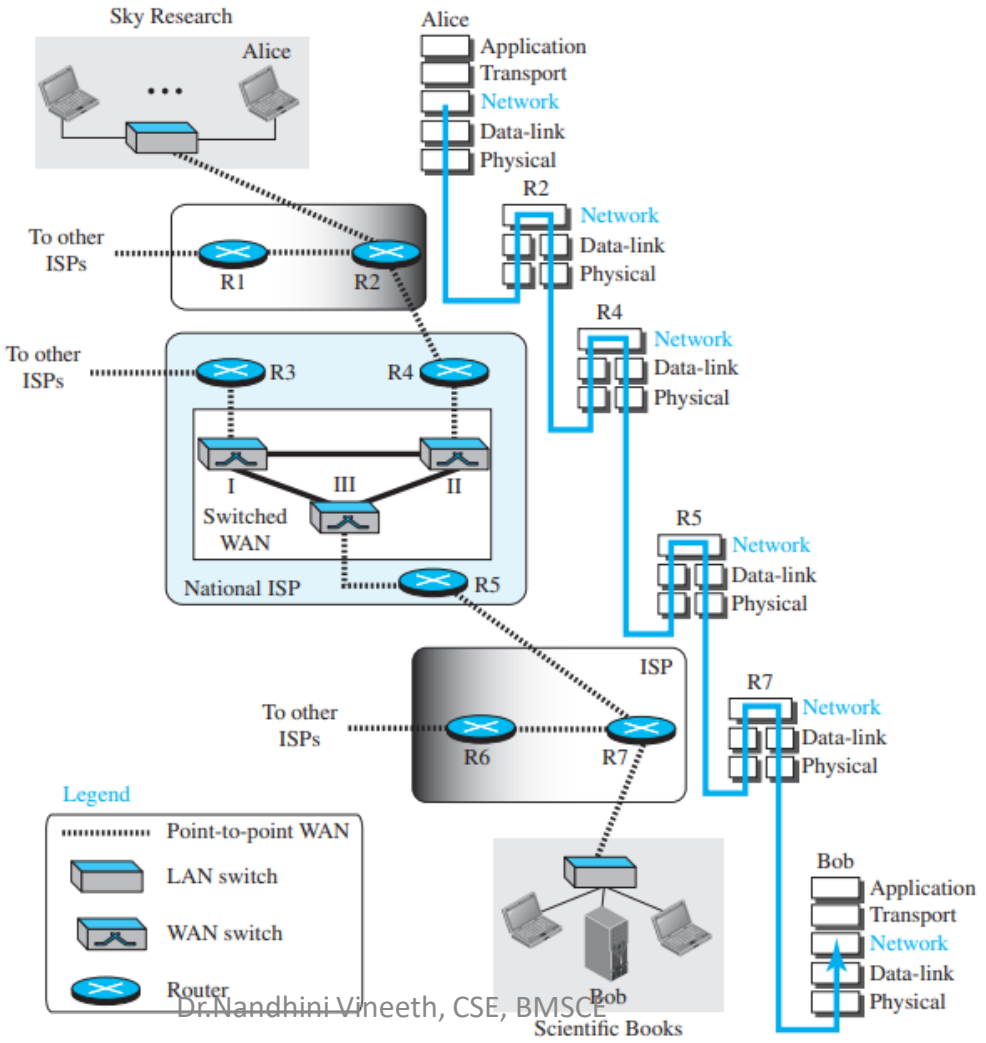
Introduction To Network Layer

UNIT III

Introduction to Network Layer

NETWORK-LAYER SERVICES

Figure 18.1 *Communication at the network layer*



Packetizing

One duty of the network layer is to **carry a payload from the source to the destination** without changing it or using it

If the packet is fragmented at the source or at routers along the path, the network layer is responsible for waiting until all fragments arrive, reassembling them, and delivering them to the upper-layer protocol.

The routers in the path are not allowed to decapsulate the packets they received unless the packets need to be fragmented.

The routers are not allowed to change source and destination addresses either.

They just inspect the addresses for the purpose of forwarding the packet to the next network on the path.

However, if a packet is fragmented, the header needs to be copied to all fragments

Routing and Forwarding

More than **one route from the source to the destination**.

The network layer is responsible for finding the **best one among these possible routes**.

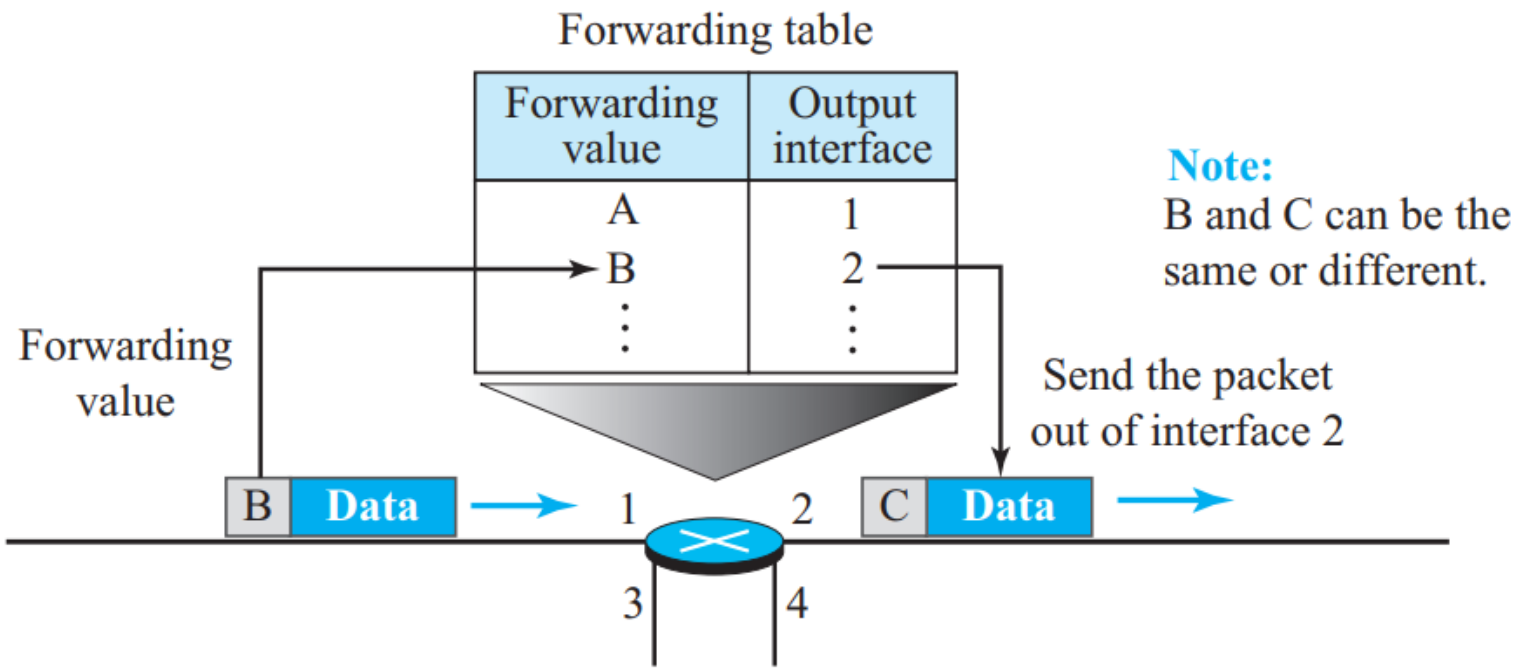
The network layer needs to have some **specific strategies** for defining the best route

routing protocols help the routers coordinate their knowledge about the neighborhood and to come up with consistent tables to be used when a packet arrives

Forwarding - the action applied by each router when a packet arrives at one of its interfaces.

The decision-making table a router normally uses for applying this action is sometimes called the forwarding table and sometimes the routing table.

Figure 18.2 *Forwarding process*



Other Services

Error Control

designers of the network layer in the Internet ignored this issue

reason for this decision is the fact that the packet in the network layer may be fragmented at each router, which makes error checking at this layer inefficient.

however, have added a checksum field to the datagram to control any corruption in the header, but not in the whole datagram

although NL does not directly provide error control, the Internet uses an auxiliary protocol, ICMP, that provides some kind of error control if the datagram is discarded or has some unknown information in the header

Flow Control:

No direct support
As higher layers take care –with buffers

Other Services

Congestion Control:

too many datagrams
May occur due to source tx too many datagrams

Some routers drop / increases congestion / mechanism discussed later

Quality of Service:

Multimedia communications- demand

Taken care by upper layer

Security:
. To provide security for a connectionless network layer, we need to have another virtual level that changes the connectionless service to a connection-oriented service. This virtual layer, called **IPSec**

PACKET SWITCHING

18.2.1 Datagram Approach: Connectionless Service

Figure 18.3 *A connectionless packet-switched network*

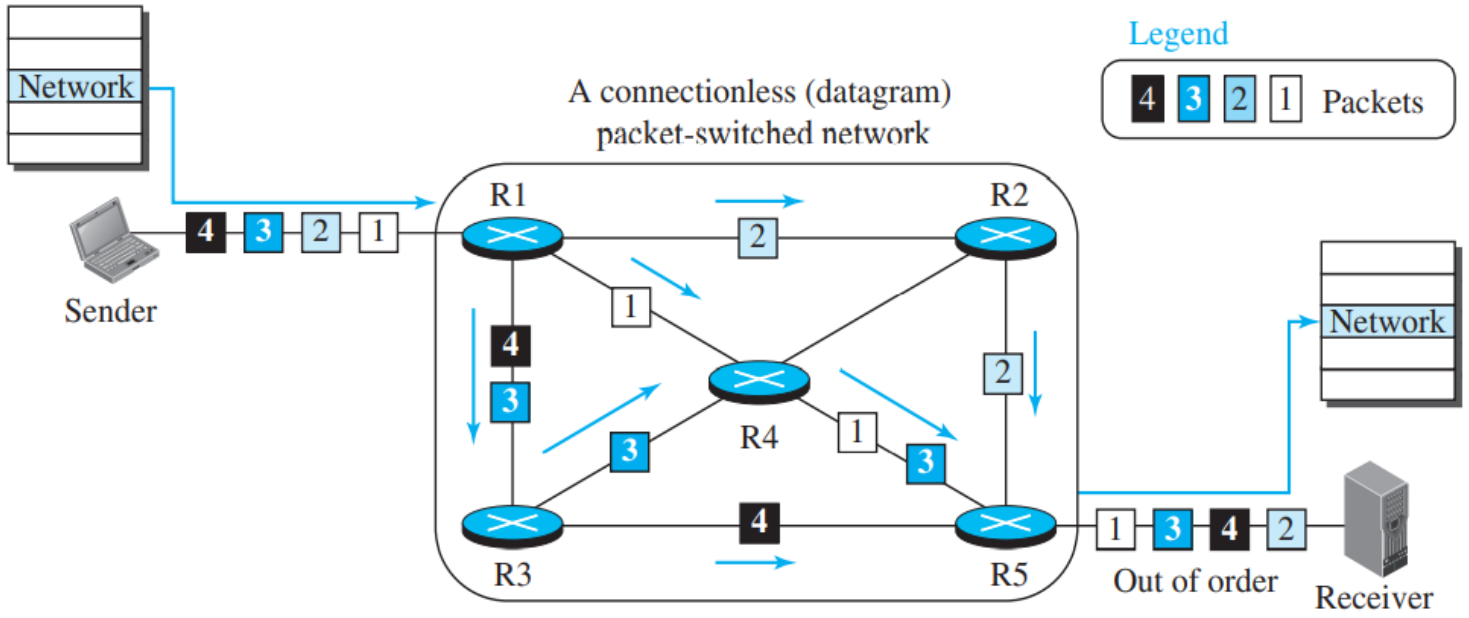
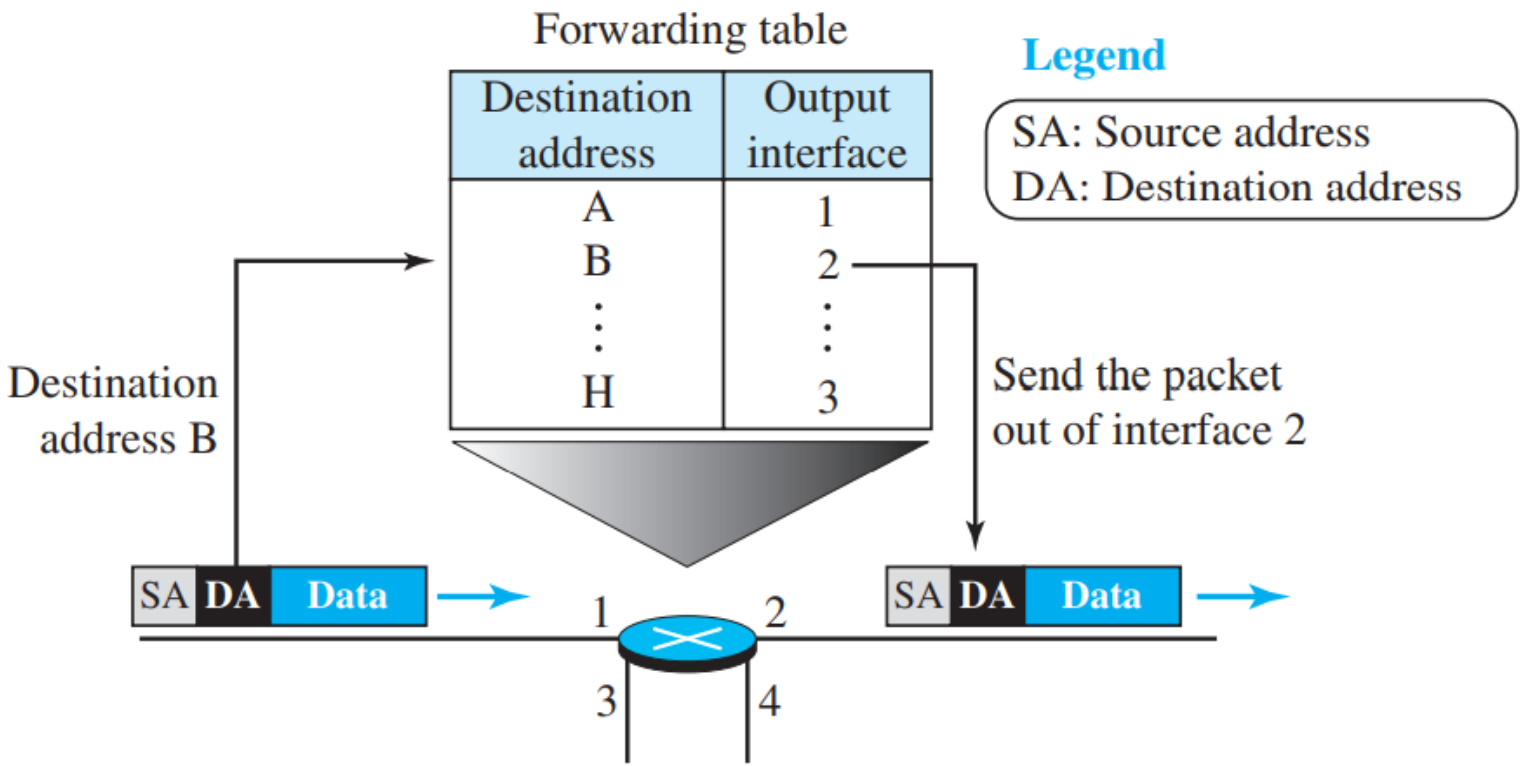
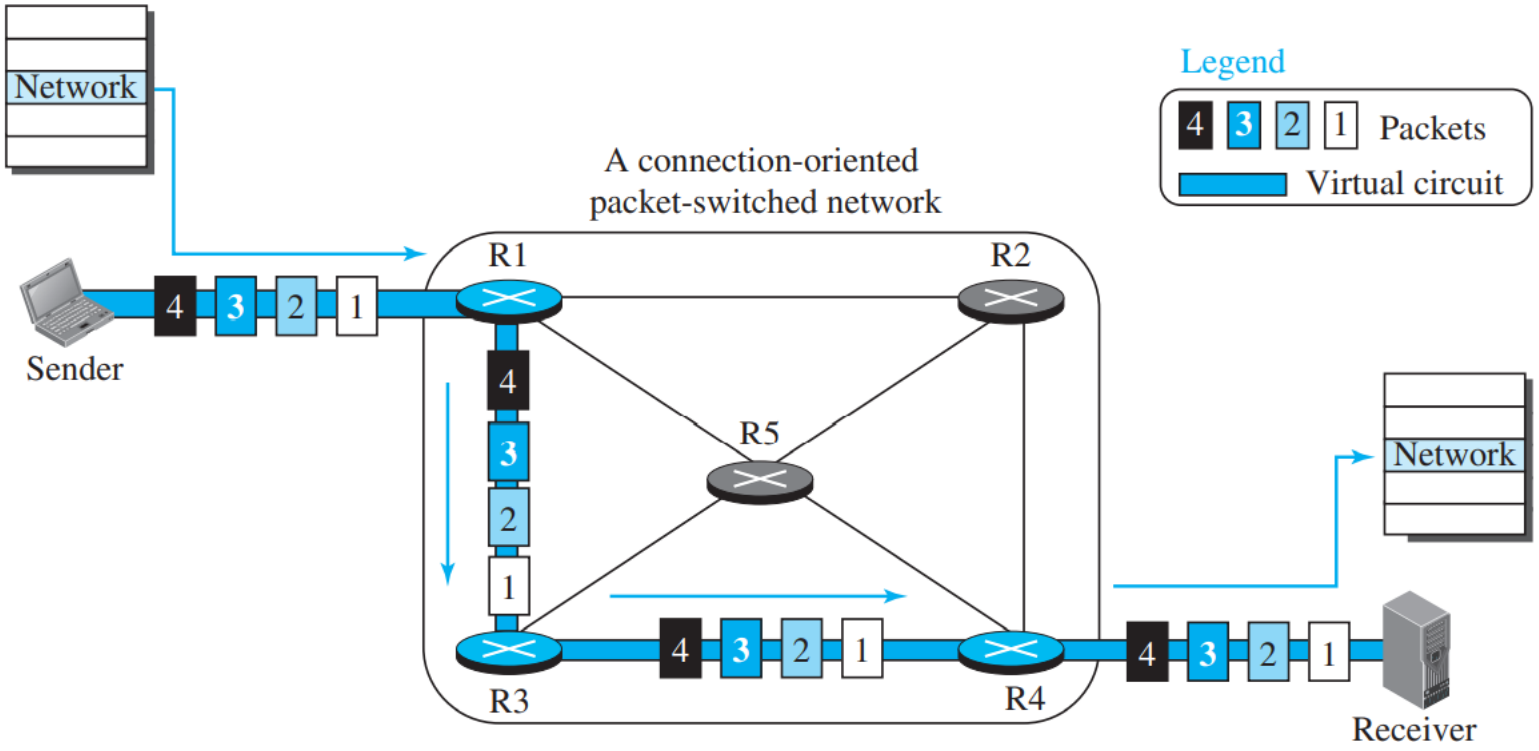


Figure 18.4 *Forwarding process in a router when used in a connectionless network*



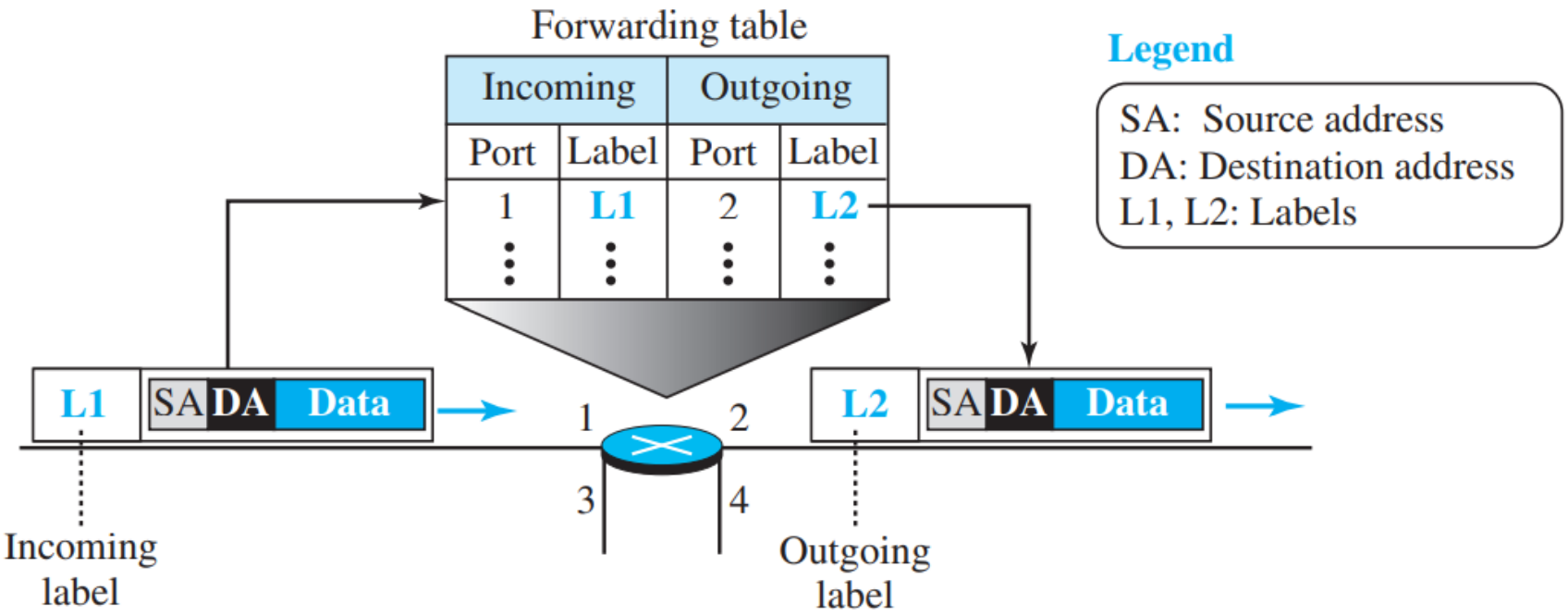
18.2.2 Virtual-Circuit Approach: Connection-Oriented Service

Figure 18.5 *A virtual-circuit packet-switched network*



Setup Phase

Figure 18.6 *Forwarding process in a router when used in a virtual-circuit network*



In the virtual-circuit approach, the forwarding decision is based on the label of the packet.

Request packet

Figure 18.7 *Sending request packet in a virtual-circuit network*

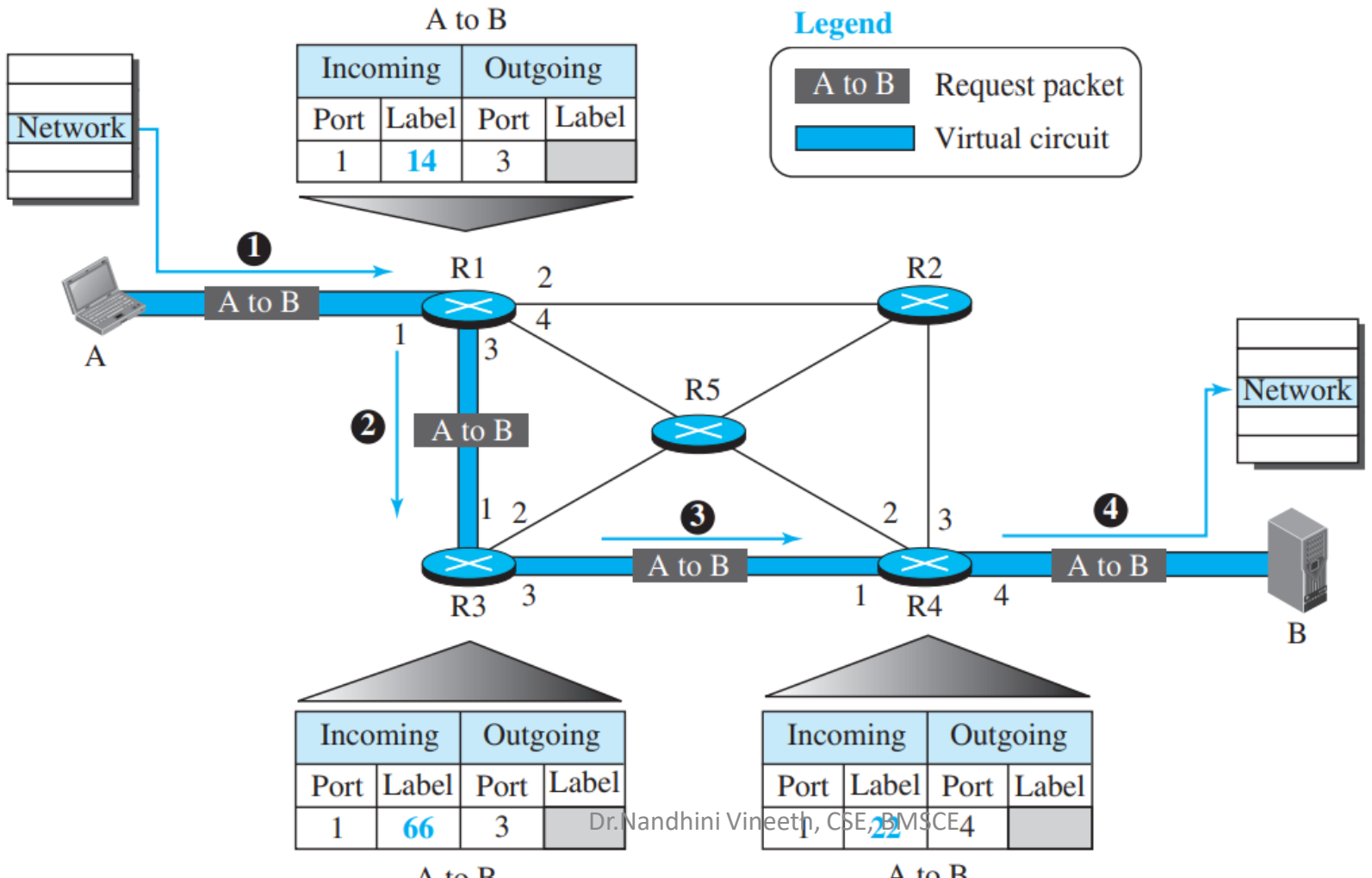


Figure 18.8 *Sending acknowledgments in a virtual-circuit network*

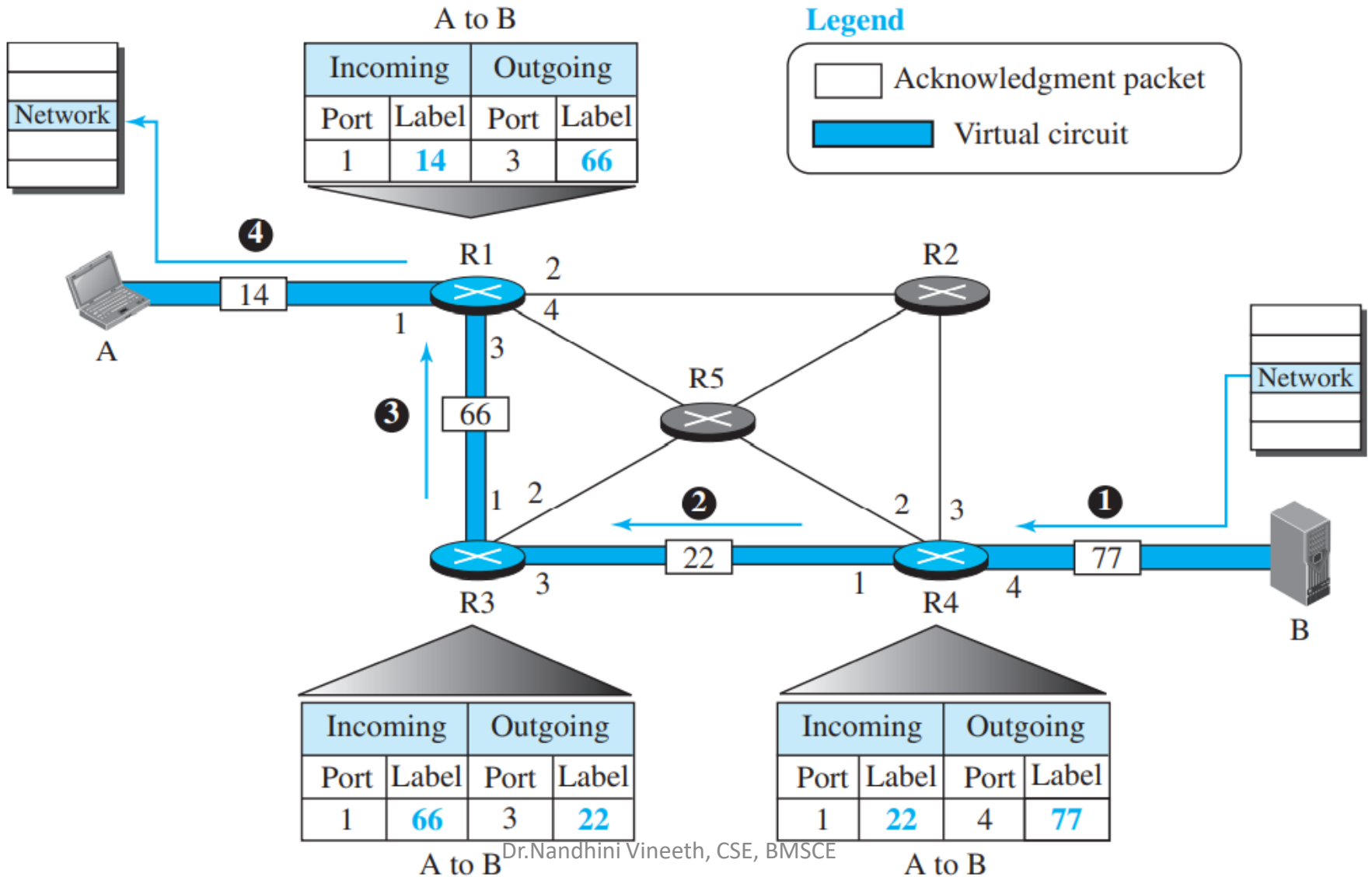
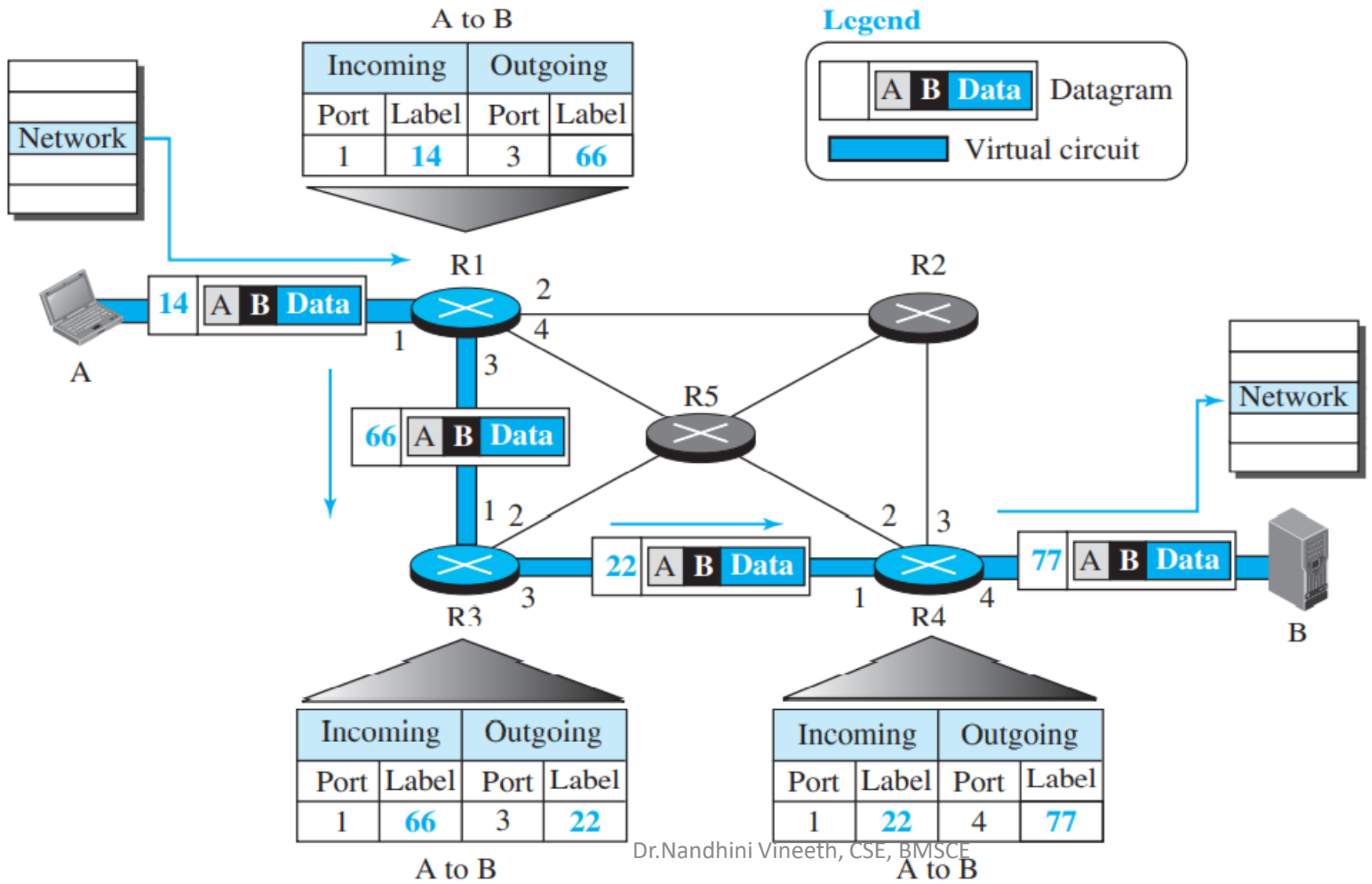


Figure 18.9 *Flow of one packet in an established virtual circuit*



18.3 NETWORK-LAYER PERFORMANCE

The performance of a network can be measured in terms of delay, throughput, and packet loss.

