COMPUTER NETWORKING NOTES

5th Semester – CSE
2020
Module 3: Network Layer

Syllabus:

1. The Network layer: What's Inside a Router?
2. Input Processing, Switching, Output Processing
3. Where Does Queuing Occur?
4. Routing control plane
5. IPv6
6. A Brief foray into IP Security
7. Routing Algorithms: The Link-State (LS) Routing Algorithm, the Distance-Vector (DV) Routing Algorithm
8. Hierarchical Routing, Routing in the Internet
9. Intra-AS Routing in the Internet: RIP, OSPF, Inter/AS Routing: BGP
10. Broadcast Routing Algorithms and Multicast

Network layer in internet systems:

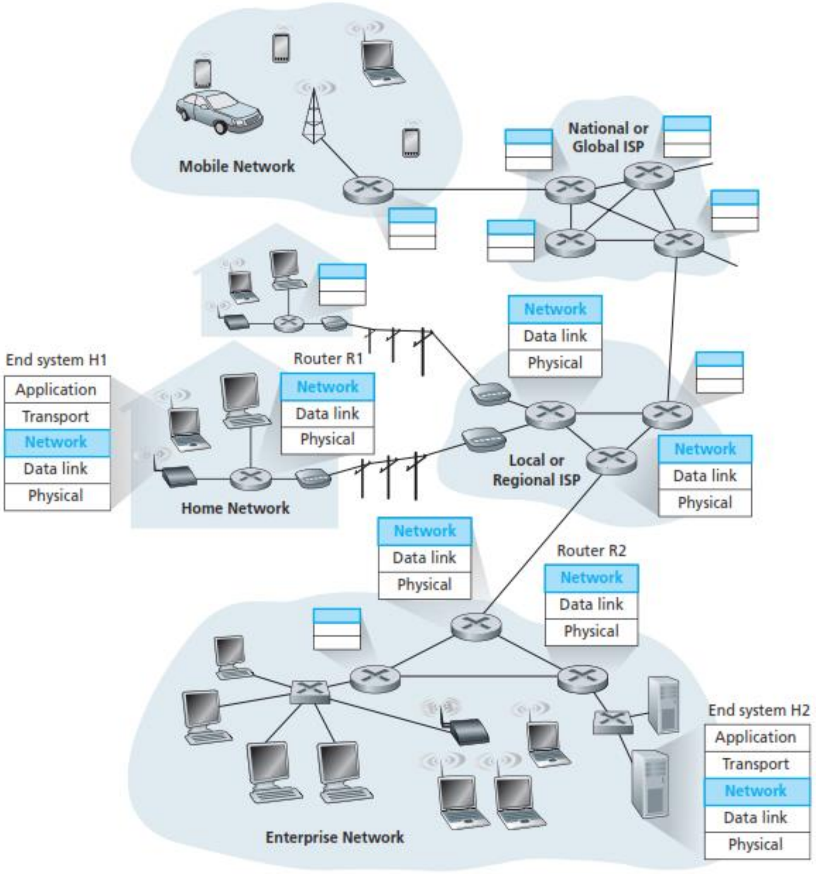


Figure 3.1: Network layer in internet systems

Network consist of number of hosts and routers. Every host has implemented with five layer TCP/IP stack protocol and router has implemented with bottom three layer of TCP/IP stack protocol. The network layer takes segments from transport layer and each segment is encapsulated into datagram or network layer packet. The packet is send to neighbors' router and network layer receives datagram from neighbors' router. The primary role of the routers is to forward datagrams from input links to output links.

Services and Functions of Network layer

a. Network layer services

The network service model defines the characteristics of end-to-end transport of packets between sending and receiving end systems.

- ***Guaranteed delivery***. This service guarantees that the packet will eventually arrive at its destination.
- ***Guaranteed delivery with bounded delay***. This service not only guarantees delivery of the packet, but delivery within a specified host-to-host delay bound
- **In-order packet delivery**. This service guarantees that packets arrive at the destination in the order that they were sent.
- **Guaranteed minimal bandwidth.** This network-layer service emulates the behavior of a transmission link of a specified bit rate between sending and receiving hosts. As long as the sending host transmits bits at a rate below the specified bit rate, then no packet is lost and each packet arrives within a pre-specified host-to-host delay.
- **Guaranteed maximum jitter.** This service guarantees that the amount of time between the transmissions of two successive packets at the sender is equal to the amount of time between their receipts at the destination.
- **Security services.** Using a secret session key known only by a source and destination host, the network layer in the source host could encrypt the payloads of all datagrams being sent to the destination host. The network layer in the destination host would then be responsible for decrypting the payloads.

b. Network layer functions

The role of the network layer is to move packets from a sending host to a receiving host with two main functions: 1. Routing or Path determination 2. Forwarding.

1. ***Routing or Path determination:*** The network layer must determine the route or path taken by packets as they flow from a sender to a receiver. The algorithms that calculate these paths are referred to as *routing algorithms.*
2. ***Forwarding:*** When a packet arrives at a router's input link, the router must move the packet to the appropriate output link.
3. ***Connection Setup:*** TCP has a three-way handshake is required before data can flow from sender to receiver for sender and receiver to set up the needed state information

Every router has a forwarding table. A router forwards a packet by examining the value of a field in the arriving packet's header, and then using this header value to index into the router's forwarding table. The value stored in the forwarding table entry for that header indicates the router's outgoing link interface to which that packet is to be forwarded. Depending on the network-layer protocol, the header value could be the destination IP address of the packet or an indication of the connection to which the packet belongs. In figure 3.2 routing algorithm determines the values that are inserted into the routers' forwarding tables. The routing algorithm may be centralized or decentralized.

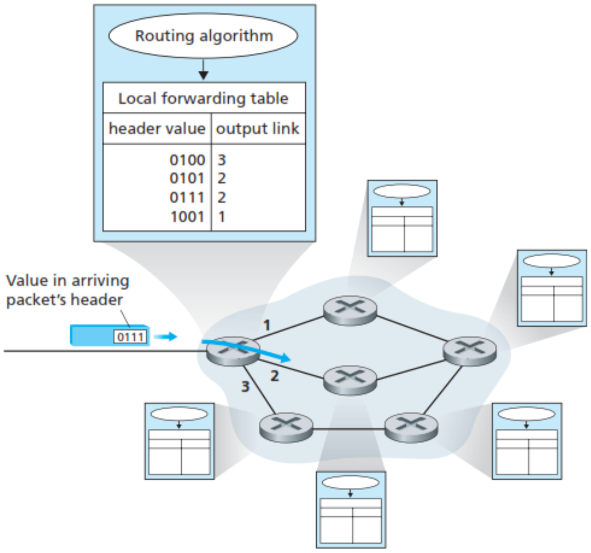


Figure 3.2: Routing algorithms forwarding tables

Virtual Circuit and Datagram Networks

Network layer can provide connectionless service or connection service between two hosts. Network-layer connection and connectionless services have some parallels with transport-layer connection-oriented and connectionless services. Computer networks that provide only a connection service at the network layer are called ***virtual-circuit (VC) networks***; computer networks that provide only a connectionless service at the network layer are called ***datagram networks***.

1. ***Virtual-Circuit (VC) networks***

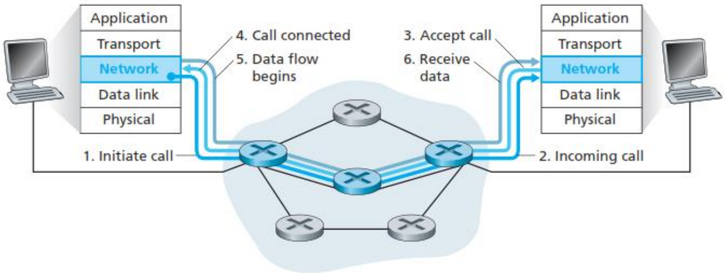


Figure 3.3: Virtual-Circuit network

Network-layer connections are virtual circuits (VCs) is consist of a path between the source and destination hosts, VC numbers, one number for each link along the path and entries in the forwarding table in each router along the path. A packet belonging to a virtual circuit will carry a VC number in its header.

In a VC network, the network's routers must maintain **connection state information** for the ongoing connections. Specifically, each time a new connection is established across a router, a new connection entry must be added to the router's forwarding table; and each time a connection is released, an entry must be removed from the table.

Three connection phases in a virtual circuit:

a. VC setup
b. Data transfer
c. VC teardown

a. **VC setup**: During this phase specifies the receiver's address, and waits for the network to set up the VC. The network layer determines the path between sender and receiver, VC number for each link along the path and adds an entry in the forwarding table in each router along the path.
b. **Data transfer:** once the VC has been established, packets can begin to flow along the VC
c. **VC teardown:** This is initiated when the sender (or receiver) informs the network layer of its desire to terminate the VC i.e forwarding tables in each of the packet routers on the path to indicate that the VC no longer exists.

2. *Datagram Networks*

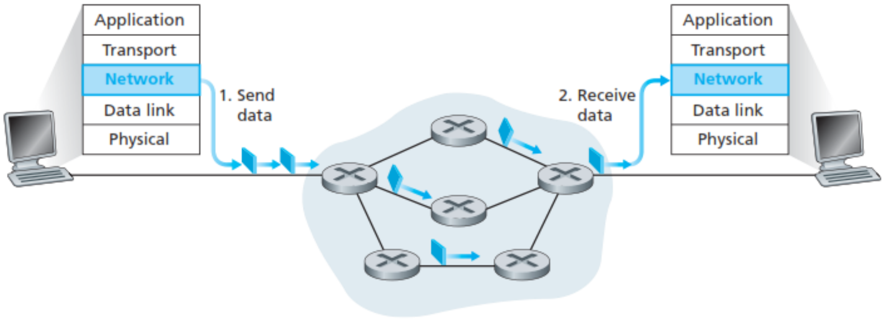


Figure 3.4: Datagram Networks

In a datagram network, each time an end system wants to send a packet, it stamps the packet with the address of the destination end system and then pops the packet into the network but here there are no VC.

As a packet is transmitted from source to destination, datagram networks maintain no connection state information but each router has a forwarding table that maps destination addresses to link interfaces; when a packet arrives at the router, the router uses the packet's destination address to look up the appropriate output link interface in the forwarding table.

Router Architecture/ inside Router

Forwarding function is transfer of packets from a router's incoming links to the appropriate outgoing links at that router. The router architecture consist of *Input ports, switching fabric, Output ports and routing processor*.

