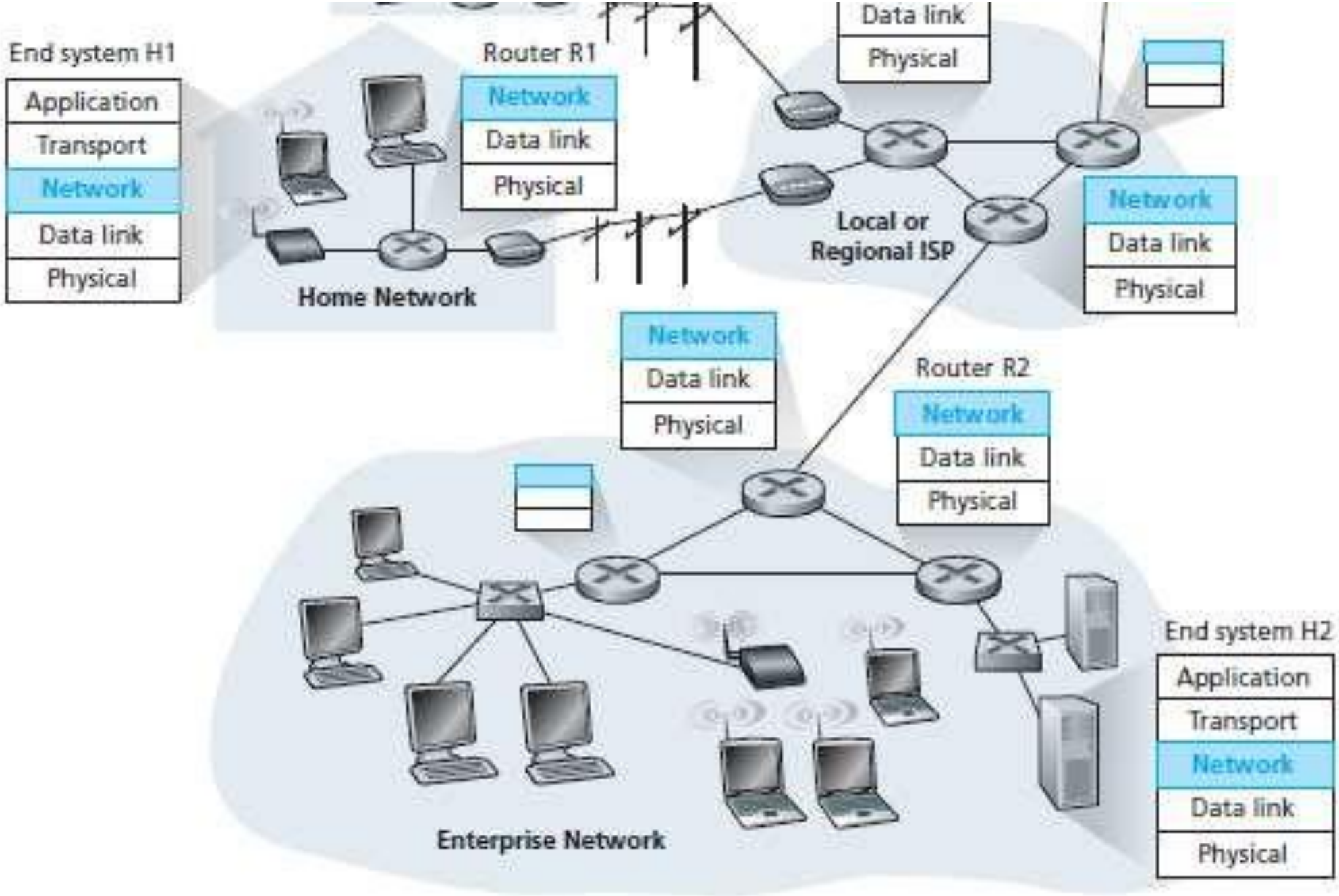
Network Layer

Module 3

Network Layer

- Network layer provides host-to-host communication service.
- Unlike the transport and application layers, there is a piece of the network layer in each and every host and router in the network.
- Two important functions of the network layer are f**orwarding** and **routing.**
- Forwarding involves the transfer of a packet from an incoming link to an outgoing link within a *single* router.
- Routing involves *all* of a network's routers, whose collective interactions via routing protocols determine the paths that packets take on their trips from source to destination node.


End system H1
Application
Transport
Network
Data link
Physical
Router R1
Network
Data link
Physical
Home Network
Data link
Physical
Local or
Regional ISP
Network
Data link
Physical
Network
Data link
Physical
Router R2
Network
Data link
Physical
Enterprise Network
End system H2
Application
Transport
Network
Data link
Physical

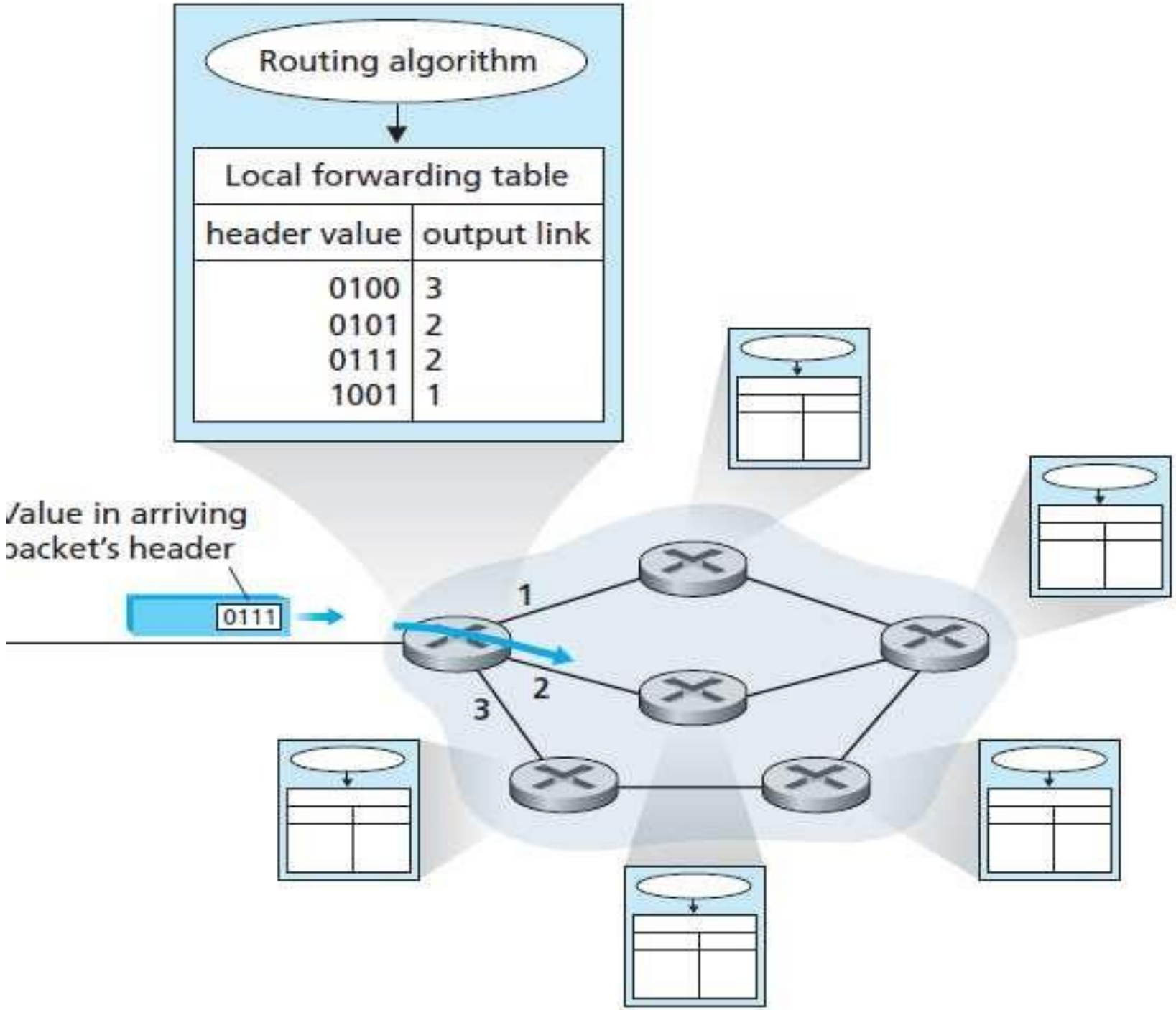
- Figure shows a simple network with two hosts, H1 and H2, and several routers on the path between H1 and H2.
- Suppose that H1 is sending information to H2.
- The network layer in H1 takes segments from the transport layer in H1, encapsulates each segment into a datagram (that is, a network-layer packet), and then sends the datagrams to its nearby router, R1.
- At the receiving host, H2, the network layer receives the datagrams from its nearby router R2, extracts the transport-layer segments, and delivers the segments up to the transport layer at H2.
- The primary role of the routers is to forward datagrams from input links to output links.
- The routers are shown with a truncated protocol stack, that is, with no upper layers above the network layer, because (except for control purposes) routers do not run application- and transport-layer protocols.

Forwarding and Routing

- The role of the network layer is to move packets from a sending host to a receiving host.
- To do so, two important network-layer functions can be identified:
- *Forwarding:* When a packet arrives at a router's input link, the router must move the packet to the appropriate output link. For example, a packet arriving from Host H1 to Router R1 must be forwarded to the next router on a path to H2.
- *Routing :* 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**. A routing algorithm would determine, for example, the path along which packets flow from H1 to H2.

- Forwarding is like a car enters the junction from one road and determines which road it should take to leave the junction.
- Routing is like the process of planning the trip from Pala to Munnar: Before embarking on the trip, the driver has consulted a map and chosen one of many paths possible, with each path consisting of a series of road segments connected at junctions.

- 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 address of the packet or an indication of the connection to which the packet belongs.


Routing algorithm
Local forwarding table
header value
output link
0100 3
0101 2
0111 2
1001 1
Value in arriving
packet's header
0111
1
2
3

- A packet with a header field value of 0111 arrives to a router.
- The router indexes into its forwarding table and determines that the output link interface for this packet is interface 2.
- The router then internally forwards the packet to interface 2.
- The routing algorithm determines the values that are inserted into the routers' forwarding tables.
- The routing algorithm may be centralized (e.g., with an algorithm executing on a central site and downloading routing information to each of the routers) or decentralized (i.e., with a piece of the distributed routing algorithm running in each router).
- In either case, a router receives routing protocol messages, which are used to configure its forwarding table.

Network Service Models

- 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 (for example, within 100 msec).
- *In-order packet delivery*. This service guarantees that packets arrive at the destination in the order that they were sent.

- *Guaranteed minimal bandwidth* As long as the sending host transmits bits (as part of packets) at a rate below the specified bit rate, then no packet is lost and each packet arrives within a prespecified host-to-host delay (for example, within 40 msec).

- *Guaranteed maximum jitter*. This service guarantees that the amount of time between the transmission of two successive packets at the sender is equal to the amount of time between their receipt 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.

Network Architecture	Service Model	Bandwidth Guarantee	No-Loss Guarantee	Ordering	Timing	Congestion Indication
Internet	Best Effort	None	None	Any order possible	Not maintained	None
ATM	CBR	Guaranteed constant rate	Yes	In order	Maintained	Congestion will not occur
ATM	ABR	Guaranteed minimum	None	In order	Not maintained	Congestion indication provided

Virtual Circuit and Datagram Networks

- Network layer can provide connectionless service or connection service between two hosts.
- A network- layer connection service begins with handshaking between the source and destination hosts; and a network-layer connectionless service does not have any handshaking preliminaries.
- There are crucial differences from transport-layer connection-oriented and connectionless services:
- In the network layer, these services are host-to-host services. transport layer these services are process to- process services.
- Computer network architectures does not provide both connectionless service or a host-to-host connection service at same time.

- 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**.

- The network-layer connection service is implemented in the routers in the network core as well as in the end systems.

Virtual circuits

- Call setup, teardown for each call *before* data can flow
- Each packet carries VC identifier (not destination host address)
- *Every* router on source-destination path maintains "state" for each passing connection
- link, router resources (bandwidth, buffers) may be *allocated* to VC (dedicated resources = predictable service)

VC implementation

A VC consists of:

1. *path* from source to destination
2. *VC numbers*, one number for each link along path
3. *entries in forwarding tables* in routers along path

- packet belonging to VC carries VC number (rather than dest address)
- VC number can be changed on each link.
  - new VC number comes from forwarding table

Virtual circuit setup

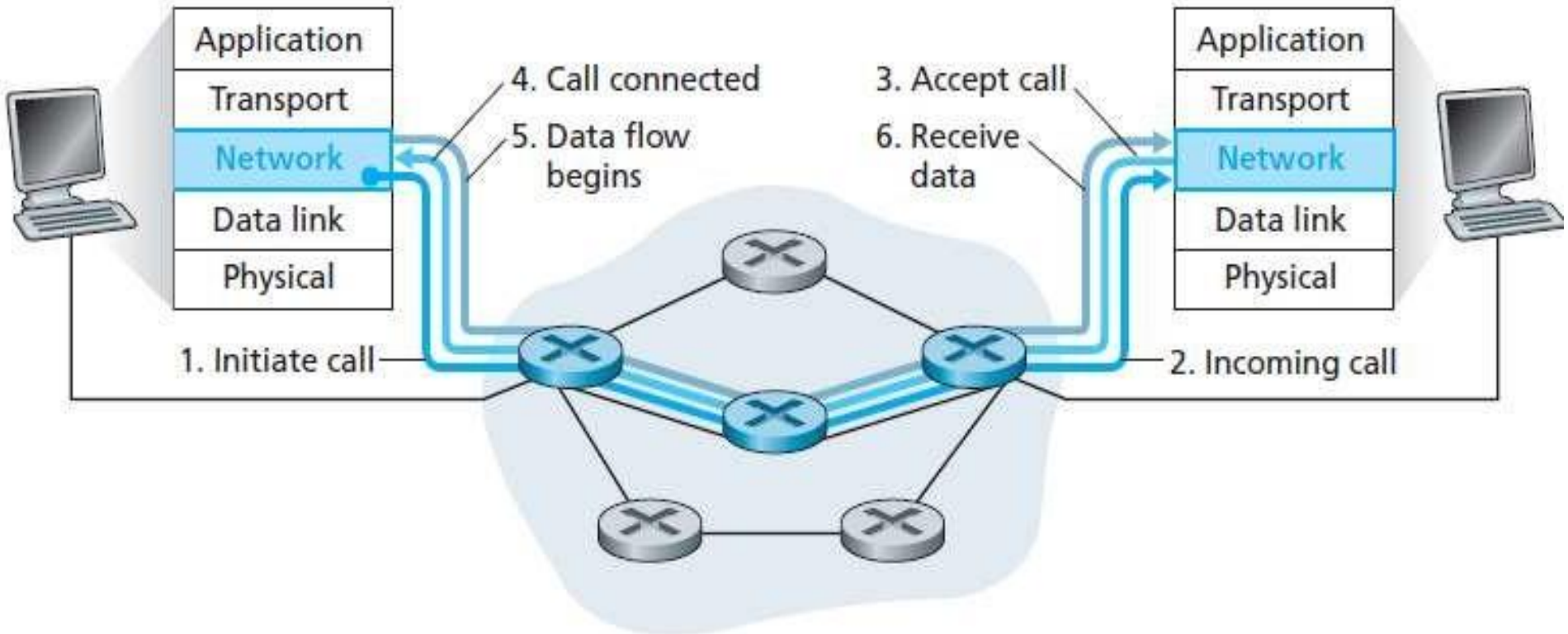


Figure 4.4 ♦ Virtual-circuit setup

Virtual circuit Phases

- There are three identifiable phases in a virtual circuit:

