

Routing

Dr. Arjan Duresi
Department of Computer Science
Louisiana State University



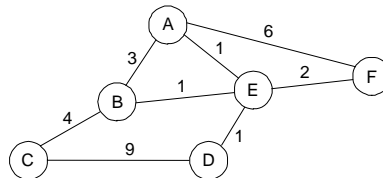
- ❑ Routing vs. Forwarding
- ❑ Routing Algorithms, Distance Vector, Link State
- ❑ Dijkstra's Algorithm
- ❑ ARPAnet Routing
- ❑ Subnetting
- ❑ Routing Protocols: RIP, OSPF, EGP, BGP-4

Why Routing

- ❑ How do the hosts determine the address(es) of the routers attached to their network?
- ❑ How does an host determine the address of other hosts attached to its network?
- ❑ How does an host select a specific router when sending an NPDU?
- ❑ How does an router determine the addresses of other routers that are attached to the same network?
- ❑ How does an router select a specific router to route an NPDU to a given destination host?

Overview

- ❑ Forwarding vs. Routing
 - ❑ forwarding: to select an output port based on destination address and routing table
 - ❑ routing: process by which routing table is built
- ❑ Network as a Graph



- ❑ Problem: Find lowest cost path between two nodes
- ❑ Factors
 - ❑ static: topology
 - ❑ dynamic: load

Routing

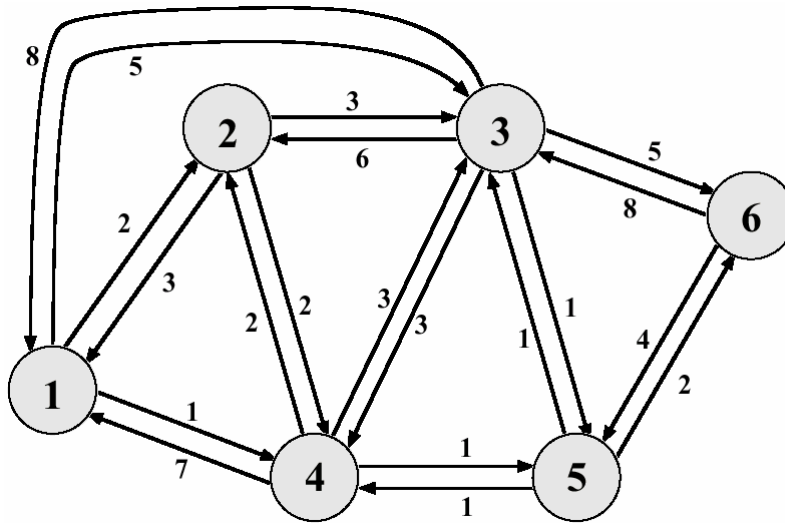
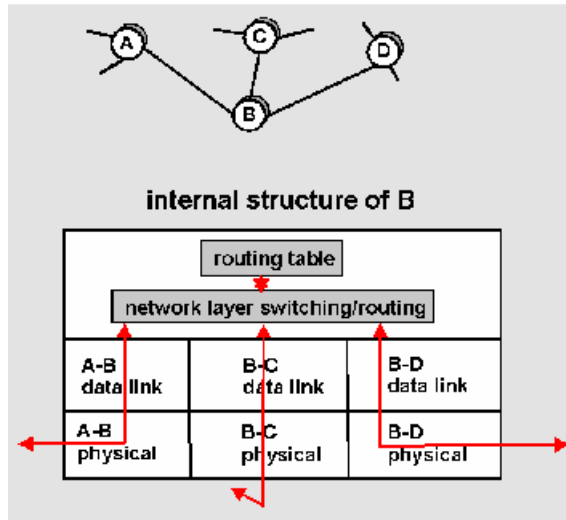


Fig 10.6

Routing vs. Forwarding



The routing function

- ❑ A network-layer packet contains:
 - ❑ Transport layer packet (port, seq, ack, data, checksum, etc)
 - ❑ Addressing info (e.g., source, destination address or VC identifier)
 - ❑ Other fields (e.g., version, length, time-to-live)
- ❑ Router/switch actions simple on packet receipt:
 - ❑ Look up packet identifier (dest. address or VC id) in routing table and forward on appropriate out-going link (or upwards if at destination)

Routing Table: issues

- ❑ How are routing tables determined/updated?
 - ❑ *who* determines table entries?
 - ❑ *what* info used in determining table entries?
 - ❑ *when* do routing table entries change?
 - ❑ *where* is routing info stored?
 - ❑ *how* to control table size?
 - ❑ *why* are routing tables determined a particular way.
What is the theoretical basis?