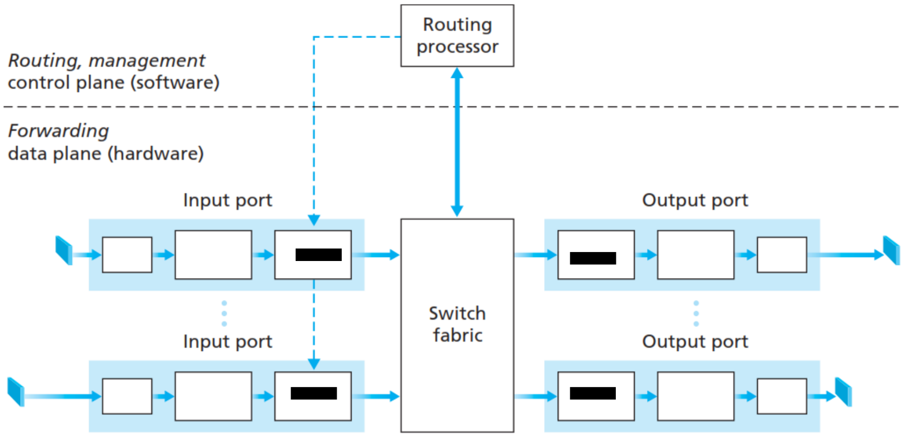


Figure 3. 5: Router Architecture

Input ports: It performs the physical layer function and link-layer functions for individual input link. It is here that the forwarding table is consulted to determine the router output port to which an arriving packet will be forwarded via the switching fabric. Control packets are forwarded from an input port to the routing processor.

Here that the router uses the forwarding table to look up the output port to which an arriving packet will be forwarded via the switching fabric. Forwarded via the switching fabric. The forwarding table is computed and updated by the routing processor, with a shadow copy typically stored at each input port. The forwarding table is copied from the routing processor to the line cards over a separate bus (e.g., a PCI bus) indicated by the dashed line from the routing processor to the input line cards, forwarding decisions can be made locally, at each input port, without invoking the centralized routing processor on a per-packet basis and thus avoiding a centralized processing bottleneck.

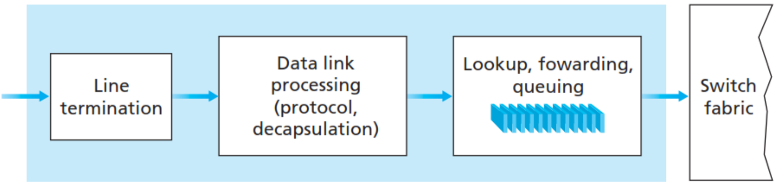

Line termination
Data link processing (protocol, decapsulation)
Lookup, fowarding, queuing
Switch fabric

Figure 3.6: Input port processing

Once a packet's output port has been determined via the lookup, the packet can be sent into the switching fabric. In some designs, a packet may be temporarily blocked from entering the switching fabric if packets from other input ports are currently using the fabric. A blocked packet will be queued at the input port and then scheduled to cross the fabric at a later point in time.

Switching fabric: The switching fabric connects the router's input ports to its output ports.

The packets are actually switched from an input port to an output port. Switching can be accomplished in a number of ways

1. Switching via memory
2. Switching via a bus

3. Switching via an interconnection network

1. Switching via memory

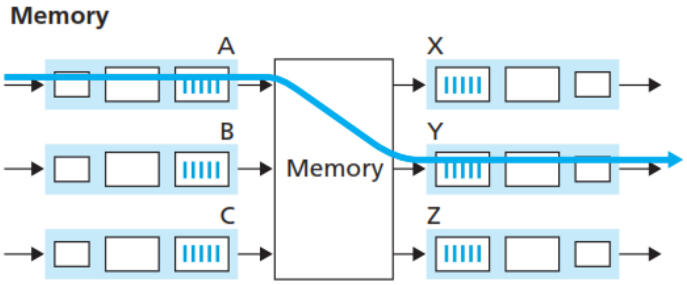


Figure 3.7: Switching via memory

- Switching between input and output ports being done under direct control of the routing processor.
- An input port with an arriving packet first signaled the routing processor via an interrupt. The packet was then copied from the input port into processor memory. The routing processor then extracted the destination address from the header, looked up the appropriate output port in the forwarding table, and copied the packet to the output port's buffers
- The overall forwarding throughput is less than *B*/2 because written into, or read from memory two packets cannot be forwarded at the same time, even if they have different destination ports.
- Many modern routers switch via memory routers that switch via memory look very much like shared-memory multiprocessors, with the processing on a line card switching (writing) packets into the memory of the appropriate output port.
- If multiple packets arrive to the router at the same time, each at a different input port, all but one must wait since only one packet can cross the bus at a time. Because every packet must cross the single bus, the switching speed of the router is limited to the bus speed.

2. Switching via a bus

The input port transfers a packet directly to the output port over a shared bus, without intervention by the routing processor.

- This is typically done by having the input port a switch-internal label (header) to the packet indicating the local output port to which this packet is being transferred and transmitting the packet onto the bus.

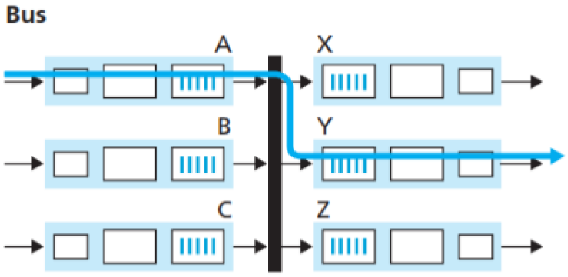


Figure 3.8: Switching via a bus

- The packet is received by all output ports, but only the port that matches the label will keep the packet. But only the port that matches the label will keep the packet. The label is then removed at the output port.
- If multiple packets arrive to the router at the same time, each at a different input port, all wait since only one packet can cross the bus at a time. Because every packet must cross the single bus, the switching speed of the router is limited to the bus speed.

3. Switching via an interconnection network

One way to overcome the bandwidth limitation of a single, shared bus use interconnection network

- A crossbar switch is an interconnection network consisting of 2N buses that connect N input ports to N output ports
- Each vertical bus intersects each horizontal bus at a cross point, which can be opened or closed at any time by the switch fabric controller
- Port A and needs to be forwarded to port Y and port B can be forwarded to port X at the same time, since the A-to-Y and B-to-X packets use different input and output busses.
- Crossbar networks are capable of forwarding multiple packets in parallel

Output ports: An output port stores packets received from the switching fabric and transmits these packets on the outgoing link by performing the necessary link-layer and physical-layer functions.

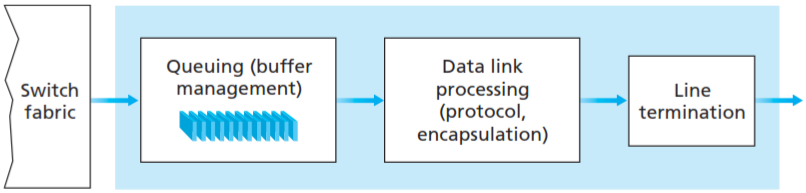

Switch fabric
Queuing (buffer management)
Data link processing (protocol, encapsulation)
Line termination

Figure 3. 9: Output port processing

Routing processor: The routing processor executes the routing protocols maintains routing tables and attached link state information, and computes the forwarding table for the router. It also performs the network management functions.

Where Does Queuing Occur?

The packet queues may form at both the input ports and the output ports will depend on the traffic load, the relative speed of the switching fabric, and the line speed.

- If queues grow large, the router’s memory can eventually be exhausted and **packet loss** will occur when no memory is available to store arriving packets.
- The amount of buffering needed is B = RTTX C/$\sqrt{N}$
  Where B= amount of buffering
  RTT = Round Trip Time
  C= Chanel Capacity

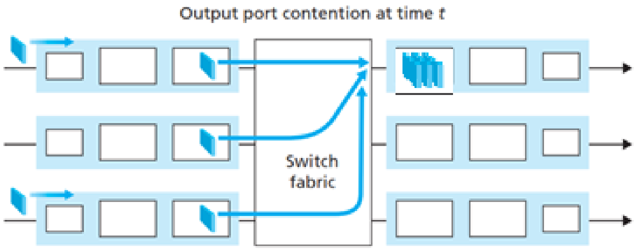
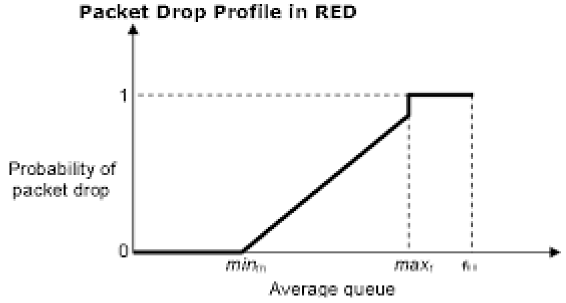


Figure 3.10: Output port queuing/Buffering

- A consequence of output port queuing is that a packet scheduler at the output port must choose one packet among those queued for transmission. This selection might be done on a simple basis, such as **first-come-first-served (FCFS)** scheduling or **weighted fair queuing (WFQ)**. Packet scheduling plays a crucial role in providing **quality-of-service** guarantees.
- If there is not enough memory to buffer an incoming packet, a decision must be made to either drop the arriving packet (**drop-tail**) or remove one or more already-queued packets to make room for the newly arrived packet.
- Before the buffer is full in order to provide a congestion signal to the sender through **active queue management** (AQM) algorithms is the **Random Early Detection (RED)** algorithm

Random Early Detection (RED)

- Under RED, a weighted average is maintained for the length of the output queue.
- If the average queue length is less than a minimum threshold (min_{th}), when a packet arrives, the packet is admitted to the queue.
- If the queue is full or the average queue length is greater than a maximum threshold(max_{th}), when a packet arrives, the packet is marked or dropped
- Finally, if the packet arrives to find an average queue length in the interval(min_{th} & max_{th})



Head-of-the-Line (HOL) blocking

Figure shows below an example in which two packets (darkly shaded) at the front of their input queues are destined for the same upper-right output port. Suppose that the switch fabric chooses to transfer the packet from the front of the upper-left queue. In this case, the darkly shaded packet in the lower-left queue must wait. But not only must this darkly shaded packet wait, so too must the lightly shaded packet that is queued behind that packet in the lower-left queue, even though there is no contention for the middle-right output port.

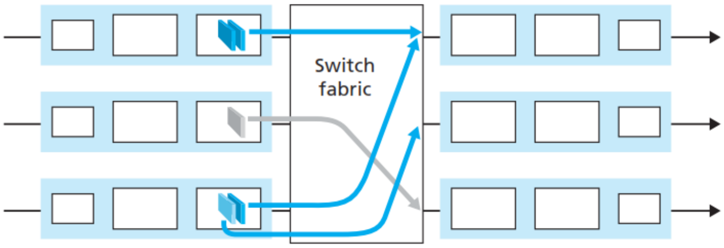


Figure 3.11: Head-of-the-Line (HOL) blocking

The Routing Control Plane

- The network-wide routing control plane is thus decentralized—with different pieces (e.g., of a routing algorithm) executing at different routers and interacting by sending control messages to each other.
- Router and switch vendors bundle their hardware data plane and software control plane together into closed (but inter-operable) platforms in a vertically integrated product.
- The control plane is implemented in the routers (e.g., local measurement/reporting of link state, forwarding table installation and maintenance) along with the data plane, and part of the control plane can be implemented externally to the router (e.g., in a centralized server, which could perform route calculation.