Delay

Transmission Delay

$Delay_{tr}$ = (Packet length) / (Transmission rate).

Propagation Delay

$Delay_{pg}$ = (Distance) / (Propagation speed).

Processing Delay

$Delay_{pr}$ = Time required to process a packet in a router or a destination host

Queuing Delay

$Delay_{qu}$ = The time a packet waits in input and output queues in a router

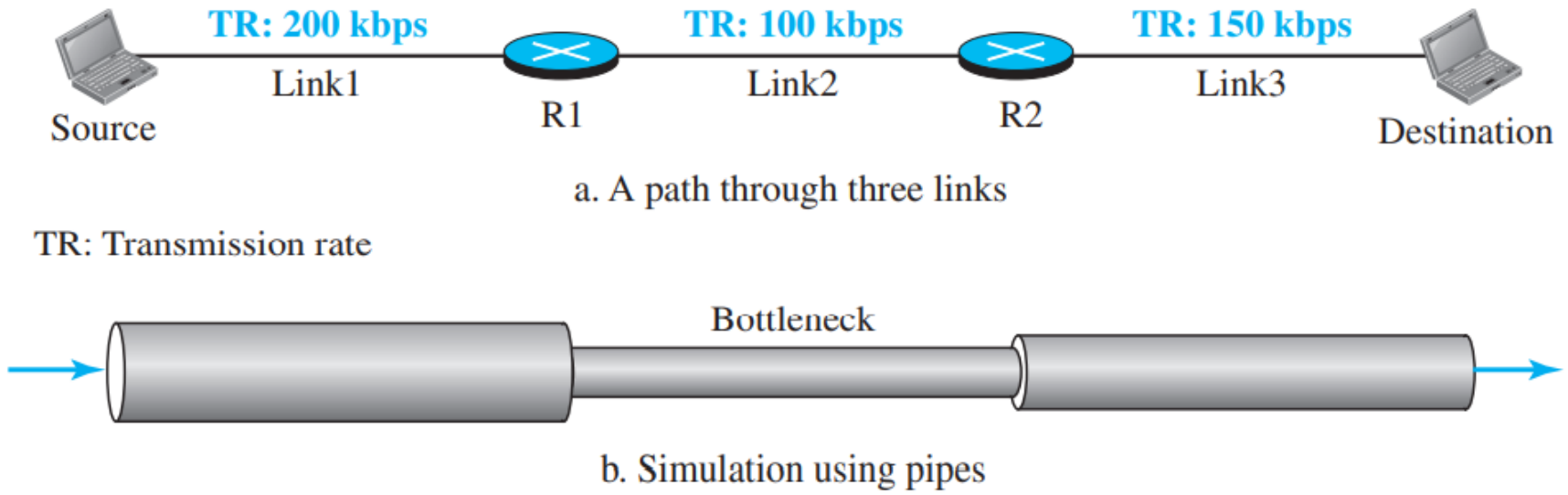
Total Delay

Total delay = $(n + 1)(Delay_{tr} + Delay_{pg} + Delay_{pr}) + (n)(Delay_{qu})$

NETWORK-LAYER PERFORMANCE

Throughput

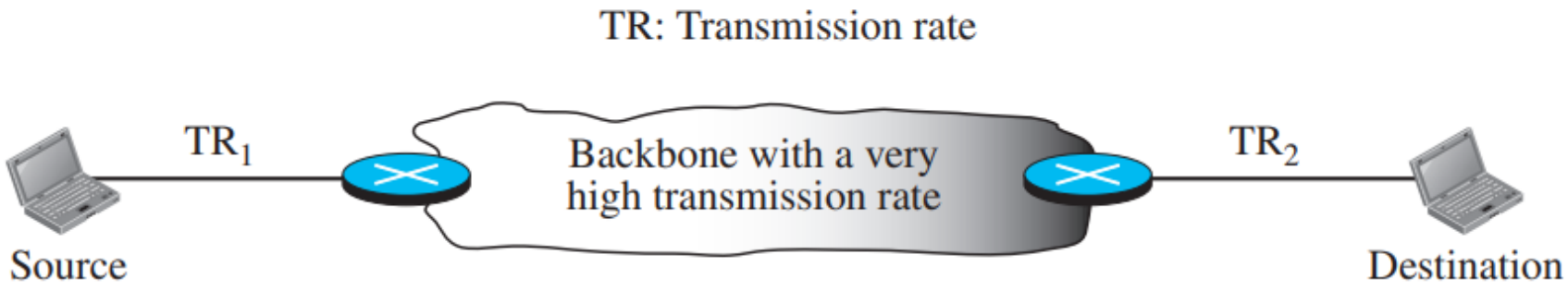
Figure 18.10 *Throughput in a path with three links in a series*



Throughput = minimum {TR1, TR2, . . . TRn}.

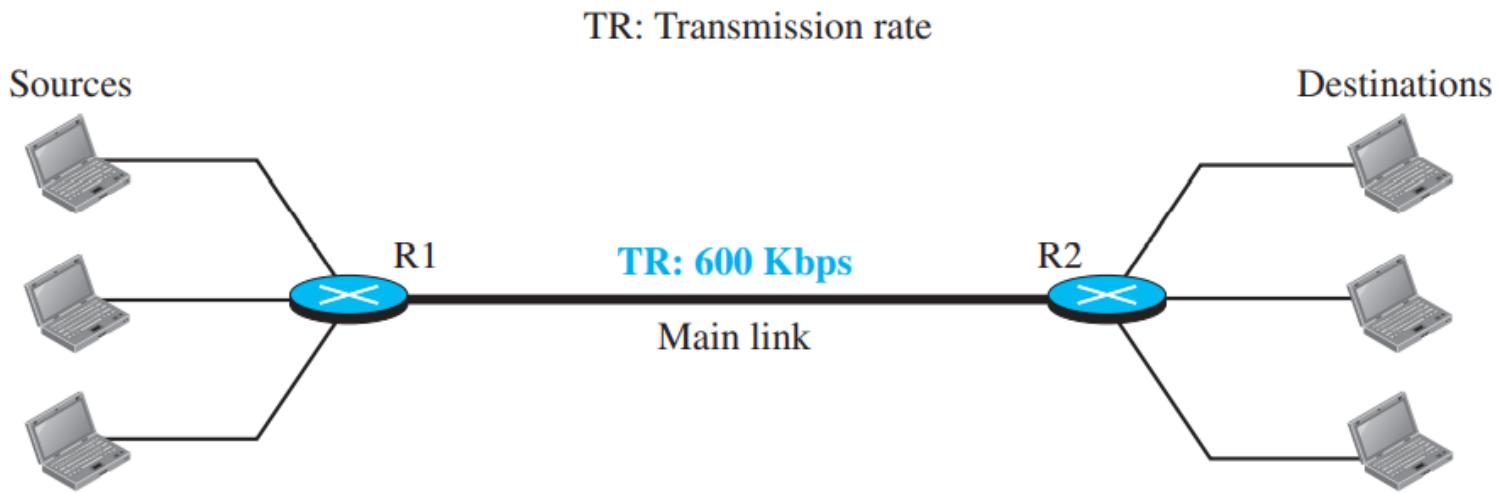
NETWORK-LAYER PERFORMANCE - Throughput

Figure 18.11 *A path through the Internet backbone*



NETWORK-LAYER PERFORMANCE

Figure 18.12 *Effect of throughput in shared links*



Packet Loss

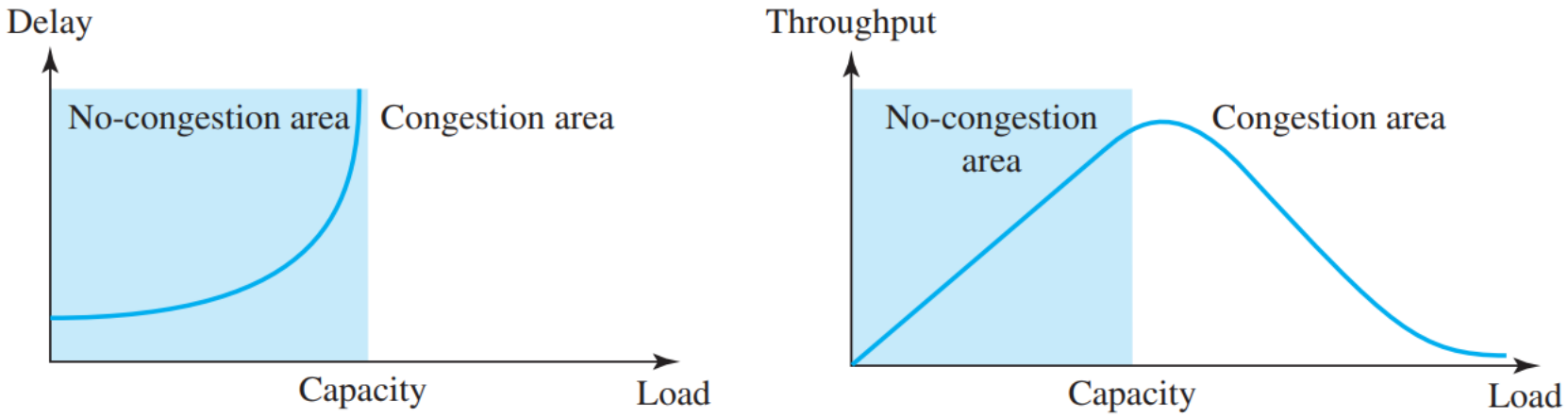
At routers due to overflow of buffers

Depending on higher layers, retransmission may or may not occur

Congestion Control

Congestion at the network layer is related to two issues, throughput and delay

Figure 18.13 *Packet delay and throughput as functions of load*



a. Delay as a function of load

b. Throughput as a function of load

Congestion Control

We can divide congestion control mechanisms into two broad categories: **open-loop congestion control (prevention)** and **closed-loop congestion control (removal).**

Open-Loop Congestion Control

Congestion control is handled by either the source or the destination

Retransmission Policy:

Retransmission in general may increase congestion in the network.
The retransmission policy and the retransmission timers must be designed to optimize efficiency and at the same time prevent congestion.

Window Policy
The type of window at the sender (decision on retransmitted packets) may also affect congestion.
Go-back- N protocol- if among 1-10 pkts, 4 th is lost, 4-10 are retransmitted. **This duplication may make the congestion worse**
Selective Repeat - in the above scenario, **only 4 is retransmitted**
The **Selective Repeat window is better than the Go-Back-N window** for congestion control

Congestion Control

Acknowledgment Policy:
Acknowledgments are also part of the load in a network. Sending fewer acknowledgments means imposing less load on the network.
A receiver may decide to acknowledge only N packets at a time instead of each.

Discarding Policy:
A good discarding policy by the routers may prevent congestion and at the same time may not harm the integrity of the transmission.
if the policy is to discard less sensitive packets when congestion is likely to happen, the quality of sound is still preserved and congestion is prevented or alleviated.

Admission Policy:
quality-of-service mechanism

Switches in a flow first check the resource requirement of a flow before admitting it to the network. A router can deny establishing a virtual-circuit connection if there is congestion in the network or if there is a possibility of future congestion.

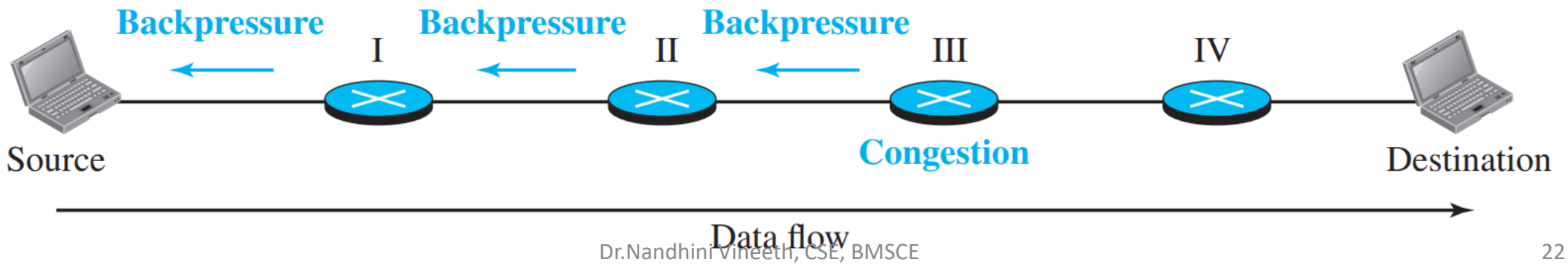
Closed-Loop Congestion

Control Closed-loop congestion control mechanisms try to alleviate congestion after it happens. Several mechanisms have been used by different protocols.

Backpressure

- a node-to-node congestion control that starts with a node and propagates, in the opposite direction of data flow, to the source.
- can be applied only to virtual circuit networks, in which each node knows the upstream node from which a flow of data is coming.

Figure 18.14 *Backpressure method for alleviating congestion*

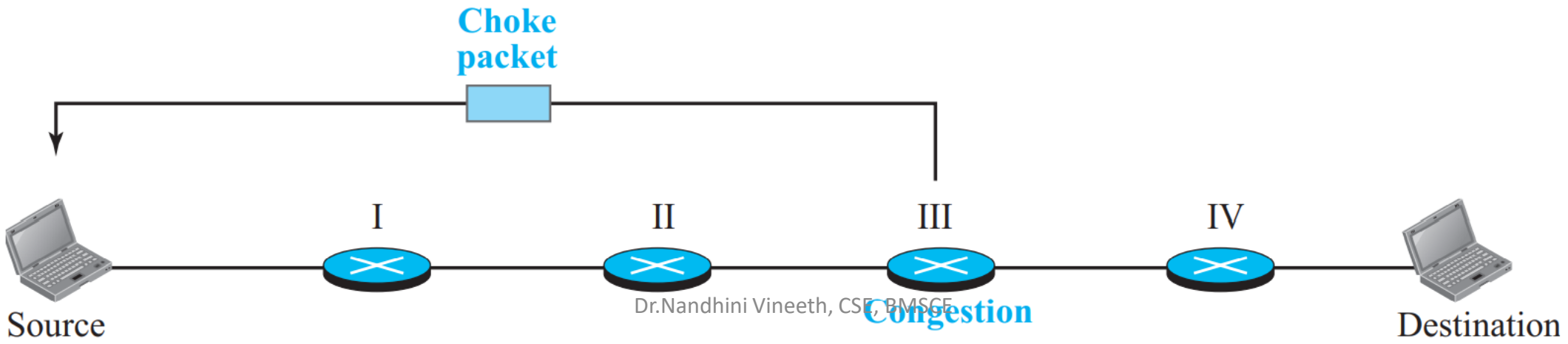


Closed-Loop Congestion

Choke Packet

- **a packet sent by a node to the source to inform it of congestion**
- the warning is from the router, which has encountered congestion, **directly to the source station**.
- When a router in the Internet is overwhelmed with IP datagrams, it may discard some of them, but it informs the source host, using a **source quench ICMP message**.

Figure 18.15 *Choke packet*



Closed-Loop Congestion

Implicit Signaling

- there is no communication between the congested node or nodes and the source
- The source guesses that there is congestion somewhere in the network from other symptoms. Like lack of ACK for long
- Used in TCP congestion control

Explicit Signaling

- The node that experiences congestion can explicitly send a signal to the source or destination.
- Diff:
  - In the choke-packet method, a separate packet is used for this purpose;
  - in the explicit-signaling method, the signal is included in the packets that carry data.
  - Explicit signaling can occur in either the forward or the backward direction. This type of congestion control can be seen in an ATM network

IPV4 ADDRESSES

An **IPv4** address is a **32-bit address** that uniquely and universally **defines the connection** of a host or a router to the Internet-**NOT HOST OR ROUTER**

IPv4 addresses are unique

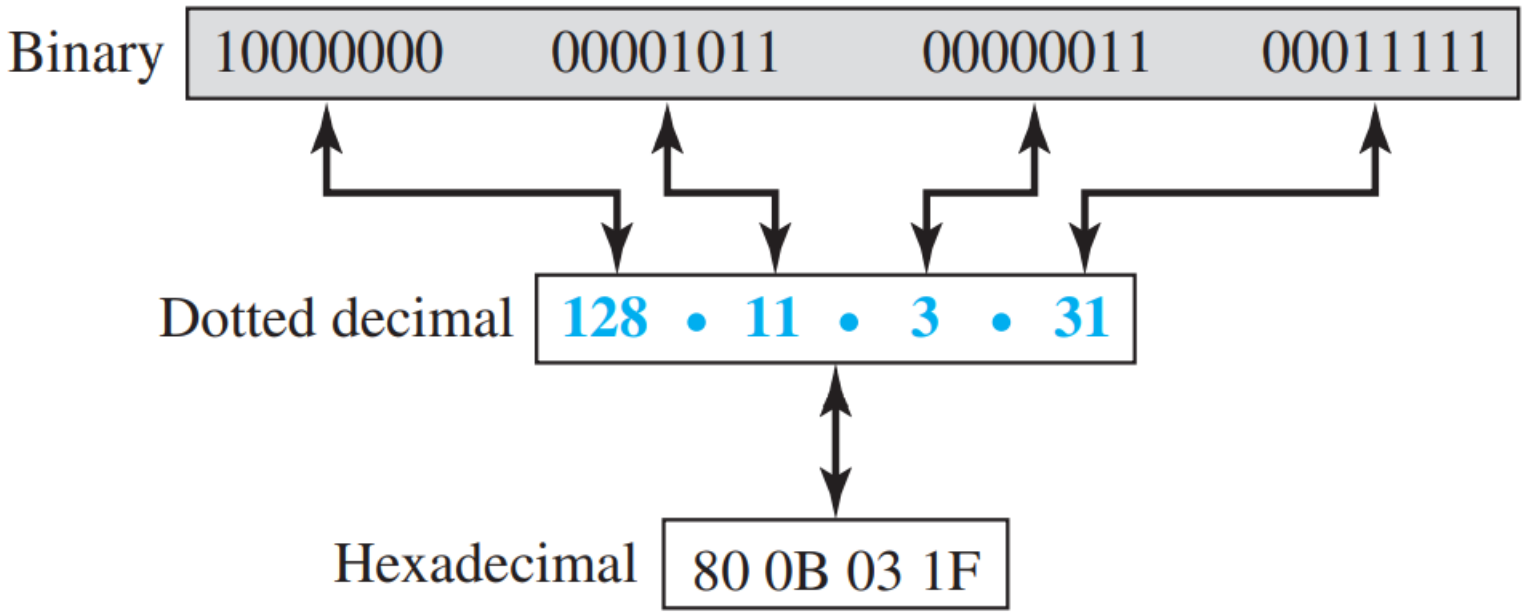
a device has two connections to the Internet, via two networks, it has two IPv4 addresses

Address Space

-32 BITS –2^32 or 4,294,967,296
Notation
dotted-decimal notation
each number in the dotted-decimal notation is between 0 and 255
32-bit address has 8 hexadecimal digits

IPV4 ADDRESSES

Figure 18.16 *Three different notations in IPv4 addressing*



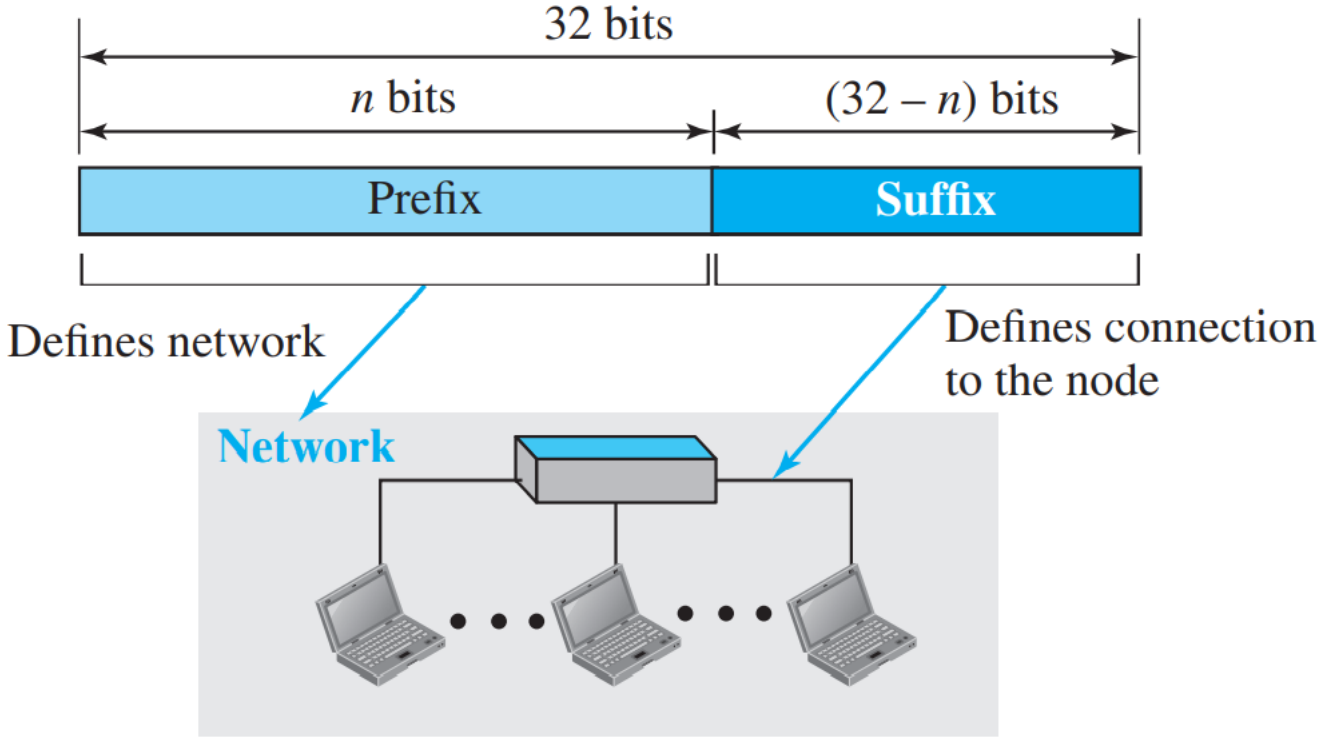
Hierarchy in Addressing

In a **postal network**, the postal address (mailing address) includes the **country, state, city, street, house number and recip name**

a **telephone number** is divided into the country code, area code, local exchange, and the connection

A 32-bit IPv4 address is also hierarchical, but divided only into two parts. The first part of the address, called the **prefix, defines the network**; **the second part of the address, called the suffix**, defines the node (connection of a device to the Internet).

Hierarchy in addressing



ADDRESSING

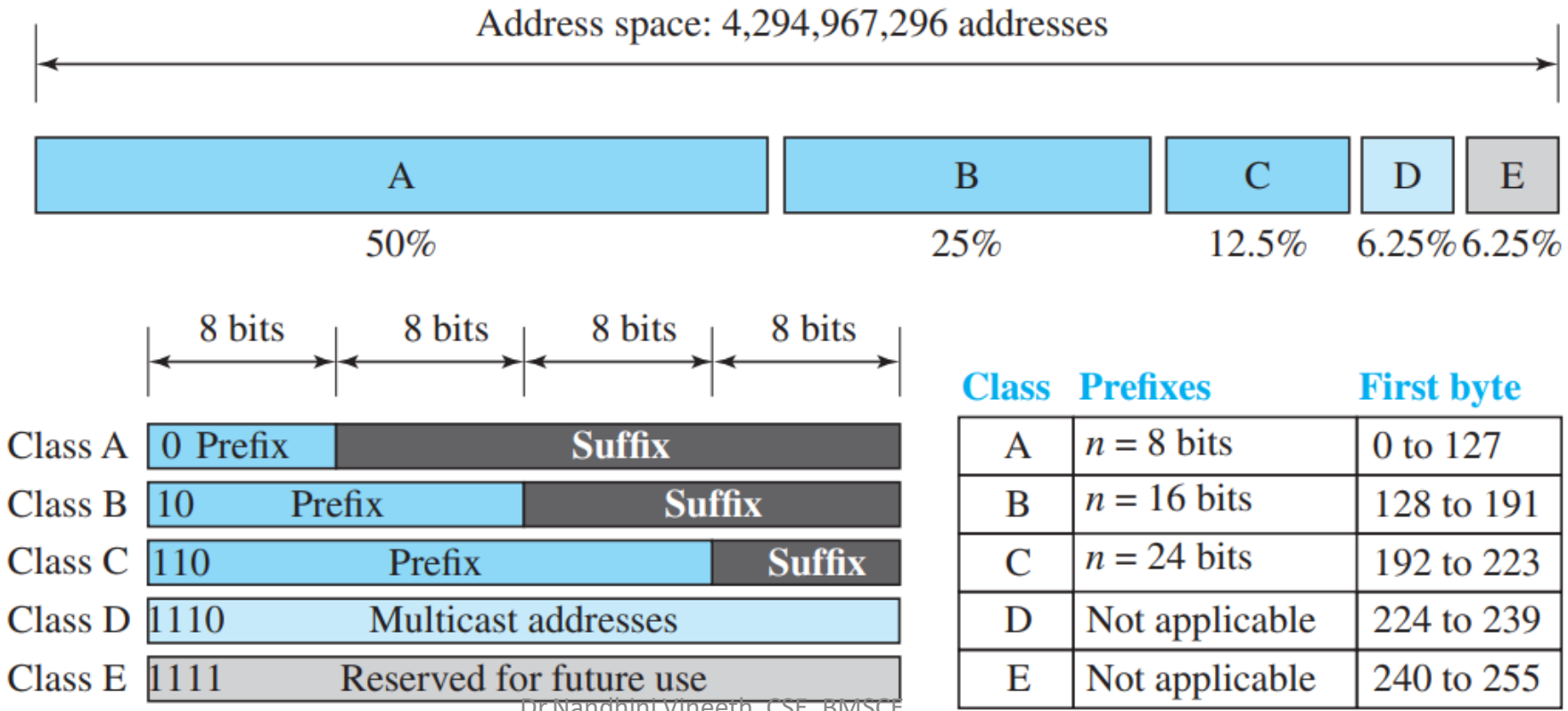
Classful addressing –Fixed length prefixes
Classless addressing – Variable length prefixes

The whole address space was divided into five classes (class A, B, C, D, and E).

This scheme is referred to as **classful addressing**.

CLASSFUL ADDRESSING

Figure 18.18 *Occupation of the address space in classful addressing*



Class	Prefixes	First byte
A	$n = 8$ bits	0 to 127
B	$n = 16$ bits	128 to 191
C	$n = 24$ bits	192 to 223
D	Not applicable	224 to 239
E	Not applicable	240 to 255

Classful addressing

Address Depletion:

Classful addressing has become obsolete because of address depletion.

Since the addresses were not distributed properly, shortage of IP addresses were faced by both organization and individuals

Ex. Class A – 8 + 24 -> 128 (2^7) organizations were given (2^24-16,777,216 nodes)

Same with Class B, reversed with Class C – very less IP address for each network

Classful addressing

Subnetting and Supernetting:

divide a large block into smaller ones

In subnetting, a class A or class B block is divided into **several subnets**.

Each subnet has a larger prefix length than the original network.

For example, if a network in class A is divided into four subnets, each subnet has a prefix of nsub = 10.

This idea did not work because most large organizations were not happy about dividing the block and giving some of the unused addresses to smaller organizations.

Supernetting:

supernetting was devised to combine several class C blocks into a larger block organizations that need more than the 256 addresses available in a class C block.

This idea did not work either because it makes the routing of packets more difficult.

Advantage of Classful Addressing:

Given an address, we can easily find the class of the address and, since the prefix length for each class is fixed, we can find the prefix length immediately.

Dynamic Host Configuration Protocol (DHCP)

A large organization or an ISP can receive a block of addresses directly from ICANN (Internet Corporation for Assigned Names and Numbers) and a small organization can receive a block of addresses from an ISP

address assignment in an organization can be done manually by network administrator or when huge automatically using the Dynamic Host Configuration Protocol (DHCP).

DHCP is an application-layer program, using the client-server paradigm, that actually helps TCP/IP at the network layer.

often called a plug and-play protocol.

A network manager can configure DHCP to assign permanent IP addresses to the host and routers.

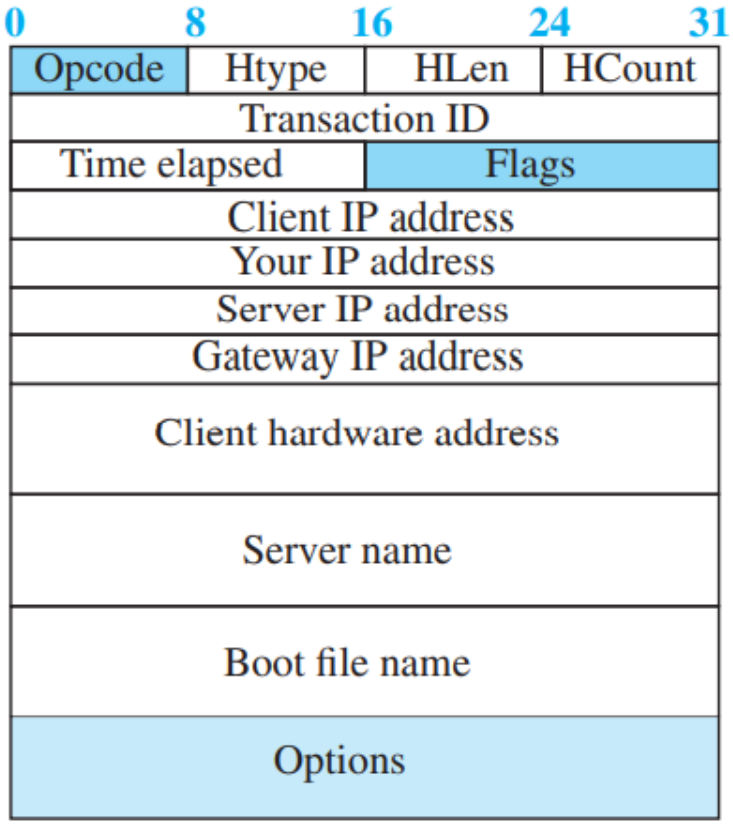
DHCP can also be configured to provide temporary, on demand, IP addresses to hosts. Ex. provide temporary IP address to a traveller to connect her laptop to the Internet while she is staying in the hotel.

four pieces of information are normally needed: the computer address, the prefix, the address of a router, and the IP address of a name server. DHCP can be used to provide these pieces of information to the host.

Dynamic Host Configuration Protocol (DHCP)

DHCP Message Format

Figure 18.25 *DHCP message format*



Fields:

Opcode: Operation code, request (1) or reply (2)
Htype: Hardware type (Ethernet, ...)
HLen: Length of hardware address
HCount: Maximum number of hops the packet can travel
Transaction ID: An integer set by the client and repeated by the server
Time elapsed: The number of seconds since the client started to boot
Flags: First bit defines unicast (0) or multicast (1); other 15 bits not used
Client IP address: Set to 0 if the client does not know it
Your IP address: The client IP address sent by the server
Server IP address: A broadcast IP address if client does not know it
Gateway IP address: The address of default router
Server name: A 64-byte domain name of the server
Boot file name: A 128-byte file name holding extra information
Options: A 64-byte field with dual purpose described in text

Dynamic Host Configuration Protocol (DHCP)

The 64-byte option field has a dual purpose.
It can carry either additional information or some specific vendor information.
The server uses a number, called a magic cookie, in the format of an IP address with the value of 99.130.83.99.
When the client finishes reading the message, it looks for this magic cookie.
If present, the next 60 bytes are options.
An option is composed of three fields: a 1-byte tag field, a 1-byte length field, and a variable-length value field.
There are several tag fields that are mostly used by vendors. If the tag field is 53, the value field defines one of the 8 message types shown in Figure 18.26. We show how these message types are used by DHCP.

