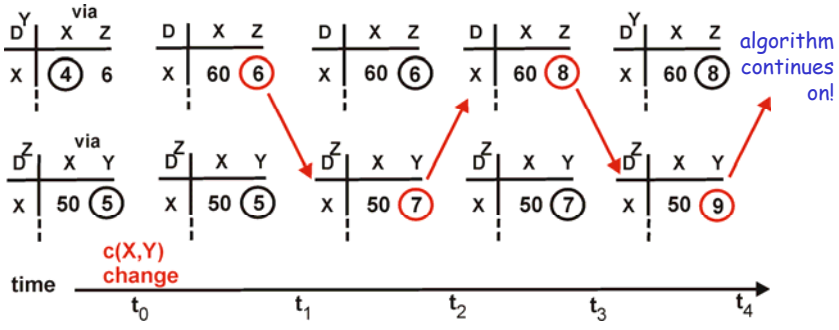


Distance Vector: Link Cost Changes

Link cost changes:

- Good news travels fast
- Bad news travels slow - "count to infinity" problem!



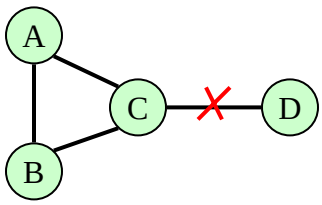


Infinite Loop Fixes
Split Horizons

- Split horizons
  - Never send the information about a route back in the direction from which it came
  - Helps prevent two-node routing loops
  - The solution may fail in some cases

Split Horizon

- Initially both A and B have a distance 2 to D by C, and C has a distance 1 (direct)
- The link C-D goes down, C inserts infinity in the routing table and advertises it to A and B
- Neither A and B tell C their routing entries for D, however they exchange that information between themselves
- When A receives information from B, it advertises it to C rather than B, C updates its routing table and advertises it to B
- All three nodes gradually increase the distance towards the infinity

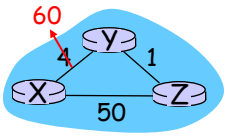


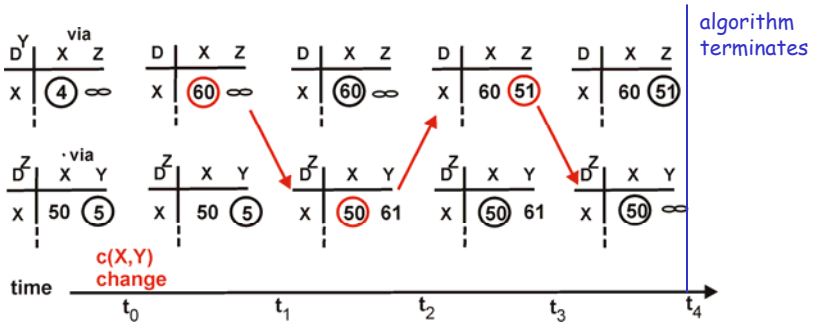
Split Horizon with Poison Reverse

- Split horizon with poison reverse improves distance vector convergence over simple split horizon by advertising all network IDs to all neighbors
  - But those network IDs learned from a given direction are advertised to the same direction with a metric of ∞, indicating that the network is unavailable
- Poison reverse has no benefit beyond split horizon in a single path internetwork
- In a multipath internetwork, split horizon with poison reverse greatly reduces count to infinity and routing loops

Split Horizon with Poison Reverse Example

- If Z routes through Y to get to X :
  - Z tells Y its (Z's) distance to X is infinite (so Y won't route to X via Z)
  - Will this completely solve count to infinity problem?





Infinite Loop Fixes
Poison Reverse Updates and Hold Down

- Increases in routing metrics generally indicate routing loops
  - When increase is observed, send "Poison Reverse" updates to remove route and place it in "Hold Down"
    - "Hold Down" causes discarding of any changes affecting recently removed routes for a period of time
- Prevents larger routing loops, but may slow down the convergence

Comparison of LS and DV algorithms

Message complexity
- LS: with n nodes, E links, O(nE) msgs sent each
- DV: exchange between neighbors only
  - convergence time varies

Speed of Convergence
- LS: $O(n^2)$ algorithm requires O(nE) msgs
  - may have oscillations
- DV: convergence time varies
  - may be routing loops
  - count-to-infinity problem

Robustness: what happens if router malfunctions?

LS:
- node can advertise incorrect *link* cost
- each node computes only its *own* table

DV:
- DV node can advertise incorrect *path* cost
- each node's table used by others
  - error propagate thru network

Network Layer Design Issues and Protocols

1. Introduction and Network Service Models
2. Routing Principles
3. Hierarchical Routing
4. The Internet (IP) Protocol
5. Routing in the Internet
6. What's Inside a Router
7. IPv6
8. Multicast Routing

Hierarchical Routing

Our routing framework thus far - idealization

- All routers identical
- Network "flat"

... *not* true in practice

Scale: with 200 million destinations:

- Can't store all destinations in routing tables!
- Routing table exchange would swamp links!

Administrative autonomy

- Internet = network of networks
- Each network admin may want to control routing in its own network

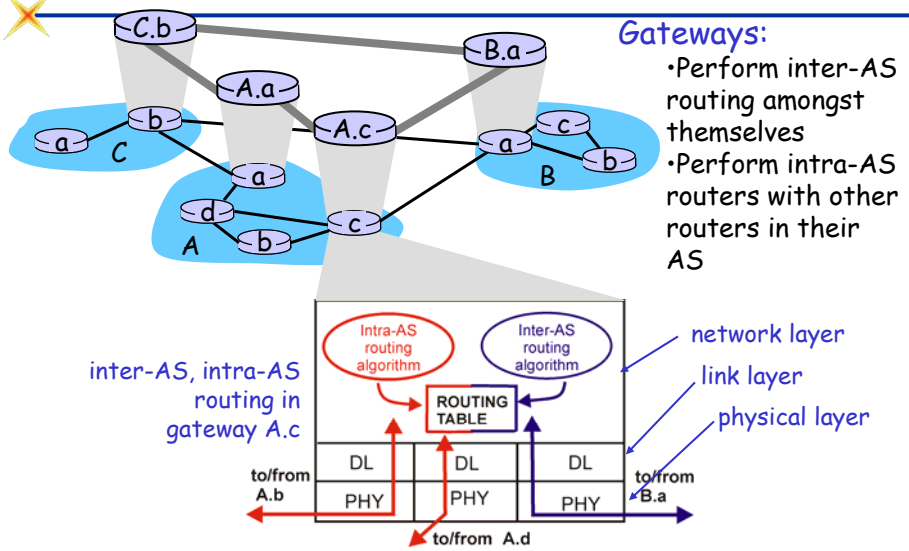
Hierarchical Routing

- Aggregate routers into regions, "autonomous systems" (AS)
- Routers in same AS run same routing protocol
  - "Intra-AS" routing protocol
  - Routers in different AS can run different intra-AS routing protocol

Gateway routers

- Special routers in AS
- Run intra-AS routing protocol with all other routers in AS
- *Also* responsible for routing to destinations outside AS
  - Run *inter-AS routing* protocol with other gateway routers

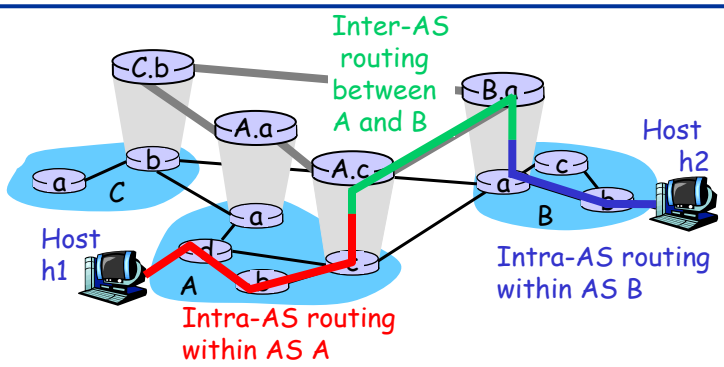
Intra-AS and Inter-AS routing



Gateways:

- Perform inter-AS routing amongst themselves
- Perform intra-AS routers with other routers in their AS

Intra-AS and Inter-AS routing



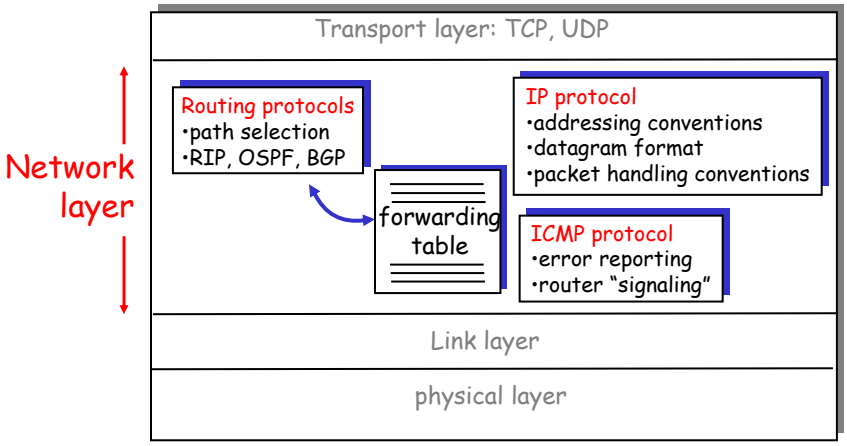
- We'll examine specific inter-AS and intra-AS Internet routing protocols shortly