Routing Techniques Elements

- ❑ **Performance criterion:** *Hops*, Distance, *Speed*, Delay, Cost
- ❑ **Decision time:** *Packet*, session
- ❑ **Decision place:** *Distributed*, centralized, Source
- ❑ **Network information source:** None, local, *adjacent nodes*, nodes along route, all nodes
- ❑ **Routing strategy:** Fixed, *adaptive*, random, flooding
- ❑ **Adaptive routing update time:** Continuous, *periodic*, topology change, major load change

Routing Algorithms

- ❑ *Centralized versus Decentralized*
 - ❑ *Centralized:* central site computes and distributed routes (equivalently: information for computing routes known globally, each router makes same computation)
 - ❑ *Decentralized:* each router sees only local information (itself and physically-connected neighbors) and computes routes on this basis
 - ❑ pros and cons?

Routing Algorithms (cont.)

- ❑ *Static versus adaptive*
 - ❑ *static*: routing tables change very slowly, often in response to human intervention
 - ❑ *dynamic*: routing tables change as network traffic or topology change
 - ❑ pros and cons?
- ❑ *Two basic approaches adopted in practice:*
 - ❑ link-state routing: centralized, dynamic (periodically run)
 - ❑ distance vector: distributed, dynamic (in direct response to changes)

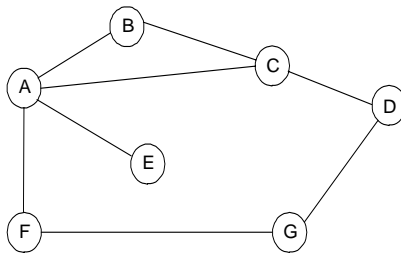
Distance Vector vs Link State

- ❑ **Distance Vector**: Each router sends a vector of distances to its neighbors. The vector contains distances to all nodes in the network. Older method. Count to infinity problem.
- ❑ **Link State**: Each router sends a vector of distances to all nodes. The vector contains only distances to neighbors. Newer method. Used currently in internet.

Distance Vector

- ❑ Each node maintains a set of triples
 - ❑ **(Destination, Cost, NextHop)**
- ❑ Exchange updates directly connected neighbors
 - ❑ periodically (on the order of several seconds)
 - ❑ whenever table changes (called *triggered* update)
- ❑ Each update is a list of pairs:
 - ❑ **(Destination, Cost)**
- ❑ Update local table if receive a “better” route
 - ❑ smaller cost
 - ❑ came from next-hop
- ❑ Refresh existing routes; delete if they time out

Example



Destination	Cost	NextHop
A	1	A
C	1	C
D	2	C
E	2	A
F	2	A
G	3	A

Routing Loops

- ❑ Example 1
 - ❑ F detects that link to G has failed
 - ❑ F sets distance to G to infinity and sends update to A
 - ❑ A sets distance to G to infinity since it uses F to reach G
 - ❑ A receives periodic update from C with 2-hop path to G
 - ❑ A sets distance to G to 3 and sends update to F
 - ❑ F decides it can reach G in 4 hops via A
- ❑ Example 2
 - ❑ link from A to E fails
 - ❑ A advertises distance of infinity to E
 - ❑ B and C advertise a distance of 2 to E
 - ❑ B decides it can reach E in 3 hops; advertises this to A
 - ❑ A decides it can reach E in 4 hops; advertises this to C
 - ❑ C decides that it can reach E in 5 hops...