Figure 18.26 *Option format*

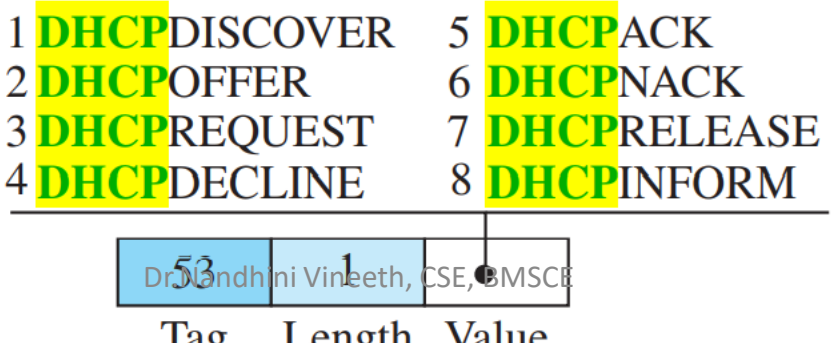
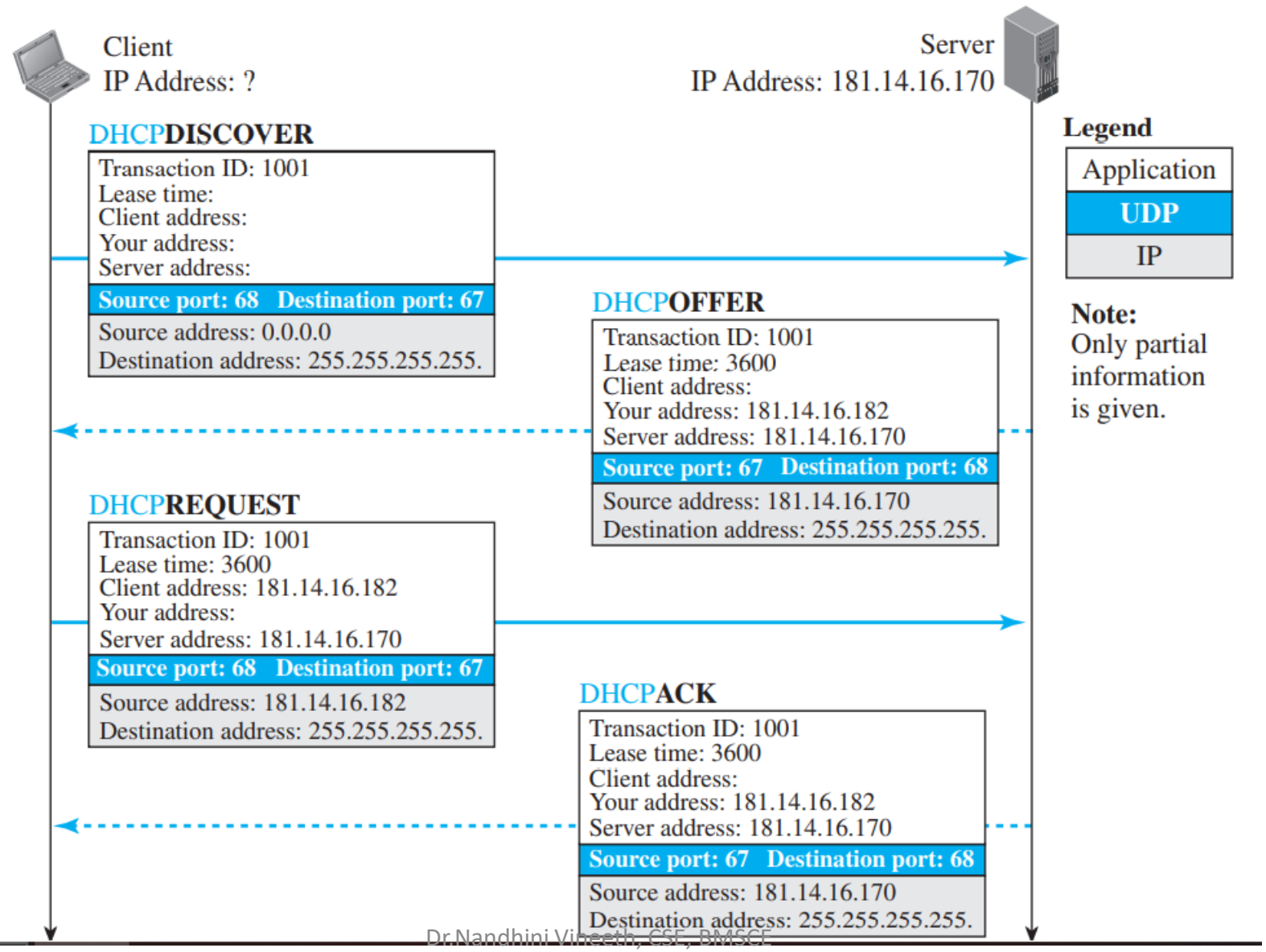


Figure 18.27 *Operation of DHCP*



Dynamic Host Configuration Protocol (DHCP)

Two Well-Known Ports

The reason for choosing the well-known port 68 instead of an ephemeral port for the client is that the response from the server to the client is broadcast

Any other systems using this ephemeral port accidentally should not be confused

Using FTP

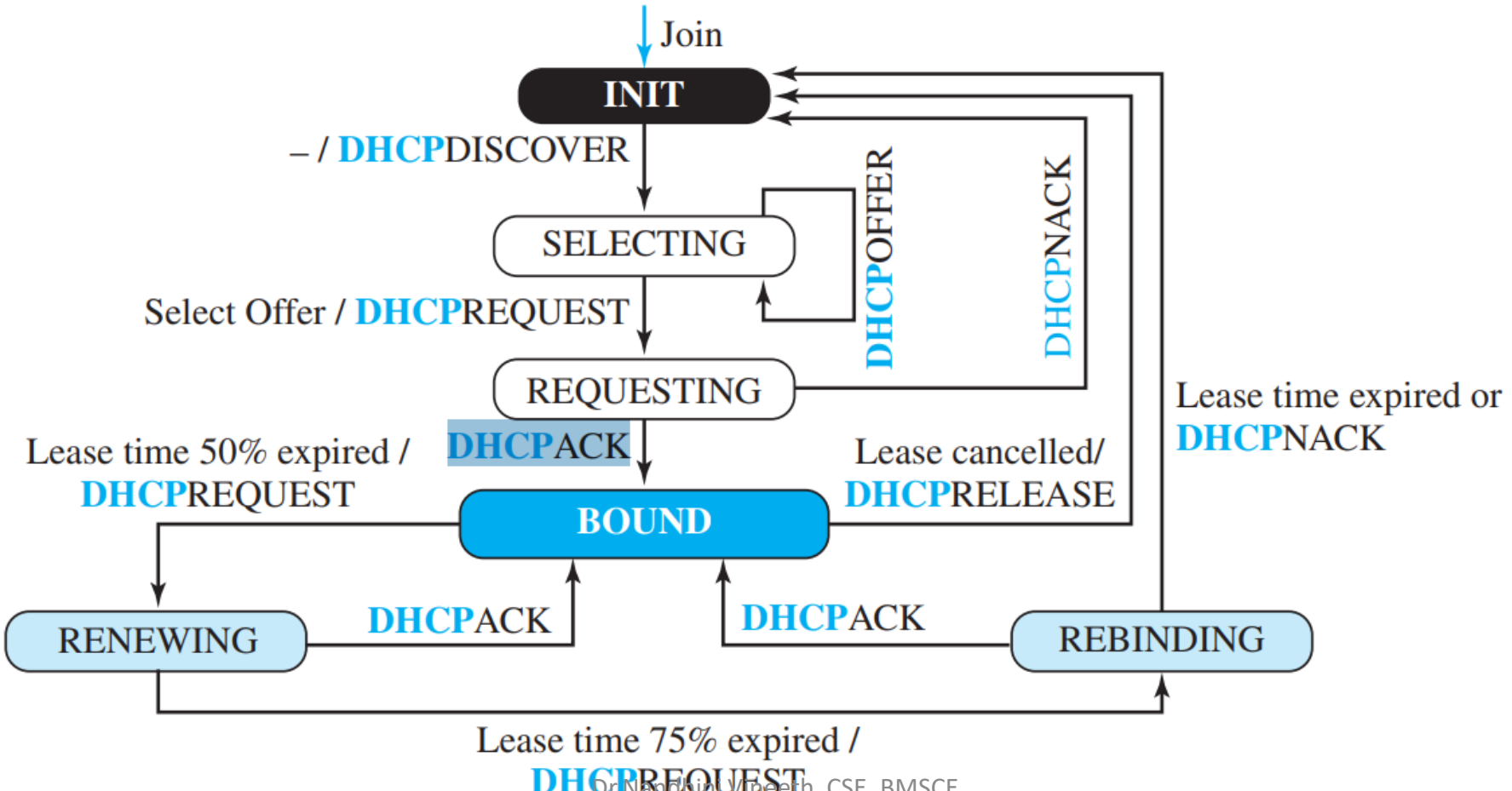
In the DHCPACK message, the server defines the pathname of a file in which the client can find complete information such as the address of the DNS server. The client can then use a file transfer protocol to obtain the rest of the needed information.

Error Control

DHCP uses the service of UDP, which is not reliable. To provide error control, DHCP uses two strategies.
First, DHCP requires that UDP use the checksum- the use of the checksum in UDP is optional.
Second, the DHCP client uses timers and a retransmission policy if it does not receive the DHCP reply to a request.

DHCP - Transition States

Figure 18.28 *FSM for the DHCP client*



Network Address Resolution (NAT)

Assume that an ISP has granted a small range of addresses to a small business or a household.

If the business grows or the household needs a larger range, the ISP may not be able to grant the demand because the addresses before and after the range may have already been allocated to other networks.

In most situations, however, only a portion of computers in a small network need access to the Internet simultaneously.

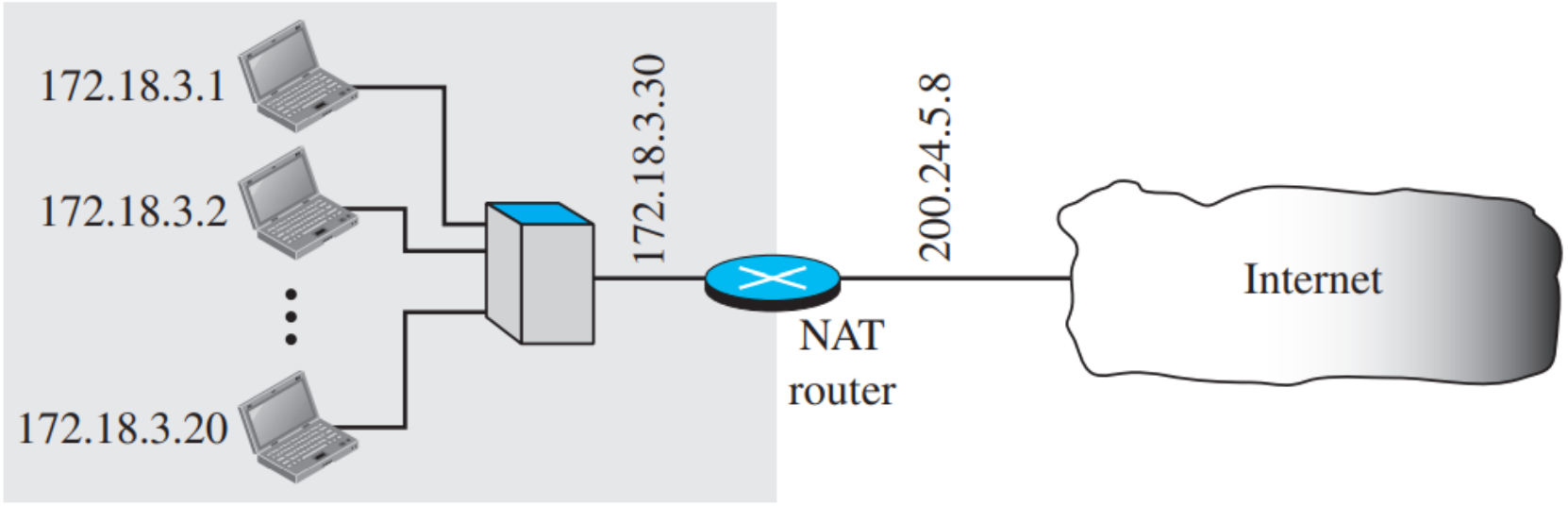
The business can use 20 (or 25) addresses from the private block addresses (discussed before) for internal communication; five addresses for universal communication can be assigned by the ISP.

A technology that can provide the mapping between the private and universal addresses, and at the same time support virtual private networks is **Network Address Translation (NAT).**

The site must have only one connection to the global Internet through a NAT-capable router that runs NAT software

Network Address Translation (NAT)

Figure 18.29 *NAT*



Network Address Translation (NAT)

Figure 18.30 *Address translation*

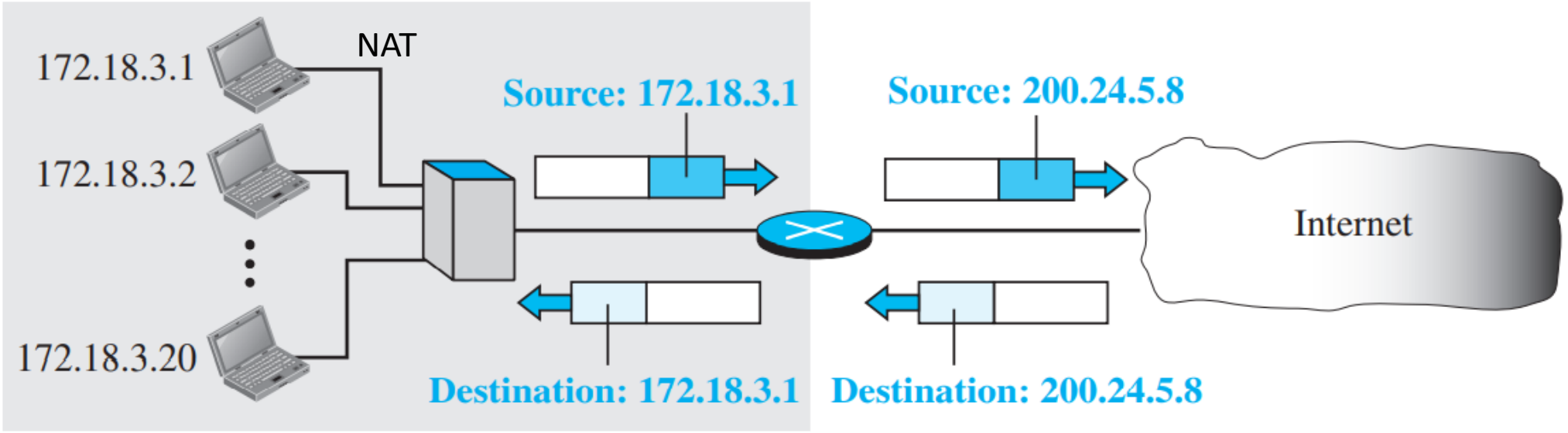
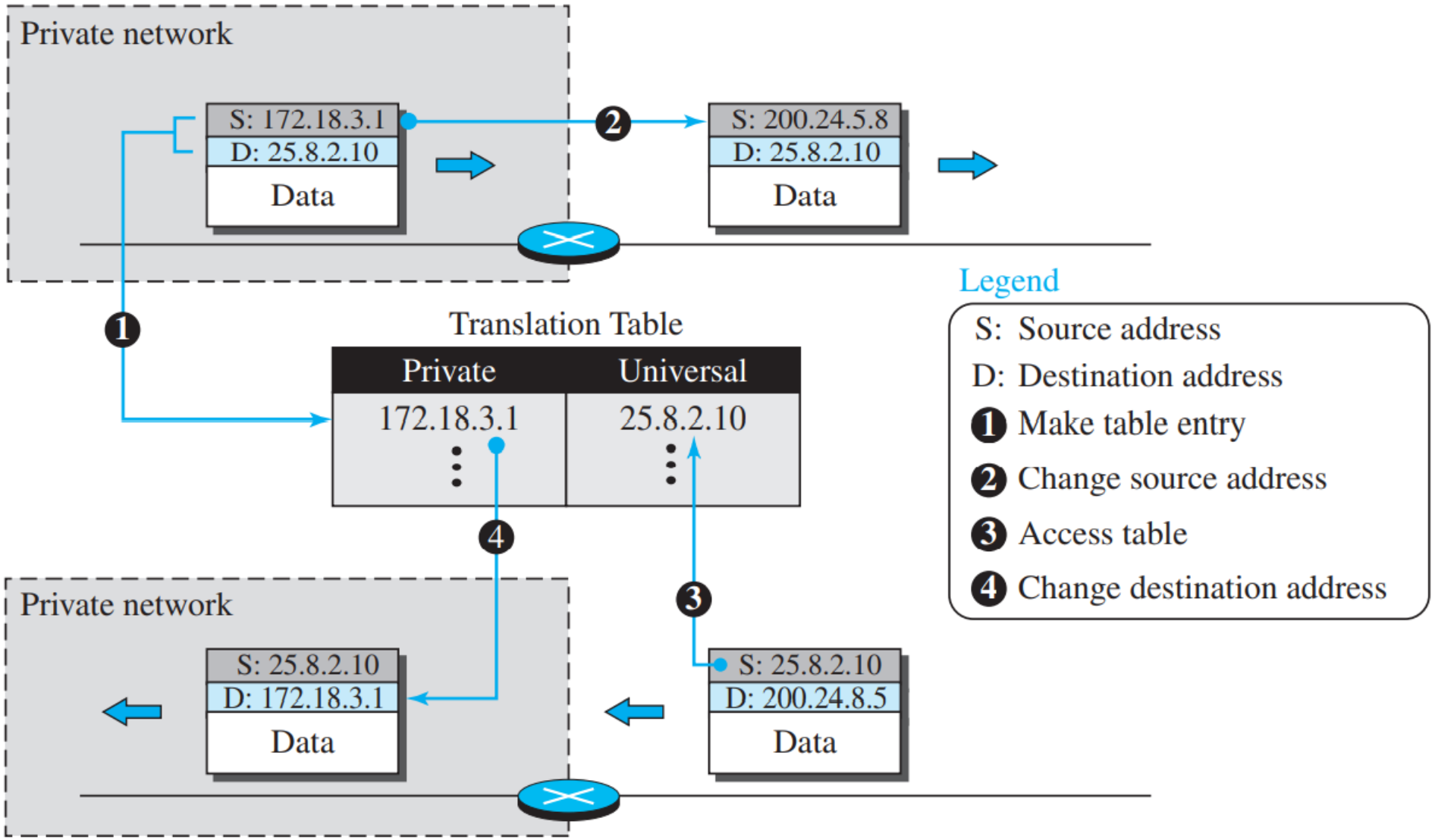


Figure 18.31 *Translation*



Using a Pool of IP Addresses

The use of only one global address by the NAT router allows only one private-network host to access a given external host. To remove this restriction, the NAT router can use a pool of global addresses.

Instead of using only one global address (200.24.5.8), the NAT router can use four addresses (200.24.5.8, 200.24.5.9, 200.24.5.10, and 200.24.5.11).

Four private-network hosts can communicate with the same external host at the same time because each pair of addresses defines a separate connection.

Drawbacks:

No more than four connections can be made to the same destination.

No private-network host can access two external server programs (e.g., HTTP and TELNET) at the same time.

two private-network hosts cannot access the same external server program (e.g., HTTP or TELNET) at the same time.

Using Both IP Addresses and Port Addresses

Table 18.1 *Five-column translation table*

Private address	*Private port*	*External address*	*External port*	*Transport protocol*
172.18.3.1	1400	25.8.3.2	80	TCP
172.18.3.2	1401	25.8.3.2	80	TCP
⋮	⋮	⋮	⋮	⋮

Classless Addressing

The larger address space, however, requires that the **length of IP addresses also be increased**, which means the format of the IP packets needs to be changed.

Although the long-range solution has already been devised and is called **IPv6**

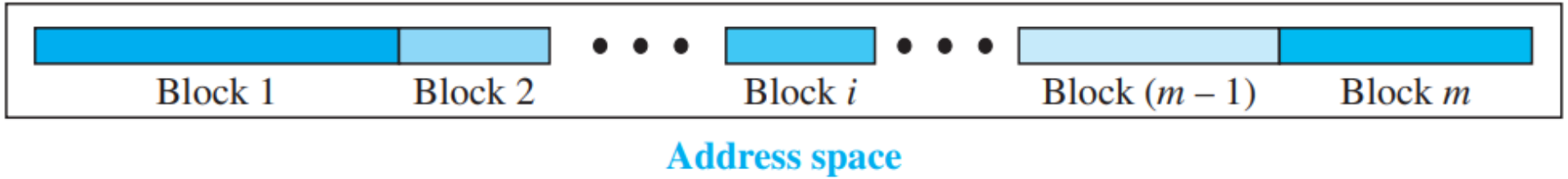
The short-term solution still uses IPv4 addresses, but it is called **classless addressing.**

Internet authorities announced a new architecture called classless addressing.
In classless addressing, **variable-length blocks** are used that belong to no classes.
block of 1 address, 2 addresses, 4 addresses, 128 addresses, and so on.
the number of addresses in a block needs to be a power of 2.

The size of the network is inversely proportional to the length of the prefix.
A small prefix means a larger network; a large prefix means a smaller network.

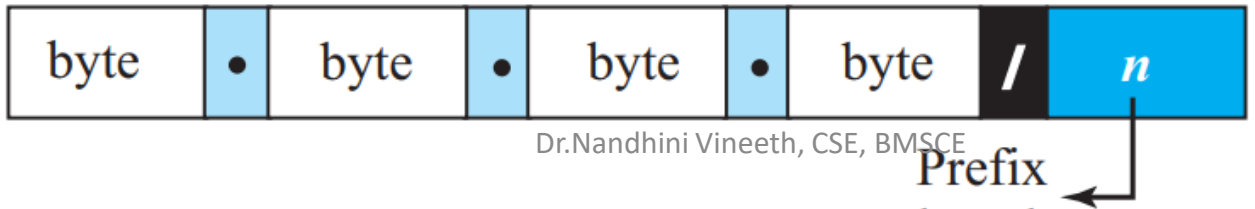
Classless Addressing

Figure 18.19 *Variable-length blocks in classless addressing*



Prefix Length: Slash Notation or **classless interdomain routing or CIDR**

Figure 18.20 *Slash notation (CIDR)*



Examples:
12.24.76.8/**8**
23.14.67.92/**12**
220.8.24.255/**25**

Classless Addressing

Extracting Information from an Address

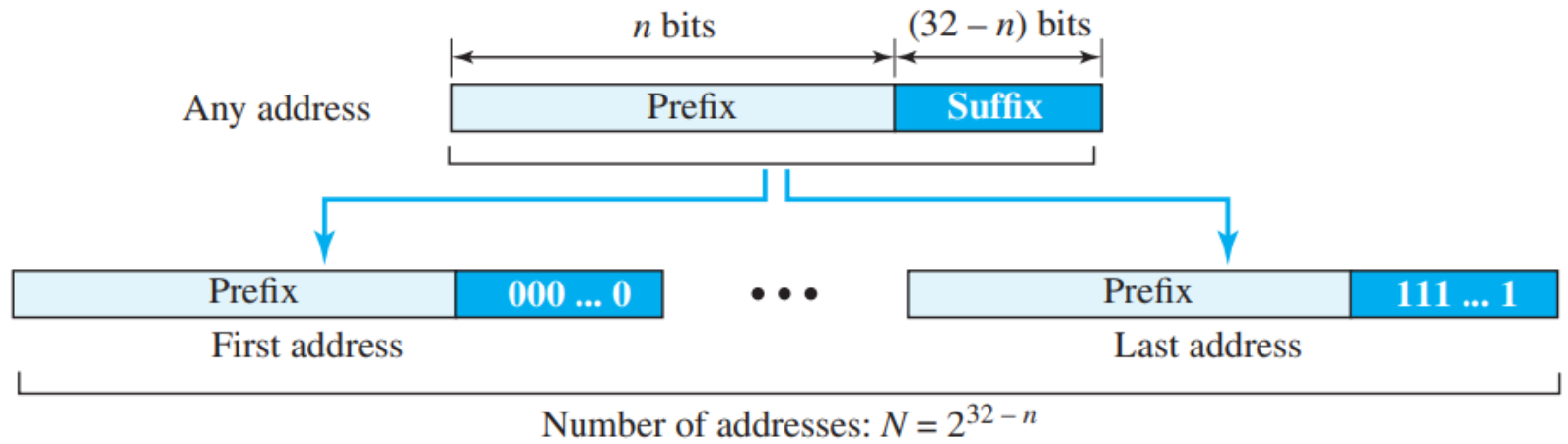
Three pieces of information are important

the number of addresses,
the first address in the block, and
the last address

1. The number of addresses in the block is found as N = 2^(32−n) .
2. To find the first address, we keep the n leftmost bits and set the (32 − n) rightmost bits all to 0s.
3. To find the last address, we keep the n leftmost bits and set the (32 − n) rightmost bits all to 1s.

A classless address is given as **167.199.170.82/27.** We can find the above three pieces of information as follows. The number of addresses in the network is 2^(32 − n) = 2^5 = 32 addresses.

Figure 18.21 *Information extraction in classless addressing*



The first address can be found by keeping the first 27 bits and changing the rest of the bits to 0s.

Address: 167.199.170.82/27	10100111	11000111	10101010	01010010
First address: 167.199.170.64/27	10100111	11000111	10101010	01000000

The last address can be found by keeping the first 27 bits and changing the rest of the bits to 1s.

Address: 167.199.170.82/27	10100111	11000111	10101010	01011111
Last address: 167.199.170.95/27	10100111	11000111	10101010	01011111

Classless Addressing

Address Mask

Another way to find the first and last addresses in the block is to use the address mask.

The address mask is a 32-bit number in which the n leftmost bits are set to 1s and the rest of the bits (32 − n) are set to 0s. A computer can easily find the address mask because it is the complement of (2^(32 − n) − 1)).

The reason for defining a mask in this way is that it can be used by a computer program to extract the information in a block, using the three bit-wise operations NOT, AND, and OR.

1. The number of addresses in the block N = NOT (mask) + 1.
2. The first address in the block = (Any address in the block) AND (mask).
3. The last address in the block = (Any address in the block) OR [(NOT (mask)].

$$\text{first address} = (\text{prefix in decimal}) \times 2^{32-n} = (\text{prefix in decimal}) \times N.$$

Example 18.2

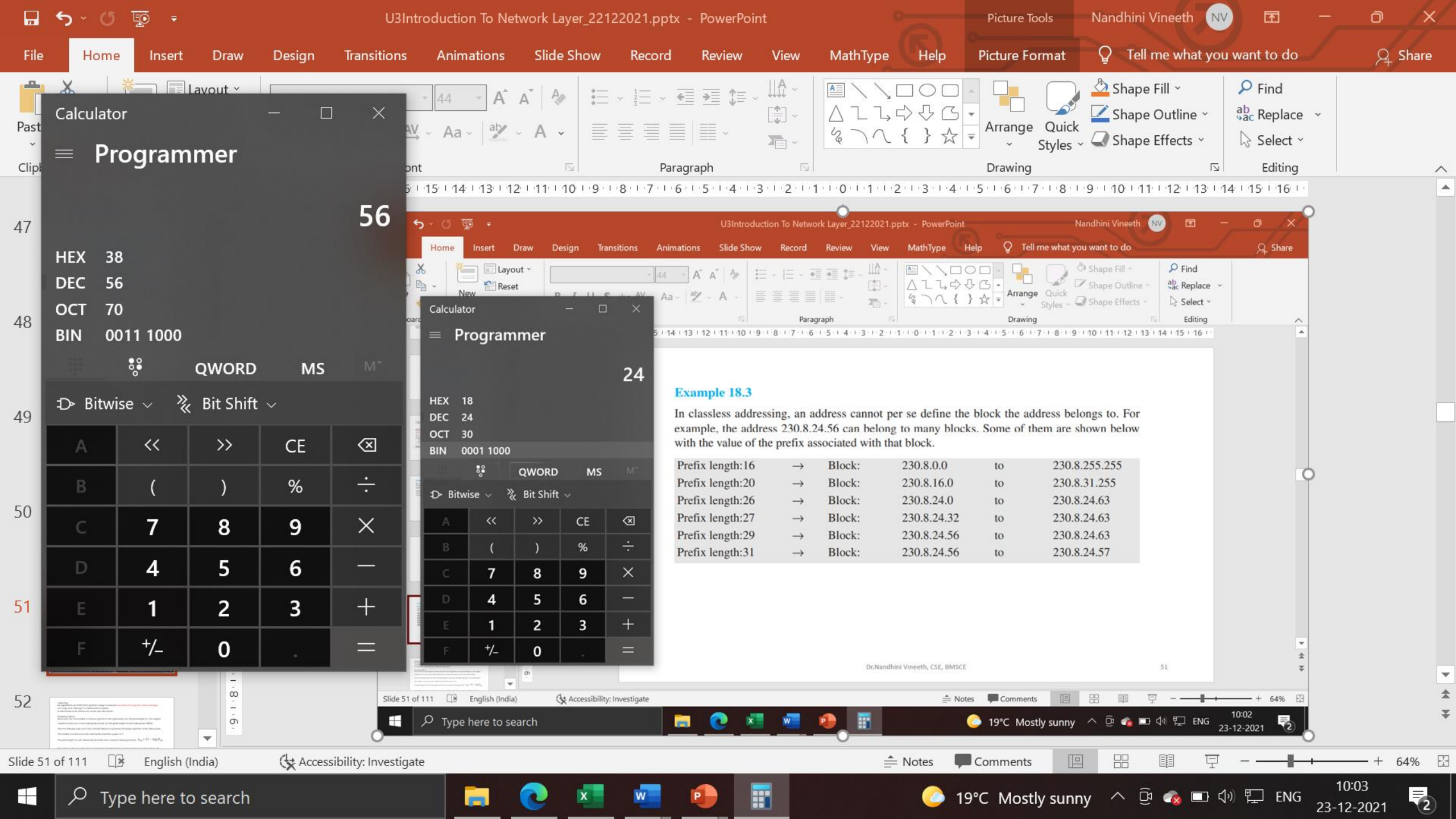
We repeat Example 18.1 using the mask. The mask in dotted-decimal notation is 256.256.256.224. The AND, OR, and NOT operations can be applied to individual bytes using calculators and applets at the book website.

Number of addresses in the block:	N = **NOT** (mask) + 1= 0.0.0.31 + 1 = 32 addresses
First address:	First = (address) **AND** (mask) = 167.199.170.82
Last address:	Last = (address) **OR** (**NOT** mask) = 167.199.170.255

Example 18.3

In classless addressing, an address cannot per se define the block the address belongs to. For example, the address 230.8.24.56 can belong to many blocks. Some of them are shown below with the value of the prefix associated with that block.

Prefix length:16	→	Block:	230.8.0.0	to	230.8.255.255
Prefix length:20	→	Block:	230.8.16.0	to	230.8.31.255
Prefix length:26	→	Block:	230.8.24.0	to	230.8.24.63
Prefix length:27	→	Block:	230.8.24.32	to	230.8.24.63
Prefix length:29	→	Block:	230.8.24.56	to	230.8.24.63
Prefix length:31	→	Block:	230.8.24.56	to	230.8.24.57


U3Introduction To Network Layer_22122021.pptx - PowerPoint
Nandhini Vineeth
Picture Tools
File Home Insert Draw Design Transitions Animations Slide Show Record Review View MathType Help Picture Format
Tell me what you want to do
Share
Calculator
Programmer
56
HEX 38
DEC 56
OCT 70
BIN 0011 1000
QWORD MS
Bitwise Bit Shift
A << >> CE
B ( ) % ÷
C 7 8 9 ×
D 4 5 6 −
E 1 2 3 +
F +/− 0 . =
Calculator
Programmer
24
HEX 18
DEC 24
OCT 30
BIN 0001 1000
QWORD MS
Bitwise Bit Shift
Example 18.3
In classless addressing, an address cannot per se define the block the address belongs to. For example, the address 230.8.24.56 can belong to many blocks. Some of them are shown below with the value of the prefix associated with that block.
Prefix length:16 → Block: 230.8.0.0 to 230.8.255.255
Prefix length:20 → Block: 230.8.16.0 to 230.8.31.255
Prefix length:26 → Block: 230.8.24.0 to 230.8.24.63
Prefix length:27 → Block: 230.8.24.32 to 230.8.24.63
Prefix length:29 → Block: 230.8.24.56 to 230.8.24.63
Prefix length:31 → Block: 230.8.24.56 to 230.8.24.57
Dr.Nandhini Vineeth, CSE, BMSCE
Slide 51 of 111
English (India)
Accessibility: Investigate
Notes
Comments
64%
Type here to search
19°C Mostly sunny
ENG
10:02
23-12-2021
Slide 51 of 111
English (India)
Accessibility: Investigate
Notes
Comments
64%
Type here to search
19°C Mostly sunny
ENG
10:03
23-12-2021

MASK - APPLED

230.8.24.56 / 20

1110 0110	0000 1000	0001 1000	0011 1000	
1111 1111	1111 1111	1111 0000	0000 0000	AND
1110 0110	0000 1000	0001 0000	0000 0000	
230.	8.	16.	0	

1110 0110	0000 1000	0001 1000	0011 1000	
0000 0000	0000 0000	0000 1111	1111 1111	OR(COMP)
1110 0110	0000 1000	0001 1111	1111 1111	
230.	8.	31.	255	

Subnetting

An organization (or an ISP) that is granted a range of addresses may divide the range into several subranges and assign each subrange to a subnetwork (or subnet).

