

COMPUTER NETWORK

(BCSC 0008)

MODULE-2

NOTES

BY: ROHIT AGARWAL

(Teaching Associate)

(CSE Dept)

MODULE-2

- **NETWORK LAYER**
- **TRANSPORT LAYER**
- **APPLICATION LAYER**
- **IPv4 VS IPv6**

NETWORK LAYER



IP Address

[*ī-pē ə-'dres*]

A number used to identify a computer or network of computers.

DOTTED DECIMAL NOTATION OF IP ADDRESS

IPv4 address in dotted-decimal notation

172 . 16 . 254 . 1



10101100.00010000.11111110.00000001

 8 bits

 32 bits (4 bytes)

HEXADECIMAL NOTATION OF IP ADDRESS

01110101

75

00011101

1D

10010101

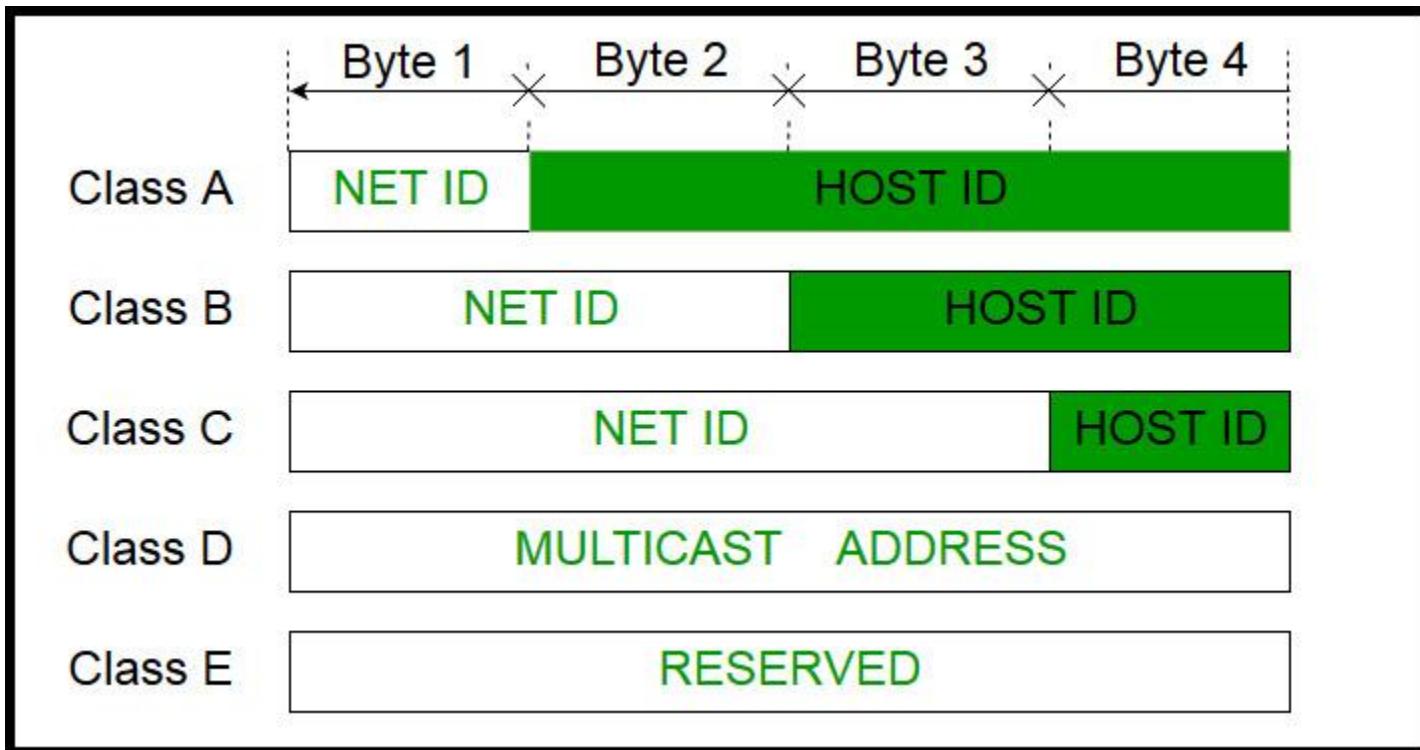
95

11101010

EA

0x751D95EA

IP ADDRESS CLASSES

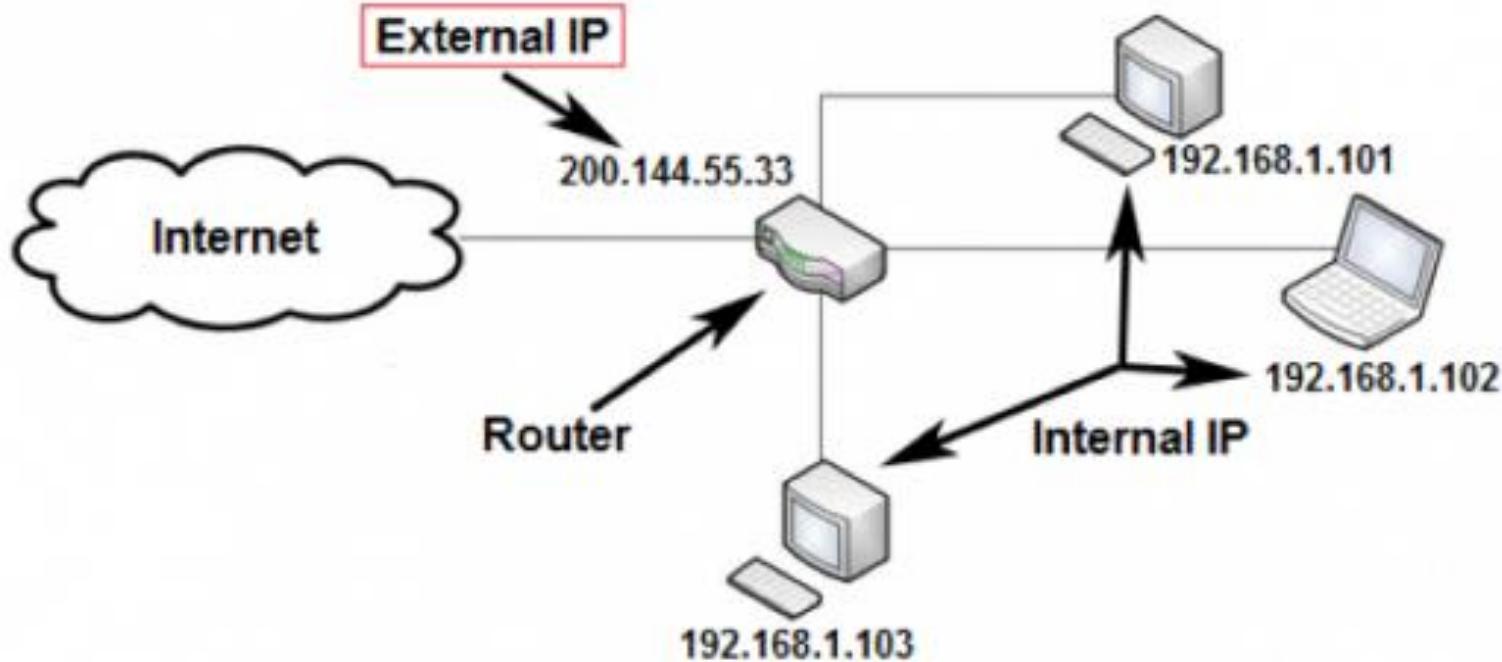


IP ADDRESS CLASSES

Address Class	RANGE	Default Subnet Mask
A	1.0.0.0 to 126.255.255.255	255.0.0.0
B	128.0.0.0 to 191.255.255.255	255.255.0.0
C	192.0.0.0 to 223.255.255.255	255.255.255.0
D	224.0.0.0 to 239.255.255.255	Reserved for Multicasting
E	240.0.0.0 to 254.255.255.255	Experimental

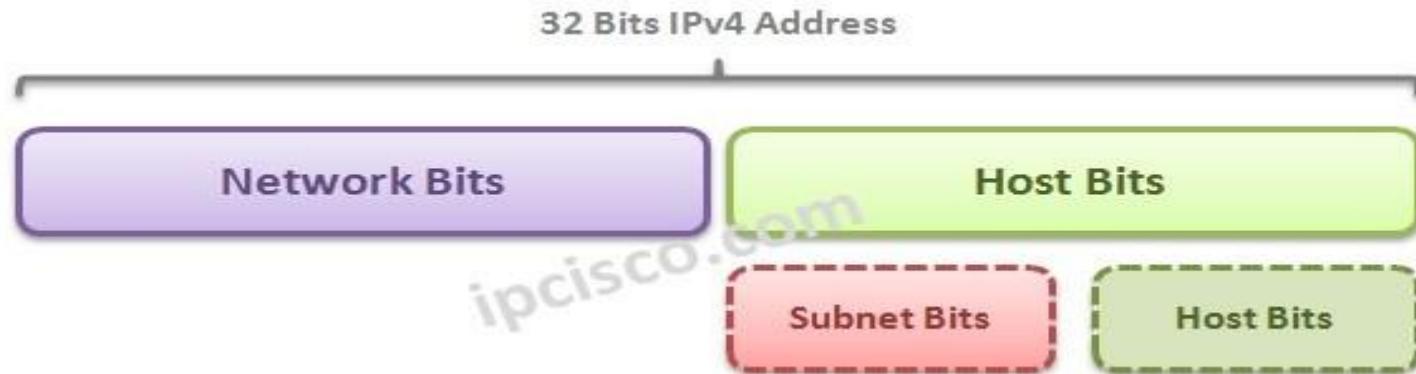
Note: Class A addresses 127.0.0.0 to 127.255.255.255 cannot be used and is reserved for loopback testing.

IP ADDRESS DISTRIBUTION WITHIN A NETWORK



Subnetting

- **What is subnetting**
 - Process of subdividing a single class of network into multiple subnetworks.
 - A subnetted network address contains a network address, **subnet address** and host address.



SUBNETTING

DEFINITION

It is a technique in which larger network is divided in to smaller sub networks , so that those sub networks will not communicate with each other.

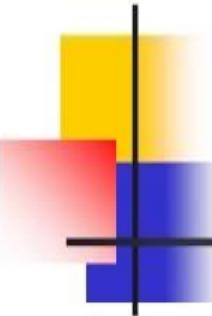
OR

The process of converting host bits in to network bit is called sub netting.

Types of Sub -netting

There are two types of sub- netting

1. Network based sub -netting
2. Host based sub -netting



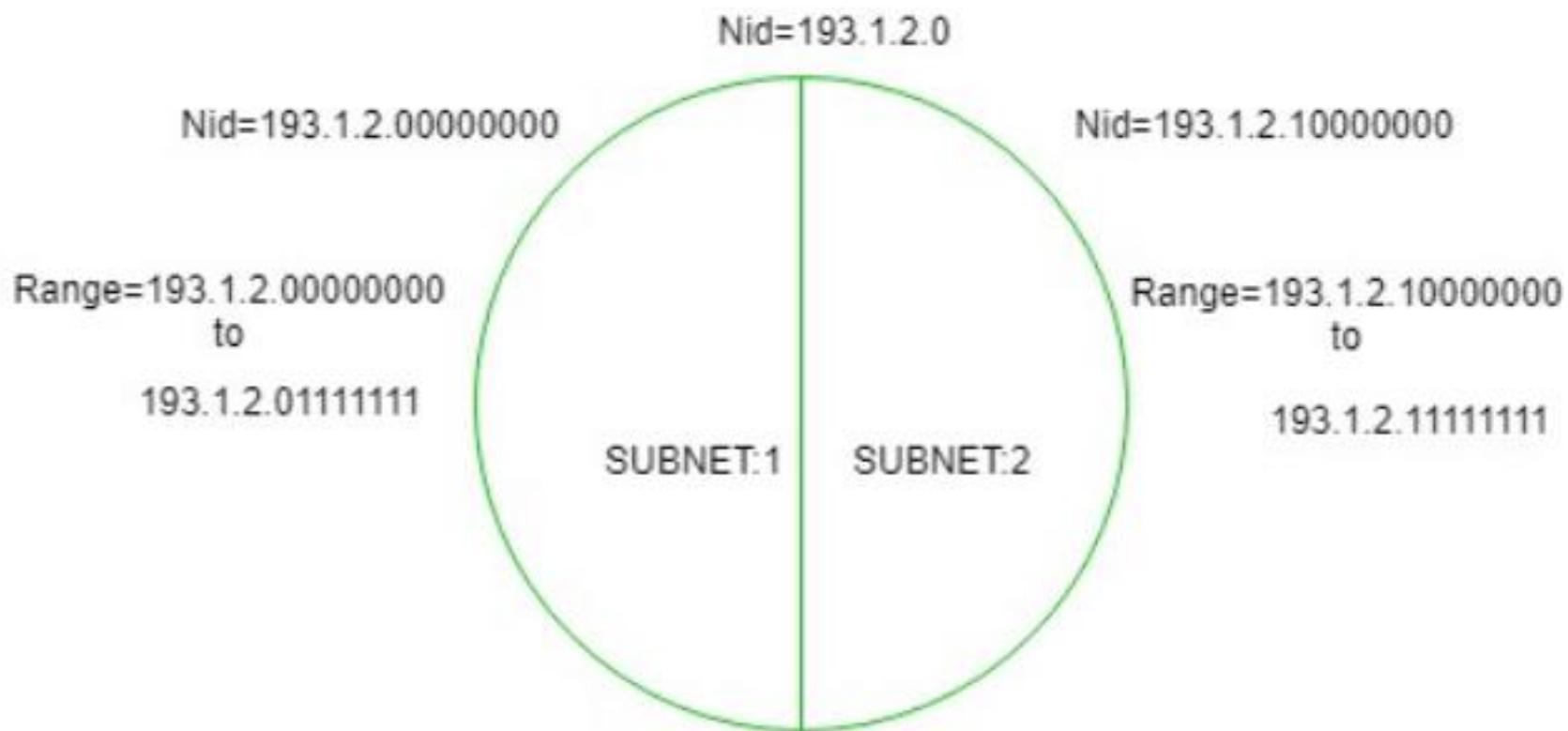
What is a Subnet Mask?

- An Address that accompanies an IP address that indicates which portion of the IP address is the Network ID and which portion of the IP address is the Host ID.
 - 152.107.102.7 (IP Address)
 - 255.255.255.0 (Subnet Mask)
- The IP Address and Subnet Mask (SNM) are interrelated and each only has meaning in the context of the other!

Subnet Masks

- Tells the device which bits are host address and network address.

Class	Subnet Mask	Binary
A	255.0.0.0	11111111.00000000.00000000.00000000
B	255.255.0.0	11111111. 11111111. 00000000.00000000
C	255.255.255.0	11111111. 11111111. 11111111.00000000



- **For Subnet-1:** The first bit which is chosen from the host id part is zero and the range will be from (193.1.2.00000000 till you get all 1's in the host ID part i.e, 193.1.2.01111111) except for the first bit which is chosen zero for subnet id part. Thus, the range of subnet-1:

193.1.2.0 to 193.1.2.127

Subnet id of Subnet-1 is : 193.1.2.0

Direct Broadcast id of Subnet-1 is : 193.1.2.127

Total number of host possible is : 126 (Out of 128, 2 id's are used for Subnet id & Direct Broadcast id)

- **For Subnet-2:** The first bit chosen from the host id part is one and the range will be from (193.1.2.100000000 till you get all 1's in the host ID part i.e, 193.1.2.11111111). Thus, the range of subnet-2:

193.1.2.128 to 193.1.2.255

Subnet id of Subnet-2 is : 193.1.2.128

Direct Broadcast id of Subnet-2 is : 193.1.2.255

Total number of host possible is : 126 (Out of 128, 2 id's are used for Subnet id & Direct Broadcast id)

Example1. An organization is assigned a class C network address of 201.35.2.0. It uses a netmask of 255.255.255.192 to divide this into sub-networks. Which of the following is/are valid host IP addresses?

- A. 201.35.2.129
- B. 201.35.2.191
- C. 201.35.2.255
- D. Both (A) and (C)

Example 2. An organization has a class C network address of 201.32.64.0. It uses a subnet mask of 255.255.255.248. Which of the following is NOT a valid broadcast address for any subnetworks?

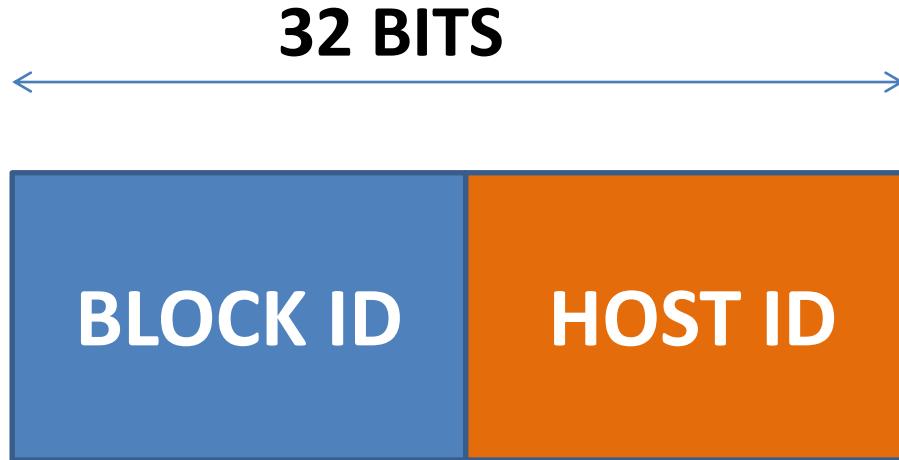
- A. 201.32.64.135
- B. 201.32.64.240
- C. 201.32.64.207
- D. 201.32.64.231

Disadvantages of Subnetting

- Subnetting can become confusing as networks grow larger.
- Subnetting itself is not an easy process if the administrator does not practice.

CIDR – Classless InterDomain Routing

- Running out of IP addresses
- class C is too small; class B is too large (more than half of the class B networks have fewer than 50 hosts)
- CIDR (Classless InterDomain Routing) allows to allocate IP address with a variable-sized block (contiguous network numbers to nearby networks), with no regard to the classes.



Notation Used

x.y.z.w/n

Here n shows number of network bits

For eg. 200.10.20.40/28 is a CIDR address

Subnet Mask: 11111111.11111111.11111111.11110000

For Eg. Using CIDR, Find out total no. of

- a) Network bits**
- b) Host bits**
- c) Possible Hosts**
- d) Network/Block ID**

Of following IP 200.10.20.40/28

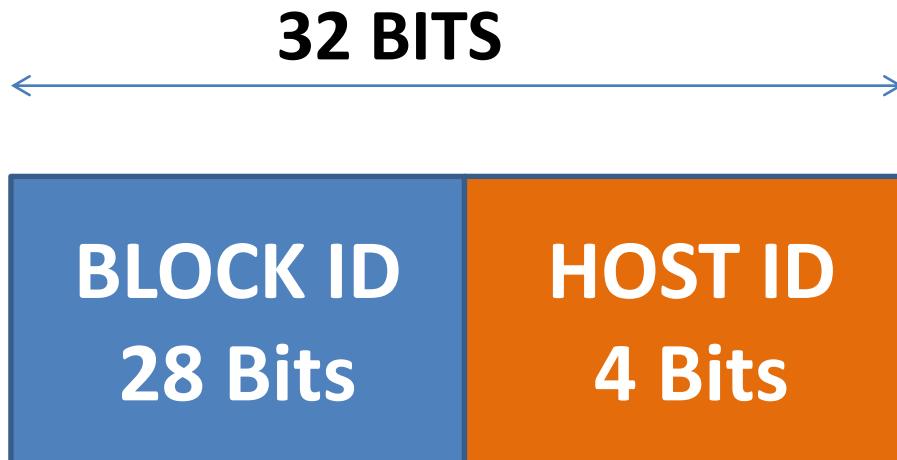
**On comparing given IP 200.10.20.40/28
with Notation x.y.z.w/n**

Here n=28

So total network bits=28 out of 32

So host bits=32 – 28 = 4

Total no. of Possible Hosts= $2^4 = 16$



As network is 28 bits and host is 4 bits

So subnet mask is

11111111.11111111.11111111.11110000

255 . 255 . 255 . 240

Given IP is 200.10.20.40

**On doing AND operation over binary of Last Octet
of given IP with Subnet Mask we get network ID**

240 11110000

40 00101000

32 00100000

So Network id/Block id is 200.110.200.32/28

Rules for CIDR Addressing

- **Address should be contiguous.**
- **Number of addresses in a block must be in power of 2**
- **First address of every block must be evenly divisible with size of blocks/total hosts.**

SUBNETTING IN CIDR

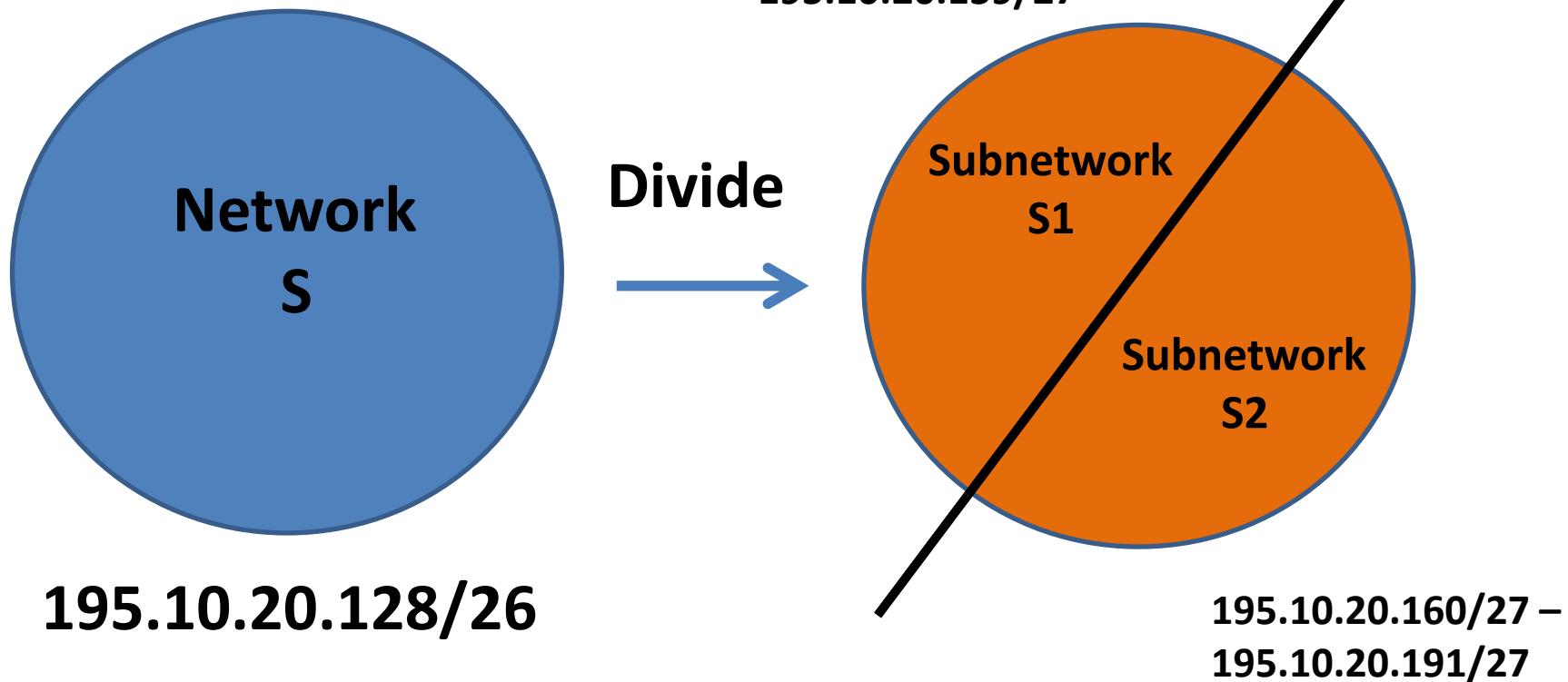
Subnetting in CIDR is same as Classful IP addressing

Bits in host will be used as subnet purpose and to be fixed for a particular network

So number of subnetworks = 2^m

where m= no. of subnet bits

For eg. Network S divides into two subnetworks S1 and S2



Variable Length Subnet Masking(VLSM)

Technique to make subnetworks of different different sizes

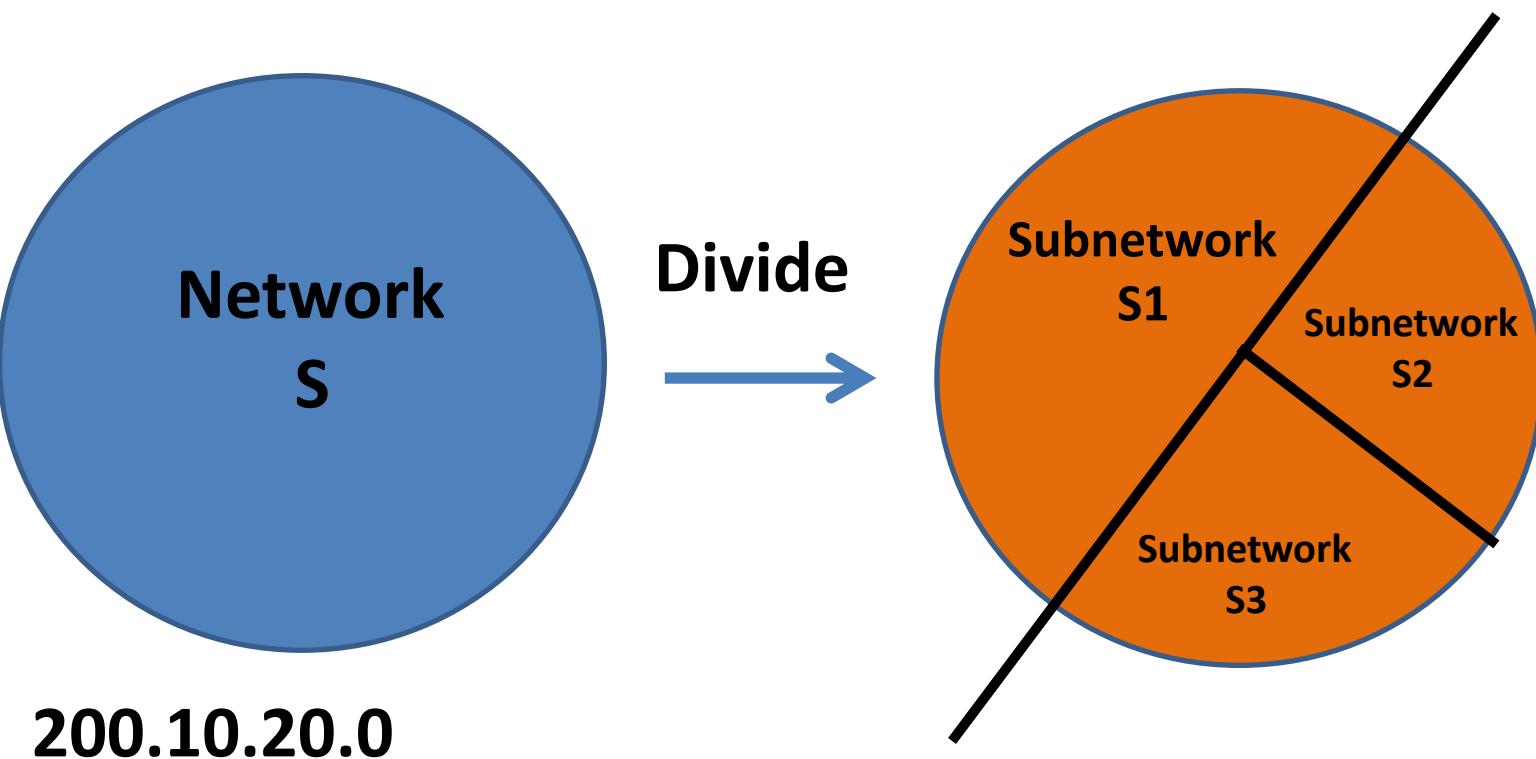
Network allow of different size subnetworks are known as Flexibility

VLSM applied in both Classful and Classless Addressing

Always Remember: Only host bits used for subnetting

VLSM in Classful Addressing

For eg. Network S divides into three subnetworks S1(50%), S2(25%) and S3(25%) of different sizes



200.10.20.00000000

200.10.20.00000000

S1

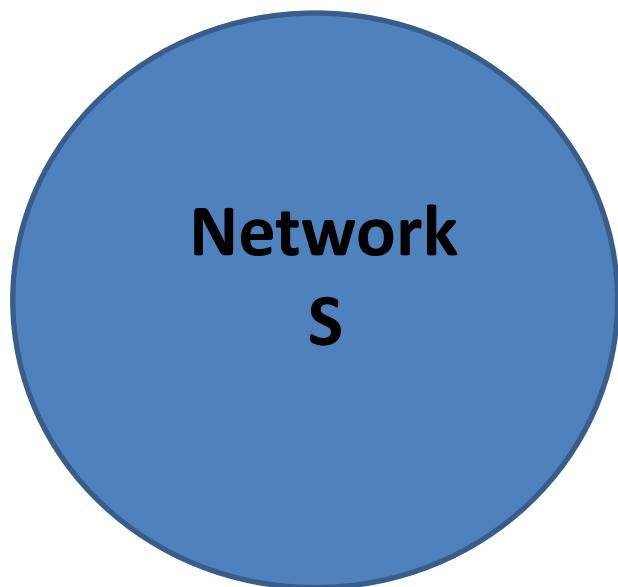
200.10.20.10000000

200.10.20.10000000

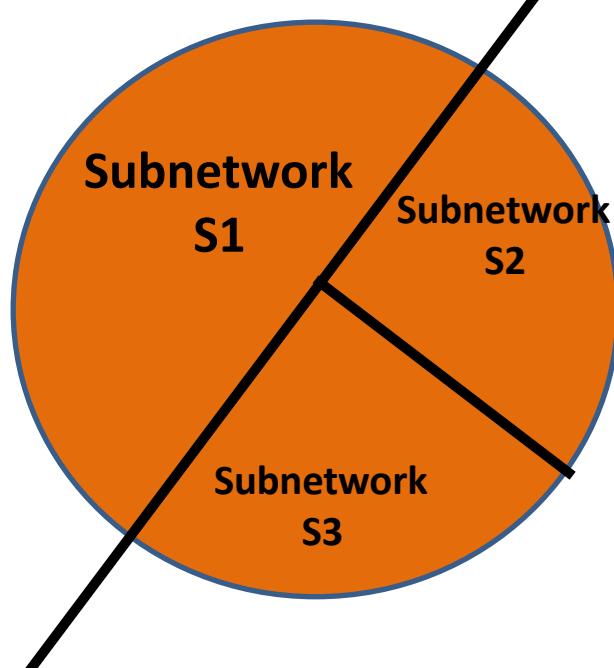
S2

200.10.20.11000000

S3



Divide



Subnet Mask S1

255.255.255.128

Available hosts

$$128-2=126$$

Subnetwork S1

200.10.20.00000000 –
200.10.20.01111111

200.10.20.0 - 200.10.20.127

Subnet Mask S2

255.255.255.192

Available hosts

$$64-2=62$$

Subnetwork S2

200.10.20.10000000 –
200.10.20.10111111

200.10.20.128 –
200.10.20.191

Subnetwork S3

200.10.20.11000000 –
200.10.20.11111111

200.10.20.192 - 200.10.20.255

Total Available hosts

$$256-6=250$$

Available hosts

$$64-2=62$$

Subnet Mask S3

255.255.255.192

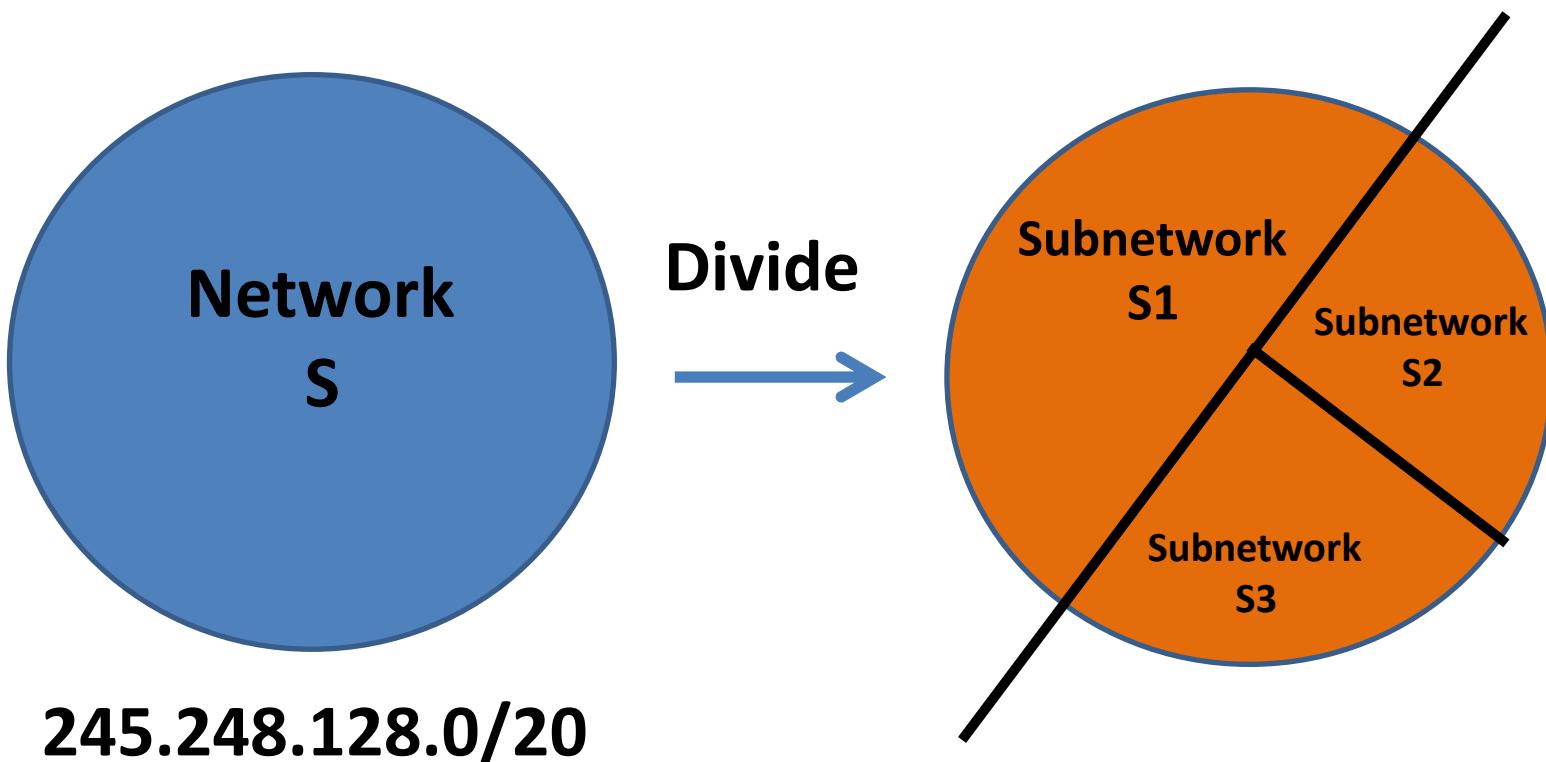
Variable Length Subnet Masking(VLSM)