Loop-Breaking Heuristics

- ❑ Set infinity to 16
- ❑ Split horizon
- ❑ Split horizon with poison reverse

Link State

- ❑ Strategy
 - ❑ send to all nodes (not just neighbors) information about directly connected links (not entire routing table)
- ❑ Link State Packet (LSP)
 - ❑ id of the node that created the LSP
 - ❑ cost of link to each directly connected neighbor
 - ❑ sequence number (SEQNO)
 - ❑ time-to-live (TTL) for this packet

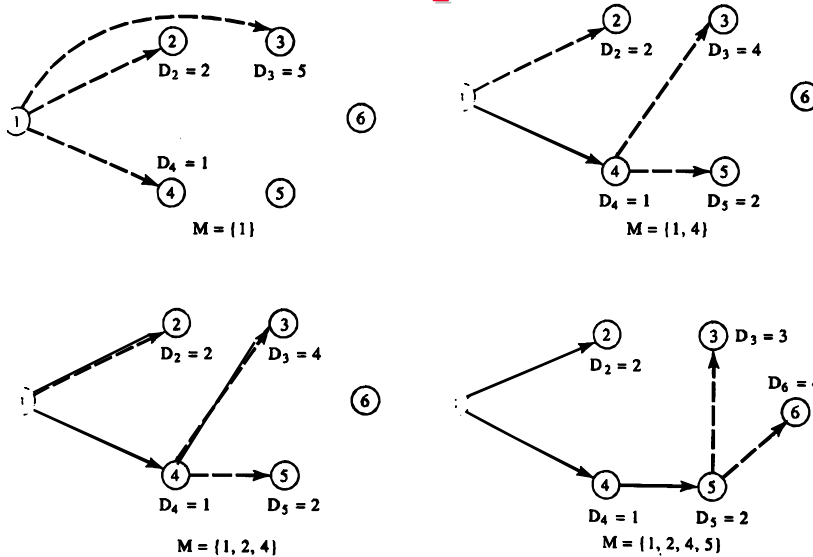
Link State (cont.)

- ❑ Reliable flooding
 - ❑ store most recent LSP from each node
 - ❑ forward LSP to all nodes but one that sent it
 - ❑ generate new LSP periodically
 - ❑ increment SEQNO
 - ❑ start SEQNO at 0 when reboot
 - ❑ decrement TTL of each stored LSP
 - ❑ discard when TTL=0

Dijkstra's Algorithm

- Goal: Find the least cost paths from a given node to all other nodes in the network
- Notation:
 - d_{ij} = Link cost from i to j if i and j are connected
 - D_n = Total path cost from s to n
 - M = Set of nodes so far for which the least cost path is known
- Method:
 - Initialize: $M = \{s\}$, $D_n = d_{sn}$
 - Find node $w \notin M$, whose D_n is minimum
 - Update D_n

Example



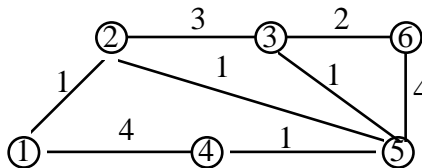
Example (Cont)

	M	D2	Path	D3	Path	D4	Path	D5	Path	D6	Path
1	{1}	2	1-2	5	1-3	1	1-4	∞	-	∞	-
2	{1,4}	2	1-2	4	1-4-3	1	1-4	2	1-4-5	∞	-
3	{1,2,4}	2	1-2	4	1-4-3	1	1-4	2	1-4-5	∞	-
4	{1,2,4,5}	2	1-2	3	1-4-5-3	1	1-4	2	1-4-5	4	1-4-5-6
5	{1,2,3,4,5}	2	1-2	3	1-4-5-3	1	1-4	2	1-4-5	4	1-4-5-6
6	{1,2,3,4,5,6}	2	1-2	3	1-4-5-3	1	1-4	2	1-4-5	4	1-4-5-6

Table 10.4a

Dijkstra's Routing Algorithm

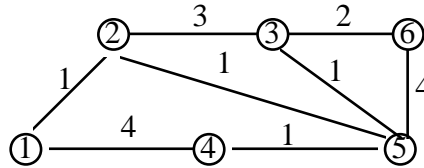
- Apply to the following network and compute paths from node 1.