A subnetwork can be divided into several sub-subnetworks

Designing Subnets

We assume the total number of addresses granted to the organization is N, the prefix length is n, the assigned number of addresses to each subnetwork is Nsub, and the prefix length for each subnetwork is nsub.

Then the following steps need to be carefully followed to guarantee the proper operation of the subnetworks.

The number of addresses in each subnetwork should be a power of 2.

The prefix length for each subnetwork should be found using the following formula: $n_{sub} = 32 - \log_2 N_{sub}$

The starting address in each subnetwork should be divisible by the number of addresses in that subnetwork. This can be achieved if we first assign addresses to larger subnetworks.

Example 18.5

An organization is granted a block of addresses with the beginning address 14.24.74.0/24. The organization needs to have 3 subblocks of addresses to use in its three subnets: one subblock of 10 addresses, one subblock of 60 addresses, and one subblock of 120 addresses. Design the subblocks.

Solution

There are $2^{32-24} = 256$ addresses in this block. The first address is 14.24.74.0/24; the last address is 14.24.74.255/24. To satisfy the third requirement, we assign addresses to subblocks, starting with the largest and ending with the smallest one.

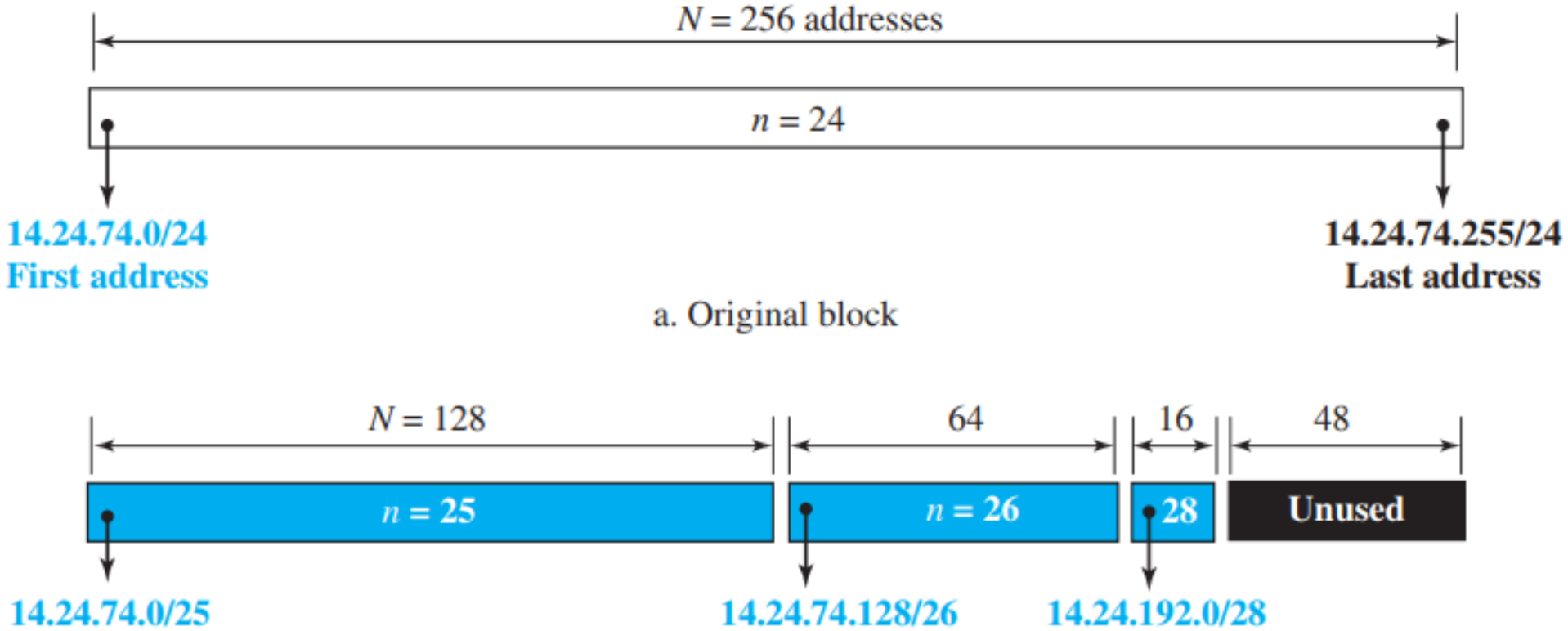
a. The number of addresses in the largest subblock, which requires 120 addresses, is not a power of 2. We allocate 128 addresses. The subnet mask for this subnet can be found as $n_1 = 32 - \log_2 128 = 25$. The first address in this block is 14.24.74.0/25; the last address is 14.24.74.127/25.

b. The number of addresses in the second largest subblock, which requires 60 addresses, is not a power of 2 either. We allocate 64 addresses. The subnet mask for this subnet can be found as $n_2 = 32 - \log_2 64 = 26$. The first address in this block is 14.24.74.128/26; the last address is 14.24.74.191/26.

c. The number of addresses in the smallest subblock, which requires 10 addresses, is not a power of 2 either. We allocate 16 addresses. The subnet mask for this subnet can be found as $n_3 = 32 - \log_2 16 = 28$. The first address in this block is 14.24.74.192/28; the last address is 14.24.74.207/28.

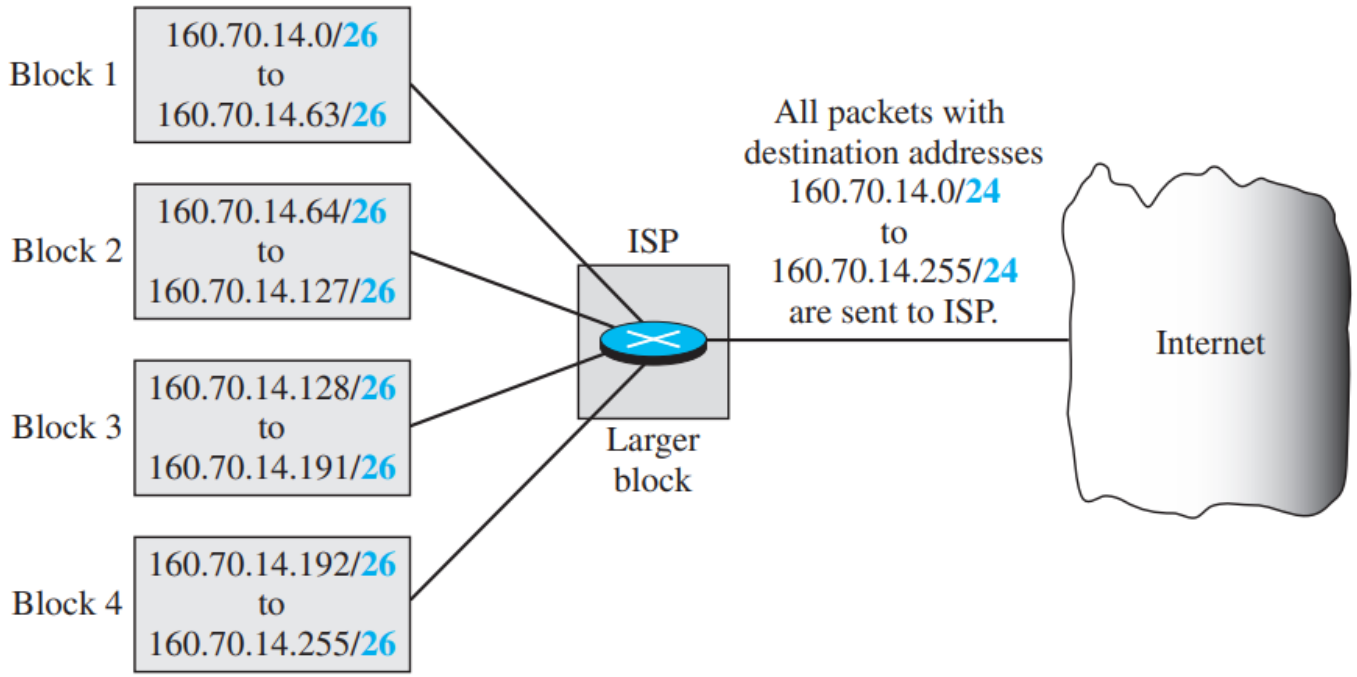
If we add all addresses in the previous subblocks, the result is 208 addresses, which means 48 addresses are left in reserve. The first address in this range is 14.24.74.208. The last address is 14.24.74.255. We don't know about the prefix length yet. Figure 18.23 shows the configuration of blocks. We have shown the first address in each block.

Figure 18.23 *Solution to Example 18.5*



Address Aggregation One of the advantages of the CIDR strategy is address aggregation (sometimes called address summarization or route summarization). When blocks of addresses are combined to create a larger block, routing can be done based on the prefix of the larger block. ICANN assigns a large block of addresses to an ISP. Each ISP in turn divides its assigned block into smaller subblocks and grants the subblocks to its customers

Figure 18.24 *Example of address aggregation*



Special Addresses

we need to mention five special addresses that are used for special purposes: **this-host address, limited-broadcast address, loopback address, private addresses, and multicast addresses**.

This-host Address The only address in the block 0.0.0.0/32 is called the this-host address. It is used whenever a host needs to send an IP datagram but it does not know its own address to use as the source address

Limited-broadcast Address The only address in the block 255.255.255.255/32 is called the limited-broadcast address. It is used whenever a router or a host needs to send a datagram to all devices in a network. The routers in the network, however, block the packet having this address as the destination; the packet cannot travel outside the network.

Loopback Address The block 127.0.0.0/8 is called the loopback address. A packet with one of the addresses in this block as the destination address never leaves the host; it will remain in the host. Any address in the block is used to test a piece of software in the machine. For example, we can write a client and a server program in which one of the addresses in the block is used as the server address. We can test the programs using the same host to see if they work before running them on different computers.

Private Addresses Four blocks are assigned as private addresses: 10.0.0.0/8, 172.16.0.0/12, 192.168.0.0/16, and 169.254.0.0/16. We will see the applications of these addresses when we discuss NAT later in the chapter.

Multicast Addresses The block 224.0.0.0/4 is reserved for multicast addresses

INTRODUCTION

An Internet as a Graph

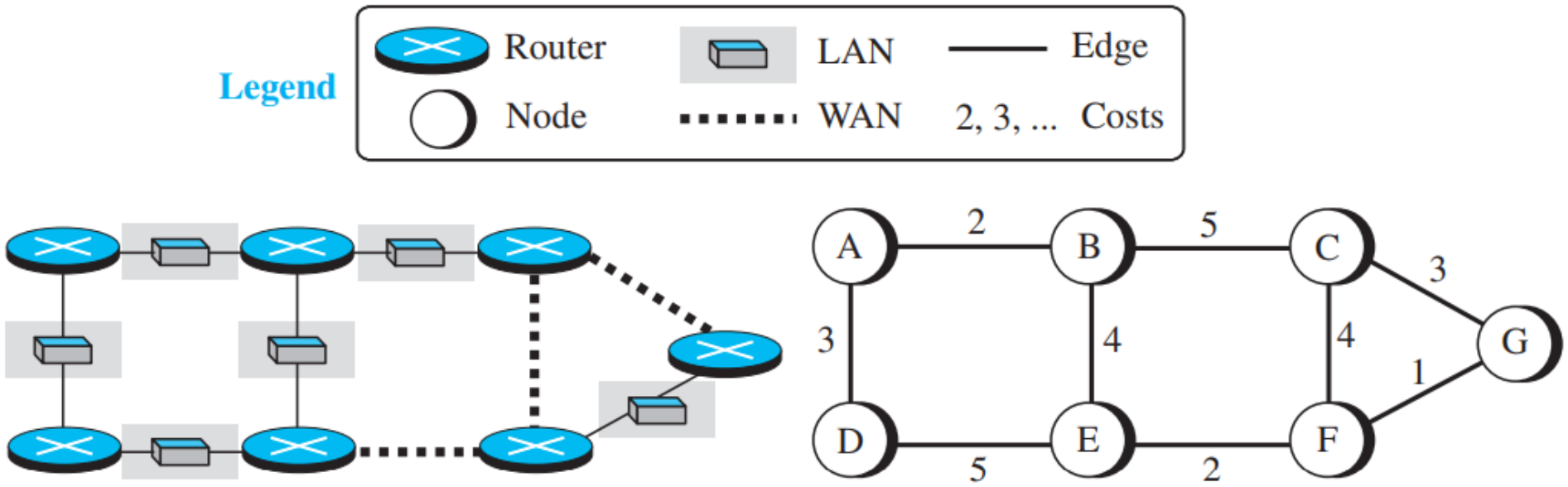
- To find the best route, an internet can be modeled as a graph.
- A graph in networks is a set of nodes and edges (lines) that connect the nodes.
- we can think of each router as a node and each network between a pair of routers as an edge.
- An internet modeled as a weighted graph, in which each edge is associated with a cost.

- If a weighted graph is used to represent a geographical area, the nodes can be cities and the edges can be roads connecting the cities; the weights, in this case, are distances between cities.

- In routing, however, the cost of an edge has a different interpretation in different routing protocols

Least-Cost Routing

Figure 20.1 *An internet and its graphical representation*



a. An internet

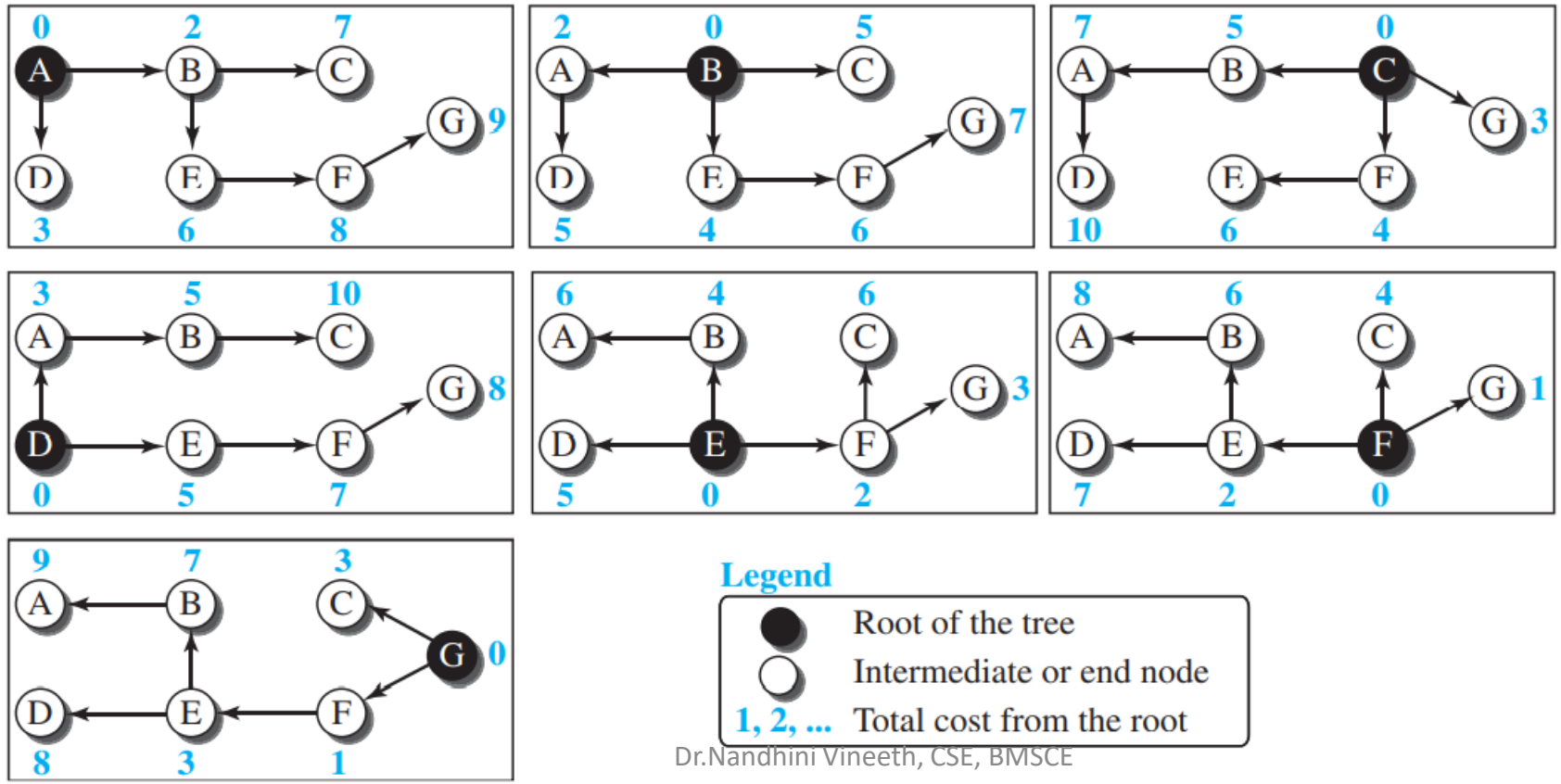
b. The weighted graph

to interpret the best route from the source router to the destination router is to find the least cost between the two

Least-Cost Trees

If there are N routers in an internet, there are (N − 1) least-cost paths from each router to any other router. This means we need N × (N − 1) least-cost paths for the whole internet.

Figure 20.2 *Least-cost trees for nodes in the internet of Figure 20.1*



ROUTING ALGORITHMS

Distance-Vector Routing

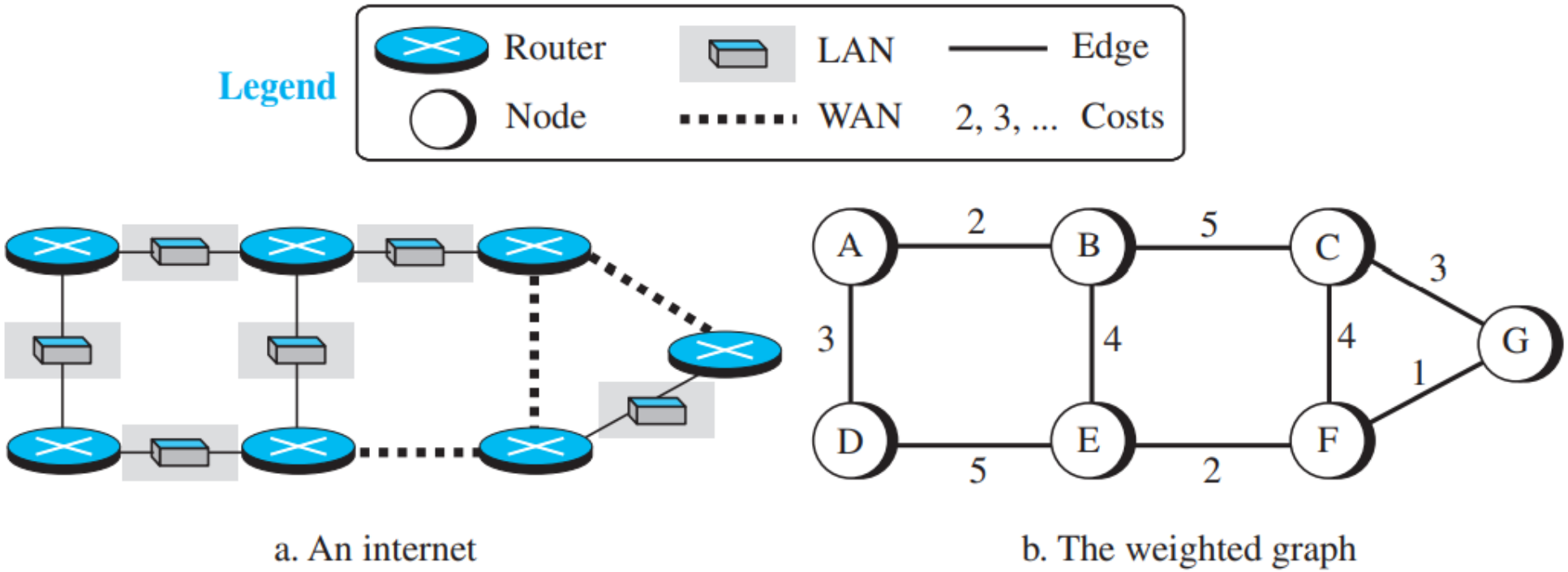
In distance-vector routing, the first thing each node creates is its own least-cost tree with the rudimentary information it has about its immediate neighbors.

Bellman-Ford Equation:
The heart of distance-vector routing is the famous **Bellman-Ford equation**.

This equation is used to **find the least cost (shortest distance) between a source node, x, and a destination node, y**, through some intermediary nodes (a, b, c, . . .) when the costs between the source and the intermediary nodes and the least costs between the intermediary nodes and the destination are given.

ROUTING ALGORITHMS

Figure 20.1 *An internet and its graphical representation*



a. An internet

b. The weighted graph

Bellman-Ford

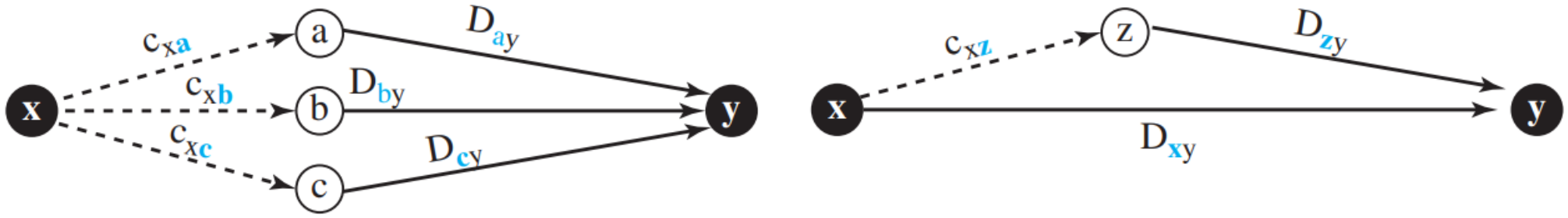
D_{ij} is the shortest distance and c_{ij} is the cost between nodes i and j.

$$D_{xy} = \min\{(c_{xa} + D_{ay}), (c_{xb} + D_{by}), (c_{xc} + D_{cy}), \ldots\}$$

we want to update an existing least cost with a least cost through an intermediary node, such as z

$$D_{xy} = \min\{D_{xy}, (c_{xz} + D_{zy})\}$$

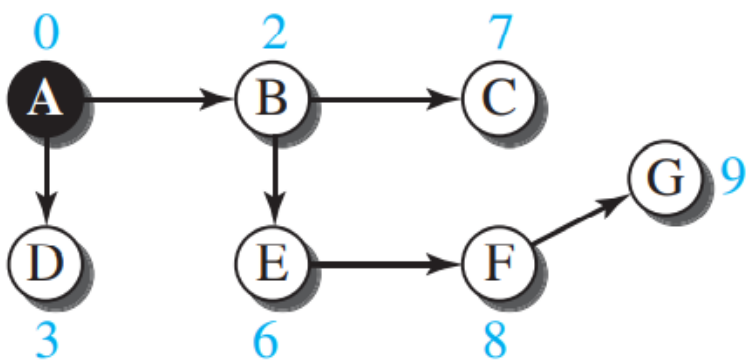
Figure 20.3 *Graphical idea behind Bellman-Ford equation*



a. General case with three intermediate nodes

b. Updating a path with a new route

Figure 20.4 *The distance vector corresponding to a tree*



a. Tree for node A

	A
A	0
B	2
C	7
D	3
E	6
F	8
G	9

b. Distance vector for node A

Figure 20.5 *The first distance vector for an internet*

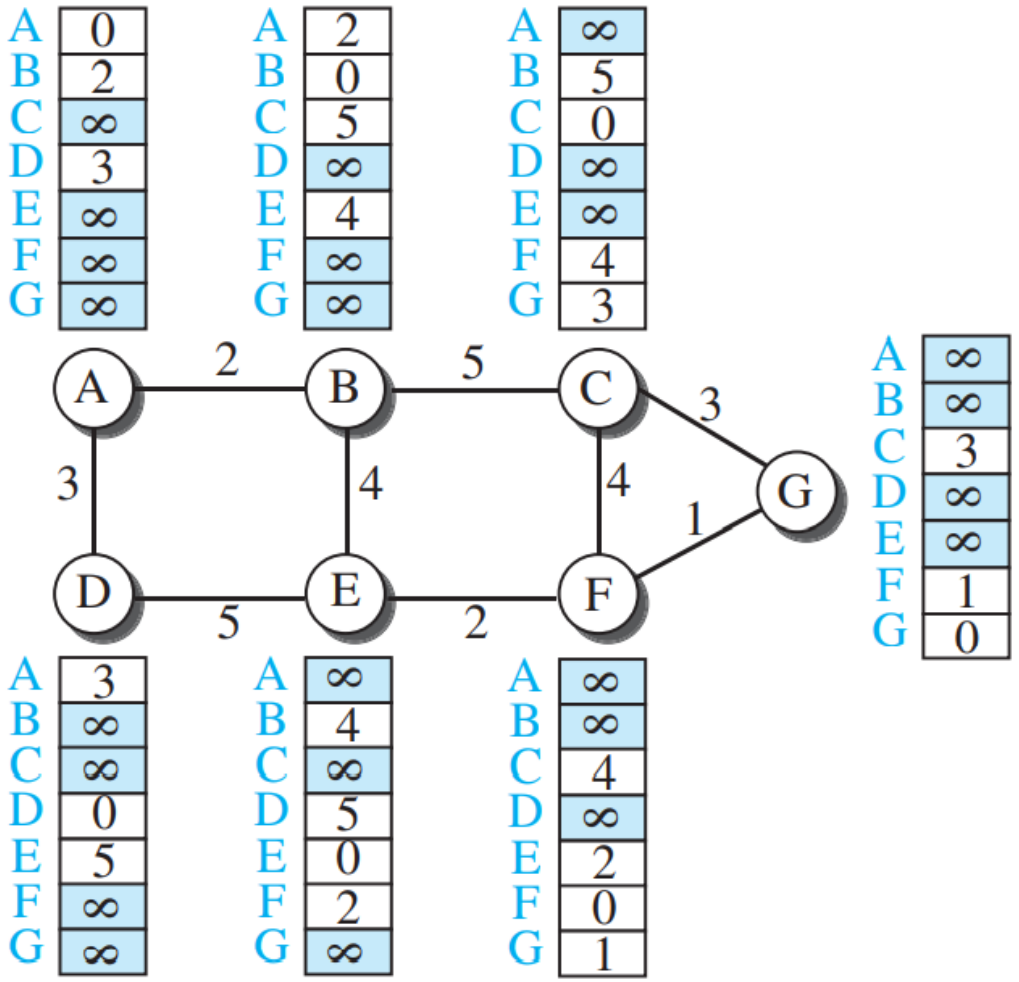
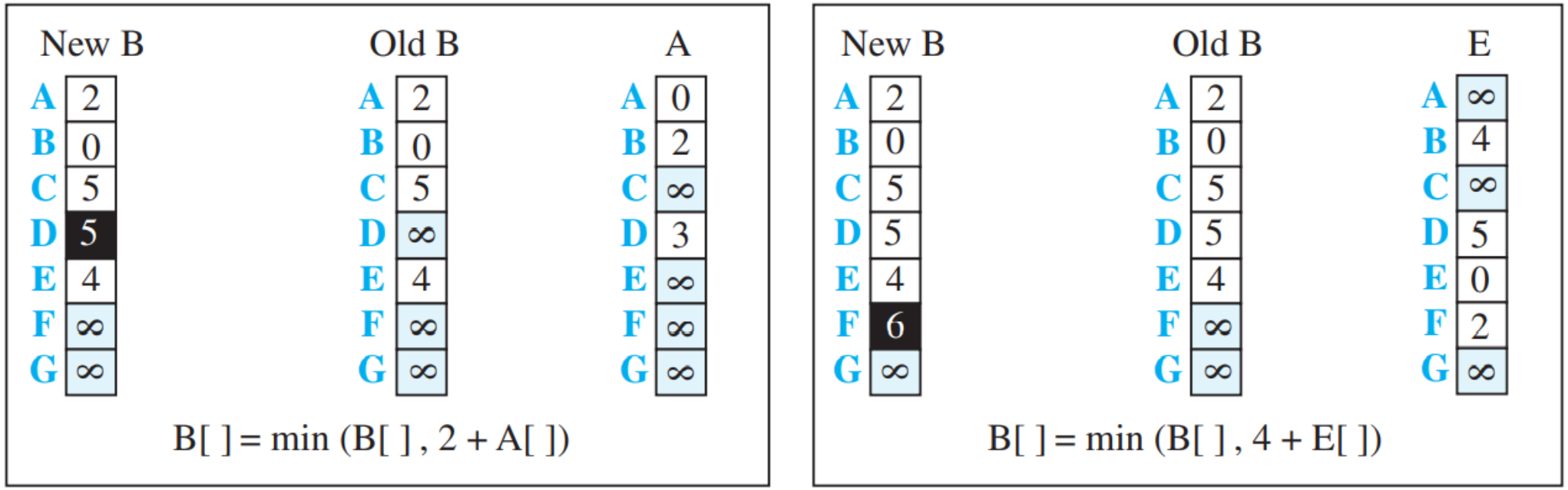


Figure 20.6 *Updating distance vectors*



a. First event: B receives a copy of A's vector.

b. Second event: B receives a copy of E's vector.

Note:
X[ ]: the whole vector

Table 20.1 *Distance-Vector Routing Algorithm for a Noa*

```
1   Distance_Vector_Routing ( )
2   {
3       // Initialize (create initial vectors for the node)
4       D[myself] = 0
```

Table 20.1 *Distance-Vector Routing Algorithm for a Node (continued)*

```
5       for (y = 1 to N)
6       {
7           if (y is a neighbor)
8               D[y] = c[myself][y]
9           else
10              D[y] = ∞
11      }
12      send vector {D[1], D[2], …, D[N]} to all neighbors
13      // Update (improve the vector with the vector received from a neighbor)
14      repeat (forever)
15      {
16          wait (for a vector D_w from a neighbor w or any change in the link)
17          for (y = 1 to N)
18          {
19              D[y] = min [D[y], (c[myself][w] + D_w[y ])]      // Bellman-Ford equation
20          }
21          if (any change in the vector)
22              send vector {D[1], D[2], …, D[N]} to all neighbors
23      }
24  } // End of Distance Vector
```

Count to Infinity

A problem with distance-vector routing is that any decrease in cost **(good news) propagates quickly**, but **any increase in cost (bad news) will propagate slowly**. For a routing protocol to work properly, if **a link is broken (cost becomes infinity)**, every other router should be aware of it immediately, but in distance-vector routing, this takes some time. The problem is referred to **as count to infinity**

Two-Node Loop One example of count to infinity is the two-node loop problem

Figure 20.7 *Two-node instability*

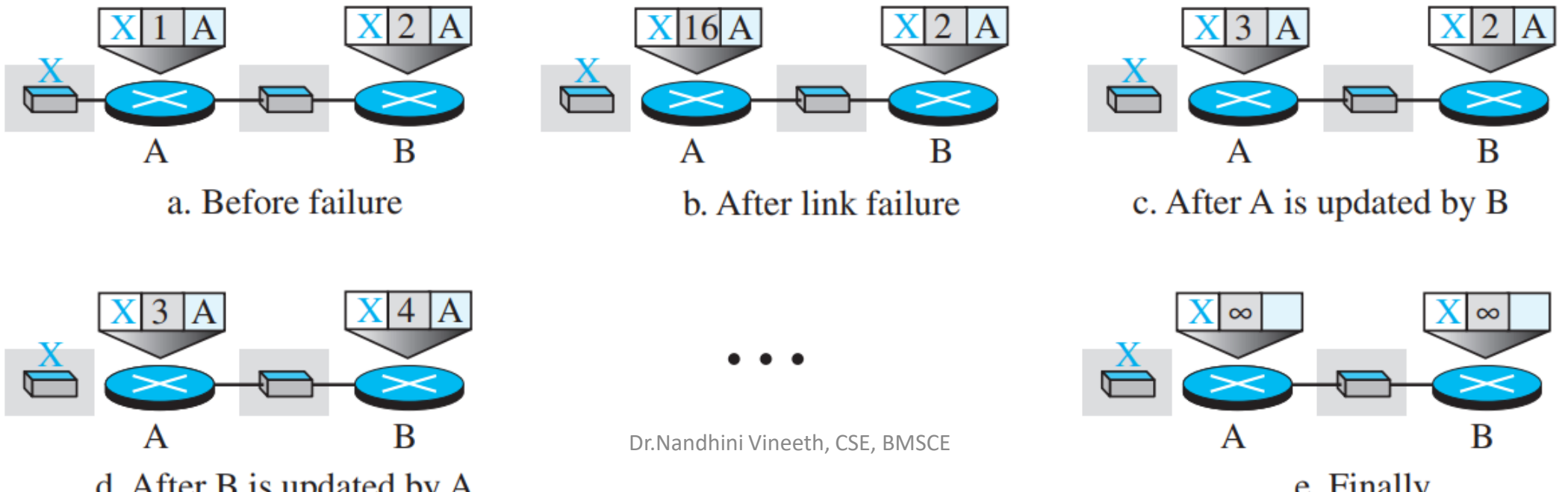
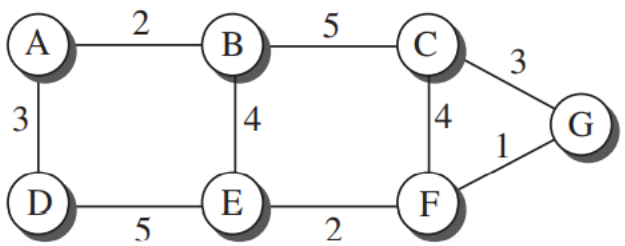


Figure 20.8 *Example of a link-state database*



a. The weighted graph

	A	B	C	D	E	F	G
A	0	2	∞	3	∞	∞	∞
B	2	0	5	∞	4	∞	∞
C	∞	5	0	∞	∞	4	3
D	3	∞	∞	0	5	∞	∞
E	∞	4	∞	5	0	2	∞
F	∞	∞	4	∞	2	0	1
G	∞	∞	3	∞	∞	1	0

b. Link state database

Split Horizon

instead of flooding the table through each interface, each node sends only part of its table through each interface. If, according to its table, node B thinks **that the optimum route to reach X is via A, it does not need to advertise this piece of information to A**

Taking information from node A, modifying it, and sending it back to node A is what creates the confusion

Poison Reverse Using the split-horizon strategy has **one drawback.**

node A cannot guess whether this is due to the split-horizon strategy (the source of information was A) or because B has not received any news about X recently

In the poison reverse strategy B can replace the distance with infinity as a warning: "Do not use this value; what I know about this route comes from you.