If n=number of Subnets

Total Usable Hosts=Total Hosts – 2n

VLSM in CIDR

For eg. Network S divides into three subnetworks
S1(50%), S2(25%) and S3(25%) of different sizes



245.248.1000000.00000000

245.248.1000000.00000000

245.248.10001000.00000000

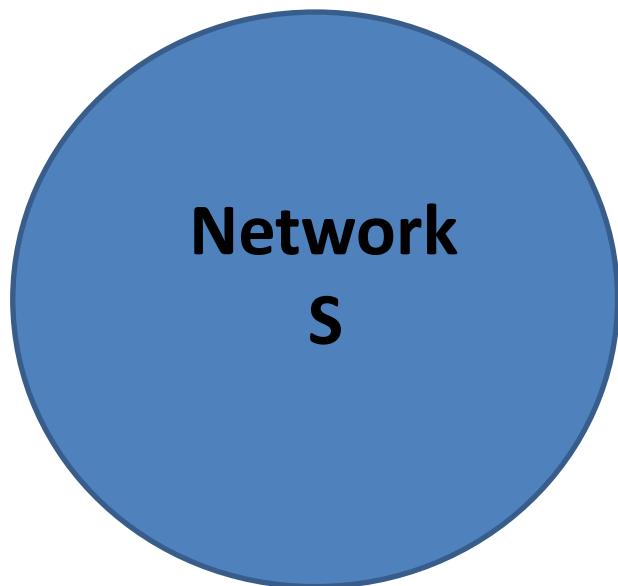
S1

245.248.10001000.00000000

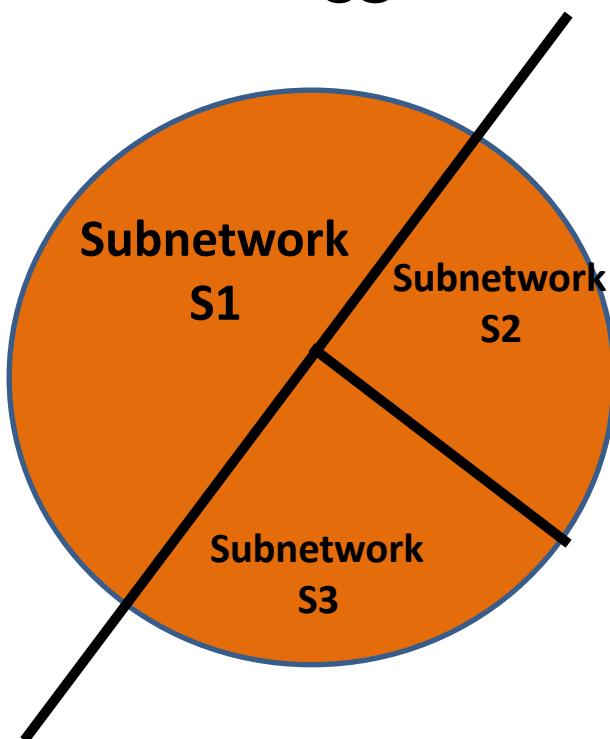
S2

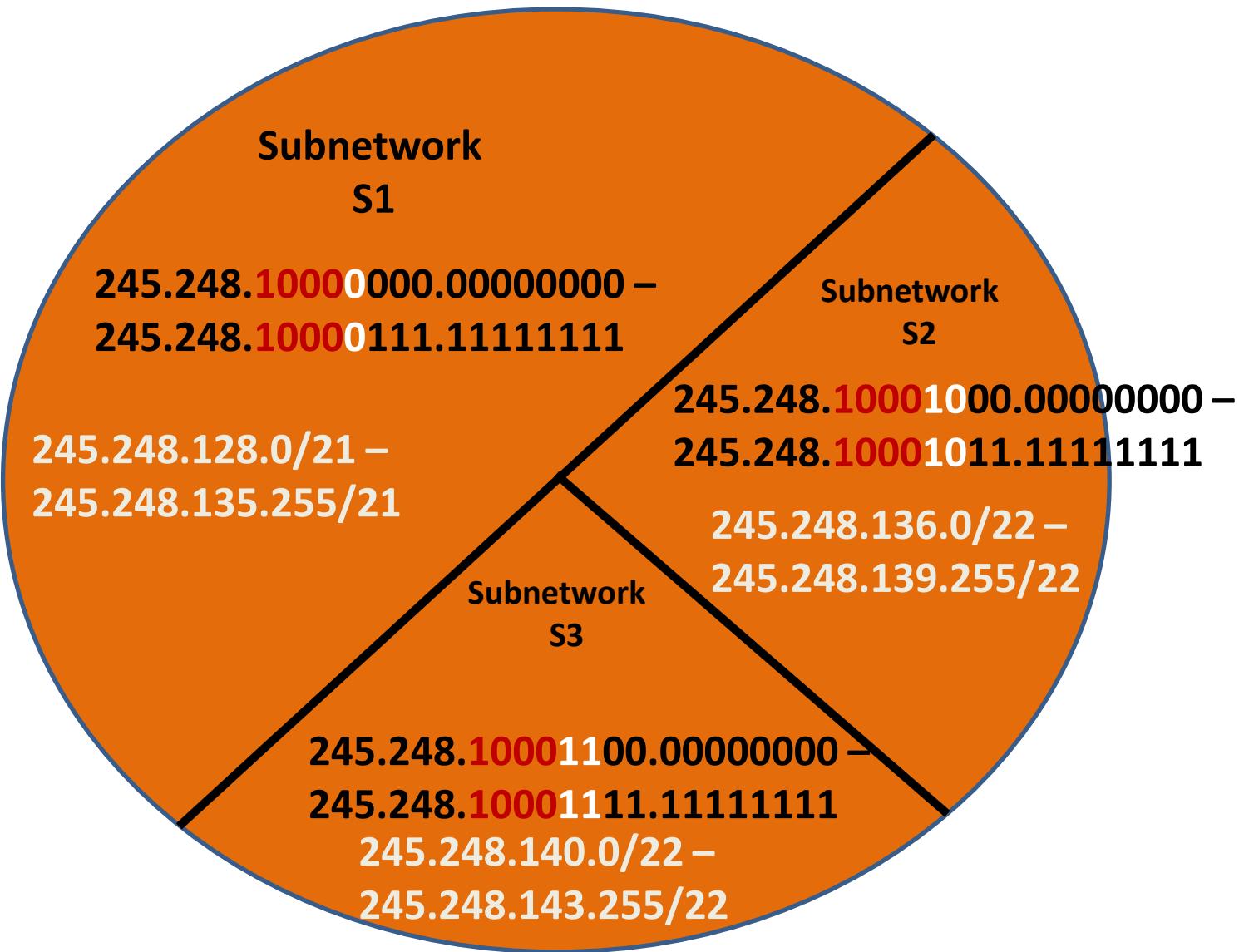
245.248.10001100.00000000

S3



Divide
→





INTERNETWORKING

&

CONNECTING

DEVICES

INTERNETWORKING

- When two or more different networks are connected together to form a bigger network, it is known as internetwork.
- These different network may be based on different technologies and may use different protocols like TCP/IP, SNA, DECnet, NCP/IPX, Apple Talk and other specialized protocols for satellites and cellular networks.
- Beside protocols, there are several other parameters that differentiate network. For example, packet size, flow control etc.

NETWORKING AND INTERNETWORKING DEVICES

1. Hub

Passive Hub

Active Hub

2. Repeater

3. Bridge

4. Router

5. Gateways

6. Switch

Application

Presentation

Session

Transport

Network

Data Link

Physical

Gateway

Router

Bridge

Repeater
or Hub

Application

Presentation

Session

Transport

Network

Data Link

Physical

HUB

➤ **Passive Hub**

A passive hub is just a connector and connects the wires coming from different sides.

A passive hub is just a point where signal coming from different stations collide.

Therefore, passive hub is the collision point.

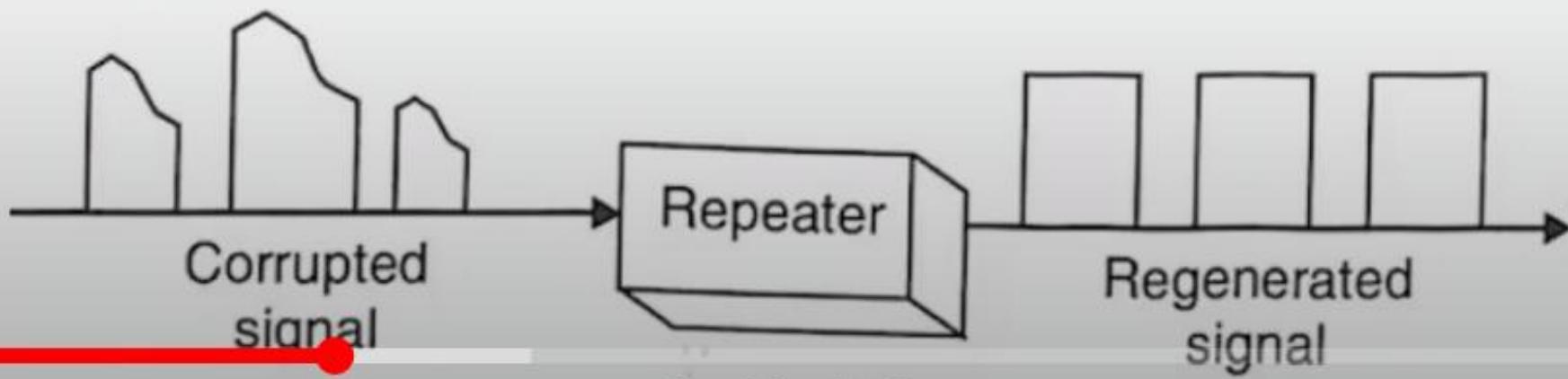
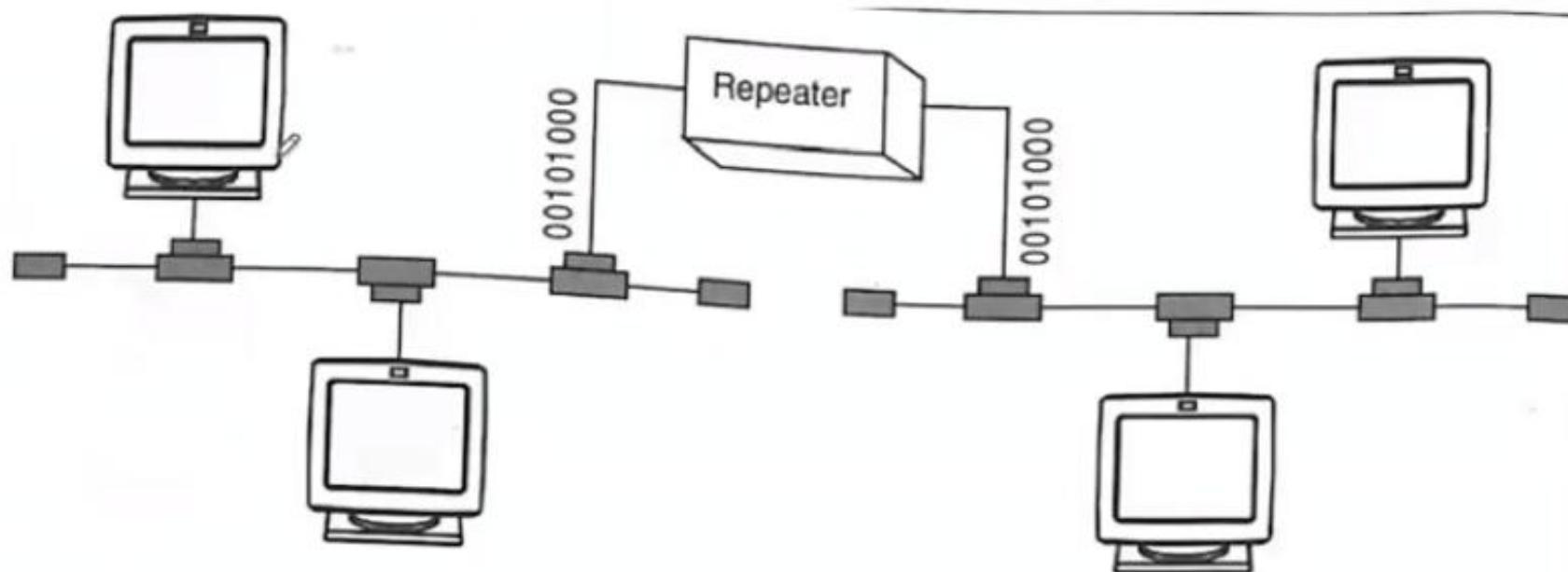
➤ **Active Hub**

An active hub is a multi-port repeater. It is used in physical star topology to create connections between the stations

2. Repeaters

- A repeater is also known as regenerator.
- It is an electronic device that operates on only physical layer of the OSI model.
- Signal can carry the information within a network only for a fixed distance.
- After this distance attenuation occurs and the signal becomes weak or corrupted.
- The purpose of repeater is to take this signal (before it becomes too weak or corrupted) and regenerate the original bit pattern.
- Thus, it places a fresh copy of signal on the link.

2. Repeaters

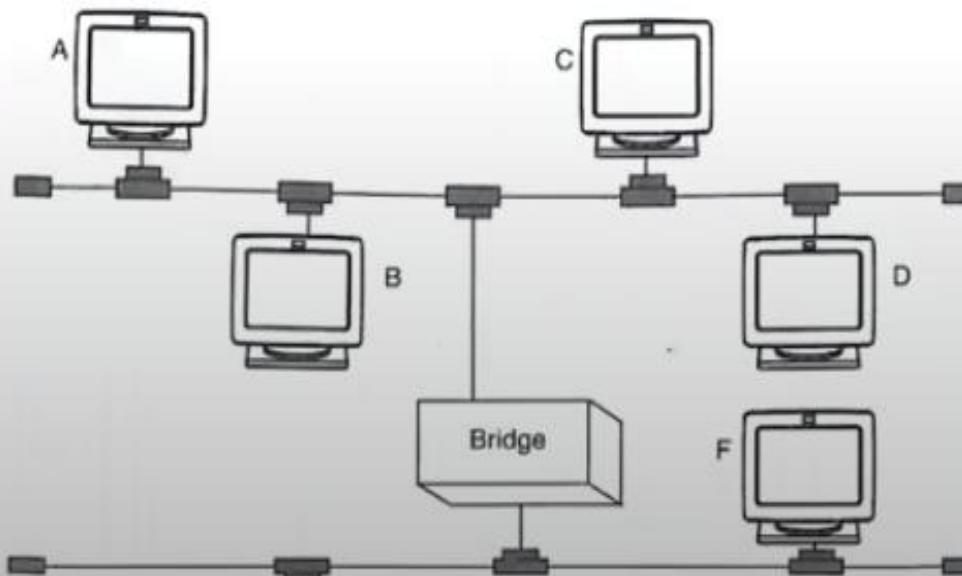


3. BRIDGES

- Bridges can be used to divide a large network into smaller segments and can also pass the frames between two originally separated LANs.
- As a physical layer device, a bridge regenerates the signal.
- As a data link layer device a bridge can access the physical or MAC address of the stations attached to it.
- Unlike repeater, a bridge performs filtering also.
- It has a logic that allows it to keep the traffic for each segment separate.
- It can check the destination address of a frame and decide if the frame should be forwarded or dropped.

3. BRIDGES

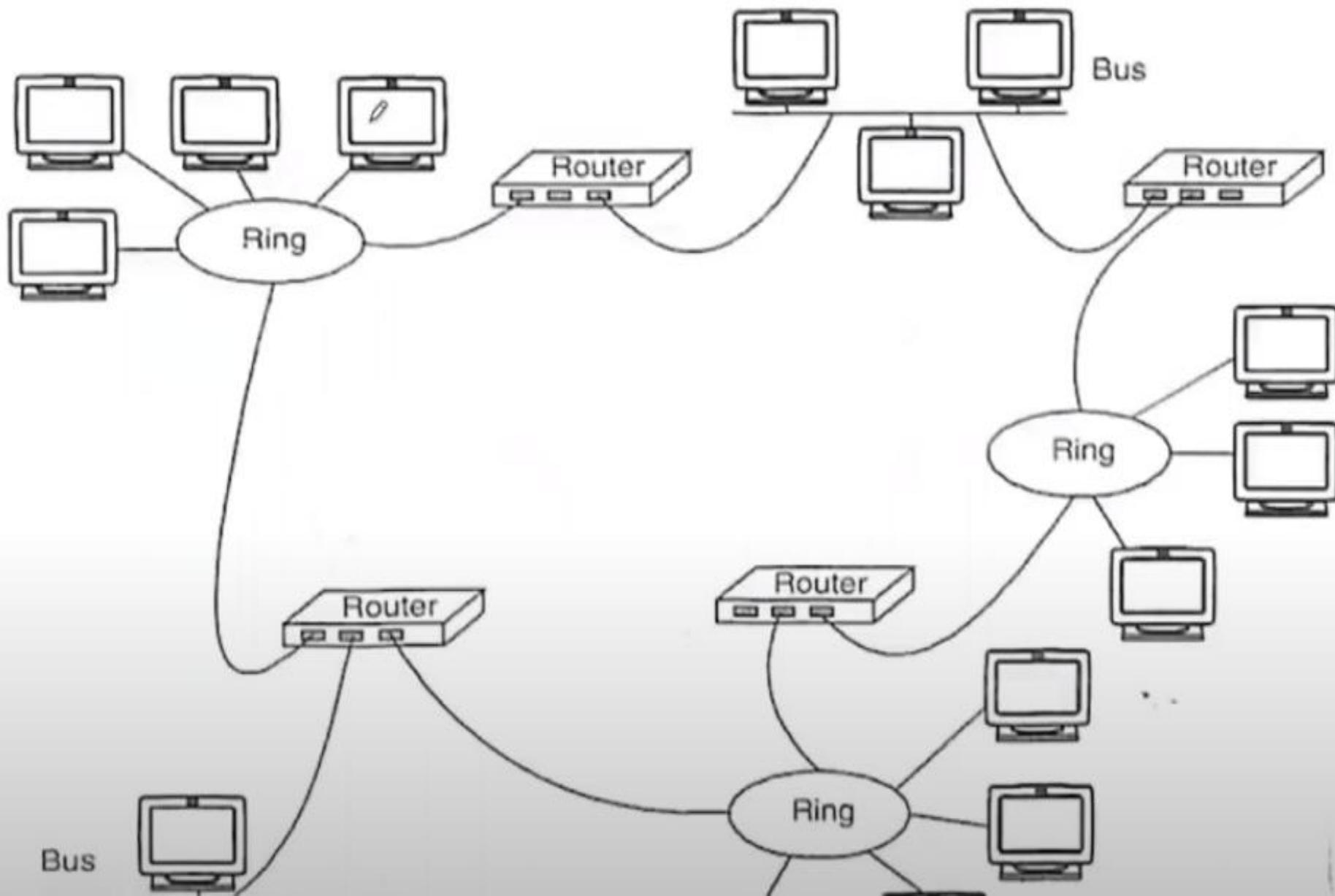
- A bridge maintains a table of MAC address of various stations attached to it.
- When a frame enters a bridge, it checks the address contained in the frame and compares this address with a table of all the stations on both segments.
- When it finds a match, it decides to which segment the destination station belongs and then passes the frame to that segment



4. ROUTERS

- Routers are multiport devices and more sophisticated as compared to repeaters and bridges.
- Routers also support filtering and encapsulation like bridges.
- They operate at physical, data link and network layer of OSI model
- Like bridges, they are self-learning, as they can communicate their existence to other devices and can learn of the existence of new routers, nodes and LAN segments.
- A router may connect LANs and WANs in the Internet and can pass or relay the packets among them. A router has access to the network layer address or logical address (IP address).

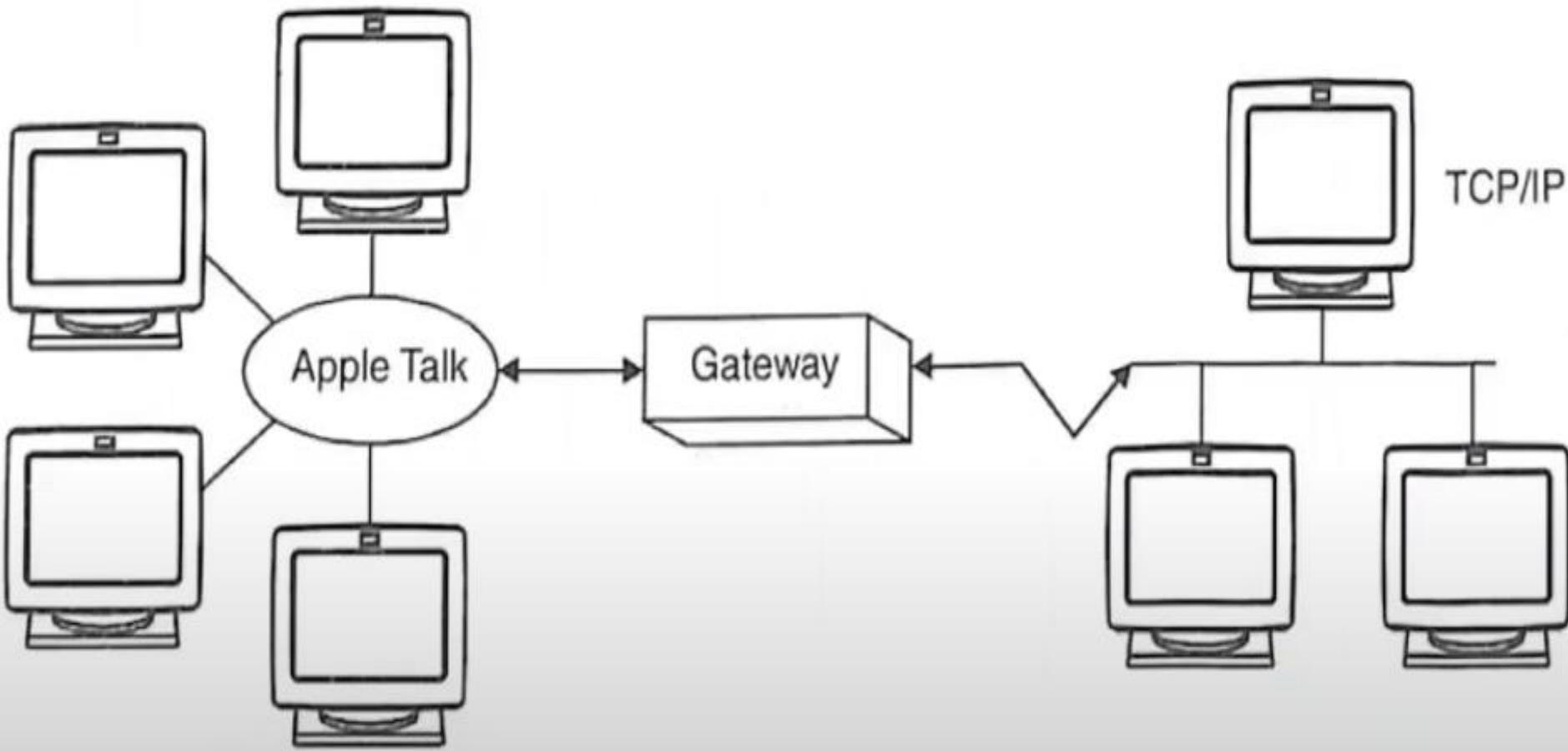
4. ROUTERS



5. GATEWAY

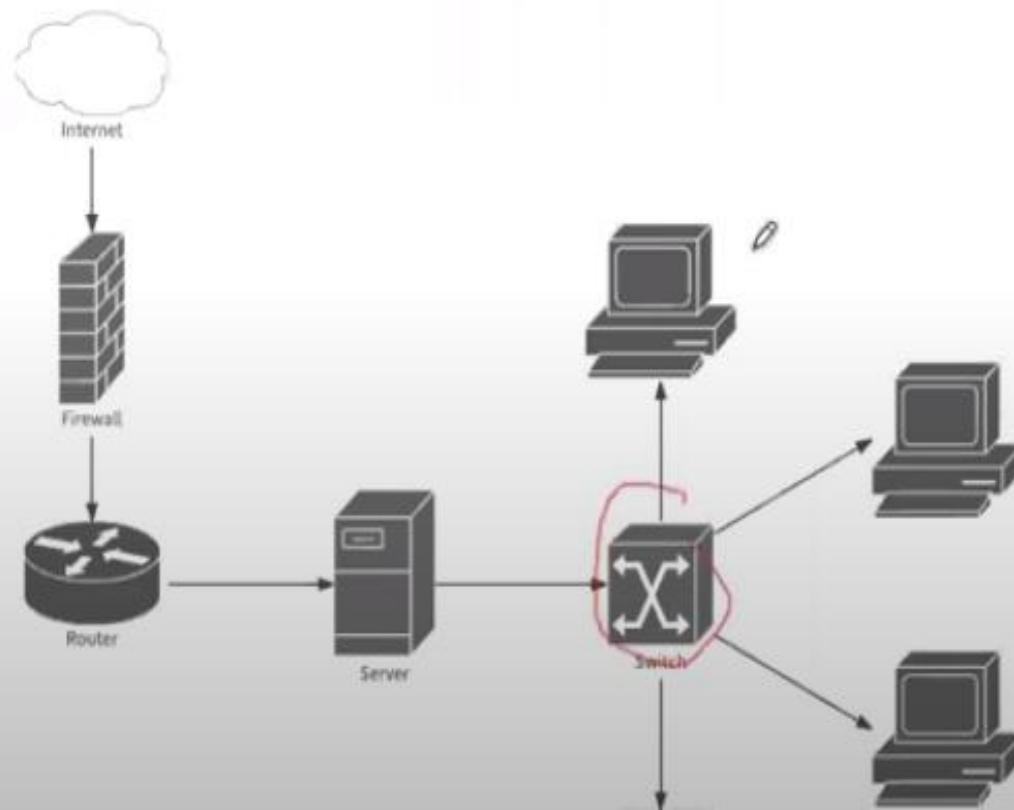
- Gateways are the internetworking devices that operate in all the seven layers of OSI model
- A gateway act as a protocol converter. It is used to connect two different type of network that uses different protocols.
- A gateway can accept a packet formatted for one protocol (e.g. Apple Talk) and convert it to a packet formatted for another protocol (e.g. TCP/IP) before forwarding it
- In addition, gateways can perform all the functions of bridges and routers.
- The gateway links two systems that do not use same communication protocols, data formatting structures, languages and architecture.

5. GATEWAY

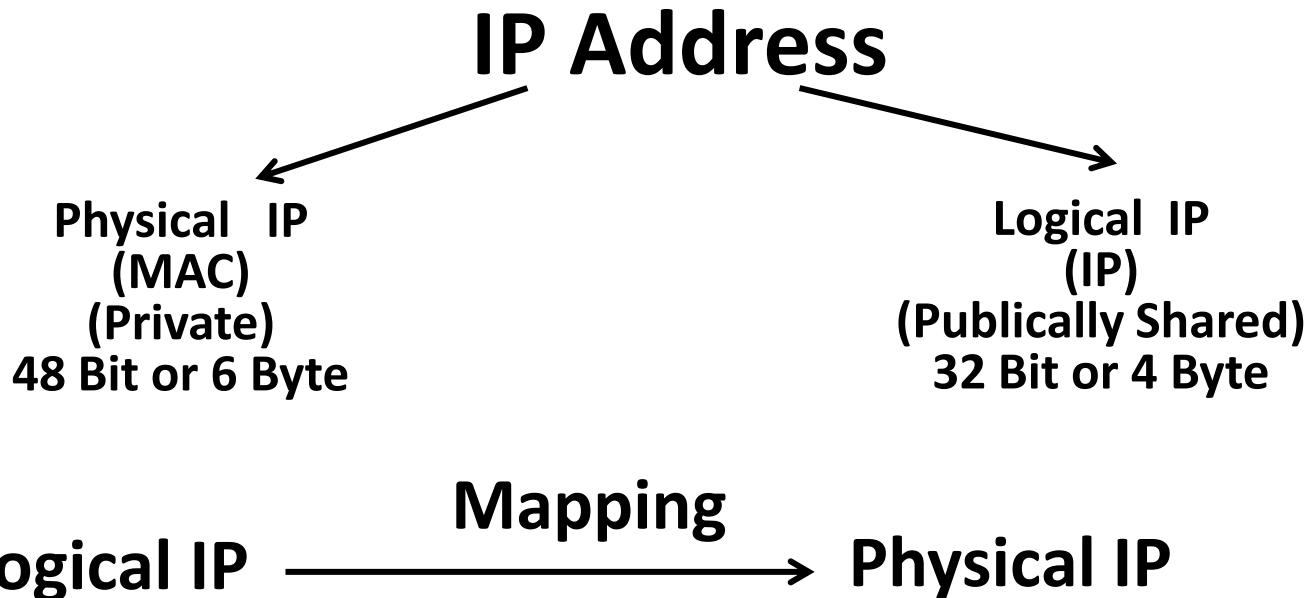


6. NETWORK SWITCH

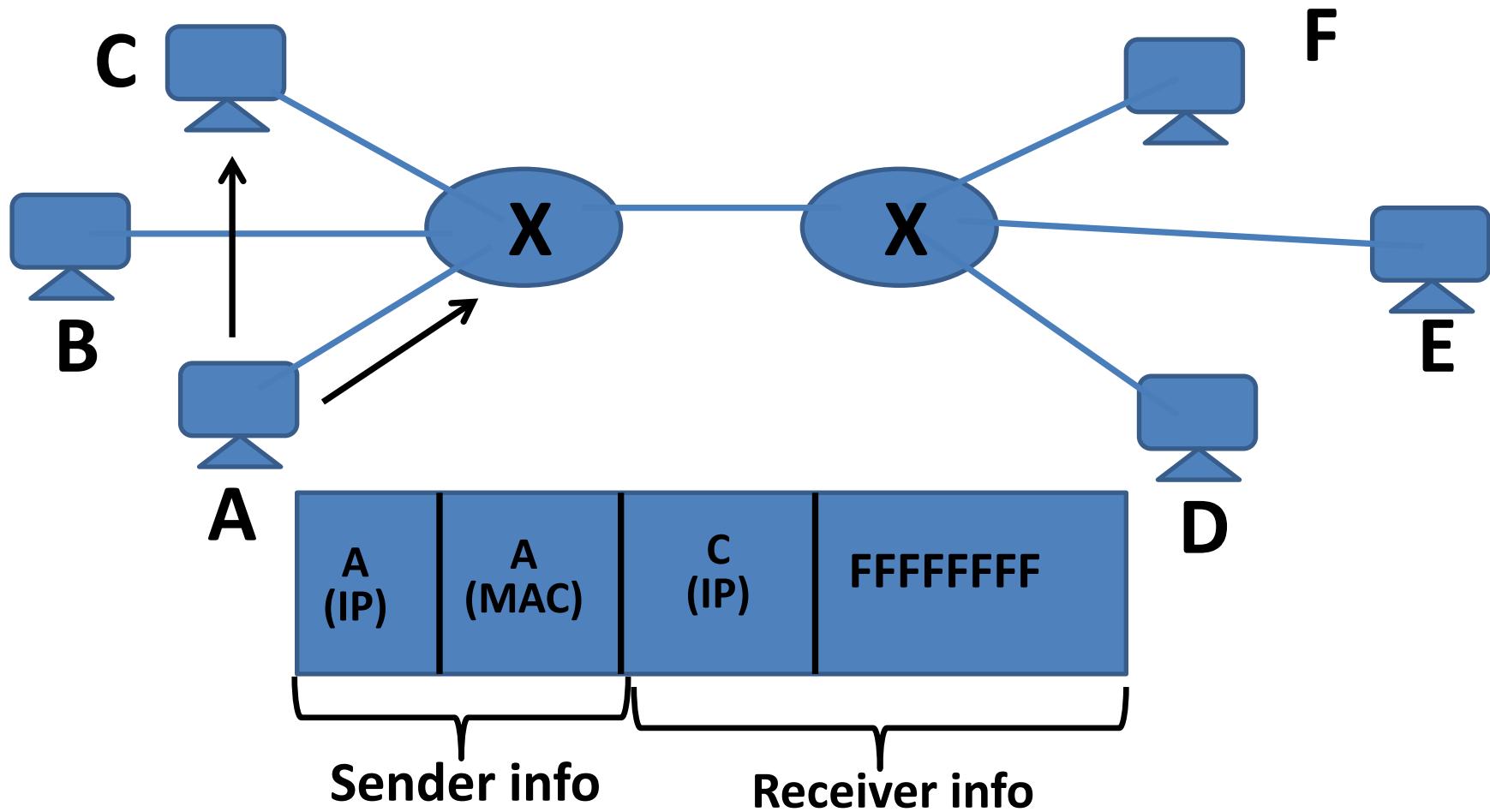
- Switches are used to connect multiple devices together on the same network.
- In a properly designed network, LAN switches are responsible for directing and controlling the data flow at the access layer to networked resources.