	M	D2	Path	D3	Path	D4	Path	D5	Path	D6	Path
1											
2											
3											
4											
5											
6											

Dijkstra's routing algorithm

- Apply to the following network and compute paths from node 1.



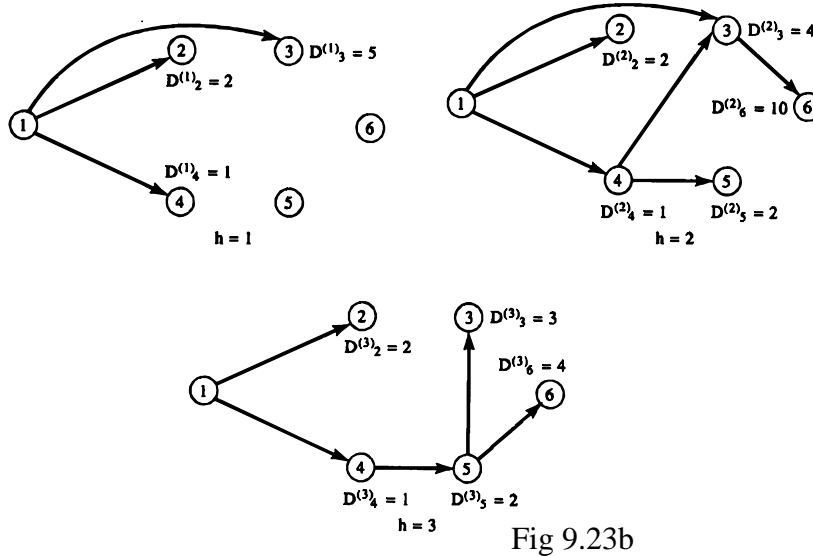
	M	D2	Path	D3	Path	D4	Path	D5	Path	D6	Path
1	{1}	1	1-2	∞	-	4	1-4	∞	-	∞	-
2	{1,2}	1	1-2	4	1-2-3	4	1-4				
3	{1,2,5}	1	1-2	4	1-2-3	4	1-4	2	1-2-5	6	1-2-3-6
4	{1,2,3,5}	1	1-2	3	1-2-5-3	3	1-2-5-4	2	1-2-5	6	1-2-3-6
5	{1,2,3,4,5}	1	1-2	3	1-2-5-3	3	1-2-5-4	2	1-2-5	5	1-2-5-3-6
6	{1,2,3,4,5,6}	1	1-2	3	1-2-5-3	3	1-2-5-4	2	1-2-5	5	1-2-5-3-6

Bellman-Ford Algorithm

- Notation:
 - h = Number of hops being considered
 - $D_n^{(h)}$ = Cost of h -hop path from s to n
- Method: Find all nodes 1 hop away
 Find all nodes 2 hops away
 Find all nodes 3 hops away
 - Initialize: $D_n^{(h)} = \infty$ for all $n \neq s$; $D_n^{(h)} = 0$ for all h
 - Find j th node for which $h+1$ hops cost is minimum

$$D_n^{(h+1)} = \min_j [D_j^{(h)} + D_{jn}]$$

Example



Example (Cont)

h	D(h ₂)	Path	D(h ₃)	Path	D(h ₄)	Path	D(h ₅)	Path	D(h ₆)	Path
0	∞	-	∞	-	∞	-	∞	-	∞	-
1	2	1-2	5	1-3	1	1-4	∞	-	∞	-
2	2	1-2	4	1-4-3	1	1-4	2	1-4-5	10	1-3-6
3	2	1-2	3	1-5-4-3	1	1-4	2	1-4-5	4	1-4-5-6
4	2	1-2	3	1-5-4-3	1	1-4	2	1-4-5	4	1-4-5-6