Three-Node Instability**:**

if the instability is between three nodes, stability cannot be guaranteed.

Link-State Routing

This method uses the term link-state to **define the characteristic of a link (an edge) that represents a network** in the internet.

In this algorithm the **cost associated with an edge defines the state of the link**

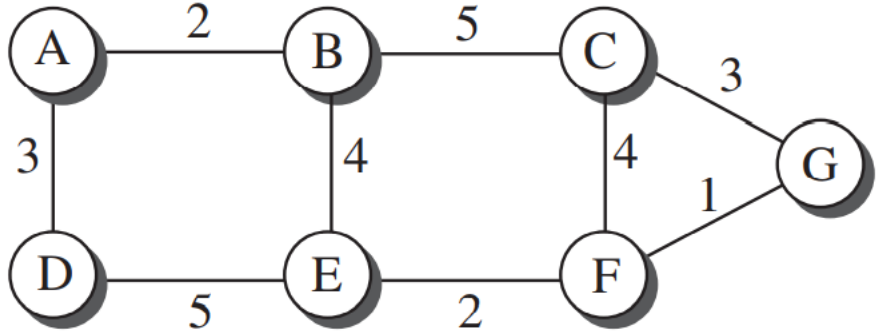
The collection of states for all links is called the **link-state database (LSDB)**

each node needs to have a duplicate of it to be able to create the least-cost tree.

The LSDB can be represented as a two-dimensional array(matrix) in which the value of each cell defines the cost of the corresponding link.

Link-State Routing

Figure 20.8 *Example of a link-state database*



a. The weighted graph

	A	B	C	D	E	F	G
A	0	2	∞	3	∞	∞	∞
B	2	0	5	∞	4	∞	∞
C	∞	5	0	∞	∞	4	3
D	3	∞	∞	0	5	∞	∞
E	∞	4	∞	5	0	2	∞
F	∞	∞	4	∞	2	0	1
G	∞	∞	3	∞	∞	1	0

b. Link state database

Table 2

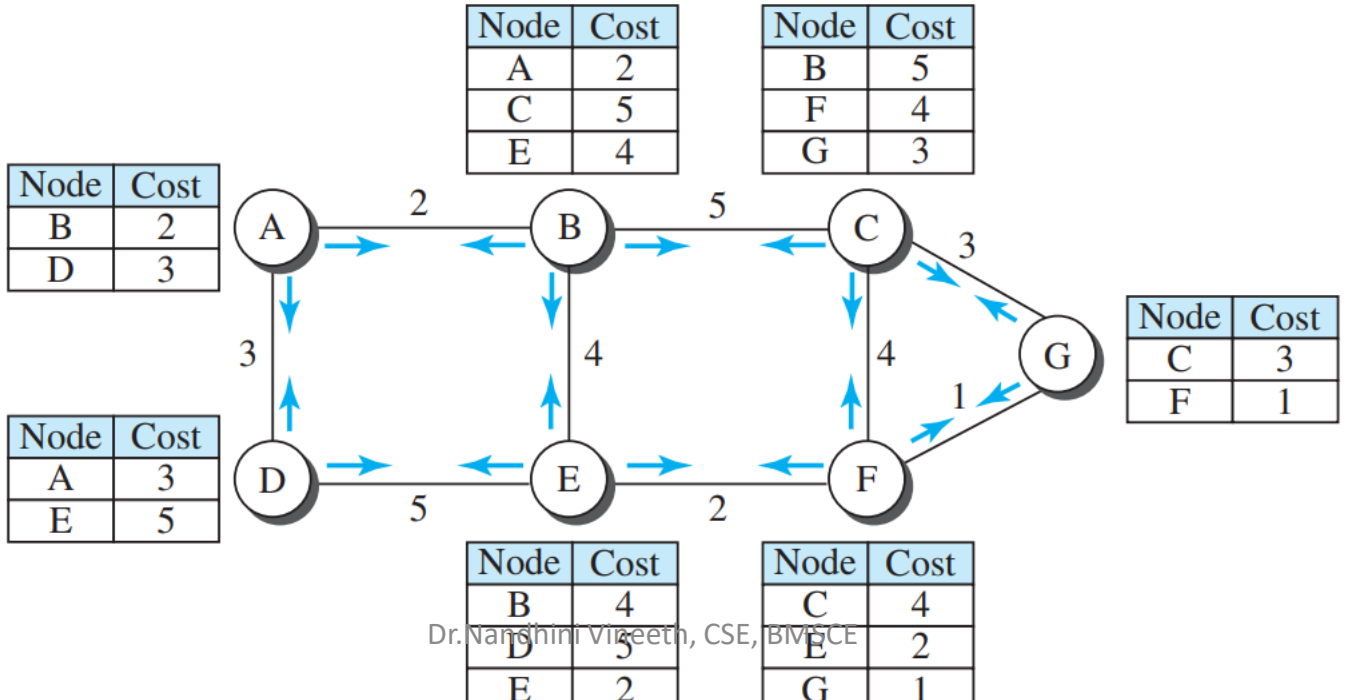
```
1   Dijkstra's Algorithm ( )
2   {
3       // Initialization
4       Tree = {root}          // Tree is made only of the root
```

Link-State Routing

Each node can create this LSDB, by a process called flooding.
The combination of these two pieces of information is called the **LS packet (LSP)- the identity of the node and the cost of the link.**

sends a copy of it **out of each interface except the one from which the packet arrived**. This guarantees that flooding stops somewhere in the network

Figure 20.9 *LSPs created and sent out by each node to build LSDB*



Link-State Routing

Formation of Least-Cost Trees

To create a least-cost tree for itself, using the shared LSDB, each node needs to run the famous Dijkstra Algorithm. This iterative algorithm uses the following steps:

1. The node chooses itself as the root of the tree, creating a tree with a single node, and sets the total cost of each node based on the information in the LSDB.

2. The node selects one node, among all nodes not in the tree, which is closest to the root, and adds this to the tree. After this node is added to the tree, the cost of all other nodes not in the tree needs to be updated because the paths may have been changed.

3. The node repeats step 2 until all nodes are added to the tree.

Dijkstra's Algorithm

Table 20.2 *Dijkstra's Algorithm*

```
1   Dijkstra's Algorithm ( )
2   {
3       // Initialization
4       Tree = {root}                    // Tree is made only of the root
```

Table 20.2 *Dijkstra's Algorithm (continued)*

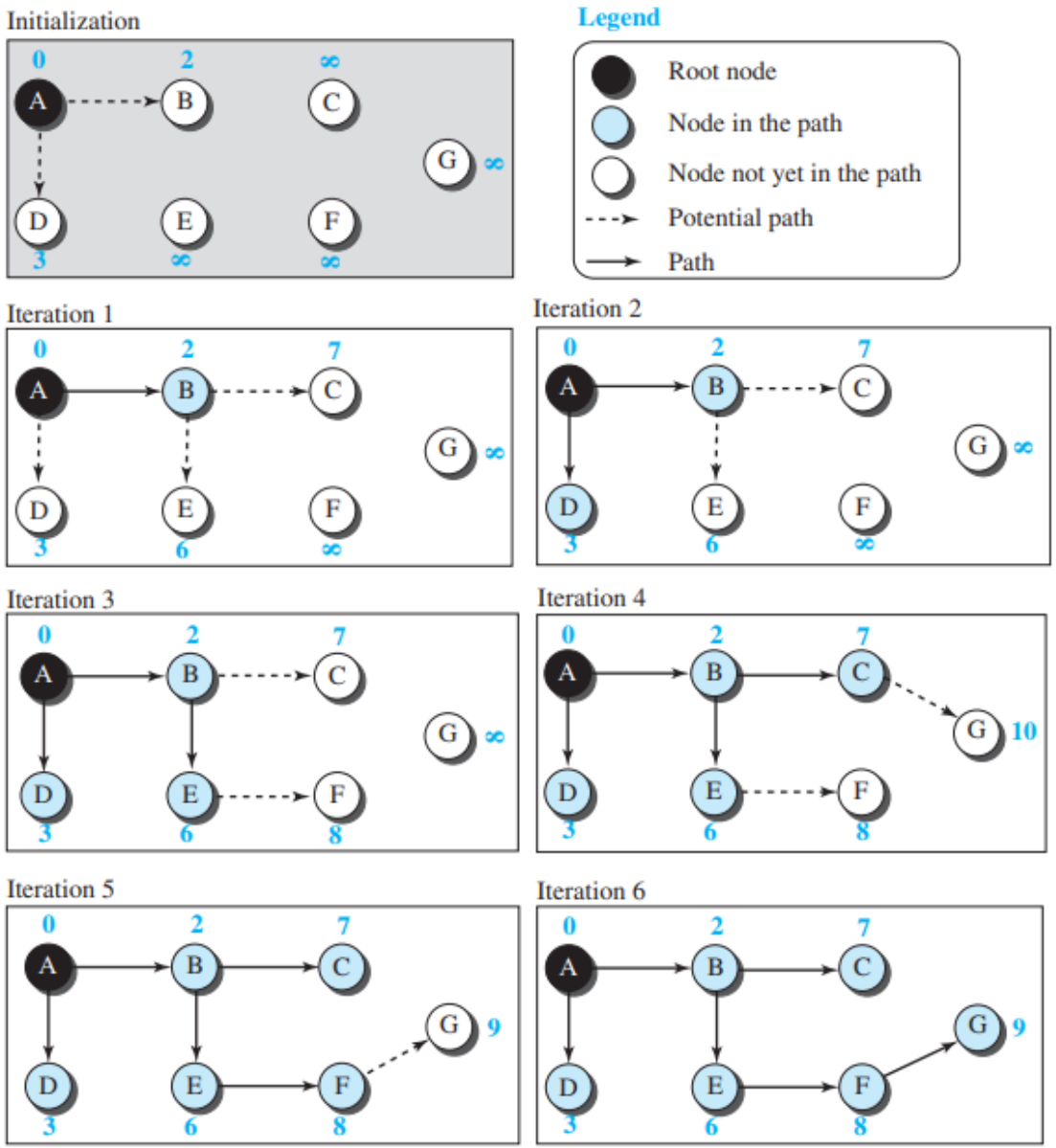
```
5       for (y = 1 to N)                 // N is the number of nodes
6       {
            y is the root)
            D[y] = 0                     // D[y] is shortest distance from root to node y
        e if (y is a neighbor)
            D[y] = c[root][y]            // c[x][y] is cost between nodes x and y in LSDB
        e
            D[y] = ∞
        ılation
16      {
17          find a node w, with D[w] minimum among all nodes not in the Tree
18          Tree = Tree ∪ {w}            // Add w to tree
19          // Update distances for all neighbors of w
20          for (every node x, which is a neighbor of w and not in the Tree)
21          {
22              D[x] = min{D[x], (D[w] + c[w][x])}
23          }
24      } until (all nodes included in the Tree)
25      // End of Dijkstra
```

Path-Vector Routing

Primary Goal is not least-cost.

assume that there are some routers in the internet that a sender wants to prevent its packets from going through.

Figure 20.10 *Least-cost tree*



Path-vector (PV) routing

does not have the drawbacks of LS or DV routing as it is not based on least-cost routing.

The best route is determined by the source using the policy it imposes on the route. (the source can control the path)
Is not actually used in an internet, and is mostly designed to route a packet between ISPs

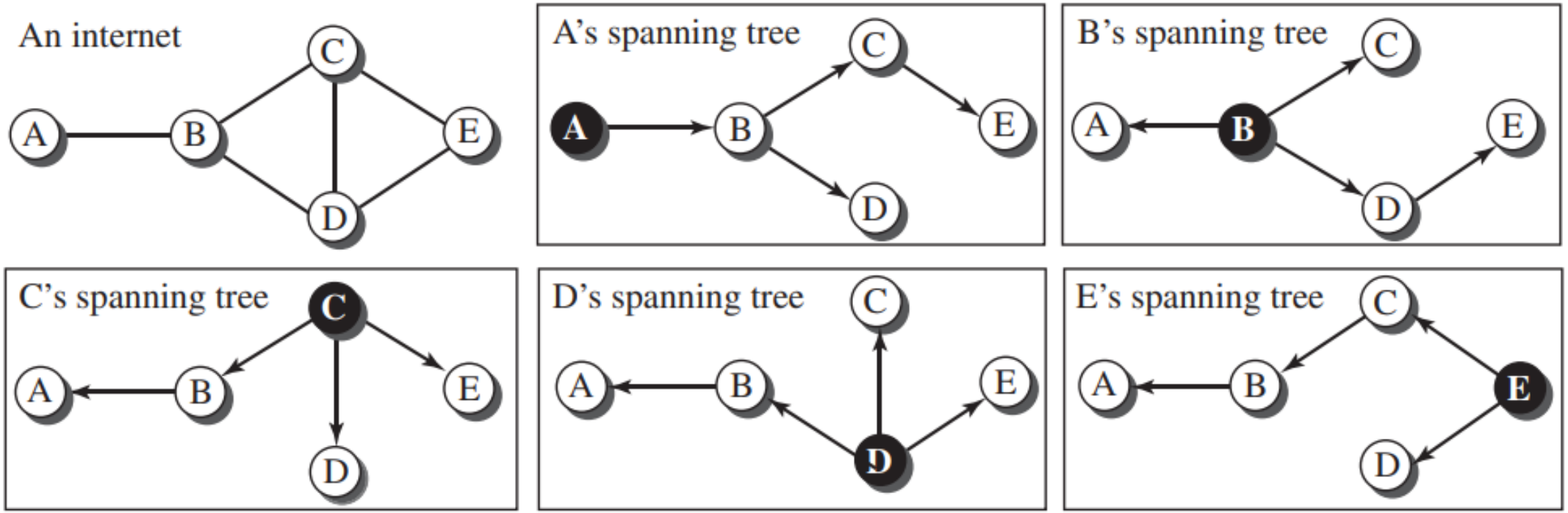
In path-vector routing, the path from a source to all destinations is also determined by the **best spanning tree.**

If there is more than one route to a destination, the source can choose the route that meets its policy best.

A source **may apply several policies at the same time**.
One of the common policies uses the minimum number of nodes to be visited (something similar to least-cost).
Another common policy is to avoid some nodes as the middle node in a route.

Path-vector (PV) routing

Figure 20.11 *Spanning trees in path-vector routing*



Path-vector (PV) routing

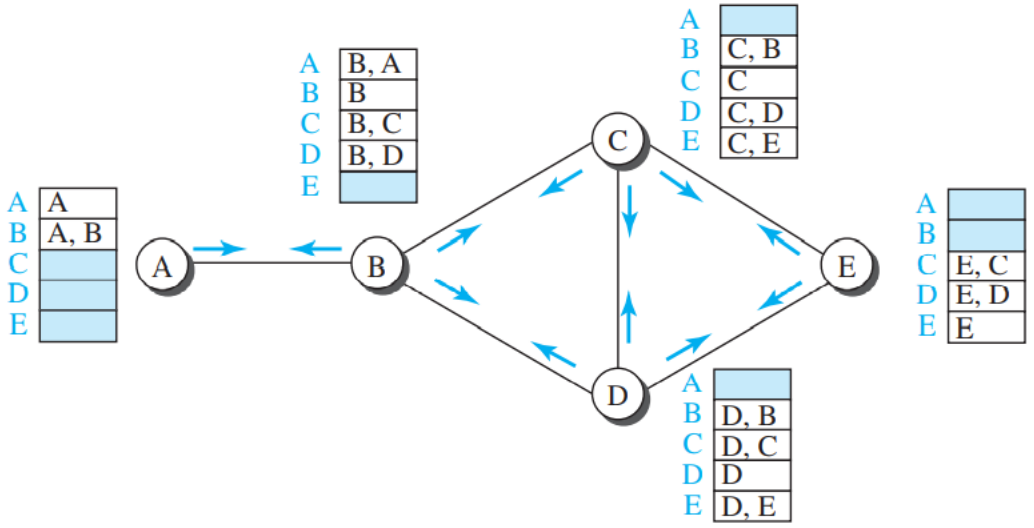
Creation of Spanning Trees

is an asynchronous and distributed routing algorithm

$$\text{Path}(x, y) = \text{best}\ \{\text{Path}(x, y), [(x + \text{Path}(v, y)]\} \quad \text{for all } v\text{'s in the internet.}$$

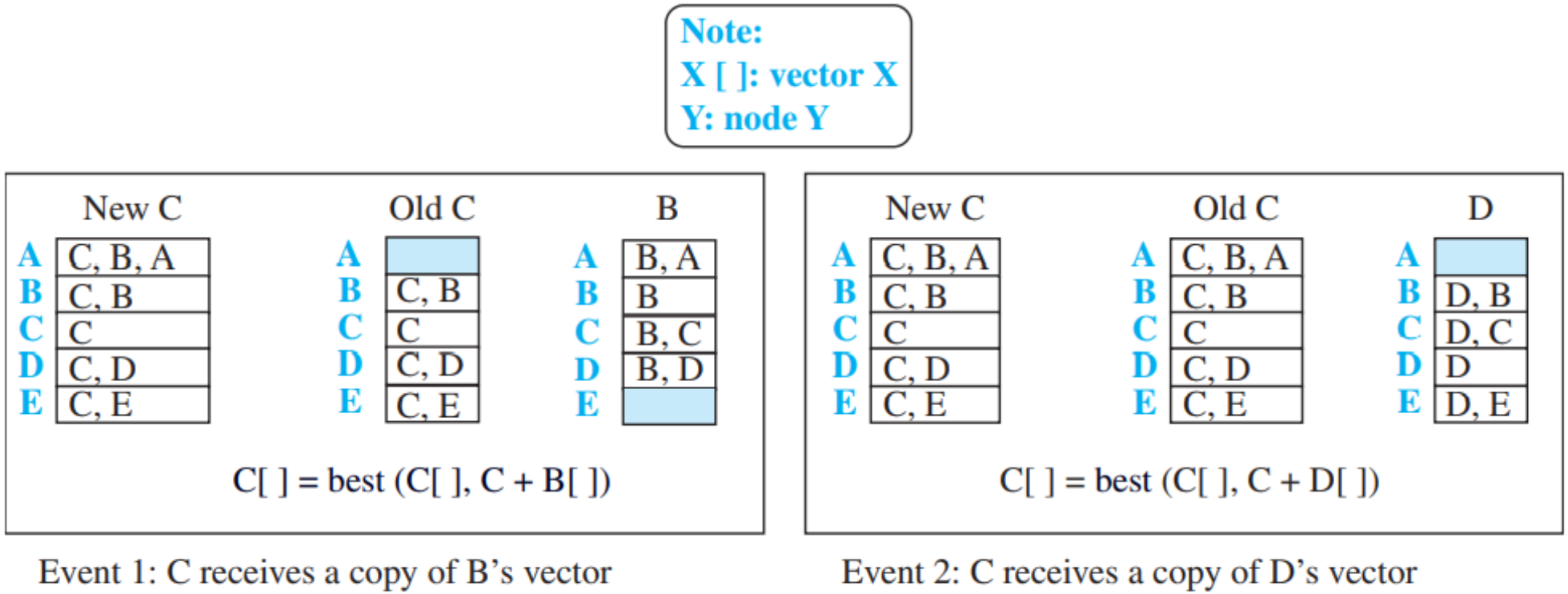
Path-vector (PV) routing

Figure 20.12 *Path vectors made at booting time*



Path-vector (PV) routing

Figure 20.13 *Updating path vectors*



Event 1: C receives a copy of B's vector

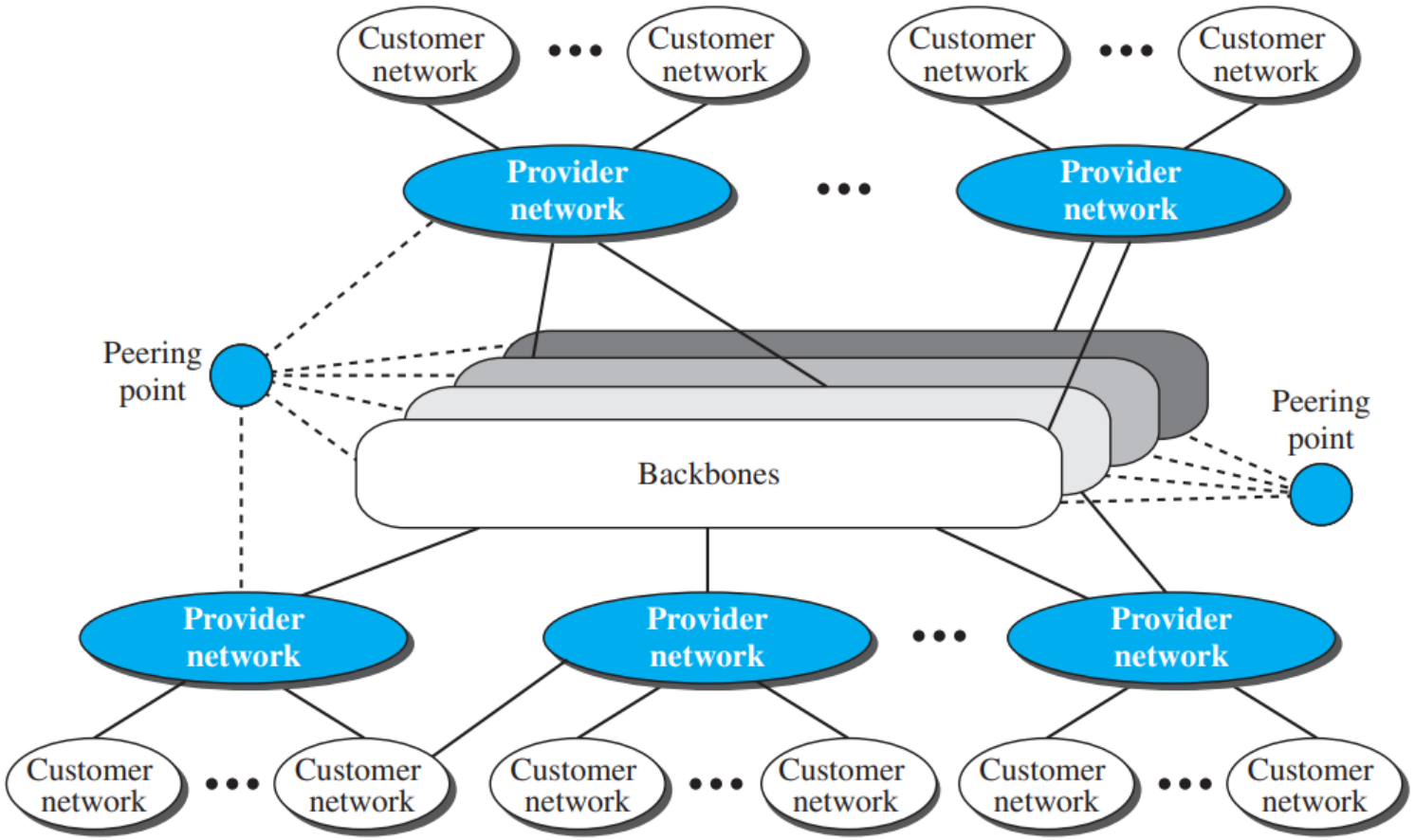
Event 2: C receives a copy of D's vector

Path-vector (PV) routi

```
1  Path_Vector_Routing ( )
2  {
3     // Initialization
4     for (y = 1 to N)
5     {
6        if (y is myself)
7           Path[y] = myself
8        else if (y is a neighbor)
9           Path[y] = myself + neighbor node
10       else
11          Path[y] = empty
12    }
13    Send vector {Path[1], Path[2], …, Path[y]} to all neighbors
14    // Update
15    repeat (forever)
16    {
17       wait (for a vector Path_w from a neighbor w)
18       for (y = 1 to N)
19       {
20          if (Path_w includes myself)
21             discard the path                    // Avoid any loop
22          else
23             Path[y] = best {Path[y], (myself + Path_w[y])}
24       }
25       If (there is a change in the vector)
26          Send vector {Path[1], Path[2], …, Path[y]} to all neighbors
27    }
28 } // End of Path Vector
```

UNICAST ROUTING PROTOCOLS

Figure 20.14 *Internet structure*



Internet Structure:

we discussed unicast routing algorithms

Protocols here are based on algo discussed

A protocol needs to define its domain of operation, the messages exchanged, communication between routers, and interaction with protocols in other domains.

UNICAST ROUTING PROTOCOLS

Hierarchical Routing

routing in the Internet cannot be done using a single protocol for two reasons:

a scalability problem and an administrative issue.

Scalability problem means that the size of the **forwarding tables becomes huge**, searching for a destination in a forwarding table becomes time-consuming, and updating creates a huge amount of traffic.

The **administrative issue** is related to the Internet structure

each ISP is run by an administrative authority.

The administrator needs to have control in its system –

reg vendors of routers, routing algo, traffic passed etc.

Hierarchical routing means considering each ISP as an **autonomous system (AS).**

Each **AS can run a routing protocol** that meets its needs, but the global Internet runs a global protocol to glue all ASs together.

The routing protocol run in each AS is referred to as intra-AS routing protocol, intradomain routing protocol, or interior gateway protocol (IGP); Ex. RIP , OSPF

the global routing protocol is referred to as inter-AS routing protocol, **interdomain routing** protocol, or exterior gateway protocol (EGP). Ex. BGP

UNICAST ROUTING PROTOCOLS

Autonomous Systems

each AS is given an 16-bit unsigned unique integer **autonomous number (ASN**) by the ICANN

categorized according to the way they are connected to other ASs.

Three types - stub ASs, multihomed ASs, and transient ASs.
Stub AS. A stub AS has **only one connection to another AS**. The data traffic can be either initiated or terminated in a stub AS; the data cannot pass through it. A good example of a stub AS is the customer network, which is either the source or the sink of data.

Multihomed AS. A multihomed AS can have **more than one connection to other ASs**, but it does not allow data traffic to pass through it. A good example of such an AS is some of the customer ASs that may use the services of more than one provider network, but their policy does not allow data to be passed through them

Transient AS. A transient AS is connected to **more than one other AS** and also allows the traffic to pass through. The **provider networks and the backbon**e are good examples of transient ASs.

Routing Information Protocol (RIP)

RIP was started as part of the **Xerox Network System (XNS),**

Hop Count
this protocol basically implements the distance-vector routing algorithm

the cost is defined as the number of hops, which means the number of networks (subnets) a packet needs to travel through from the source router to the final destination

In RIP, the maximum cost of a **path can be 15**, which means 16 is considered as infinity (no connection). For this reason, RIP can be used only in autonomous systems in which the diameter of the AS is not more than 15 hops.

Forwarding Tables - RIP

Figure 20.15 *Hop counts in RIP*

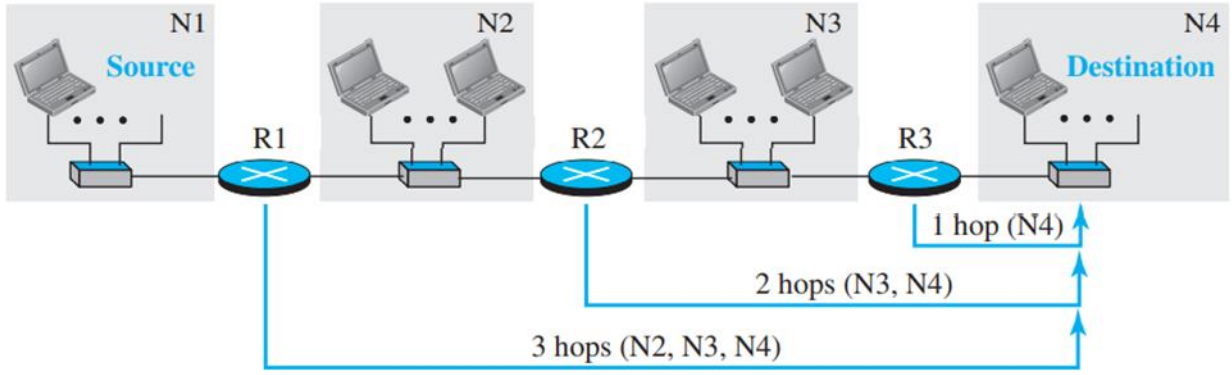


Figure 20.16 *Forwarding tables*

Forwarding table for R1

Destination network	Next router	Cost in hops
N1	——	1
N2	——	1
N3	R2	2
N4	R2	3

Forwarding table for R2

Destination network	Next router	Cost in hops
N1	R1	2
N2	——	1
N3	——	1
N4	R3	2

Forwarding table for R3

Destination network	Next router	Cost in hops
N1	R2	3
N2	R2	2
N3	——	1
N4	——	1

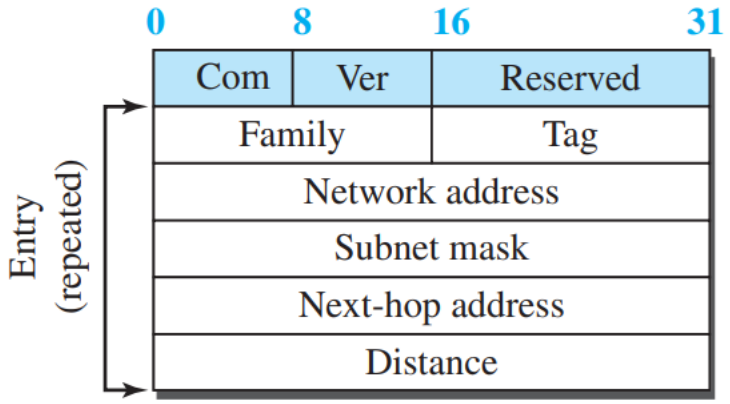
RIP Implementation

RIP is a routing protocol to help IP route its datagrams through the AS

the RIP messages are encapsulated inside UDP user datagrams, which in turn are encapsulated inside IP datagrams.

RIP runs at the **application layer**, but creates forwarding tables for IP at the network layer.
RIP has gone through two versions: **RIP-1 and RIP-2.**
RIP-2 defines the format of the message

Figure 20.17 *RIP message format*



Fields

Com: Command, request (1), response (2)
Ver: Version, current version is 2
Family: Family of protocol, for TCP/IP value is 2
Tag: Information about autonomous system
Network address: Destination address
Subnet mask: Prefix length
Next-hop address: Address length
Distance: Number of hops to the destination

RIP

RIP has two types of messages: request and response.

A request message is sent by a router that has just come up or by a router that has some time-out entries.

A request message **can ask about specific entries or all entries.**

A response (or update) message can be either solicited or unsolicited. A solicited response message is sent only in answer to a request message. It contains information about the destination specified in the corresponding request message.
An unsolicited response message, on the other hand, is sent periodically, **every 30 seconds** or when there is a change in the forwarding table.

RIP Algorithm

RIP implements the same algorithm as the distance-vector routing algorithm
some changes need to be made to the algorithm to enable a router to update its forwarding table:

RIP

Instead of sending only distance vectors, a router needs to send the **whole contents of its forwarding table** in a response message.

The receiver **adds one hop to each cost and changes the next router field** to the address of the sending router.