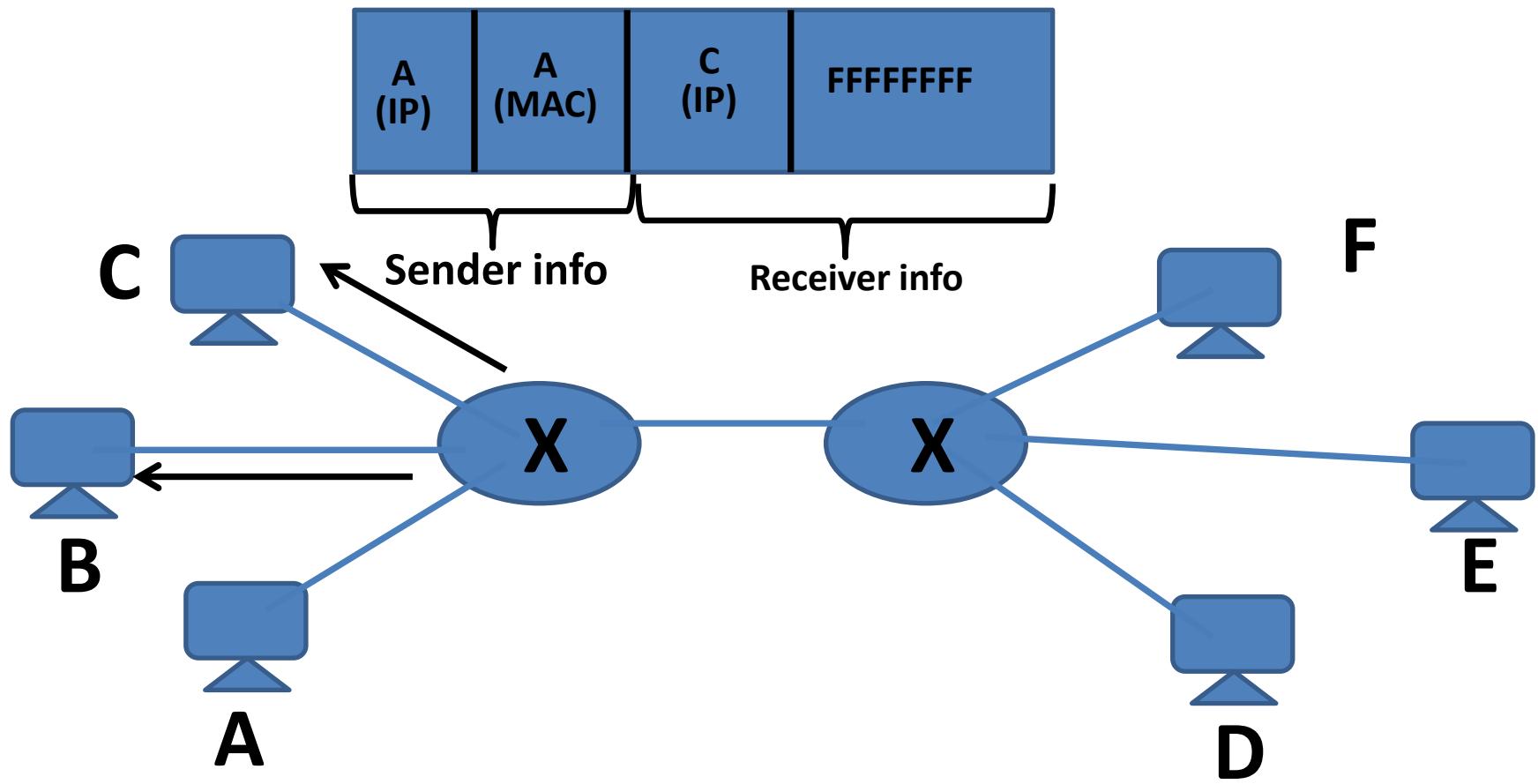


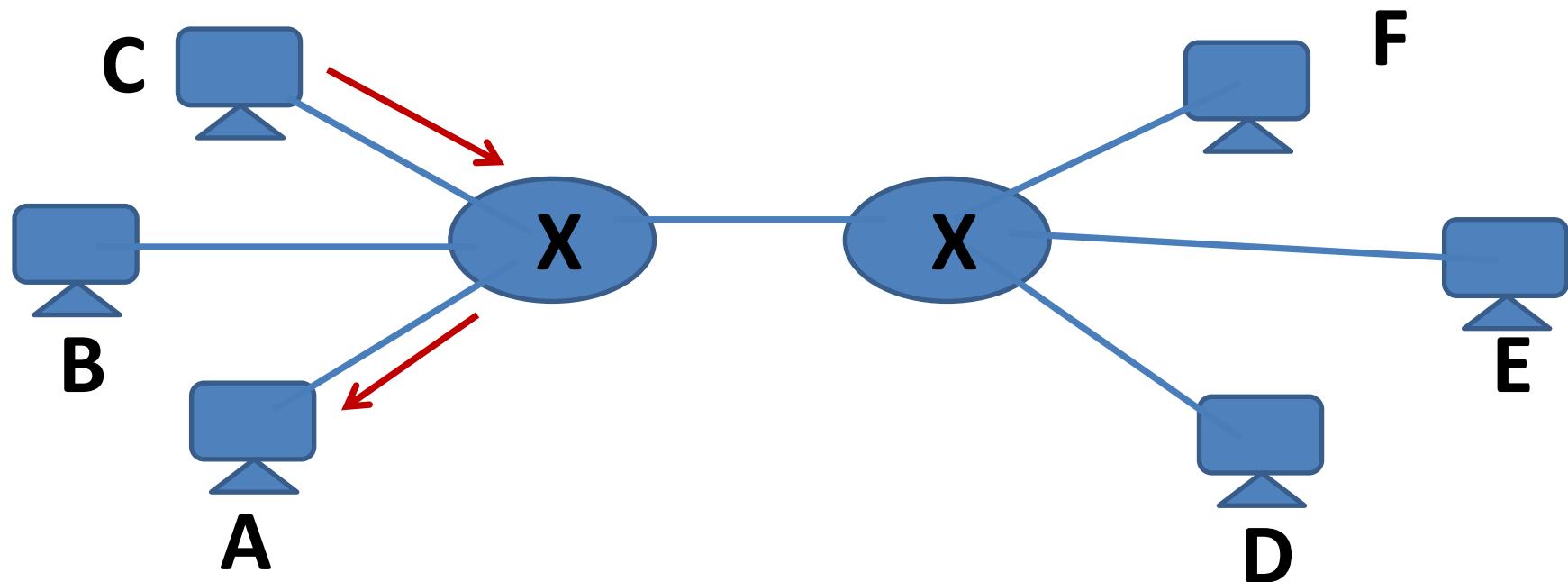
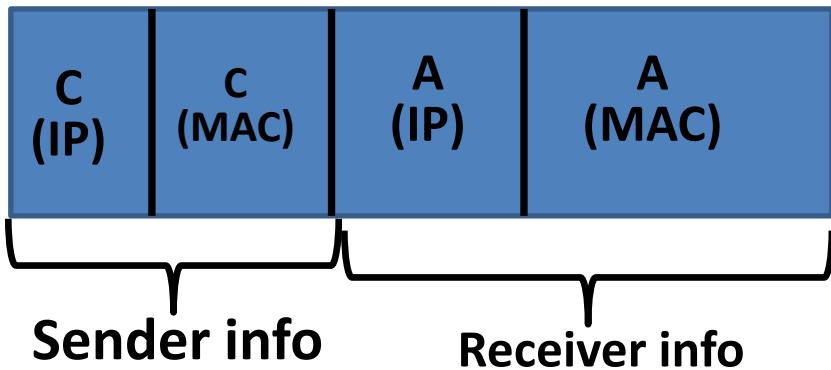
ADDRESS MAPPING



**BroadCasting to establish connection
for data transfer
HANDSHAKING**

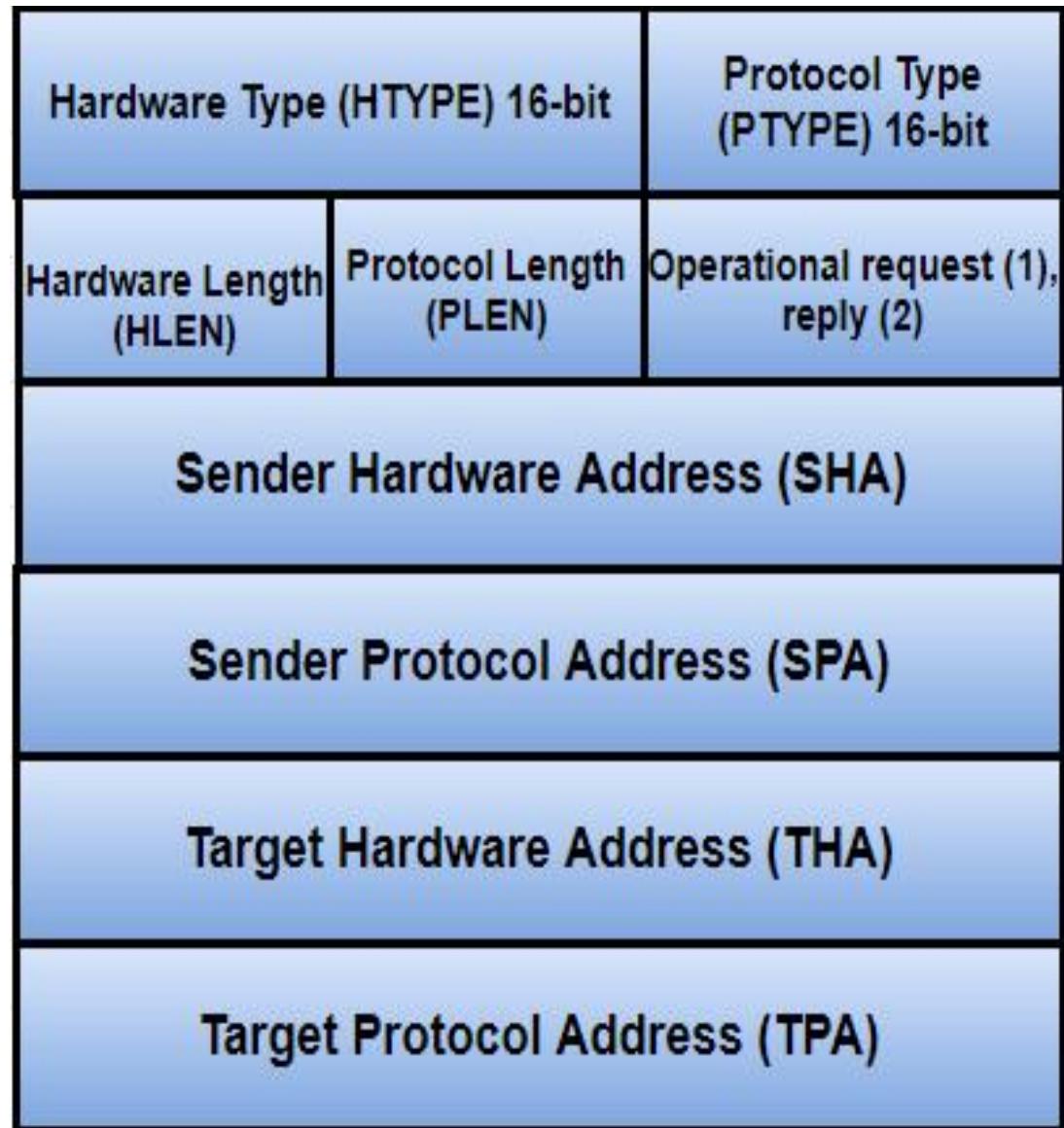






Address Mapping

- Host to Host
(H to H)
- Host to Router
(H to R)
- Router to Host
(R to H)
- Router to Router
(R to R)



FORWARDING

&

DELIVERY

DELIVERY

The network layer supervises the handling of the packets by the underlying physical networks. We define this handling as the delivery of a packet.

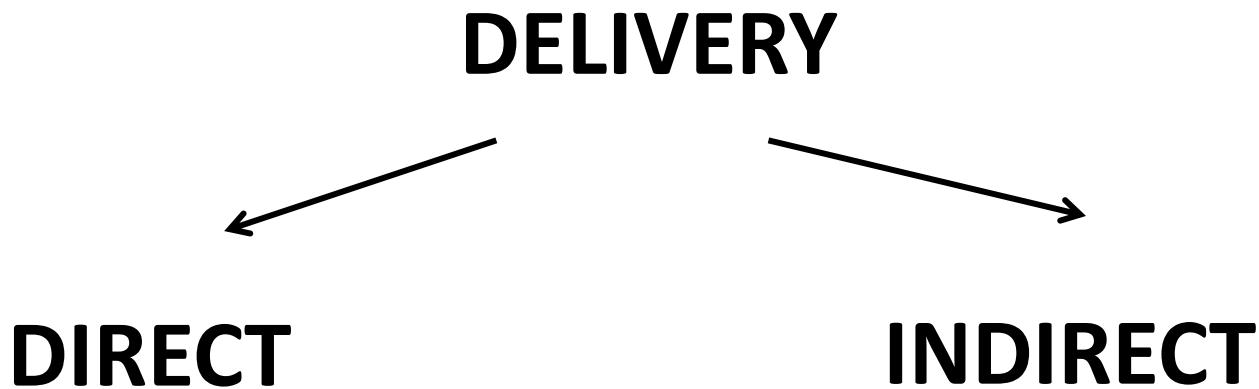
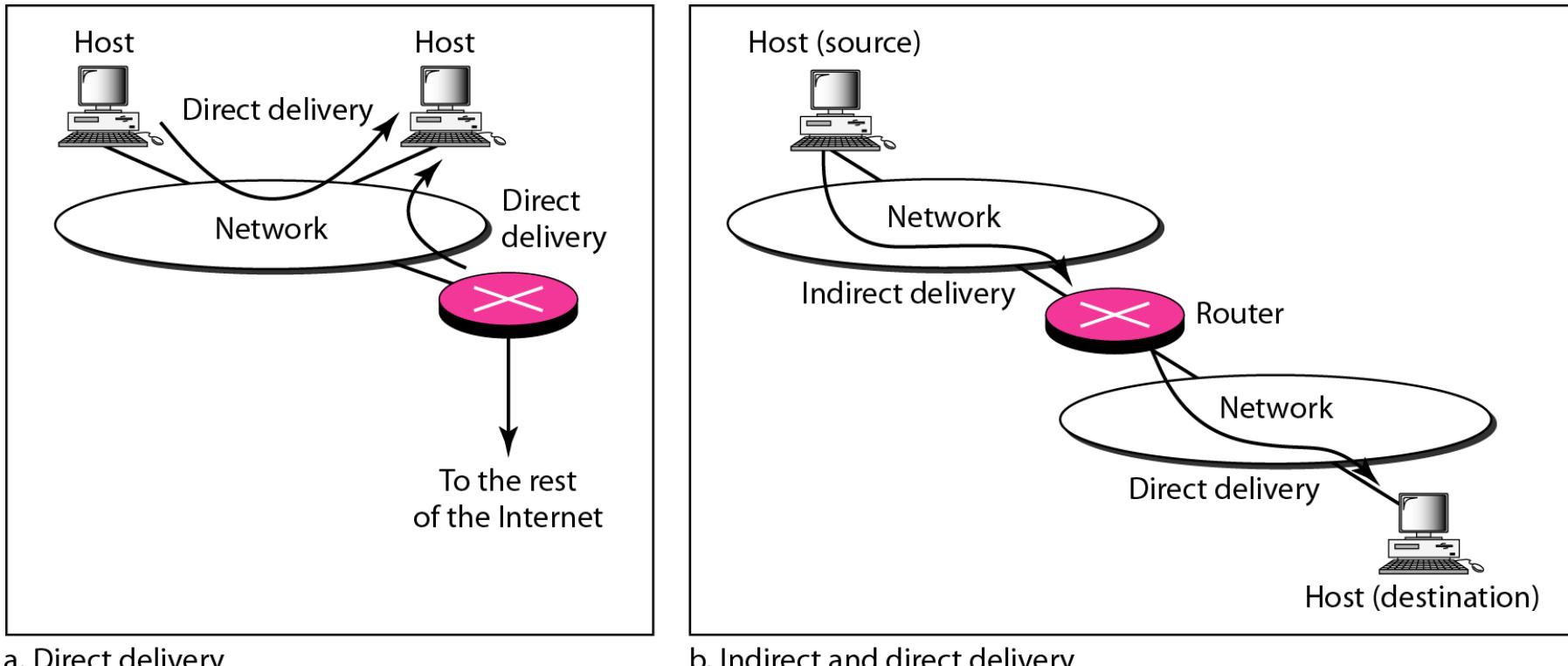


Figure 22.1 Direct and indirect delivery



FORWARDING

Forwarding means to place the packet in its route to its destination. Forwarding requires a host or a router to have a routing table.

When a host has a packet to send or when a router has received a packet to be forwarded, it looks at this table to find the route to the final destination.

Figure 22.2 Route method versus next-hop method

a. Routing tables based on route

Destination	Route
Host B	R1, R2, host B

Routing table
for host A

Destination	Route
Host B	R2, host B

Routing table
for R1

Destination	Route
Host B	Host B

Routing table
for R2

b. Routing tables based on next hop

Destination	Next hop
Host B	R1

Destination	Next hop
Host B	R2

Destination	Next hop
Host B	---

Host A



Network

R1

Host B



Network

R2

Figure 22.3 Host-specific versus network-specific method

Routing table for host S based
on host-specific method

Destination	Next hop
A	R1
B	R1
C	R1
D	R1

Routing table for host S based
on network-specific method

Destination	Next hop
N2	R1

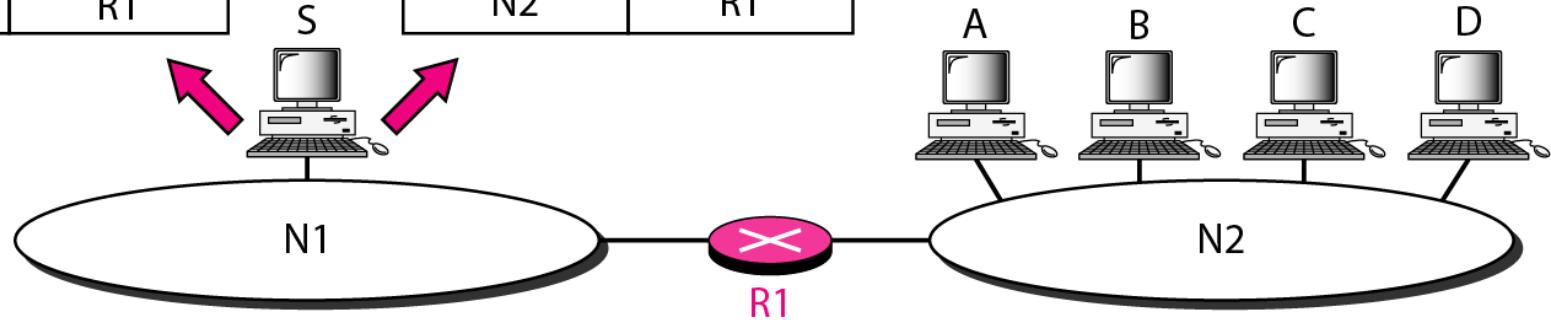


Figure 22.4 *Default method*

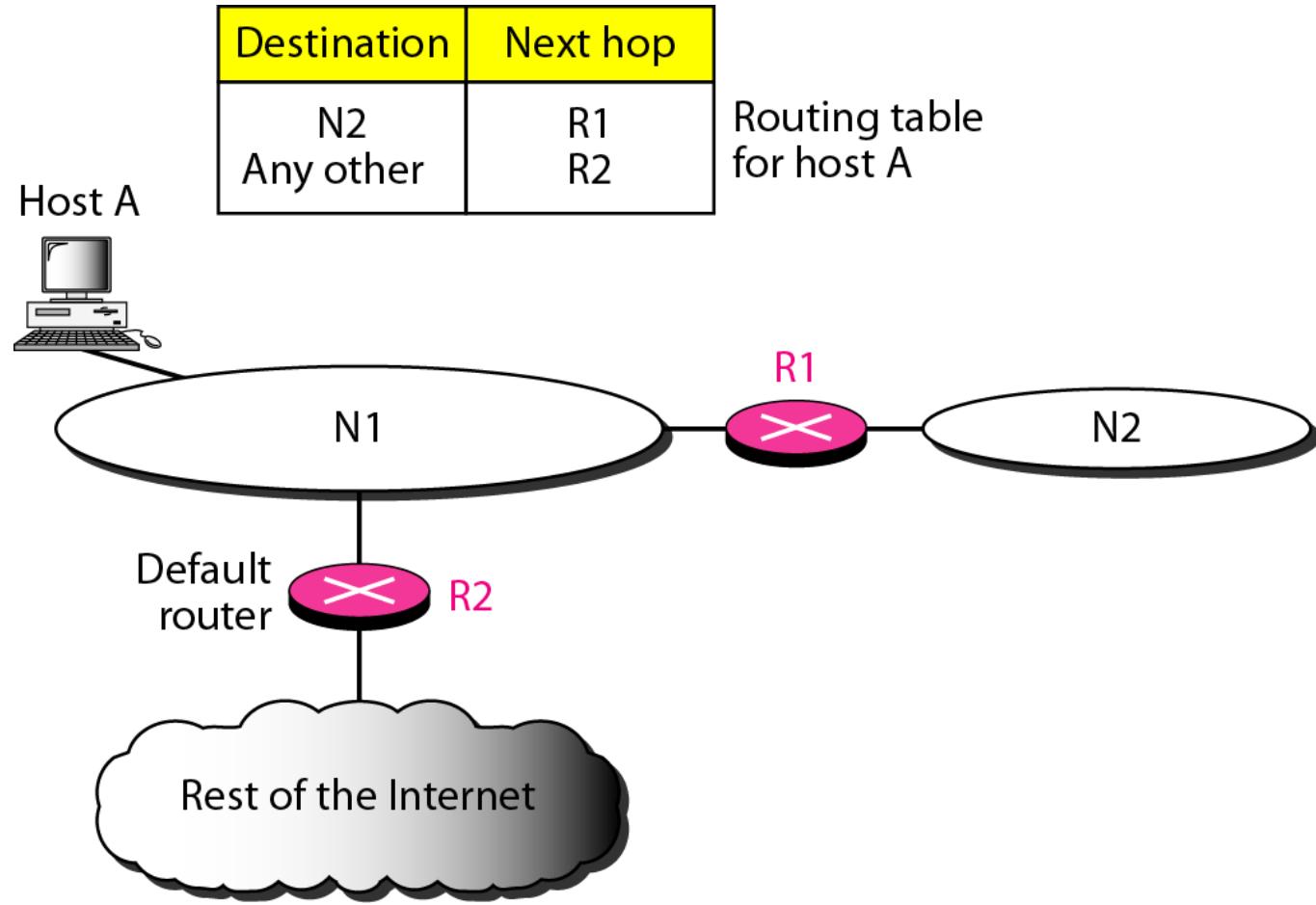
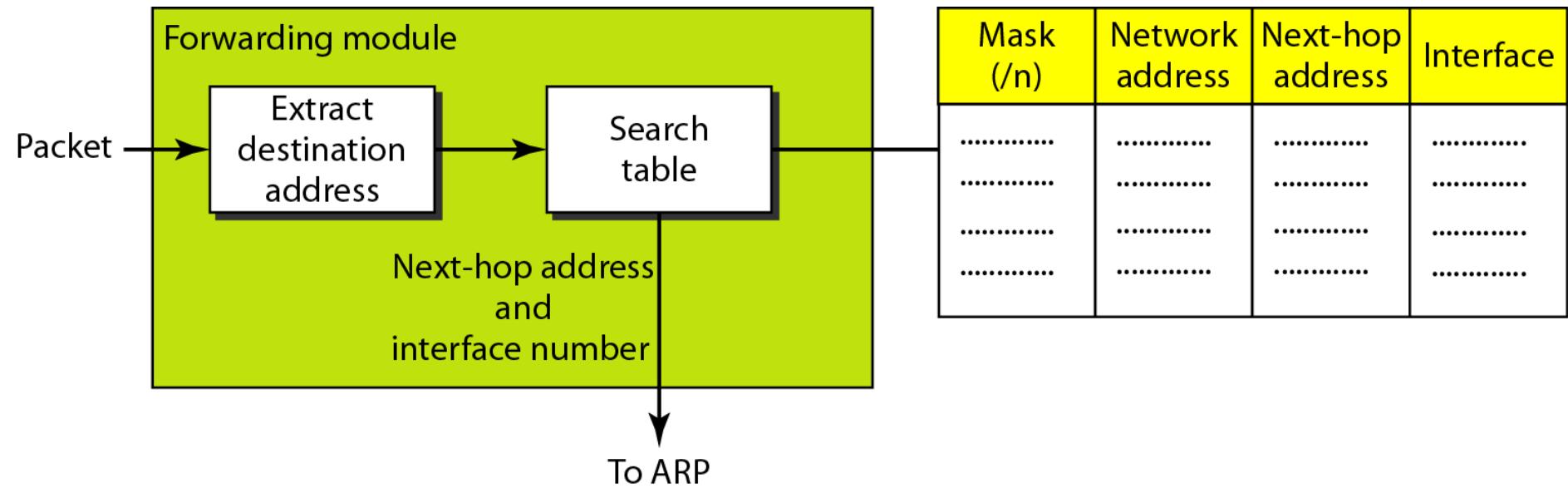


Figure 22.5 Simplified forwarding module in classless address



Example 22.1

Make a routing table for router R1, using the configuration in Figure 22.6.

Solution

Table 22.1 shows the corresponding table.

Figure 22.6 Configuration for Example 22.1

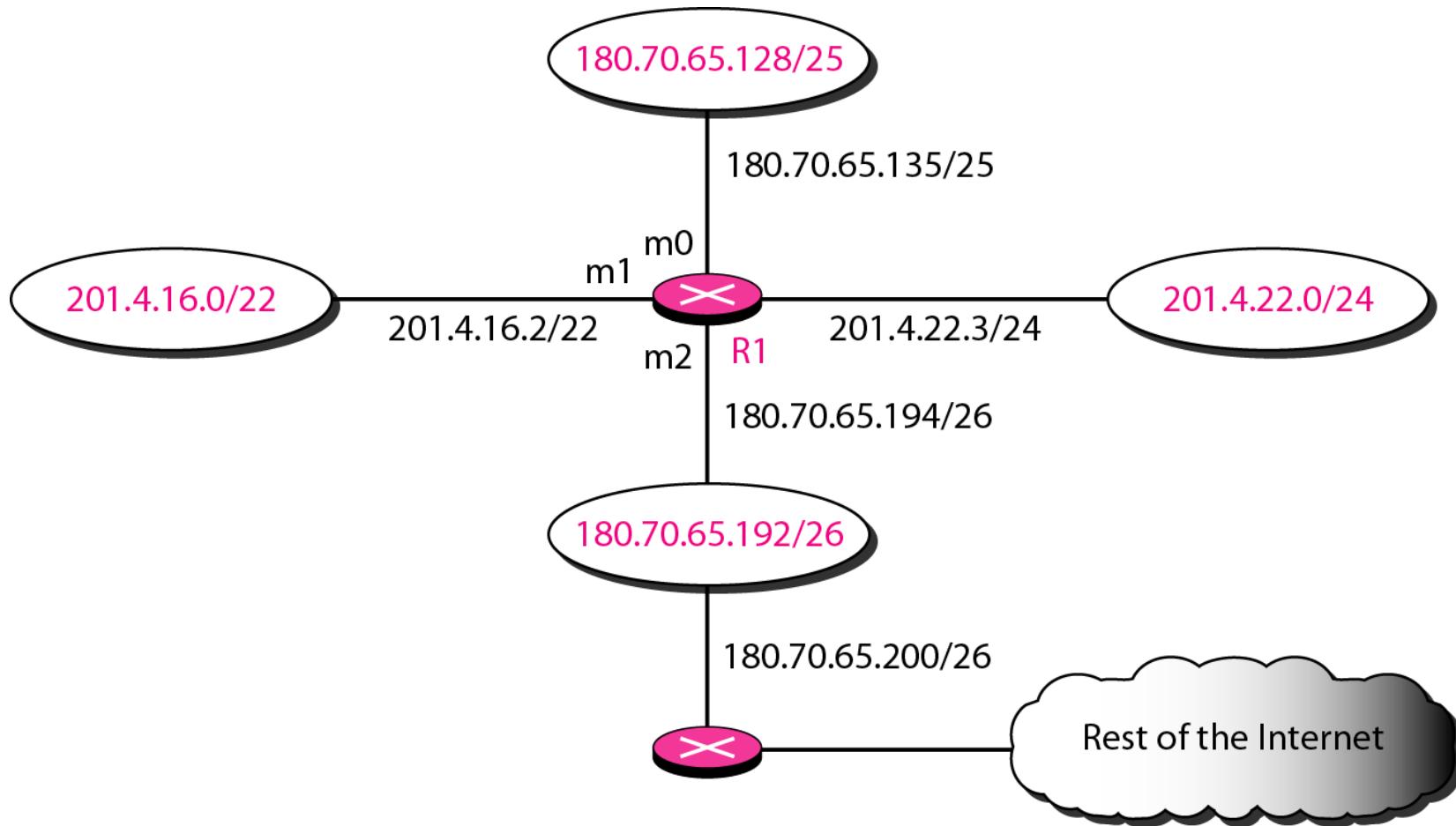


Table 22.1 *Routing table for router R1 in Figure 22.6*

<i>Mask</i>	<i>Network Address</i>	<i>Next Hop</i>	<i>Interface</i>
/26	180.70.65.192	—	m2
/25	180.70.65.128	—	m0
/24	201.4.22.0	—	m3
/22	201.4.16.0	m1
Any	Any	180.70.65.200	m2

Figure 20.5 IPv4 datagram format

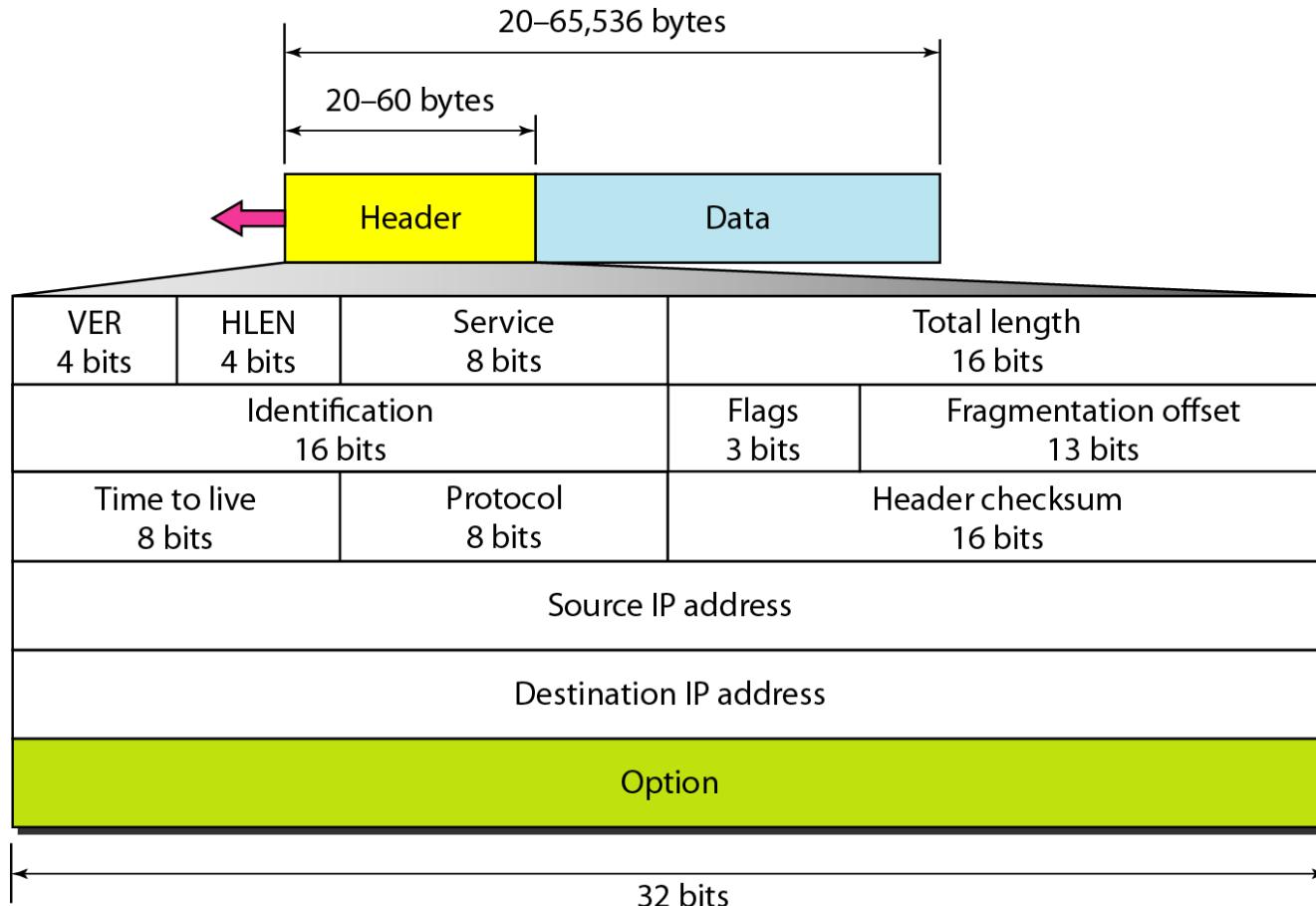
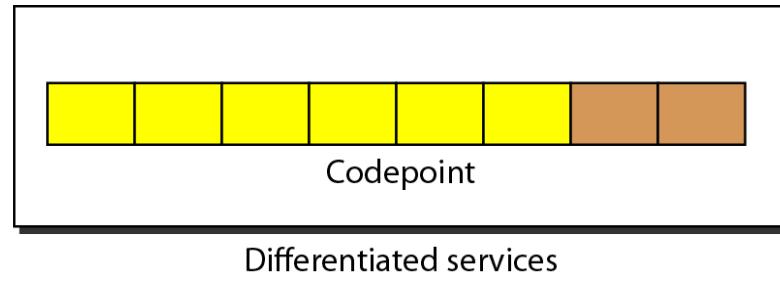
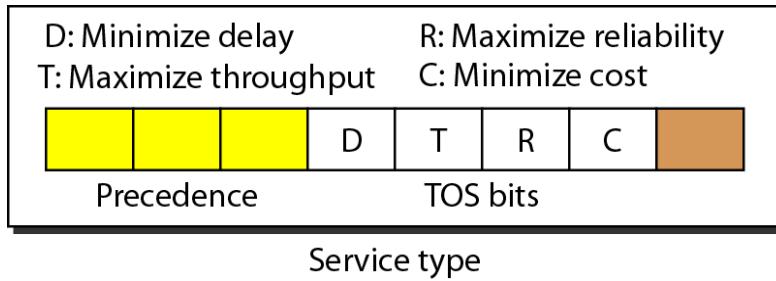
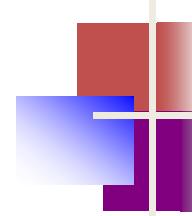


Figure 20.6 *Service type or differentiated services*





Note

The precedence subfield was part of version 4, but never used.