The Internet Protocol (IP): IPv6

Internet addressing and forwarding are important components of the Internet Protocol (IP).

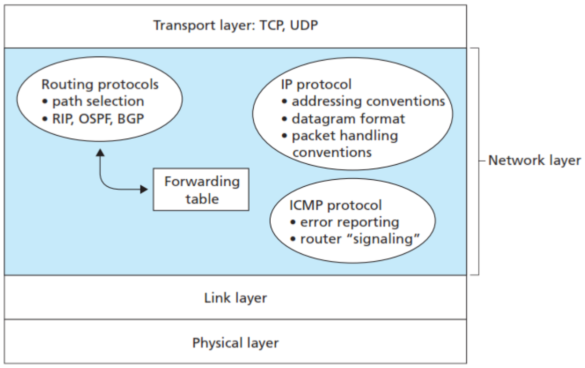


Figure 3.12: A look inside the Internet's network layer

Internet's network layer has three major components.

- The ***first*** component is the IP protocol
- The ***second*** major component is the routing component
- ***Final*** component of the network layer is a facility to report errors in datagrams and respond to requests for certain Network-layer information.

IPv6 Datagram Format

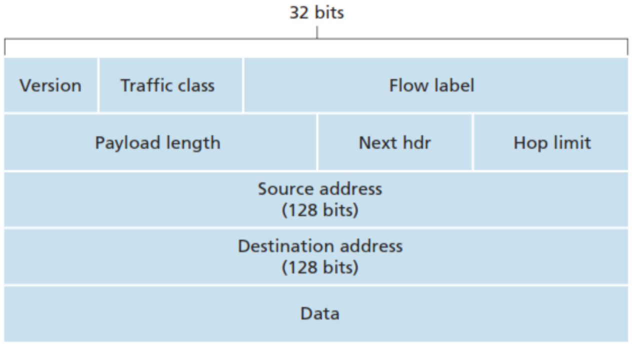


Figure 3.13: IPv6 Datagram Format

- *Expanded addressing capabilities.* IPv6 increases the size of the IP address from 32 to 128 bits. every devices on the planet can be IP-addressable. In addition to unicast and multicast addresses, IPv6 has introduced a new type of address, called an any cast addressing.
- **A streamlined 40-byte header:** 40-byte fixed-length header for faster processing of the IP datagram. A new encoding of options allows for more flexible options processing.
- **Flow labeling and priority:** labeling of packets belonging to particular flows for which the sender requests special handling, such as a nondefault quality of service or real-time service the flows, even if the exact meaning of a flow has not yet been determined.
  - The IPv6 header also has an 8-bit traffic class field. This field, like the TOS field in IPv4, can be used to give priority to certain datagrams within a flow, or it can be used to give priority to datagrams from certain applications (for example, ICMP) over datagrams from other applications
- **Version.** This 4-bit field identifies the IP version number. Not surprisingly, IPv6 carries a value of 6 in this field.
- **Traffic class**. This 8-bit field is similar in spirit to the TOS field we saw in IPv4.
- **Flow label.** 20-bit field is used to identify a flow of datagrams.
- **Payload length.** This 16-bit value is treated as an unsigned integer giving the number of bytes in the IPv6 datagram following the fixed-length, 40-byte datagram header.
- **Next header**. This field identifies the protocol to which the contents (data field) of this datagram will be delivered (for example, to TCP or UDP). The field uses the same values as the protocol field in the IPv4 header.
- **Hop limit**. The contents of this field are decremented by one by each router that forwards the datagram. If the hop limit count reaches zero, the datagram is discarded.
- **Source and destination addresses**. The various formats of the IPv6 128-bit address are described in RFC 4291.
- **Data.** This is the payload portion of the IPv6 datagram. When the datagram reaches its destination, the payload will be removed from the IP datagram and passed on to the protocol specified in the next header field.
- **Fragmentation/Reassembly:** IPv6 does not allow for fragmentation and reassembly at intermediate routers; these operations can be performed only by the source and destination. If an IPv6 datagram received by a router is too large to be forwarded over the outgoing link, the router simply drops the datagram and sends a “Packet Too Big” ICMP error message (see below) back to the sender. The sender can then resend the data, using a smaller IP datagram size. Fragmentation and reassembly is a time-consuming operation; removing this functionality from the routers and placing it squarely in the end systems considerably speeds up IP forwarding within the network.

- **Header checksum.** Because the transport-layer (for example, TCP and UDP) and link-layer (for example, Ethernet) protocols in the Internet layers perform check summing, the designers of IP probably felt that this functionality was sufficiently redundant in the network layer that it could be removed. Once again, fast processing of IP packets was a central concern.
- **Options.** An options field is no longer a part of the standard IP header. However, it has not gone away. Instead, the options field is one of the possible next headers pointed to from within the IPv6 header.

IP Security

Security being a major concern today, Internet researchers have moved on to design new network-layer protocols that provide a variety of security services.

- **Cryptographic agreement**. Mechanisms that allow the two communicating hosts to agree on cryptographic algorithms and keys.
- **Encryption of IP datagram payloads**. When the sending host receives a segment from the transport layer, IPsec encrypts the payload. The payload can only be decrypted by IPsec in the receiving host.
- **Data integrity.** IPsec allows the receiving host to verify that the datagram's header fields and encrypted payload were not modified while the datagram was en route from source to destination.
- **Origin authentication.** When a host receives an IPsec datagram from a trusted source with a trusted key, the host is assured that the source IP address in the datagram is the actual source of the datagram.

Routing Algorithms

Global Routing Algorithm

- A global routing algorithm computes the least-cost path between a source and destination using complete, global knowledge about the network.
- The algorithm takes the connectivity between all nodes and all link costs as inputs then obtain this information before actually performing the calculation.
- It run at one site (a centralized global routing algorithm) or replicated at multiple sites.
  - Eg: link-state (LS) algorithms

Decentralized Routing Algorithm

- The calculation of the least-cost path is carried out in an iterative, distributed manner. No node has complete information about the costs of all network links.
- Each node begins with only the knowledge of the costs of its own directly attached links. Then, through an iterative process of calculation and exchange of information with its neighboring nodes
- A node gradually calculates the least-cost path to a destination or set of destinations
  - Eg: Distance-Vector (DV) Algorithm

Static Routing Algorithms

Routes change very slowly over time, often as a result of human intervention

Dynamic routing algorithms

- Change the routing paths as the network traffic loads or topology change. A dynamic algorithm can be run either periodically or in direct response to topology or link cost changes. While dynamic algorithms are more responsive to network changes, they are also more susceptible to problems such as routing loops and oscillation in routes

Load-sensitive algorithm

- Routing algorithms is according to whether they are load sensitive or load-insensitive
- Link costs vary dynamically to reflect the current level of congestion in the underlying link. If a high cost is associated with a link that is currently congested, a routing algorithm will tend to choose routes around such a congested link
  - Eg: RIP, OSPF, and BGP

Link-State (LS) Algorithm for Source Node: Dijkstra’s algorithm

Note:- 1st Iterations - finds 1st closest path from Source node to All nodes. (14)

Kth Iteration - Find Kth closest node from Source node in N/w

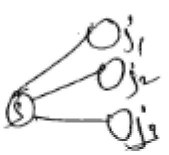
$N = \{sets\}$, D_i = current min cost from the Source node to node 'i'

D_j = Previous cost

Step 1: Initialization

$N = \{S\}$

$D_j = C_{sj} \ \forall \ j \neq S$



$D_S = 0$

if not neighbors then $D_j = \infty$

Step2: Find closest node $i \notin N$ such that

$D_i = \min D_j \quad j \in N$

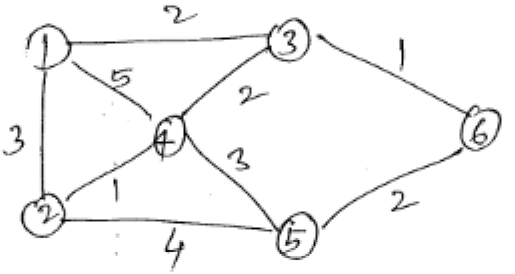
Add 'i' to N

Step3: Updating min cost after node 'i' added (for each node $j \notin N$)

$D_j = \min\{D_j, D_i + C_{ij}\}$

Go to Step 2:

Dijkstra's Algorithm (link cost should be +ve)



Determine shortest path from source to all other node in n/w.

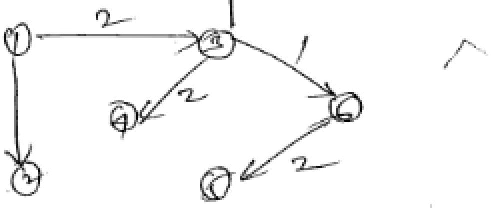
- Source Node: ①, find cost (min) from ∀ nodes.

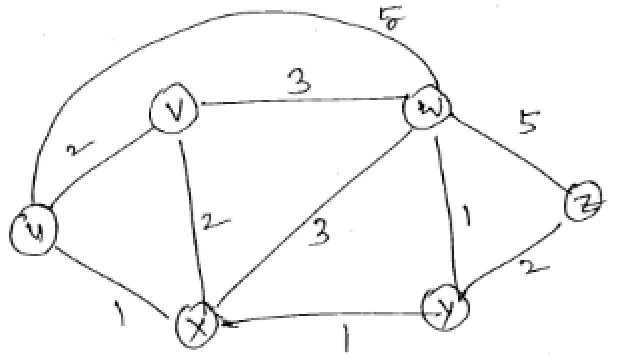
Link state routing table | ~~Forwarding~~ Table.