We call each route in the modified forwarding table the received route and each route in the old forwarding table the old route. The received router selects the old routes as the new ones except in the following three cases:

1. If the received route **does not exist in the old forwarding table**, it should be added to the route.
2. If the cost of the **received route is lower than the cost of the old one**, the received route should be selected as the new one.
3. If the cost of the **received route is higher than the cost of the old one, but the value of the next router is the same in both routes**, the received route should be selected as the new one. This is the case where the route was **actually advertised by the same router in the past, but now the situation has been changed**. For example, suppose a neighbor has previously advertised a route to a destination with cost 3, but now there is no path between this neighbor and that destination. The neighbor advertises this destination with cost value infinity (16 in RIP). The receiving router must not ignore this value even though its old route has a lower cost to the same destination.
4. The new forwarding table needs to be sorted according to the destination route (mostly using the longest prefix first).

RIP TIMERS

RIP **uses three timers** to support its operation.

The **periodic timer** controls the advertising of **regular update messages**. Each router has one periodic timer that is randomly set to a number between 25 and 35 seconds (to prevent all routers sending their messages at the same time and creating excess traffic).

The **expiration timer** governs the **validity of a route**. When a router receives update information for a route, the expiration timer is **set to 180 seconds** for that particular route. Every time a new update for the route is received, the timer is reset. If there is a problem on an internet and **no update is received within the allotted 180 seconds, the route is considered expired** and the hop count of the r**oute is set to 16**, which means the **destination is unreachable**. Every route has its own expiration timer.

The **garbage collection timer** is used to purge a route from the forwarding table. When the information about a route becomes invalid, the router does not immediately purge that route from its table. Instead, it continues to advertise the route with a metric value of 16. At the same time, a **garbage collection timer is set to 120** seconds for that route. When the count reaches zero, the route is purged from the table. This timer allows neighbors to become aware of the invalidity of a route prior to purging.

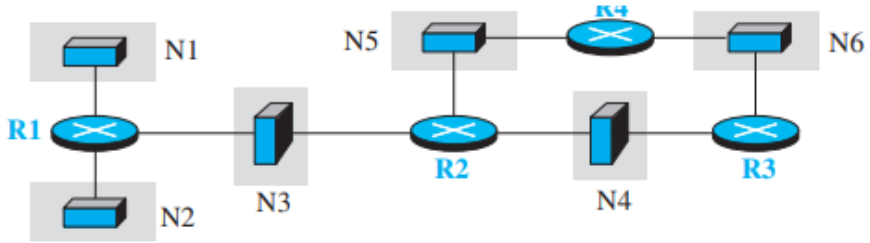
Performance

Update Messages. The **update messages in RIP have a very simple format** and are sent only to neighbors; they are **local**. They do not normally create traffic because the routers try to avoid sending them at the same time.

Convergence of Forwarding Tables. RIP uses the distance-vector algorithm, which can converge slowly if the domain is large, but, since RIP allows only 15 hops in a domain (**16 is considered as infinity**), there is **normally no problem in convergence**. The only problems that may slow down convergence are **count-to-infinity and loops created in the domain;** use of poison-reverse and split-horizon strategies added to the RIP extension may alleviate the situation

Robustness.
Distance-vector routing is based on the concept that **each router sends what it knows about the whole domain to its neighbors.** This means that the calculation of the forwarding table depends on information received from immediate neighbors, which in turn receive their information from their own neighbors. If **there is a failure or corruption in one router, the problem will be propagated to all routers and the** forwarding in each router will be affected.

Figure 20.18 *Example of an autonomous system using RIP*



Legend

Des.: Destination network
N. R.: Next router
Cost: Cost in hops
← : New route
← : Old route

R1

Des.	N. R.	Cost
N1	——	1
N2	——	1
N3	——	1

R2

Des.	N. R.	Cost
N3	——	1
N4	——	1
N5	——	1

R3

Des.	N. R.	Cost
N4	——	1
N6	——	1

R4

Des.	N. R.	Cost
N5	——	1
N6	——	1

Forwarding tables after all routers booted

New R1

Des.	N. R.	Cost
N1	——	1
N2	——	1
N3	——	1
N4	R2	2
N5	R2	2

Old R1

Des.	N. R.	Cost
N1	——	1
N2	——	1
N3	——	1

R2 Seen by R1

Des.	N. R.	Cost
N3	R2	2
N4	R2	2
N5	R2	2

New R3

Des.	N. R.	Cost
N3	R2	2
N4	——	1
N5	R2	2
N6	——	1

Old R3

Des.	N. R.	Cost
N4	——	1
N6	——	1

R2 Seen by R3

Des.	N. R.	Cost
N3	R2	2
N4	R2	2
N5	R2	2

New R4

Des.	N. R.	Cost
N3	R2	2
N4	R2	2
N5	——	1
N6	——	1

Old R4

Des.	N. R.	Cost
N5	——	1
N6	——	1

R2 Seen by R4

Des.	N. R.	Cost
N3	R2	2
N4	R2	2
N5	R2	2

Changes in the forwarding tables of R1, R3, and R4 after they receive a copy of R2's table

Final R1

Des.	N. R.	Cost
N1	——	1
N2	——	1
N3	——	1
N4	R2	2
N5	R2	2
N6	R2	3

Final R2

Des.	N. R.	Cost
N1	R1	2
N2	R1	2
N3	——	1
N4	——	1
N5	——	1
N6	R3	2

Final R3

Des.	N. R.	Cost
N1	R2	3
N2	R2	3
N3	R2	2
N4	——	1
N5	R2	2
N6	——	1

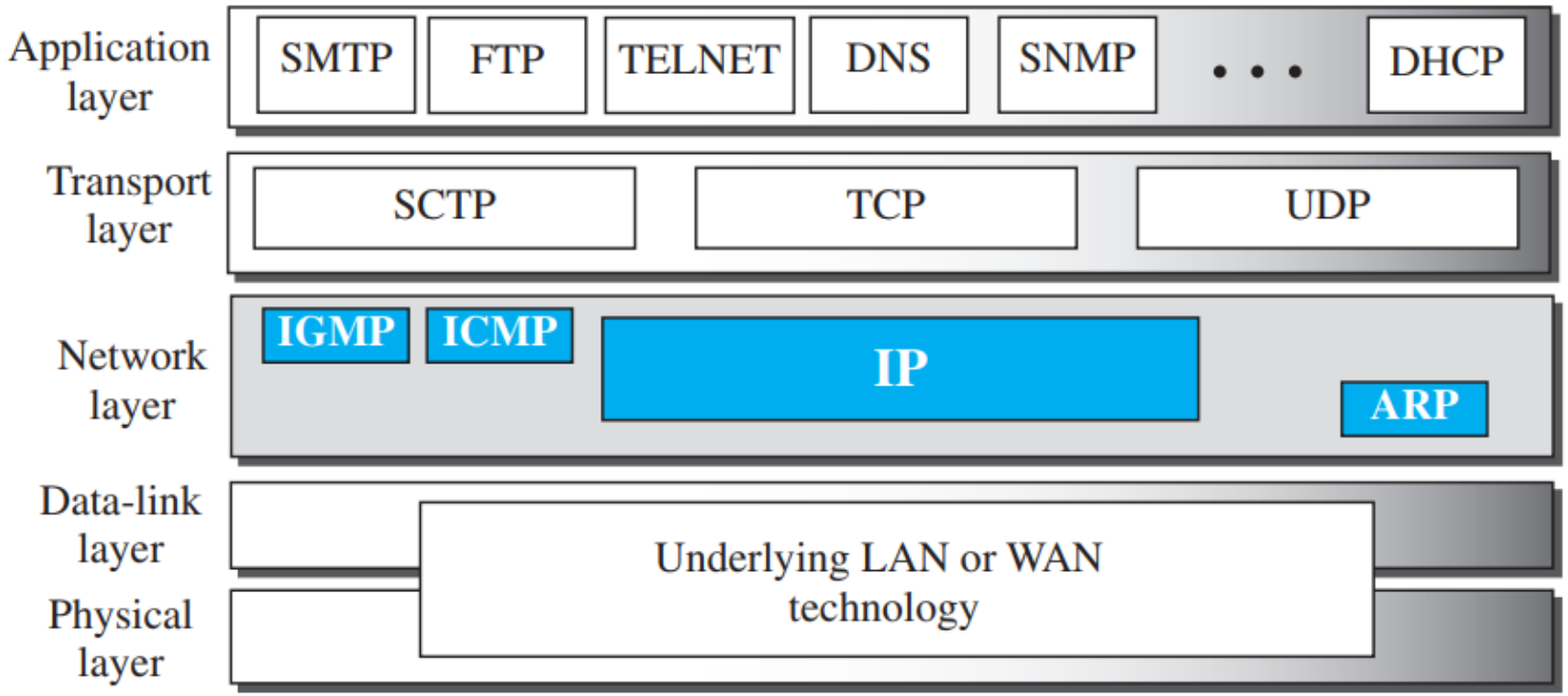
Final R4

Des.	N. R.	Cost
N1	R2	3
N2	R2	3
N3	R2	2
N4	R2	2
N5	——	1
N6	——	1

Forwarding tables for all routers after they have been stablized

INTERNET PROTOCOL (IP)

Figure 19.1 *Position of IP and other network-layer protocols in TCP/IP protocol suite*



IPv4

IPv4 is an unreliable datagram protocol—a best-effort delivery service.

IPv4 is also a connectionless protocol that uses the datagram approach.

The term best-effort means that IPv4 packets can be corrupted, be lost, arrive out of order, or be delayed, and may create congestion for the network.

If reliability is important, IPv4 must be paired with a reliable transport-layer protocol such as TCP

Figure 19.2 *IP datagram*

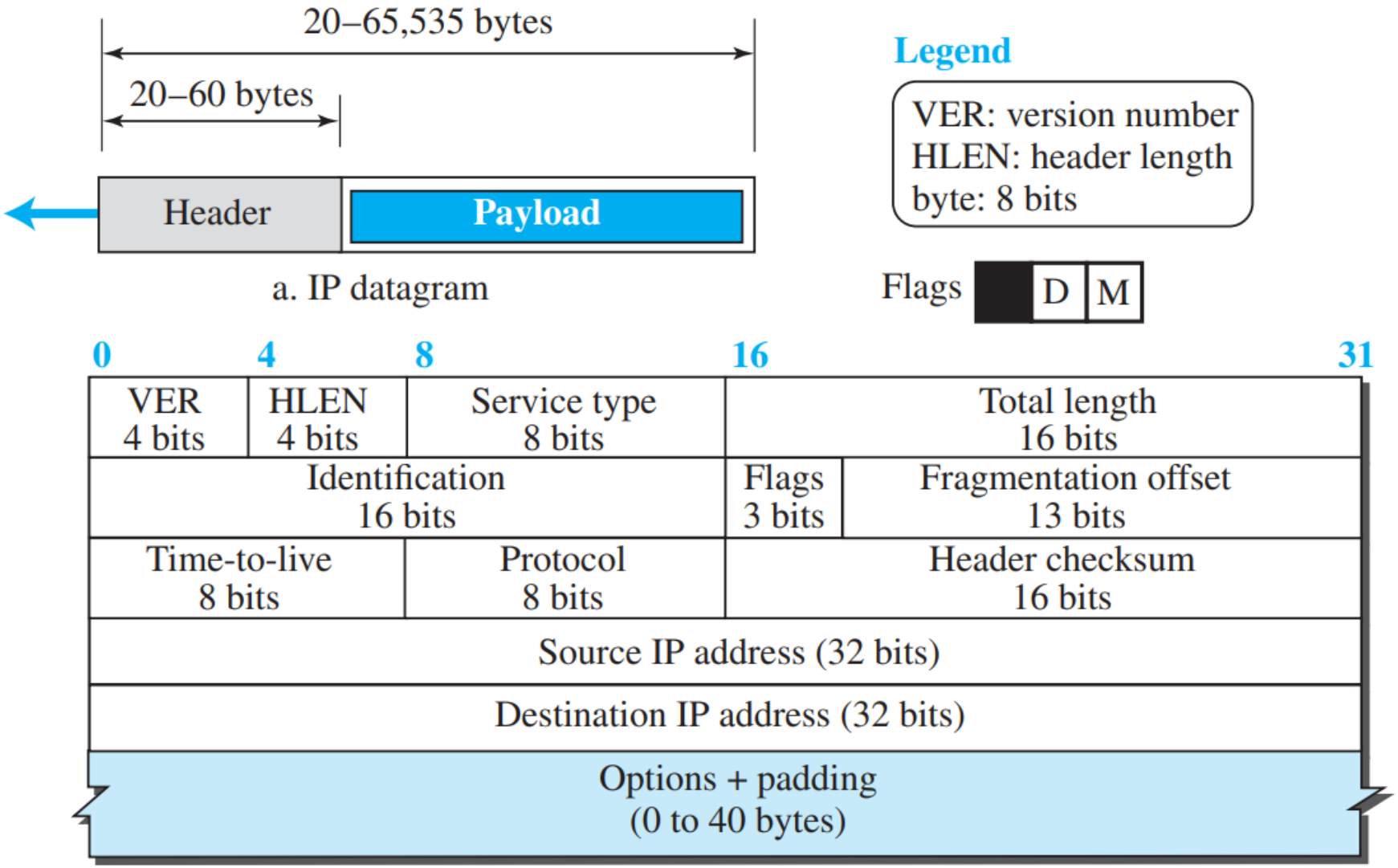
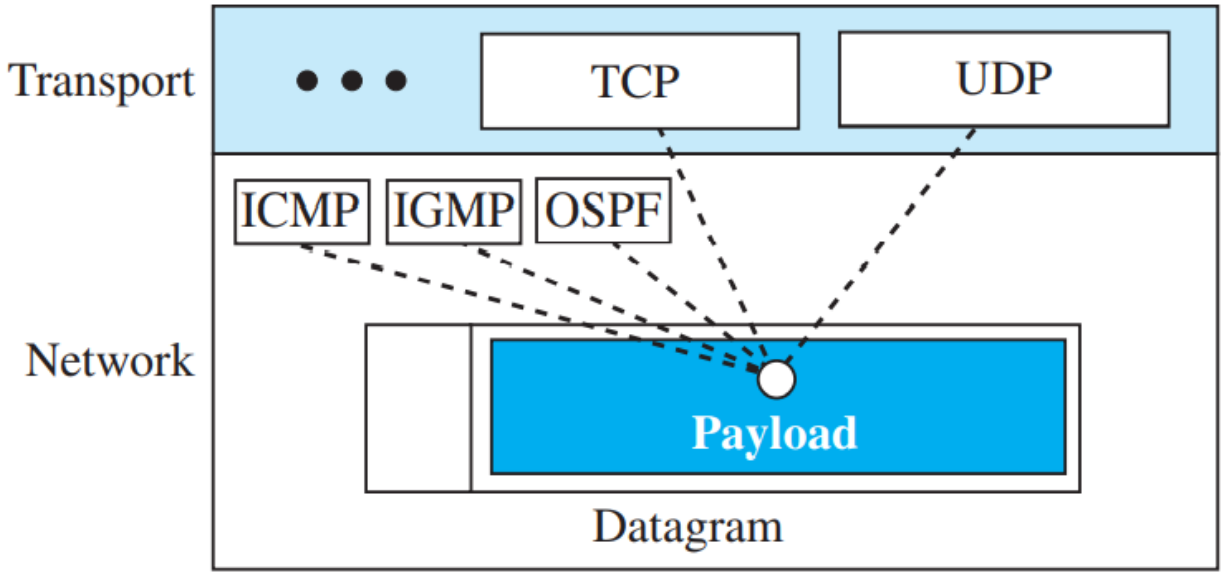


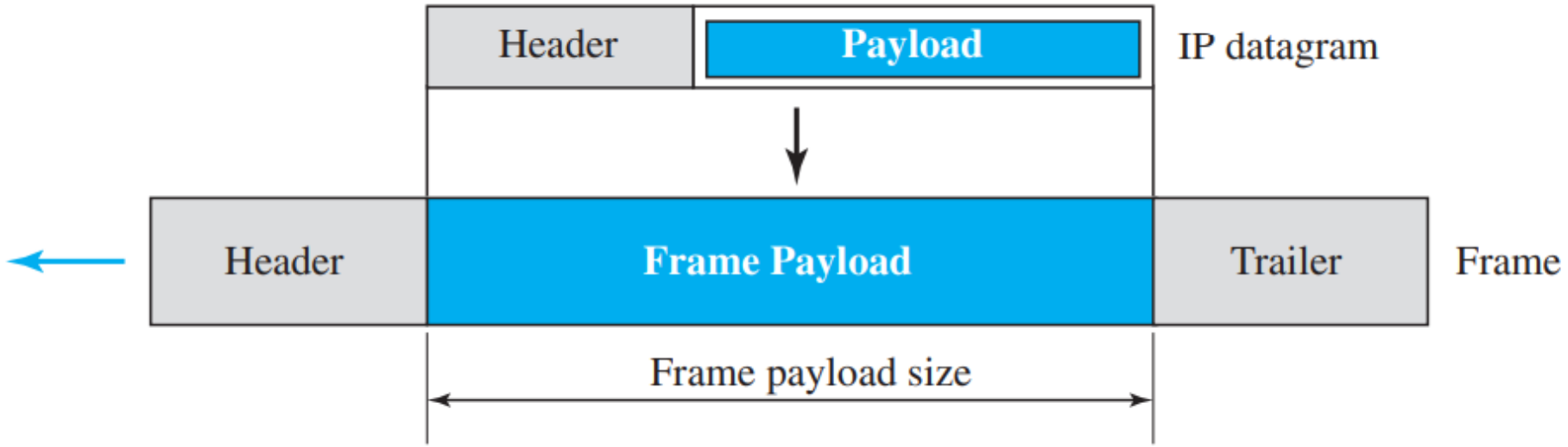
Figure 19.3 *Multiplexing and demultiplexing using the value of the protocol field*



Some protocol values

ICMP	01
IGMP	02
TCP	06
UDP	17
OSPF	89

Figure 19.5 *Maximum transfer unit (MTU)*



MTU: Maximum size of frame payload

Figure 19.6 *Fragmentation example*

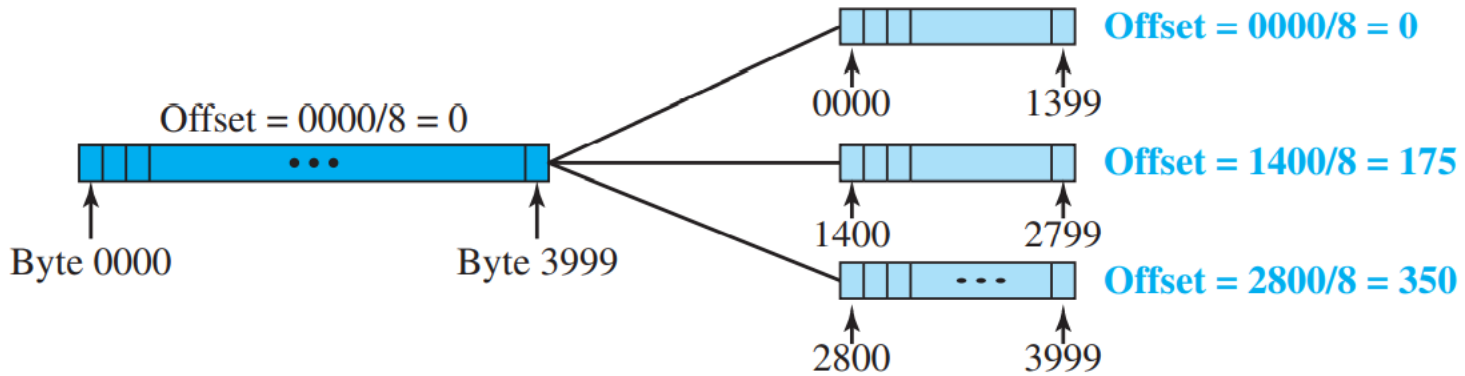
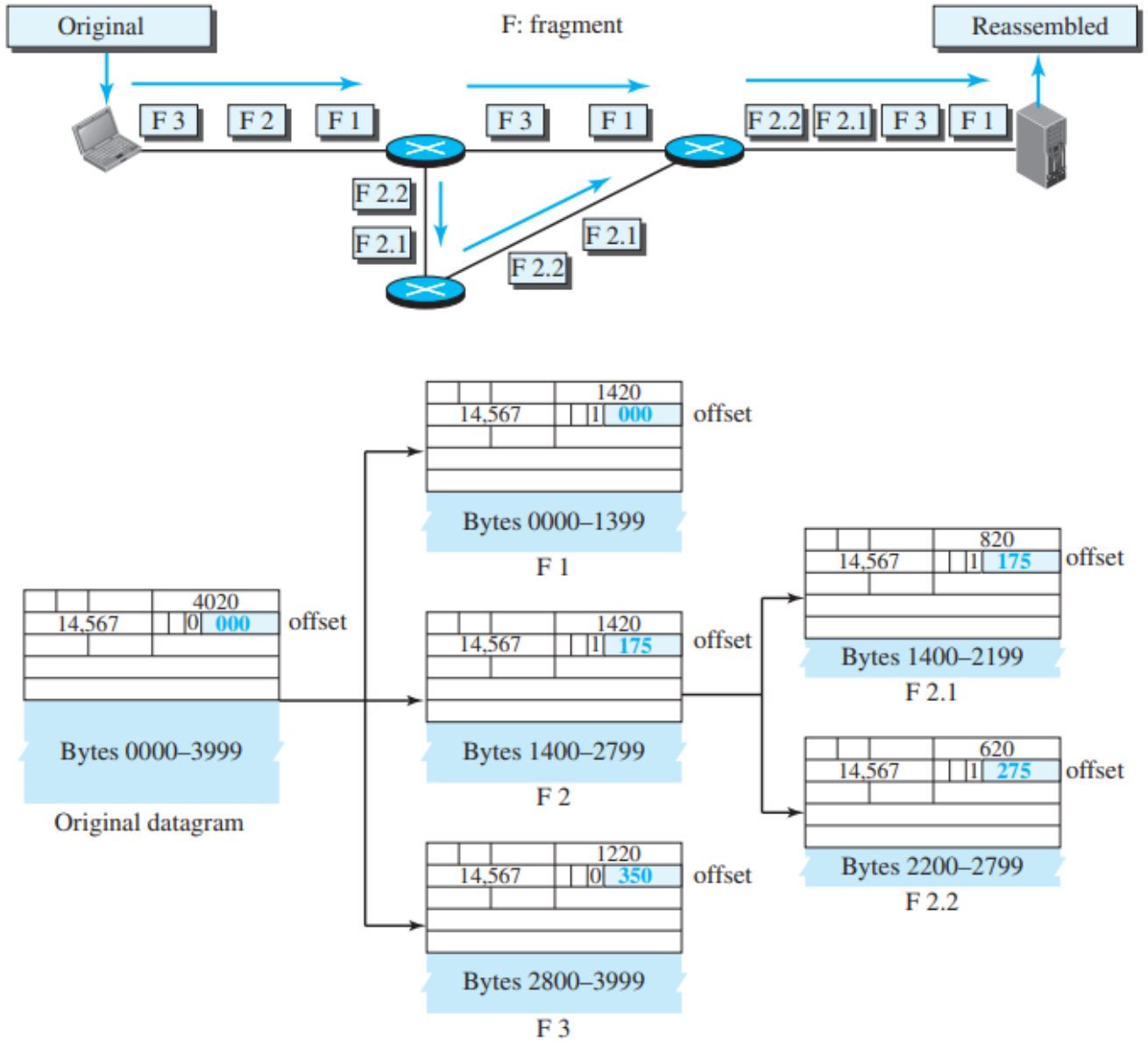


Figure 19.7 *Detailed fragmentation example*



Example 19.1

An IPv4 packet has arrived with the first 8 bits as $(01000010)_2$. The receiver discards the packet. Why?

Solution

There is an error in this packet. The 4 leftmost bits $(0100)_2$ show the version, which is correct. The next 4 bits $(0010)_2$ show an invalid header length ($2 \times 4 = 8$). The minimum number of bytes in the header must be 20. The packet has been corrupted in transmission.

Example 19.2

In an IPv4 packet, the value of HLEN is $(1000)_2$. How many bytes of options are being carried by this packet?

Solution

The HLEN value is 8, which means the total number of bytes in the header is 8×4, or 32 bytes. The first 20 bytes are the base header, the next 12 bytes are the options.

Example 19.3

In an IPv4 packet, the value of HLEN is 5, and the value of the total length field is $(0028)_{16}$. How many bytes of data are being carried by this packet?

Solution

The HLEN value is 5, which means the total number of bytes in the header is 5×4, or 20 bytes (no options). The total length is $(0028)_{16}$ or 40 bytes, which means the packet is carrying 20 bytes of data (40 – 20).

Example 19.4

An IPv4 packet has arrived with the first few hexadecimal digits as shown.

$$(45000028000100000102 \ldots)_{16}$$

Example 19.6

A packet has arrived with an M bit value of 0. Is this the first fragment, the last fragment, or a middle fragment? Do we know if the packet was fragmented?

Solution

If the M bit is 0, it means that there are no more fragments; the fragment is the last one. However, we cannot say if the original packet was fragmented or not. A nonfragmented packet is considered the last fragment.

Example 19.7

A packet has arrived with an M bit value of 1. Is this the first fragment, the last fragment, or a middle fragment? Do we know if the packet was fragmented?

Solution

If the M bit is 1, it means that there is at least one more fragment. This fragment can be the first one or a middle one, but not the last one. We don't know if it is the first one or a middle one; we need more information (the value of the fragmentation offset).

Example 19.8

A packet has arrived with an M bit value of 1 and a fragmentation offset value of 0. Is this the first fragment, the last fragment, or a middle fragment?

Solution

Because the M bit is 1, it is either the first fragment or a middle one. Because the offset value is 0, it is the first fragment.

Example 19.9

A packet has arrived in which the offset value is 100. What is the number of the first byte? Do we know the number of the last byte?

Solution

To find the number of the first byte, we multiply the offset value by 8. This means that the first byte number is 800. We cannot determine the number of the last byte unless we know the length of the data.

Example 19.10

A packet has arrived in which the offset value is 100, the value of HLEN is 5, and the value of the total length field is 100. What are the numbers of the first byte and the last byte?

Solution

The first byte number is $100 \times 8 = 800$. The total length is 100 bytes, and the header length is 20 bytes (5×4), which means that there are 80 bytes in this datagram. If the first byte number is 800, the last byte number must be 879.

OPTIONS

The header of the IPv4 datagram is made of two parts:
a fixed part and a variable part.

The fixed part is 20 bytes long and was discussed in the previous section. The variable part comprises the options that can be a maximum of 40 bytes (in multiples of 4-bytes) to preserve the boundary of the header.

Options are divided into two broad categories: **single-byte options and multiple-byte options**.

Single-Byte Options:
There are two single-byte options.
No Operation A no-operation option is a 1-byte option used as a filler between options.
End of Option An end-of-option option is a 1-byte option used for padding at the end of the option field.

Multliple-Byte Options
There are four multiple-byte options.

Record Route
A record route option is used to record the Internet routers that handle the datagram. It can list up to nine router addresses. It can be used for debugging and management purposes.

Strict Source Route A strict source route option is used by the source to predetermine a route for the datagram as it travels through the Internet. Dictation of a route by the source can be useful for several purposes. The sender can choose a route with a specific type of service, such as minimum delay or maximum throughput. Alternatively, it may choose a route that is safer or more reliable for the sender's purpose. For example, a sender can choose a route so that its datagram does not travel through a competitor's network.

Loose Source Route A loose source route option is similar to the strict source route, but it is **less rigid**. Each router in the list must be visited, but the datagram can visit other routers as well.

Timestamp A timestamp option is used to **record the time of datagram processing by a router**. The time is expressed **in milliseconds from midnight**, Universal time or Greenwich mean time. Knowing the time a datagram is processed can help users and managers track the behavior of the routers in the Internet. We can estimate the time it takes for a datagram to go from one router to another. We say estimate because, although all routers may use Universal time, their local clocks may not be synchronized.

Security of IPv4 Datagrams

Packet Sniffing

TO MAKE IT USELESS, ENCRYPTION REQUIRED

Packet Modification

a data integrity mechanism

IP Spoofing

masquerade as somebody else
origin authentication mechanism

IPSec

Defining Algorithms and Keys
Packet Encryption
Data Integrity - MD5, SHA
Origin Authentication

ICMPV4

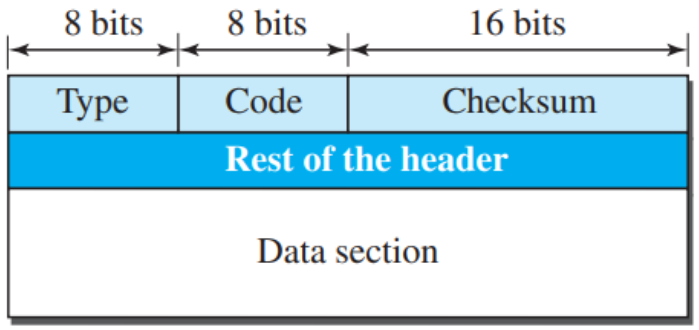
- The IPv4 has no error-reporting or error-correcting mechanism
- The IP protocol also lacks a mechanism for host and management queries
- Internet Control Message Protocol version 4 (ICMPV4), companion to the IP protocol overcomes these two shortcomings
- messages are not passed directly to the data-link layer first encapsulated inside IP datagrams before going to the lower layer. When an IP datagram encapsulates an ICMP message, the value of the protocol field in the IP datagram is set to 1 to indicate that the IP payroll is an ICMP message

ICMP

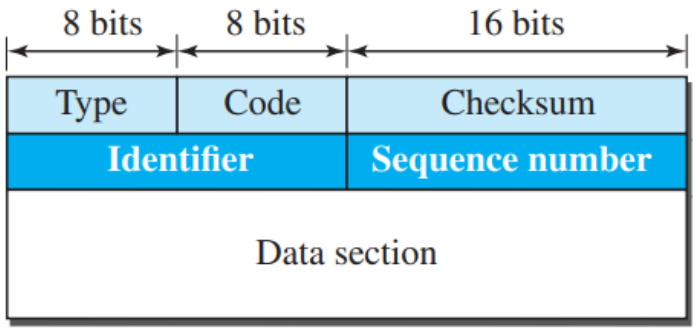
- ICMP messages are divided into two broad categories:
  - **error-reporting messages** report problems that a router or a host (destination) may encounter when it processes an IP packet.
  - **query messages-** occur in pairs help a host or a network manager get specific information from a router or another host
  - An ICMP message has an **8-byte header and a variable-size data section**.
  - Type- defines the type of the message.
  - code - specifies the reason for the particular message type.
  - The last common field is the checksum field
  - The rest of the header is specific for each message type

ICMP

Figure 19.8 *General format of ICMP messages*



Error-reporting messages



Query messages

Type and code values

Error-reporting messages	Query messages
03: Destination unreachable (codes 0 to 15)	08 and 00: Echo request and reply (only code 0)
04: Source quench (only code 0)	13 and 14: Timestamp request and reply (only code 0)
05: Redirection (codes 0 to 3)	
11: Time exceeded (codes 0 and 1)	
12: Parameter problem (codes 0 and 1)	

ICMP

- error messages -carries information for finding the original packet that had the error.
- Query messages, the data section carries extra information based on the type of query.
- Error Reporting Messages
  - main responsibilities of ICMP is to report some errors that may occur during the processing of the IP datagram.
  - ICMP does not correct errors, it simply reports them.
  - Are always sent to the original source because the only information available in the datagram about the route is the source and destination IP addresses.