Table 20.1 *Types of service*

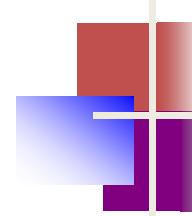
<i>TOS Bits</i>	<i>Description</i>
0000	Normal (default)
0001	Minimize cost
0010	Maximize reliability
0100	Maximize throughput
1000	Minimize delay

Table 20.2 *Default types of service*

<i>Protocol</i>	<i>TOS Bits</i>	<i>Description</i>
ICMP	0000	Normal
BOOTP	0000	Normal
NNTP	0001	Minimize cost
IGP	0010	Maximize reliability
SNMP	0010	Maximize reliability
TELNET	1000	Minimize delay
FTP (data)	0100	Maximize throughput
FTP (control)	1000	Minimize delay
TFTP	1000	Minimize delay
SMTP (command)	1000	Minimize delay
SMTP (data)	0100	Maximize throughput
DNS (UDP query)	1000	Minimize delay
DNS (TCP query)	0000	Normal
DNS (zone)	0100	Maximize throughput

Table 20.3 *Values for codepoints*

<i>Value</i>	<i>Protocol</i>
1	ICMP
2	IGMP
6	TCP
17	UDP
89	OSPF



Note

The total length field defines the total length of the datagram including the header.

Figure 20.7 *Encapsulation of a small datagram in an Ethernet frame*

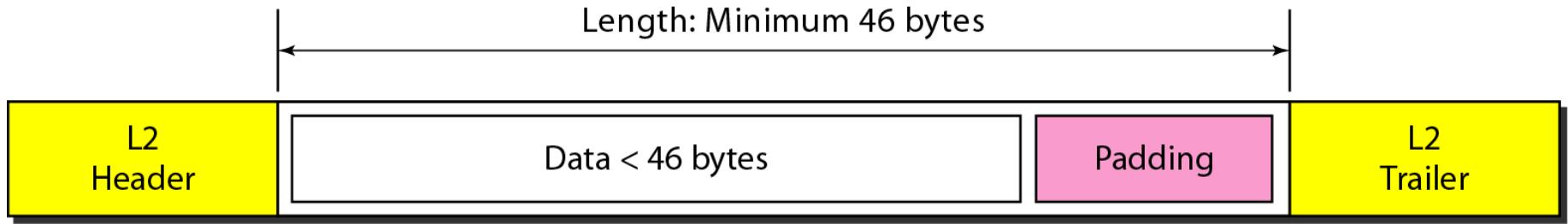


Figure 20.8 *Protocol field and encapsulated data*

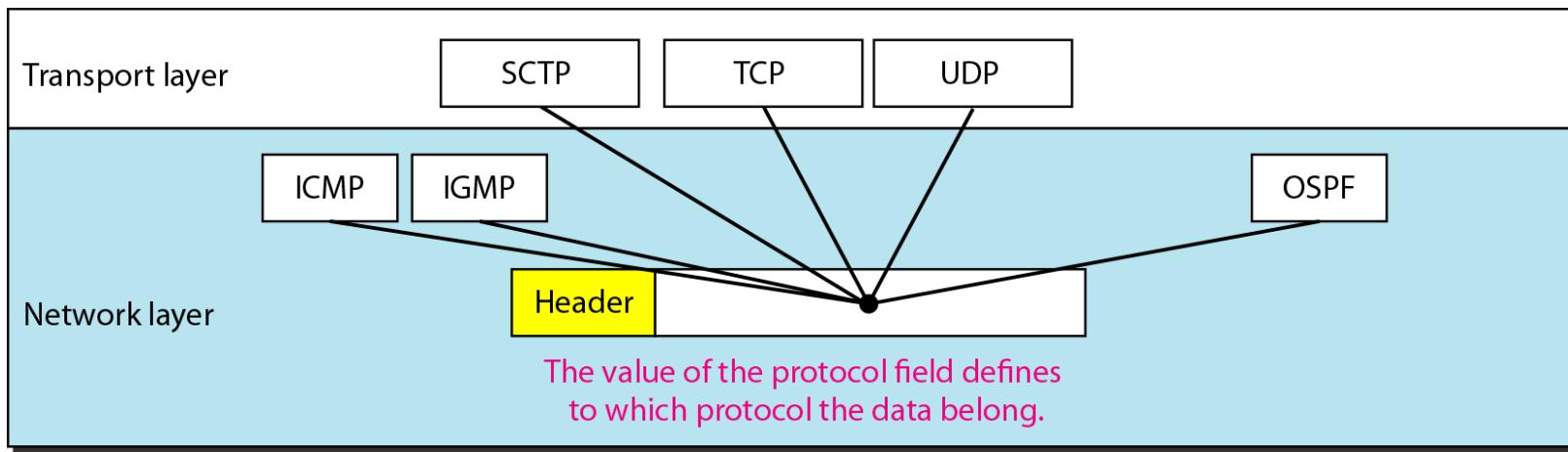


Table 20.4 *Protocol values*

<i>Value</i>	<i>Protocol</i>
1	ICMP
2	IGMP
6	TCP
17	UDP
89	OSPF

Example 20.1

An IPv4 packet has arrived with the first 8 bits as shown:

01000010

The receiver discards the packet. Why?

Solution

There is an error in this packet. The 4 leftmost bits (0100) show the version, which is correct. The next 4 bits (0010) 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 20.2

In an IPv4 packet, the value of HLEN is 1000 in binary. 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 20.3

In an IPv4 packet, the value of HLEN is 5, and the value of the total length field is 0x0028. 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 40 bytes, which means the packet is carrying 20 bytes of data ($40 - 20$).

Example 20.4

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

0x45000028000100000102 . . .

How many hops can this packet travel before being dropped? The data belong to what upper-layer protocol?

Solution

To find the time-to-live field, we skip 8 bytes. The time-to-live field is the ninth byte, which is 01. This means the packet can travel only one hop. The protocol field is the next byte (02), which means that the upper-layer protocol is IGMP.

Figure 20.9 *Maximum transfer unit (MTU)*

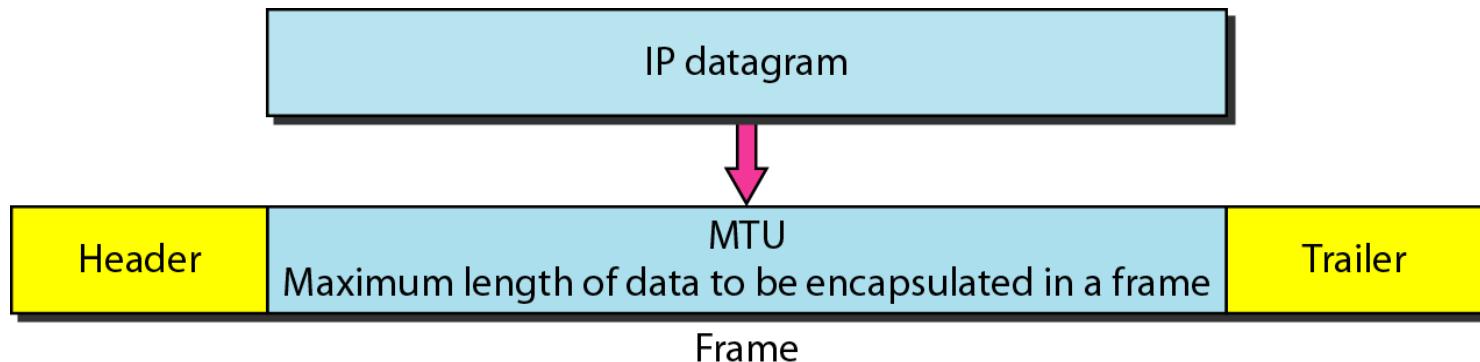
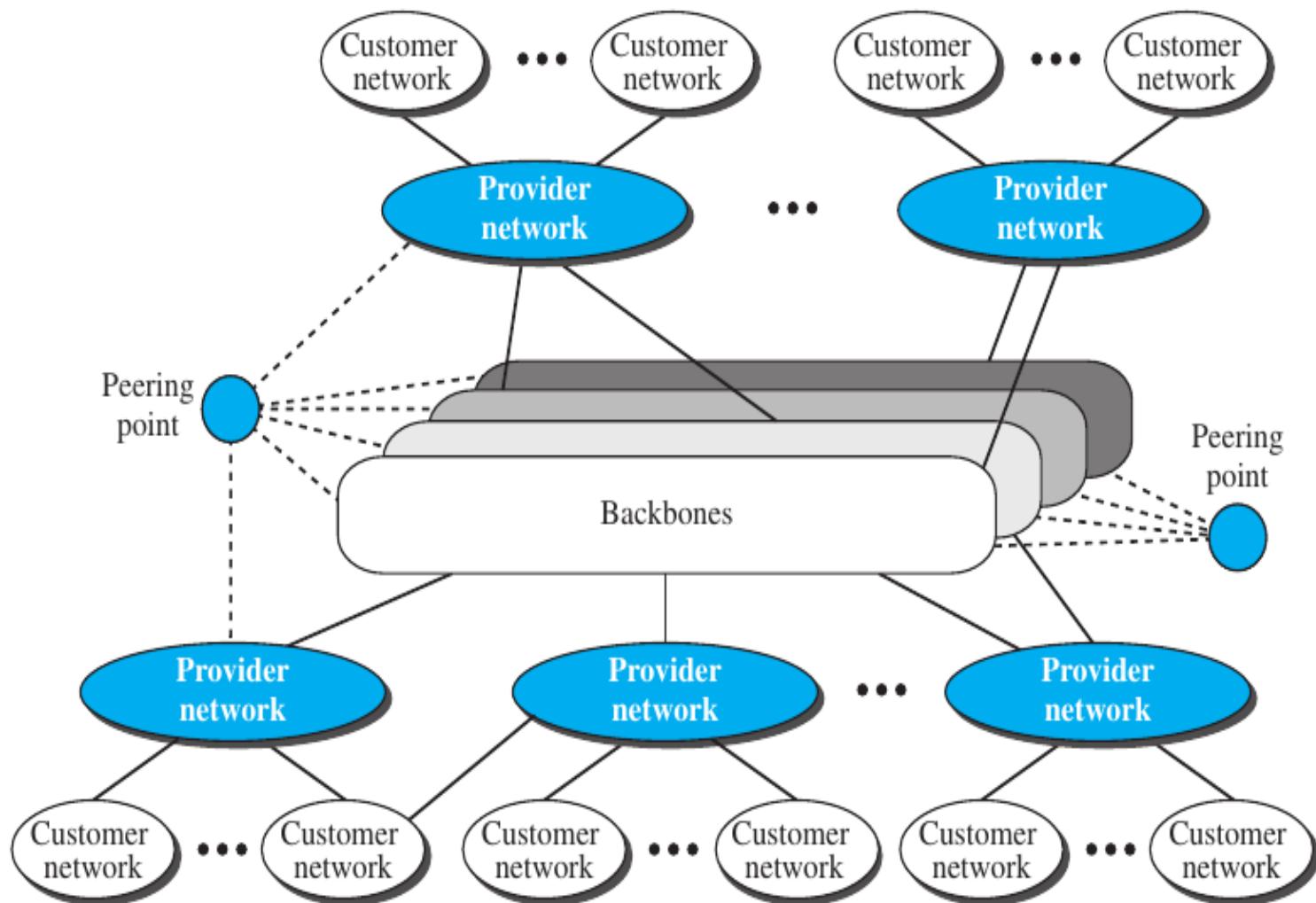


Table 20.5 *MTUs for some networks*

<i>Protocol</i>	<i>MTU</i>
Hyperchannel	65,535
Token Ring (16 Mbps)	17,914
Token Ring (4 Mbps)	4,464
FDDI	4,352
Ethernet	1,500
X.25	576
PPP	296

ROUTING



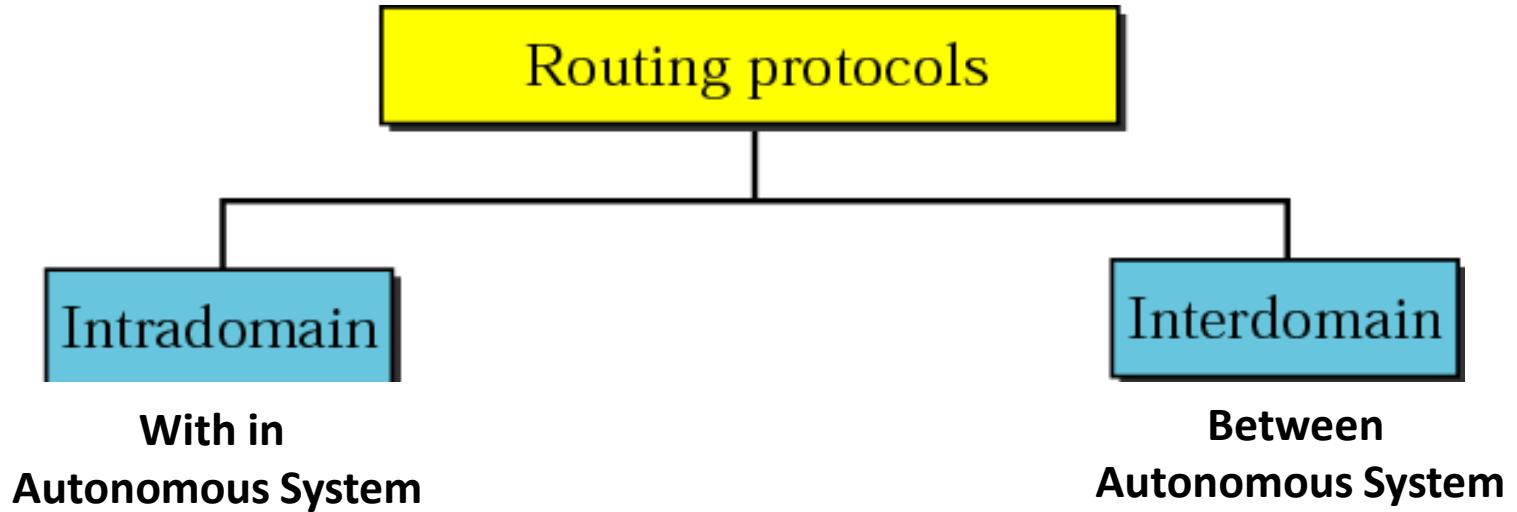


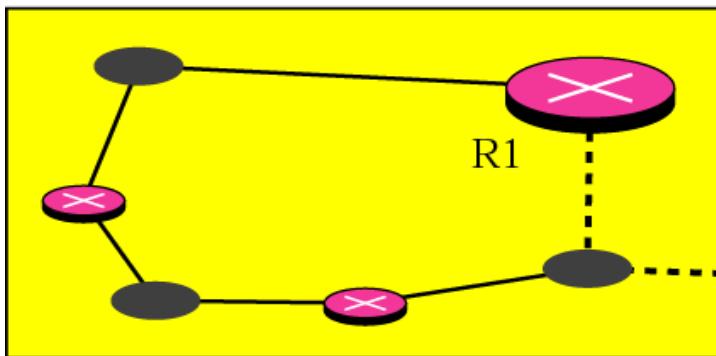
Figure 14.1 Autonomous systems

An *autonomous system* is a set of networks and routers under the control of a single administrative authority.

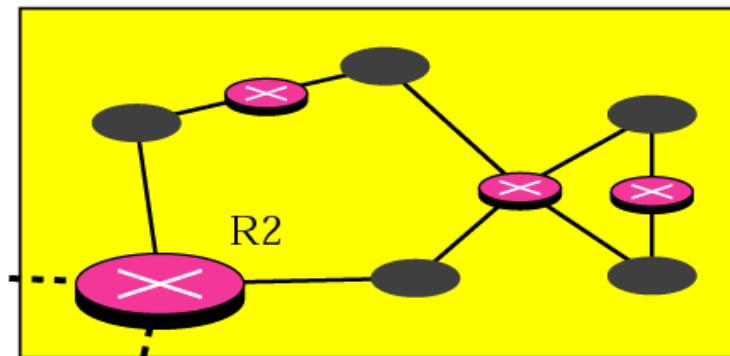
Routing within an autonomous system is *intradomain* routing.

Routing between autonomous systems is *interdomain* routing.

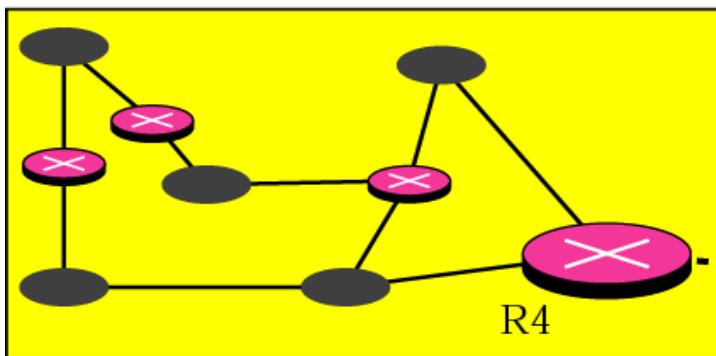
Autonomous system



Autonomous system

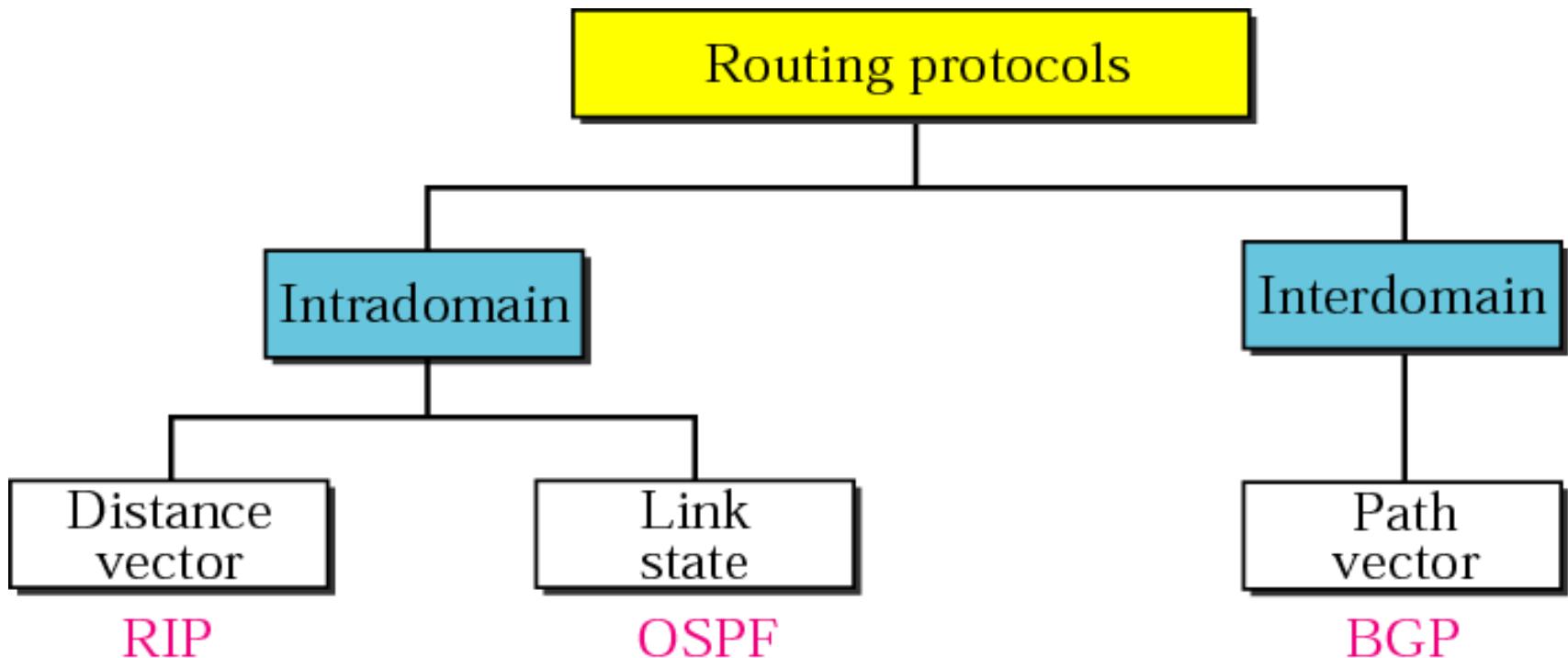


Autonomous system



Autonomous system

Figure 14.2 *Popular routing protocols*



14.2 DISTANCE VECTOR ROUTING

In distance vector routing, the least cost route between any two nodes is the route with minimum distance.

In this protocol each node maintains a vector (table) of minimum distances to every node

Figure 14.3 Distance vector routing tables

To Cost Next

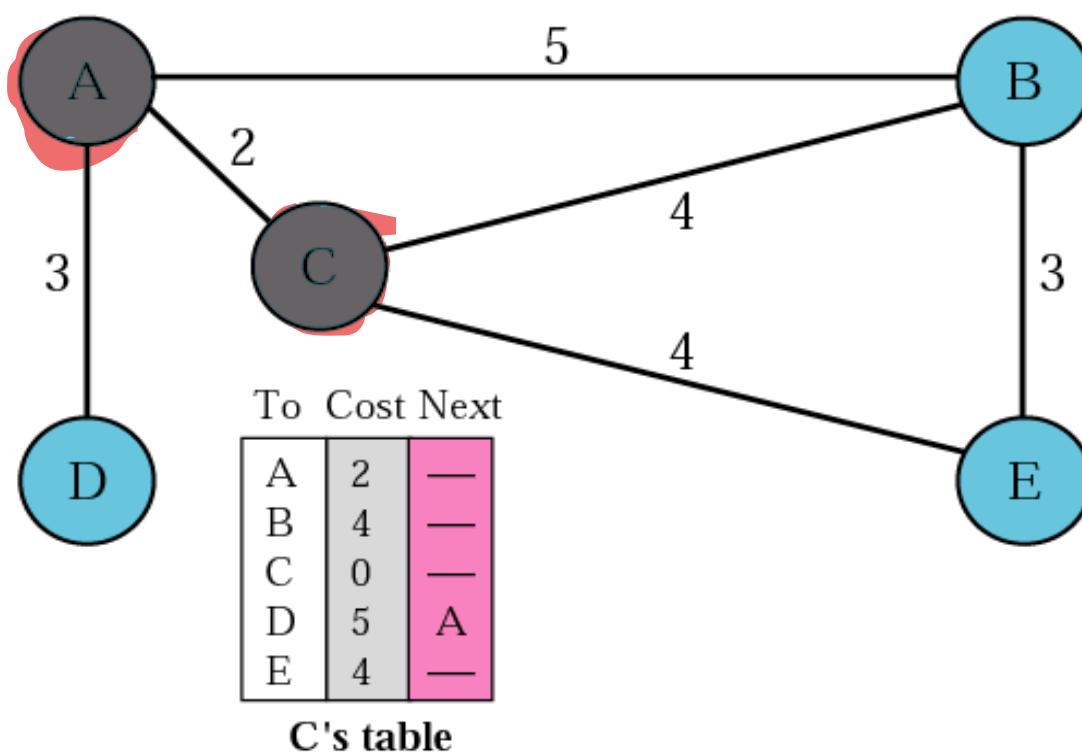
	To	Cost	Next
A	0	—	
B	5	—	
C	2	—	
D	3	—	
E	6	C	

A's table

To Cost Next

	To	Cost	Next
A	3	—	
B	8	A	
C	5	A	
D	0	—	
E	9	A	

D's table



To Cost Next

	To	Cost	Next
A	5	—	
B	0	—	
C	4	—	
D	8	A	
E	3	—	

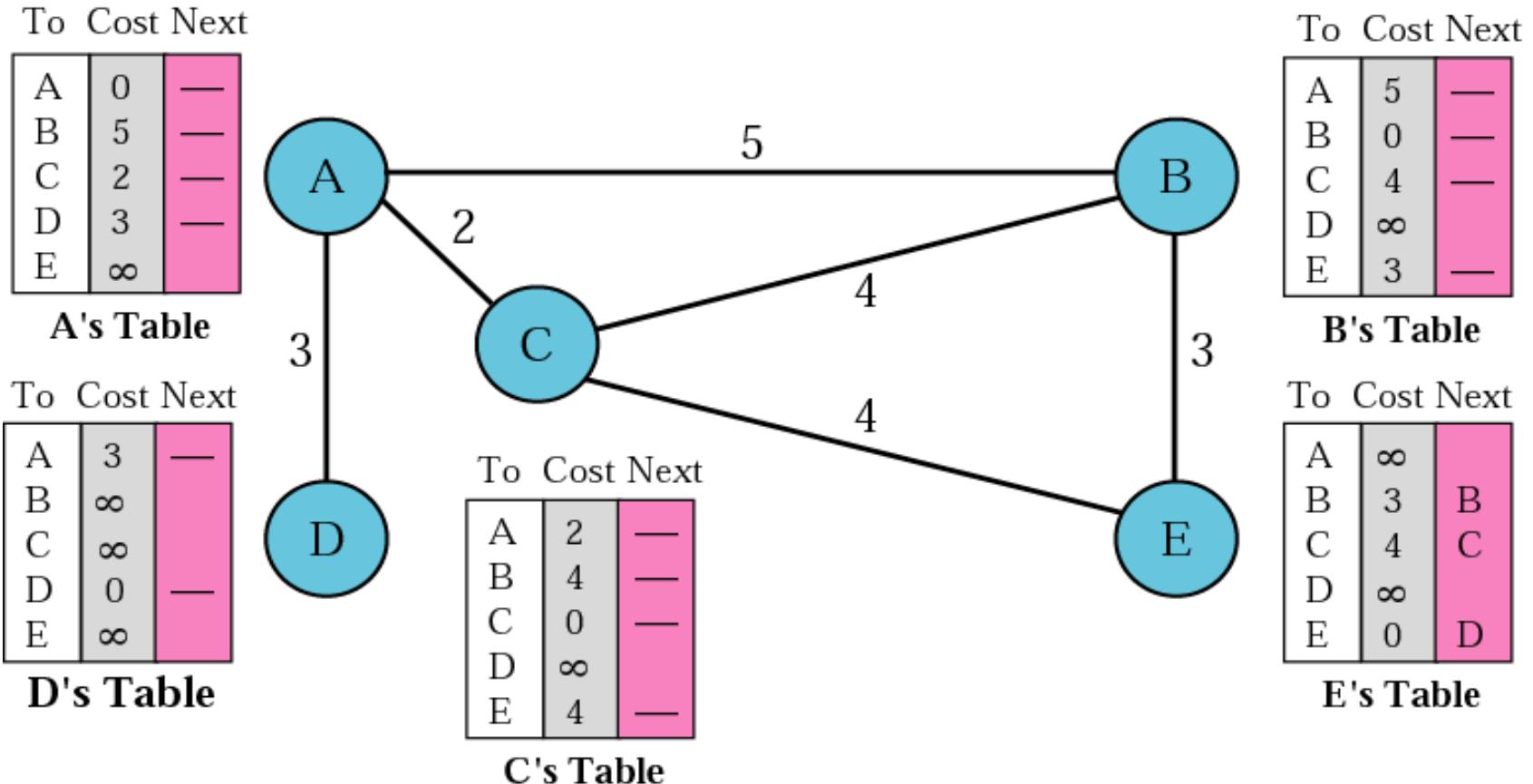
B's table

To Cost Next

	To	Cost	Next
A	6	C	
B	3	—	
C	4	—	
D	9	C	
E	0	—	

E's table

Figure 14.4 Initialization of tables in distance vector routing



In distance vector routing, each node shares its table with its immediate neighbor periodically (eg every 30s) and when there is a change.

Figure 14.5 *Updating in distance vector routing*

Step 1: Add cost (2) to table received from neighbor (C).
Step 2: Compare Modified Table with Old Table (row by row).
If Next node entry is different, select the row with the smaller cost. If tie, keep the old one.
If Next node entry the same, select the new row value (regardless of whether new value is smaller or not).

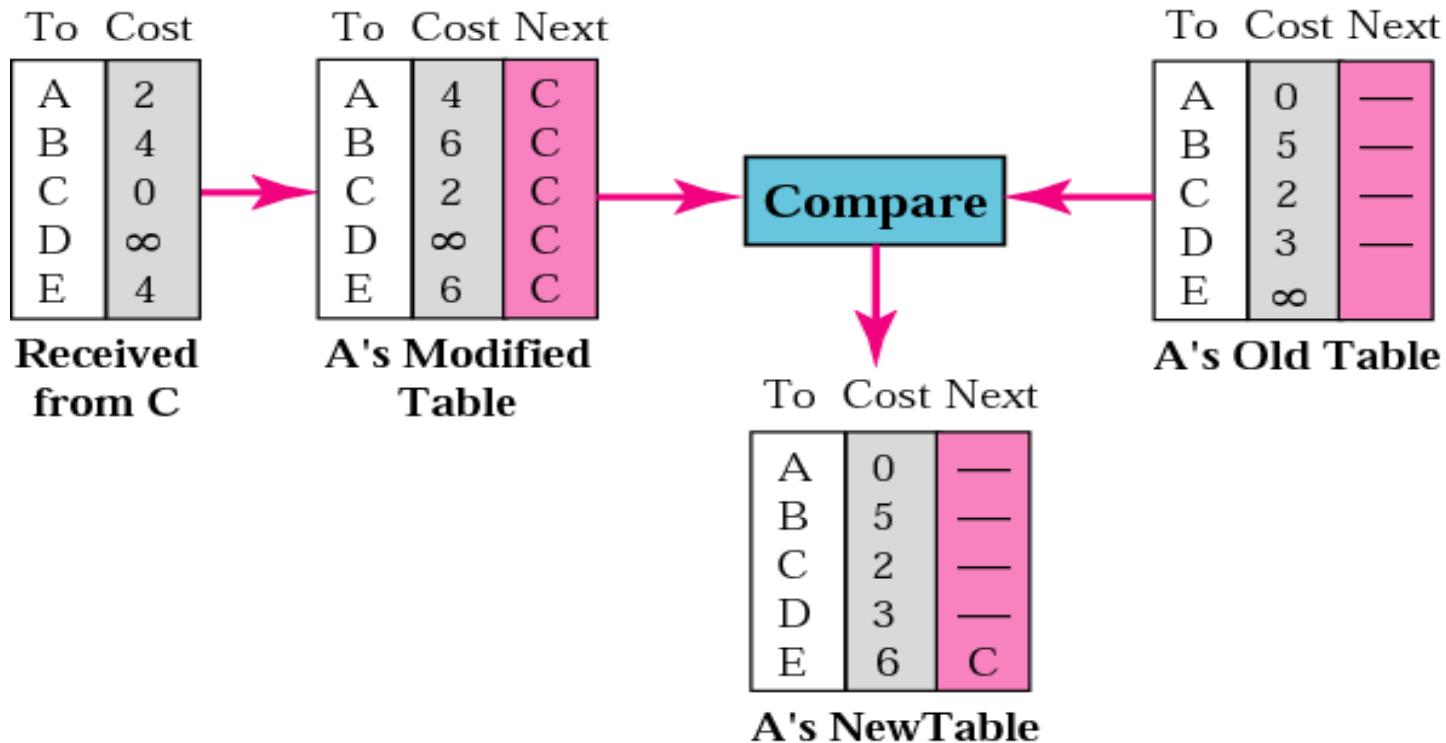
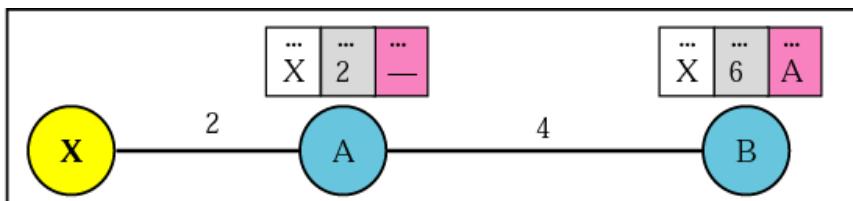


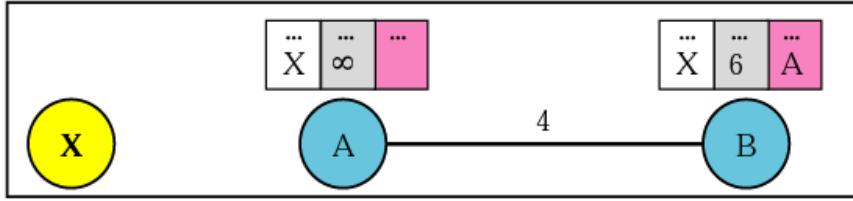
Figure 14.6 Two-node instability – what can happen with distance vector routing

Count to Infinity Problem

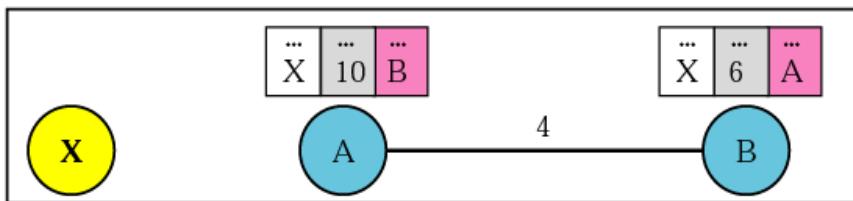
Before failure



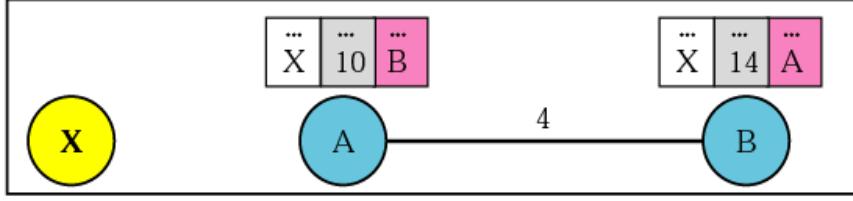
After failure



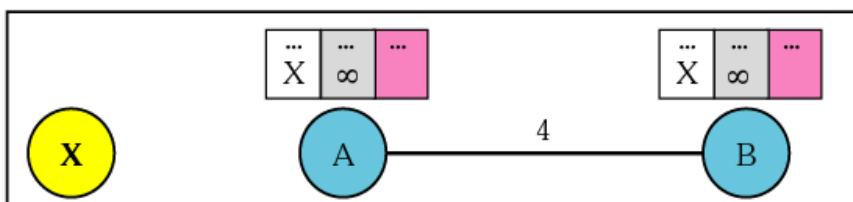
After A receives update from B



After B receives update from A



Finally



Both A and B know where X is.