⇒ Costs

Iterations	N (set)	D2	D3	D4	D5	D6
Initial	{1}	3	2 ✓	5	∞	∞ → not connected
1st	{1,3}	3 ✓	2	4	∞	3
2nd	{1,2,3}	3	2	4	7	3 ✓
3rd	{1,2,3,6}	3	2	4 ✓	5	3
4th	{1,2,3,4,6}	3	2	4	5 ✓	3
5th	{1,2,3,4,5,6}	3	2	4	5	3

Des	Nxt Node	Cost
2	2	3
3	3	2
4	3	4
5	3	5
6	3	3

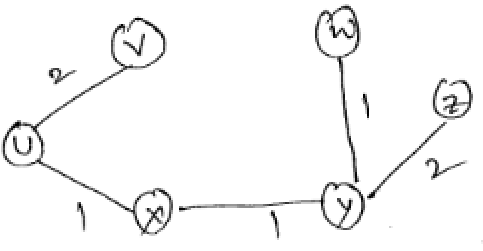




Source = u

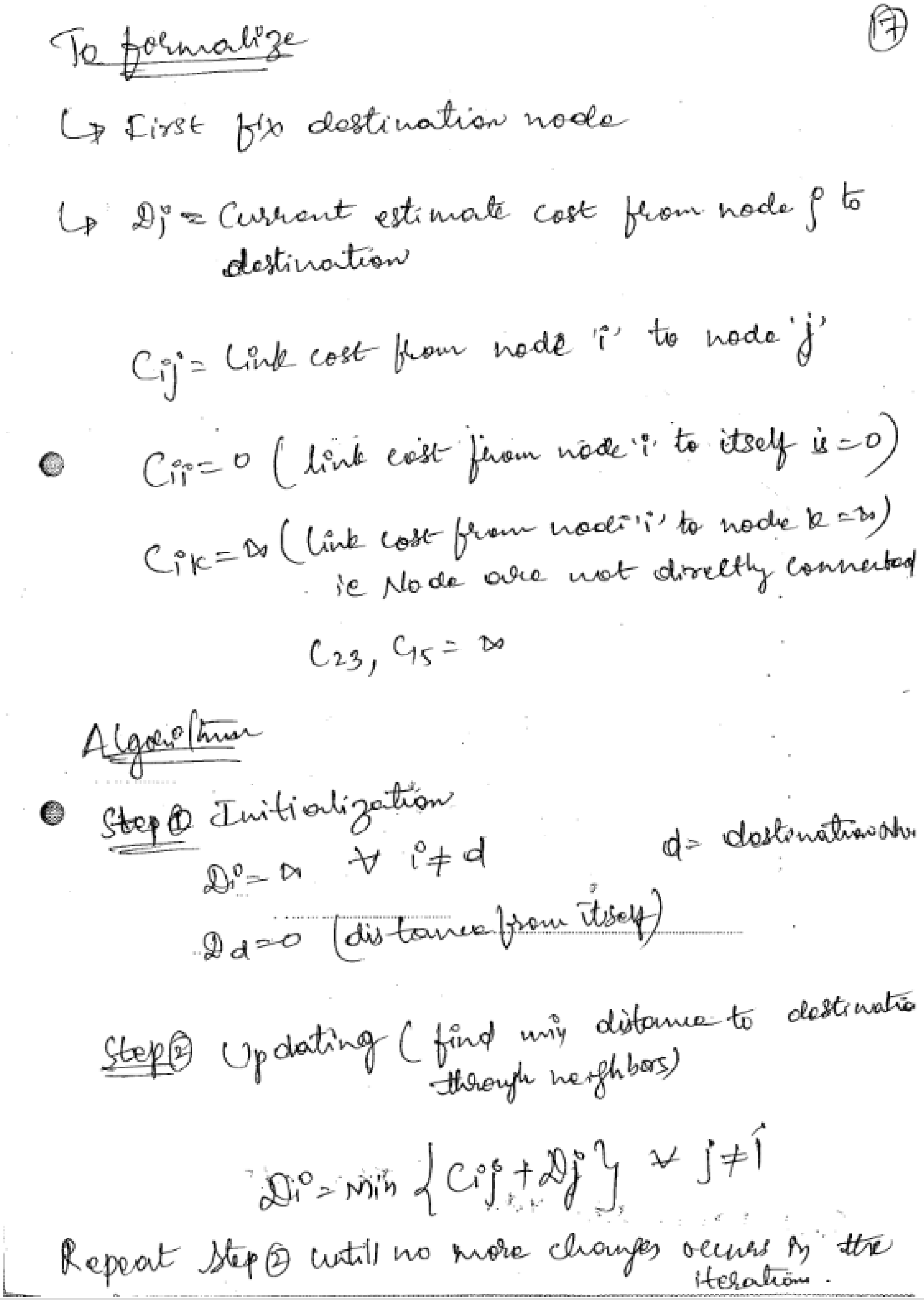
Iterations	N'(set)	V✓	W	X✓	Y✓	Z
Initial	{u}	2	5	1✓	∞	∞
1st	{u,x}	2✓	4	1	2	∞
2nd	{u,x,v}	2	4	1	2✓	∞
3rd	{u,x,v,y}	2	3	1	2	4
4th	{u,x,v,y,w}	2	3	1	2	4
5th	{u v,w,x,y,z}	2	3	1	2	4

Des	Nxt Hop	Cost
V	V	2
W	X	3
X	X	1
Y	X	2
Z	X	4



The Distance-Vector (DV) Routing Algorithm: Bellman Ford Algorithm

- LS algorithm is an algorithm using global information, the Distance Vector (DV) algorithm is iterative, asynchronous, and distributed.
- Distributed in that each node receives some information from one or more of its directly attached neighbors, performs a calculation, and then distributes the results of its calculation back to its neighbors
- It is iterative in that this process continues on until no more information is exchanged between neighbors.



To formalize (17)

↳ First fix destination node

↳ D_j = Current estimate cost from node j to destination

C_{ij} = Link cost from node 'i' to node 'j'

● $C_{ii} = 0$ (link cost from node 'i' to itself is = 0)

$C_{ik} = \infty$ (link cost from node 'i' to node k = ∞) ie Node are not directly connected

$C_{23}, C_{15} = \infty$

Algorithm

● Step ① Initialization d = destination no

$D_i = \infty \quad \forall\ i \neq d$

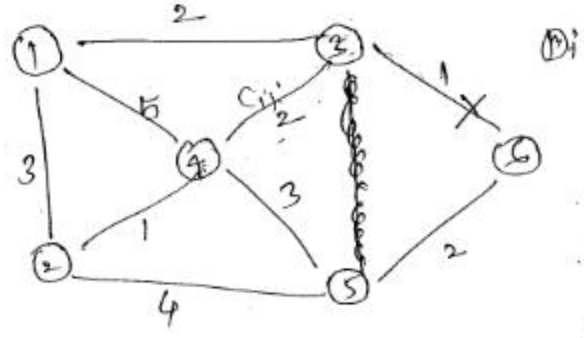
$D_d = 0$ (distance from itself)

Step ② Updating (find min distance to destination through neighbors)

$D_i = \min\{C_{ij} + D_j\} \quad \forall\ j \neq i$

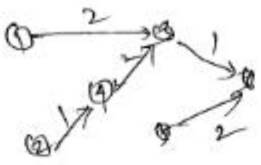
Repeat Step ② until no more changes occurs by the iteration.

Node ⑥ Destination calculate min distance from ∀ Nodes.



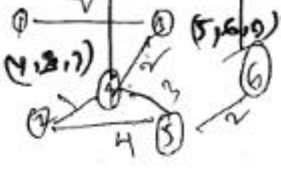
$D_i = \min\,(C_{ij} + D_j)$ 2+3 9 x

-1 = if Node Not connected.



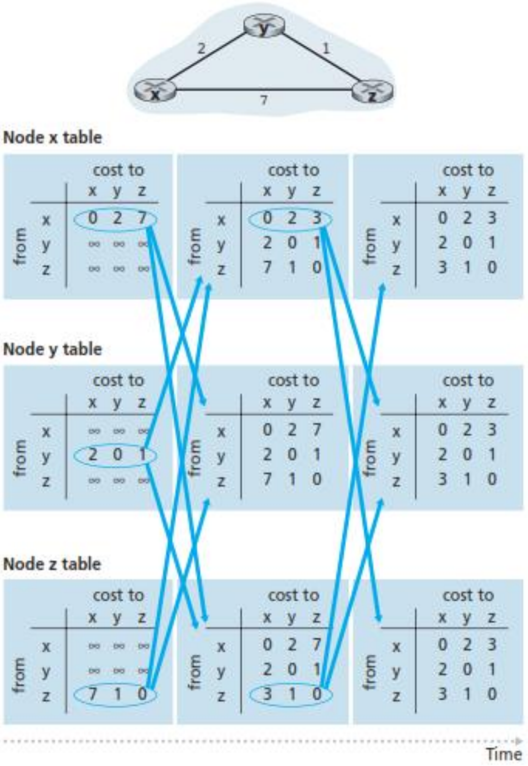
Iterations	Node1	Node2	Node3	Node4	Node5
Initial	(-1, ∞)	(-1, ∞)	(-1, ∞)	(-1, ∞)	(-1, ∞)
1st	(-1, ∞)	(-1, ∞)	(6, 1)	(-1, ∞)	(6, 2)
2nd	(3, 3) (cij+Dj)	(5, 6)	(6, 1)	(3, 3)	(6, 2)
3rd	(3, 3)	(4, 4)	(6, 1)	(3, 3)	(6, 2)
Break			update		
1st	(3, 3)	(4, 4)	(4, 5) (2+3)	(3, 3)	(6, 2)
2nd	3, 7 (5+2)	(4, 4)	(4, 5)	(2+5) (5,5)	(6, 2)
3rd	(3, 7)	(4, 6)	(4, 7)	(2, 5)	(6, 2)
4th	(3, 9)	(4, 6)	(4, 7)	(2, 5)	(6, 2)

(1,3,3) (2,4,4) (3,6,1) (4,3,7)



(5,6,2) (6,6,0)

Figure illustrates the operation of the DV algorithm for the simple three node network shown at the top of the figure. The operation of the algorithm is illustrated in asynchronous manner, where all nodes simultaneously receive distance vectors from their neighbors, compute their new distance vectors, and inform their neighbors if their distance vectors have changed



Node x table

	cost to		
from	x	y	z
x	0	2	7
y	∞	∞	∞
z	∞	∞	∞

	cost to		
from	x	y	z
x	0	2	3
y	2	0	1
z	7	1	0

	cost to		
from	x	y	z
x	0	2	3
y	2	0	1
z	3	1	0

Node y table

	cost to		
from	x	y	z
x	∞	∞	∞
y	2	0	1
z	∞	∞	∞

	cost to		
from	x	y	z
x	0	2	7
y	2	0	1
z	7	1	0

	cost to		
from	x	y	z
x	0	2	3
y	2	0	1
z	3	1	0

Node z table

	cost to		
from	x	y	z
x	∞	∞	∞
y	∞	∞	∞
z	7	1	0

	cost to		
from	x	y	z
x	0	2	7
y	2	0	1
z	3	1	0

	cost to		
from	x	y	z
x	0	2	3
y	2	0	1
z	3	1	0

Distance-Vector Algorithm: Link-Cost Changes and Link Failure

Hierarchical Routing

- In DV and LS algorithms, network is viewed as simply a collection of interconnected routers which runs same routing algorithm to calculate paths through the entire network. In practice, this model and its view of a homogenous set of routers(all executing the same routing algorithm) has 2 main issues:
  - Scale : As the number of routers becomes large, the overhead involved in computing, storing, and communicating routing information becomes prohibitive. Today's public Internet consists of hundreds of millions of hosts. The overhead required to broadcast LS updates among all of the routers in the public Internet would leave no bandwidth left for sending data packets. A distance-vector algorithm that iterated among such a large number of routers would surely never converge.
  - Administrative autonomy. Ideally, an organization should be able to run and administer its network as it wishes, while still being able to connect its network to other outside networks.