1. *VC setup*
2. *Data transfer.*
3. *VC teardown.*

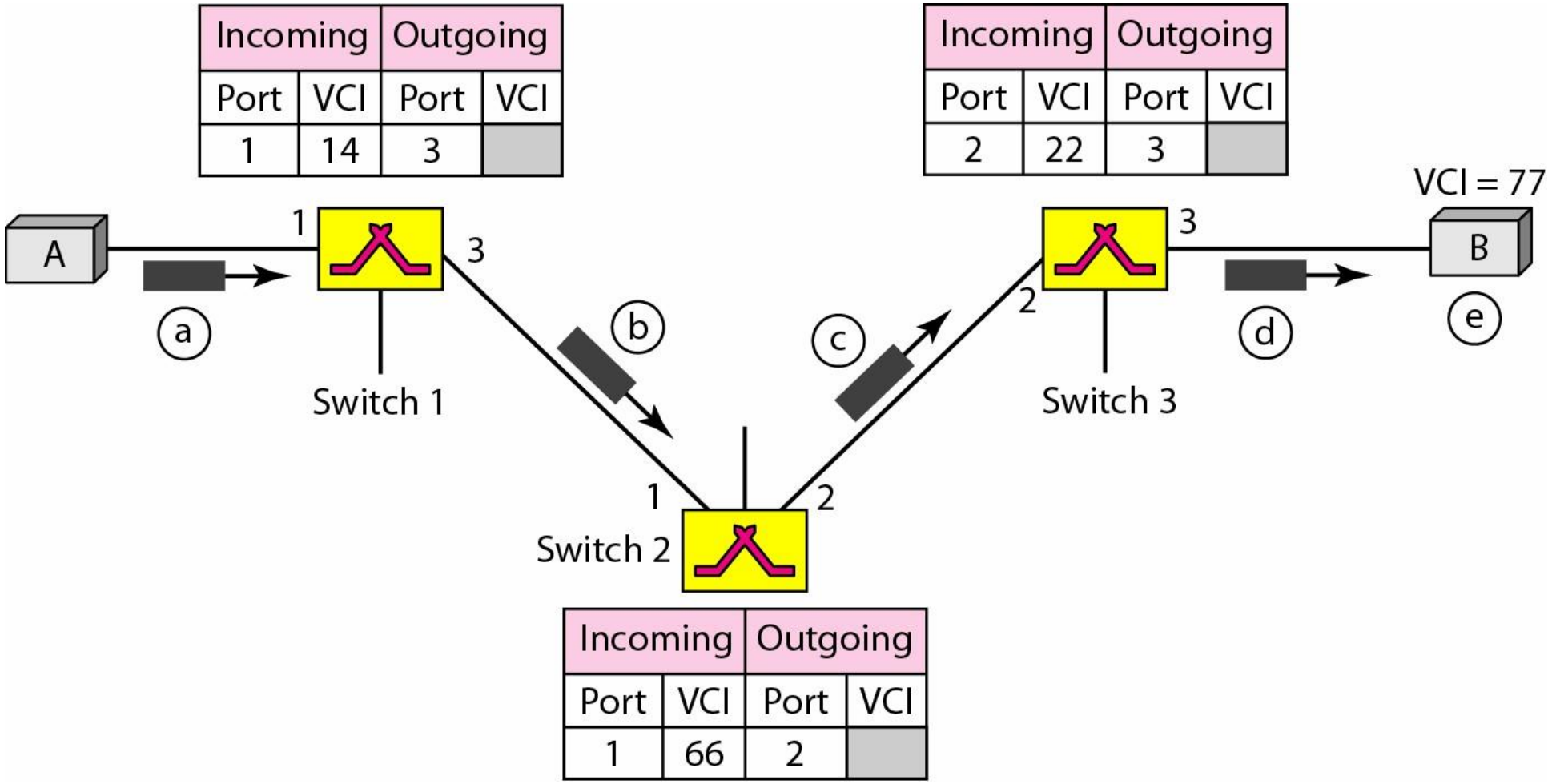
Virtual setup

- *VC setup. During the setup phase, the sending transport layer contacts the network* layer, specifies the receiver's address, and waits for the network to set up the VC.

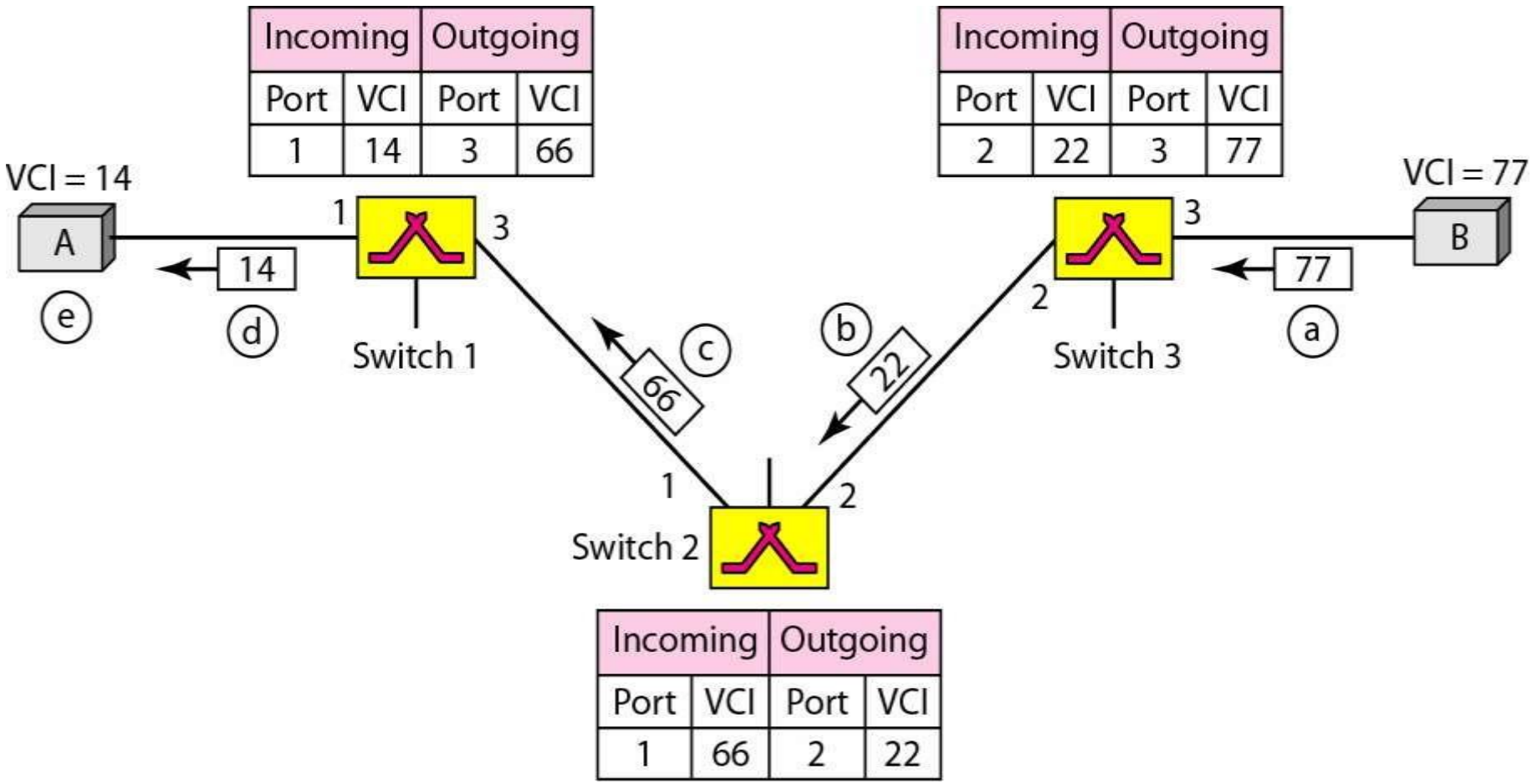
- The network layer determines the path between sender and receiver, that is, the series of links and routers through which all packets of the VC will travel.

- The network layer also determines the VC number for each link along the path.
- Finally, the network layer adds an entry in the forwarding table in each router

Setup request in a virtual circuit



Acknowledgment Phase



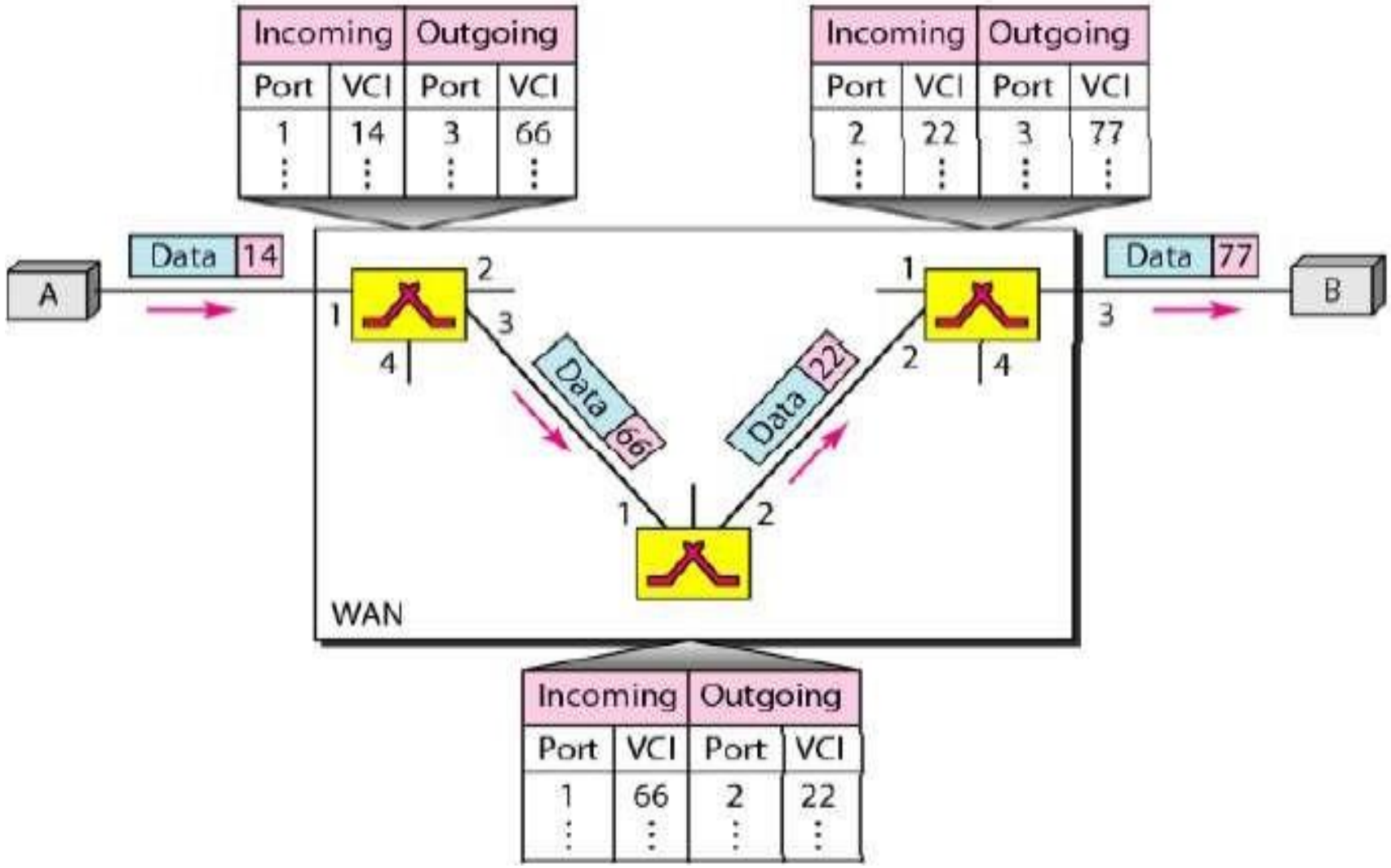
VC Data transfer.

- *Data transfer. once the VC has been established, packets* can begin to flow along the VC.

VCI transfer is used to fix the route of packet transfer

VC (Virtual Circuit) data transfer refers to a method of communication in which a logical, pre-established path (a "virtual circuit") is created between two endpoints before data transfer begins. This path remains fixed throughout the duration of the communication session, providing the appearance of a dedicated physical circuit, even though the underlying network infrastructure is shared by multiple communications.

VC Data Transfer



VC teardown.

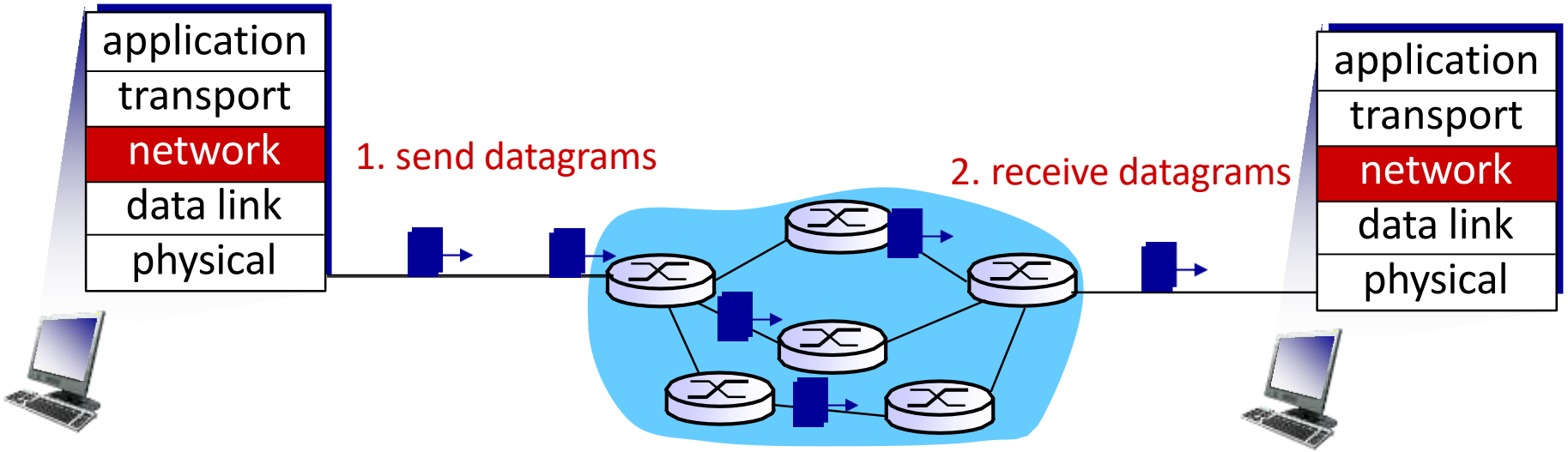
- *VC teardown. This is initiated when the sender (or receiver) informs the network* layer of its desire to terminate the VC.

- The network layer will then typically inform the end system on the other side of the network of the call termination and update the forwarding tables in each of the packet routers on the path to indicate that the VC no longer exists.

VC (Virtual Circuit) teardown is the process of terminating or dismantling a virtual circuit that was previously established for communication between two endpoints. In data communication, after the data transfer is complete, the virtual circuit no longer needs to remain active, so the resources (such as bandwidth and routing paths) associated with it are released back to the network for other potential communications.