Link between A and X fails. A updates its table immediately. But before A can tell B, B sends its info to A!

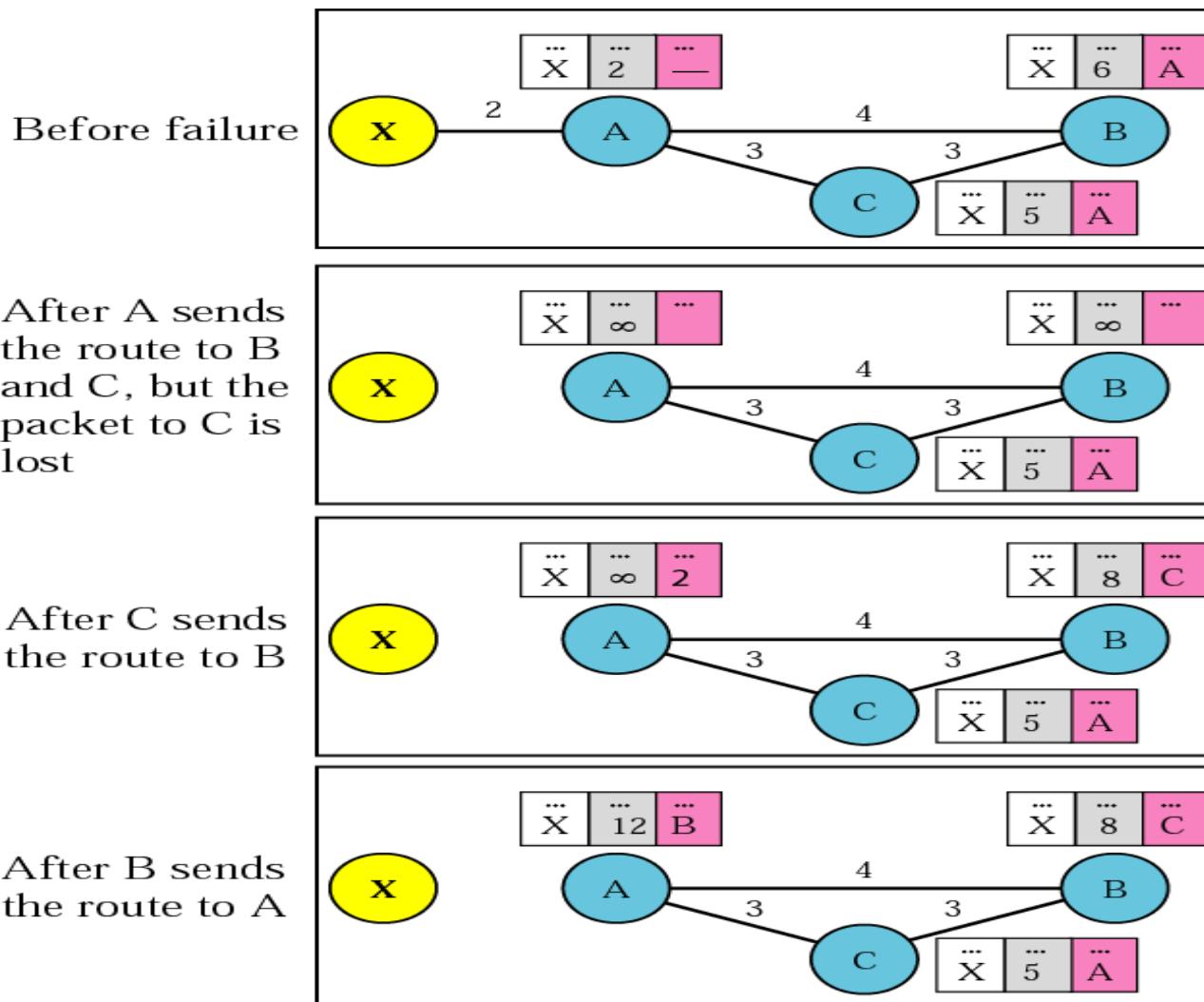
A, using B's info, updates its table (error!).

Then A send its table to B and B updates its table (more error). Both routers keep updating tables, eventually hitting infinity. In the meantime, chaos!

Possible Solutions to two-node instability:

1. Define infinity to be a much smaller value, such as 100. Then it doesn't take too long to become stable. But now you can't use distance vector routing in large networks.
2. **Split Horizon** – instead of flooding entire table to each node, only part of its table is sent. More precisely, if node B thinks that the optimum router to reach X is via A, then B does not need to advertise this piece of info to A – the info has already come from A.
3. **Split Horizon and Poison Reverse** – Normally, the distance vector protocol uses a timer. If there is no news about a route, the node deletes the route from its table. So when A never hears from B about the route to X, it deletes it. Instead, Node B still advertises the value for X, but if the source of info is A, it replaces the distance with infinity, saying “Do not use this value; what I know about this route comes from you.”

Three-node instability – no solutions here!



14.3 RIP

The Routing Information Protocol (RIP) is an intradomain routing protocol used inside an autonomous system.

It is a very simple protocol based on distance vector routing.

RIP

The **Routing Information Protocol (RIP)** is an intradomain routing protocol used inside an autonomous system. It is a very simple protocol based on distance vector routing. RIP implements distance vector routing directly with some considerations:

1. In an autonomous system, we are dealing with routers and networks (links). The routers have routing tables; networks do not.
2. The destination in a routing table is a network, which means the first column defines a network address.
3. The metric used by RIP is very simple; the distance is defined as the number of links (networks) to reach the destination. For this reason, the metric in RIP is called a **hop count**.
4. Infinity is defined as 16, which means that any route in an autonomous system using RIP cannot have more than 15 hops.
5. The next-node column defines the address of the router to which the packet is to be sent to reach its destination.

- RIP implements the same algorithm as the distance-vector routing algorithm we discussed in the previous section. However, some changes need to be made to the algorithm to enable a router to update its forwarding table:
 - 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.

Timers in RIP

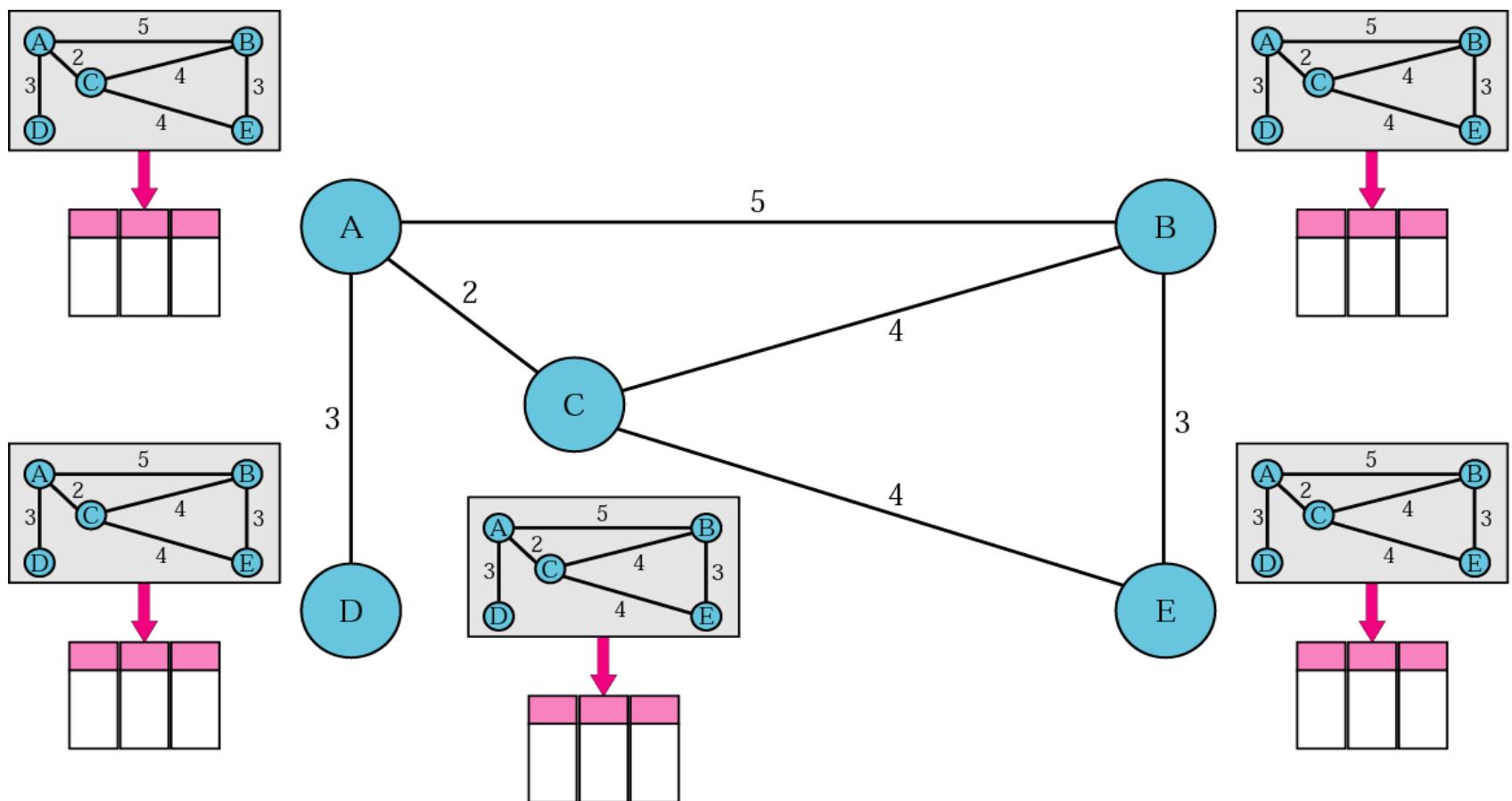
- RIP uses three timers to support its operation. The periodic timer controls the advertising of regular update messages. Each router has one periodic timer that is randomly set to a number between 25 and 35 seconds (to prevent all routers sending their messages at the same time and creating excess traffic). The timer counts down; when zero reached, the update message is sent, and the timer is randomly set once again.
- 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 the internet and no update is received within the allotted 180 seconds, the route is considered expired and the hop count of the route is set to 16, which means the destination is unreachable. Every route has its own expiration timer.
- The garbage collection timer 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.

LINK STATE ROUTING

In link state routing, if each node in the domain has the entire topology of the domain, the node can use Dijkstra's algorithm to build a routing table.

Figure 14.15 Concept of link state routing

The figure shows a simple domain with five nodes. Each node uses the same topology to create a routing table, but the routing table for each node is unique because the calculations are based on different interpretations of the topology. This is analogous to a city map. While each person may have the same map, each needs to take a different route to reach her specific destination.



Every router has knowledge about the network, but from its own perspective.¹⁰¹

Figure 14.16 *Link state knowledge*

Each router knows (maintains) its states of its links.

Each router floods this info (via a Link State Packet) to other routers periodically (when there is a change in the topology, or every 60 to 120 minutes).

Each router takes in this data and, using Dijkstra's algorithm, creates the shortest path tree and corresponding routing table.

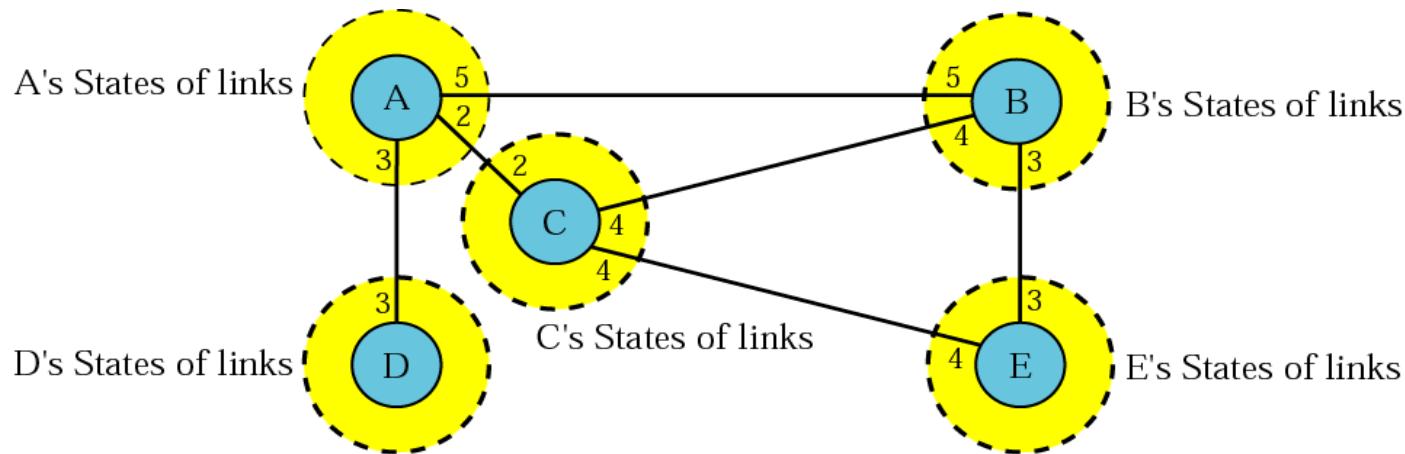


Figure 14.17 Dijkstra algorithm

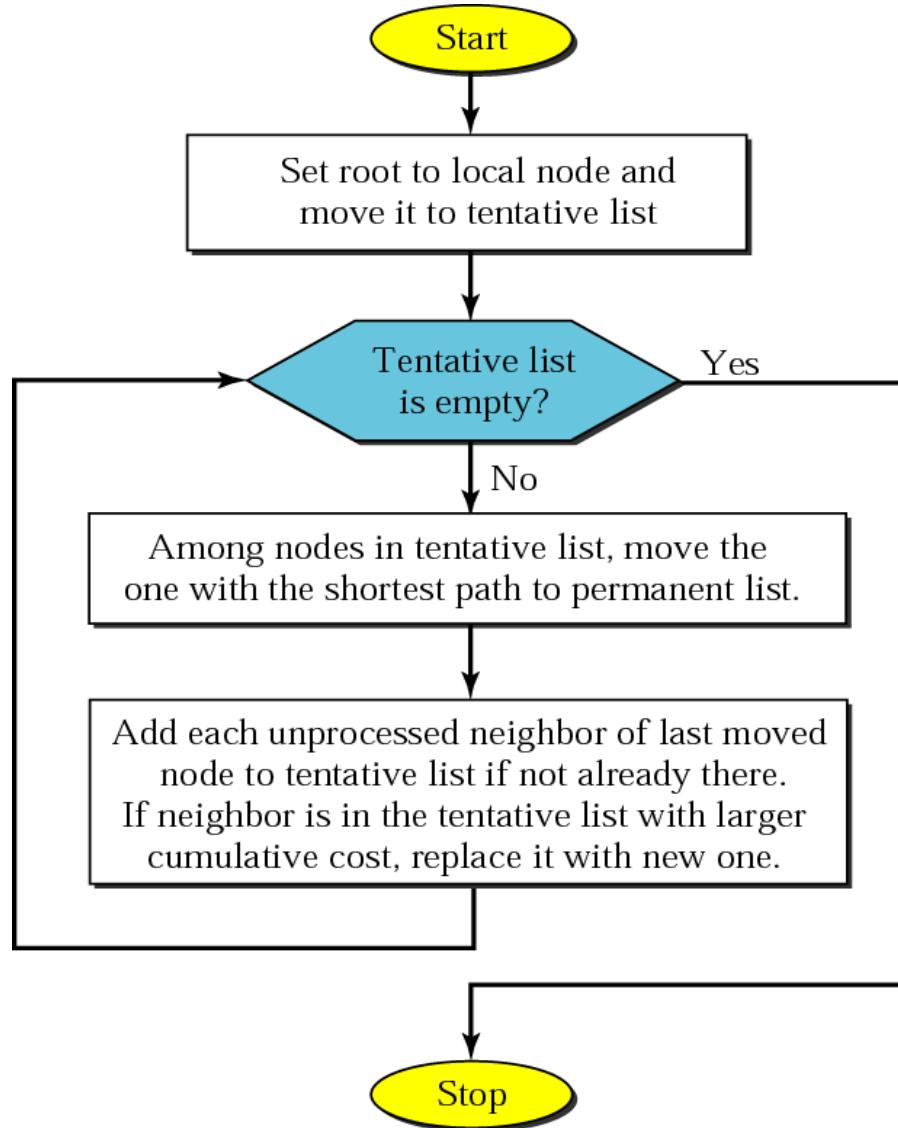


Figure 14.18 Example of formation of shortest path tree

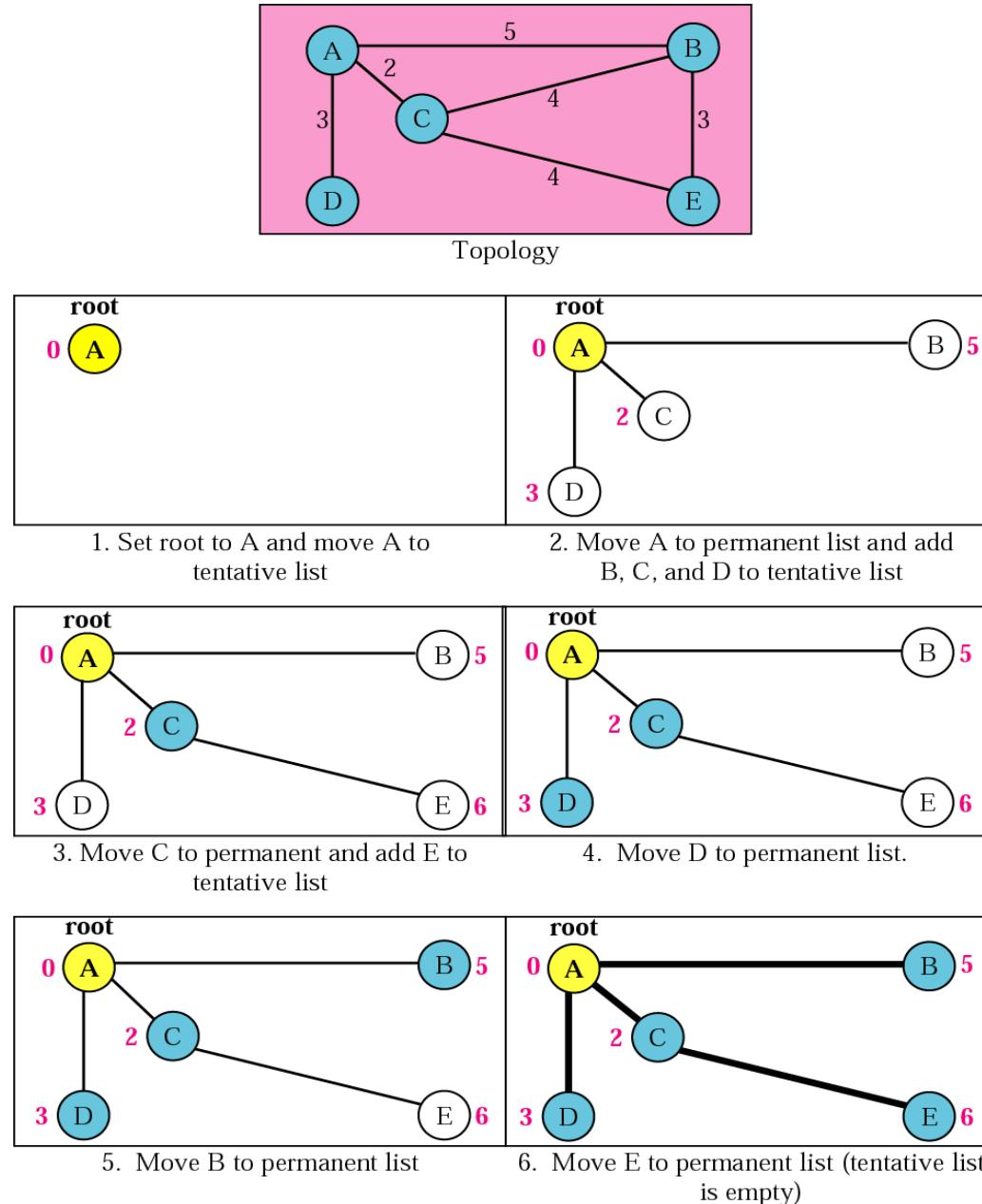


Table 14.1 Routing table for node A

<i>Node</i>	<i>Cost</i>	<i>Next Router</i>
A	0	—
B	5	—
C	2	—
D	3	—
E	6	C

Now let's try using the Dijkstra's algorithm introduced in TDC 361.

Comparison of Protocols

Link State

- Knowledge of every router's links (entire graph)
- Every router has $O(\# \text{ edges})$
- Trust a peer's info, do routing computation yourself
- Use Dijkstra's algorithm
- Send updates on any link-state changes
- Ex: OSPF, IS-IS
- Adv: Fast to react to changes

Distance Vector

- Knowledge of neighbors' distance to destinations
- Every router has $O (\# \text{neighbors} * \# \text{nodes})$
- Trust a peer's routing computation
- Use Bellman-Ford algorithm
- Send updates periodically or routing decision change
- Ex: RIP, IGRP
- Adv: Less info & lower computational overhead

OSPF

The Open Shortest Path First (OSPF) protocol is an intradomain routing protocol based on link state routing.

Its domain is also an autonomous system.

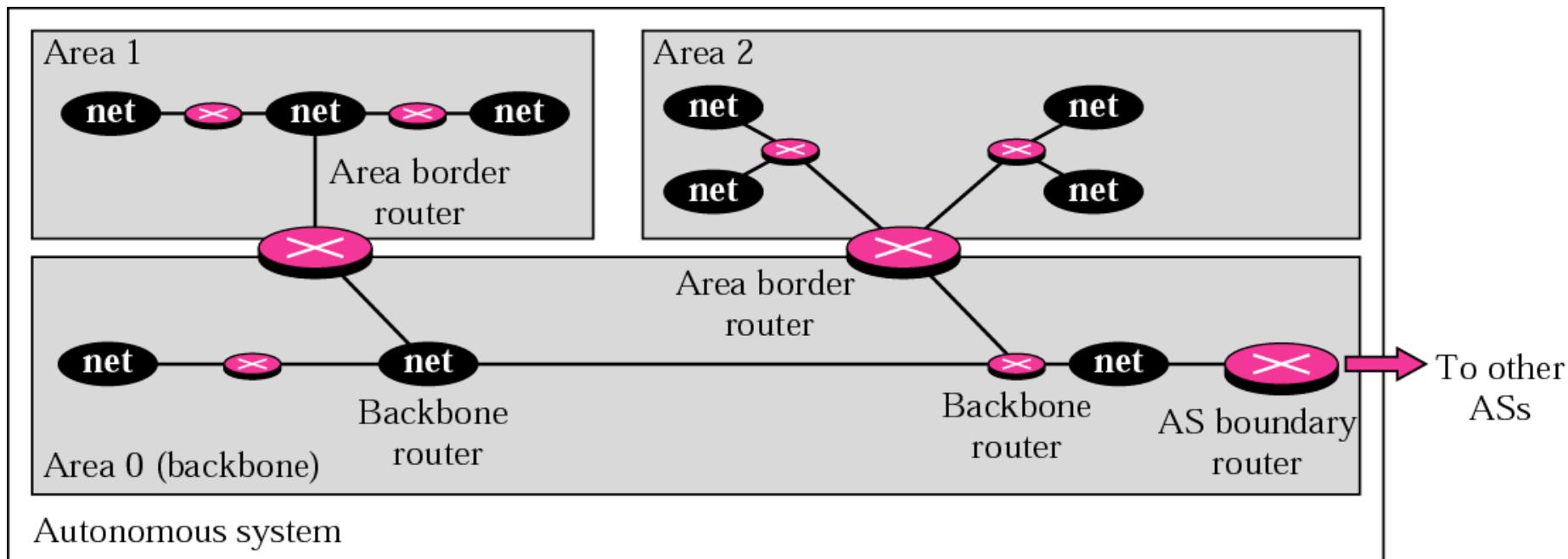
Also known as Dijkstra's Algorithm

Figure 14.19 Areas in an autonomous system

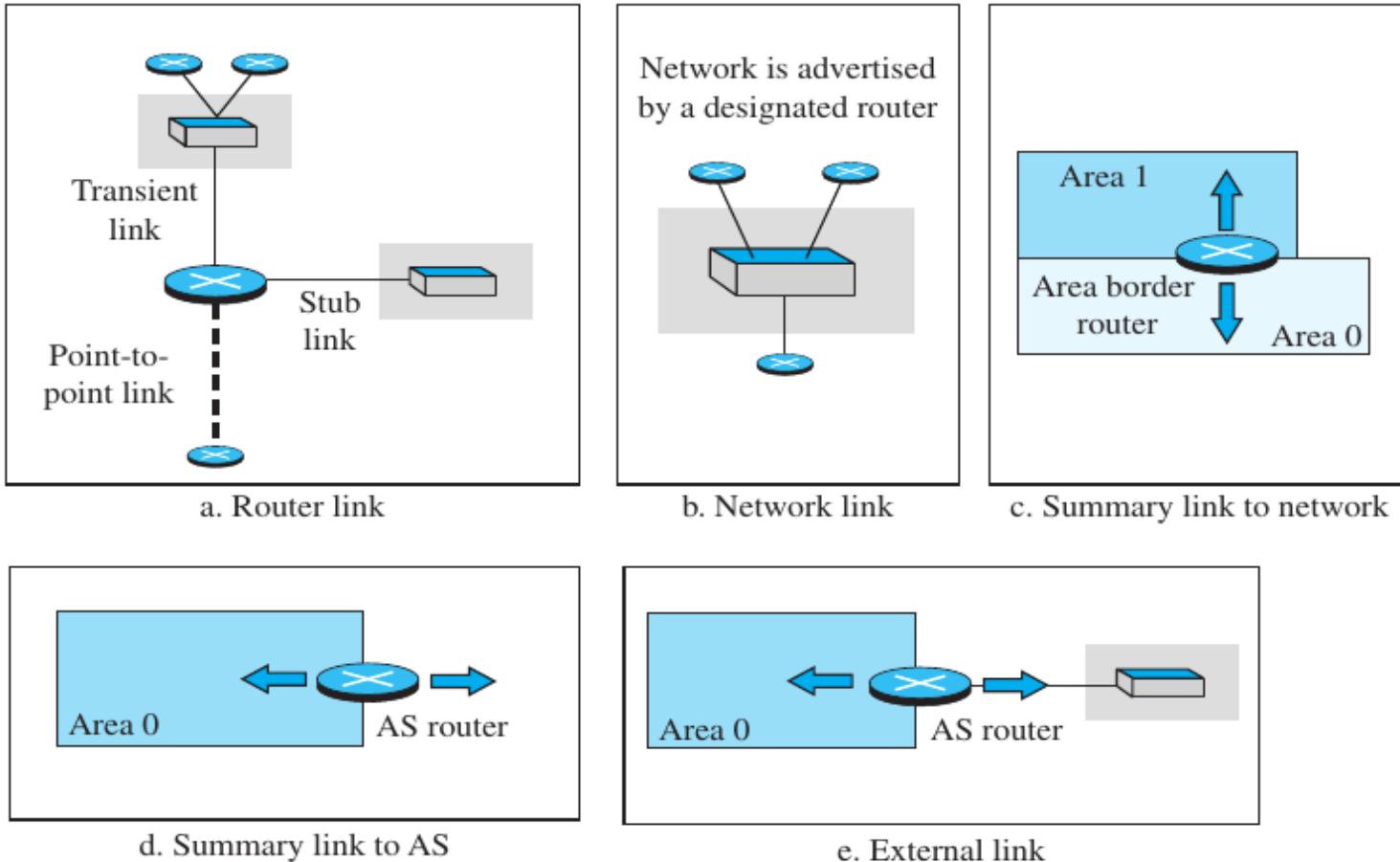
OSPF divides an autonomous system into areas. All networks inside an area must be connected.

area border router; backbones; backbone routers;
boundary routers

The cost associated with a route is called the metric. Metric could be min delay, max thruput, etc.



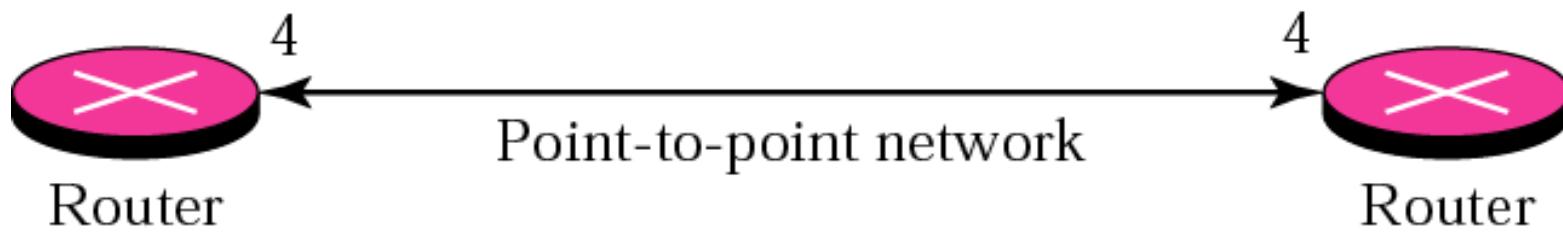
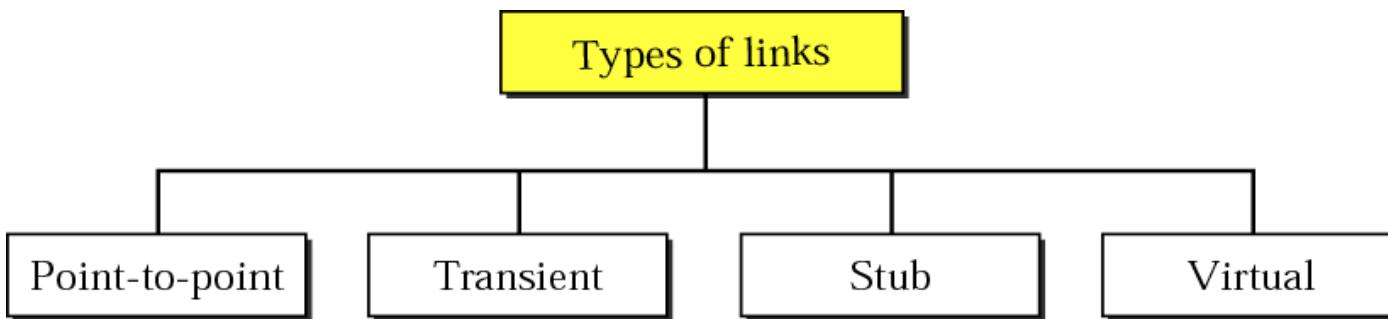
LSPs (Link State Packets)



LSPs

1. Router Link: A router link advertises the existence of a router as a node. In addition to giving the address of the announcing router, this type of advertisement can define one or more types of links that connect the advertising router to other entities.

Figure 14.21 *Point-to-point link*



No hosts in between; T-1 connection common

Types of link

- A **transient link** announces a link to a transient network, a network that is connected to the rest of the networks by one or more routers. This type of advertisement should define the address of the transient network and the cost of the link.
- A **stub link** advertises a link to a stub network, a network that is not a through network. Again, the advertisement should define the address of the network and the cost.
- A **point-to-point link** should define the address of the router at the end of the point-to-point line and the cost to get there.

2. Network Link: A network link advertises the network as a node. However, since a network cannot do announcements itself (it is a passive entity), one of the routers is assigned as the designated router and does the advertising.

3. Summary link to network: This is done by an area border router; it advertises the summary of links collected by the backbone to an area or the summary of links collected by the area to the backbone.

LSPs

- 4. Summary link to AS:** This is done by an AS router that advertises the summary links from other ASs to the backbone area of the current AS, information which later can be disseminated to the areas so that they will know about the networks in other ASs.
- 5. External link.** This is also done by an AS router to announce the existence of a single network outside the AS to the backbone area to be disseminated into the areas.

OSPF message format

0	8	16	31
Version	Type	Message length	
		Source router IP address	
		Area identification	
Checksum	Authentication type		
	Authentication		

OSPF common header

LS age	E	T	LS type
LS ID			
Advertising router			
LS sequence number			
LS checksum	Length		

Link-state general header

14.6 PATH VECTOR ROUTING

Path vector routing is similar to distance vector routing. There is at least one node, called the speaker node, in each AS that creates a routing table and advertises it to other speaker nodes in the neighboring ASs..