- Both of these problems can be solved by organizing routers into autonomous systems (ASs), with each AS consisting of a group of routers that are typically under the same administrative control (e.g., operated by the same ISP or belonging to the same company network).
- Routers within the same AS all run the same routing algorithm (for example, an LS or DV algorithm) and have information about each other.
- The routing algorithm running within an autonomous system is called an **intra-autonomous system routing protocol**.
- Routers which connect with other autonomous systems are called **gateway routers**.

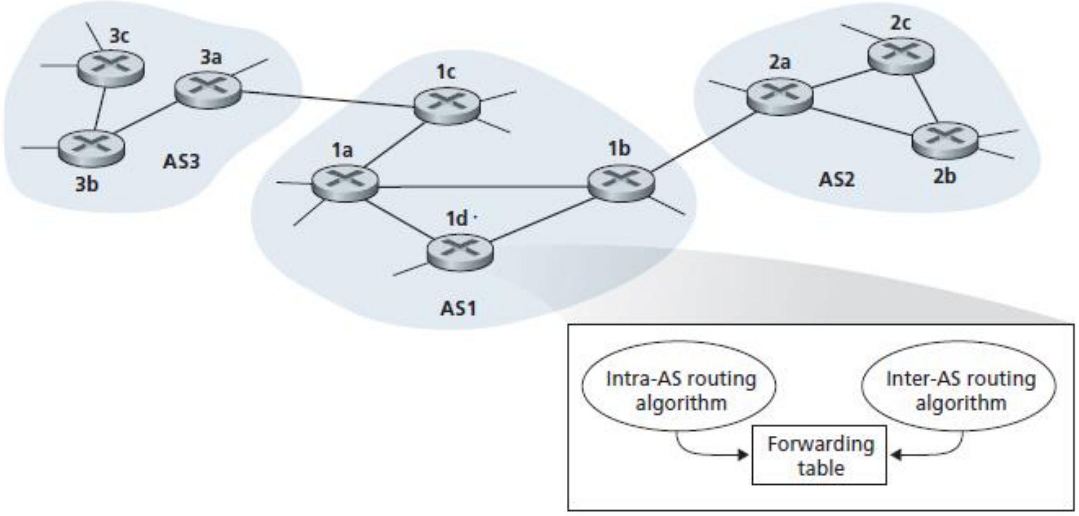


Figure 4.32 ♦ An example of interconnected autonomous systems

- Figure 4.32 provides a simple example with three ASs: AS1, AS2, and AS3.In this figure, the heavy lines represent direct link connections between pairs of routers. The thinner lines hanging from the routers represent subnets that are directly connected to the routers. AS1 has four routers—1a, 1b, 1c, and 1d which run the intra-AS routing protocol used within AS1. Thus, each of these four routers knows how to forward packets along the optimal path to any destination within AS1. Similarly, autonomous systems AS2 and AS3 each have three routers. Note that the intra-AS routing protocols running in AS1, AS2, and AS3 need not be the same. Also note that the routers 1b, 1c, 2a, and 3a are all gateway routers.

- How does a router, within some AS, know how to route a packet to a destination that is outside the AS?
  If we consider AS1 in the fig, it needs
  1. to learn which destinations are reachable via AS2 and which destinations are reachable via AS3, and
  2. to propagate this reachability information to all the routers within AS1, so that each

  router can configure its forwarding table to handle external-AS destinations.
- Above 2 tasks of obtaining reachability information from neighboring ASs and propagating the reachability information to all routers internal to the AS—are handled by the **inter-AS routing protocol**.
- Since the inter-AS routing protocol involves communication between two ASs, the two communicating ASs must run the same inter-AS routing protocol.
- In the Internet all ASs run the same inter-AS routing protocol, called BGP4.
- As shown in Figure 4.32, each router receives information from an intra-AS routing protocol and an inter-AS routing protocol, and uses the information from both protocols to configure its forwarding table.
- suppose that AS2 and AS3 connect to other ASs, which are not shown in the diagram. Also suppose that AS1 learns from the inter-AS routing protocol that subnet *x* is reachable both from AS2, via gateway 1b, and from AS3, via gateway 1c. AS1 would then propagate this information to all its routers, including router 1d. In order to configure its forwarding table, router 1d would have to determine to which gateway router, 1b or 1c, it should direct packets that are destined for subnet *x*.
- One approach, which is often employed in practice, is to use **hot-potato routing**. In hot-potato routing, the AS gets rid of the packet (the hot potato) as quickly as possible (more precisely, as inexpensively as possible). This is done by having a router send the packet to the gateway router that has the smallest router-to-gateway cost among all gateways with a path to the destination.
- When an AS learns about a destination from a neighboring AS, the AS can advertise this routing information to some of its other neighboring ASs.

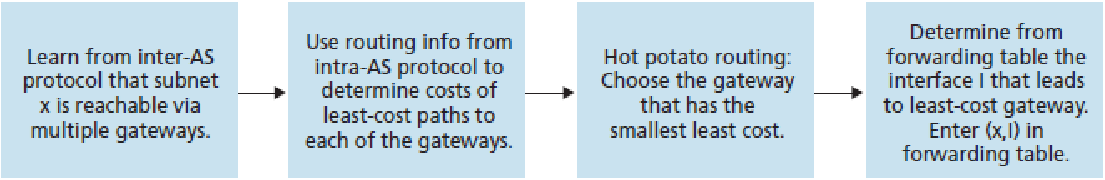

Learn from inter-AS protocol that subnet x is reachable via multiple gateways.
Use routing info from intra-AS protocol to determine costs of least-cost paths to each of the gateways.
Hot potato routing: Choose the gateway that has the smallest least cost.
Determine from forwarding table the interface I that leads to least-cost gateway. Enter (x,I) in forwarding table.

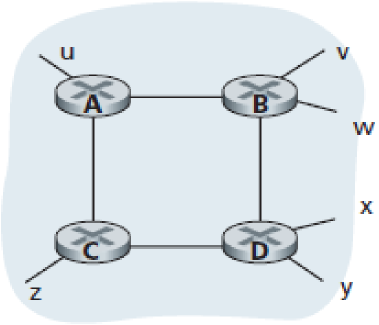
Figure 4.33 ♦ Steps in adding an outside-AS destination in a router's forwarding table

- Problems of scale and administrative authority are solved by defining autonomous systems.

- Within an AS, all routers run the same intra-AS routing protocol. Among themselves, the ASs run the same inter-AS routing protocol. The problem of scale is solved because an intra-AS router need only know about routers within its AS.
- The problem of administrative authority is solved since an organization can run whatever intra-AS routing protocol it chooses. However, each pair of connected ASs needs to run the same inter-AS routing protocol to exchange reachability information.

Intra-AS Routing in the Internet: RIP(Routing Information Protocol)

- RIP operates in a manner very close to the idealized DV protocol.
- It uses hop count as a cost metric. That is, each link has a cost of 1.
- RIP uses the term *hop*, which is the number of subnets traversed along the shortest path from source router to destination subnet, including the destination subnet.
- Figure 4.34 illustrates an AS with six leaf subnets. The table in the figure indicates the number of hops from the source A to each of the leaf subnets.



Destination	Hops
u	1
v	2
w	2
x	3
y	3
z	2

Figure 4.34 ♦ Number of hops from source router A to various subnets

- The maximum cost of a path is limited to 15, thus limiting the use of RIP to autonomous systems that are fewer than 15 hops in diameter.
- In RIP, routing updates are exchanged between neighbors approximately every 30 seconds using a **RIP response message.** Response messages are also known as **RIP advertisements**. RIP advertisements are sent even when there are no changes. All the entries in the table are sent every time.

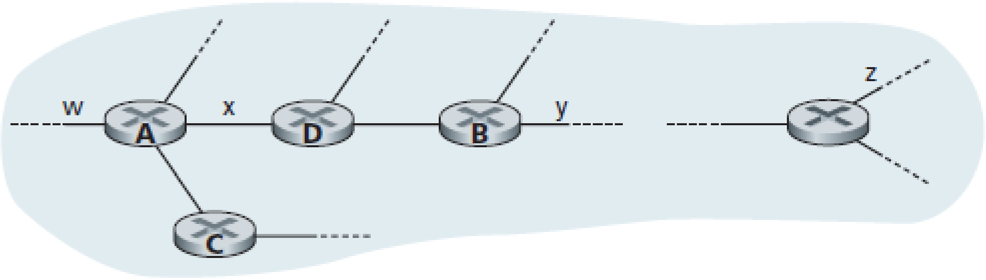
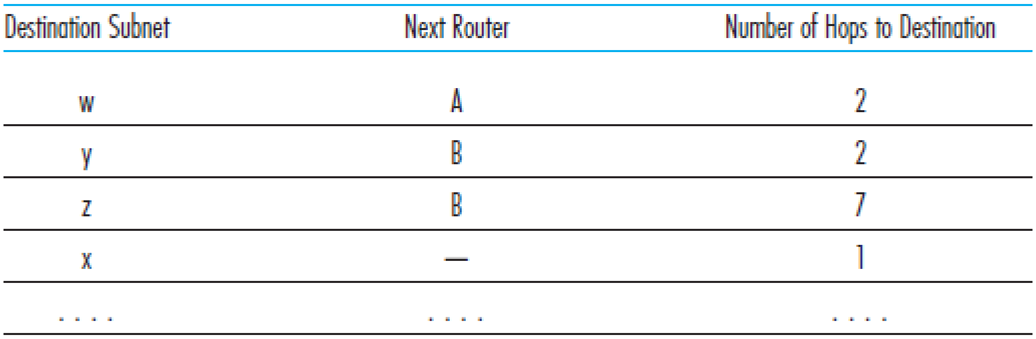


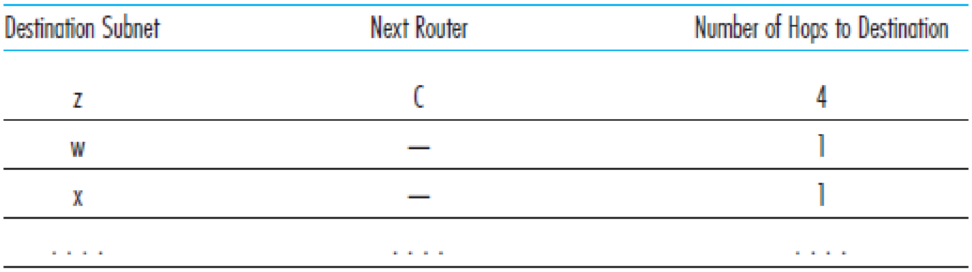
Figure 4.35 ♦ A portion of an autonomous system



Destination Subnet	Next Router	Number of Hops to Destination
w	A	2
y	B	2
z	B	7
x	—	1
....	....	....