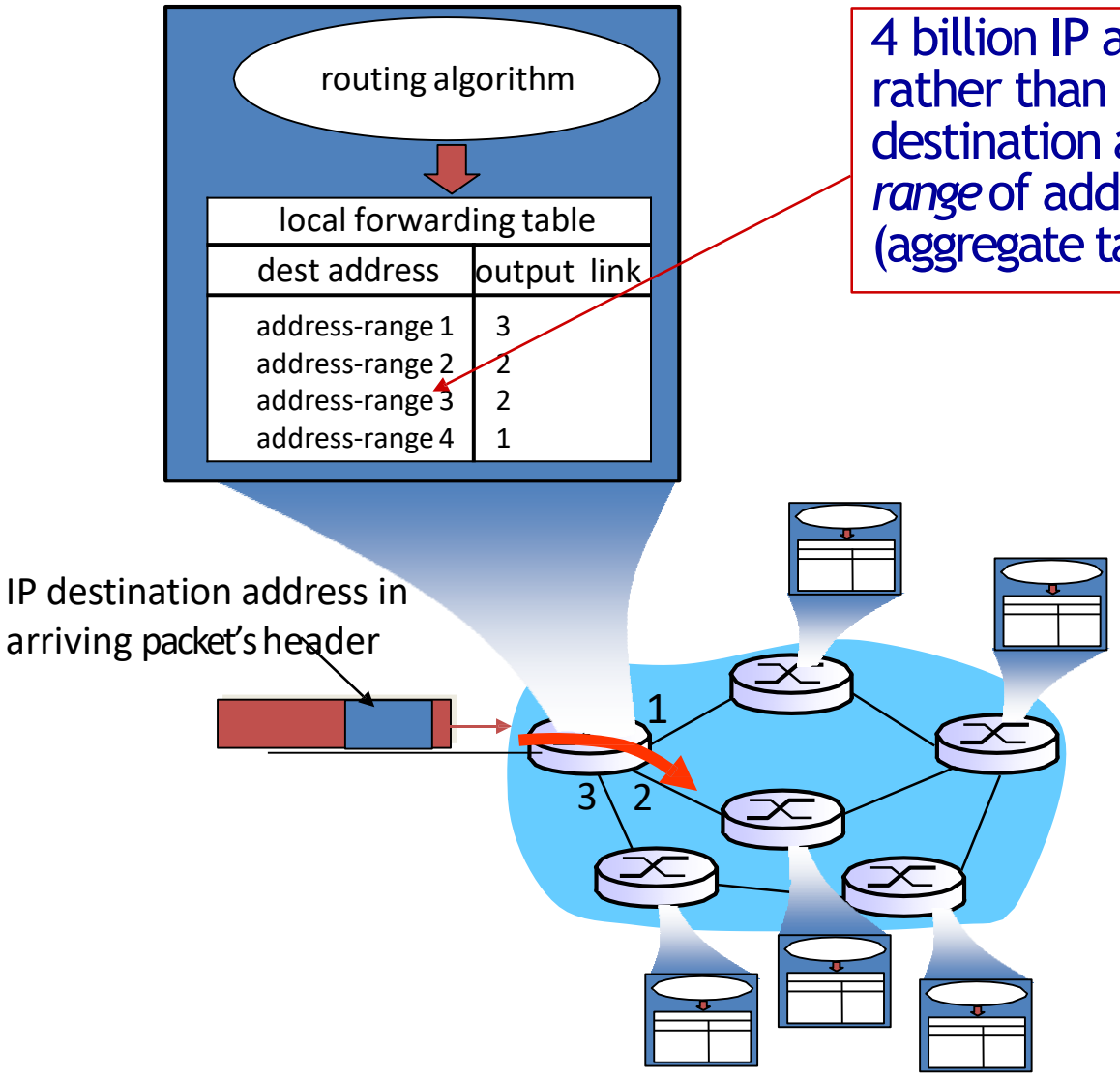
Datagram networks

- 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



Router do not have information about end-to-end connection. It has no idea whether a server is down or not

Datagram forwarding table



Datagram forwarding table

Destination Address Range	Link Interface
11001000 00010111 00010000 00000000 through 11001000 00010111 00010111 11111111	0
11001000 00010111 00011000 00000000 through 11001000 00010111 00011000 11111111	1
11001000 00010111 00011001 00000000 through 11001000 00010111 00011111 11111111	2
otherwise	3

11001000 00010111 00010110 10100001

11001000 00010111 00011000 10101010

Datagram forwarding table

- With this style of forwarding table, the router matches a **prefix of the packet's destination** address with the entries in the table; if there's a match, the router forwards the packet to a link associated with the match.

- For example, suppose the packet's destination address is 11001000 00010111 00010110 10100001; because the 21-bit prefix of this address matches the first entry in the table, the router forwards the packet to link interface 0.

- If a prefix doesn't match any of the first three entries, then the router forwards the packet to interface 3.

Datagram forwarding table

- it is possible for a destination address to match more than one entry. For example, the first 24 bits of the address 11001000 00010111 00011000 10101010 match the second entry in the table, and the first 21 bits of the address match the third entry in the table.

- When there are multiple matches, the router uses the **longest prefix matching rule; that** is, it finds the longest matching entry in the table and forwards the packet to the link interface associated with the longest prefix match.

Longest prefix matching

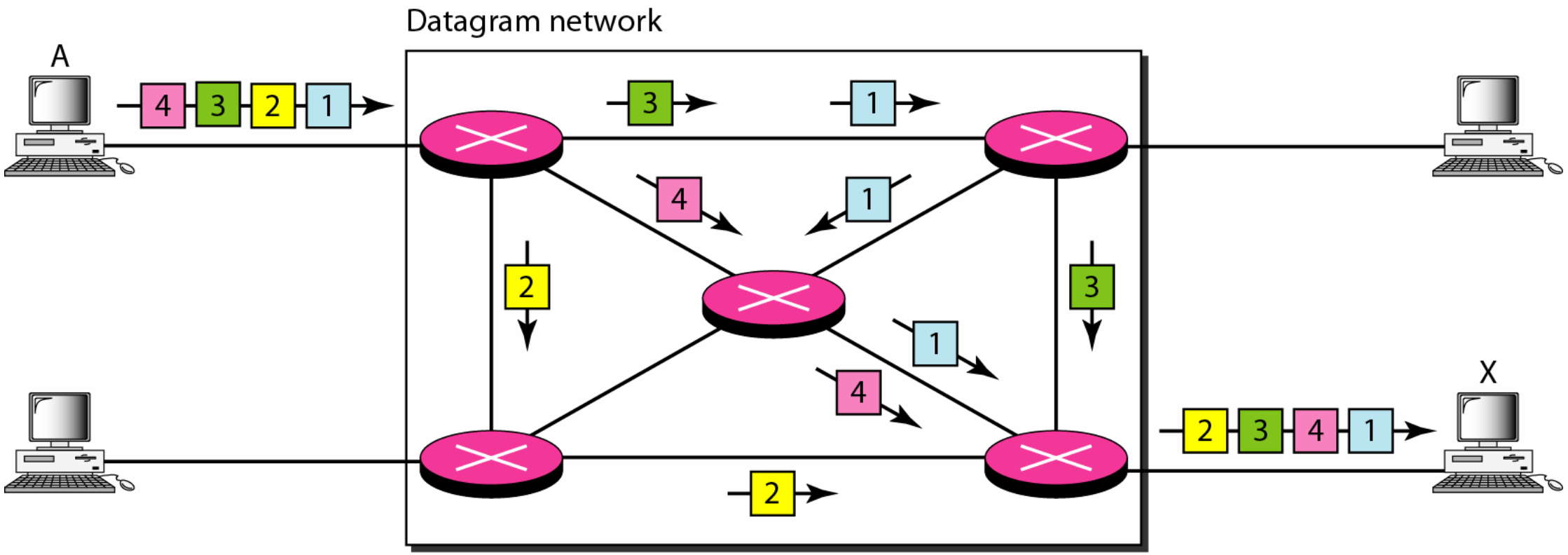
longest prefix matching

when looking for forwarding table entry for given destination address, use *longest* address prefix that matches destination address.

Datagram

- Each packet treated independently of all others
- Packets can take any practical route
- Datagram switching is done at network layer
- These switches are called routers.
- Packets may arrive out of order
- Packets may go missing
- Up to receiver to re-order packets and recover from missing packets
- The connecting switches are not keeping information about connection state hence it is also connectionless networks
- No setup and teardown phases.

Datagram network



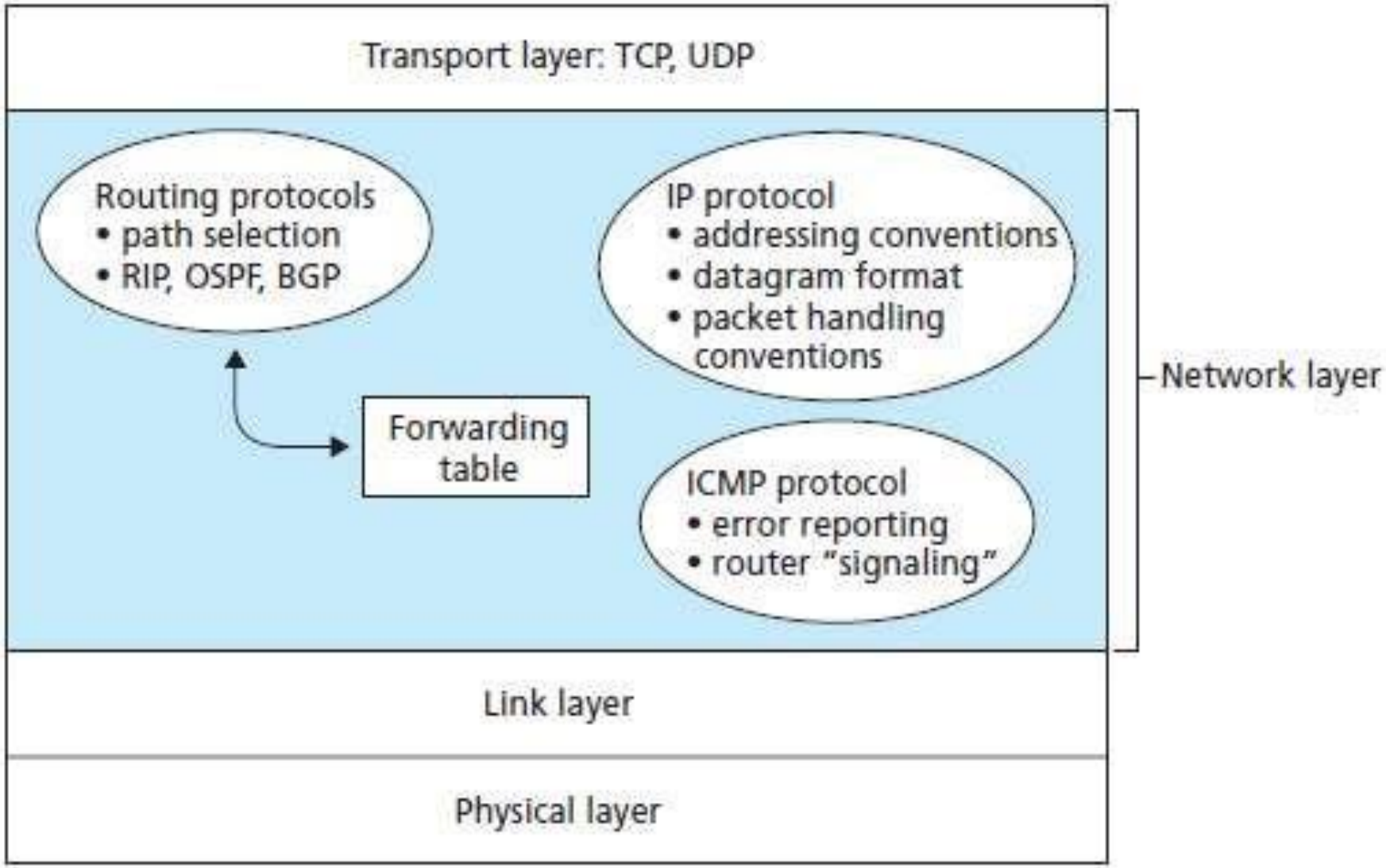
Router

- A router is a networking device that forwards data packets between computer networks.
- Routers perform the traffic directing functions on the Internet.
- A packet is typically forwarded from one router to another router through the networks that constitute an internetwork (e.g. the Internet) until it reaches its destination node.
- A router is connected to two or more data lines from different IP networks.
- When a data packet comes in on one of the lines, the router reads the network address information in the packet header to determine the ultimate destination. Then, using information in its routing table or routing policy, it directs the packet to the next network on its journey.

The Internet Protocol (IP)

- **Forwarding and Addressing in the Internet**
- Internet addressing and forwarding are important components of the Internet Protocol (IP). There are two versions of IP in use today.
- IP protocol version 4, which is usually referred to simply as IPv4
- IP version 6
- **Internet's network layer has three major components.**
- The first component is the IP protocol.
- The second major component is the routing component, which determines the path a datagram follows from source to destination.
- The final component of the network layer is a facility to report errors in datagrams and respond to requests for certain network-layer information.

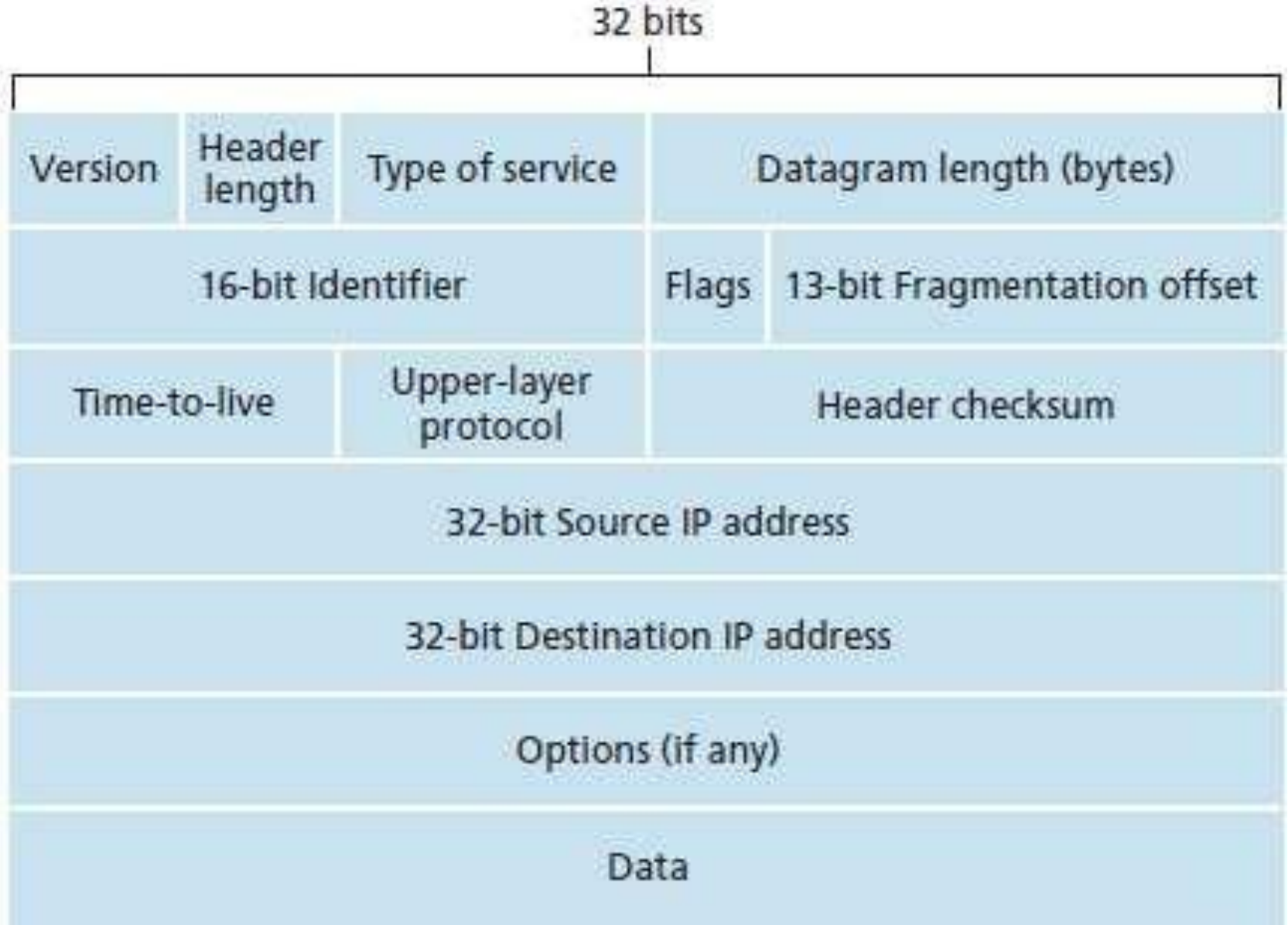
Inside the Internet's network layer



Datagram Format

- A network-layer packet is referred to as a *datagram.*
- The datagram plays a central role in the Internet

IPv4 datagram format



- *Version number.* These 4 bits specify the IP protocol version of the datagram.By looking at the version number, the router can determine how to interpret the remainder of the IP datagram.
- *Header length. T*hese 4 bits are needed to determine where in the IP datagram the data actually begins. Most IP datagrams, the typical IP datagram has a 20-byte header.
- *Type of service.* The type of service (TOS) bits were included in the IPv4 header to allow different types of IP datagrams to be distinguished from each other. For example, it might be useful to distinguish real-time datagrams (such as those used by an IP telephony application) from non-real-time traffic (for example, FTP).
- *Datagram length.* This is the total length of the IP datagram (header plus data), measured in bytes. Since this field is 16 bits long, the theoretical maximum size of the IP datagram is 65,535 bytes. However, datagrams are rarely larger than 1,500 bytes.