Table 10.4b

Flooding

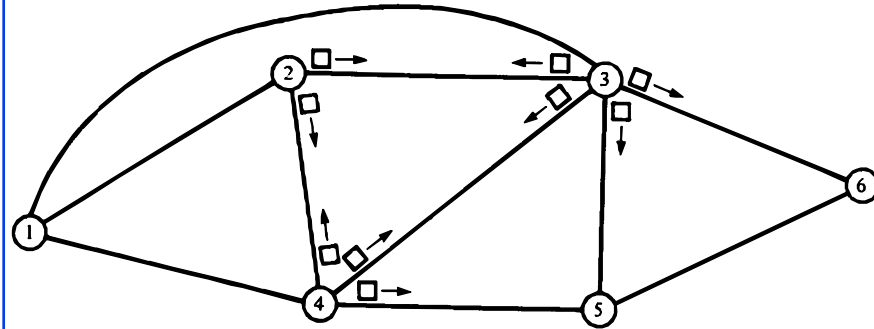


Fig 10.8

Flooding

- ❑ Uses all possible paths
- ❑ Uses minimum hop path Used for source routing

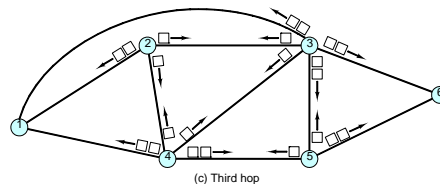
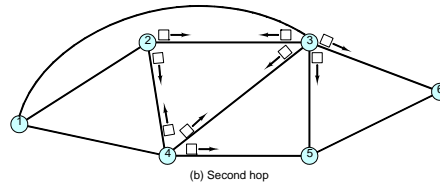
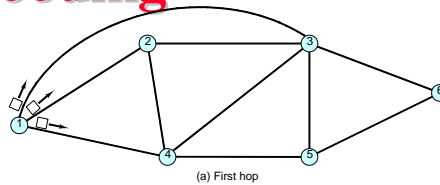


Fig 10.8

ARPAnet Routing (1969-78)

- Features: Metric = Queue length,
 - measures number of packets enqueued on each link
 - took neither latency or bandwidth into consideration
- Each node sends a vector of costs (to all nodes) to neighbors. Distance vector
- Each node computes new cost vectors based on the new info using Bellman-Ford algorithm

ARPAnet Routing Algorithm

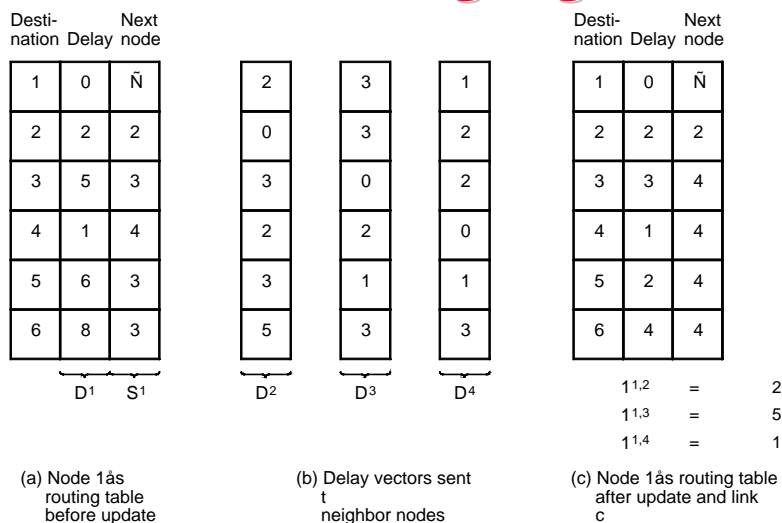


Fig 10.10

ARPAnet Routing (1979-86)

- ❑ Problem with earlier algorithm: Thrashing (packets went to areas of low queue length rather than the destination), Speed not considered
- ❑ Solution: Cost=Measured delay over 10 seconds
- ❑ Each node **floods** a vector of cost to neighbors.
Link-state. Converges faster after topology changes.
- ❑ Each node computes new cost vectors based on the new info using Dijkstra's algorithm

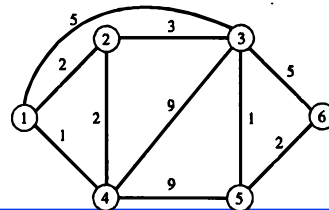


Fig 10.11

ARPAnet Routing (1987+)