Internet Protocol

1. Introduction and Network Service Models
2. Routing Principles
3. Hierarchical Routing
4. The Internet (IP) Protocol
   - 4.1 IPv4 addressing
   - 4.2 Moving a datagram from source to destination
   - 4.3 Datagram format
   - 4.4 IP fragmentation
   - 4.5 ICMP: Internet Control Message Protocol
   - 4.6 DHCP: Dynamic Host Configuration Protocol
   - 4.7 NAT: Network Address Translation
5. Routing in the Internet
6. What's Inside a Router
7. IPv6
8. Multicast Routing

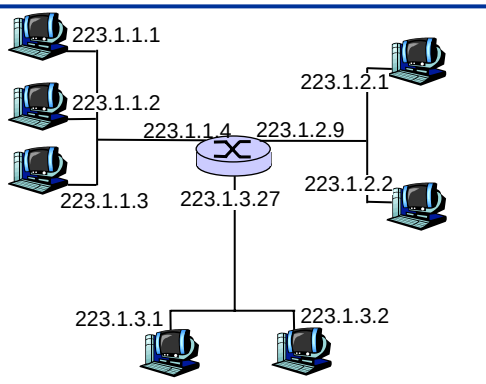
The Internet Network layer

Host, router network layer functions:



IP Addressing: Introduction

- IP address: 32-bit identifier for host, router *interface*
- *Interface:* connection between host/router and physical link
  - Router's typically have multiple interfaces
  - Host may have multiple interfaces
  - IP addresses associated with each interface

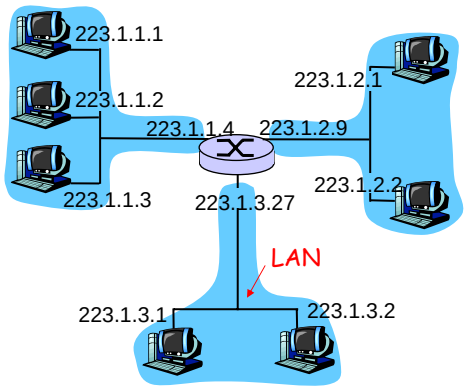


223.1.1.1 = 11011111 00000001 00000001 00000001

223 1 1 1

IP Addressing

- IP address:
  - network part (high order bits)
  - host part (low order bits)
- *What's a network ?* (from IP address perspective)
  - Device interfaces with same network part of IP address
  - Can physically reach each other without intervening router

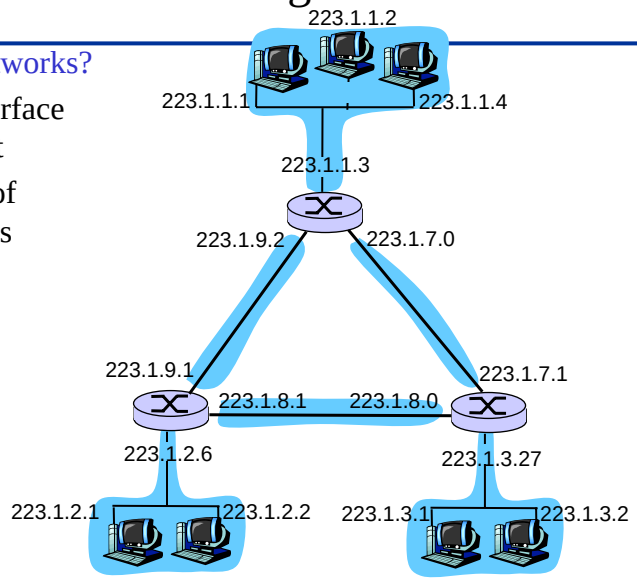


network consisting of 3 IP networks (for IP addresses starting with 223, first 24 bits are network address)

IP Addressing

How to find the networks?

- Detach each interface from router, host
- Create "islands of isolated networks

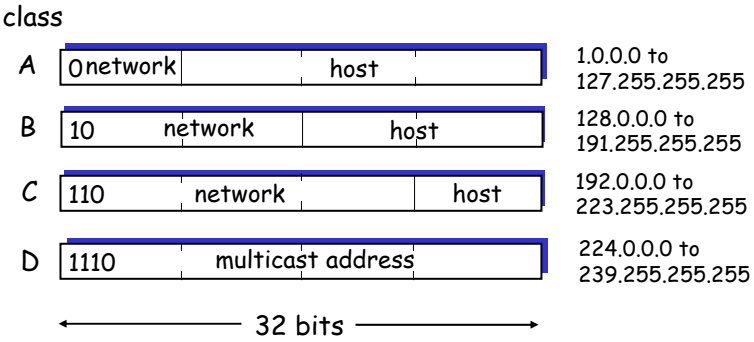


Interconnected system consisting of six networks

IP Addresses

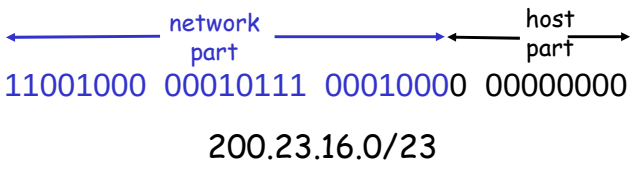
Given notion of a "network", let's re-examine IP addresses:

"Class-full" addressing:



IP addressing: CIDR

- Class-full addressing:
  - Inefficient use of address space, address space exhaustion
  - For example, class B net allocated enough addresses for 65K hosts, even if only 2K hosts in that network
- CIDR: Classless InterDomain Routing
  - Network portion of address of arbitrary length
  - Address format: a.b.c.d/x, where x is # bits in network portion of address



200.23.16.0/23

IP addresses: Host Address Acquisition?

Q: How does *host* get IP address?

- Hard-coded by system admin in a file
  - Wintel: control-panel->network->configuration->tcp/ip->properties
  - UNIX: /etc/rc.config
- DHCP: Dynamic Host Configuration Protocol: dynamically get address from as server
  - "plug-and-play"

IP addresses: Network Address Acquisition?

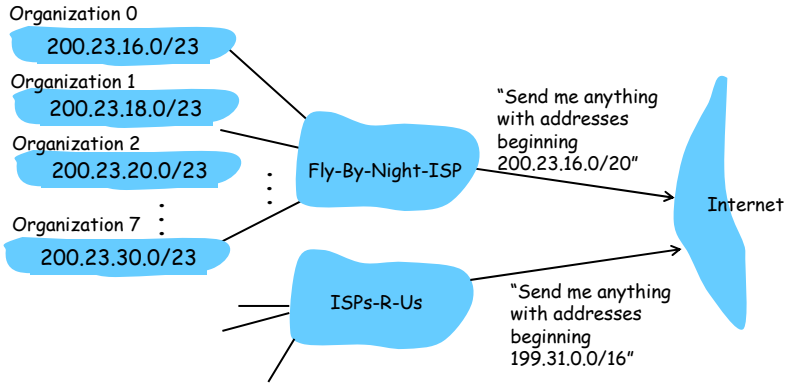
Q: How does *network* get network part of IP address?

A: Gets allocated portion of its provider ISP's address space

ISP's block	11001000 00010111 00010000	00000000	200.23.16.0/20
Organization 0	11001000 00010111 00010000	00000000	200.23.16.0/23
Organization 1	11001000 00010111 00010010	00000000	200.23.18.0/23
Organization 2	11001000 00010111 00010100	00000000	200.23.20.0/23
...	.....	....	....
Organization 7	11001000 00010111 00011110	00000000	200.23.30.0/23

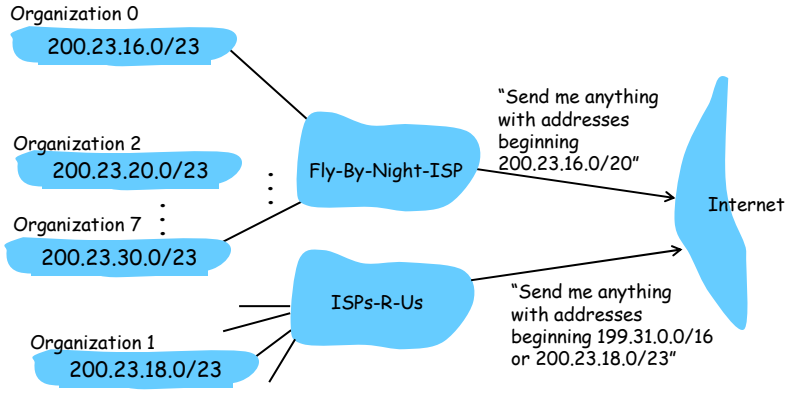
Hierarchical Addressing: Route Aggregation

Hierarchical addressing allows efficient advertisement of routing information:



Hierarchical Addressing: More Specific Routes

ISPs-R-Us has a more specific route to Organization 1



IP addressing: The Last Word...

Q: How does an ISP get block of addresses?