The topics discussed in this section include:

Initialization

Sharing

Updating

Figure 14.48 Initial routing tables in path vector routing

Dest. Path

A1	AS1
A2	AS1
A3	AS1
A4	AS1
A5	AS1

A1 Table

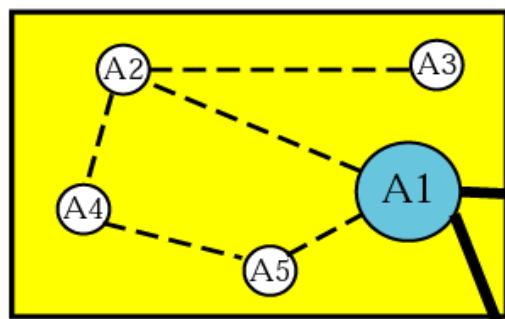
AS 1

Dest. Path

C1	AS3
C2	AS3
C3	AS3

C1 Table

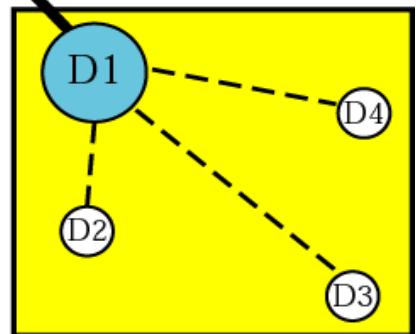
AS 3



Dest. Path

D1	AS4
D2	AS4
D3	AS4
D4	AS4

D1 Table



AS 4

Dest. Path

B1	AS2
B2	AS2
B3	AS2
B4	AS2

B1 Table

AS 2

Figure 14.49 *Stabilized tables for four autonomous systems*

Dest. Path

A1	AS1
...	
A5	AS1
B1	AS1-AS2
...	...
B4	AS1-AS2
C1	AS1-AS3
...	
C3	AS1-AS3
D1	AS1-AS2-AS4
...	
D4	AS1-AS2-AS4

A1 Table

Dest. Path

A1	AS2-AS1
...	
A5	AS2-AS1
B1	AS2
...	...
B4	AS2
C1	AS2-AS3
...	
C3	AS2-AS3
D1	AS2-AS3-AS4
...	
D4	AS2-AS3-AS4

B1 Table

Dest. Path

A1	AS3-AS1
...	
A5	AS3-AS1
B1	AS3-AS2
...	...
B4	AS3-AS2
C1	AS3
...	
C3	AS3
D1	AS3-AS4
...	
D4	AS3-AS4

C1 Table

Dest. Path

A1	AS4-AS3-AS1
...	
A5	AS4-AS3-AS1
B1	AS4-AS3-AS2
...	...
B4	AS4-AS3-AS2
C1	AS4-AS3
...	
C3	AS4-AS3
D1	AS4
...	
D4	AS4

D1 Table

14.7 BGP

Border Gateway Protocol (BGP) is an interdomain routing protocol using path vector routing. It first appeared in 1989 and has gone through four versions.

BGP interconnects three different types of AS:

- 1. Stub AS, e.g. a corporate network*
- 2. Multihomed AS, e.g. a large corporate network with connections to multiple ASs, but does not allow traffic to pass thru (transient)*
- 3. Transit AS - one that allows transient traffic, such as an Internet backbone*

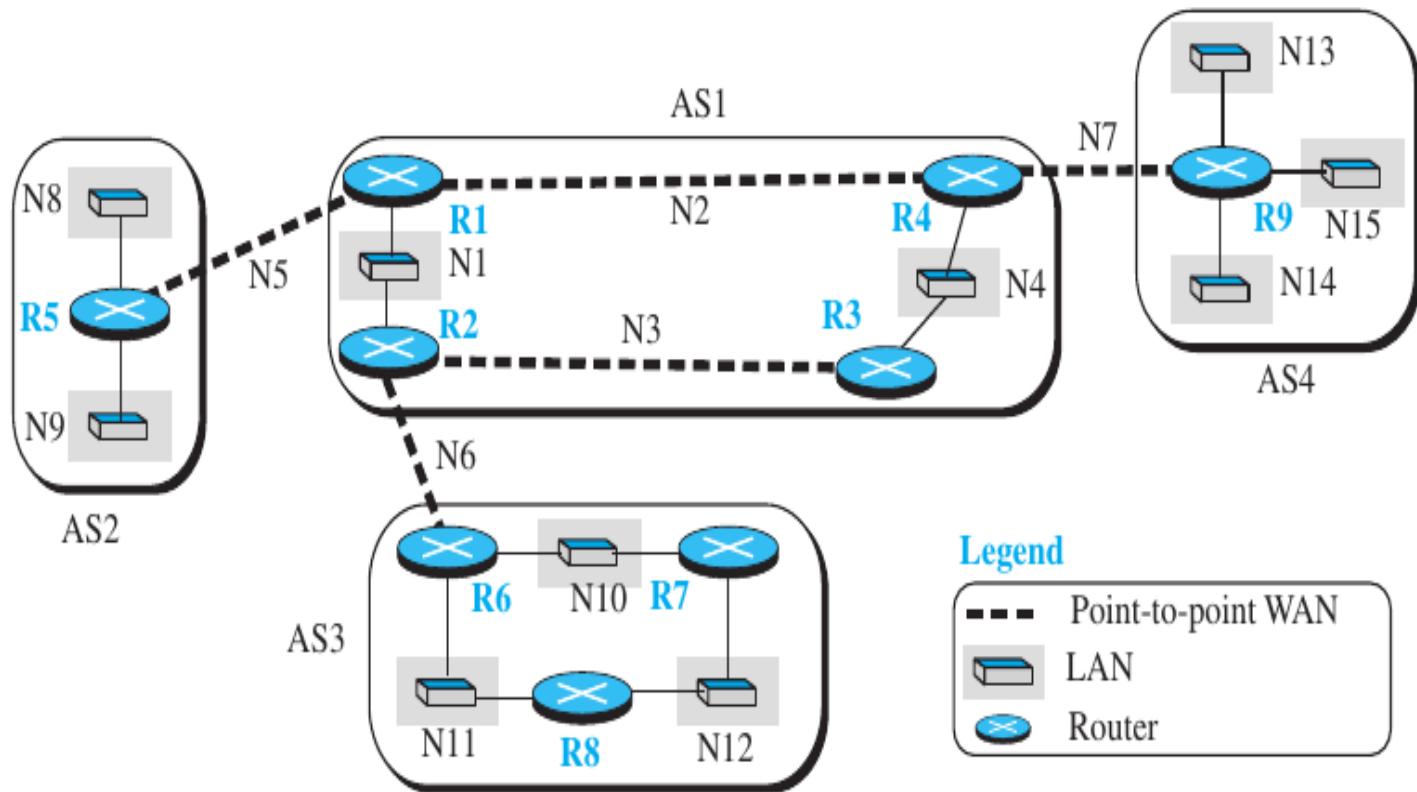


Figure 14.50 Internal and external BGP sessions

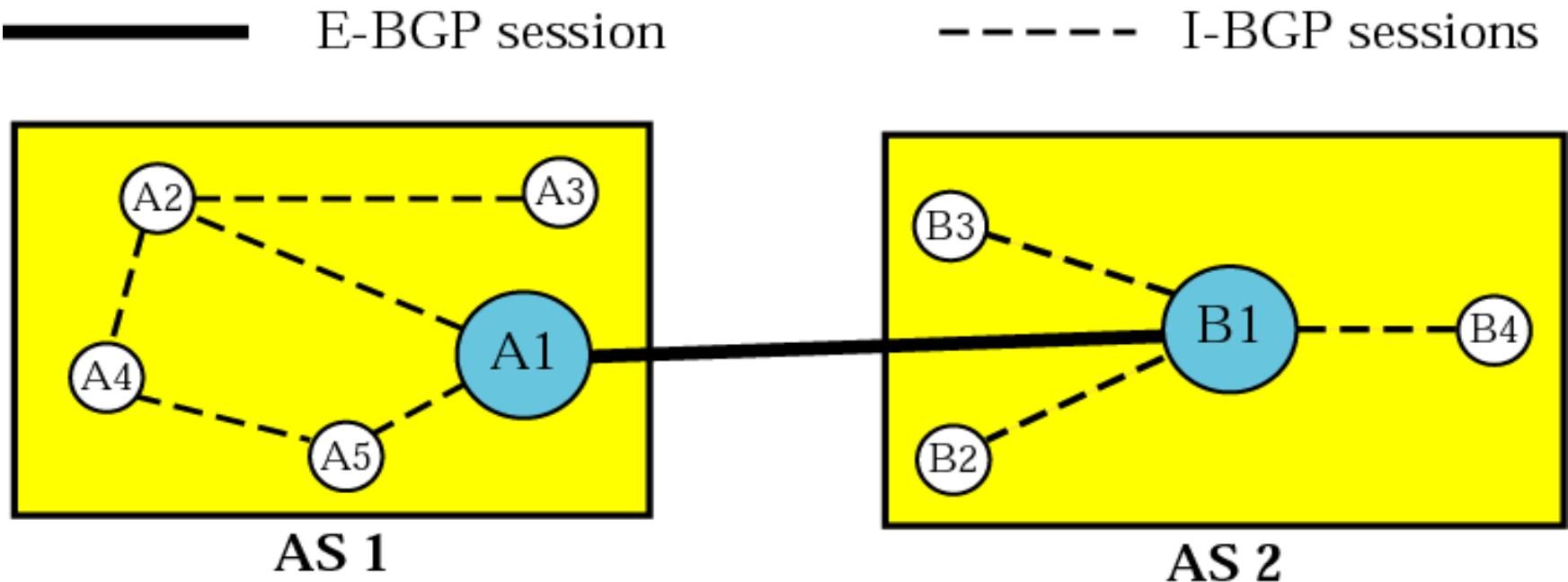
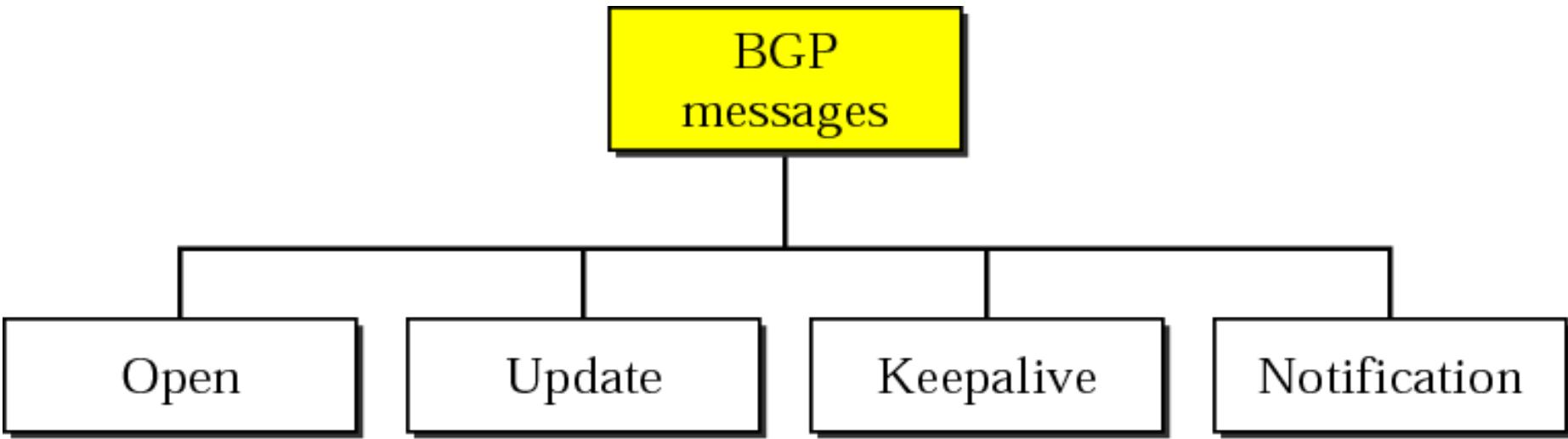


Figure 14.51 *Types of BGP messages*



BGP Messages

Open Message. To create a neighborhood relationship, a router running BGP opens a TCP connection with a neighbor and sends an open message.

Update Message. The update message is the heart of the BGP protocol. It is used by a router to withdraw destinations that have been advertised previously, to announce a route to a new destination, or both. Note that BGP can withdraw several destinations that were advertised before, but it can only advertise one new destination (or multiple destinations with the same path attributes) in a single update message.

Keepalive Message. The BGP peers that are running exchange keepalive messages regularly (before their hold time expires) to tell each other that they are alive.

Notification. A notification message is sent by a router whenever an error condition is detected or a router wants to close the session.



Note:

*BGP supports classless addressing and
CIDR.*



Note:

*BGP uses the services of TCP
on port 179.*

*RIP uses the services of UDP on port
520.*

20-2 IPv4

*The Internet Protocol version 4 (**IPv4**) is the delivery mechanism used by the TCP/IP protocols.*

Topics discussed in this section:

Datagram

Fragmentation

Checksum

Options

Figure 20.4 Position of IPv4 in TCP/IP protocol suite

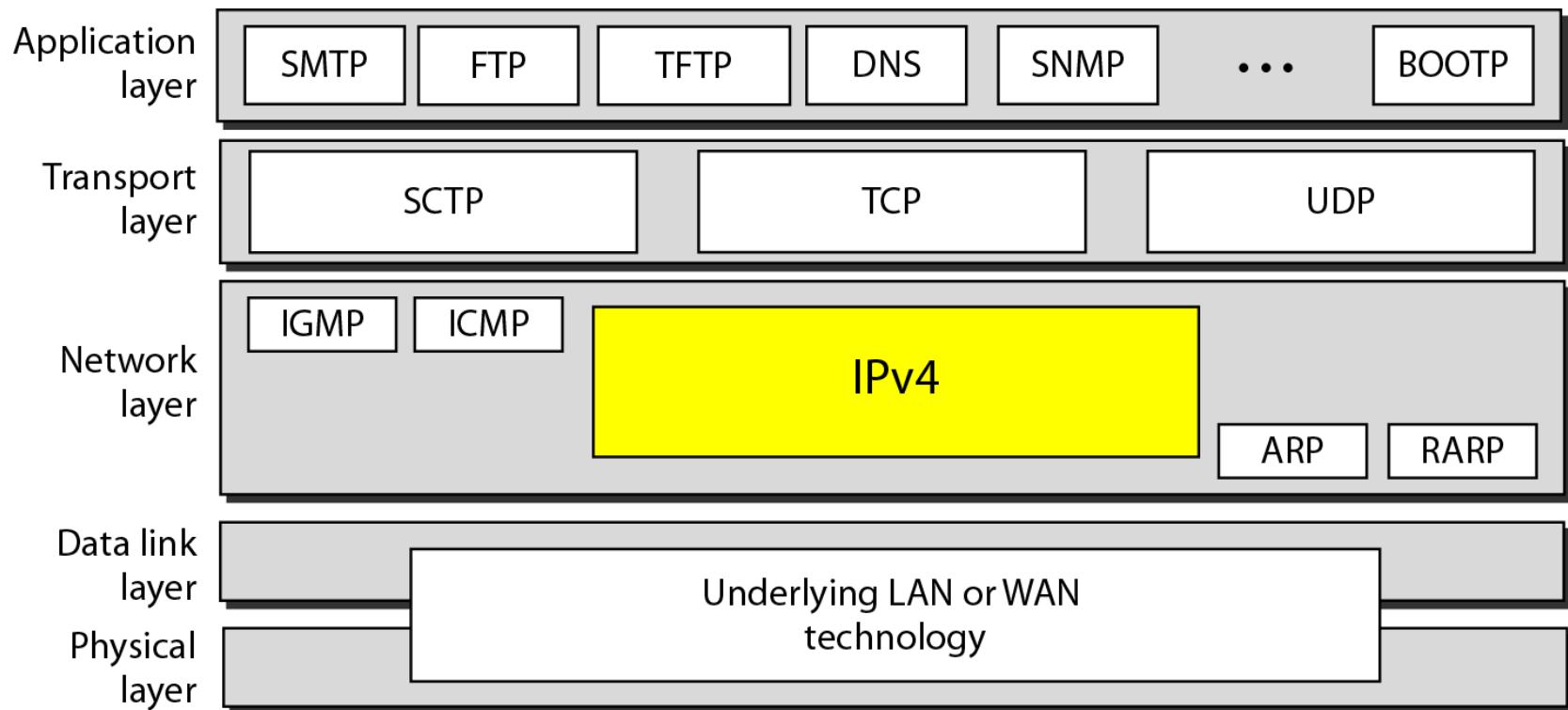
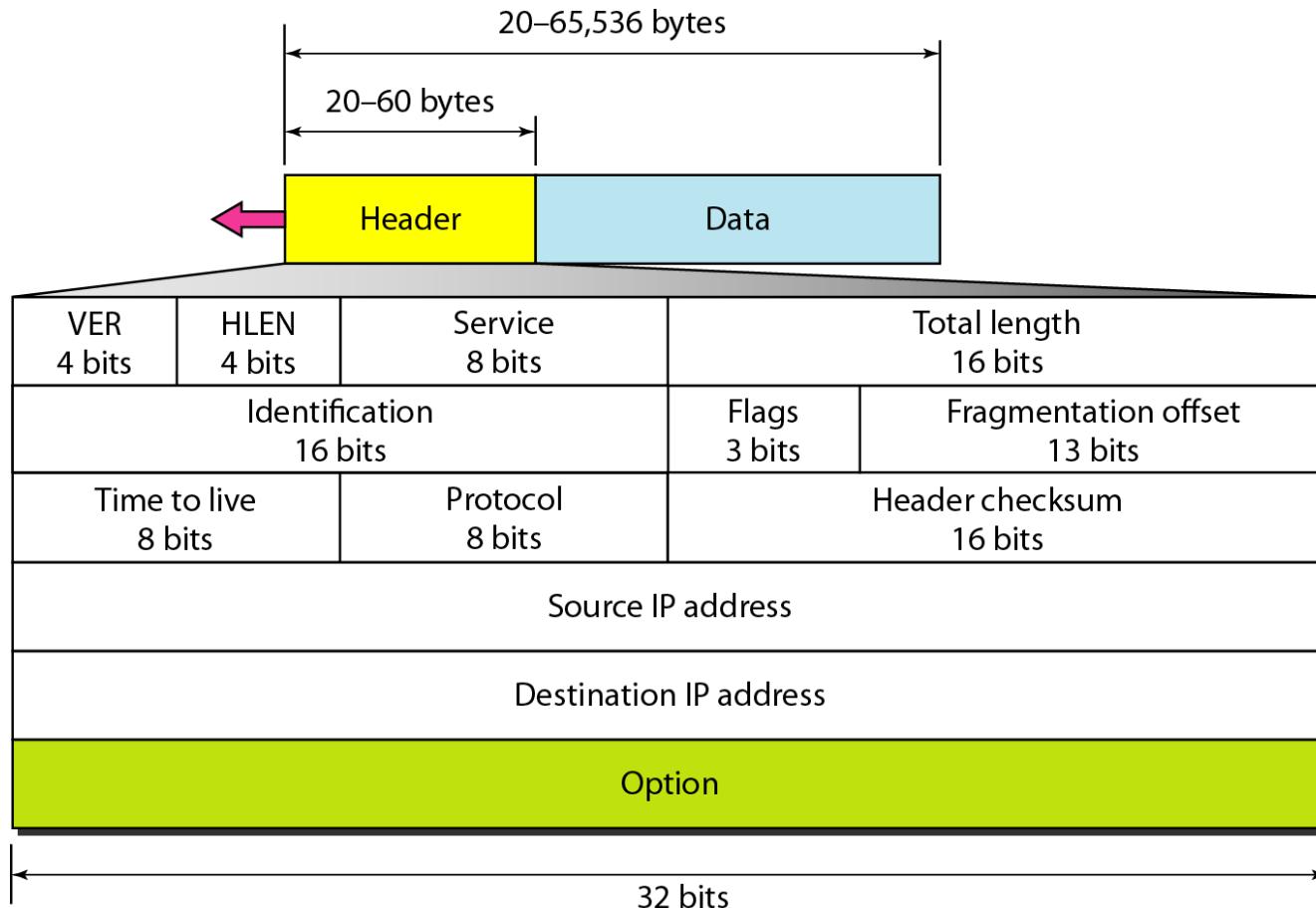


Figure 20.5 IPv4 datagram format



Header length (HLEN). This is a 4-bit field

This field is needed because the length of the header is variable (between 20 and 60 bytes).

When there are no options, the header length is 20 bytes, and the value of this field is 5 ($5 \times 4 = 20$).

When the option field is at its maximum size, the value of this field is 15 ($15 \times 4 = 60$).

Service Type

First 3 bits are called precedence bits.

Next 4 bits are called type of service (TOS) bits.

Last bit is not used.

The precedence defines the priority of the datagram

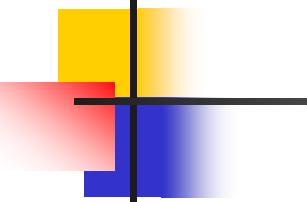
TOS bits is a 4-bit subfield with each bit having a special meaning. Although a bit can be either 0 or 1,

Table 20.1 *Types of service*

<i>TOS Bits</i>	<i>Description</i>
0000	Normal (default)
0001	Minimize cost
0010	Maximize reliability
0100	Maximize throughput
1000	Minimize delay

Table 20.2 *Default types of service*

<i>Protocol</i>	<i>TOS Bits</i>	<i>Description</i>
ICMP	0000	Normal
BOOTP	0000	Normal
NNTP	0001	Minimize cost
IGP	0010	Maximize reliability
SNMP	0010	Maximize reliability
TELNET	1000	Minimize delay
FTP (data)	0100	Maximize throughput
FTP (control)	1000	Minimize delay
TFTP	1000	Minimize delay
SMTP (command)	1000	Minimize delay
SMTP (data)	0100	Maximize throughput
DNS (UDP query)	1000	Minimize delay
DNS (TCP query)	0000	Normal
DNS (zone)	0100	Maximize throughput



Note

The total length field defines the total length of the datagram including the header.

Identification

This 16-bit field identifies a datagram originating from the source host. The combination of the identification and source IPv4 address must uniquely define a datagram as it leaves the source host.

Figure 20.10 *Flags used in fragmentation*



Flags. This is a 3-bit field.

The first bit is reserved.

The second bit is called the do Not fragment bit. If its value is 1, the machine must not fragment the datagram. If its value is 0, the datagram can be fragmented if necessary.

The third bit is called the more fragment bit. If its value is 1, it means the datagram is not the last fragment; there are more fragments after this one. If its value is 0, it means this is the last or only fragment

Figure 20.8 *Protocol field and encapsulated data*

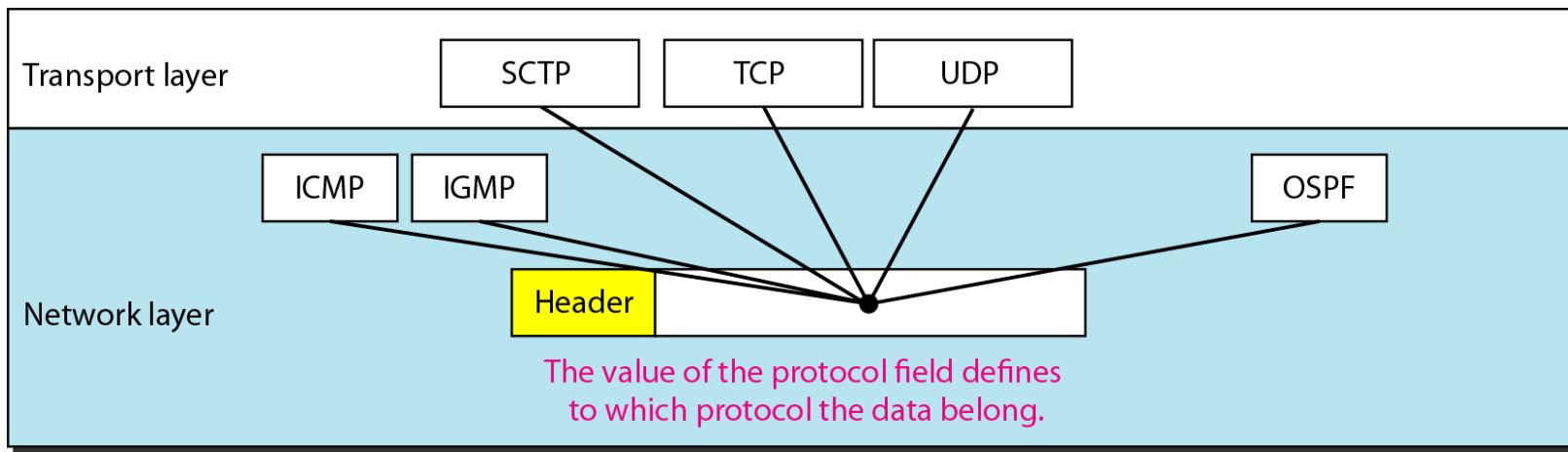
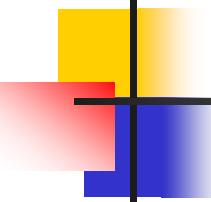


Table 20.4 *Protocol values*

<i>Value</i>	<i>Protocol</i>
1	ICMP
2	IGMP
6	TCP
17	UDP
89	OSPF



Example 20.1

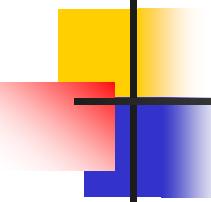
An IPv4 packet has arrived with the first 8 bits as shown:

01000010

The receiver discards the packet. Why?

Solution

There is an error in this packet. The 4 leftmost bits (0100) show the version, which is correct. The next 4 bits (0010) 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.

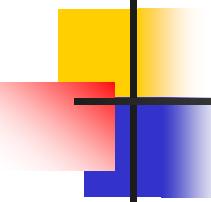


Example 20.2

In an IPv4 packet, the value of HLEN is 1000 in binary. 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.

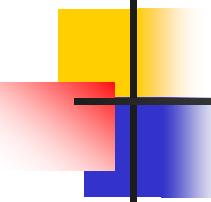


Example 20.3

In an IPv4 packet, the value of HLEN is 5, and the value of the total length field is 0x0028. 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 40 bytes, which means the packet is carrying 20 bytes of data ($40 - 20$).



Example 20.4

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

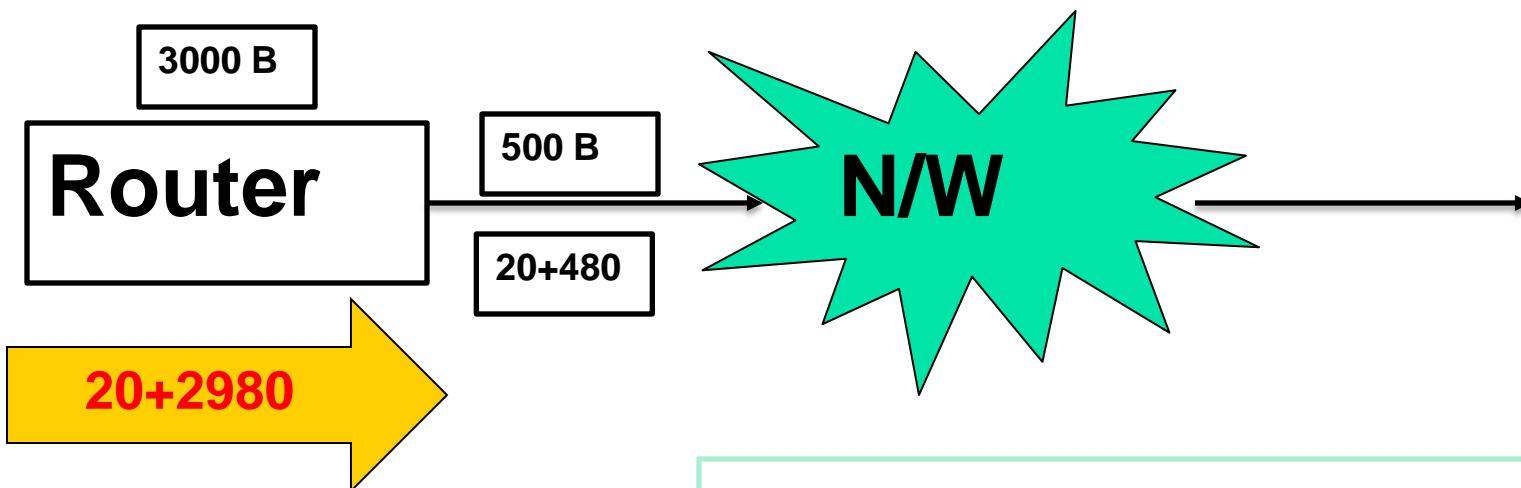
0x45000028000100000102 ...

How many hops can this packet travel before being dropped? The data belong to what upper-layer protocol?

Solution

To find the time-to-live field, we skip 8 bytes. The time-to-live field is the ninth byte, which is 01. This means the packet can travel only one hop. The protocol field is the next byte (02), which means that the upper-layer protocol is IGMP.

A Datagram of 3000 B (20 B header +2980 B Data) reached at router and must be forwarded to a link with MTU of 500 B. how many fragmentations will be generated and also write MF (more fragment), offset and total length value for all



$$2980/480 = 7 \text{ fragment}$$

P7	P6	P5	P4	P3	P2	P1
100+20	480+20	480+20	480+20	480+20	480+20	480+20
0	1	1	1	1	1	1 MF
360	300	240	180	120	60	0 Offset

For P2

Offset=480/8=60

For P3

Offset=480+480/8=120

An IP datagram of size 1000 bytes arrives at a router. The router has to forward this packet on a link whose MTU (maximum transmission unit) is 100 bytes. Assume that the size of the IP header is 20 bytes. The number of fragments that the IP datagram will be divided into for transmission is :

Note : This question was asked as Numerical Answer Type.

- (A) 10
- (B) 50
- (C) 12
- (D) 13

MTU = 100 bytes

Size of IP header = 20 bytes

So, size of data that can be transmitted in one fragment = $100 - 20 = 80$ bytes

Size of data to be transmitted = Size of datagram – size of header = $1000 - 20 = 980$ bytes

Now, we have a datagram of size 1000 bytes.

So, we need $\text{ceil}(980/80) = 13$ fragments.

Thus, there will be 13 fragments of the datagram.

An IP router with a Maximum Transmission Unit (MTU) of 1500 bytes has received an IP packet of size 4404 bytes with an IP header of length 20 bytes. The values of the relevant fields in the header of the third IP fragment generated by the router for this packet are

- (A) MF bit: 0, Datagram Length: 1444; Offset: 370
- (B) MF bit: 1, Datagram Length: 1424; Offset: 185
- (C) MF bit: 1, Datagram Length: 1500; Offset: 37
- (D) MF bit: 0, Datagram Length: 1424; Offset: 2960

$$\begin{aligned}\text{Number of packet fragments} &= \lceil (\text{total size of packet})/(\text{MTU}) \rceil \\ &= \lceil 4384/1480 \rceil \\ &= \lceil 2.962 \rceil \\ &= 3\end{aligned}$$

So Datagram with data 4404 byte fragmented into 3 fragments.
The first frame carries bytes 0 to 1479 (because MTU is 1500 bytes and HLEN is 20 byte so the total bytes in fragments is maximum $1500-20=1480$). the offset for this datagram is $0/8 = 0$.

The second fragment carries byte 1480 to 2959. The offset for this datagram is $1480/8 = 185$.finally the third fragment carries byte 2960 to 4404.the offset is 370.and for all fragments except last one the M bit is 1.so in the third bit M is 0..