Figure 4.36 ♦ Routing table in router *D* before receiving advertisement from router *A*

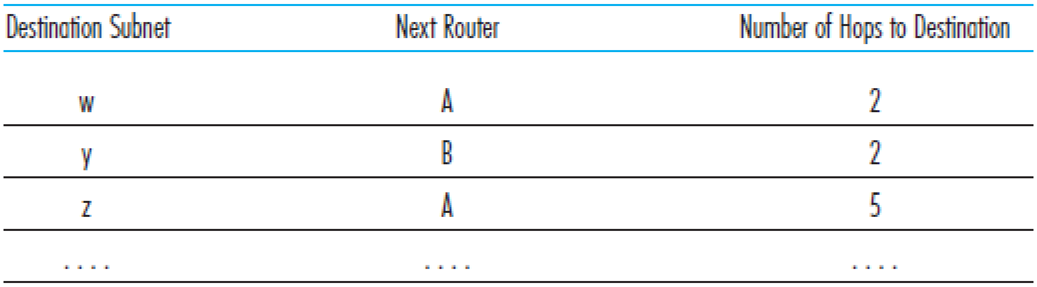
- Now suppose that 30 seconds later, router *D* receives from router *A* the advertisement shown in Figure 4.37. It says subnet **z** is only four hops away from A.



Destination Subnet	Next Router	Number of Hops to Destination
z	C	4
w	—	1
x	—	1
....	....	....

Figure 4.37 ♦ Advertisement from router A

- Router *D* learns that there is now a path through router *A* to subnet *z* that is shorter than the path through router *B*. Thus, router *D* updates its routing table to account for the shorter shortest path, as shown in Figure 4.38.



Destination Subnet	Next Router	Number of Hops to Destination
w	A	2
y	B	2
z	A	5
....	....	....

Figure 4.38 ♦ Routing table in router *D* after receiving advertisement from router *A*

- Figure 4.39 sketches how RIP is typically implemented in a UNIX system, for example, a UNIX workstation serving as a router. A process called *routed* executes RIP, that is, maintains routing information and exchanges messages with *routed* processes running in neighboring routers. Because RIP is implemented as an application-layer process, it can send and receive messages over a standard socket and use a standard transport protocol.

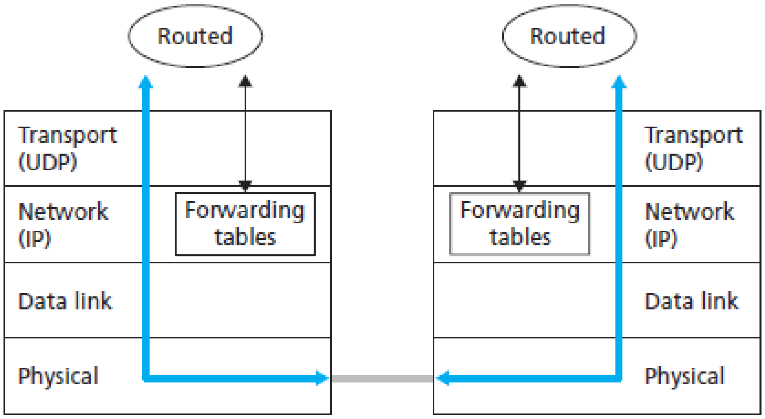


Figure 4.39 ♦ Implementation of RIP as the *routed* daemon

Intra-AS Routing in the Internet: OSPF (Open Shortest Path First)

- Like RIP, OSPF routing is widely used for intra-AS routing in the Internet.
- At its heart, OSPF is a link-state protocol that uses flooding of link-state information and a Dijkstra least-cost path algorithm. With OSPF, a router constructs a complete topological map (that is, a graph) of the entire autonomous system. The router then locally runs Dijkstra's shortest-path algorithm to determine a shortest-path tree to all *subnets*, with itself as the root node.
- A router broadcasts linkstate information whenever there is a change in a link's state (for example, a change in cost or a change in up/down status). It also broadcasts a link's state periodically (at least once every 30 minutes), even if the link's state has not changed.

Special features

1. ***Security:*** Exchanges between OSPF routers can be authenticated. With authentication, only trusted routers can participate in the OSPF protocol within an AS, thus preventing malicious intruders from injecting incorrect information into router tables. By default, OSPF packets between routers are not authenticated and could be forged. Two types of authentication can be configured—simple and MD5. With simple authentication, the same password is configured on each router. When a router sends an OSPF packet, it includes the password in plaintext. Clearly, simple authentication is not very secure. MD5 authentication is based on shared secret keys that are configured in all the routers.
2. **Multiple same-cost paths:** When multiple paths to a destination have the same cost, OSPF allows multiple paths to be used (that is, a single path need not be chosen for carrying all traffic when multiple equal-cost paths exist).
3. **Integrated support for unicast and multicast routing:** Multicast OSPF (MOSPF) provides simple extensions to OSPF to provide for multicast routing. MOSPF uses the existing OSPF link database and adds a new type of link-state advertisement to the existing OSPF link-state broadcast mechanism.
4. **Support for hierarchy within a single routing domain:** An OSPF autonomous system can be configured hierarchically into areas. Each area runs its own OSPF link-state routing algorithm, with each router in an area broadcasting its link state to all other routers in that area. Within each area, one or more area border routers are responsible for routing packets outside the area. Lastly, exactly one OSPF area in the

AS is configured to be the backbone area. The primary role of the backbone area is to route traffic between the other areas in the AS.

Inter-AS Routing: BGP

- Border Gateway Protocol BGP provides each AS a means to

  1. Obtain subnet reachability information from neighboring ASs.

  2. Propagate the reachability information to all routers internal to the AS.

  3. Determine "good" routes to subnets based on the reachability information and on AS policy.

- BGP allows each subnet to advertise its existence to the rest of the Internet. BGP makes sure that all the ASs in the Internet know about the subnet and how to get there. Without BGP, each subnet would be isolated—alone and unknown by the rest of the Internet.
- In BGP, pairs of routers exchange routing information over semipermanent TCP connections using port 179. The semi-permanent TCP connections for the network in Figure 4.32 are shown in Figure 4.40.
- There is typically one such BGP TCP connection for each link that directly connects two routers in two different Ass. In Figure 4.40, there is a TCP connection between gateway routers 3a and 1c and another TCP connection between gateway routers 1b and 2a. There are also semipermanent BGP TCP connections between routers within an AS.
- For each TCP connection, the two routers at the end of the connection are called **BGP peers**, and the TCP connection along with all the BGP messages sent over the connection is called a **BGP session**. Furthermore, a BGP session that spans two Ass is called an **external BGP** (**eBGP**) **session**, and a BGP session between routers in the same AS is called an **internal BGP** (**iBGP**) **session**.

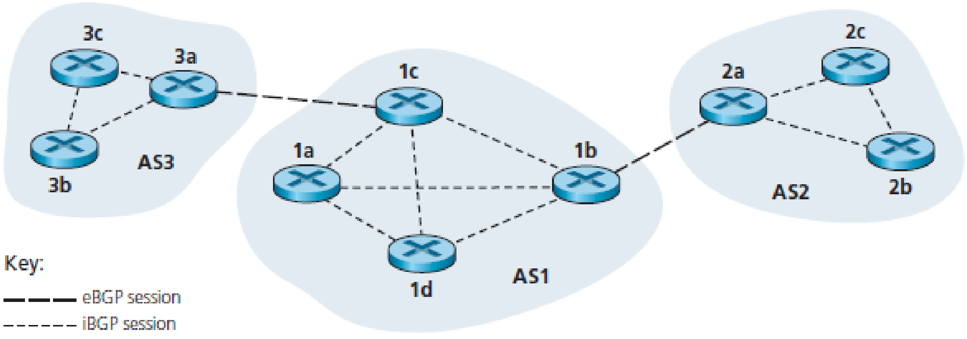


Figure 4.40 ♦ eBGP and iBGP sessions

- In BGP, destinations are not hosts but instead are CIDRized **prefixes**, with each prefix representing a subnet or a collection of subnets.

Routing advertisements

- As shown in the fig 4.40, using the eBGP session between the gateway routers 3a and 1c, AS3 sends AS1 the list of prefixes that are reachable from AS3; and AS1 sends AS3 the list of prefixes that are reachable from AS1.

- Similarly, AS1 and AS2 exchange prefix reachability information through their gateway routers 1b and 2a.
- When a gateway router (in any AS) receives eBGP-learned prefixes, the gateway router uses its iBGP sessions to distribute the prefixes to the other routers in the AS. Thus, all the routers in AS1 learn about AS3 prefixes, including the gateway router 1b. The gateway router 1b (in AS1) can therefore re-advertise AS3's prefixes to AS2. When a router (gateway or not) learns about a new prefix, it creates an entry for the prefix in its forwarding table

Path Attributes and BGP Routes

- In BGP, an autonomous system is identified by its globally unique **autonomous system number(ASN)**.
- When a router advertises a prefix across a BGP session, it includes with the prefix a number of **BGP attributes**. A prefix along with its attributes is called a **route**. Thus, BGP peers advertise routes to each other.
- Two of the more important attributes are AS-PATH and NEXT-HOP:

AS-PATH: This attribute contains the ASs through which the advertisement for the prefix has passed. When a prefix is passed into an AS, the AS adds its ASN to the ASPATH attribute. For example, consider Figure 4.40 and suppose that prefix 138.16.64/24 is first advertised from AS2 to AS1. if AS1 then advertises the prefix to AS3, AS-PATH would be AS2 AS1.

NEXT-HOP: Whenever there are multiple links from current AS to next-hop AS, NEXT-HOP indicates specific internal-AS router to next-hop AS.