A: ICANN: Internet Corporation for Assigned Names and Numbers

- Allocates addresses
- Manages DNS
- Assigns domain names, resolves disputes

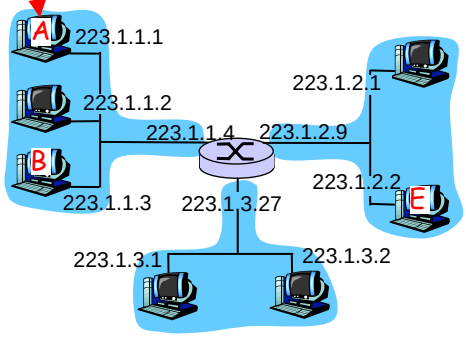
Getting a Datagram from Source to Destination

IP datagram:

- Datagram remains unchanged, as it travels source to destination
- Address fields of interest here

Forwarding Table in A

Dest. Net.	next router	Nhops
223.1.1		1
223.1.2	223.1.1.4	2
223.1.3	223.1.1.4	2



Getting a Datagram From Source to Destination

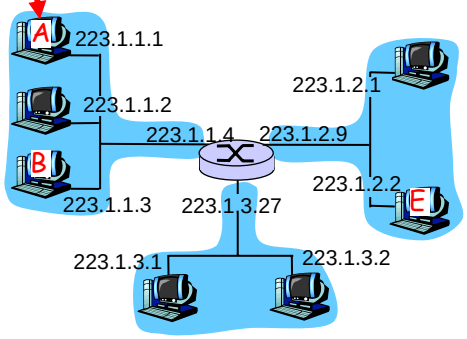
misc fields	223.1.1.1	223.1.1.3	data

Starting at A, send IP datagram addressed to B:

- Look up net. address of B in forwarding table
- Find B is on same net. as A
- Link layer will send datagram directly to B inside link-layer frame
  - B and A are directly connected

Forwarding Table in A

Dest. Net.	next router	Nhops
223.1.1		1
223.1.2	223.1.1.4	2
223.1.3	223.1.1.4	2



Getting a Datagram From Source to Destination

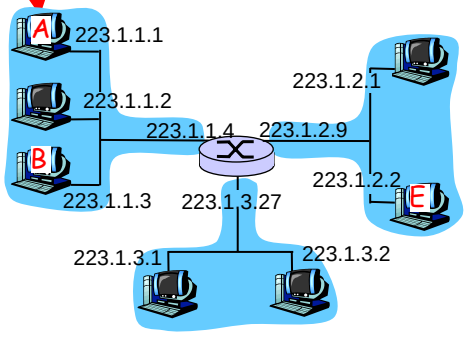
misc fields	223.1.1.1	223.1.2.3	data

Starting at A, dest. E:

- Look up network address of E in forwarding table
- E on *different* network
  - A, E not directly attached
- Routing table: next hop router to E is 223.1.1.4
- Link layer sends datagram to router 223.1.1.4 inside link-layer frame
- Datagram arrives at 223.1.1.4
- continued…..

Forwarding Table in A

Dest. Net.	next router	Nhops
223.1.1		1
223.1.2	223.1.1.4	2
223.1.3	223.1.1.4	2



Getting a Datagram from Source to Destination

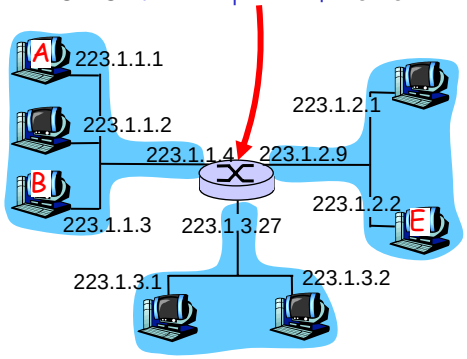
misc fields	223.1.1.1	223.1.2.3	data

Arriving at 223.1.4, destined for 223.1.2.2

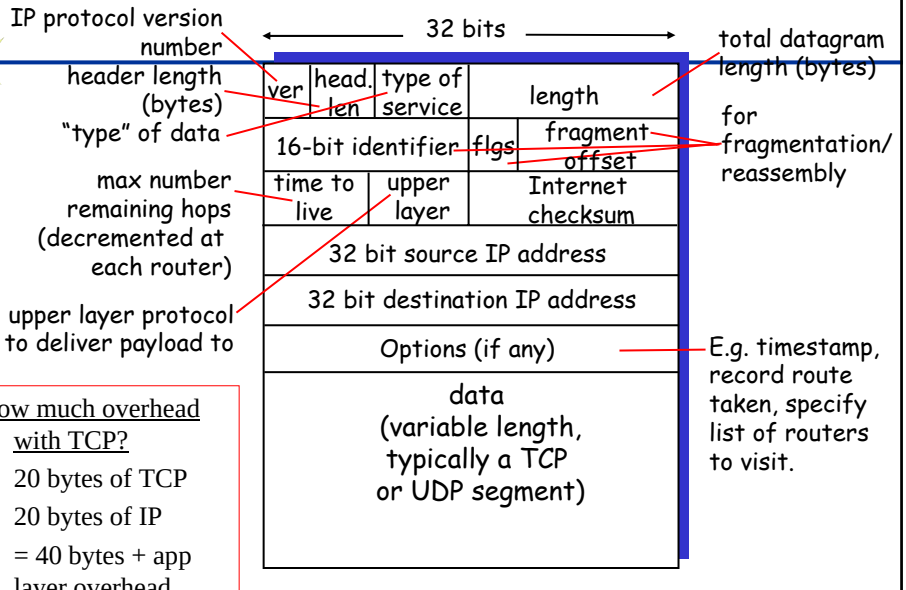
- Look up network address of E in router's forwarding table
- E on *same* network as router's interface 223.1.2.9
  - router, E directly attached
- Link layer sends datagram to 223.1.2.2 inside link-layer frame via interface 223.1.2.9
- Datagram arrives at 223.1.2.2!!!

Forwarding Table in Router

Dest. Net	router	Nhops	interface
223.1.1	-	1	223.1.1.4
223.1.2	-	1	223.1.2.9
223.1.3	-	1	223.1.3.27



IP datagram format

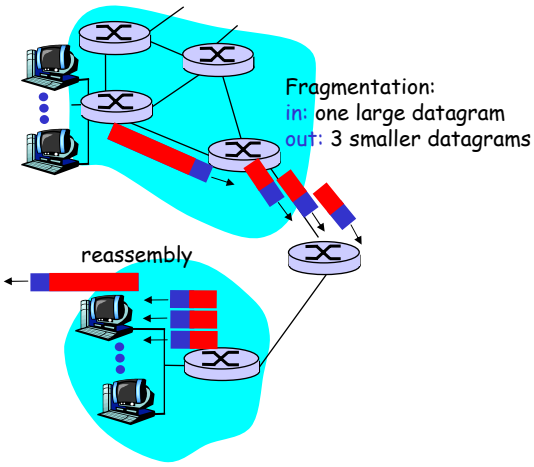


How much overhead with TCP?

- 20 bytes of TCP
- 20 bytes of IP
- = 40 bytes + app layer overhead

IP Fragmentation & Reassembly

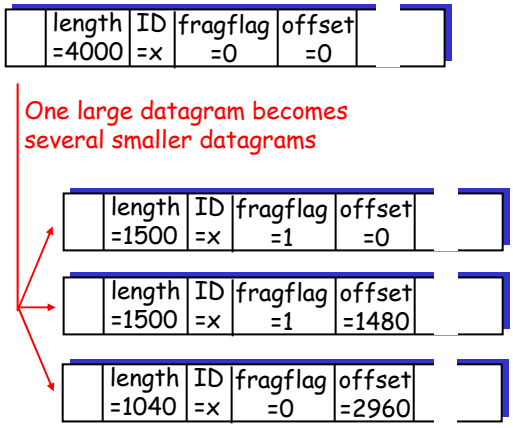
- Network links have MTU (max.transfer size) - largest possible link-level frame.
  - different link types, different MTUs
- Large IP datagram divided ("fragmented") within net
  - One datagram becomes several datagrams
  - "Reassembled" only at final destination
  - IP header bits used to identify, order related fragments



IP Fragmentation and Reassembly

Example

- 4000 byte datagram
- MTU = 1500 bytes



ICMP: Internet Control Message Protocol

- Used by hosts, routers, gateways to communication network-level information
  - Error reporting: unreachable host, network, port, protocol
  - Echo request/reply (used by ping)
- Network-layer "above" IP:
  - ICMP msgs carried in IP datagrams
- ICMP message: type, code plus first 8 bytes of IP datagram causing error

Type	Code	description
0	0	echo reply (ping)
3	0	dest. network unreachable
3	1	dest host unreachable
3	2	dest protocol unreachable
3	3	dest port unreachable
3	6	dest network unknown
3	7	dest host unknown
4	0	source quench (congestion control - not used)
8	0	echo request (ping)
9	0	route advertisement
10	0	router discovery
11	0	TTL expired
12	0	bad IP header

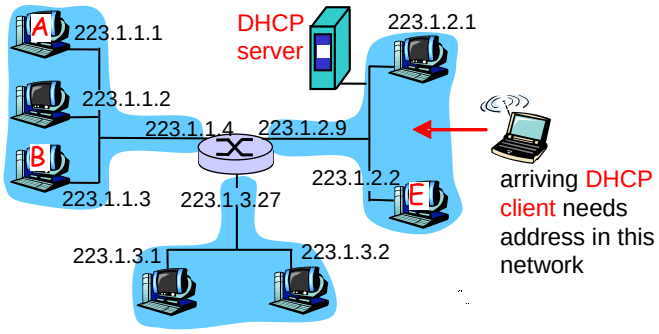
DHCP: Dynamic Host Configuration Protocol

Goal: Allow host to *dynamically* obtain its IP address from network server when it joins network