Figure 20.9 *Maximum transfer unit (MTU)*

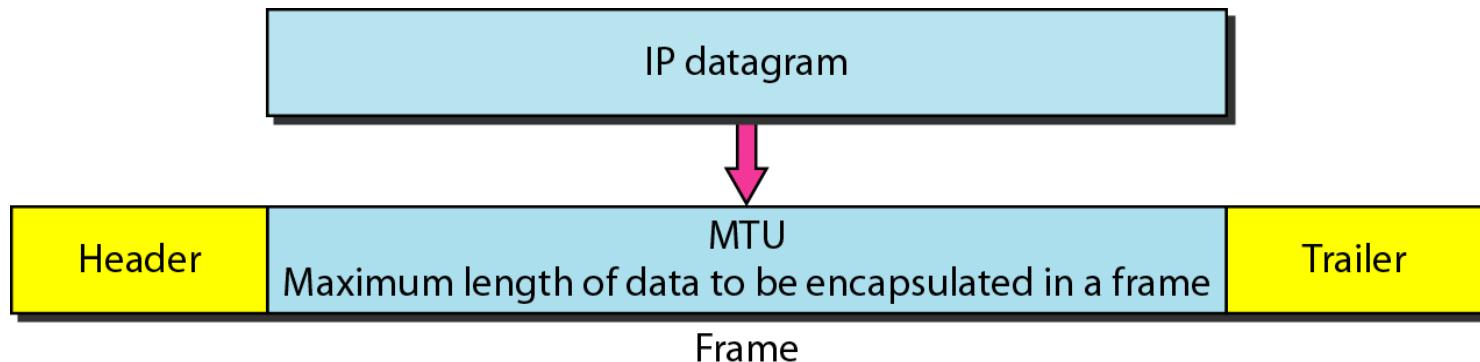


Table 20.5 *MTUs for some networks*

<i>Protocol</i>	<i>MTU</i>
Hyperchannel	65,535
Token Ring (16 Mbps)	17,914
Token Ring (4 Mbps)	4,464
FDDI	4,352
Ethernet	1,500
X.25	576
PPP	296

Figure 20.11 *Fragmentation example*

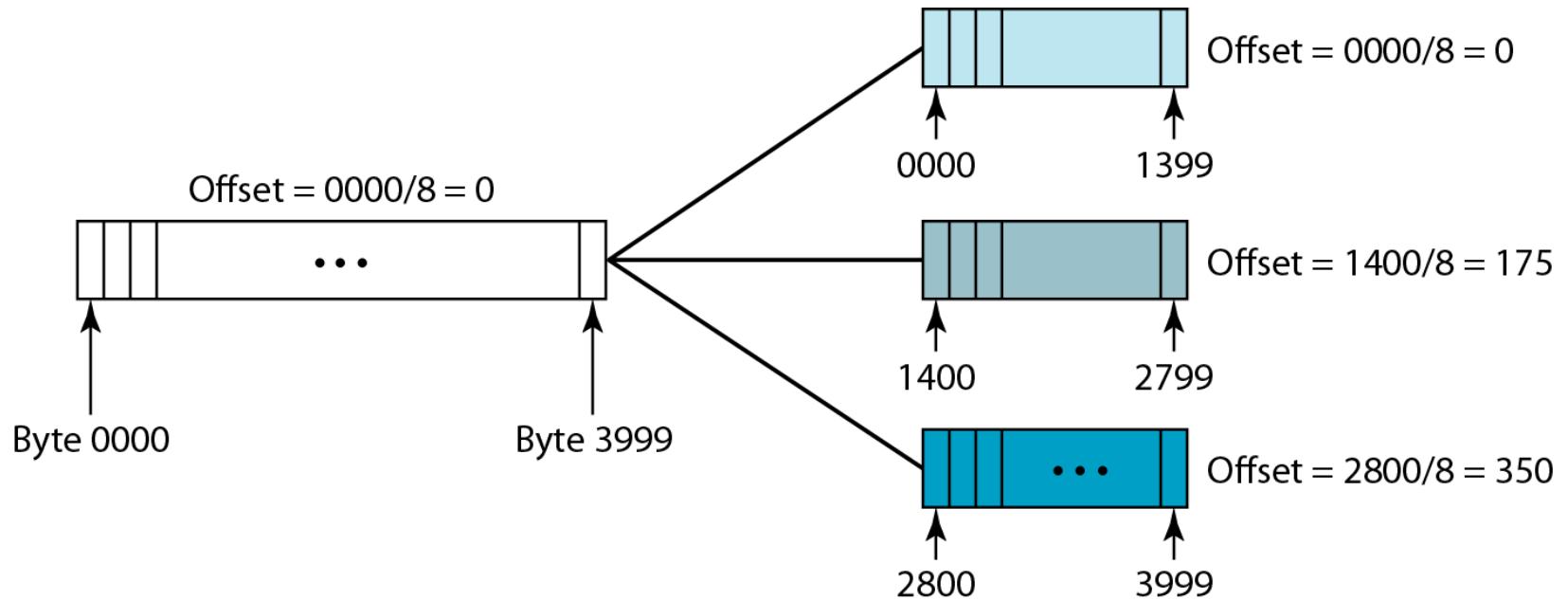
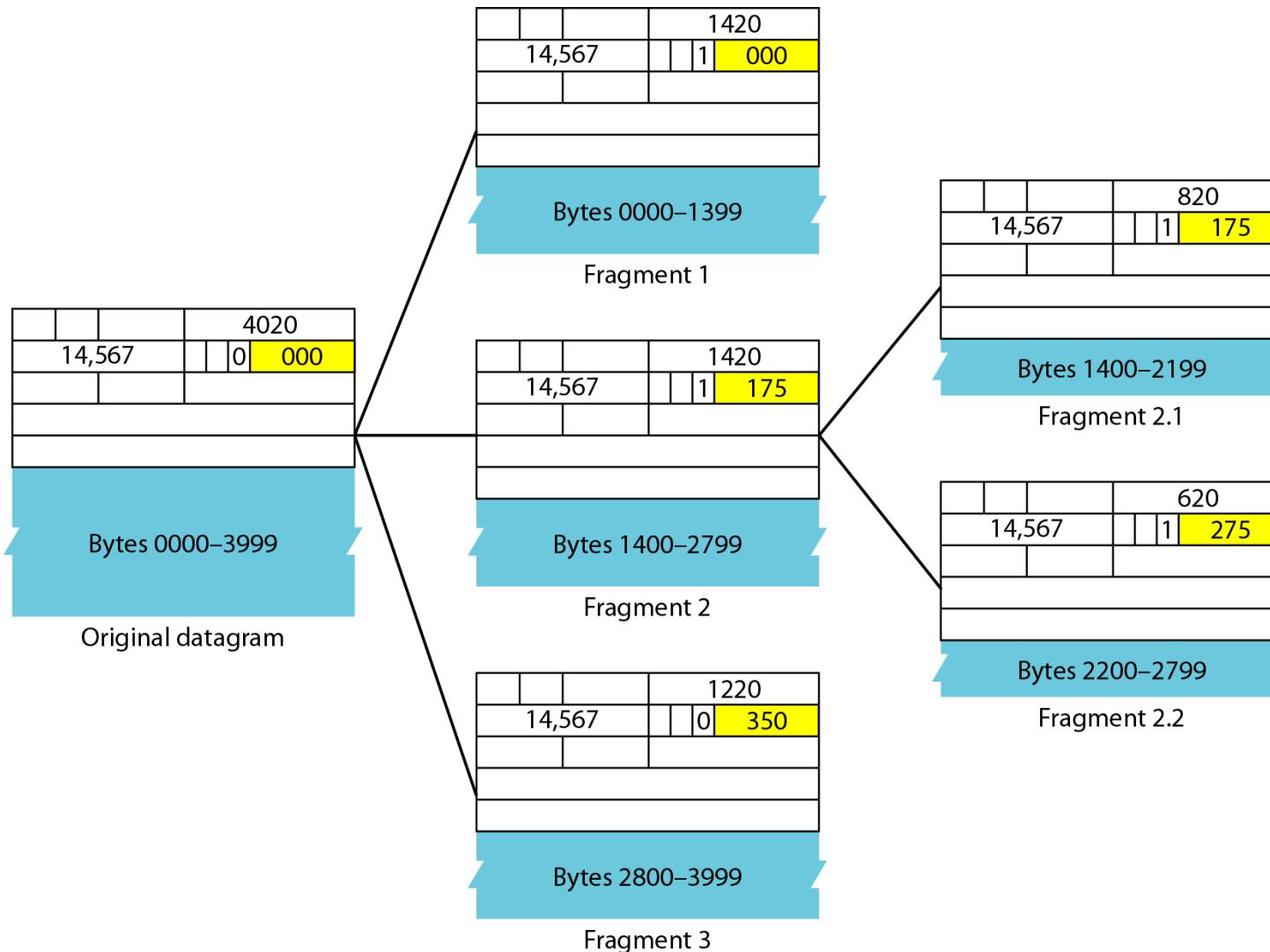
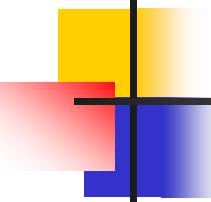


Figure 20.12 Detailed fragmentation example



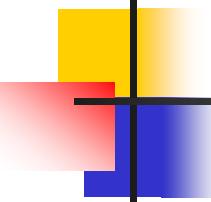


Example 20.5

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 non-fragmented packet is considered the last fragment.

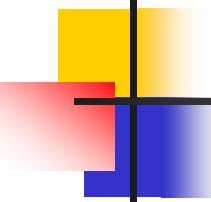


Example 20.6

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).

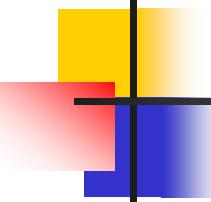


Example 20.7

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.

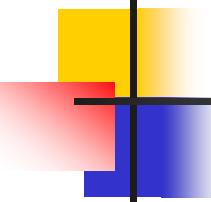


Example 20.8

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.

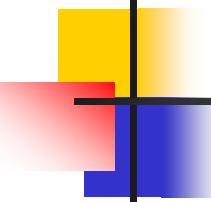


Example 20.9

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.



Example 20.10

Figure 20.13 shows an example of a checksum calculation for an IPv4 header without options. The header is divided into 16-bit sections. All the sections are added and the sum is complemented. The result is inserted in the checksum field.

Figure 20.13 Example of checksum calculation in IPv4

4	5	0	28								
1			0	0							
4	17		0								
10.12.14.5											
12.6.7.9											
4, 5, and 0	→ 4 5 0 0										
28	→ 0 0 1 C										
1	→ 0 0 0 1										
0 and 0	→ 0 0 0 0										
4 and 17	→ 0 4 1 1										
0	→ 0 0 0 0										
10.12	→ 0 A 0 C										
14.5	→ 0 E 0 5										
12.6	→ 0 C 0 6										
7.9	→ 0 7 0 9										
Sum	→ 7 4 4 E										
Checksum	→ 8 B B 1										

20-3 IPv6

IPv6 is of 128 bits(16 byte)

The network layer protocol in the TCP/IP protocol suite is currently IPv4. Although IPv4 is well designed, data communication has evolved since the inception of IPv4 in the 1970s.

IPv4 has some deficiencies that make it unsuitable for the fast-growing Internet.

Figure 20.15 IPv6 datagram header and payload

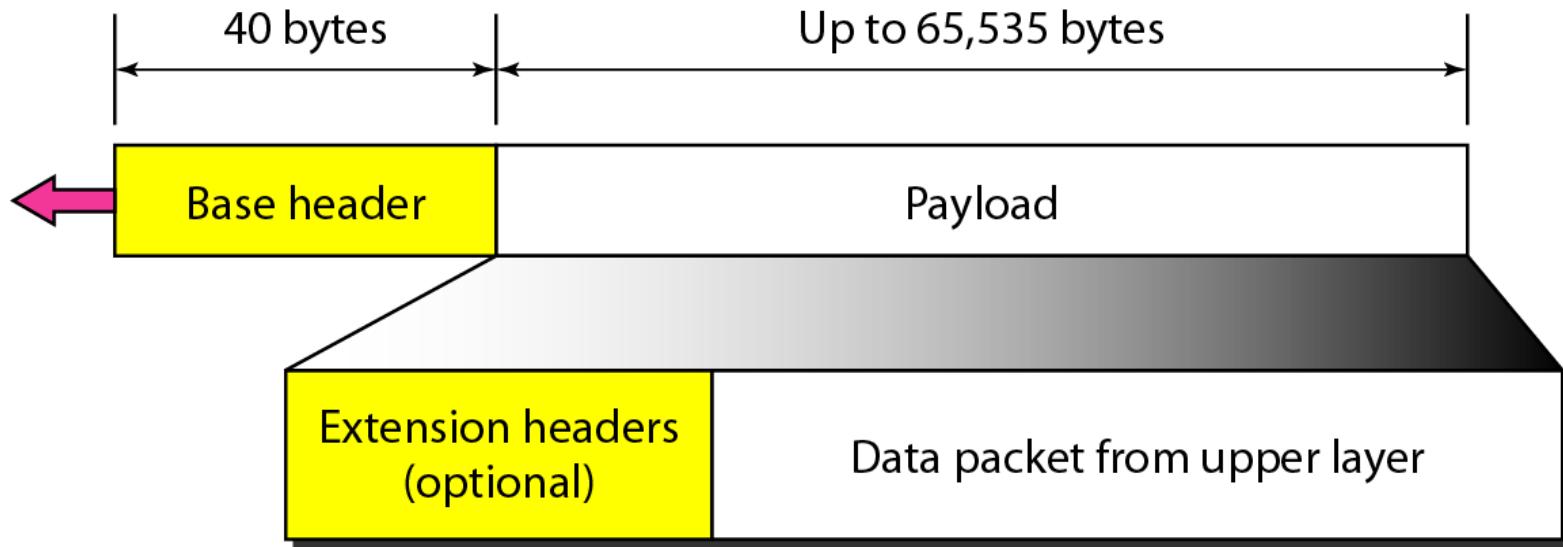
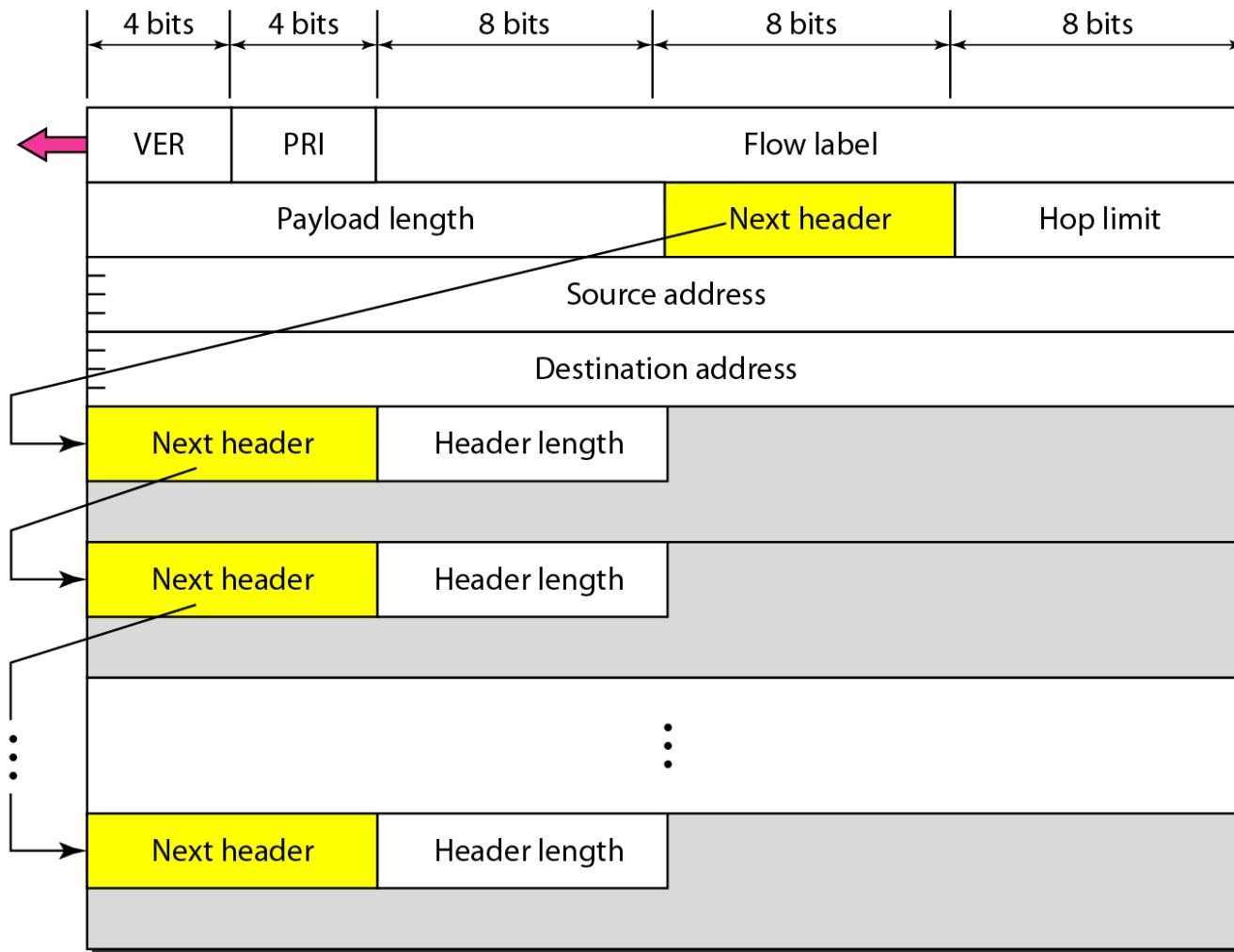


Figure 20.16 Format of an IPv6 datagram



Flow label. The flow label is a 3-byte (24-bit) field that is designed to provide special handling for a particular flow of data. We will discuss this field later.

- o **Payload length.** The 2-byte payload length field defines the length of the IP datagram excluding the base header.
- o **Next header.** The next header is an 8-bit field defining the header that follows the base header in the datagram. The next header is either one of the optional extension headers used by IP or the header of an encapsulated packet such as UDP or TCP.

Table 20.6 *Next header codes for IPv6*

<i>Code</i>	<i>Next Header</i>
0	Hop-by-hop option
2	ICMP
6	TCP
17	UDP
43	Source routing
44	Fragmentation
50	Encrypted security payload
51	Authentication
59	Null (no next header)
60	Destination option

Table 20.9 *Comparison between IPv4 and IPv6 packet headers*

<i>Comparison</i>
1. The header length field is eliminated in IPv6 because the length of the header is fixed in this version.
2. The service type field is eliminated in IPv6. The priority and flow label fields together take over the function of the service type field.
3. The total length field is eliminated in IPv6 and replaced by the payload length field.
4. The identification, flag, and offset fields are eliminated from the base header in IPv6. They are included in the fragmentation extension header.
5. The TTL field is called hop limit in IPv6.
6. The protocol field is replaced by the next header field.
7. The header checksum is eliminated because the checksum is provided by upper-layer protocols; it is therefore not needed at this level.
8. The option fields in IPv4 are implemented as extension headers in IPv6.

20-4 TRANSITION FROM IPv4 TO IPv6

Because of the huge number of systems on the Internet, the transition from IPv4 to IPv6 cannot happen suddenly. It takes 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.

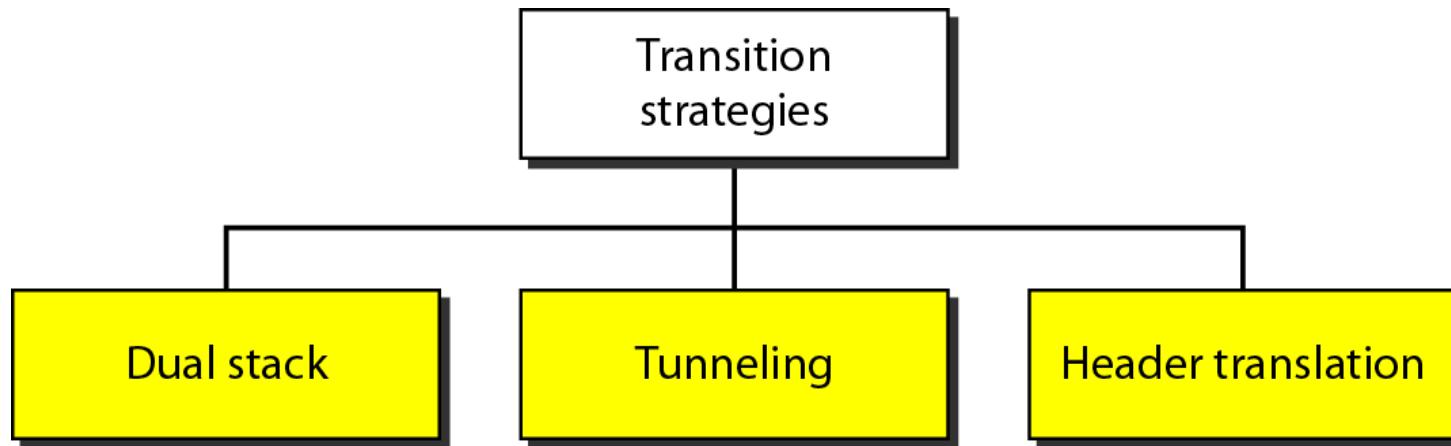
Topics discussed in this section:

Dual Stack

Tunneling

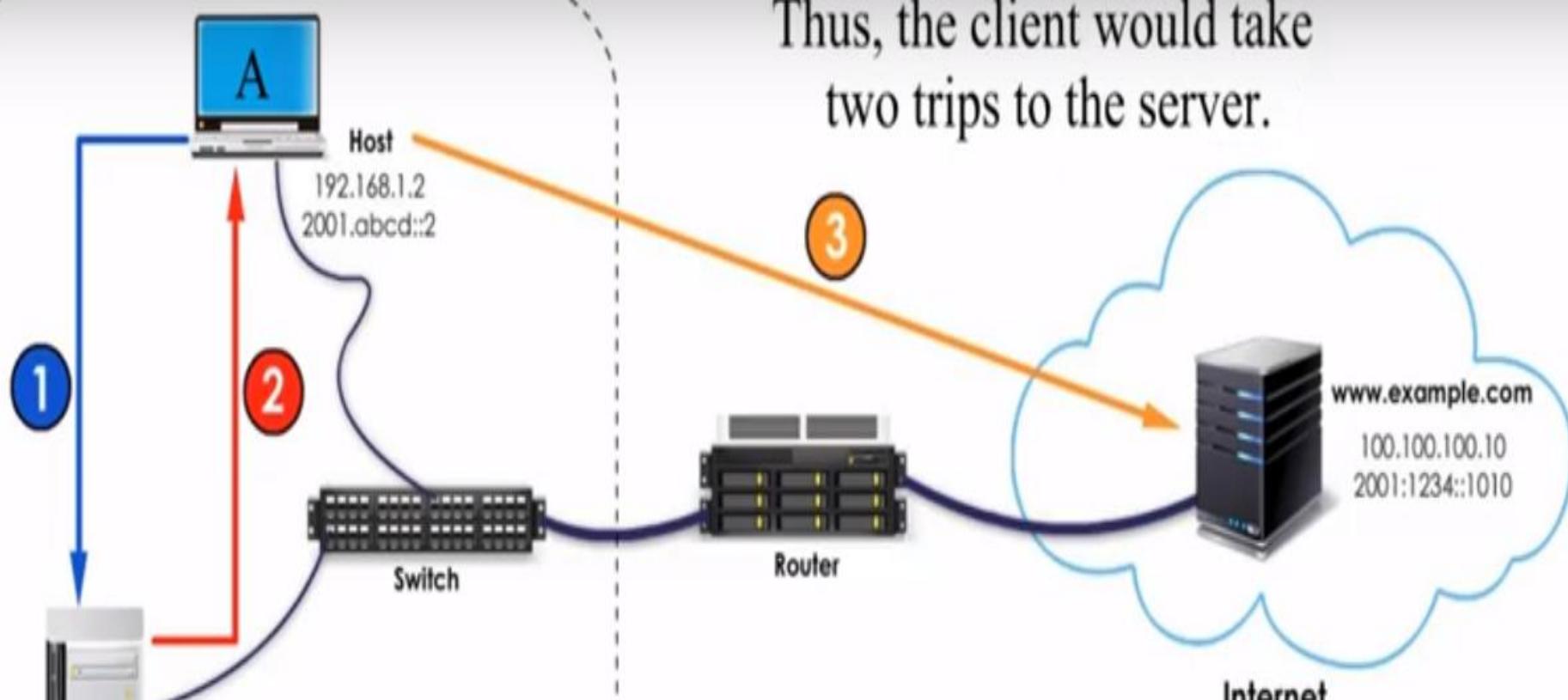
Header Translation

Figure 20.18 *Three transition strategies*



A dual-stacked device supports both IPv4 and IPv6.
Such a device can be a PC, a server, or a router.

Thus, the client would take two trips to the server.

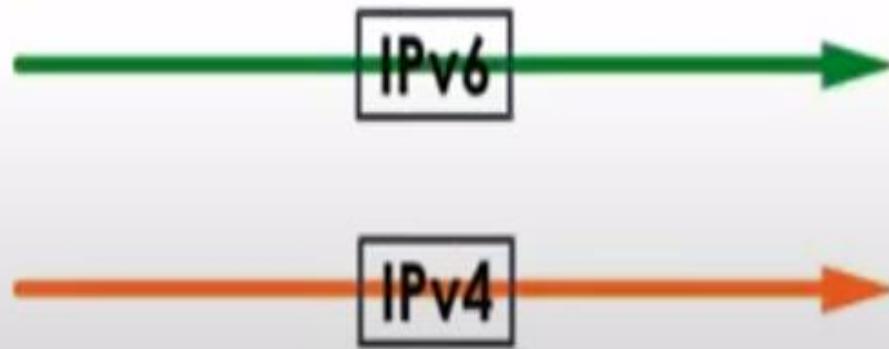


IPv4 and IPv6 Network

- 1** I need www.example.com IP address
- 2** Type AAAA record: 2001:1234::1010
Type A record: 100.100.100.10
- 3** IPv6 Session with 2001:1234::1010



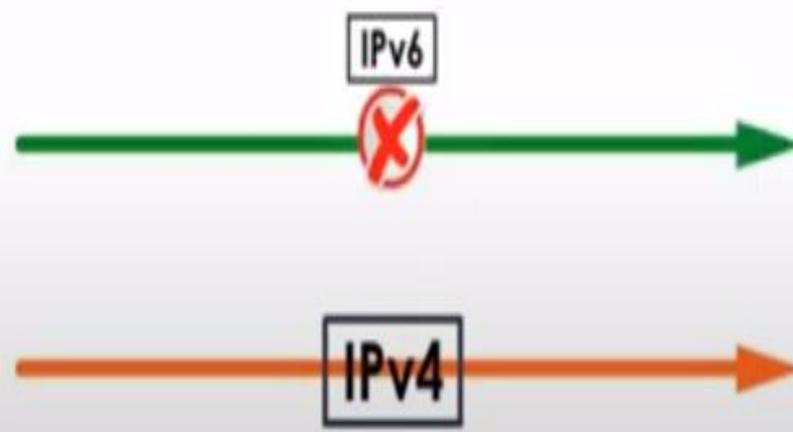
Happy Eyeballs



Internet



After 300ms



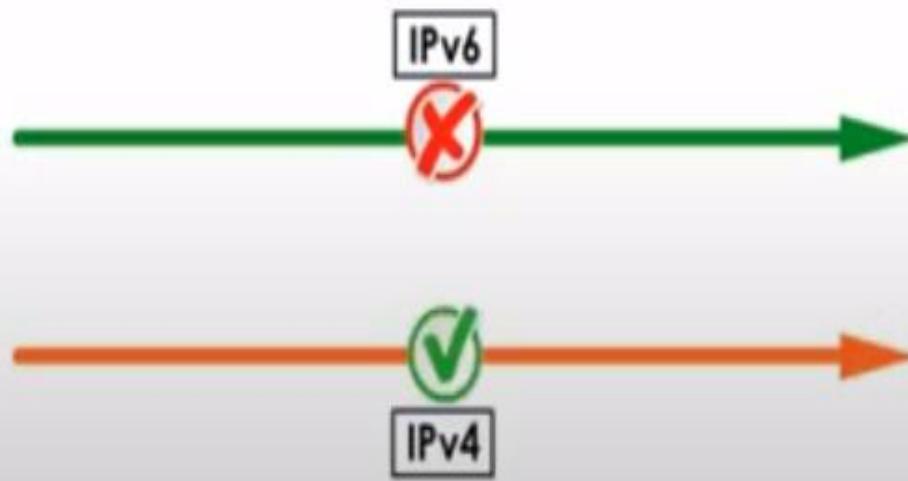
Happy Eyeballs



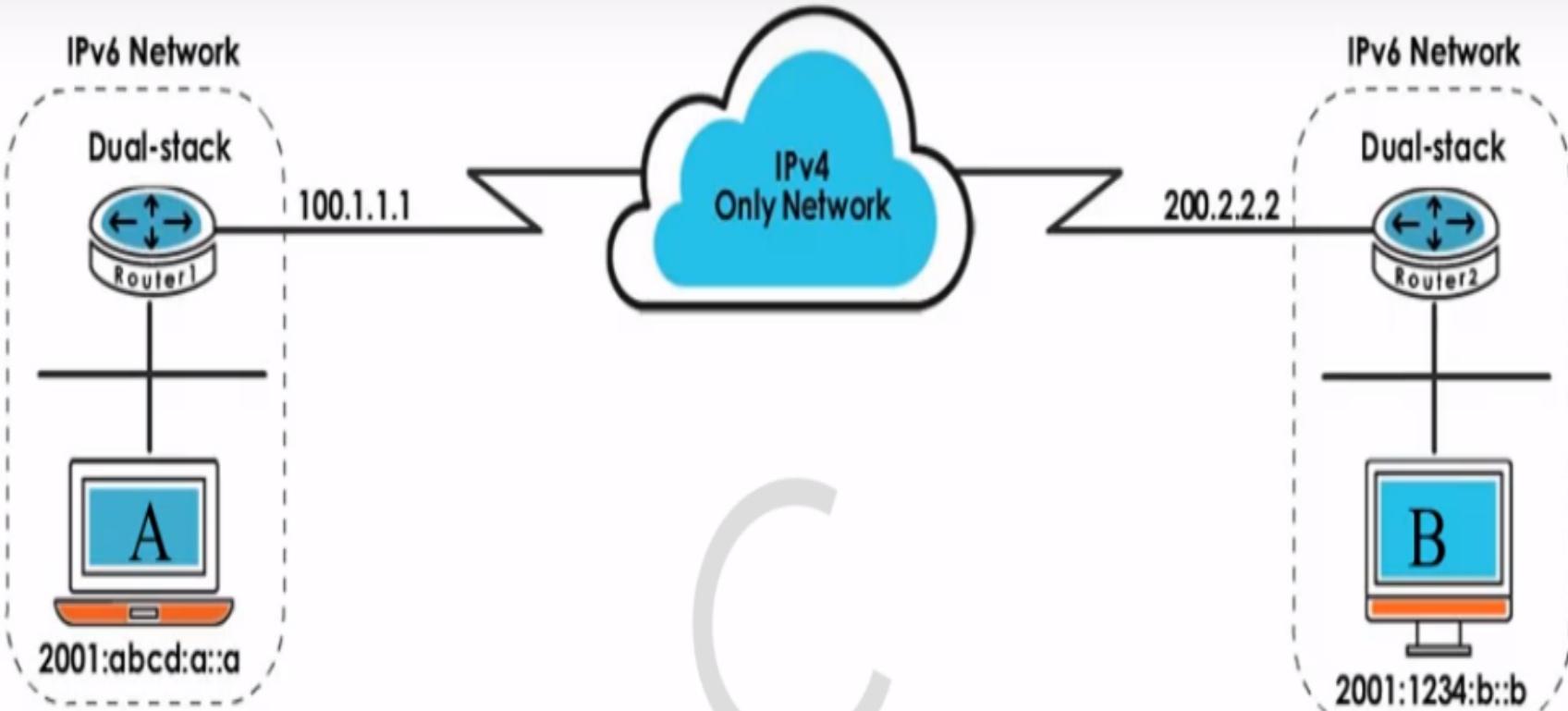
After 300ms



Happy Eyeballs



Internet



IPv6 Packet

IPv6 SRC:	IPv6 DST:
2001:abcd:a::a	2001:1234:b::b

Manual tunnel



IPv6 Packet

IPv6 SRC:	IPv6 DST:
2001:abcd:a::a	2001:1234:b::b

IPv4 SRC: 100.1.1.1
IPv4 DST: 200.2.2.2

41

IPv6 SRC: 2001:abcd:a::a
IPv6 DST: 2001:1234:b::b

Data

IPv6 Network

IPv4 Network

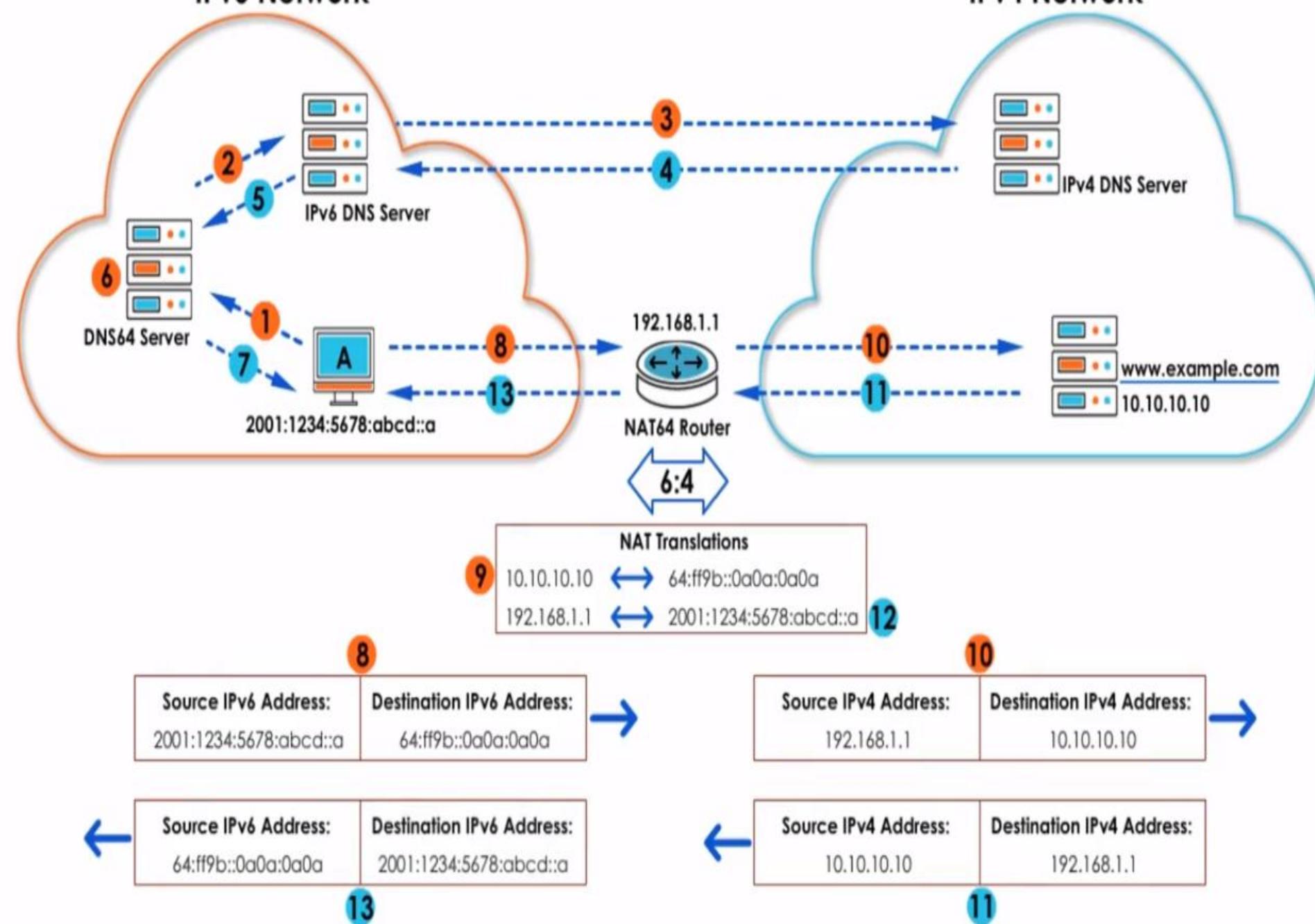


Table 20.11 *Header translation*

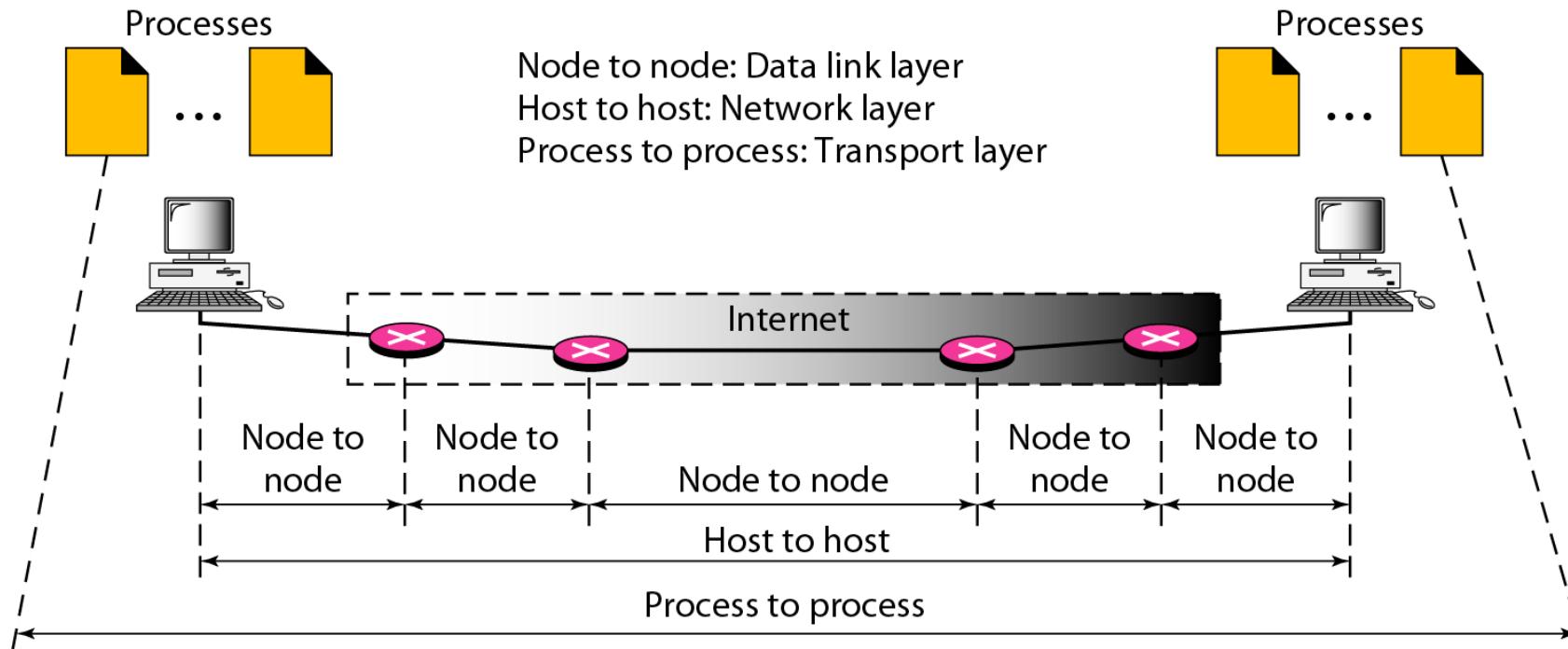
<i>Header Translation Procedure</i>
1. The IPv6 mapped address is changed to an IPv4 address by extracting the rightmost 32 bits.
2. The value of the IPv6 priority field is discarded.
3. The type of service field in IPv4 is set to zero.
4. The checksum for IPv4 is calculated and inserted in the corresponding field.
5. The IPv6 flow label is ignored.
6. Compatible extension headers are converted to options and inserted in the IPv4 header. Some may have to be dropped.
7. The length of IPv4 header is calculated and inserted into the corresponding field.
8. The total length of the IPv4 packet is calculated and inserted in the corresponding field.