1500 bytes is ideal datagram length

- *Identifier, flags, fragmentation offset.* These three fields have to do with so-called IP fragmentation
- *Time-to-live.* The time-to-live (TTL) field is included to ensure that datagrams do not circulate forever in the network. This field is decremented by one each time the datagram is processed by a router. If the TTL field reaches 0, the datagram must be dropped.
- *Protocol.* This field is used only when an IP datagram reaches its final destination. The value of this field indicates the specific transport-layer protocol.
- *Header checksum.* The header checksum aids a router in detecting bit errors in a
- received IP datagram. The header checksum is computed by treating each 2 bytes in the header as a number and summing these numbers using 1s complement arithmetic.

Total size = header + data

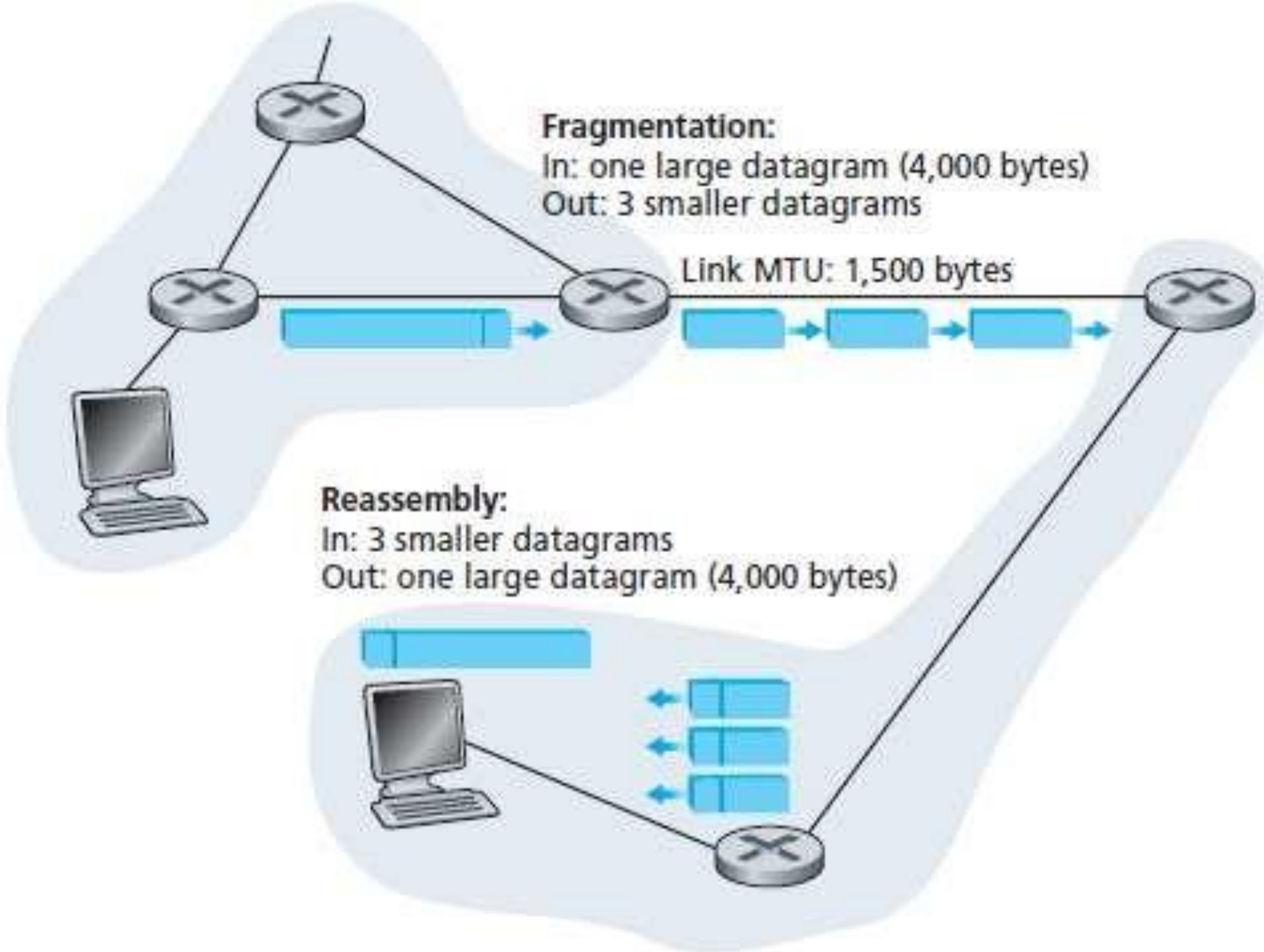
- *Source and destination IP addresses.* When a source creates a datagram, it inserts its IP address into the source IP address field and inserts the address of the ultimate destination into the destination IP address field.
- *Options.* The options fields allow an IP header to be extended.
- *Data (payload). T*he data field of the IP datagram contains the transport-layer segment (TCP or UDP) to be delivered to the destination.

- If the datagram carries a TCP segment, then each (nonfragmented) datagram carries a total of 40 bytes of header (20 bytes of IP header plus 20 bytes of TCP header) along with the application-layer message.

IP Datagram Fragmentation

- Some protocols can carry big datagrams, whereas other protocols can carry only little packets. For example, Ethernet frames can carry up to 1,500 bytes of data, whereas frames for some wide-area links can carry no more than 576 bytes.
- The maximum amount of data that a link-layer frame can carry is called the maximum transmission unit (MTU).
- A router that interconnects several links, each running different link-layer protocols with different MTUs.
- Suppose it receive an IP datagram from one link. The forwarding table determine the outgoing link, and this outgoing link has an MTU that is smaller than the length of the IP datagram.

- The solution is to fragment the data in the IP datagram into two or more smaller IP datagrams, encapsulate each of these smaller IP datagrams in a separate link-layer frame; and send these frames over the outgoing link.
- Each of these smaller datagrams is referred to as a **fragment**.
- Fragments need to be reassembled before they reach the transport layer at the destination.
- The designers of IPv4 decided to put the job of datagram reassembly in the end systems rather than in network routers.
- When a destination host receives a series of datagrams from the same source, it needs to determine whether any of these datagrams are fragments of some original, larger datagram.
- If some datagrams are fragments, it must further determine when it has received the last fragment and how the fragments it has received should be pieced back together to form the original datagram.
- To allow the destination host to perform these reassembly tasks, the designers of IP (version 4) put *identification, flag,* and *fragmentation offset* fields in the IP datagram header.


Fragmentation:
In: one large datagram (4,000 bytes)
Out: 3 smaller datagrams
Link MTU: 1,500 bytes
Reassembly:
In: 3 smaller datagrams
Out: one large datagram (4,000 bytes)

- At the destination, the payload of the datagram is passed to the transport layer only after the IP layer has fully reconstructed the original IP datagram.
- If one or more of the fragments does not arrive at the destination, the incomplete datagram is discarded and not passed to the transport layer.

IP fragments

Fragment	Bytes	ID	Offset	Flag
1st fragment	1,480 bytes in the data field of the IP datagram	identification = 777	offset = 0 (meaning the data should be inserted beginning at byte 0)	flag = 1 (meaning there is more)
2nd fragment	1,480 bytes of data	identification = 777	offset = 185 (meaning the data should be inserted beginning at byte 1,480. Note that 185 · 8 = 1,480)	flag = 1 (meaning there is more)
3rd fragment	1,020 bytes (= 3,980–1,480–1,480) of data	identification = 777	offset = 370 (meaning the data should be inserted beginning at byte 2,960. Note that 370 · 8 = 2,960)	flag = 0 (meaning this is the last fragment)

- But fragmentation also has its costs.
- First, it complicates routers and end systems, which need to be designed to accommodate datagram fragmentation and reassembly.
- Second, fragmentation can be used to create DoS attacks, whereby the attacker sends a series of unexpected fragments.
- A classic example is the Jolt2 attack, where the attacker sends a stream of small fragments to the target host, none of which has an offset of zero.
- The target can collapse as it attempts to rebuild datagrams out of the degenerate packets.
- Another class of exploits sends overlapping IP fragments, that is, fragments whose offset values are set so that the fragments do not align properly.
- Vulnerable operating systems, not knowing what to do with overlapping fragments, can crash .
- A new version of the IP protocol, IPv6, does away with fragmentation altogether, thereby streamlining IP packet processing and making IP less vulnerable to attack.

IPv4 Addressing

- When IP in the host wants to send a datagram, it does so over the link.
- The boundary between the host and the physical link is called an **interface**.
- A router thus has multiple interfaces, one for each of its links.
- Because every host and router is capable of sending and receiving IP datagrams, IP requires each host and router interface to have its own IP address.
- Thus, an IP address is technically associated with an interface.
- Each IP address is 32 bits long (equivalently, 4 bytes), and there are thus a total of 2^{32} possible IP addresses.

IPV4 Address

The identifier used in the IP layer of the TCP/IP protocol suite to identify each device connected to the Internet is called **the Internet address or IP address.**

An IPv4 address is a 32-bit address that uniquely and universally defines the connection of a host or a router to the Internet;

An IP address is the address of the interface.

IPV4 Address Space

An IPv4 address is 32 bits long.

The address space of IPv4 is 2^{32} or 4,294,967,296.

The IPv4 addresses are unique and universal.

IP Address Notation

- There are three common notations to show an IPv4 address:

1. binary notation (base 2)
2. dotted-decimal notation (base 256)
3. hexadecimal notation (base 16).

The most prevalent, however, is base 256.

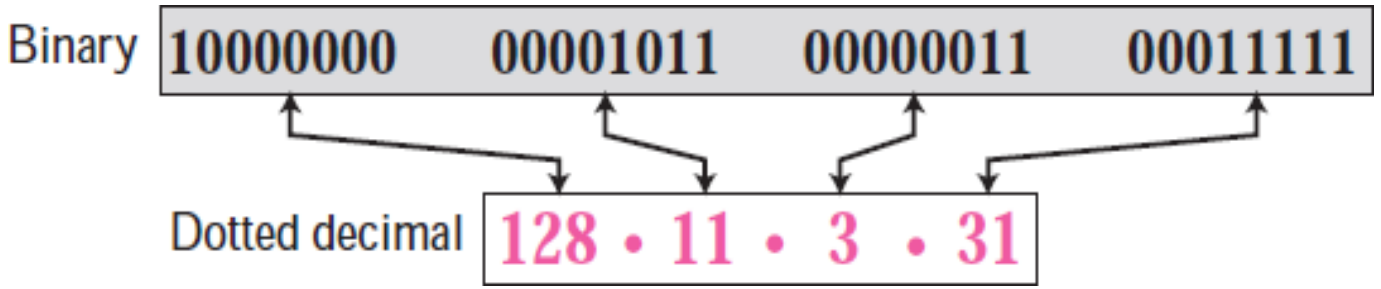
IP Address Notation: **binary notation**

- In **binary notation**, **an IPv4 address is displayed as 32 bits.**
- To make the address more readable, one or more spaces is usually inserted between each octet (8 bits).
- Each octet is often referred to as a byte.
- **01110101 10010101 00011101 11101010**

Dotted-Decimal Notation: Base 256

- To make the IPv4 address more compact and easier to read, an IPv4 address is usually written in decimal form with a decimal point (dot) separating the bytes.

- This format is referred to as **dotted-decimal notation.**



Hexadecimal notation

- We sometimes see an IPv4 address in **hexadecimal notation.**
- **Each hexadecimal digit** is equivalent to four bits.

- This means that a 32-bit address has 8 hexadecimal digits.
- This notation is often used in network programming.

- **1000 0001 0000 1011 0000 1011 1110 1111**
- 0X810B0BEF or 810B0BEF16

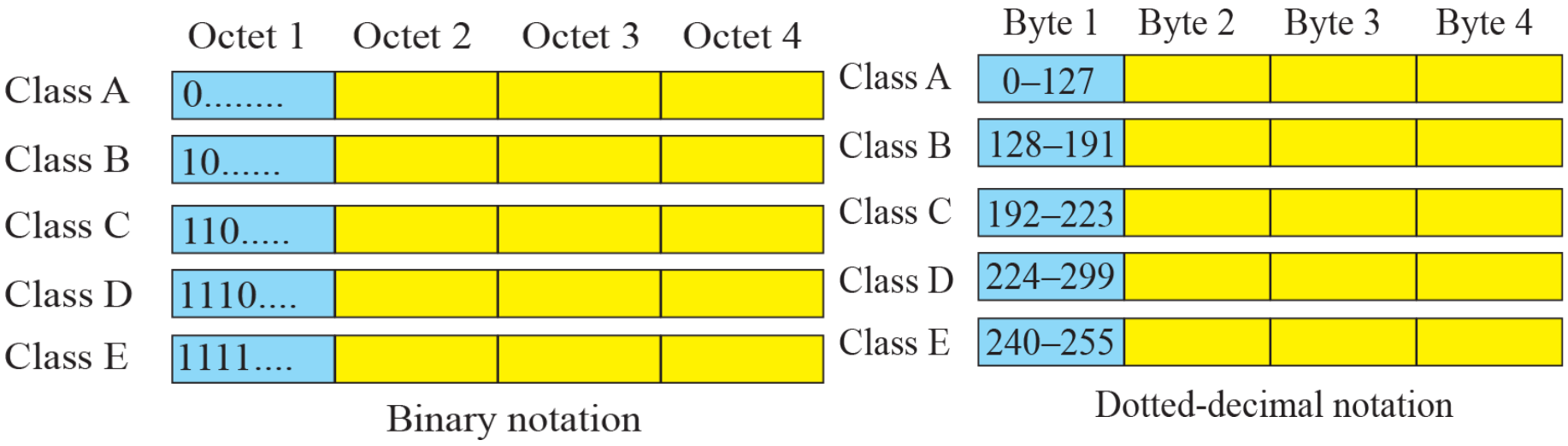
Classful addressing.

- In classful addressing, the IP address space is divided into five **classes: A, B, C, D, and E.**

- **Each class occupies some part of the whole address space**

Finding the class of address

We can find the class of an address when the address is given either in binary or dotted decimal notation. In the binary notation, the first few bits can immediately tell us the class of the address; in the dotted-decimal notation, the value of the first byte can give the class of an address



Finding Class of an IP Address

Find the class of each address:

a. 00000001 00001011 00001011 11101111
b. 11000001 10000011 00011011 11111111
c. 10100111 11011011 10001011 01101111
d. 11110011 10011011 11111011 00001111

Finding Class of an IP Address

Find the class of each address:

a. 00000001 00001011 00001011 11101111
b. 11000001 10000011 00011011 11111111
c. 10100111 11011011 10001011 01101111
d. 11110011 10011011 11111011 00001111

a. The first bit is 0. This is a class A address.
b. The first 2 bits are 1; the third bit is 0. This is a class C address.
c. The first bit is 1; the second bit is 0. This is a class B address.
d. The first 4 bits are 1s. This is a class E address.