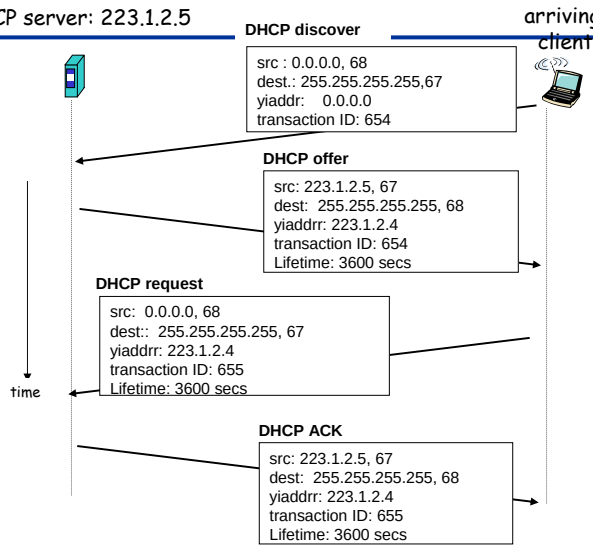
- Can renew its lease on address in use
- Allows reuse of addresses, only hold address while connected and "on"
- Support for mobile users who want to join network DHCP overview:

- Host broadcasts "DHCP discover" message
- DHCP server responds with "DHCP offer" message
- host requests IP address: "DHCP request" message
- DHCP server sends address: "DHCP ack" message

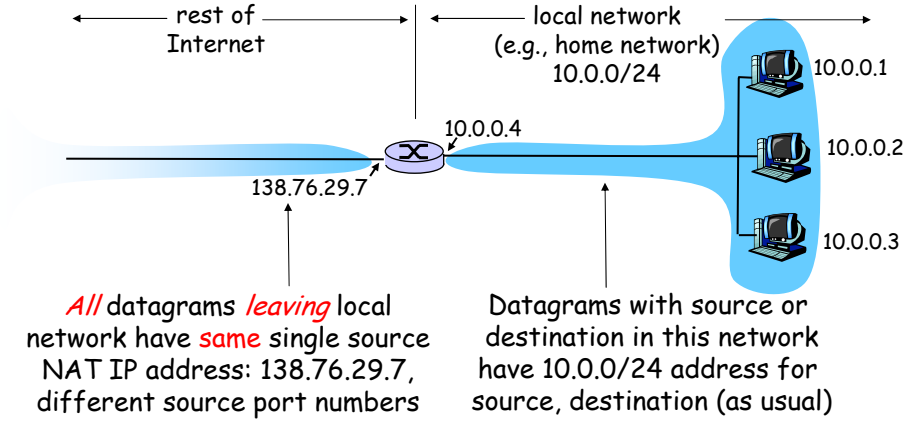
DHCP Client-Server Scenario



DHCP Client-Server Scenario



NAT: Network Address Translation



NAT: Network Address Translation

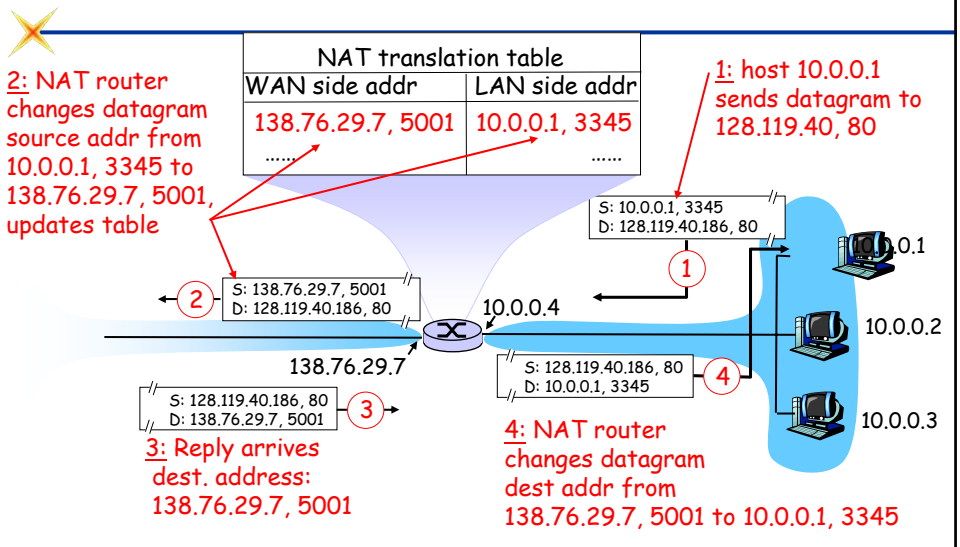
- Motivation: local network uses just one IP address as far as outside word is concerned:
  - No need to be allocated range of addresses from ISP: - just one IP address is used for all devices
  - Can change addresses of devices in local network without notifying outside world
  - Can change ISP without changing addresses of devices in local network
  - Devices inside local net not explicitly addressable, visible by outside world (a security plus).

NAT: Network Address Translation

Implementation: NAT router must:

- *Outgoing datagrams: replace* (source IP address, port #) of every outgoing datagram to (NAT IP address, new port #)
  - . . . remote clients/servers will respond using (NAT IP address, new port #) as destination address
- *Remember (in NAT translation table)* every (source IP address, port #) to (NAT IP address, new port #) translation pair
- *Incoming datagrams: replace* (NAT IP address, new port #) in dest fields of every incoming datagram with corresponding (source IP address, port #) stored in NAT table

NAT: Network Address Translation



NAT: Network Address Translation

- 16-bit port-number field:
  - 60,000 simultaneous connections with a single LAN-side address!
- NAT is controversial:
  - Routers should only process up to layer 3
  - Violates end-to-end argument
    - NAT possibility must be taken into account by application designers, eg, P2P applications
  - Address shortage should instead be solved by IPv6

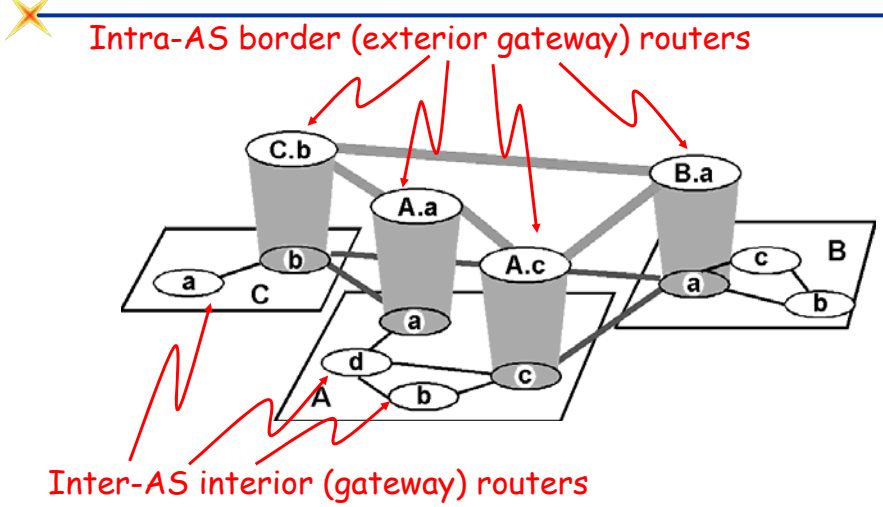
Routing in the Internet

1. Introduction and Network Service Models
2. Routing Principles
3. Hierarchical Routing
4. The Internet (IP) Protocol
5. Routing in the Internet
   - 4.5.1 Intra-AS routing: RIP and OSPF
   - 4.5.2 Inter-AS routing: BGP
6. What's Inside a Router?
7. IPv6
8. Multicast Routing

Routing in the Internet

- The Global Internet consists of Autonomous Systems (AS) interconnected with each other:
  - **Stub AS**: small corporation: one connection to other AS's
  - **Multihomed AS**: large corporation (no transit): multiple connections to other AS's
  - **Transit AS**: provider, hooking many AS's together
- Two-level routing:
  - **Intra-AS:** administrator responsible for choice of routing algorithm within network
  - **Inter-AS:** unique standard for inter-AS routing: BGP

Internet AS Hierarchy



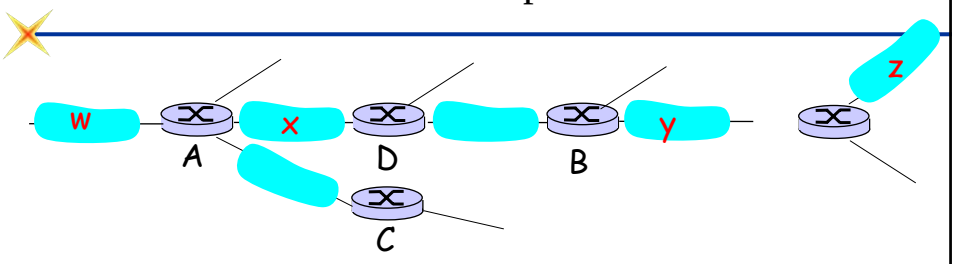
Intra-AS Routing

- Also known as Interior Gateway Protocols (IGP)
- Most common Intra-AS routing protocols:
  - RIP: Routing Information Protocol
  - OSPF: Open Shortest Path First
  - IGRP: Interior Gateway Routing Protocol (Cisco proprietary)

RIP (Routing Information Protocol)

- Distance vector algorithm
- Included in BSD-UNIX Distribution in 1982
- Distance metric: # of hops (max = 15 hops)
  - *Can you guess why?*
- Distance vectors: exchanged among neighbors every 30 sec via Response Message (also called **advertisement**)
- Each advertisement: list of up to 25 destination nets within AS

RIP: Example



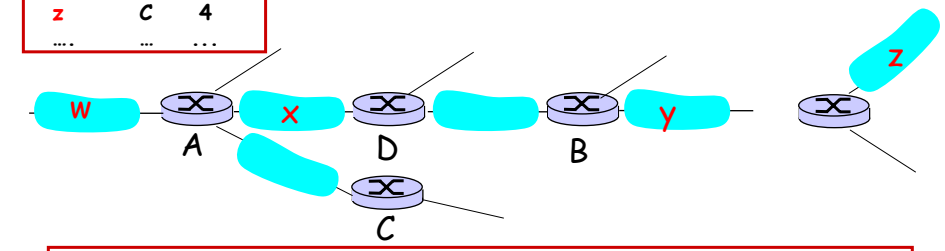
Destination Network	Next Router	Num. of hops to dest.
w	A	2
y	B	2
z	B	7
x	--	1
….	….	….