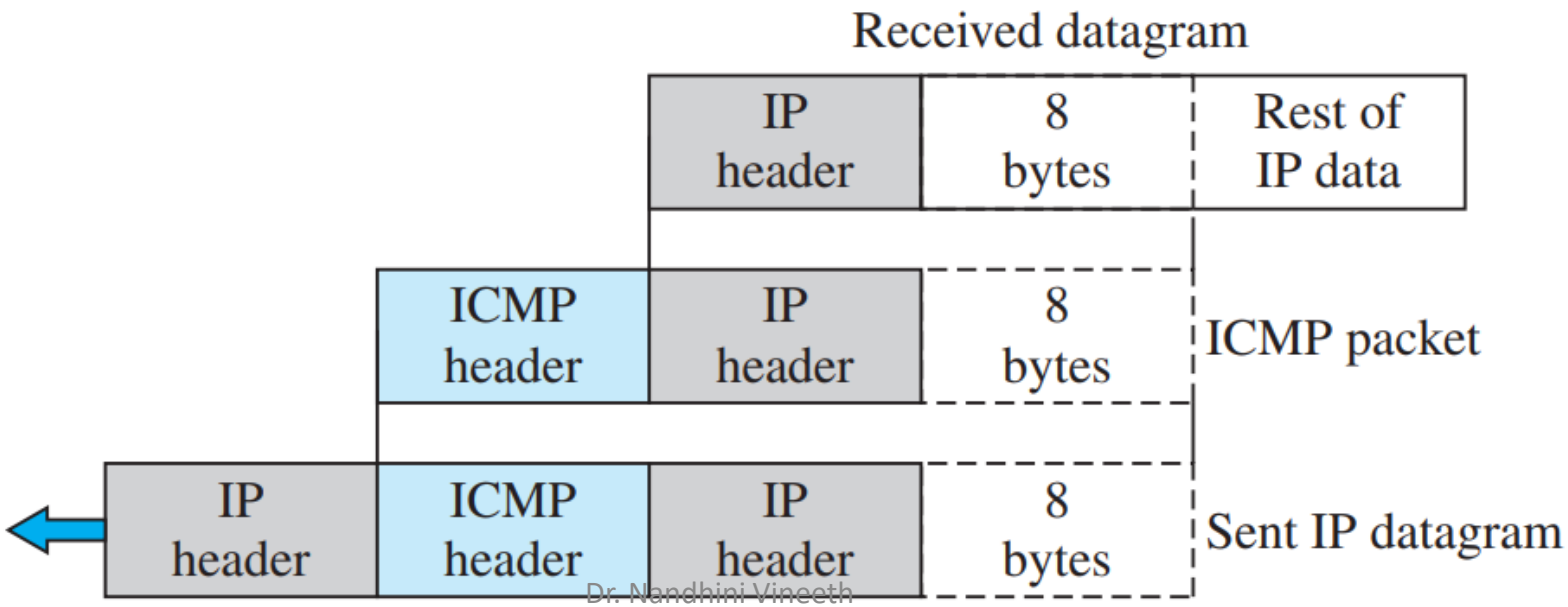
ICMP

- ICMP follows some rules in reporting messages.
- First, no error message will be generated for a datagram having a multicast address or special address (such as this host or loopback).
- Second, no ICMP error message will be generated in response to a datagram carrying an ICMP error message.
- Third, no ICMP error message will be generated for a fragmented datagram that is not the first fragment.
- Note that all error messages contain a data section that includes the IP header of the original datagram plus the first 8 bytes of data in that datagram. The original datagram header is added to give the original source, which receives the error message, information about the datagram itself. The 8 bytes of data are included because the first 8 bytes provide information about the port numbers (UDP and TCP) and sequence number (TCP). This information is needed so the source can inform the protocols (TCP or UDP) about the error.

ICMP

ICMP forms an error packet, which is then encapsulated in an IP datagram

Figure 19.9 *Contents of data field for the error messages*



ICMP

- Destination Unreachable (type 3).
  - This message uses different codes (0 to 15) to define the type of error message and the reason why a datagram has not reached its final destination. For example, code 0 -when we use the HTTP protocol to access a web page, but the server is down. The message “destination host is not reachable” is created and sent back to the source.
- Source Quench (type 4)
  - informs the sender that the network has encountered congestion and the datagram has been dropped; the source needs to slow down sending more datagrams. In other words, ICMP adds a kind of congestion control mechanism to the IP protocol by using this type of message.
- Redirection Message (type 5)
  - used when the source uses a wrong router to send out its message. The router redirects the message to the appropriate router, but informs the source that it needs to change its default router in the future. The IP address of the default router is sent in the message.
  - We discussed the purpose of the time-to-live (TTL) field in the IP datagram and explained that it prevents a datagram from being aimlessly circulated in the Internet. When the TTL value becomes 0, the datagram is dropped by the visiting router and a time exceeded message (type 11) with code 0 is sent to the source to inform it about the situation. The time-exceeded message (with code 1) can also be sent when not all fragments of a datagram arrive within a predefined period of time.
- Parameter Problem (type 12)
  - when either there is a problem in the header of a datagram (code 0) or some options are missing or cannot be interpreted (code 1).

ICMP

- Query Messages
  - can be used independently without relation to an IP datagram.
  - a query message needs to be encapsulated in a datagram, as a carrier.
  - Query messages are used to probe or test the liveliness of hosts or routers in the Internet, find the one-way or the round-trip time for an IP datagram between two devices, or even find out whether the clocks in two devices are synchronized.
  - query messages come in pairs: request and reply. The echo request (type 8) and the echo reply (type 0) pair of messages are used by a host or a router to test the liveliness of another host or router. A host or router sends

ICMP

- an echo request message to another host or router; if the latter is alive, it responds with an echo reply message.
- applications of this pair in two debugging tools: ping and traceroute.
- The timestamp request (type 13) and the timestamp reply (type 14) pair of messages are used to find the round-trip time between two devices or to check whether the clocks in two devices are synchronized.
  - The timestamp request message sends a 32-bit number, which defines the time the message is sent.
  - The timestamp reply resends that number, but also includes two new 32-bit numbers representing the time the request was received and the time the response was sent. If all timestamps represent Universal time, the sender can calculate the one-way and round-trip time.

Deprecated Messages

- Three pairs of messages are declared obsolete by IETF:
- 1. Information request and replay messages –done by Address Resolution Protocol (ARP)
- 2. Address mask request and reply messages -done by the Dynamic Host Configuration Protocol (DHCP)
- 3. Router solicitation and advertisement messages are not used today- done by the Dynamic Host Configuration Protocol (DHCP)

Debugging Tools

- There are several tools that can be used in the Internet for debugging.
  - To determine the viability of a host or router.
  - We can trace the route of a packet.
  - We introduce two tools that use ICMP for debugging: ping and traceroute.
    - Ping-
      - Program is used to find if a host is alive and responding.
      - to see how it uses ICMP packets.
      - The source host sends ICMP echo-request messages; the destination, if alive, responds with ICMP echo-reply messages.
      - The ping program sets the identifier field in the echo-request and echo-reply message and starts the sequence number from 0; this number is incremented by 1 each time a new message is sent.
      - that ping can calculate the round-trip time. It inserts the sending time in the data section of the message. When the packet arrives, it subtracts the arrival time from the departure time to get the round-trip time (RTT).

Example

- The following shows how we send a ping message to the auniversity.edu site.
- We set the identifier field in the echo request and reply message and start the sequence number from 0; this number is incremente by one each time a new message sent.
- Note that ping can calculate the round-trip time. It inserts the sending time in the data section of the message.
- When the packet arrives, it subtracts the arrival time from the departure time to get the round-trip time (rtt).

```
$ ping auniversity.edu
PING auniversity.edu (152.181.8.3)   56 (84)  bytes of data.
64 bytes from auniversity.edu (152.181.8.3): icmp_seq=0    ttl=62    time=1.91 ms

64 bytes from auniversity.edu (152.181.8.3): icmp_seq=1    ttl=62    time=2.04 ms
64 bytes from auniversity.edu (152.181.8.3): icmp_seq=2    ttl=62    time=1.90 ms
64 bytes from auniversity.edu (152.181.8.3): icmp_seq=3    ttl=62    time=1.97 ms
64 bytes from auniversity.edu (152.181.8.3): icmp_seq=4    ttl=62    time=1.93 ms
64 bytes from auniversity.edu (152.181.8.3): icmp_seq=5    ttl=62    time=2.00 ms
--- auniversity.edu statistics ---
6 packets transmitted, 6 received, 0% packet loss
rtt min/avg/max = 1.90/1.95/2.04 ms
```

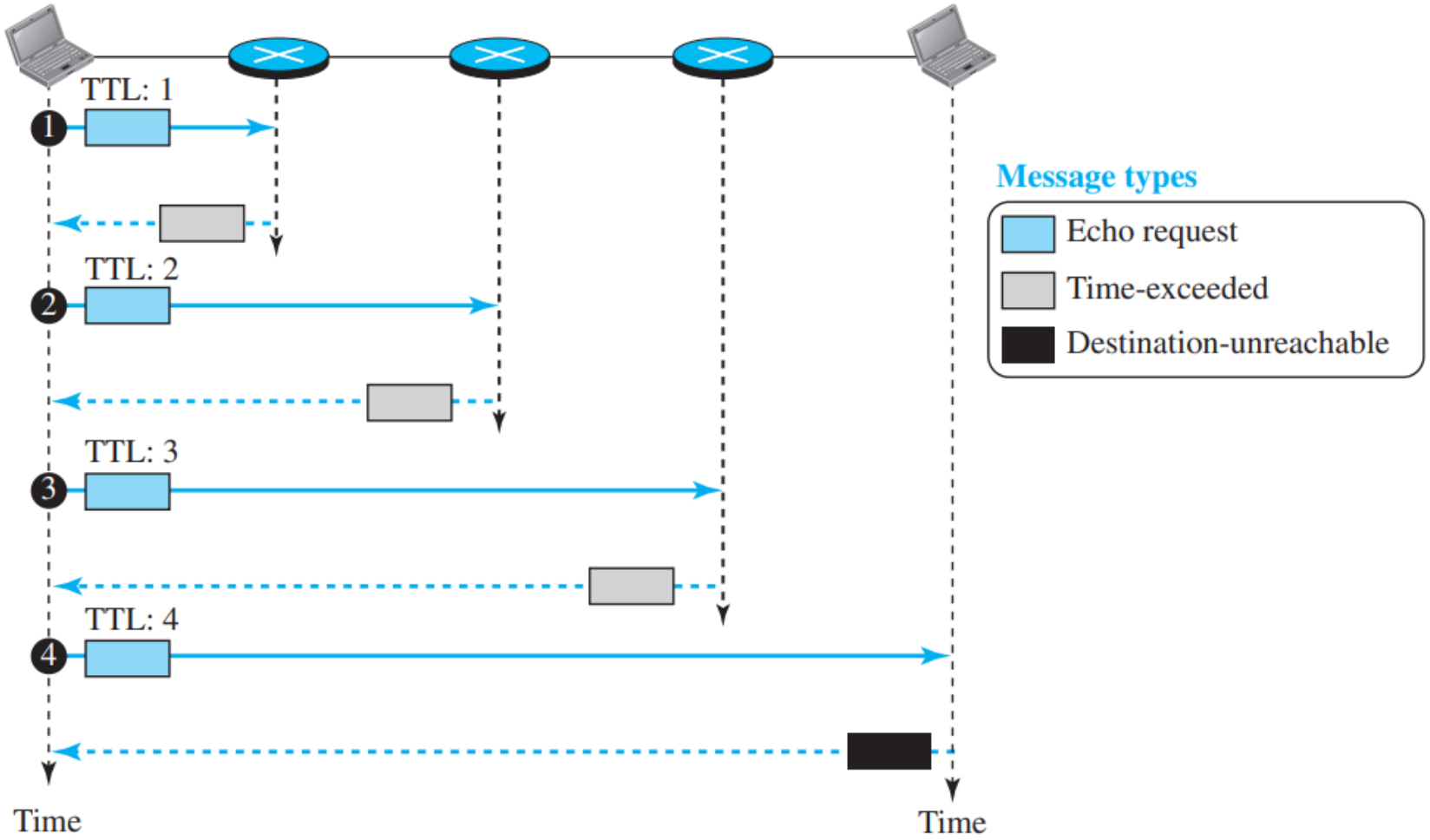
Traceroute or Tracert

- The traceroute program in UNIX or tracert in Windows can be used to trace the path of a packet from a source to the destination.
- It can find the IP addresses of all the routers that are visited along the path.
- The program is usually set to check for the maximum of 30 hops (routers) to be visited.
- The number of hops in the Internet is normally less than this. Since these two programs behave differently in Unix and Windows, we explain them separately

Traceroute

- the traceroute program gets help from two error-reporting messages: time-exceeded and destination-unreachable.
- The traceroute is an application layer program, there is no traceroute server program but only the client program is needed, because, as we can see, the client program never reaches the application layer in the destination host.
- The traceroute application program is encapsulated in a UDP user datagram, but traceroute intentionally uses a port number that is not available at the destination.
- If there are n routers in the path, the traceroute program sends (n + 1) messages. The first n messages are discarded by the n routers, one by each router; the last message is discarded by the destination host.
- The traceroute client program uses the (n + 1) ICMP error-reporting messages received to find the path between the routers.
- the value of n is found automatically.
- In Figure 19.10, the value of n is 3.
- The first traceroute message is sent with time-to-live (TTL) value set to 1; the message is discarded at the first router and a time-exceeded ICMP error message is sent, from which the traceroute program can find the IP address of the first router (the source IP address of the error message) and the router name (in the data section of the message).
- The second traceroute message is sent with TTL set to 2, which can find the IP address and the name of the second router.
- Similarly, the third message can find the information about router 3.
- The fourth message, however, reaches the destination host.
- This host is also dropped, but for another reason. The destination host cannot find the port number specified in the UDP user datagram.
- This time ICMP sends a different message, the destination-unreachable message with code 3 to show the port number is not found. After receiving this different ICMP message, the traceroute program knows that the final destination is reached. It uses the information in the received message to find the IP address and the name of the final destination.

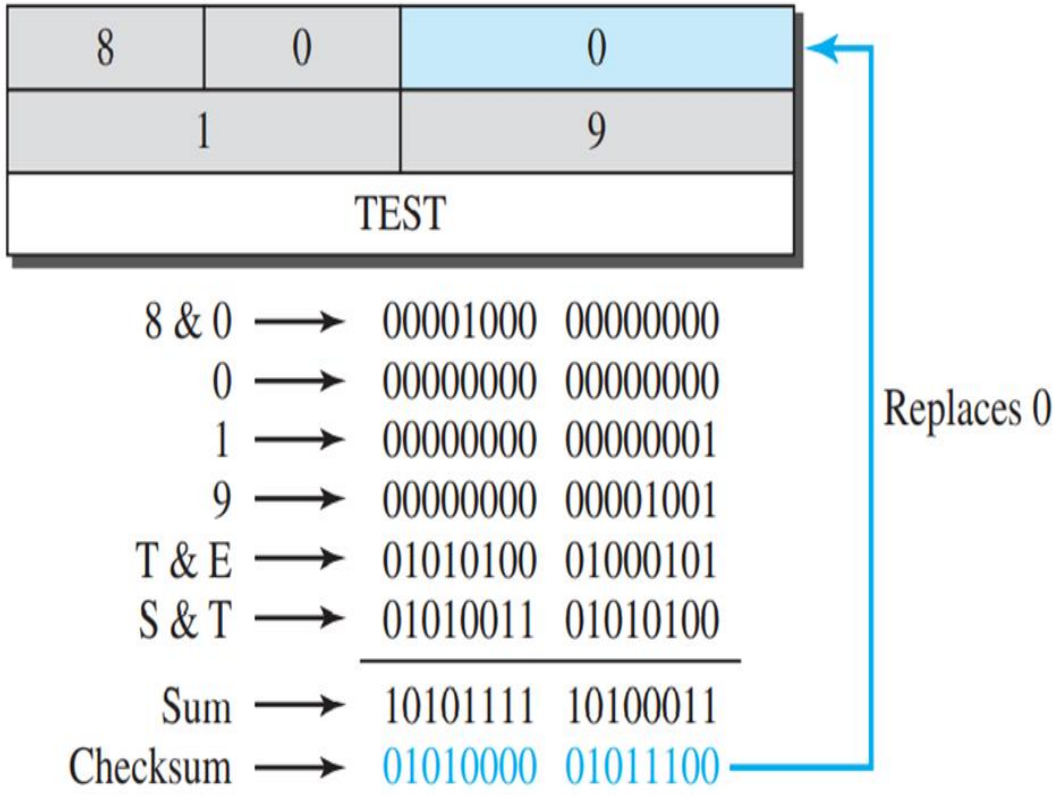
Figure 19.10 *Use of ICMPv4 in traceroute*



ICMP Checksum

- In ICMP the checksum is calculated over the entire message (header and data).
- Figure 19.11 shows an example of checksum calculation for a simple echo-request message. We randomly chose the identifier to be 1 and the sequence number to be 9. The message is divided into 16-bit (2-byte) words. The words are added and the sum is complemented. Now the sender can put this value in the checksum field.

Figure 19.11 *Example of checksum calculation*



Next Generation IP

- The address depletion of IPv4 and other shortcomings of this protocol prompted a new version of IP in the early 1990s.
- The new version, which is called Internet Protocol version 6 (IPv6) or IP new generation (IPng) was a proposal to augment the address space of IPv4
- The following lists the main changes in the IPv6 protocol: larger address space, better header format, new options, allowance for extension, support for resource allocation, and support for more security

IPv6 ADDRESSING

- The main reason for migration from IPv4 to IPv6 is the small size of the address space in IPv4.
- An IPv6 address is 128 bits or 16 bytes (octets) long, four times the address length in IPv4.
- Representation:
  - two notations: binary and colon hexadecimal.
  - Binary notation is used when the addresses are stored in a computer. The colon hexadecimal notation (or colon hex for short) divides the address into eight sections, each made of four hexadecimal digits separated by colons.

Binary (128 bits)	**1111111011110110 ... 1111111100000000**
Colon Hexadecimal	**FEF6:BA98:7654:3210:ADEF:BBFF:2922:FF00**

IPv6 ADDRESSING

- Although an IPv6 address, even in hexadecimal format, is very long, many of the digits are zeros. In this case, we can abbreviate the address. The leading zeros of a section can be omitted
- Often called zero compression, can be applied to colon hex notation if there are consecutive sections consisting of zeros only. We can remove all the zeros and replace them with a double semicolon.
- Note that this type of abbreviation is allowed only once per address. If there is more than one run of zero sections, only one of them can be compressed.
- Mixed Notation:
  - colon hex and dotted decimal notation.
  - This is appropriate during the transition period in which an IPv4 address is embedded in an IPv6 address (as the rightmost 32 bits).
  - We can use the colon hex notation for the leftmost six sections and four-byte dotted-decimal notation instead of the rightmost two sections.
  - However, this happens when all or most of the leftmost sections of the IPv6 address are 0s.
  - For example, the address (::130.24.24.18) is a legitimate address in IPv6, in which the zero compression shows that all 96 leftmost bits of the address are zeros.
  - **FDEC:0:0:0:0:BBFF:0:FFFF can be represented as FDEC::BBFF:0:FFFF**

CIDR Notation

IPv6 uses hierarchical addressing.
IPv6 allows slash or CIDR notation.
Ex. FDEC::BBFF:0:FFFF/60 Address Space:
IPv6 contains 2128 addresses. The size of the space is 340, 282, 366, 920, 938, 463, 374, 607, 431, 768, 211, 456.

Almost 2 percent of the addresses in the space can be assigned to the people on planet Earth (16 million) and the rest are reserved for special purposes. Each person can have 288 addresses to use. Address depletion in this version is impossible.

Three Address Types

- In IPv6, a destination address can belong to one of three categories: unicast, anycast, and multicast.
- Unicast Address
- A unicast address defines a single interface (computer or router). The packet sent to a unicast address will be routed to the intended recipient.
- Anycast Address
- An anycast address defines a group of computers that all share a single address. A packet with an anycast address is delivered to only one member of the group, the most reachable one.
- For example, when there are several servers that can respond to an inquiry. The request is sent to the one that is most reachable. The hardware and software generate only one copy of the request; the copy reaches only one of the servers. IPv6 does not designate a block for anycasting; the addresses are assigned from the unicast block.

Three Address Types

- **Multicast Address**
- A multicast address also defines a group of computers.
- Difference
  - In anycasting, only one copy of the packet is sent to one of the members of the group;
  - in multicasting each member of the group receives a copy.
- IPv6 has designated a block for multicasting from which the same address is assigned to the members of the group.
- IPv6 does not define broadcasting, even in a limited version.
- IPv6 considers broadcasting as a special case of multicasting.

22.1.3 Address Space Allocation

- Like the address space of IPv4, the address space of IPv6 is divided into several blocks of varying size and each block is allocated for a special purpose. Most of the blocks are still unassigned and have been set aside for future use. Table 22.1 shows only the assigned blocks. In this table, the last column shows the fraction each block occupies in the whole address space.

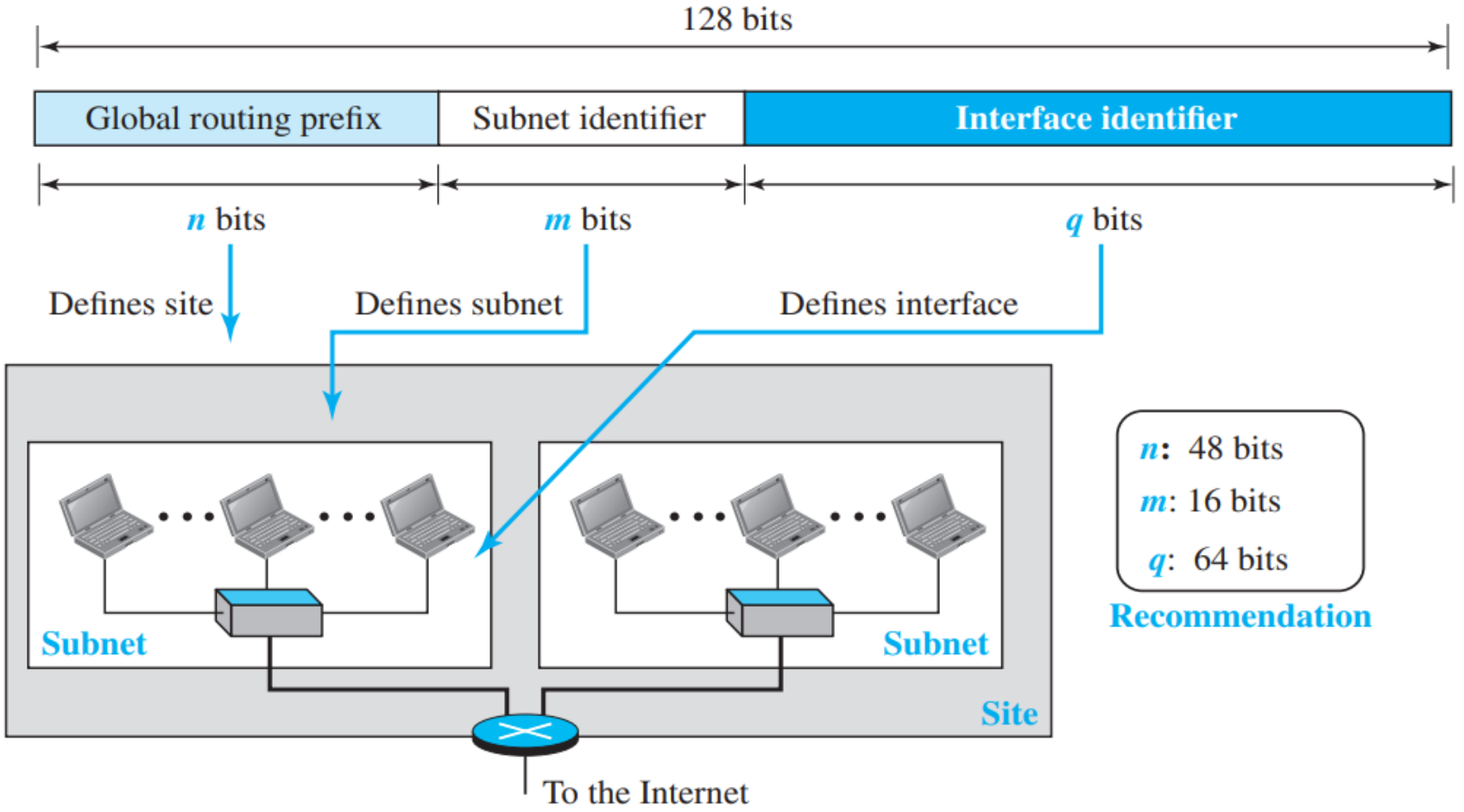
Table 22.1 *Prefixes for assigned IPv6 addresses*

Block prefix	*CIDR*	*Block assignment*	*Fraction*
0000 0000	0000::/8	Special addresses	1/256
001	**2000::/3**	**Global unicast**	**1/8**
1111 110	FC00::/7	Unique local unicast	1/128
1111 1110 10	FE80::/10	Link local addresses	1/1024
1111 1111	FF00::/8	Multicast addresses	1/256

Global Unicast Addresses

- The block in the address space that is used for unicast (one-to-one) communication between two hosts in the Internet is called the global unicast address block.
- CIDR for the block is 2000::/3, which means that the three leftmost bits are the same for all addresses in this block (001).
- The size of this block is 2125 bits, which is more than enough for Internet expansion for many years to come.
- An address in this block is divided into three parts: global routing prefix (n bits), subnet identifier (m bits), and interface identifier (q bits), as shown in Figure 22.1.
- The figure also shows the recommended length for each part.

Figure 22.1 *Global unicast address*



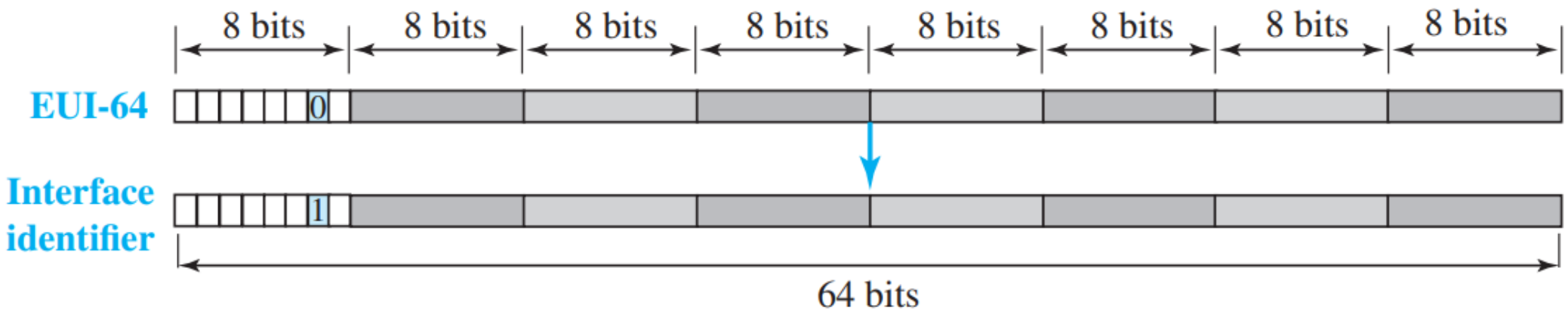
Global Unicast Addresses

- The global routing prefix is used to route the packet through the Internet to the organization site, such as the ISP that owns the block. Since the first three bits in this part are fixed (001), the rest of the 45 bits can be defined for up to 245 sites (a private organization or an ISP).
- The global routers in the Internet route a packet to its destination site based on the value of n. The next m bits (16 bits based on recommendation) define a subnet in an organization. This means that an organization can have up to 216 = 65,536 subnets, which is more than enough.
- The last q bits (64 bits based on recommendation) define the interface identifier. The interface identifier is similar to hostid in IPv4 addressing, although the term interface identifier is a better choice because, the host identifier actually defines the interface, not the host.
- If the host is moved from one interface to another, its IP address needs to be changed. In IPv4 addressing, there is not a specific relation between the hostid (at the IP level) and link-layer address (at the data-link layer) because the link-layer address is normally much longer than the hostid. The IPv6 addressing allows this relationship.
- A link-layer address whose length is less than 64 bits can be embedded as the whole or part of the interface identifier, eliminating the mapping process.
- Two common link layer addressing schemes can be considered for this purpose: the 64-bit extended unique identifier (EUI-64) defined by IEEE and the 48-bit link-layer address defined by Ethernet.

Mapping EUI-64

- To map a 64-bit physical address, the global/local bit of this format needs to be changed from 0 to 1 (local to global) to define an interface address, as shown in Figure 22.2.
- Mapping Ethernet MAC Address Mapping a 48-bit Ethernet address into a 64-bit interface identifier is more involved.
- We need to change the local/global bit to 1 and insert an additional 16 bits. The additional 16 bits are defined as 15 ones followed by one zero, or FFFE16. Figure 22.3 shows the mapping.

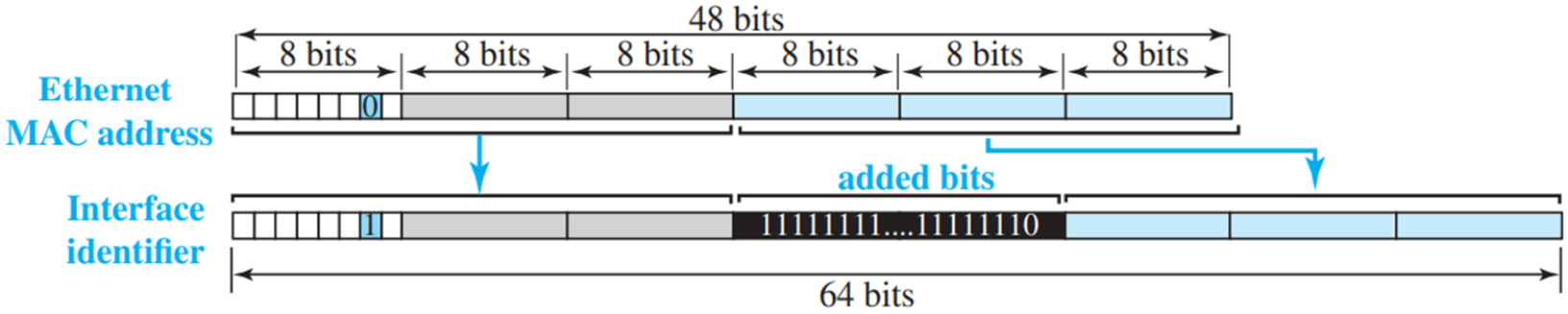
Figure 22.2 *Mapping for EUI-64*



Example problems

- Example 22.1
- An organization is assigned the block 2000:1456:2474/48. What is the CIDR notation for the blocks in the first and second subnets in this organization?

Figure 22.3 *Mapping for Ethernet MAC*



Solution

Theoretically, the first and second subnets should use the blocks with subnet identifier 000116 and 000216. This means that the blocks are 2000:1456:2474:0000/64 and 2000:1456:2474:0001/64.

Example problems

- Example 22.2
- Using the format we defined for Ethernet addresses, find the interface identifier if the physical address in the EUI is (F5-A9-23-EF-07-14-7A-D2)16.
- **Solution**
- We only need to change the seventh bit of the first octet from 0 to 1 and change the format to colon hex notation. The result is F7A9:23EF:0714:7AD2.
- Example 22.3
- Using the format we defined for Ethernet addresses, find the interface identifier if the Ethernet physical address is (F5-A9-23-14-7A-D2)16.
- **Solution**
- We only need to change the seventh bit of the first octet from 0 to 1, insert two octets FFFE16 and change the format to colon hex notation. The result is F7A9:23FF:FE14:7AD2 in colon hex.