- ❑ Problem with 2nd Method: Correlation between delays reported and those experienced later : High in light loads, low during heavy loads
 - ⇒ Oscillations under heavy loads
 - ⇒ Unused capacity at some links, over-utilization of others,
- More variance in delay more frequent updates More overhead

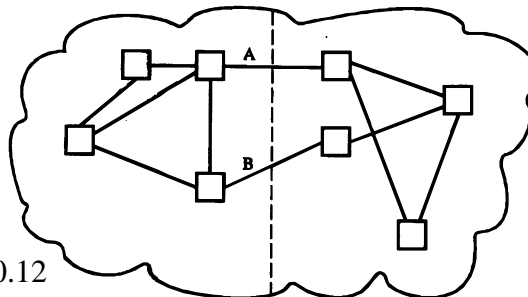


Fig 10.12

Routing Algorithm

- Delay is averaged over 10 s
- Link utilization = $r = 2(s-t)/(s-2t)$
where t =measured delay, s =service time per packet (600 bit times)
- Exponentially weighted average utilization
 $U(n+1) = \alpha U(n) + (1-\alpha)r(n+1)$
 $= 0.5 U(n) + 0.5 r(n+1)$
with $\alpha = 0.5$
- Link cost = $fn(U)$

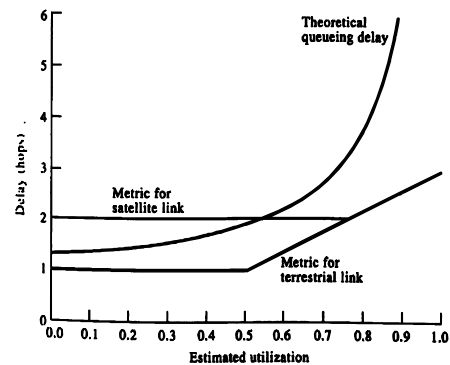


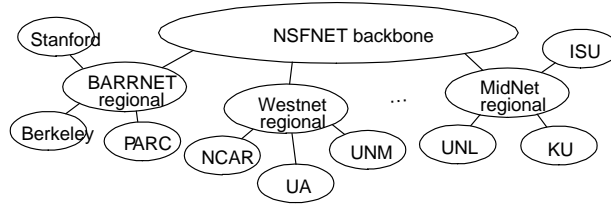
Fig 10.13

How to Make Routing Scale

- Flat versus Hierarchical Addresses
- Inefficient use of Hierarchical Address Space
 - class C with 2 hosts ($2/255 = 0.78\%$ efficient)
 - class B with 256 hosts ($256/65535 = 0.39\%$ efficient)
- Still Too Many Networks
 - routing tables do not scale
 - route propagation protocols do not scale