Routing table in D

RIP: Example

Dest	Next	hops
w	-	-
x	-	-
z	C	4
….	…	…

Advertisement from A to D



Destination Network	Next Router	Num. of hops to dest.
w	A	2
y	B	2
z	~~B~~ A	~~7~~ 5
x	--	1
….	….	….

Routing table in D

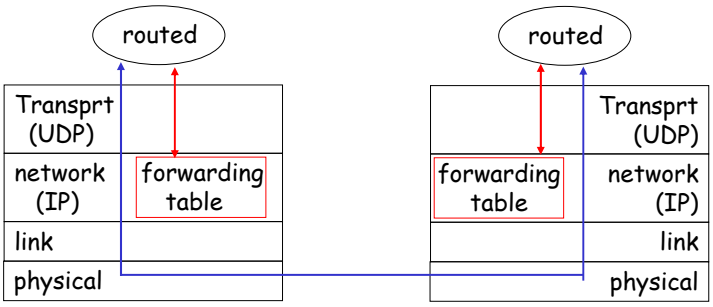
RIP: Link Failure and Recovery

If no advertisement heard after 180 sec --> neighbor/link declared dead

- Routes via neighbor invalidated
- New advertisements sent to neighbors
- Neighbors in turn send out new advertisements (if tables changed)
- Link failure info quickly propagates to entire net
- Poison reverse used to prevent ping-pong loops (infinite distance = 16 hops)

RIP Table processing

- RIP routing tables managed by **application-level** process called route-d (daemon)
- Advertisements sent in UDP packets, periodically repeated



RIP Table example (continued)

Router: *giroflee.eurocom.fr*

Destination	Gateway	Flags	Ref	Use	Interface
127.0.0.1	127.0.0.1	UH	0	26492	lo0
192.168.2.	192.168.2.5	U	2	13	fa0
193.55.114.	193.55.114.6	U	3	58503	le0
192.168.3.	192.168.3.5	U	2	25	qaa0
224.0.0.0	193.55.114.6	U	3	0	le0
default	193.55.114.129	UG	0	143454	

- Three attached class C networks (LANs)
- Router only knows routes to attached LANs
- Default router used to “go up”
- Route multicast address: 224.0.0.0
- Loopback interface (for debugging)

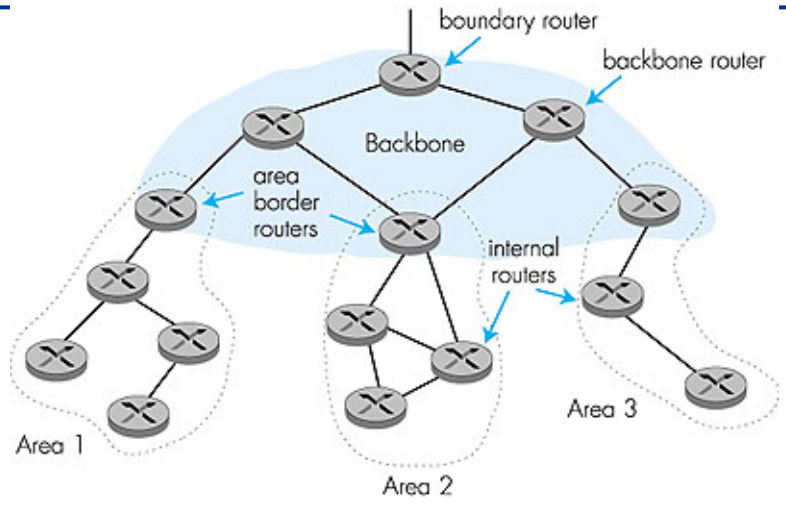
OSPF (Open Shortest Path First)

- “open”: publicly available
- Uses Link State algorithm
  - LS packet dissemination
  - Topology map at each node
  - Route computation using Dijkstra’s algorithm
- OSPF advertisement carries one entry per neighbor router
- Advertisements disseminated to entire AS (via flooding)
  - Carried in OSPF messages directly over IP (rather than TCP or UDP

OSPF “advanced” features (not in RIP)

- Security: all OSPF messages authenticated (to prevent malicious intrusion)
- Multiple same-cost paths allowed (only one path in RIP)
- For each link, multiple cost metrics for different TOS (e.g., satellite link cost set “low” for best effort; high for real time)
- Integrated uni- and multicast support:
  - Multicast OSPF (MOSPF) uses same topology data base as OSPF
- Hierarchical OSPF in large domains.

Hierarchical OSPF



Hierarchical OSPF

- Two-level hierarchy: local area, backbone.
  - Link-state advertisements only in area
  - each nodes has detailed area topology; only know direction (shortest path) to nets in other areas.
- **Area border routers:** "summarize" distances to nets in own area, advertise to other Area Border routers.
- **Backbone routers:** run OSPF routing limited to backbone.
- **Boundary routers:** connect to other AS's.

Inter-AS routing in the Internet: BGP

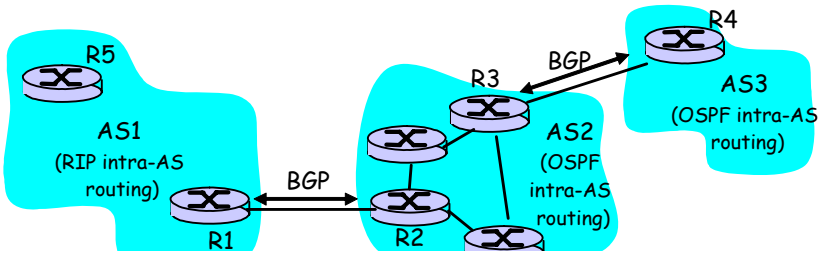


Figure 4.5.2-new2: BGP use for inter-domain routing

Internet inter-AS routing: BGP

- BGP (Border Gateway Protocol): *the* de facto standard
- **Path Vector** protocol:
  - similar to Distance Vector protocol
  - each Border Gateway broadcast to neighbors (peers) *entire path* (i.e., sequence of AS's) to destination
  - BGP routes to networks (ASs), not individual hosts
  - E.g., Gateway X may send its path to dest. Z:

Path (X,Z) = X,Y1,Y2,Y3,…,Z

Internet inter-AS routing: BGP

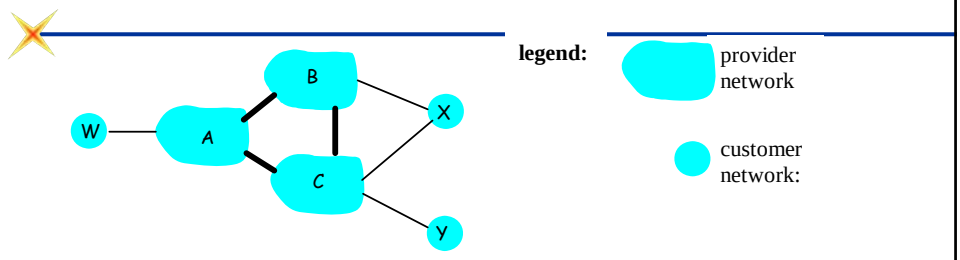
Suppose: gateway X send its path to peer gateway W

- W may or may not select path offered by X
  - cost, policy (don't route via competitors AS), loop prevention reasons.
- If W selects path advertised by X, then:

Path (W,Z) = w, Path (X,Z)

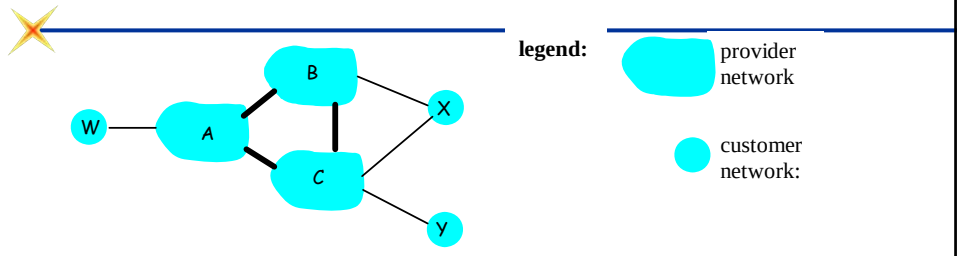
- Note: X can control incoming traffic by controlling it route advertisements to peers:
  - e.g., don't want to route traffic to Z -> don't advertise any routes to Z

BGP: controlling who routes to you



- A,B,C are provider networks
- X,W,Y are customer (of provider networks)
- X is dual-homed: attached to two networks
  - X does not want to route from B via X to C
  - .. so X will not advertise to B a route to C

BGP: controlling who routes to you



- A advertises to B the path AW
- B advertises to W the path BAW
- Should B advertise to C the path BAW?
  - No way! B gets no "revenue" for routing CBAW since neither W nor C are B's customers
  - B wants to force C to route to w via A
  - B wants to route *only* to/from its customers!

BGP operation

Q: What does a BGP router do?