Example problems

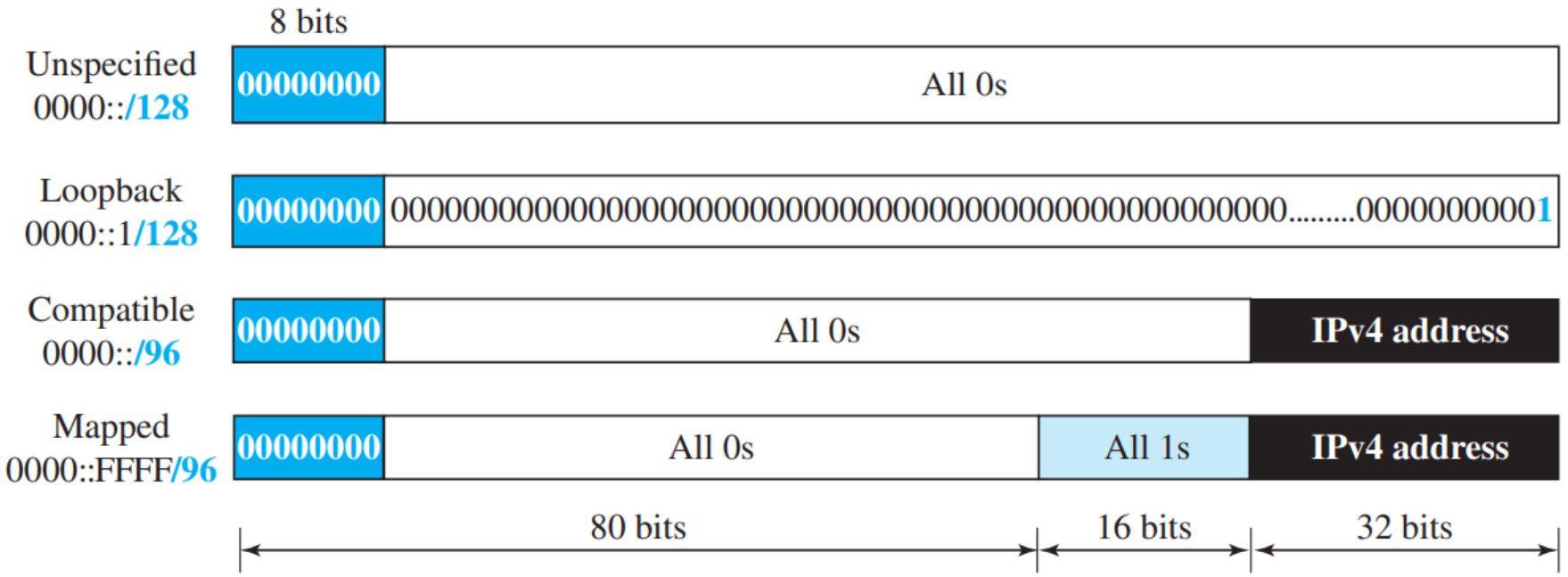
- Example 22.4
- An organization is assigned the block 2000:1456:2474/48. What is the IPv6 address of an interface in the third subnet if the IEEE physical address of the computer is (F5-A9-23-14-7A-D2)16?
- Solution
- The interface identifier for this interface is F7A9:23FF:FE14:7AD2. If we append this identifier to the global prefix and the subnet identifier, we get:
  - **2000:1456:2474:0003:F7A9:23FF:FE14:7AD2/128**

Special Addresses

- Characteristics and purposes of assigned and reserved blocks are discussed in the first row of Table 22.1. Addresses that use the prefix (0000::/8) are reserved, but part of this block is used to define some special addresses. Figure 22.4 shows the assigned addresses in this block.
- Mapping for Ethernet MAC 2000:1456:2474:0003:F7A9:23FF:FE14:7AD2/128

Special Addresses

Figure 22.4 *Special addresses*

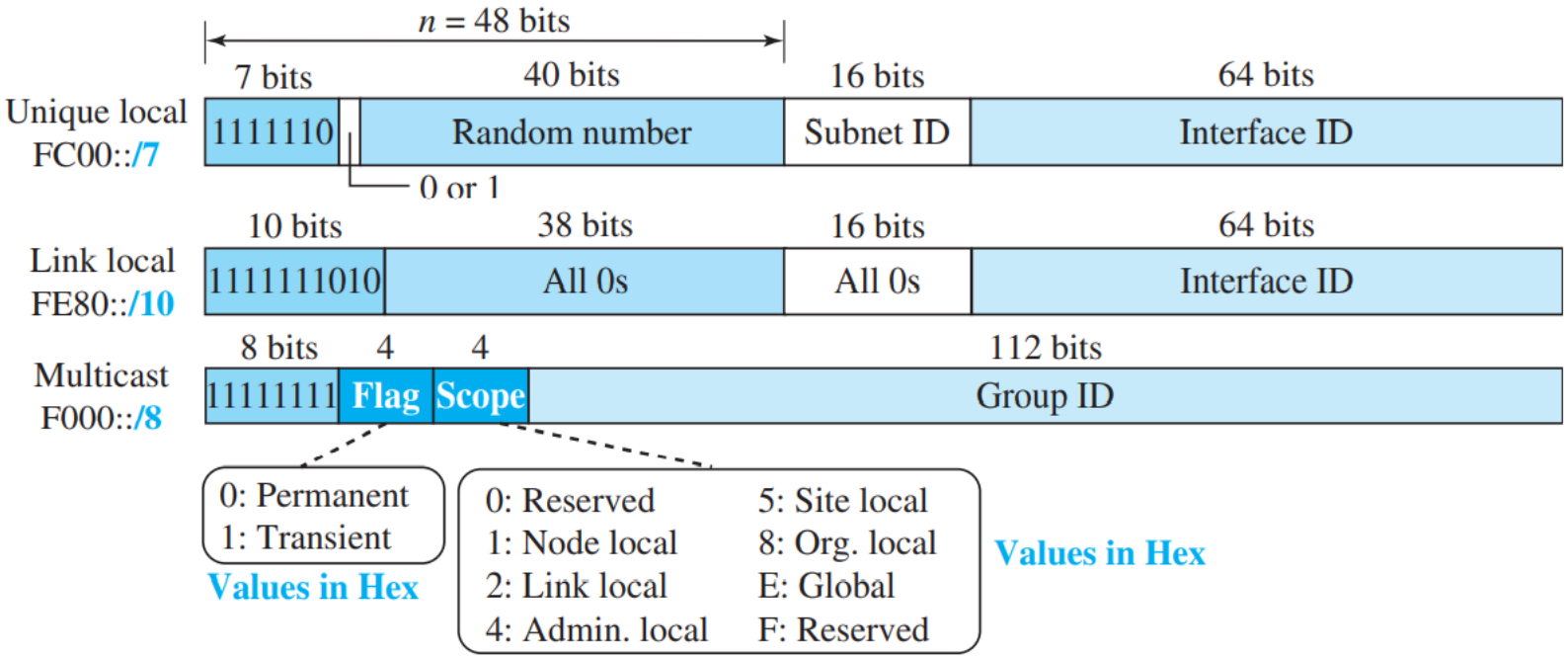


Special Addresses

- The unspecified address is a subblock containing only one address, which is used during bootstrap when a host does not know its own address and wants to send an inquiry to find.
- The loopback address also consists of one address.
  - In IPv4 the block is made of a range of addresses; in IPv6, the block has only a single address in it
  - During the transition from IPv4 to IPv6, hosts can use their IPv4 addresses embedded in IPv6 addresses.
  - Two formats have been designed for this purpose: compatible and mapped.
    - A compatible address is an address of 96 bits of zero followed by 32 bits of IPv4 address. It is used when a computer using IPv6 wants to send a message to another computer using IPv6.
    - A mapped address is used when a computer already migrated to version 6 wants to send an address to a computer still using version 4.
    - Mapped and compatible addresses is that they are designed such that, when calculating the checksum, one can use either the embedded address or the total address because extra 0s or 1s in multiples of 16 do not have any effect in checksum calculation.
    - This is important for UDP and TCP, which use a pseudoheader to calculate the checksum, because the checksum calculation is not affected if the address of the packet is changed from IPv6 to IPv4 by a router.
    - Other assigned blocks IPv6 uses two large blocks for private addressing and one large block for multicasting, as shown in Figure 22.5.
    - A subblock in a unique local unicast block can be privately created and used by a site (a private address in a network).

Special Addresses

Figure 22.5 *Unique local unicast block*



Special Addresses

- Subblocks:
  - The packet carrying this type of address as the destination address is not expected to be routed. This type of address has the identifier 1111 110, the next bit can be 0 or 1 to define how the address is selected (locally or by an authority). The next 40 bits are selected by the site using a randomly generated number of length 40 bits. This means that the total of 48 bits defines a subblock that looks like a global unicast address. The 40-bit random number makes the probability of duplication of the address extremely small. Note the similarity between the format of these addresses and the global unicast. The second block, designed for private addresses, is the link local block.
  - This type of address has the block identifier 1111111010. The next 54 bits are set to zero. The last 64 bits can be changed to define the interface for each computer. Note the similarity between the format of these addresses and the global unicast address. We discussed multicast addresses of IPv4 earlier in the chapter. Multicast addresses are used to define a group of hosts instead of just one. In IPv6 a large block of addresses are assigned for multicasting. All these addresses use the prefix 11111111. The second field is a flag that defines the group address as either permanent or transient.
  - A permanent group address is defined by the Internet authorities and can be accessed at all times. A transient group address, on the other hand, is used only temporarily. Systems engaged in a teleconference, for example, can use a transient group address. The third field defines the scope of the group address.

Autoconfiguration

- One of the interesting features of IPv6 addressing is the autoconfiguration of hosts.
- In IPv4, the host and routers are originally configured manually by the network manager. However, the Dynamic Host Configuration Protocol, DHCP, can be used to allocate an IPv4 address to a host that joins the network. In IPv6, DHCP protocol can still be used to allocate an IPv6 address to a host, but a host can also configure itself.
- When a host in IPv6 joins a network, it can configure itself using the following process:
- 1. The host first creates a link local address for itself. This is done by taking the 10-bit link local prefix (1111 1110 10), adding 54 zeros, and adding the 64-bit interface identifier, which any host knows how to generate from its interface card. The result is a 128-bit link local address.
- 2. The host then tests to see if this link local address is unique and not used by other hosts. Since the 64-bit interface identifier is supposed to be unique, the link local

Examples

- **Example 22.5**
- Assume a host with Ethernet address (F5-A9-23-11-9B-E2)16 has joined the network. What would be its global unicast address if the global unicast prefix of the organization is 3A21:1216:2165 and the subnet identifier is A245:1232?
- **Solution**
- The host first creates its interface identifier as F7A9:23FF:FE11:9BE2 using the Ethernet address read from its card. The host then creates its link local address as: FE80::F7A9:23FF:FE11:9BE2
- Assuming that this address is unique, the host sends a router solicitation message and receives the router advertisement message that announces the combination of global unicast prefix and the subnet identifier as 3A21:1216:2165:A245:1232. The host then appends its interface identifier to this prefix to find and store its global unicast address as: 3A21:1216:2165:A245:1232:F7A9:23FF:FE11:9BE2

Renumbering

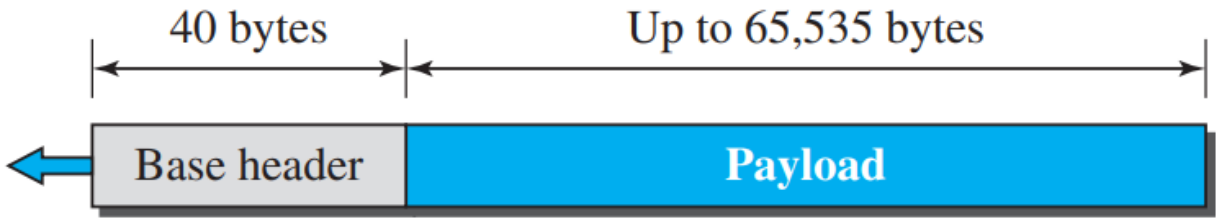
- To allow sites to change the service provider, renumbering of the address prefix (n) was built into IPv6 addressing. Each site is given a prefix by the service provider to which it is connected. If the site changes the provider, the address prefix needs to be changed. A router to which the site is connected can advertise a new prefix and let the site use the old prefix for a short time before disabling it. During the transition period, a site has two prefixes.
- The main problem in using the renumbering mechanism is the support of the DNS, which needs to propagate the new addressing associated with a domain name. A new protocol for DNS, called **Next Generation DNS**, is under study to provide support for this mechanism.

THE IPv6 PROTOCOL

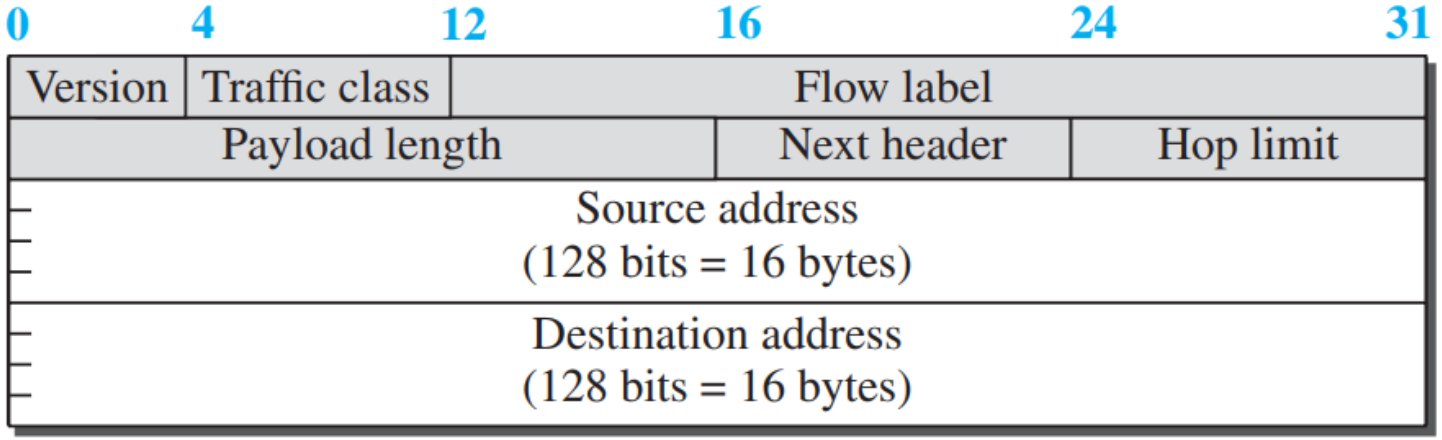
- The following shows changes (other that address size and format) implemented in the protocol
- Better header format.
  - IPv6 uses a new header format in which options are separated from the base header and inserted, when needed, between the base header and the data. This simplifies and speeds up the routing process because most of the options do not need to be checked by routers.
- New options.
  - IPv6 has new options to allow for additional functionalities.
- Allowance for extension.
  - IPv6 is designed to allow the extension of the protocol if required by new technologies or applications.
- Support for resource allocation.
  - In IPv6, the type-of-service field has been removed, but two new fields, traffic class and flow label, have been added to enable the source to request special handling of the packet. This mechanism can be used to support traffic such as real-time audio and video.
- Support for more security.
  - The encryption and authentication options in IPv6 provide confidentiality and integrity of the packet.

Packet Format

Figure 22.6 *IPv6 datagram*



a. IPv6 packet

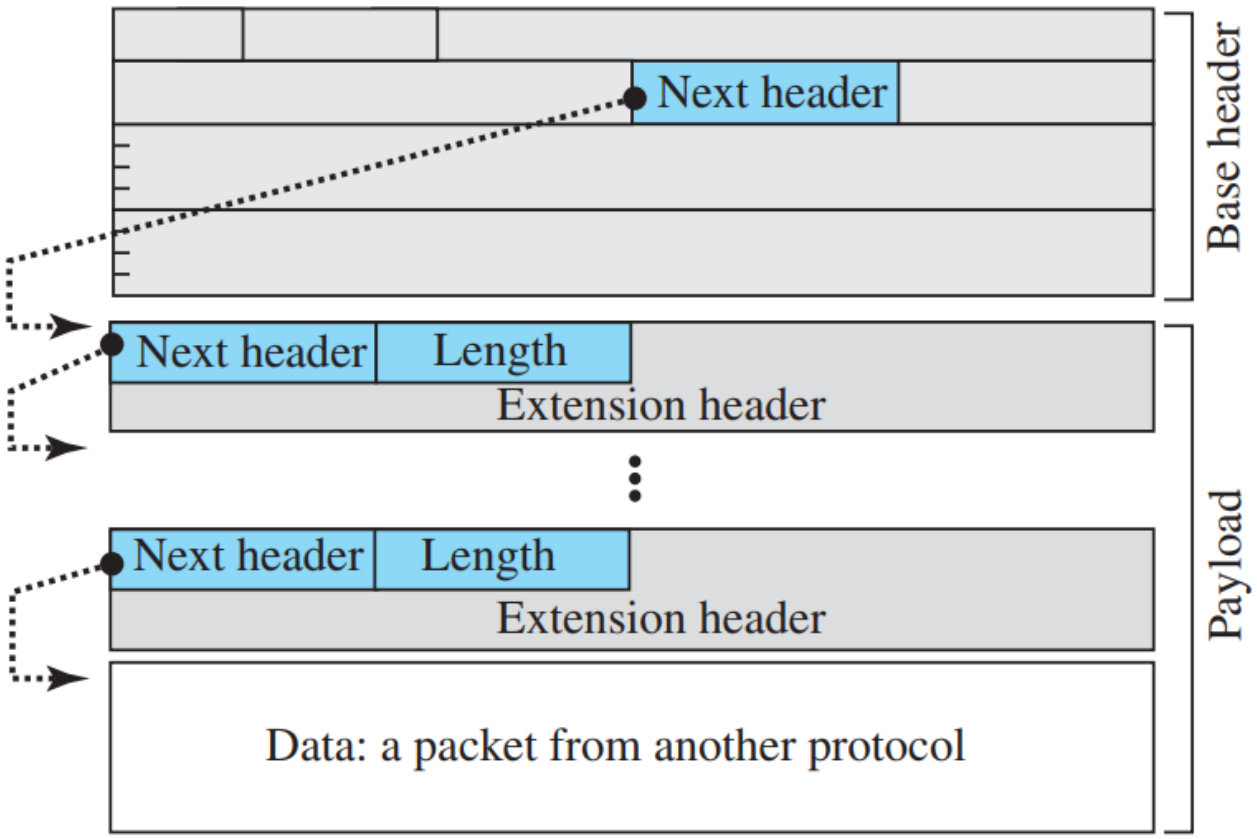


b. Base header

Packet Format

- Version.
  - The 4-bit version field defines the version number of the IP. For IPv6, the value is 6.
- Traffic class.
  - The 8-bit traffic class field is used to distinguish different payloads with different delivery requirements. It replaces the type-of-service field in IPv4.
- Flow label.
  - The flow label is a 20-bit field that is designed to provide special handling for a particular flow of data. We will discuss this field later.
- Payload length.
  - The 2-byte payload length field defines the length of the IP datagram excluding the header. Note that IPv4 defines two fields related to the length: header length and total length. In IPv6, the length of the base header is fixed (40 bytes); only the length of the payload needs to be defined.
- Next header.
  - The next header is an 8-bit field defining the type of the first extension header (if present) or the type of the data that follows the base header in the datagram. This field is similar to the protocol field in IPv4, but we talk more about it when we discuss the payload.
- Hop limit.
  - The 8-bit hop limit field serves the same purpose as the TTL field in IPv4.
- Source and destination addresses.
  - The source address field is a 16-byte (128-bit) Internet address that identifies the original source of the datagram. The destination address field is a 16-byte (128-bit) Internet address that identifies the destination of the datagram.
- Payload.
  - Compared to IPv4, the payload field in IPv6 has a different format and meaning

Figure 22.7 *Payload in an IPv6 datagram*



Some next-header codes

00: Hop-by-hop option
02: ICMPv6
06: TCP
17: UDP
43: Source-routing option
44: Fragmentation option
50: Encrypted security payload
51:Authentication header
59: Null (no next header)
60: Destination option

Packet Format

- The payload in IPv6 means a combination of zero or more extension headers (options) followed by the data from other protocols (UDP, TCP, and so on).
- In IPv6, options, which are part of the header in IPv4, are designed as extension headers.
- The payload can have as many extension headers as required by the situation. Each extension header has two mandatory fields, next header and the length, followed by information related to the particular option. Note that each next header field value (code) defines the type of the next header (hop-by-hop option, source routing option, ...); the last next header field defines the protocol (UDP, TCP, . . .) that is carried by the datagram.

Concept of Flow and Priority in IPv6

- The IP protocol was originally designed as a connectionless protocol. the tendency is to use the IP protocol as a connection-oriented protocol. The MPLS technology allows encapsulation of an IPv4 packet in an MPLS header using a label field. In version 6, the flow label has been directly added to the format of the IPv6 datagram to allow us to use IPv6 as a connection-oriented protocol.
- To a router, a flow is a sequence of packets that share the same characteristics, such as traveling the same path, using the same resources, having the same kind of security, and so on. A router that supports the handling of flow labels has a flow label table. The table has an entry for each active flow label; each entry defines the services required by the corresponding flow label. When the router receives a packet, it consults its flow label table to find the corresponding entry for the flow label value defined in the packet.
- It then provides the packet with the services mentioned in the entry. However, note that the flow label itself does not provide the information for the entries of the flow label table; the information is provided by other means, such as the hop-by-hop options or other protocols.
- In its simplest form, a flow label can be used to speed up the processing of a packet by a router. When a router receives a packet, instead of consulting the forwarding table and going through a routing algorithm to define the address of the next hop, it can easily look in a flow label table for the next hop.
- In its more sophisticated form, a flow label can be used to support the transmission of real-time audio and video. Real-time audio or video, particularly in digital form, requires resources such as high bandwidth, large buffers, long processing time, and so on. A process can make a reservation for these resources beforehand to guarantee that real-time data will not be delayed due to a lack of resources. The use of real-time data and the reservation of these resources require other protocols such as Real-Time Transport Protocol (RTP) and Resource Reservation Protocol (RSVP) in addition to IPv6

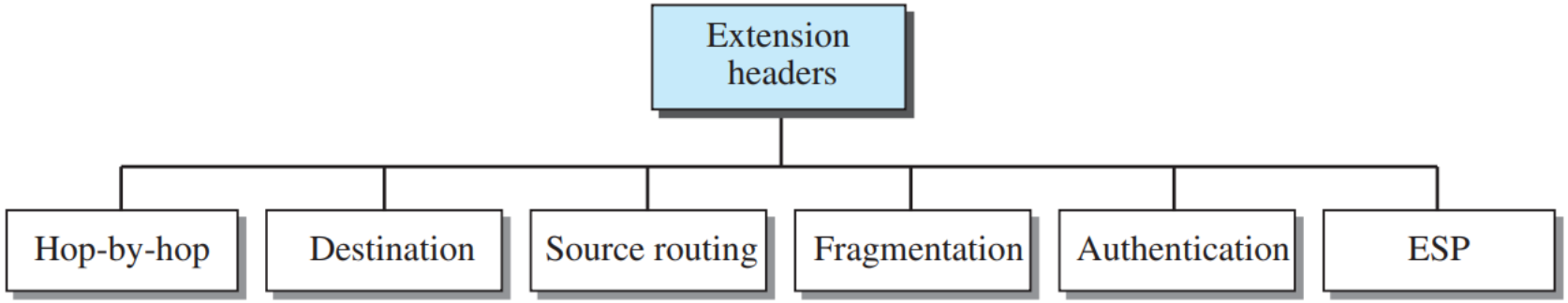
Fragmentation and Reassembly

- There are still fragmentation and reassembly of datagrams in the IPv6 protocol, but there is a major difference in this respect.
- IPv6 datagrams can be fragmented only by the source, not by the routers; the reassembly takes place at the destination.
- The fragmentation of packets at routers is not allowed to speed up the processing of packets in the router. The fragmentation of a packet in a router needs a lot of processing. The packet needs to be fragmented, all fields related to the fragmentation need to be recalculated.
- In IPv6, the source can check the size of the packet and make the decision to fragment the packet or not. When a router receives the packet, it can check the size of the packet and drop it if the size is larger than allowed by the MTU of the network ahead.
- The router then sends a packet-too-big ICMPv6 error message to inform the source.

Extension Header

- An IPv6 packet is made of a base header and some extension headers.
- The length of the base header is fixed at 40 bytes.
- Base header can be followed by up to six extension headers.
- Many of these headers are options in IPv4.
- Six types of extension headers have been defined.
- These are hop-by-hop option, source routing, fragmentation, authentication, encrypted security payload, and destination option

Figure 22.8 *Extension header types*



Extension Header

- The **hop-by-hop option** is used when the source needs to pass information to all routers visited by the datagram. For example, perhaps routers must be informed about certain management, debugging, or control functions. Or, if the length of the datagram is more than the usual 65,535 bytes, routers must have this information.
- Only three hop-by-hop options have been defined: Pad1, PadN, and jumbo payload.
  - Pad1. This option is 1 byte long and is designed for alignment purposes. Some options need to start at a specific bit of the 32-bit word. If an option falls short of this requirement by exactly one byte, Pad1 is added.
  - PadN. PadN is similar in concept to Pad1. The difference is that PadN is used when 2 or more bytes are needed for alignment.
  - Jumbo payload. Recall that the length of the payload in the IP datagram can be a maximum of 65,535 bytes. However, if for any reason a longer payload is required, we can use the jumbo payload option to define this longer length.

Extension Header

- **Destination Option :**
  - The destination option is used when the source needs to pass information to the destination only.
  - Intermediate routers are not permitted access to this information. The format of the destination option is the same as the hop-by-hop option. So far, only the Pad1 and PadN options have been defined.
- **Source Routing:** The source routing extension header combines the concepts of the strict source route and the loose source route options of IPv4.
- **Fragmentation**
  - The concept of fragmentation in IPv6 is the same as that in IPv4.
  - The place where fragmentation occurs differs.
  - In IPv4, the source or a router is required to fragment if the size of the datagram is larger than the MTU of the network over which the datagram travels.
  - In IPv6, only the original source can fragment.
  - A source must use a Path MTU Discovery technique to find the smallest MTU supported by any network on the path.
  - The source then fragments using this knowledge. If the source does not use a Path MTU Discovery technique, it fragments the datagram to a size of 1280 bytes or smaller. This is the minimum size of MTU required for each network connected to the Internet. Authentication The authentication extension header has a dual purpose: it validates the message sender and ensures the integrity of data. The former is needed so the receiver can be sure that a message is from the genuine sender and not from an imposter. The latter is needed to check that the data is not altered in transition by some hacker.

Encrypted Security Payload

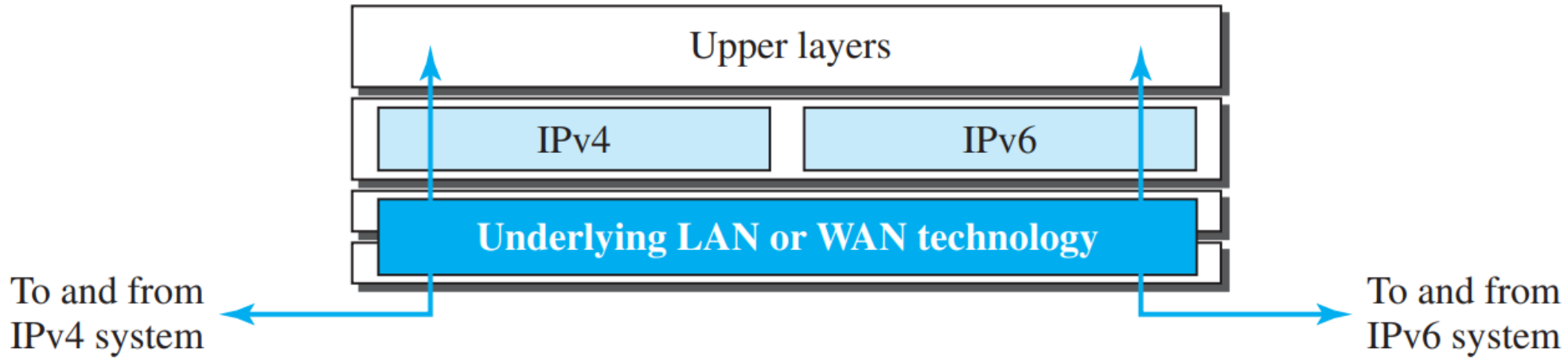
- The **encrypted security payload (ESP)** is an extension that provides confidentiality and guards against eavesdropping.
- **Comparison of Options between IPv4 and IPv6**
- The following shows a quick comparison between the options used in IPv4 and the options used in IPv6 (as extension headers).
- The no-operation and end-of-option options in IPv4 are replaced by Pad1 and PadN options in IPv6.
- The record route option is not implemented in IPv6 because it was not used.
- The timestamp option is not implemented because it was not used.
- The source route option is called the source route extension header in IPv6.
- The fragmentation fields in the base header section of IPv4 have moved to the fragmentation extension header in IPv6.
- The authentication extension header is new in IPv6.
- The encrypted security payload extension header is new in IPv6.

TRANSITION FROM IPv4 TO IPv6

- Although we have a new version of the IP protocol, how can we make the transition to stop using IPv4 and start using IPv6? The first solution that comes to mind is to define a transition day on which every host or router should stop using the old version and start using the new version. However, this is not practical; because of the huge number of systems in the Internet, the transition from IPv4 to IPv6 cannot happen suddenly. It will take a considerable amount of time before every system in the Internet can move from IPv4 to IPv6. The transition must be smooth to prevent any problems between IPv4 and IPv6 systems.
- **22.4.1 Strategies**
- Three strategies have been devised for transition: dual stack, tunneling, and header translation. One or all of these three strategies can be implemented during the transition period.
- **Dual Stack**
- It is recommended that all hosts, before migrating completely to version 6, have a dual stack of protocols during the transition. In other words, a station must run IPv4 and IPv6 simultaneously until all the Internet uses IPv6. See Figure 22.11 for the layout of a dual-stack configuration.

TRANSITION FROM IPv4 TO IPv6

Figure 22.11 *Dual stack*



TRANSITION FROM IPv4 TO IPv6

- To determine which version to use when sending a packet to a destination, the source host queries the DNS. If the DNS returns an IPv4 address, the source host sends an IPv4 packet. If the DNS returns an IPv6 address, the source host sends an IPv6 packet.
- **Tunneling:**
- Tunneling is a strategy used when two computers using IPv6 want to communicate with each other and the packet must pass through a region that uses IPv4. To pass through this region, the packet must have an IPv4 address.
- So the IPv6 packet is encapsulated in an IPv4 packet when it enters the region, and it leaves its capsule when it exits the region. It seems as if the IPv6 packet enters a tunnel at one end and emerges at the other end. To make it clear that the IPv4 packet is carrying an IPv6 packet as data, the protocol value is set to 41.
- **Header Translation:**
- Header translation is necessary when the majority of the Internet has moved to IPv6 but some systems still use IPv4. The sender wants to use IPv6, but the receiver does not understand IPv6. Tunneling does not work in this situation because the packet must be in the IPv4 format to be understood by the receiver. In this case, the header format must be totally changed through header translation. The header of the IPv6 packet is converted to an IPv4 header

Figure 22.12 *Tunneling strategy*

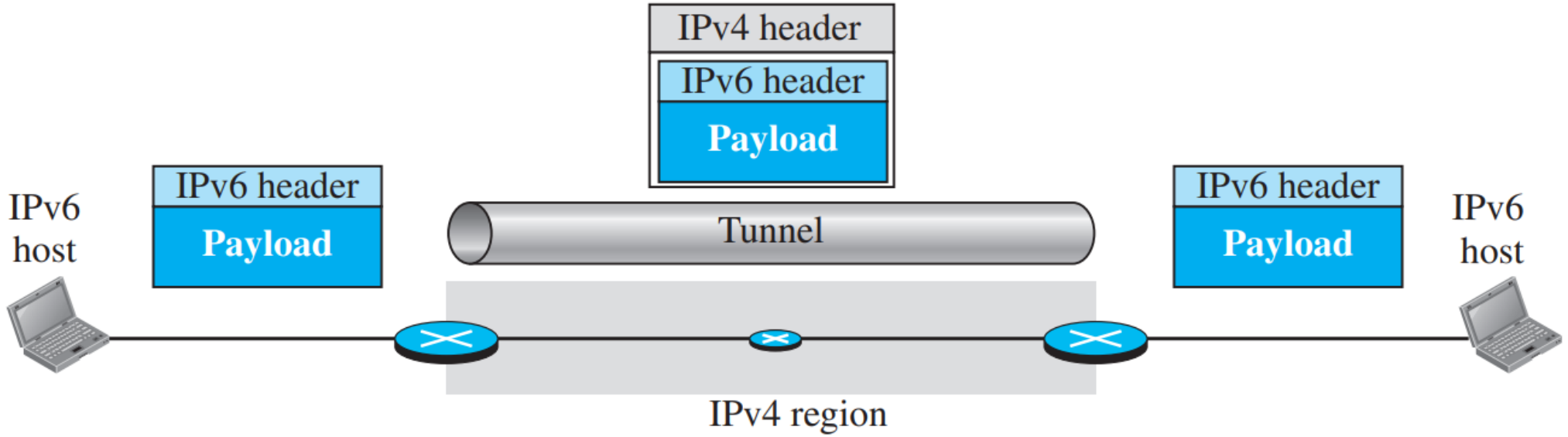
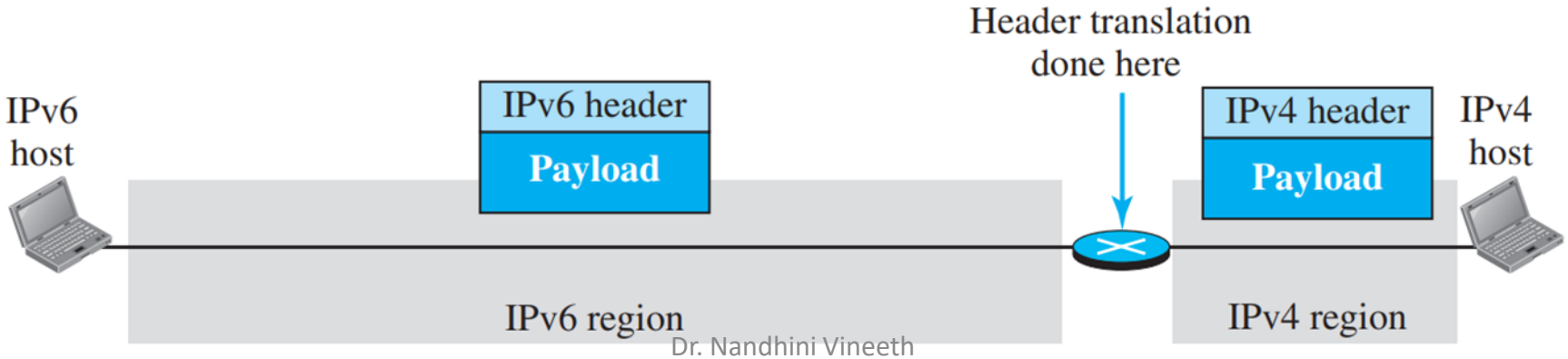


Figure 22.13 *Header translation strategy*



Use of IP Addresses

- During the transition a host may need to use two addresses, IPv4 and IPv6. When the transition is complete, IPv4 addresses should disappear. The DNS servers need to be ready to map a host name to either address type during the transition, but the IPv4 directory will disappear after all hosts in the world have migrated to IPv6.

PAGE NUMBERS OF REMAINING TOPICS

UNIT III
ICMPV4 - 614
IPV6 - 705

UNIT IV
23.1—732
23.2 TILL 767
24.1—776
24.3 – TILL 830

UNIT V
25.1—857-858
26.1-911

26.6- TILL 961