Finding Class of an IP Address

Find the class of each address:

a. 227.12.14.87
b. 193.14.56.22
c. 14.23.120.8
d. 252.5.15.111

Finding Class of an IP Address

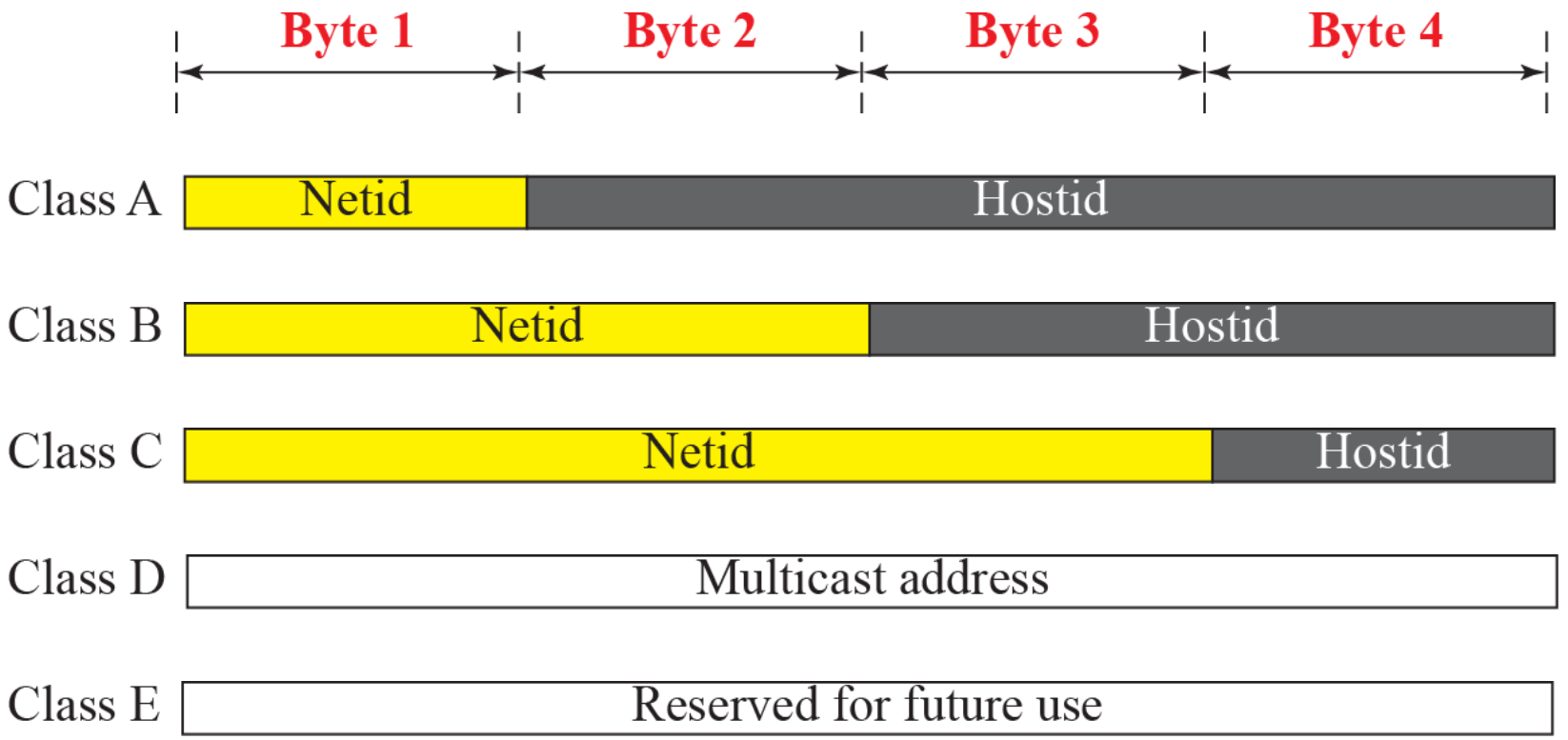
Find the class of each address:

a. 227.12.14.87
b. 193.14.56.22
c. 14.23.120.8
d. 252.5.15.111

Solution

a. The first byte is 227 (between 224 and 239); the class is D.
b. The first byte is 193 (between 192 and 223); the class is C.
c. The first byte is 14 (between 0 and 127); the class is A.
d. The first byte is 252 (between 240 and 255); the class is E.

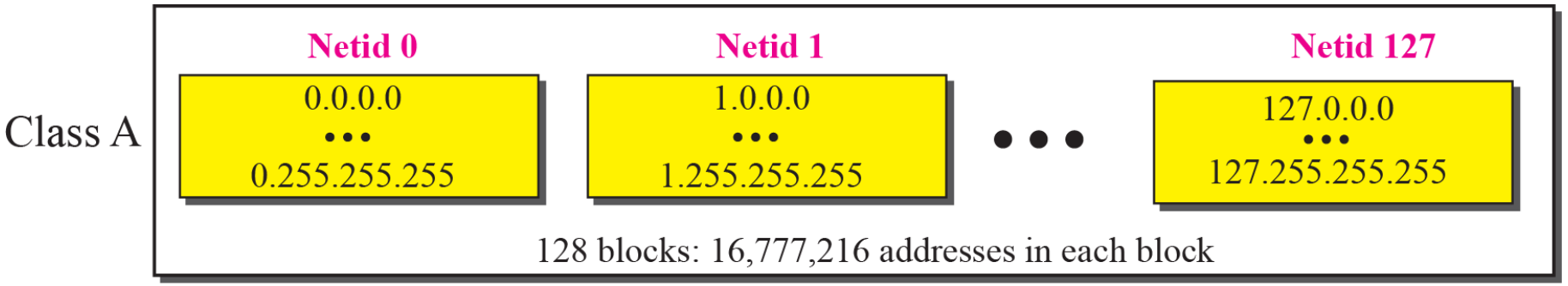
Netid and hostid



Classful Addressing :Classes and Blocks

- One problem with classful addressing is that each class is divided into a fixed number of blocks with each block having a fixed size.

Blocks in Class An EXAMPLE



Millions of class A addresses are wasted.

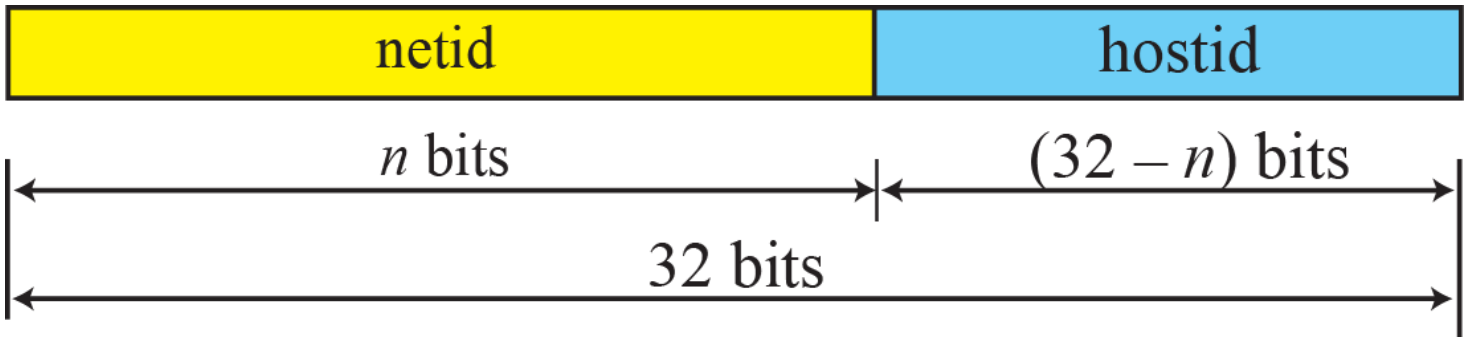
Class D addresses are made of one block, used for multicasting.

The only block of class E addresses was reserved for future purposes.

The range of addresses allocated to an organization in classful addressing was a block of addresses in Class A, B, or C.

Two-level addressing in classful addressing

Netid : Define the network.
Hostid: define a particular host connected to that network



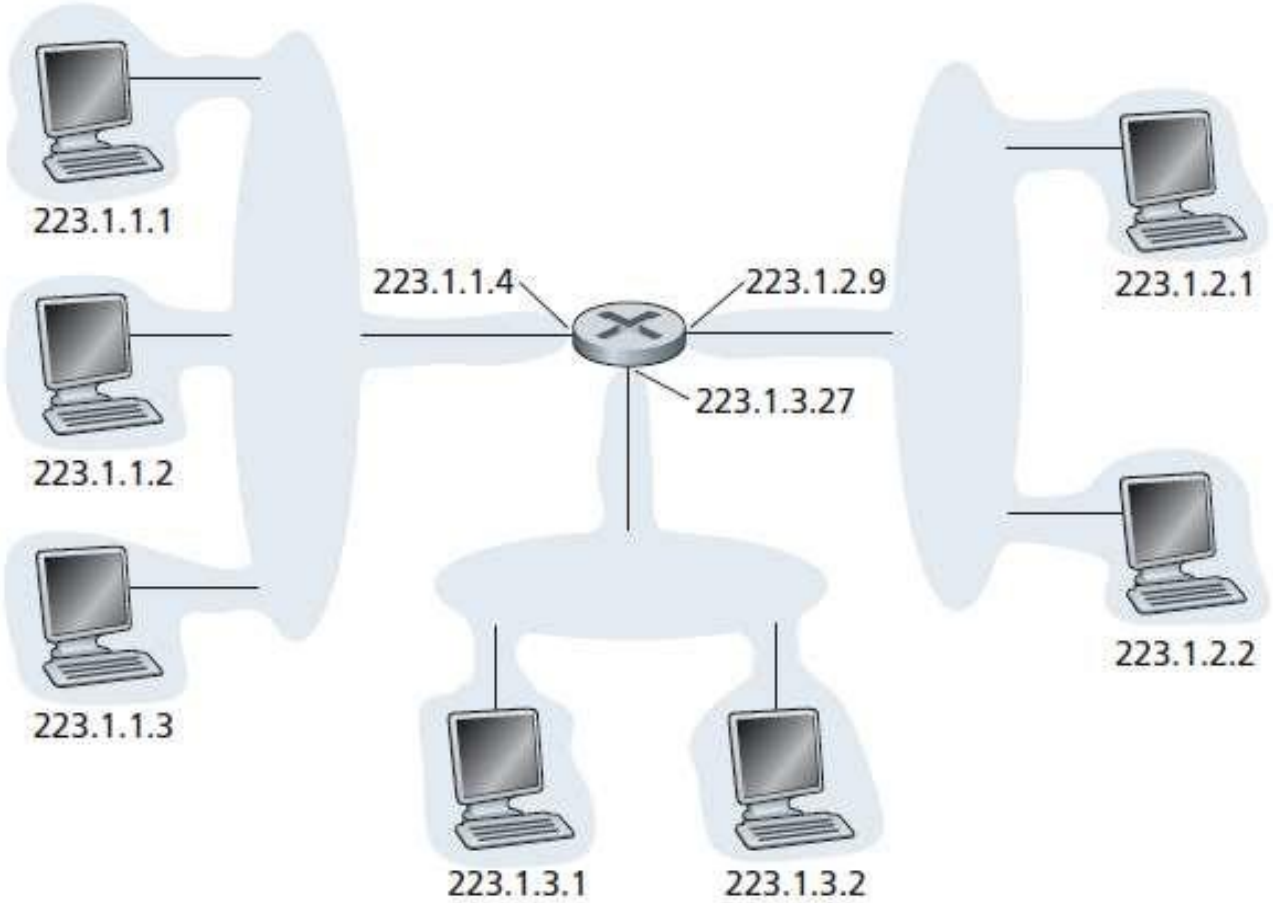
Class A: $n = 8$

Class B: $n = 16$

Class C: $n = 24$

Subnet

- A subnet is also called an *IP network* or simply a *network* in the Internet literature.



- In the figure one router (with three interfaces) is used to interconnect seven hosts.
- The three hosts in the upper-left portion of Figure and the router interface to which they are connected, all have an IP address of the form 223.1.1.xxx.
- That is, they all have the same leftmost 24 bits in their IP address.
- The four interfaces are also interconnected to each other by a network *that contains no routers*.
- This network could be interconnected by an Ethernet LAN.
- In IP terms, this network interconnecting three host interfaces and one router interface forms a **subnet.**

- IP addressing assigns an address to this subnet: 223.1.1.0/24, where the /24 notation, sometimes known as a **subnet mask**.
- Indicates that the leftmost 24 bits of the 32-bit quantity define the subnet address.
- Any additional hosts attached to the 223.1.1.0/24 subnet would be *required* to have an address of the form 223.1.1.xxx.
- *To determine the subnets, detach each interface from its host or router, creating islands of isolated networks, with interfaces terminating the end points of the isolated networks. Each of these isolated networks is called a* ***subnet****.*

Classless Interdomain Routing

- The Internet's address assignment strategy is known as **Classless Interdomain Routing** (**CIDR**—pronounced *cider*) .
- CIDR generalizes the notion of subnet addressing.
- As with subnet addressing, the 32-bit IP address is divided into two parts and again has the dotted-decimal form *a.b.c.d/x*, where *x* indicates the number of bits in the first part of the address.
- The *x* most significant bits of an address of the form *a.b.c.d/x* constitute the network portion of the IP address, and are often referred to as the **prefix** (or *network prefix*) of the address.
- An organization is typically assigned a block of contiguous addresses, that is, a range of addresses with a common prefix .
- In this case, the IP addresses of devices within the organization will share the common prefix.

Obtaining a Block of Addresses

- In order to obtain a block of IP addresses for use within an organization's subnet, a network administrator might first contact its ISP, which would provide addresses from a larger block of addresses that had already been allocated to the ISP.
- For example, the ISP may itself have been allocated the address block 200.23.16.0/20.
- The ISP, in turn, could divide its address block into eight equal-sized contiguous address blocks and give one of these address blocks out to each of up to eight organizations that are supported by this ISP.
- IP addresses are managed under the authority of the Internet Corporation for Assigned Names and Numbers (ICANN).

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

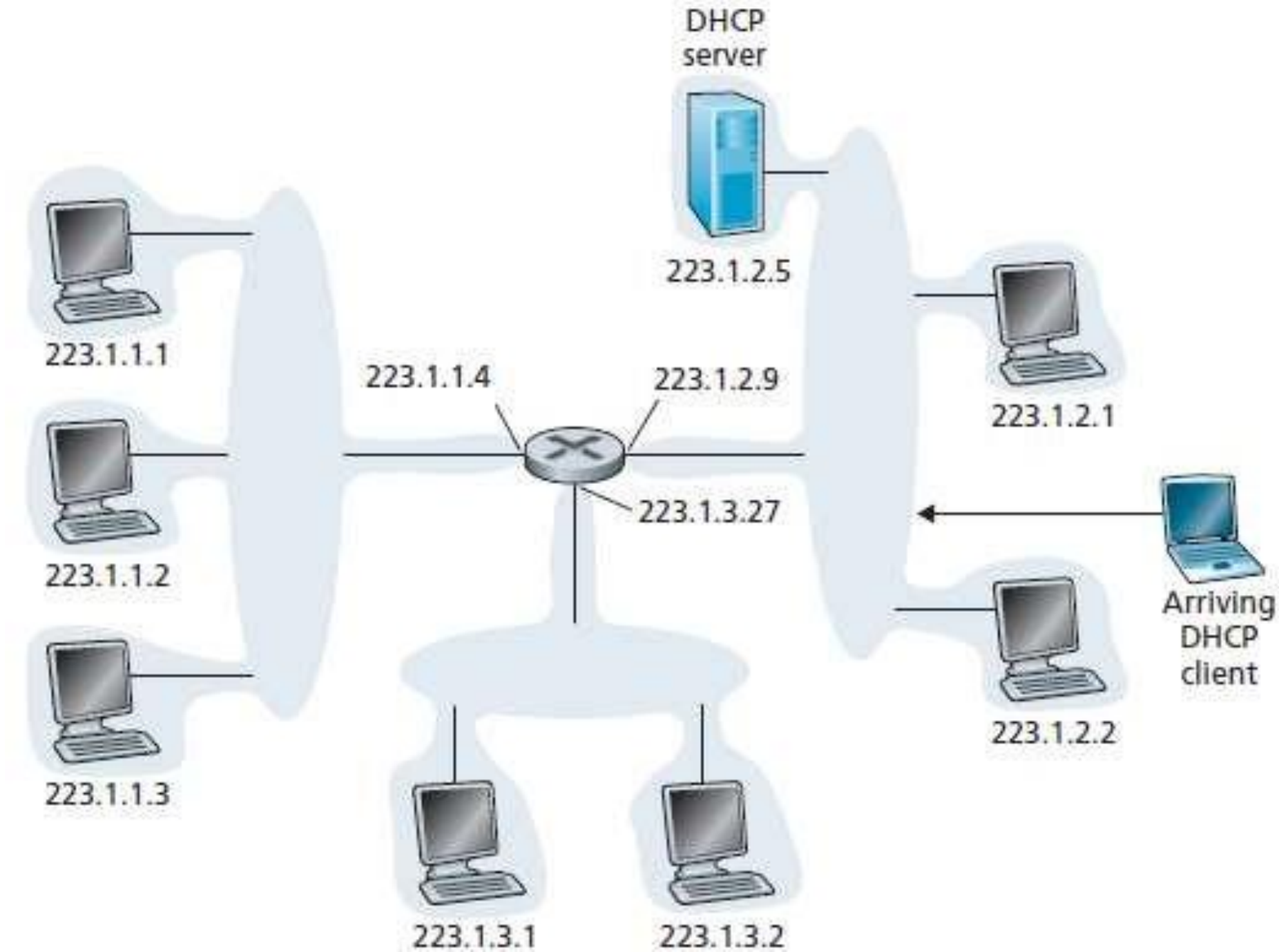
Obtaining a Host Address: the Dynamic Host Configuration Protocol

- Once an organization has obtained a block of addresses, it can assign individual IP addresses to the host and router interfaces in its organization.
- A system administrator will typically manually configure the IP addresses into the router (often remotely, with a network management tool).
- Host addresses can also be configured manually, but more often this task is now done using the **Dynamic Host Configuration Protocol (DHCP).**
- DHCP allows a host to obtain (be allocated) an IP address automatically.
- A network administrator can configure DHCP.