- Receiving and filtering route advertisements from directly attached neighbor(s).
- Route selection.
  - To route to destination X, which path)of several advertised) will be taken?
- Sending route advertisements to neighbors.

BGP messages

- BGP messages exchanged using TCP.
- BGP messages:
  - OPEN: opens TCP connection to peer and authenticates sender
  - UPDATE: advertises new path (or withdraws old)
  - KEEPALIVE keeps connection alive in absence of UPDATES; also ACKs OPEN request
  - NOTIFICATION: reports errors in previous msg; also used to close connection

Why different Intra- and Inter-AS routing ?

Policy:

- Inter-AS: admin wants control over how its traffic routed, who routes through its net.
- Intra-AS: single admin, so no policy decisions needed

Scale:

- hierarchical routing saves table size, reduced update traffic

Performance:

- Intra-AS: can focus on performance
- Inter-AS: policy may dominate over performance

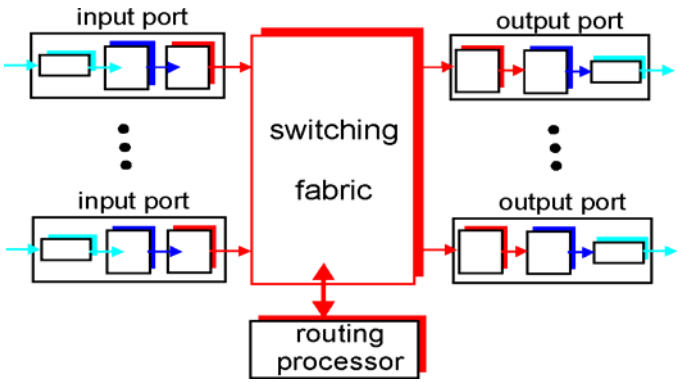
Router Design

1. Introduction and Network Service Models
2. Routing Principles
3. Hierarchical Routing
4. The Internet (IP) Protocol
5. Routing in the Internet
6. What's Inside a Router?
7. IPv6
8. Multicast Routing

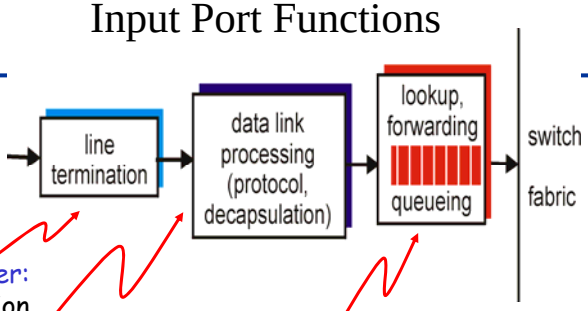
Router Architecture Overview

Two key router functions:

- Run routing algorithms/protocol (RIP, OSPF, BGP)
- *Switching* datagrams from incoming to outgoing link



Input Port Functions



Physical layer:
bit-level reception

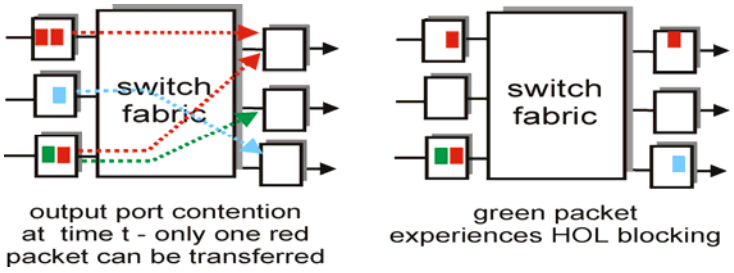
Data link layer:
e.g., Ethernet
see chapter 5

Decentralized switching:

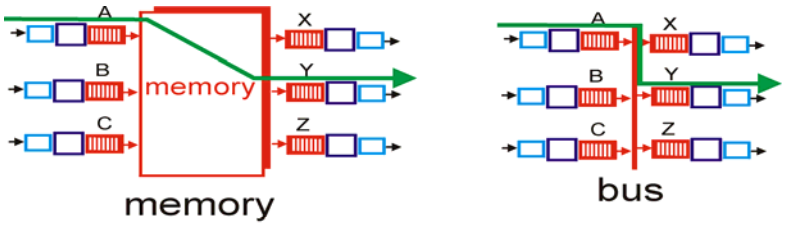
- Given datagram dest., lookup output port using routing table in input port memory
- Goal: complete input port processing at 'line speed'
- Queuing: if datagrams arrive faster than forwarding rate into switch fabric

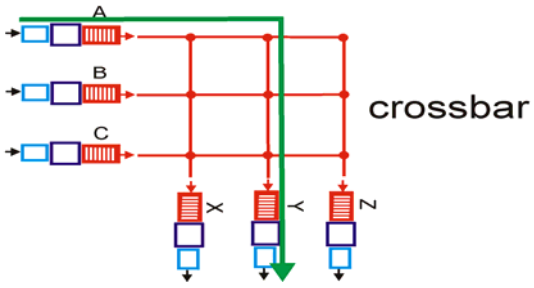
Input Port Queuing

- Fabric slower that input ports combined -> queueing may occur at input queues
- Head-of-the-Line (HOL) blocking: queued datagram at front of queue prevents others in queue from moving forward
- *queueing delay and loss due to input buffer overflow!*



Three types of switching fabrics

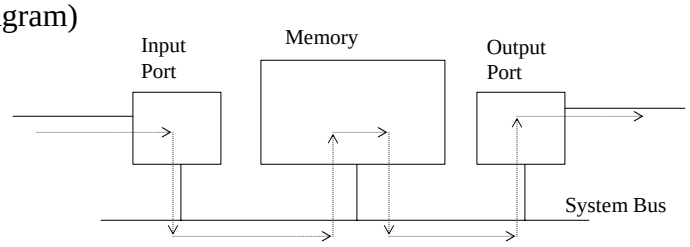




Switching Via Memory

First generation routers:

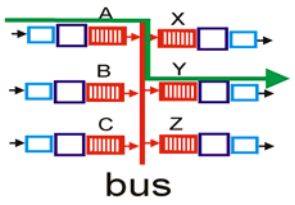
- Packet copied by system's (single) CPU
- Speed limited by memory bandwidth (2 bus crossings per datagram)



Modern routers:

- Input port processor performs lookup, copy into memory
- Cisco Catalyst 8500

Switching Via a Bus

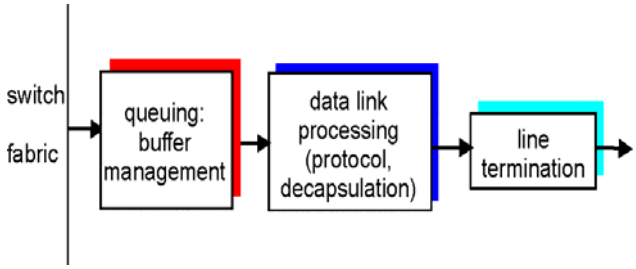


- Datagram from input port memory to output port memory via a shared bus
- Bus contention: switching speed limited by bus bandwidth
- 1 Gbps bus, Cisco 1900: sufficient speed for access and enterprise routers (not regional or backbone)

Switching Via An Interconnection Network

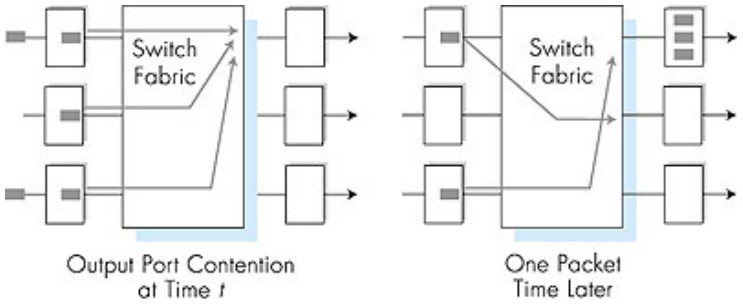
- Overcome bus bandwidth limitations
- Banyan networks, other interconnection nets initially developed to connect processors in multiprocessor
- Advanced design: fragmenting datagram into fixed length cells, switch cells through the fabric.
- Cisco 12000: switches Gbps through the interconnection network

Output Ports



- *Buffering* required when datagrams arrive from fabric faster than the transmission rate
- *Scheduling discipline* chooses among queued datagrams for transmission

Output port queueing



- buffering when arrival rate via switch exceeds output line speed
- *queueing (delay) and loss due to output port buffer overflow!*

Routing Design Issues and Protocols

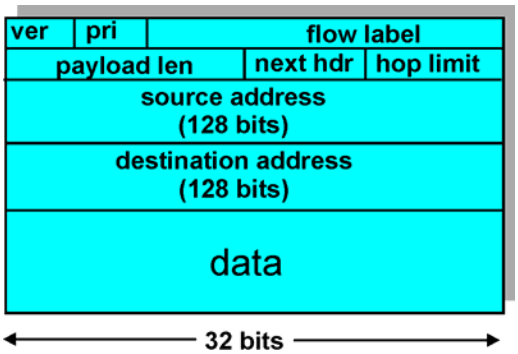
1. Introduction and Network Service Models
2. Routing Principles
3. Hierarchical Routing
4. The Internet (IP) Protocol
5. Routing in the Internet
6. What's Inside a Router?
7. IPv6
8. Multicast Routing

IPv6

- Initial motivation: 32-bit address space completely allocated by 2008.
- Additional motivation:
  - Header format helps speed processing/forwarding
  - Header changes to facilitate QoS
  - New "anycast" address: route to "best" of several replicated servers
- IPv6 datagram format:
  - Fixed-length 40 byte header
  - No fragmentation allowed