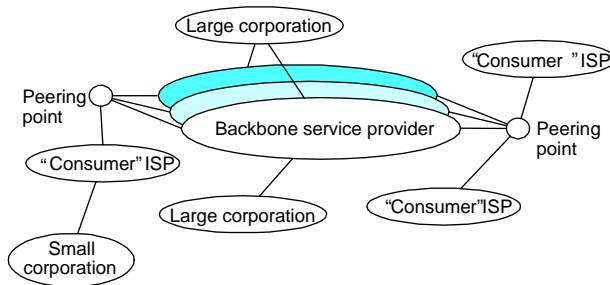
Internet Structure

Recent Past



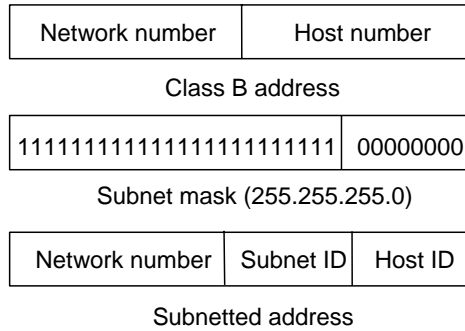
Internet Structure

Today

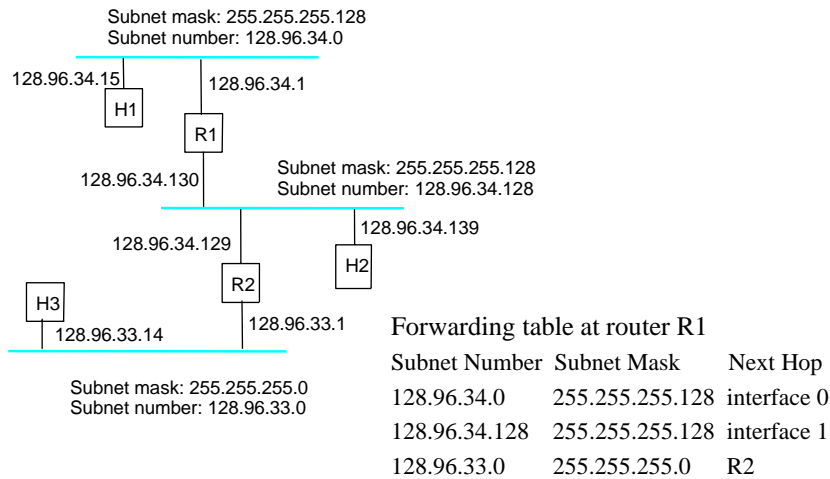


Subnetting

- ❑ Add another level to address/routing hierarchy: *subnet*
- ❑ *Subnet masks* define variable partition of host part
- ❑ Subnets visible only within site



Subnet Example



Forwarding Algorithm

```
D = destination IP address
for each entry (SubnetNum, SubnetMask, NextHop)
  D1 = SubnetMask & D
  if D1 = SubnetNum
    if NextHop is an interface
      deliver datagram directly to D
    else
      deliver datagram to NextHop
```

- ❑ Use a default router if nothing matches
- ❑ Not necessary for all 1s in subnet mask to be contiguous
- ❑ Can put multiple subnets on one physical network
- ❑ Subnets not visible from the rest of the Internet

Supernetting

- ❑ Assign block of contiguous network numbers to nearby networks
- ❑ Called CIDR: Classless Inter-Domain Routing
- ❑ Represent blocks with a single pair
(`first_network_address`, `count`)
- ❑ Restrict block sizes to powers of 2
- ❑ Use a bit mask (CIDR mask) to identify block size
- ❑ All routers must understand CIDR addressing

Route Propagation

- Know a smarter router
 - hosts know local router
 - local routers know site routers
 - site routers know core router
 - core routers know everything
- Autonomous System (AS)
 - corresponds to an administrative domain
 - examples: University, company, backbone network
 - assign each AS a 16-bit number
- Two-level route propagation hierarchy
 - interior gateway protocol (each AS selects its own)
 - exterior gateway protocol (Internet-wide standard)

Autonomous Systems

- An internet connected by homogeneous routers under the administrative control of a single entity

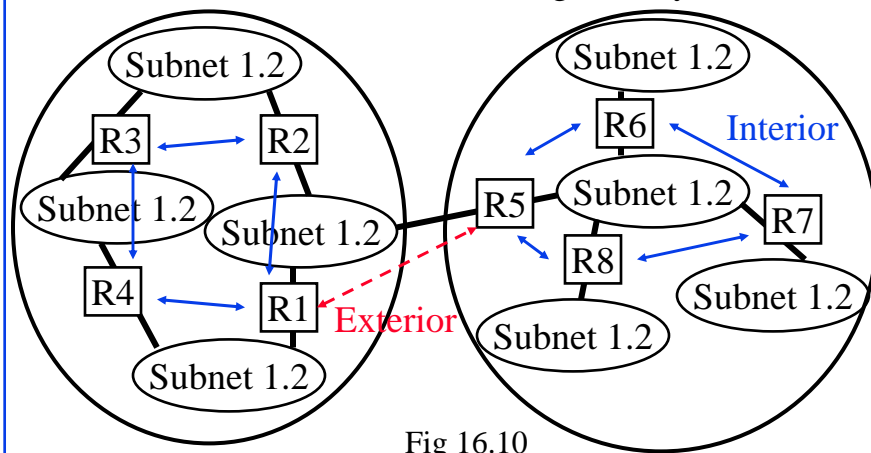


Fig 16.10

Routing Protocols

- ❑ There is a difference between a routing protocol and a routable protocol.
 - ❑ A routing protocol is one that is used to propagate route path information on a network
 - ❑ A routable protocol is one that has the ability to be routed as opposed to a non-routable protocol such as NetBIOS
- ❑ IP is a routable protocol, it needs a routing protocol to route between subnets.