- Given host receives the same IP address each time it connects to the network, or a host may be assigned a **temporary IP address** that will be different each time the host connects to the network.
- In addition to host IP address assignment, DHCP also allows a host to learn additional information, such as its subnet mask, the address of its first-hop router (often called the default gateway), and the address of its local DNS server.
- Because of DHCP's ability to automate the network-related aspects of connecting a host into a network, it is often referred to as a **plug-and-play protocol**.

- This capability makes it *very* attractive to the network administrator who would otherwise have to perform these tasks manually!
- DHCP is also enjoying widespread use in residential Internet access networks and in wireless LANs, where hosts join and leave the network frequently.
- Consider, for example, the student who carries a laptop from a dormitory room to a library to a classroom. It is likely that in each location, the student will be connecting into a new subnet and hence will need a new IP address at each location.
- DHCP is ideally suited to this situation, as there are many users coming and going, and addresses are needed for only a limited amount of time.
- DHCP is similarly useful in residential ISP access networks.
- Consider, for example, a residential ISP that has 2,000 customers, but no more than 400 customers are ever online at the same time. In this case, rather than needing a block of 2,048 addresses, a DHCP server that assigns addresses dynamically needs only a block of 512 addresses (for example, a block of the form a.b.c.d/23).
- As the hosts join and leave, the DHCP server needs to update its list of available IP addresses.
- Each time a host joins, the DHCP server allocates an arbitrary address from its current pool of available addresses; each time a host leaves, its address is returned to the pool.

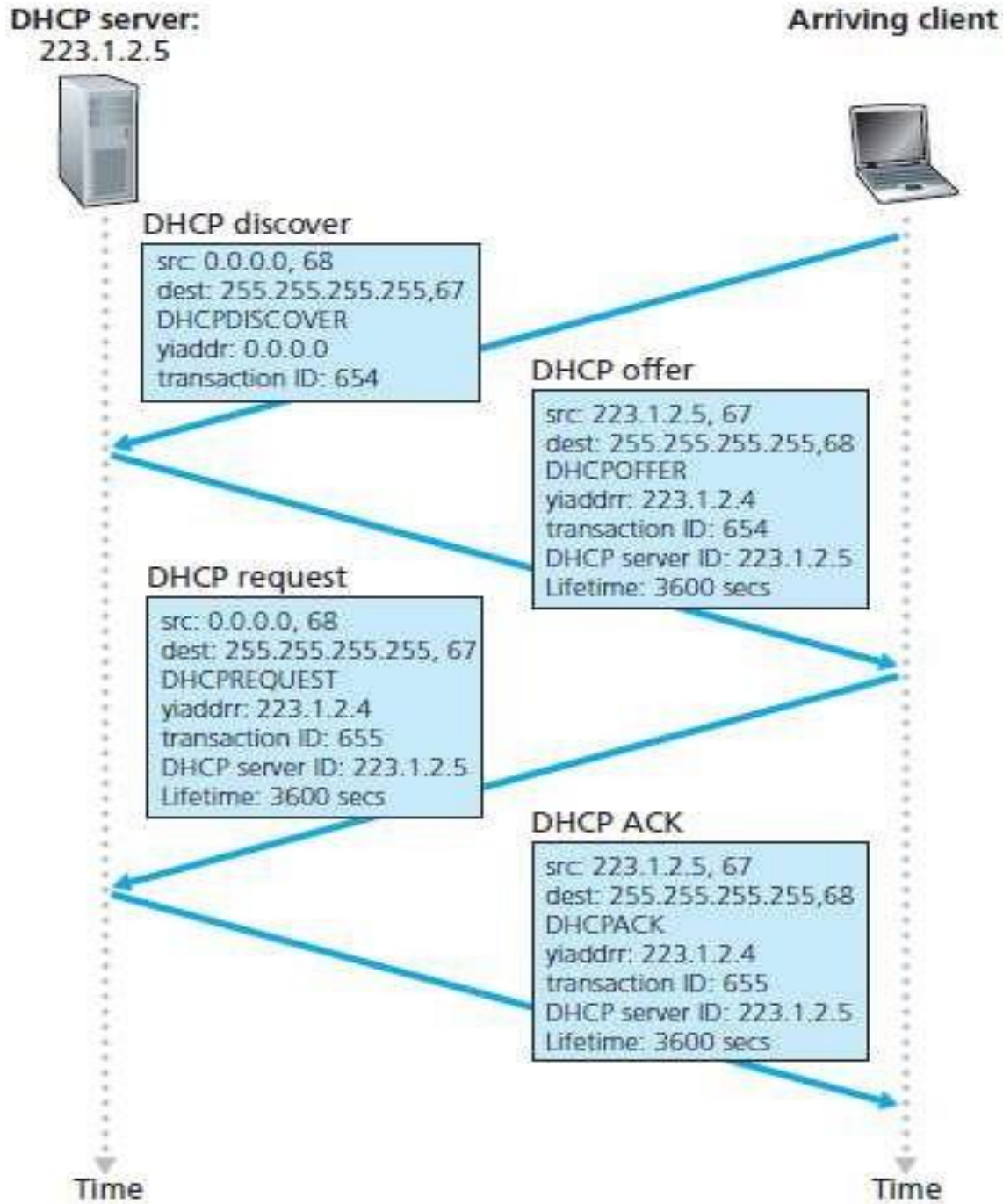
- DHCP is a client-server protocol. A client is typically a newly arriving host wanting to obtain network configuration information, including an IP address for itself.
- In the simplest case, each subnet will have a DHCP server.
- If no server is present on the subnet, a DHCP relay agent (typically a router) that knows the address of a DHCP server for that network is needed.
- Figure shows a DHCP server attached to subnet 223.1.2/24, with the router serving as the relay agent for arriving clients attached to subnets 223.1.1/24 and 223.1.3/24.


DHCP
server
223.1.2.5
223.1.1.1
223.1.1.4
223.1.2.9
223.1.2.1
223.1.1.2
223.1.3.27
Arriving
DHCP
client
223.1.2.2
223.1.1.3
223.1.3.1
223.1.3.2

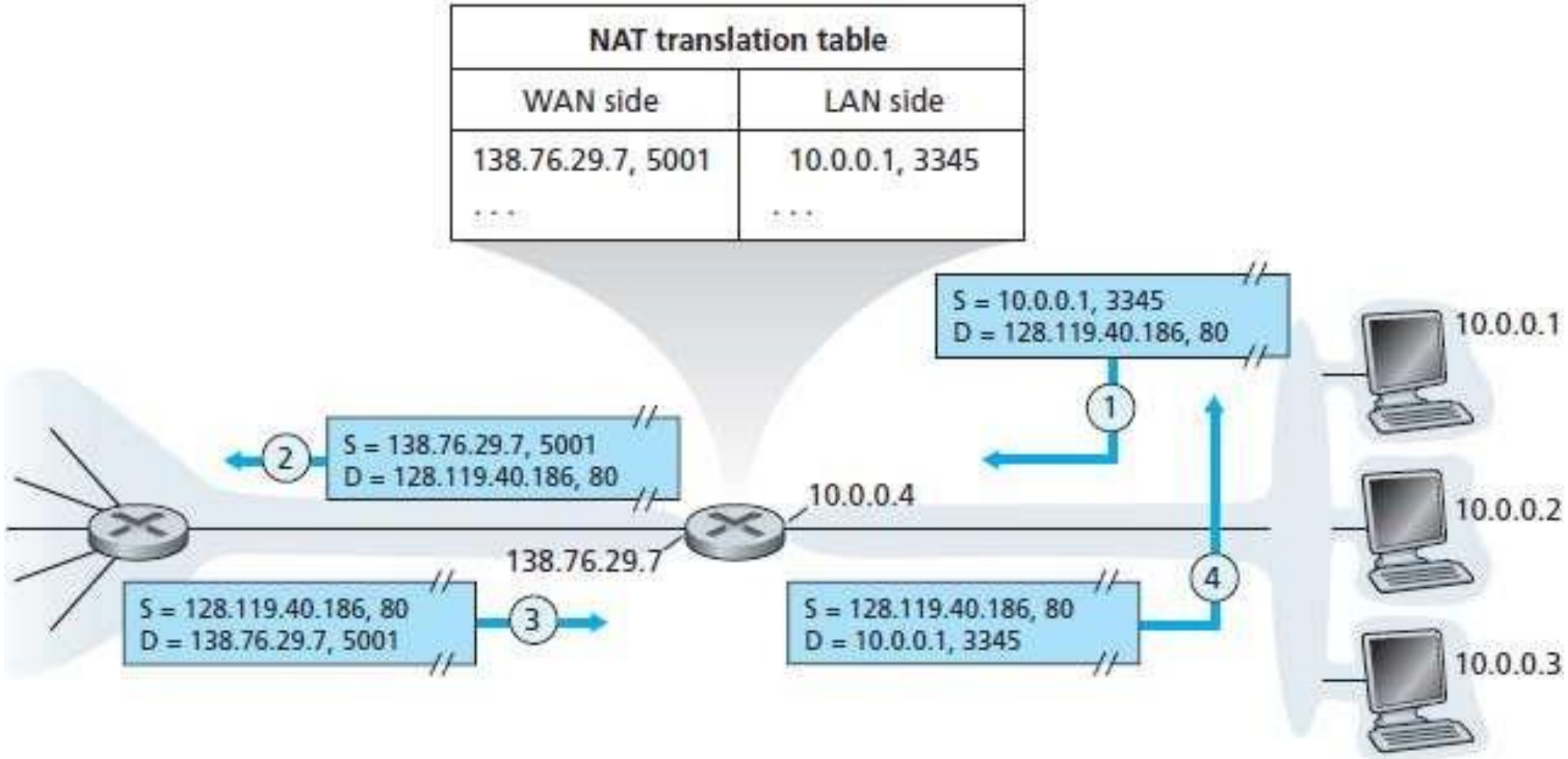
- For a newly arriving host, the DHCP protocol is a four-step process:
- *DHCP server discovery*. The first task of a newly arriving host is to find a DHCP server with which to interact. This is done using a **DHCP discover message**, which a client sends within a UDP packet to port 67.
- The UDP packet is encapsulated in an IP datagram.
- The DHCP client creates an IP datagram containing its DHCP discover message along with the broadcast destination IP address of 255.255.255.255 and a “this host” source IP address of 0.0.0.0.
- The DHCP client passes the IP datagram to the link layer, which then broadcasts this frame to all nodes attached to the subnet.

- *DHCP server offer(s).* A DHCP server receiving a DHCP discover message responds to the client with a **DHCP offer message** that is broadcast to all nodes on the subnet, again using the IP broadcast address of 255.255.255.255.
- Each server offer message contains the transaction ID of the received discover message, the proposed IP address for the client, the network mask, and an IP **address lease time**—the amount of time for which the IP address will be valid.

- *DHCP request.* The newly arriving client will choose from among one or more server offers and respond to its selected offer with a **DHCP request message**, echoing back the configuration parameters.
- *DHCP ACK*. The server responds to the DHCP request message with a **DHCP ACK message**, confirming the requested parameters.
- Once the client receives the DHCP ACK, the interaction is complete and the client can use the DHCP-allocated IP address for the lease duration.


DHCP server:
223.1.2.5
Arriving client
DHCP discover
src: 0.0.0.0, 68
dest: 255.255.255.255,67
DHCPDISCOVER
yiaddr: 0.0.0.0
transaction ID: 654
DHCP offer
src: 223.1.2.5, 67
dest: 255.255.255.255,68
DHCPOFFER
yiaddrr: 223.1.2.4
transaction ID: 654
DHCP server ID: 223.1.2.5
Lifetime: 3600 secs
DHCP request
src: 0.0.0.0, 68
dest: 255.255.255.255, 67
DHCPREQUEST
yiaddrr: 223.1.2.4
transaction ID: 655
DHCP server ID: 223.1.2.5
Lifetime: 3600 secs
DHCP ACK
src: 223.1.2.5, 67
dest: 255.255.255.255,68
DHCPACK
yiaddrr: 223.1.2.4
transaction ID: 655
DHCP server ID: 223.1.2.5
Lifetime: 3600 secs
Time
Time

Network Address Translation (NAT)



- Devices within a given home network can send packets to each other using 10.0.0.0/24 addressing.
- However, packets forwarded *beyond* the home network into the larger global Internet clearly cannot use these addresses.
- The NAT-enabled router does not *look* like a router to the outside world. Instead
- the NAT router behaves to the outside world as a *single* device with a *single* IP address.
- All traffic leaving the home router for the larger Internet has a source IP address of 138.76.29.7, and all traffic entering the home router must have a destination address of 138.76.29.7.
- In essence, the NAT-enabled router is hiding the details of the home network from the outside world.

- use a **NAT translation table** at the NAT router, and to include port numbers as well as IP addresses in the table entries.
- Suppose a user sitting in a home network behind host 10.0.0.1 requests a Web page on some Web server (port 80) with IP address 128.119.40.186.
- The host 10.0.0.1 assigns the (arbitrary) source port number 3345 and sends the datagram into the LAN.
- The NAT router receives the datagram, generates a new source port number 5001 for the datagram, replaces the source IP address with its WAN-side IP address 138.76.29.7, and replaces the original source port number 3345 with the new source port number 5001.

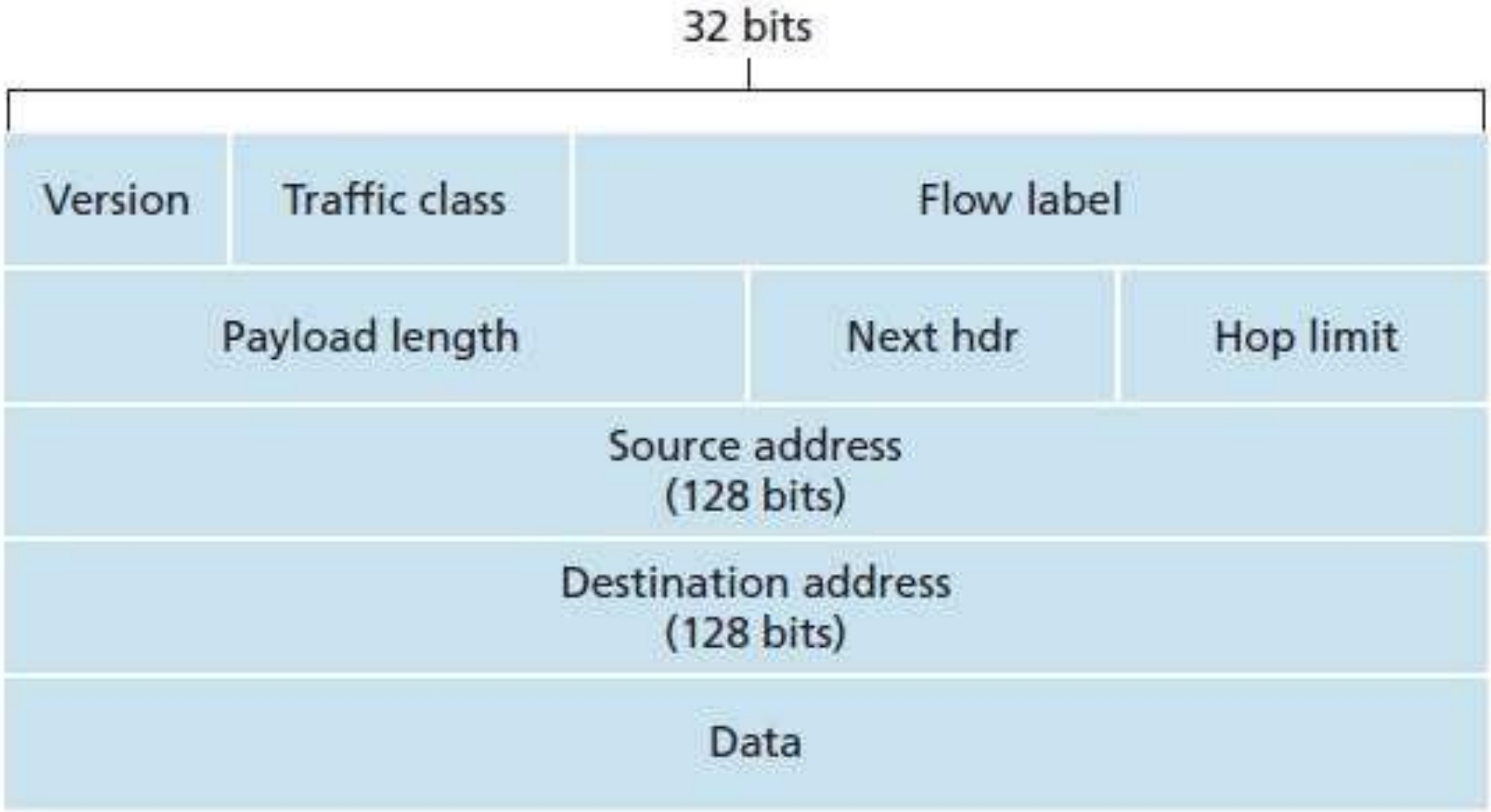
Internet Control Message Protocol (ICMP)

- ICMP is used by hosts and routers to communicate network- layer information to each other.
- The most typical use of ICMP is for error reporting.
- For example, when running a Telnet, FTP, or HTTP session, have encountered an error message such as “Destination network unreachable.” This message had its origins in ICMP.
- ICMP messages are carried as IP payload, just as TCP or UDP segments are carried as IP payload.
- ICMP messages have a type and a code field, and contain the header and the first 8 bytes of the IP datagram that caused the ICMP message to be generated.