IPv6 Header (Cont)

Priority: identify priority among datagrams in flow
Flow Label: identify datagrams in same "flow."
(concept of"flow" not well defined).
Next header: identify upper layer protocol for data



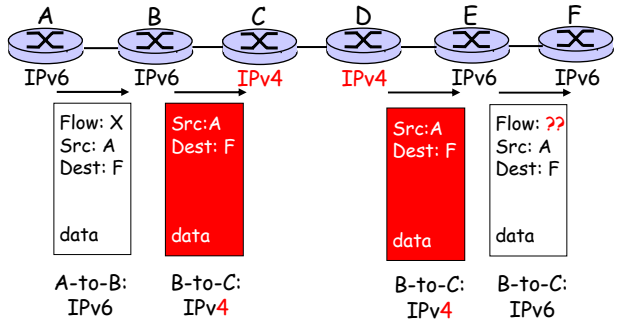
Other Changes from IPv4

- *Checksum*: removed entirely to reduce processing time at each hop
- *Options:* allowed, but outside of header, indicated by "Next Header" field
- *ICMPv6:* new version of ICMP
  - additional message types, e.g. "Packet Too Big"
  - multicast group management functions

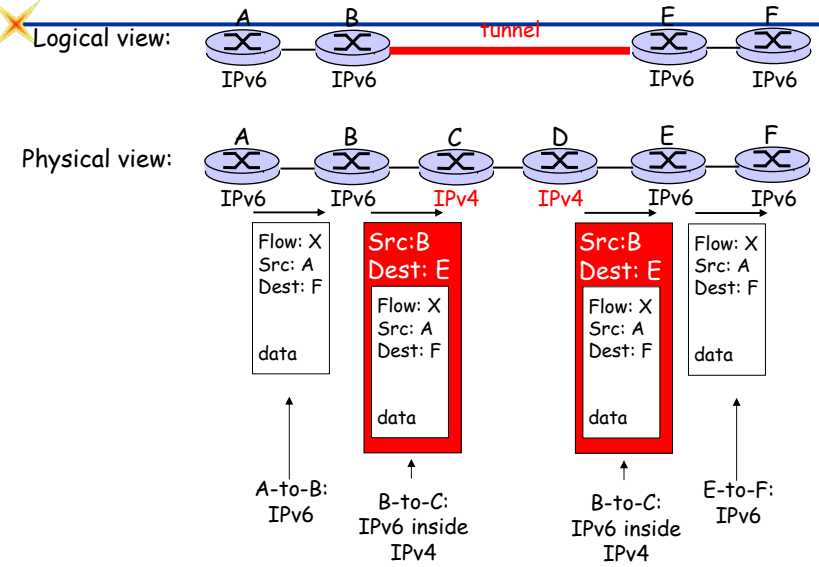
Transition From IPv4 To IPv6

- Not all routers can be upgraded simultaneous
  - no "flag days"
  - How will the network operate with mixed IPv4 and IPv6 routers?
- Two proposed approaches:
  - *Dual Stack*: some routers with dual stack (v6, v4) can "translate" between formats
  - *Tunneling:* IPv6 carried as payload in IPv4 datagram among IPv4 routers

Dual Stack Approach



Tunneling

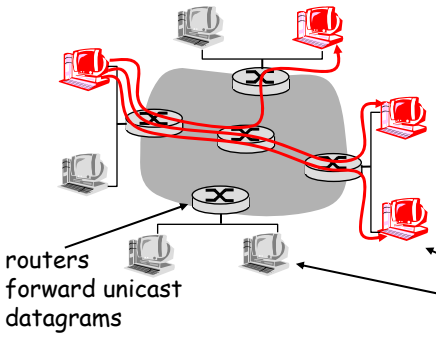


Multicasting

1. Introduction and Network Service Models
2. Routing Principles
3. Hierarchical Routing
4. The Internet (IP) Protocol
5. Routing in the Internet
6. What's Inside a Router?
7. IPv6
8. Multicast Routing

Multicast: One Sender to Many Receivers

- Multicast: act of sending datagram to multiple receivers with single "transmit" operation
  - analogy: one teacher to many students
- Question: how to achieve multicast

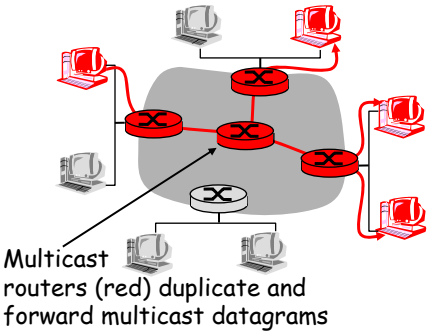


Multicast via unicast

- source sends N unicast datagrams, one addressed to each of N receivers

Multicast: One Sender to Many Receivers

- Multicast: act of sending datagram to multiple receivers with single "transmit" operation
  - analogy: one teacher to many students
- Question: how to achieve multicast

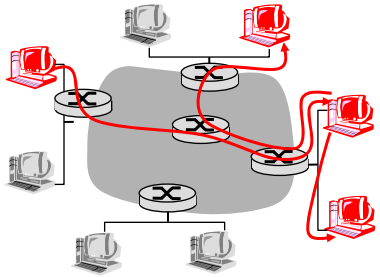


Network multicast

- Router actively participate in multicast, making copies of packets as needed and forwarding towards multicast receivers

Multicast: One Sender to Many Receivers

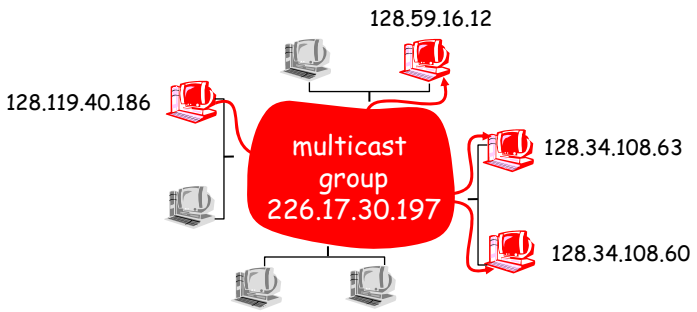
- Multicast: act of sending datagram to multiple receivers with single "transmit" operation
  - analogy: one teacher to many students
- Question: how to achieve multicast



Application-layer multicast

- End systems involved in multicast copy and forward unicast datagrams among themselves

Internet Multicast Service Model

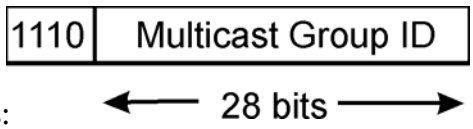


multicast group concept: use of indirection

- hosts addresses IP datagram to multicast group
- routers forward multicast datagrams to hosts that have "joined" that multicast group

Multicast Groups

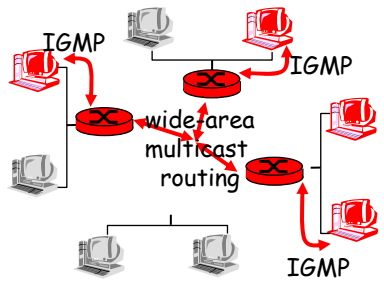
- class D Internet addresses reserved for multicast:



- host group semantics:
  - anyone can "join" (receive) multicast group
  - anyone can send to multicast group
  - no network-layer identification to hosts of members
- *needed:* infrastructure to deliver mcast-addressed datagrams to all hosts that have joined that multicast group

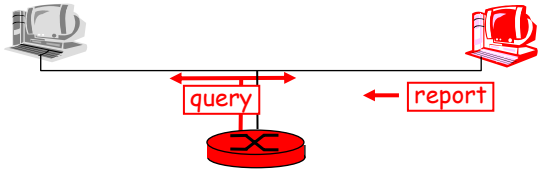
Joining a Multicast Group: Two-Step Process

- *Local:* host informs local mcast router of desire to join group: IGMP (Internet Group Management Protocol)
- *Wide area:* local router interacts with other routers to receive mcast datagram flow
  - Many protocols (e.g., DVMRP, MOSPF, PIM)



IGMP: Internet Group Management Protocol

- *Host:* sends IGMP report when application joins mcast group
  - IP_ADD_MEMBERSHIP socket option
  - Host need not explicitly "unjoin" group when leaving
- *Router:* sends IGMP query at regular intervals
  - Host belonging to a mcast group must reply to query



IGMP

IGMP version 1

- *Router:* Host Membership Query msg broadcast on LAN to all hosts
- *Host:* Host Membership Report msg to indicate group membership
  - Randomized delay before responding
  - Implicit leave via no reply to Query
- RFC 1112

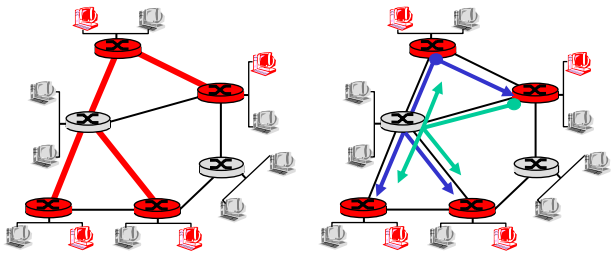
IGMP v2: additions include

- group-specific Query
- Leave Group msg
  - Last host replying to Query can send explicit Leave Group msg
  - Router performs group-specific query to see if any hosts left in group
  - RFC 2236