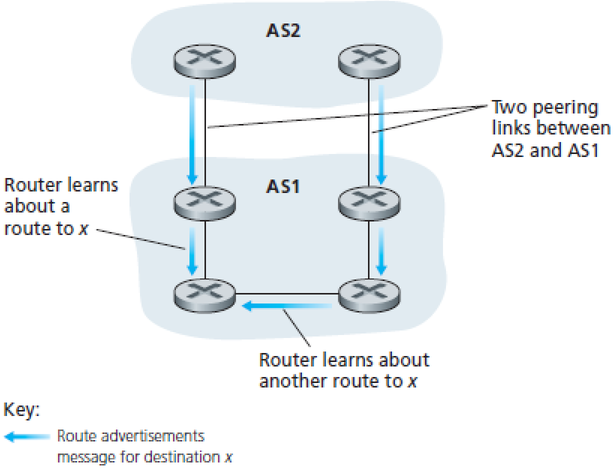

AS2
Two peering links between AS2 and AS1
AS1
Router learns about a route to *x*
Router learns about another route to *x*
Key:
Route advertisements message for destination *x*

Figure 4.41 ♦ NEXT-HOP attributes in advertisements are used to determine which peering link to use

BGP Route Selection

- BGP uses eBGP and iBGP to distribute routes to all the routers within ASs. From this distribution, a router may learn about more than one route to any one prefix, in which case the router must select one of the possible routes.

- The input into this route selection process is the set of all routes that have been learned and accepted by the router. If there are two or more routes to the same prefix, then BGP sequentially invokes the following elimination rules until one route remains:
  - Routes are assigned a local preference value as one of their attributes. The local preference of a route could have been set by the router or could have been learned by another router in the same AS. This is a policy decision that is left up to the AS's network administrator. The routes with the highest local preference values are selected.
  - From the remaining routes (all with the same local preference value), the route with the shortest AS-PATH is selected. If this rule were the only rule for route selection, then BGP would be using a DV algorithm for path determination, where the distance metric uses the number of AS hops rather than the number of router hops.
  - From the remaining routes (all with the same local preference value and the same AS-PATH length), the route with the closest NEXT-HOP router is selected.
  - If more than one route still remains, the router uses BGP identifiers to select the route.

Routing Policy

- Figure 4.42 shows six interconnected autonomous systems: A, B, C, W, X, and Y.
- Let's assume that autonomous systems W, X, and Yare stub networks and that A, B, and C are backbone provider networks. We'll also assume that A, B, and C, all peer with each other, and provide full BGP information to their customer networks.
- All traffic entering a **stub network** must be destined for that network, and all traffic leaving a stub network must have originated in that network. W and Y are clearly stub networks. X is a **multihomed stub network,** since it is connected to the rest of the network via two different providers. However, like W and Y, X itself must be the source/destination of all traffic leaving/entering X.
- X should be prevented from forwarding traffic between B and C. This is accomplished by controlling the manner in which BGP routes are advertised. Even though X may know of a path, say XCY, that reaches network Y, it will *not* advertise this path to B. So, B would never forward traffic destined to Y (or C) via X. This simple example illustrates how a selective route advertisement policy can be used to implement customer/provider routing relationships.

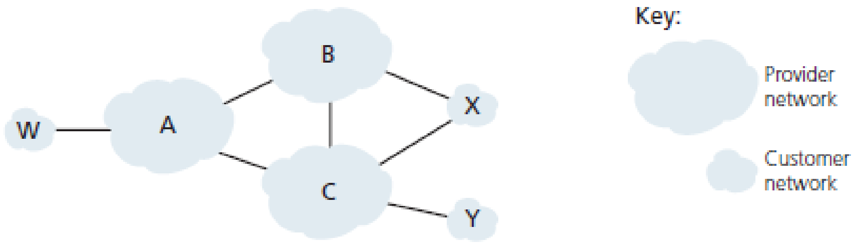


Figure 4.42 ♦ A simple BGP scenario

- A advertises path AW to B. B can thus install the route BAW into its routing information base. Clearly, B also wants to advertise the path BAW to its customer, X, so that X knows that it can route to W via B. But should B advertise the path BAW to C? If it does so, then C could route traffic to W via CBAW. If A, B and C are all backbone providers, than B might rightly feel that it should not have to shoulder the burden (and cost) of carrying transit traffic between A and C. B might rightly feel that it is A's and C's job (and cost) to make sure that C can route to/from A's customers via a direct connection between A and C.

- There are currently no official standards that govern how backbone ISPs route among themselves. However, a rule of thumb followed by commercial ISPs is that any traffic flowing across an ISP's backbone network must have either a source or a destination (or both) in a network that is a customer of that ISP. Otherwise the traffic would be getting a free ride on the ISP's network.

Broadcast Routing Algorithms

- In broadcast routing, the network layer provides a service of delivering a packet sent from a source node to all other nodes in the network.

N-wayunicast approach

- Given N destination nodes, the source node simply makes N copies of the packet, addresses each copy to a different destination, and then transmits the N copies to the N destinations using unicast routing. This **N-wayunicast** approach to broadcasting is simple—no new network-layer routing protocol, packet-duplication, or forwarding functionality is needed.

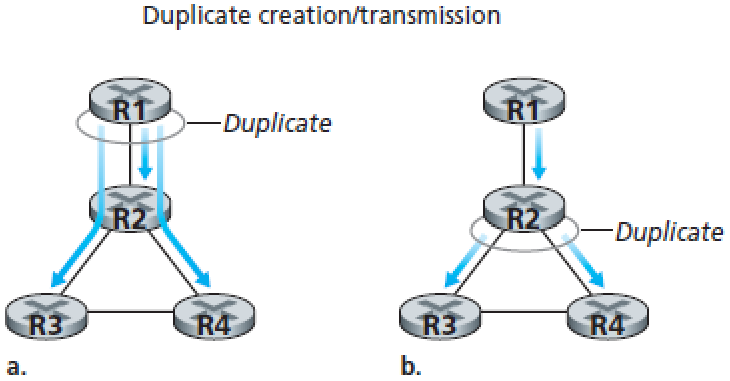


Figure 4.43 ♦ Source-duplication versus in-network duplication

Drawbacks:

- If the source node is connected to the rest of the network via a single link, then N separate copies of the (same) packet will traverse this single link. It would clearly be more efficient to send only a single copy of a packet over this first hop and then have the node at the other end of the first hop make and forward any additional needed copies. That is, it would be more efficient for the network nodes themselves (rather than just the source node) to create duplicate copies of a packet. For example, in Figure 4.43(b), only a single copy of a packet traverses the R1-R2 link. That packet is then duplicated at R2, with a single copy being sent over links R2-R3 and R2-R4.
- Sender should know all receivers and their addresses. To do this additional protocol mechanisms (such as a broadcast membership or destination-registration protocol) would be required. This would add more overhead and, importantly, additional complexity to a protocol that had initially seemed quite simple.
- it would be unwise to rely on the unicast routing infrastructure to achieve broadcast.

Uncontrolled Flooding

- The most obvious technique for achieving broadcast is a **flooding** approach in which the source node sends a copy of the packet to all of its neighbors. When a node receives a broadcast packet, it duplicates the packet and forwards it to all of its neighbors (except the neighbor from which it received the packet).
- Iif the graph is connected, this scheme will eventually deliver a copy of the broadcast packet to all nodes in the graph.
- Although this scheme is simple and elegant, it has a fatal flaw: If the graph has cycles, then one or more copies of each broadcast packet will cycle indefinitely. For example, in Figure 4.43, R2 will flood to R3, R3 will flood to R4, R4 will flood to R2, and R2 will flood (again!) to R3, and so on. This simple scenario results in the endless cycling of two broadcast packets, one clockwise, and one counterclockwise.
- When a node is connected to more than two other nodes, it will create and forward multiple copies of the broadcast packet, each of which will create multiple copies of itself (at other nodes with more than two neighbors), and so on. This **broadcast storm**, resulting from the endless multiplication of broadcast packets, would eventually result in so many broadcast packets being created that the network would be rendered useless.

Controlled Flooding

- Broadcast storm can be avoided by using following methods:
  1. Sequence number controlled flooding
  2. Reverse path forwarding
  3. Spanning tree broadcast

sequence-number-controlled flooding:

- A source node puts its address (or other unique identifier) as well as a **broadcast sequence number** into a broadcast packet, then sends the packet to all of its neighbors.
- Each node maintains a list of the source address and sequence number of each broadcast packet it has already received, duplicated, and forwarded.
- When a node receives a broadcast packet, it first checks whether the packet is in this list. If so, the packet is dropped. If not, packet is duplicated and forwarded to all the node’s neighbors (except the node from which the packet has just been received).

Reverse path forwarding:

- When a router receives a broadcast packet with a given source address, it transmits the packet on all of its outgoing links (except the one on which it was received) only if the packet arrived on the link that is on its own shortest unicast path back to the source. Otherwise, the router simply discards the incoming packet without forwarding it on any of its outgoing links.
- Such a packet can be dropped because the router knows it either will receive or has already received a copy of this packet on the link that is on its own shortest path back to the sender.
- Note that RPF does not use unicast routing to actually deliver a packet to a destination, nor does it require that a router know the complete shortest path from itself to the source. RPF need only know the next neighbor on its unicast shortest path to the sender.

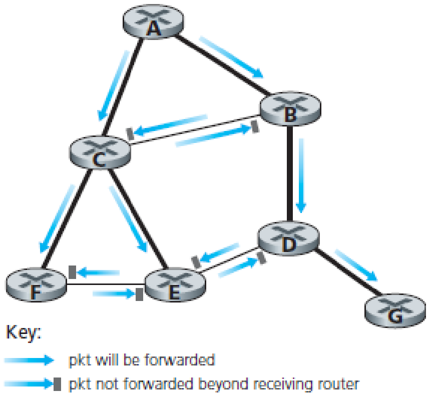


Figure 4.44 ♦ Reverse path forwarding

- ➢ Figure 4.44 illustrates RPF. Suppose that the links drawn with thick lines represent the least-cost paths from the receivers to the source (*A*). Node *A* initially broadcasts a source-*A* packet to nodes *C* and *B*. Node *B* will forward the source-*A* packet it has received from *A* (since *A* is on its least-cost path to *A*) to both *C* and *D*. *B* will ignore (drop, without forwarding) any source-*A* packets it receives from any other nodes (for example, from routers *C* or *D*). Let us now consider node *C*, which will receive a source-*A* packet directly from *A* as well as from *B*. Since *B* is not on *C*'s own shortest path back to *A*, *C* will ignore any source-*A* packets it receives from *B*. On the other hand, when *C* receives a source-*A* packet directly from *A*, it will forward the packet to nodes *B*, *E*, and *F*.