IPv6

- The 32-bit IP address space was beginning to be used up, with new subnets and IP nodes being attached to the Internet (and being allocated unique IP addresses) at a breathtaking rate.
- Need for a large IP address space, a new IP protocol, IPv6, was developed.

IPv6 Datagram Format



The most important changes introduced in IPv6 are:

- *Expanded addressing capabilities*. IPv6 increases the size of the IP address from 32 to 128 bits. This ensures that the world won't run out of IP addresses.
- In addition to unicast and multicast addresses, IPv6 has introduced a new type of address, called an **anycast address**, which allows a datagram to be delivered to any one of a group of hosts.
- *A streamlined 40-byte header*. a number of IPv4 fields have been dropped or made optional.
- The resulting 40-byte fixed-length header allows for faster processing of the IP datagram.
- *Flow labeling and priority.* The IPv6 header 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.

The IPv6 format contains:

- *Version.* This 4-bit field identifies the IP version number.
- *Traffic class.* This 8-bit field is similar in spirit to the TOS field we saw in IPv4.
- *Flow label.* this 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 .
- *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.

- 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 back to the sender.
- The sender can then resend the data, using a smaller IP datagram size.
- Because the transport-layer (for example, TCP and UDP) and link-layer (for example, Ethernet) protocols in the Internet layers perform checksumming, this functionality was sufficiently redundant in the network layer and is removed.
- Once again, fast processing of IP packets was a central concern.
- The removal of the options field results in a fixed-length, 40-byte IP header.

How will the public Internet, which is based on IPv4, be transitioned to IPv6?

- One option would be to declare a flag day—a given time and date when all Internet machines would be turned off and upgraded from IPv4 to IPv6.
- **Dual-stack** approach, where IPv6 nodes also have a complete IPv4 implementation. Such a node, referred to as an IPv6/IPv4 node.
- **Tunneling :**Suppose two IPv6 nodes want to interoperate using IPv6 datagrams but are connected to each other by intervening IPv4 routers.
- We refer to the intervening set of IPv4 routers between two IPv6 routers as a **tunnel**
- With tunneling, the IPv6 node on the sending side of the tunnel takes the *entire* IPv6 datagram and puts it in the data (payload) field of an IPv4 datagram.

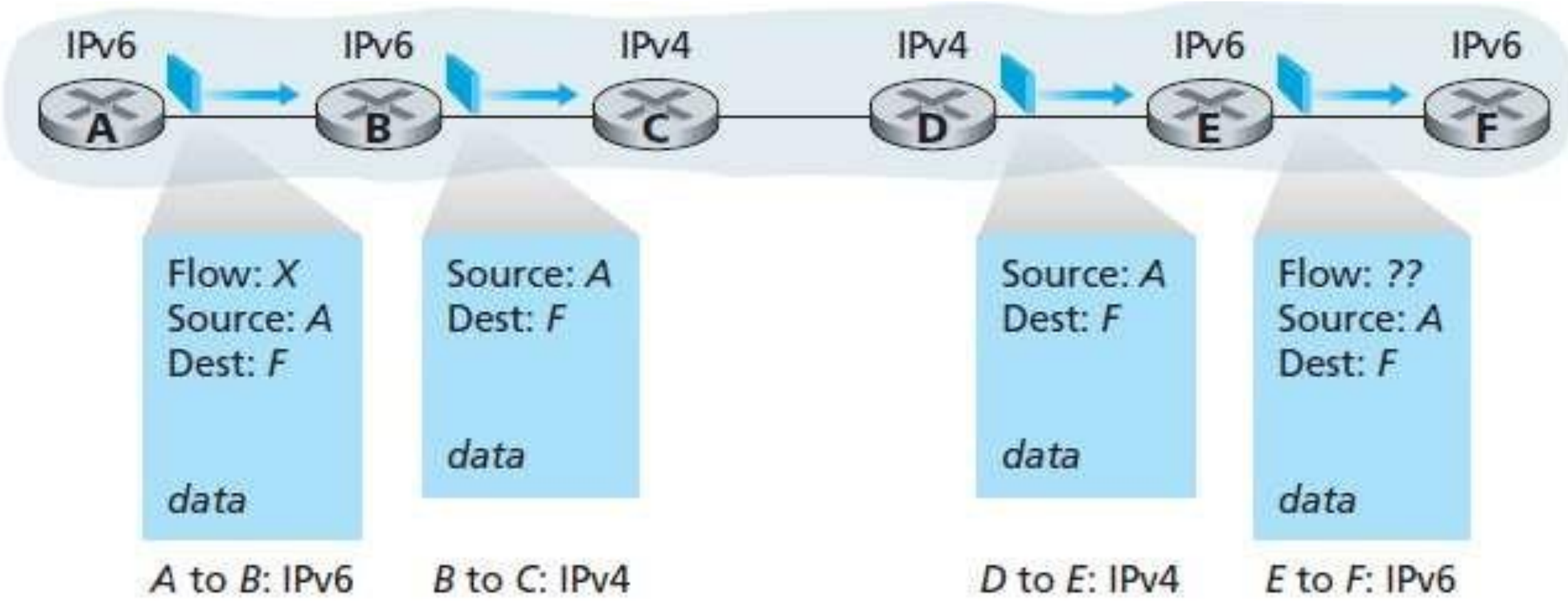


Figure 4.25 ♦ A dual-stack approach

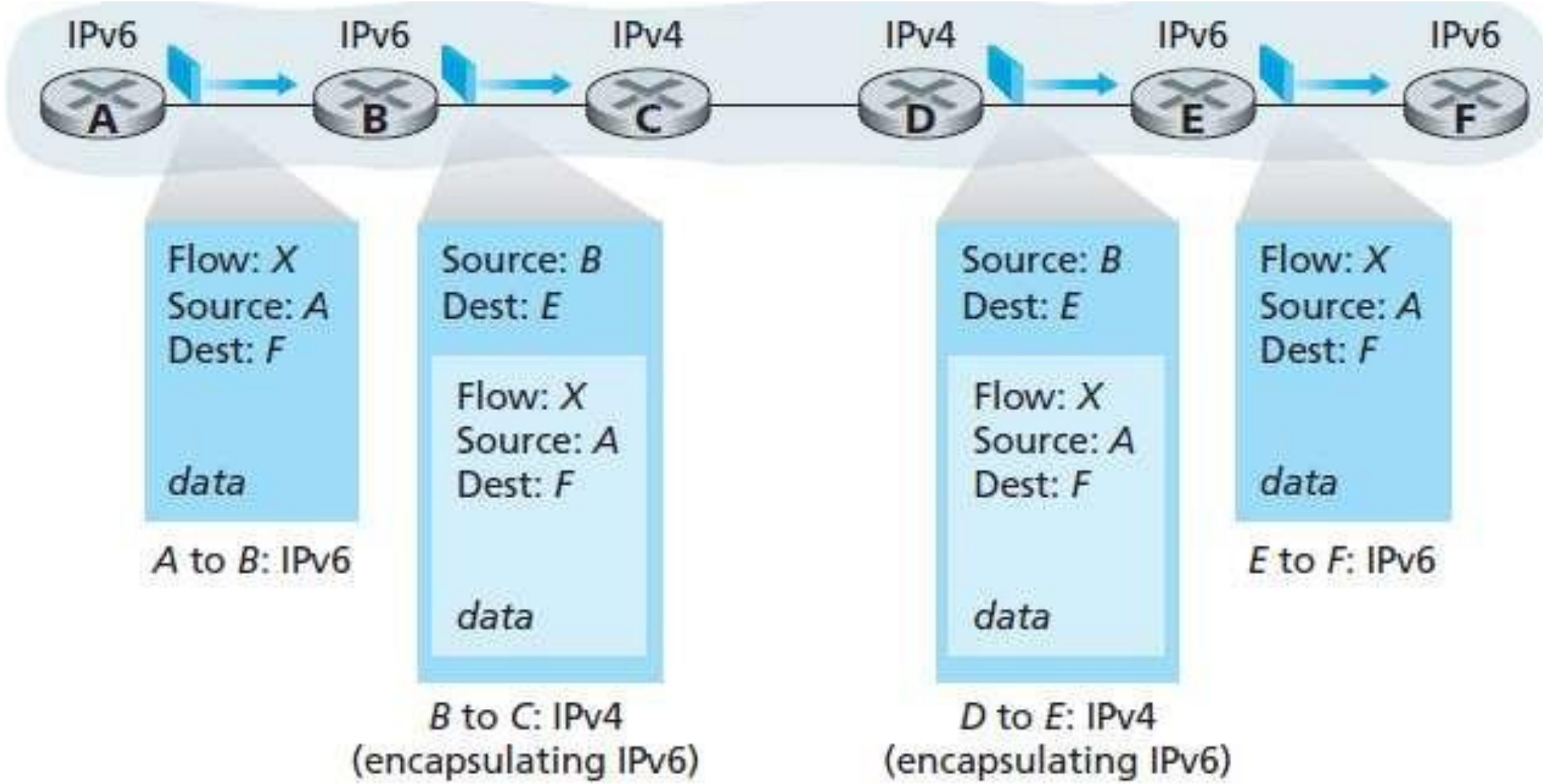


Figure 4.26 ♦ Tunneling

IP Security

- IPsec protocol has been designed to be backward compatible with IPv4 and IPv6. In particular, in order to reap the benefits of IPsec, we don't need to replace the protocol stacks in *all* the routers and hosts in the Internet.
- IPsec's transport mode, two hosts first establish an IPsec session between themselves.
- On the sending side, the transport layer passes a segment to IPsec. IPsec then encrypts the segment, appends additional security fields to the segment, and encapsulates the resulting payload in an ordinary IP datagram.
- The sending host then sends the datagram into the Internet, which transports it to the destination host. There, IPsec decrypts the segment and passes the unencrypted segment to the transport layer.

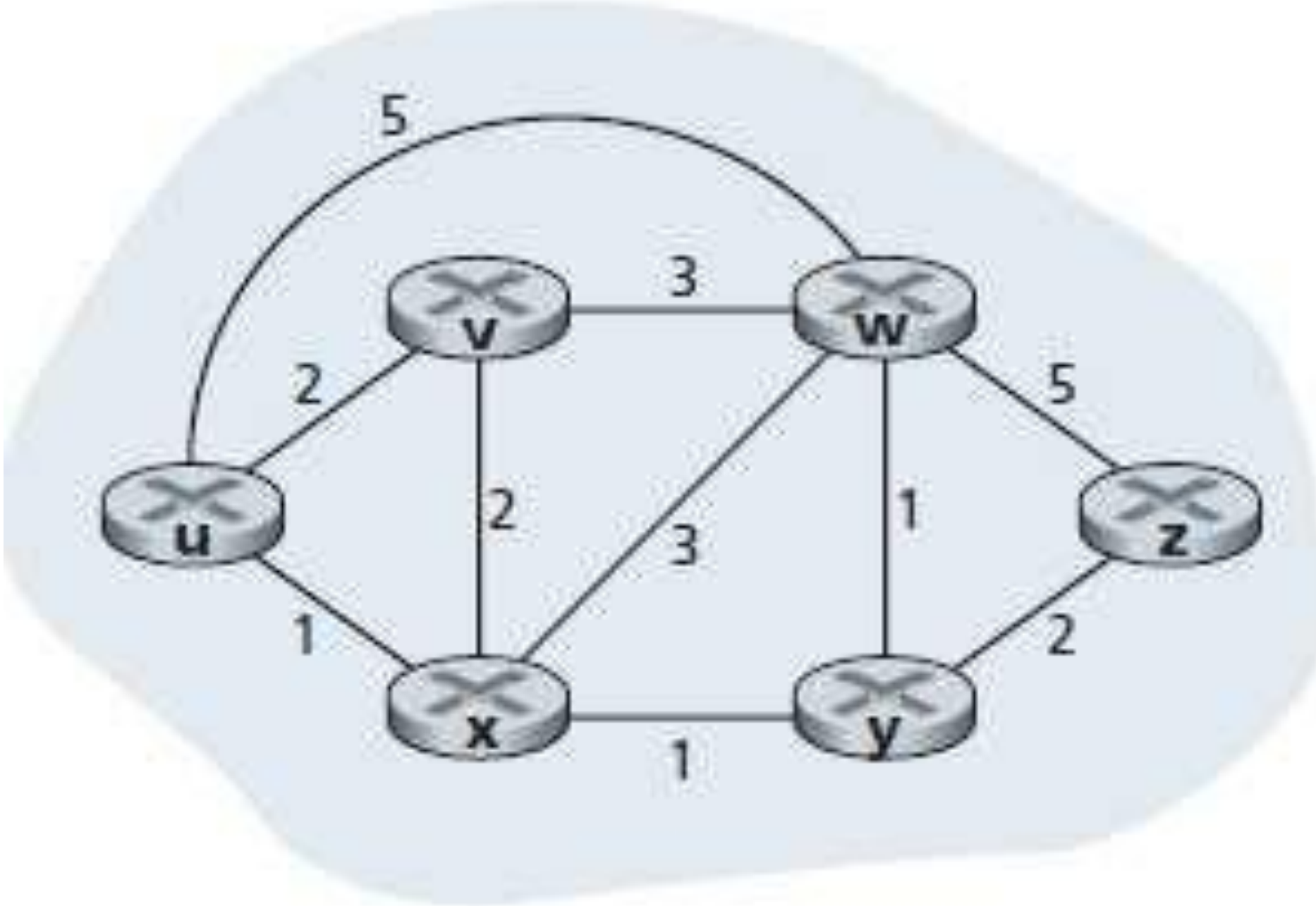
The services provided by an IPsec session

- *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 the host is assured that the source IP address in the datagram is the actual source of the datagram.

Routing Algorithms

- The job of routing is to determine good paths (equivalently, routes), from senders to receivers, through the network of routers.
- Typically a host is attached directly to one router, the **default router** for the host (also called the **first-hop router** for the host).
- Whenever a host sends a packet, the packet is transferred to its default router.
- We refer to the default router of the source host as the **source router** and the default router of the destination host as the **destination router**.

- The purpose of a routing algorithm is then simple: given a set of routers, with links connecting the routers, a routing algorithm finds a "good" path from source router to destination router.
- Typically, a good path is one that has the least cost.
- A graph is used to formulate routing problems.
- Recall that a **graph** $G = (N,E)$ is a set N of nodes and a collection E of edges, where each edge is a pair of nodes from N.
- In the context of network-layer routing, the nodes in the graph represent routers—the points at which packet-forwarding decisions are made—and the edges connecting these nodes represent the physical links between these routers.
- Such a graph is an abstraction of a computer network.