Routing Protocols

- ❑ Interior Router Protocol (IRP): Used for passing routing information among routers internal to an autonomous system
- ❑ Exterior Router Protocol (ERP): Used for passing routing information among routers between autonomous systems
- ❑ Routing Information Protocol (RIP): First generation ARPAnet IRP protocol. Entire routing table sent to neighbors.
 - ⇒ Distance vector routing.

Popular Interior Gateway Protocols - RIP

- ❑ RIP: Route Information Protocol
 - ❑ First generation ARPAnet IRP protocol. Entire routing table sent to neighbors.
 - ❑ developed for XNS
 - ❑ distributed with Unix
 - ❑ distance-vector algorithm
 - ❑ based on hop-count

OSPF

- ❑ Open Shortest Path First (OSPF): Interior routing protocol.
 - ❑ Recent Internet standard
 - ❑ Uses link-state algorithm, Link costs flooded
 - ❑ Supports load balancing, provides least-cost path routes using a fully user configurable routing metric (any fn of delay, data rate, dollar cost, etc.)
 - ❑ Supports authentication

EGP: Exterior Gateway Protocol

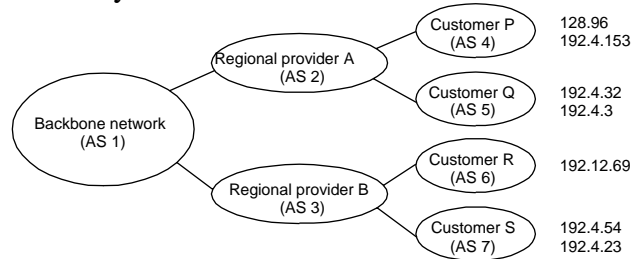
- ❑ Overview
 - ❑ designed for tree-structured Internet
 - ❑ concerned with *reachability*, not optimal routes
- ❑ Protocol messages
 - ❑ neighbor acquisition: one router requests that another be its peer; peers exchange reachability information
 - ❑ neighbor reachability: one router periodically tests if the another is still reachable; exchange HELLO/ACK messages; uses a k-out-of-n rule
 - ❑ routing updates: peers periodically exchange their routing tables (distance-vector)

BGP-4: Border Gateway Protocol

- ❑ AS Types
 - ❑ stub AS: has a single connection to one other AS
 - ❑ carries local traffic only
 - ❑ multihomed AS: connections to more than one AS
 - ❑ refuses to carry transit traffic
 - ❑ transit AS: has connections to more than one AS
 - ❑ carries both transit and local traffic
- ❑ Each AS has:
 - ❑ one or more border routers
 - ❑ one BGP *speaker* that advertises:
 - ❑ local networks
 - ❑ other reachable networks (transit AS only)
 - ❑ gives *path* information

BGP Example

- ❑ Speaker for AS2 advertises reachability to P and Q
 - ❑ network 128.96, 192.4.153, 192.4.32, and 192.4.3, can be reached directly from AS2



- ❑ Speaker for backbone advertises
 - ❑ networks 128.96, 192.4.153, 192.4.32, and 192.4.3 can be reached along the path (AS1, AS2).
- ❑ Speaker can cancel previously advertised paths

Summary



- ❑ Routing vs. Forwarding
- ❑ Routing Algorithms, Distance Vector, Link State
- ❑ Dijkstra's Algorithm
- ❑ ARPAnet Routing
- ❑ Subnetting
- ❑ Routing Protocols: RIP, OSPF, EGP, BGP-4