IGMP v3: nearing development as Internet draft

Multicast Routing: Problem Statement

- ***Goal:*** find a tree (or trees) connecting routers having local mcast group members
  - *Tree:* not all paths between routers used
  - *Source-based:* different tree from each sender to rcvrs
  - *Shared-tree:* same tree used by all group members



Shared tree Source-based trees

Approaches for building mcast trees

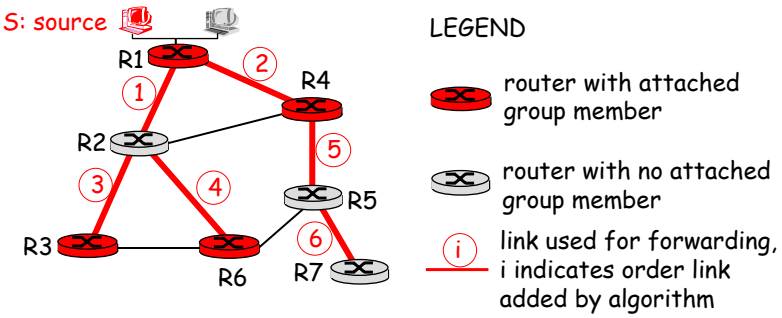
Approaches:

- Source-based tree: one tree per source
  - Shortest path trees
  - Reverse path forwarding
- Group-shared tree: group uses one tree
  - Minimal spanning (Steiner)
  - Center-based trees

…We first look at basic approaches, then specific protocols adopting these approaches

Shortest Path Tree

- Mcast forwarding tree: tree of shortest path routes from source to all receivers
  - Dijkstra's algorithm

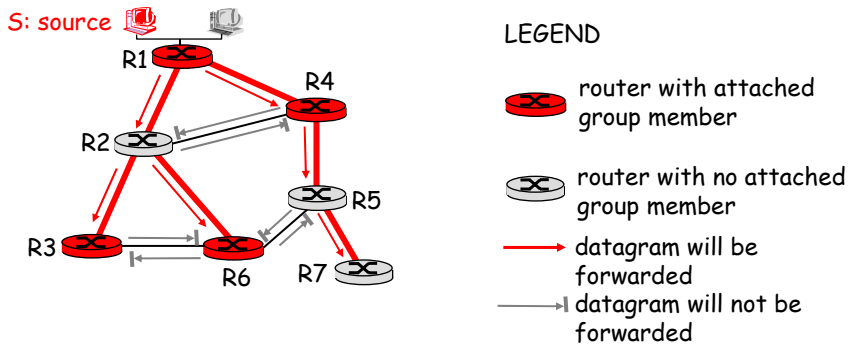


Reverse Path Forwarding

- Rely on router's knowledge of unicast shortest path from it to sender
- Each router has simple forwarding behavior:

if (mcast datagram received on incoming link on shortest path back to center)
then flood datagram onto all outgoing links
else ignore datagram

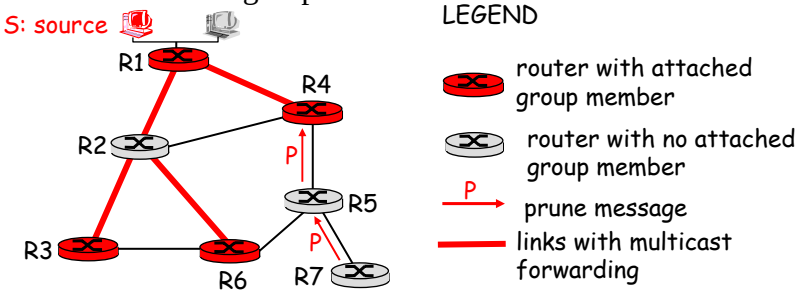
Reverse Path Forwarding: example



- Result is a source-specific *reverse* SPT
  - May be a bad choice with asymmetric links

Reverse Path Forwarding: pruning

- forwarding tree contains subtrees with no mcast group members
  - no need to forward datagrams down subtree
  - "prune" msgs sent upstream by router with no downstream group members



Shared-Tree: Steiner Tree

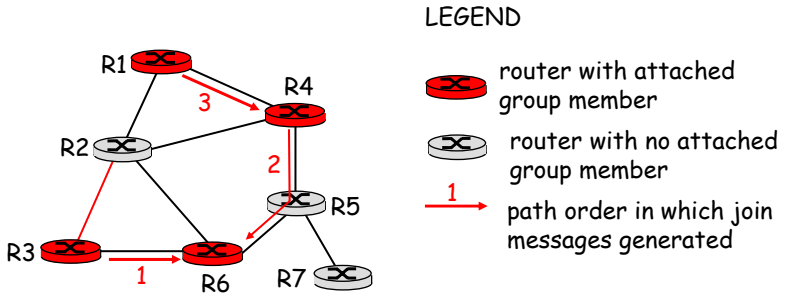
- Steiner Tree: minimum cost tree connecting all routers with attached group members
- Problem is NP-complete
- Excellent heuristics exists
- Not used in practice:
  - Computational complexity
  - Information about entire network needed
  - Monolithic: rerun whenever a router needs to join/leave

Center-based trees

- Single delivery tree shared by all
- One router identified as *"center"* of tree
- To join:
  - Edge router sends unicast *join-msg* addressed to center router
  - *Join-msg* "processed" by intermediate routers and forwarded towards center
  - *Join-msg* either hits existing tree branch for this center, or arrives at center
  - Path taken by *join-msg* becomes new branch of tree for this router

Center-based trees: an example

Suppose R6 chosen as center:



Internet Multicasting Routing: DVMRP

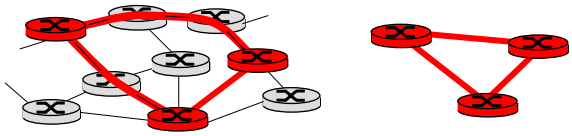
- DVMRP: distance vector multicast routing protocol, RFC1075
- *Flood and Prune:* reverse path forwarding, source-based tree
  - RPF tree based on DVMRP's own routing tables constructed by communicating DVMRP routers
  - No assumptions about underlying unicast
  - Initial datagram to mcast group flooded everywhere via RPF
  - Routers not wanting group: send upstream prune msgs

DVMRP: continued…

- *soft state:* DVMRP router periodically (1 min.) "forgets" branches are pruned:
  - mcast data again flows down unpruned branch
  - downstream router: reprune or else continue to receive data
- routers can quickly regraft to tree
  - following IGMP join at leaf
- odds and ends
  - commonly implemented in commercial routers
  - Mbone routing done using DVMRP

Tunneling

Q: How to connect "islands" of multicast routers in a "sea" of unicast routers?



physical topology

logical topology

- Mcast datagram encapsulated inside "normal" (non-multicast-addressed) datagram
- Normal IP datagram sent thru "tunnel" via regular IP unicast to receiving mcast router
- Receiving mcast router unencapsulates to get mcast datagram

PIM: Protocol Independent Multicast

- Not dependent on any specific underlying unicast routing algorithm (works with all)
- Two different multicast distribution scenarios :

Dense:

- Group members densely packed, in "close" proximity.
- Bandwidth more plentiful

Sparse:

- # networks with group members small wrt # interconnected networks
- Group members "widely dispersed"
- Bandwidth not plentiful

Consequences of Sparse-Dense Dichotomy:

Dense

- Group membership by routers *assumed* until routers explicitly prune
- *Data-driven* construction on mcast tree (e.g., RPF)
- Bandwidth and non-group-router processing *profligate*

Sparse:

- No membership until routers explicitly join
- *receiver- driven* construction of mcast tree (e.g., center-based)
- Bandwidth and non-group-router processing *conservative*

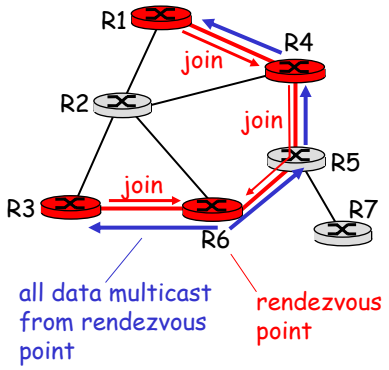
PIM- Dense Mode

flood-and-prune RPF, similar to DVMRP but

- underlying unicast protocol provides RPF info for incoming datagram
- less complicated (less efficient) downstream flood than DVMRP reduces reliance on underlying routing algorithm
- has protocol mechanism for router to detect it is a leaf-node router

PIM - Sparse Mode

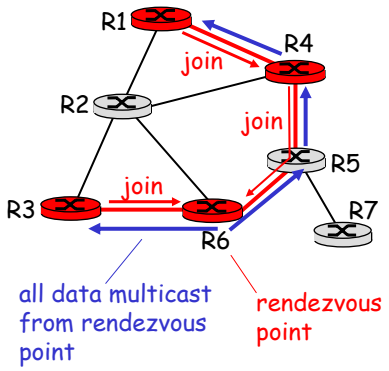
- Center-based approach
- Router sends *join* msg to rendezvous point (RP)
  - intermediate routers update state and forward *join*
- After joining via RP, router can switch to source-specific tree
  - Increased performance: less concentration, shorter paths



PIM - Sparse Mode

Sender(s):

- Unicast data to RP, which distributes down RP-rooted tree
- RP can extend mcast tree upstream to source
- RP can send *stop* msg if no attached receivers
  - "No one is listening!"



Network Layer: summary

What we've covered:

- Network layer services
- Routing principles: link state and distance vector
- Hierarchical routing
- IP
- Internet routing protocols RIP, OSPF, BGP
- What's inside a router?
- IPv6