5
3
v
w
2
5
2
1
3
u
z
1
2
x
y
1

- An edge also has a value representing its cost.
- Typically, an edge's cost may reflect the physical length of the corresponding link, the link speed, or the monetary cost associated with a link.
- For any edge (x,y) in E, we denote $c(x,y)$ as the cost of the edge between nodes x and y.
- If the pair (x,y) does not belong to E, we set $c(x,y) = \infty$.
- Consider only undirected graphs (i.e., graphs whose edges do not have a direction), so that edge (x,y) is the same as edge (y,x) and that $c(x,y) = c(y,x)$.
- Also, a node y is said to be a **neighbor** of node x if (x,y) belongs to E.

- A natural goal of a routing algorithm is to identify the least costly paths between sources and destinations.
- To make this problem more precise, recall that a **path** in a graph $G = (N,E)$ is a sequence of nodes $(x1, x2,..., xp)$ such that each of the pairs $(x1,x2),(x2,x3),...,(xp-1,xp)$ are edges in E.
- The cost of a path $(x1,x2,..., xp)$ is simply the sum of all the edge costs along the path, that is, $c(x1,x2) + c(x2,x3) + ...+ c(xp-1,xp)$.
- Given any two nodes x and y, there are typically many paths between the two nodes, with each path having a cost. One or more of these paths is a **least-cost path**.
- The least-cost problem is therefore clear: Find a path between the source and destination that has least cost.
- For example, the least-cost path between source node u and destination node w is (u, x, y, w) with a path cost of 3.
- Note that if all edges in the graph have the same cost, the least-cost path is also the **shortest path.**

Classification of Routing Algorithms

- **global routing algorithm / decentralized routing algorithm**
- **static routing algorithms / Dynamic routing algorithms**
- **load-sensitive algorithm / load-insensitive algorithm**

global routing algorithm

- A **global routing algorithm** computes the least-cost path between a source and destination using complete, global knowledge about the network.
- That is, the algorithm takes the connectivity between all nodes and all link costs as inputs.
- This then requires that the algorithm somehow obtain this information before actually performing the calculation.
- The calculation itself can be run at one site.
- In practice, algorithms with global state information are often referred to as **link-state** (**LS**) **algorithms**, since the algorithm must be aware of the cost of each link in the network.

decentralized routing algorithm

- In a **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.
- Instead, 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 (that is, nodes that are at the other end of links to which it itself is attached), a node gradually calculates the least-cost path to a destination or set of destinations.
- The decentralized routing algorithm is called a distance-vector (DV) algorithm, because each node maintains a vector of estimates of the costs (distances) to all other nodes in the network.

- In **static routing algorithms**, routes change very slowly over time, often as a result of human intervention (for example, a human manually editing a router's forwarding table).
- **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.

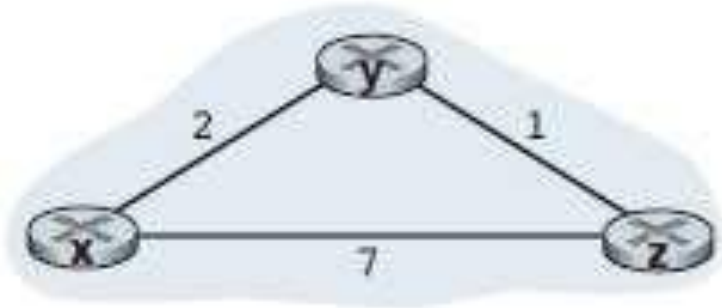
- In a **load-sensitive algorithm**, link costs vary dynamically to reflect the current level of congestion in the underlying link.
- If a high cost is associated with a link that is currently congested, a routing algorithm will tend to choose routes around such a congested link.
- Today's Internet routing algorithms (such as RIP, OSPF, and BGP) are **load-insensitive**, as a link's cost does not explicitly reflect its current (or recent past) level of congestion.

The Distance-Vector (DV) Routing Algorithm

- The **distancevector** (**DV**) algorithm is iterative, asynchronous, and distributed.
- It is *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.
- The algorithm is *asynchronous* in that it does not require all of the nodes to operate.

```
1  Initialization:
2     for all destinations y in N:
3        D_x(y) = c(x,y)    /* if y is not a neighbor then c(x,y) = ∞ */
4     for each neighbor w
5        D_w(y) = ? for all destinations y in N
6     for each neighbor w
7        send distance vector D_x = [D_x(y): y in N] to w
8
9  loop
10    wait (until I see a link cost change to some neighbor w or
11          until I receive a distance vector from some neighbor w)
12
13    for each y in N:
14       D_x(y) = min_v{c(x,v) + D_v(y)}
15
16    if D_x(y) changed for any destination y
17       send distance vector D_x = [D_x(y): y in N] to all neighbors
18
19 forever
```

- In the DV algorithm, a node *x* updates its distance-vector estimate when it either sees a cost change in one of its directly attached links or receives a distancevector update from some neighbor.
- the only information a node will have is the costs of the links to its directly attached neighbors and information it receives from these neighbors.
- Each node waits for an update from any neighbor calculates its new distance vector when receiving an update and distributes its new distance vector to its neighbors .
- DV-like algorithms are used in many routing protocols in practice, including the Internet's RIP and BGP, ISO IDRP, Novell IPX, and the original ARPAnet.



Node x table

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

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

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

Node y table

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

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

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

The Link-State (LS) Routing Algorithm

- In a link-state algorithm, the network topology and all link costs are known, that is, available as input to the LS algorithm.
- In practice this is accomplished by having each node broadcast link-state packets to *all* other nodes in the network, with each link-state packet containing the identities and costs of its attached links.
- All nodes have an identical and complete view of the network.
- Each node can then run the LS algorithm and compute the same set of least-cost paths as every other node.

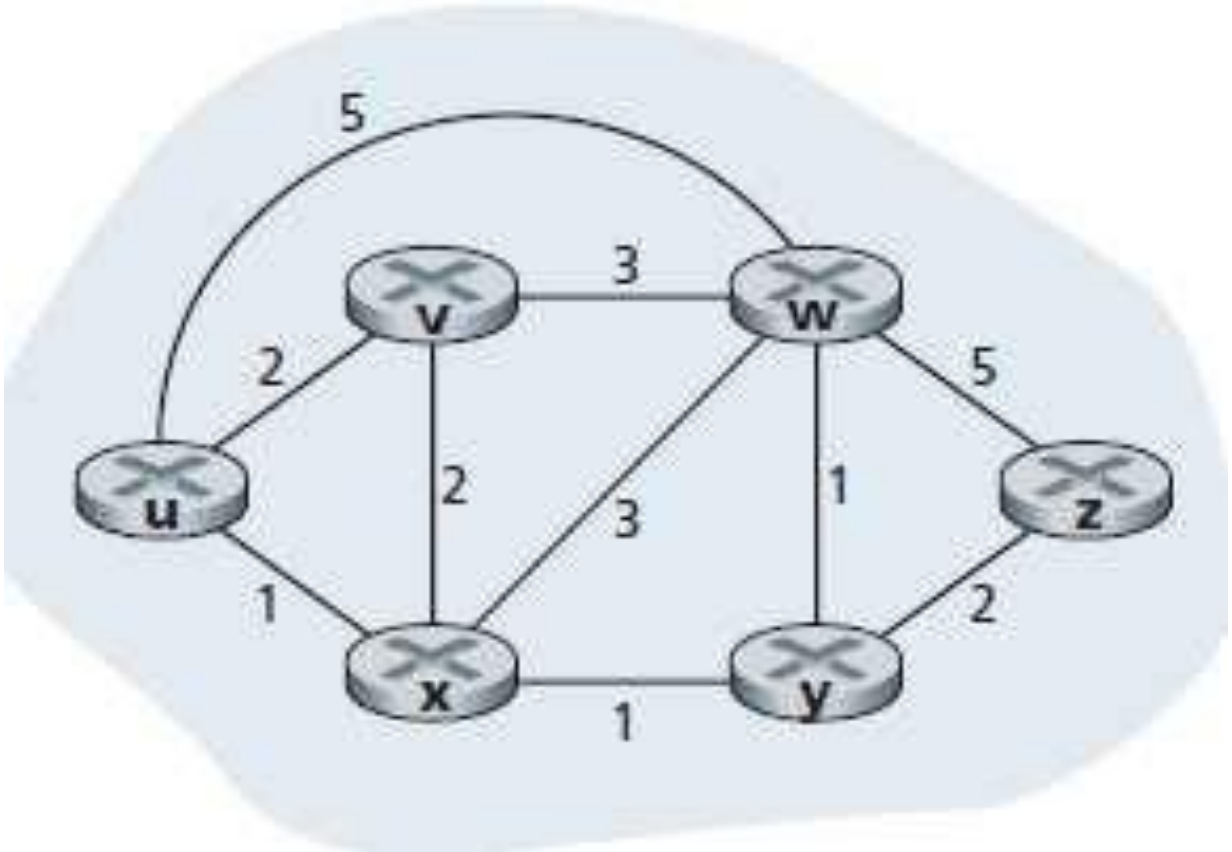
- The link-state routing algorithm we present below is known as *Dijkstra's algorithm.*
- Dijkstra's algorithm computes the least-cost path from one node (the source, which we will refer to as *u*) to all other nodes in the network.
- Dijkstra's algorithm is iterative and has the property that after the *k*th iteration of the algorithm, the least-cost paths are known to *k* destination nodes.
- Notations used are :
- *D*(*v*): cost of the least-cost path from the source node to destination *v*
- *p*(*v*): previous node (neighbor of *v*) along the current least-cost path.
- *N*': subset of nodes.

- The global routing algorithm consists of an initialization step followed by a loop.
- The number of times the loop is executed is equal to the number of nodes in the network.
- Upon termination, the algorithm will have calculated the shortest paths from the source node *u* to every other node in the network.

Link-State (LS) Algorithm for Source Node *u*

```
1  Initialization:
2     N' = {u}
3     for all nodes v
4       if v is a neighbor of u
5         then D(v) = c(u,v)
6       else D(v) = ∞
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 each neighbor v of 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     least path cost to w plus cost from w to v */
15 until N'= N
```

Example



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

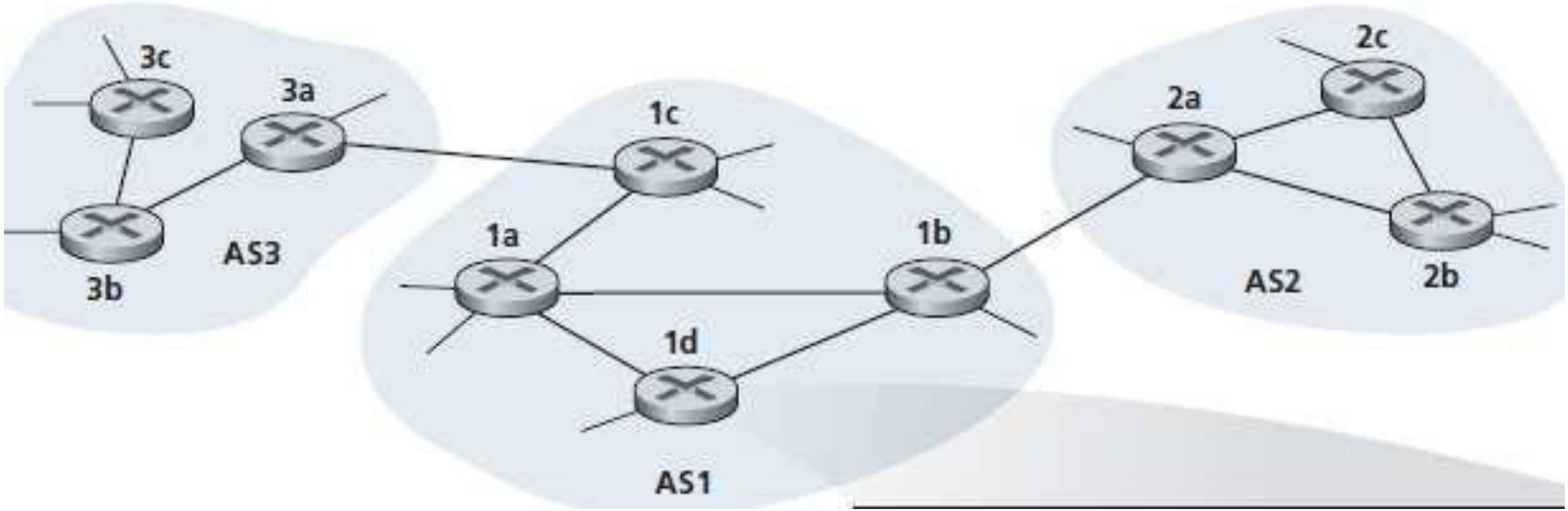
A Comparison of LS and DV Routing Algorithms

- In the LS algorithm, each node talks with *all* other nodes (via broadcast), but it tells them *only* the costs of its directly connected links.
- In the DV algorithm, each node talks to *only* its directly connected neighbors, but it provides its neighbors with least-cost estimates from itself to *all* the nodes (that it knows about) in the network.

- *Message complexity.*
- LS requires each node to know the cost of each link in the network. This requires O(|N| |E|) messages to be sent.
- The DV algorithm requires message exchanges between directly connected neighbors at each iteration.
- *Speed of convergence.*
- LS is an $O(|N|^2)$ algorithm requiring O(|N| |E|)) messages. The DV algorithm can converge slowly and can have routing loops.
- *Robustness.*
- Route calculations are somewhat separated under LS, providing a degree of robustness.
- Under DV, a node can advertise incorrect least-cost paths to any or all destinations.

Autonomous systems (ASs)

- The problems of scaling and administration of large group of routers 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.
- Routers within the same AS all run the same routing algorithm (for example, an LS or DV algorithm) and have information about each .
- The routing algorithm running within an autonomous system is called an **intra-autonomous system routing protocol**.
- It will be necessary, of course, to connect ASs to each other, and thus one or more of the routers in an AS will have the added task of being responsible for forwarding packets to destinations outside the AS;
- these routers are called **gateway routers**.

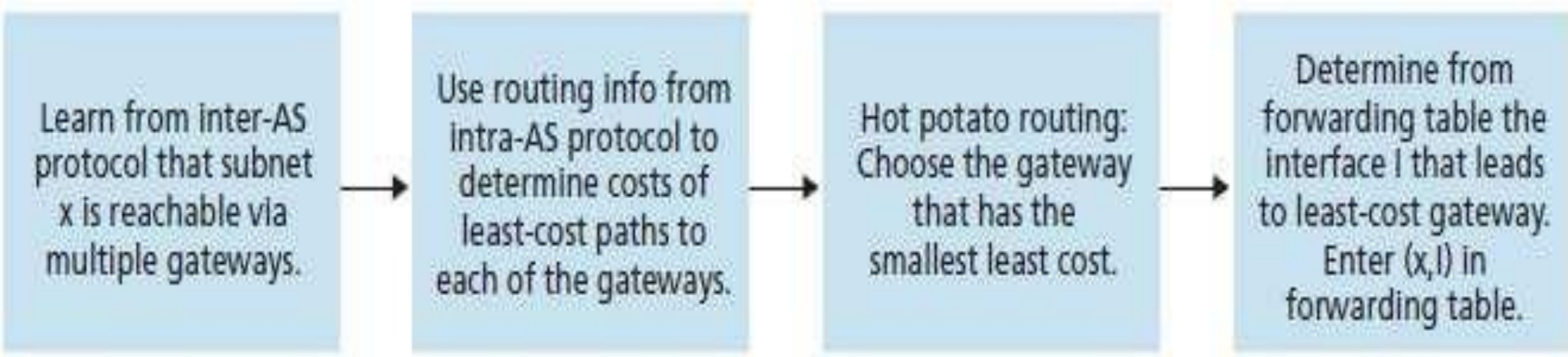

3c
3a
3b
AS3
1c
1a
1b
1d
AS1
2c
2a
2b
AS2

- 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 fact, in the Internet all ASs run the same inter-AS routing protocol, called BGP4. **Border Gateway Protocol**

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.
- hot-potato routing, would use information from the intra-AS routing protocol to determine the path costs, and then choose the path with the least cost.

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



- In summary, the 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.

- Two intra-AS routing protocols are **Routing Information**
- **Protocol (RIP)** and **Open Shortest Path First (OSPF)**.
- The inter-AS routing protocol is BGP used in today's Internet.