Spanning-Tree Broadcast

- ➢ While sequence-number-controlled flooding and RPF avoid broadcast storms, they do not completely avoid the transmission of redundant broadcast packets. For example, in Figure 4.44, nodes *B*, *C*, *D*, *E*, and *F* receive either one or two redundant packets. Ideally, every node should receive only one copy of the broadcast packet.
- ➢ Another approach to providing broadcast is for the network nodes to first construct a spanning tree. When a source node wants to send a broadcast packet, it sends the packet out on all of the incident links that belong to the spanning tree. A node receiving a broadcast packet then forwards the packet to all its neighbors in the spanning tree (except the neighbor from which it received the packet). Not only does spanning tree eliminate redundant broadcast packets, but once in place, the spanning tree can be used by any node to begin a broadcast, as shown in Figures 4.45(a) and 4.45(b).

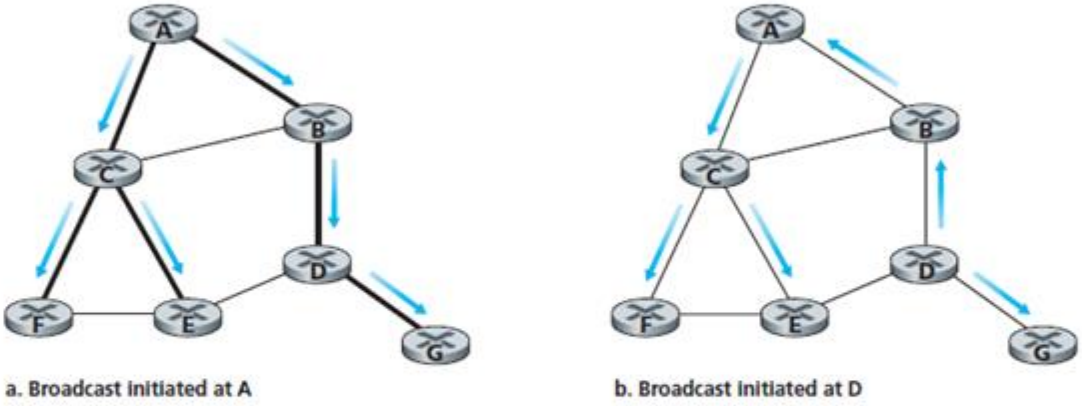
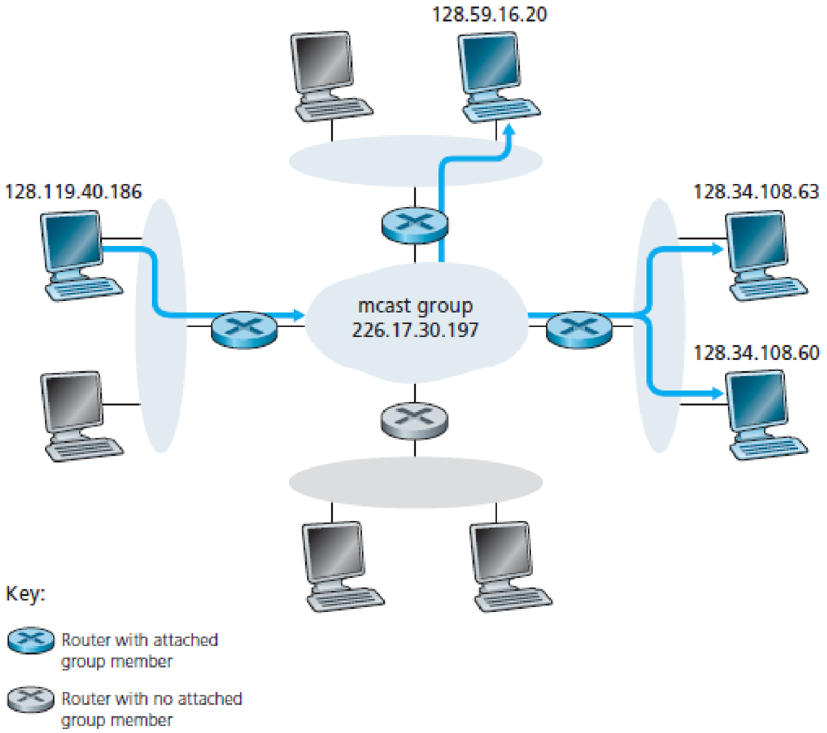


Figure 4.45 ♦ Broadcast along a spanning tree

Multicast

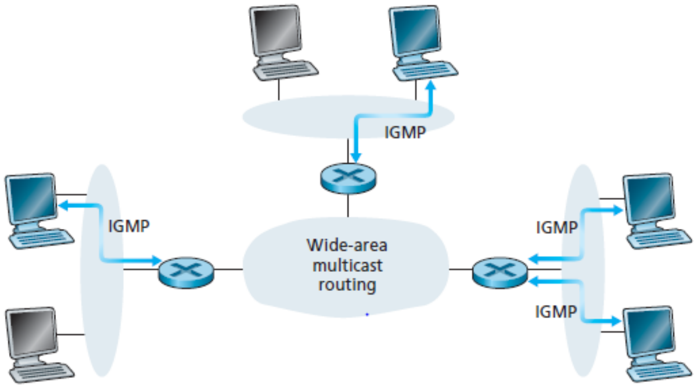
- **multicast routing** enables a single source node to send a copy of a packet to a subset of the other network nodes.
- 2 issues that should be addressed in multicast are
  1. How to identify the receivers of a multicast packet?
  2. How to address a packet sent to these receivers?
- Above problems are solved by **address indirection.**
- A multicast packet is addressed using **address indirection**. That is, a single identifier is used for the group of receivers, and a copy of the packet that is addressed to the group using this single identifier is delivered to all of the multicast receivers associated with that group.
- In the Internet, the single identifier that represents a group of receivers is a class D multicast IP address. The group of receivers associated with a class D address is referred to as a **multicast group**. The multicast group abstraction is illustrated in Figure 4.47. Here, four hosts (shown in shaded color) are associated with the multicast group address of 226.17.30.197 and will receive all datagrams addressed to that multicast address.



Internet Group Management Protocol

- The IGMP protocol version 3 operates between a host and its directly attached router.
- IGMP provides the means for a host to inform its attached router that an application running on the host wants to join a specific multicast group.
- IGMP has only three message types.
  1. **The membership_query message** is sent by a router to all hosts on an attached interface (for example, to all hosts on a local area network) to determine the set of all multicast groups that have been joined by the hosts on that interface.
  2. **IGMP membership_report :** Hosts respond to a membership_query message with

3. an IGMP membership_report message. membership_report messages can also be generated by a host when an application first joins a multicast group without waiting for a membership_query message from the router.
4. **leave_group :** This message is optional. the router infers that a host is no longer in the multicast group if it no longer responds to a membership_query message with the given group address. This is an example of what is sometimes called soft state in an Internet protocol.



5. **Figure 4.48** ♦ The two components of network-layer multicast in the Internet: IGMP and multicast routing protocols

Multicast Routing Algorithms

- ➢ The multicast routing problem is illustrated in Figure 4.49. Hosts joined to the multicast group are shaded in color. Their immediately attached router is also shaded in color. As shown in Figure 4.49, only a subset of routers (those with attached hosts that are joined to the multicast group) actually needs to receive the multicast traffic.
- ➢ In Figure 4.49, only routers A, B, E, and F need to receive the multicast traffic. Since none of the hosts attached to router D are joined to the multicast group and since router C has no attached hosts, neither C nor D needs to receive the multicast group traffic. The goal of multicast routing, then, is to find a tree of links that connects all of the routers that have attached hosts belonging to the multicast group.

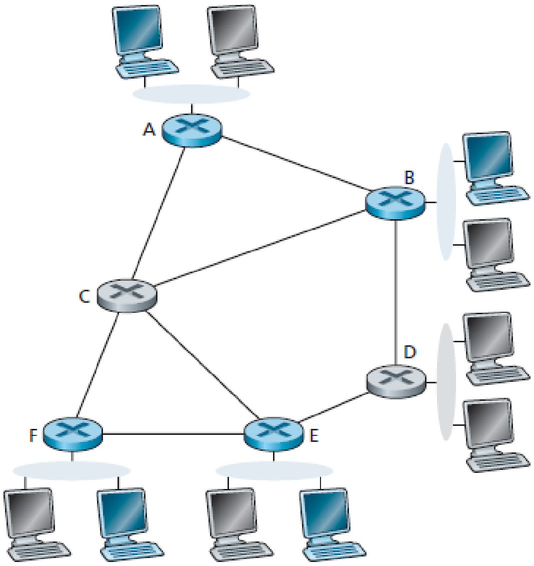


Figure 4.49 ♦ Multicast hosts, their attached routers, and other routers

- In practice, two approaches have been adopted for determining the multicast routing tree:
  - ***Multicast routing using a group-shared tree:***
    - Constructs a single, shared routing tree to toute packets from all senders.
    - Center-based approach is used to construct the multicast routing tree, with edge routers with attached hosts belonging to the multicast group sending (via unicast) join messages addressed to the center node.
    - As in the broadcast case, a join message is forwarded using unicast routing toward the center until it either arrives at a router that already belongs to the multicast tree or arrives at the center. All routers along the path that the join message follows will then forward received multicast packets to the edge router that initiated the multicast join.
  - ***Multicast routing using a source-based tree:***
    - Constructs a multicast routing tree for each source in the multicast group.
    - In practice, an RPF algorithm (with source node *x*) is used to construct a multicast forwarding tree for multicast datagrams originating at source *x*. The RPF broadcast algorithm we studied earlier requires a bit of tweaking for use in multicast. If there were thousands of routers downstream from, each of these thousands of routers would receive unwanted multicast packets.
    - The solution to the problem of receiving unwanted multicast packets under RPF is known as **pruning**. A multicast router that receives multicast packets and has no attached hosts joined to that group will send a prune message to its upstream router. If a router receives prune messages from each of its downstream routers, then it can forward a prune message upstream.

Multicast Routing in the Internet

- Distance-Vector Multicast Routing Protocol (DVMRP):
  - First multicast routing protocol used in the Internet.
  - Implements source-based trees with reverse path forwarding and pruning. DVMRP uses an RPF algorithm with pruning.
- Protocol-Independent Multicast (PIM) routing protocol:
  - most widely used Internet multicast routing protocol.
  - It explicitly recognizes two multicast distribution scenarios. In **dense mode**, multicast group members are densely located, that is, many or most of the routers in the area need to be involved in routing multicast datagrams. PIM dense mode is a flood-and-prune reverse path forwarding technique similar in spirit to DVMRP. In **sparse mode**, the number of routers with attached group members is small with respect to the total number of routers. Group members are widely dispersed. PIM sparse mode uses rendezvous points to set up the multicast distribution tree.

- Source-Specific Multicast (SSM):
  - Only a single sender is allowed to send traffic into the multicast tree, considerably simplifying tree construction and maintenance.