TRANSPORT LAYER

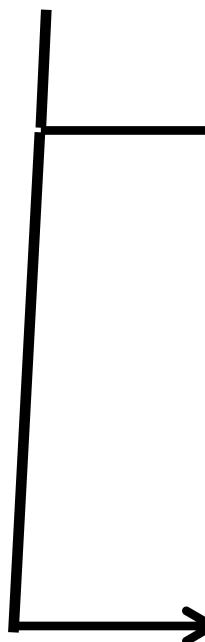
INTRODUCTION

- The transport layer is responsible for message delivery from process running in source computer to process running destination computer (PROCESS TO PROCESS DELIVERY))
- It provides *logical communication* between app processes running on different hosts
- transport protocols run in end systems
 - Send side: breaks app messages into **segments**, passes to network layer
 - Recieve side: reassembles segments into messages, passes to app layer
- more than one transport protocol available to apps
 - Internet: TCP and UDP

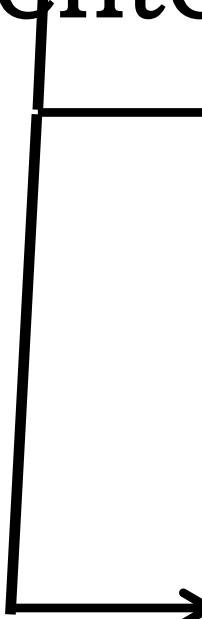
Types of data deliveries: Internet Stack



TRANSPORT LAYER SERVICES

- 
- Connectionless Vs Connection Oriented Service
 - Reliable Vs Unreliable Service

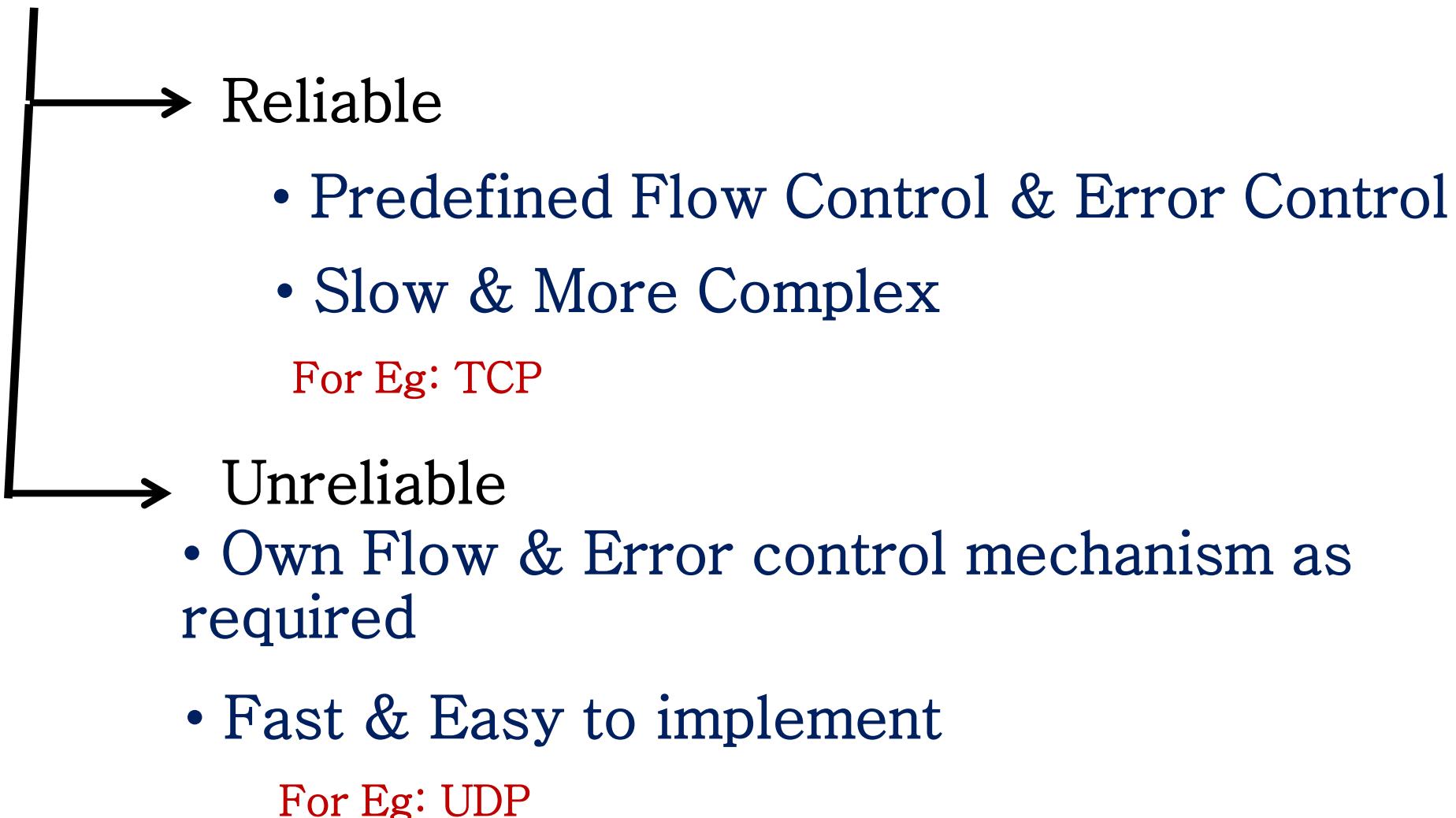
Connectionless Vs Connection Oriented Service

- 
- Connectionless Service
 - Without Connection Establishment or Connection Release
 - Packets may be lost or delay
 - No Acknowledgement

For Eg: UDP
 - Connection Oriented Service
 - Connection Establishment or Connection Release
 - Packets neither be lost
 - Acknowledgement

For Eg: TCP

Reliable Vs Unreliable Service



TRANSPORT LAYER DESIGN ISSUE

- Connection Management
- Addressing
- Data Transfer
- Multiplexing & Demultiplexing
- Flow Control
- Error Control
- Congestion Control

1. Connection Management: Establishing, Maintaining & Releasing Connection:

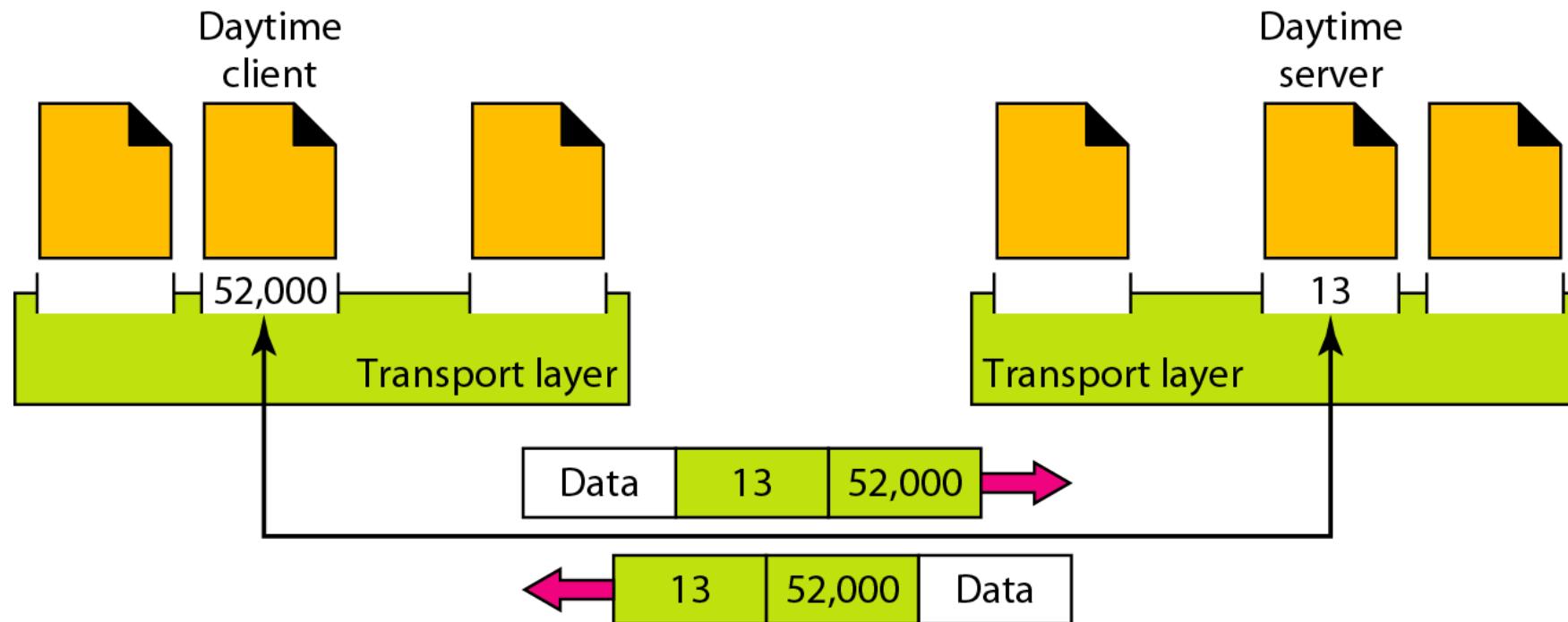
The transport layer establishes, maintains & releases end-to-end transport connection on the request of upper layers. Establishing a connection involves allocation of buffers for storing user data, synchronizing the sequence numbers of packets etc. A connection is released at the request of upper layer.

2. Addressing: In order to deliver the message from one process to another, an addressing scheme is required. Several processes may be running on a system at a time. In order to identify the correct process out of the various running processes, transport layer uses an addressing scheme involving the concept of port numbers. Each process has a specific port number.

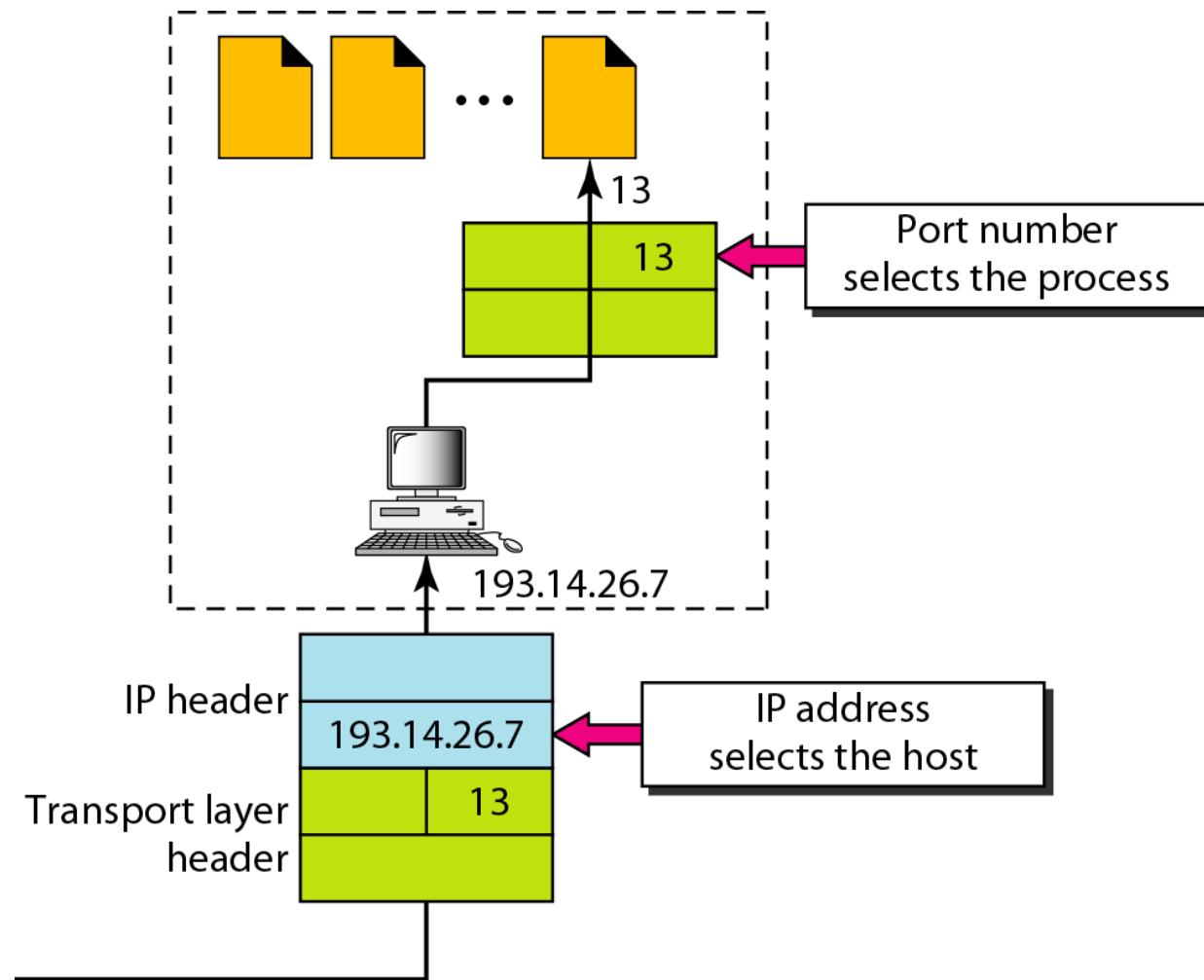
In order to provide communication between two different processes on different networks, both IP address and port number, i.e. socket address is required. Socket address is a combination of IP address and port number.

Addressing

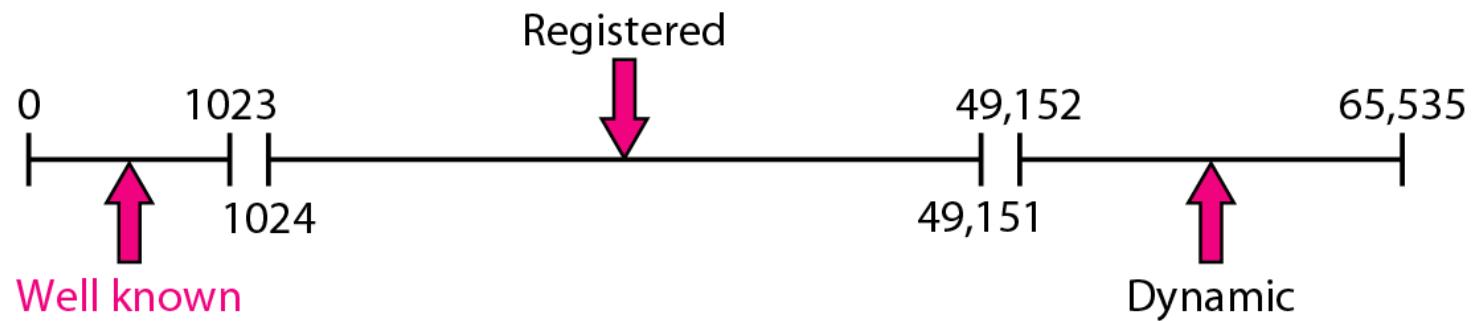
- Using Port address (16 bit)
- Ranges from 0 to 65,535



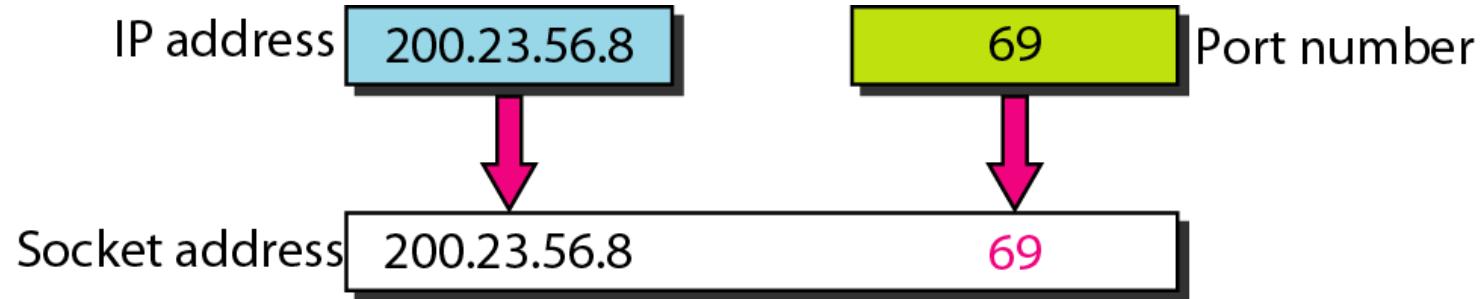
IP addresses versus port numbers



IANA ranges



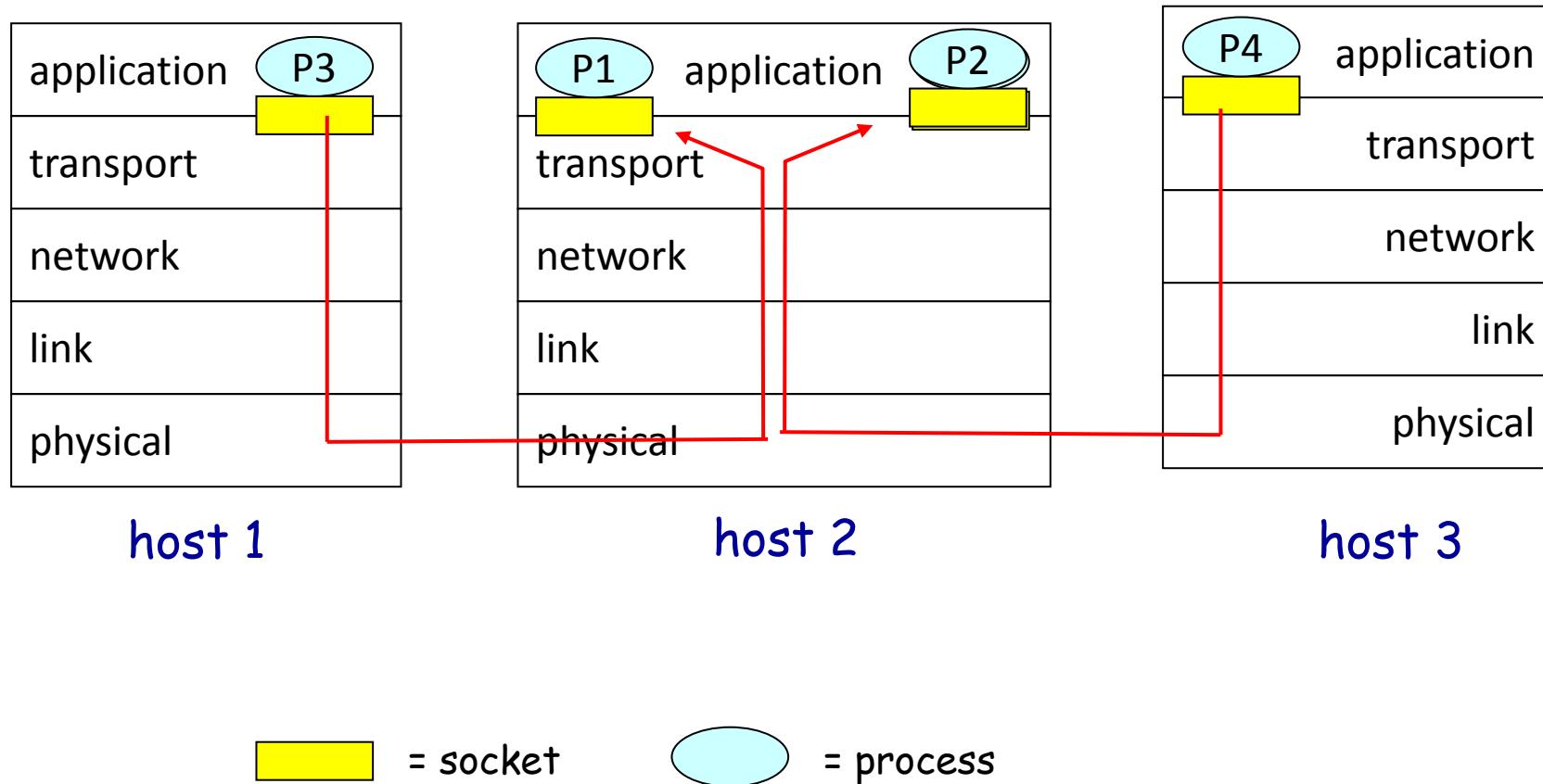
Socket address (IP address + Port Address)



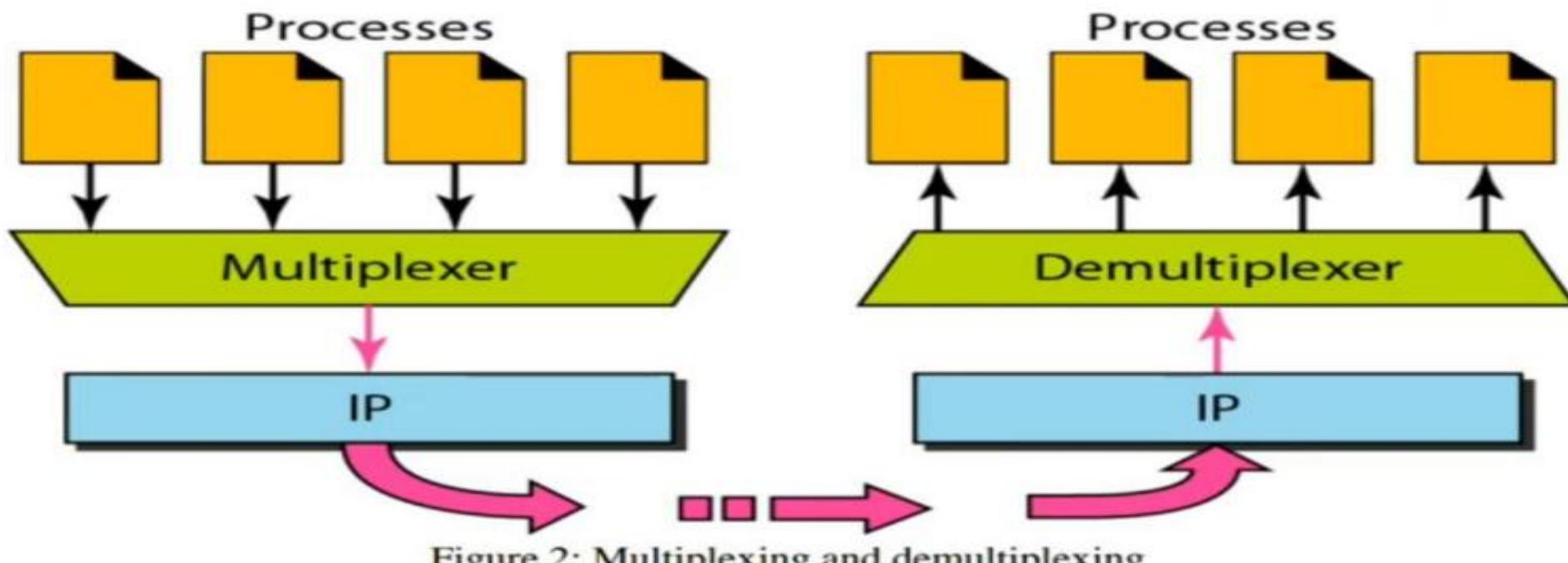
What is the use of socket?

- The socket mechanism provides a means of inter-process communication (IPC).
- Socket is basically an API for enabling communication between two end points.
- A **socket** is one endpoint of a **two way** communication link between two programs running on the network.

Socket API



3. **Data Transfer:** Transport layer breaks user data into smaller units and attaches a transport layer header to each unit forming a TPDU (Transport Layer Data Unit). The TPDU is handed over to the network layer for its delivery to destination. The TPDU header contains port number, sequence number, acknowledgement number, checksum and other fields.
4. **Multiplexing and Demultiplexing:** The addressing mechanism allows multiplexing and demultiplexing by the transport layer, as shown in Figure 2



Multiplexing/demultiplexing

Demultiplexing at rcv host:

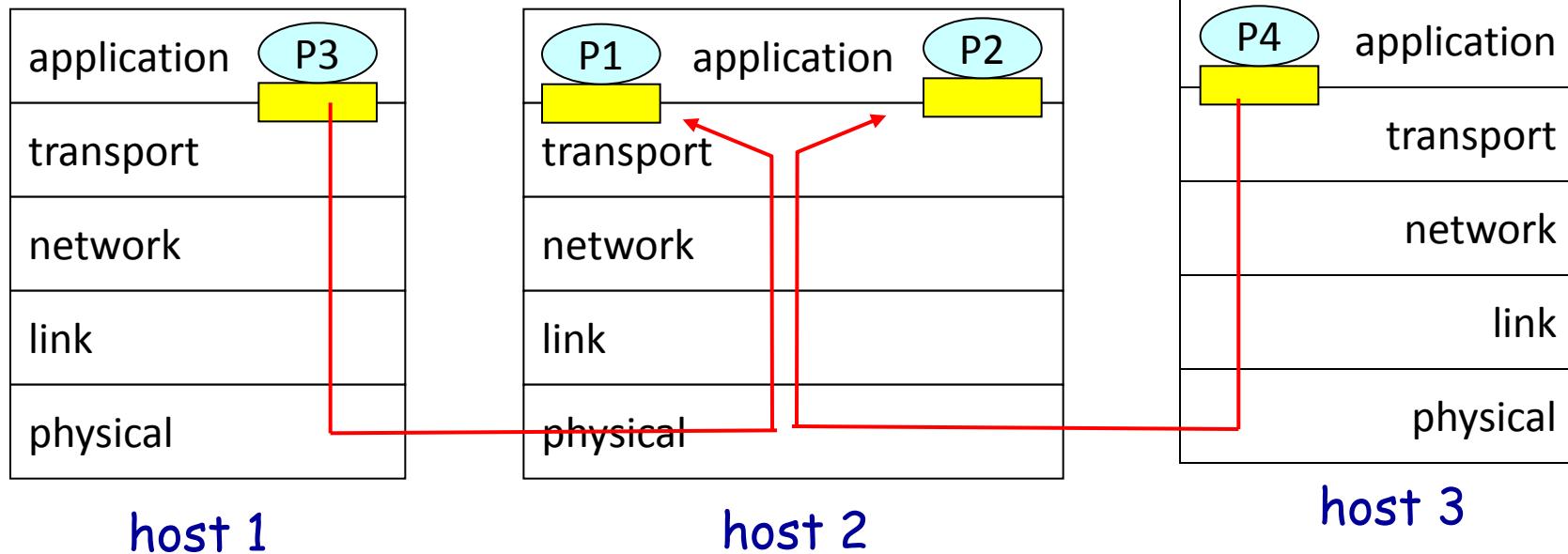
delivering received segments
to correct socket

= socket

= process

Multiplexing at send host:

gathering data from multiple
sockets, enveloping data with
header (later used for
demultiplexing)



5. **Flow Control:** Like data link layer, transport layer also performs flow control. However, flow control at transport layer is performed end-to-end rather than node-to-node. Transport Layer uses a sliding window protocol to perform flow control.

6. **Error Control:** Transport layer also provides end-to-end error control facility. Transport layer deals with several different types of errors:
 - Error due to damaged bits.
 - Error due to non delivery of TPDUs.
 - Error due to duplicate delivery of TPDUs.
 - Error due to delivery of TPDU to a wrong destination.

7. **Congestion Control:** Transport layer also handles congestion in the networks. Several different congestion control algorithms are used to avoid congestion.

Transport Layer Protocols (UDP and TCP)

- UDP provides
 - best effort delivery
 - Connectionless
 - Unreliable
 - Out of order delivery
- TCP
 - Reliable
 - In-order delivery
 - Congestion control
 - Flow control
 - Connection setup

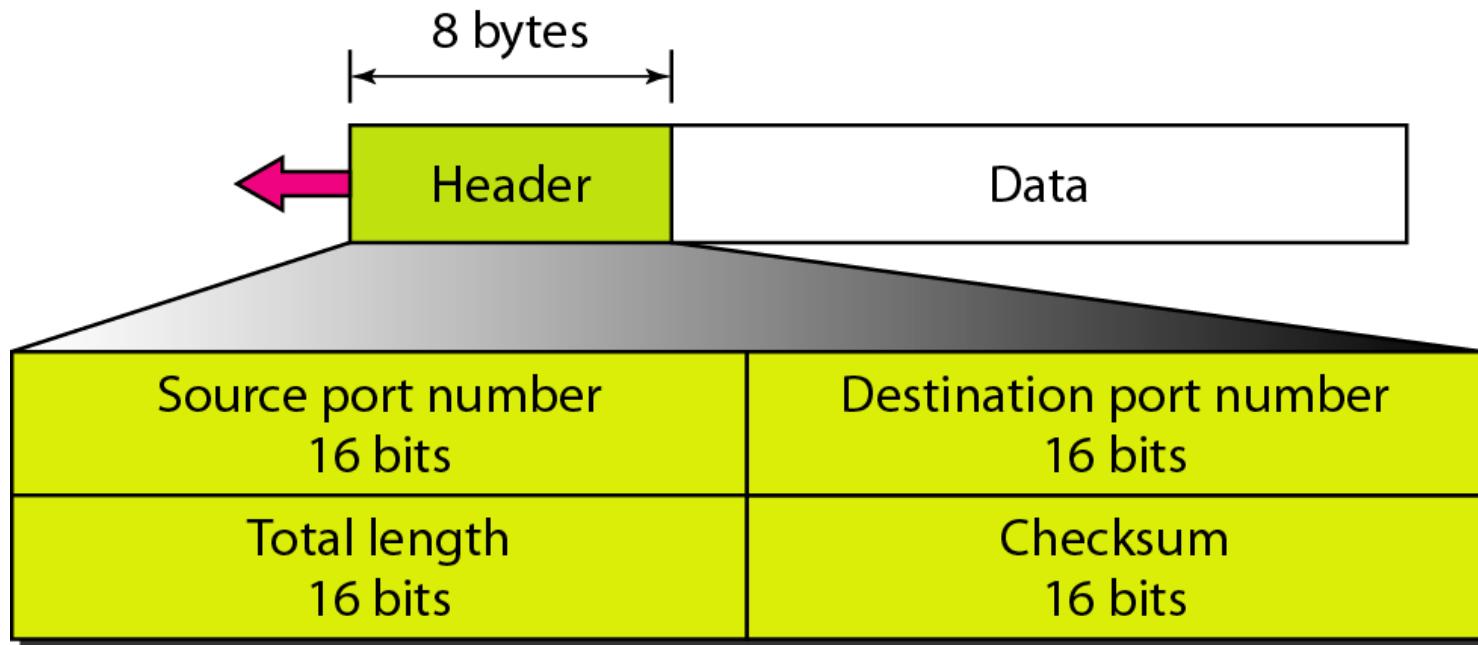
USER DATAGRAM PROTOCOL (UDP)

The User Datagram Protocol (UDP) is called a connectionless, unreliable transport protocol. It does not add anything to the services of IP except to provide process-to-process communication instead of host-to-host communication.

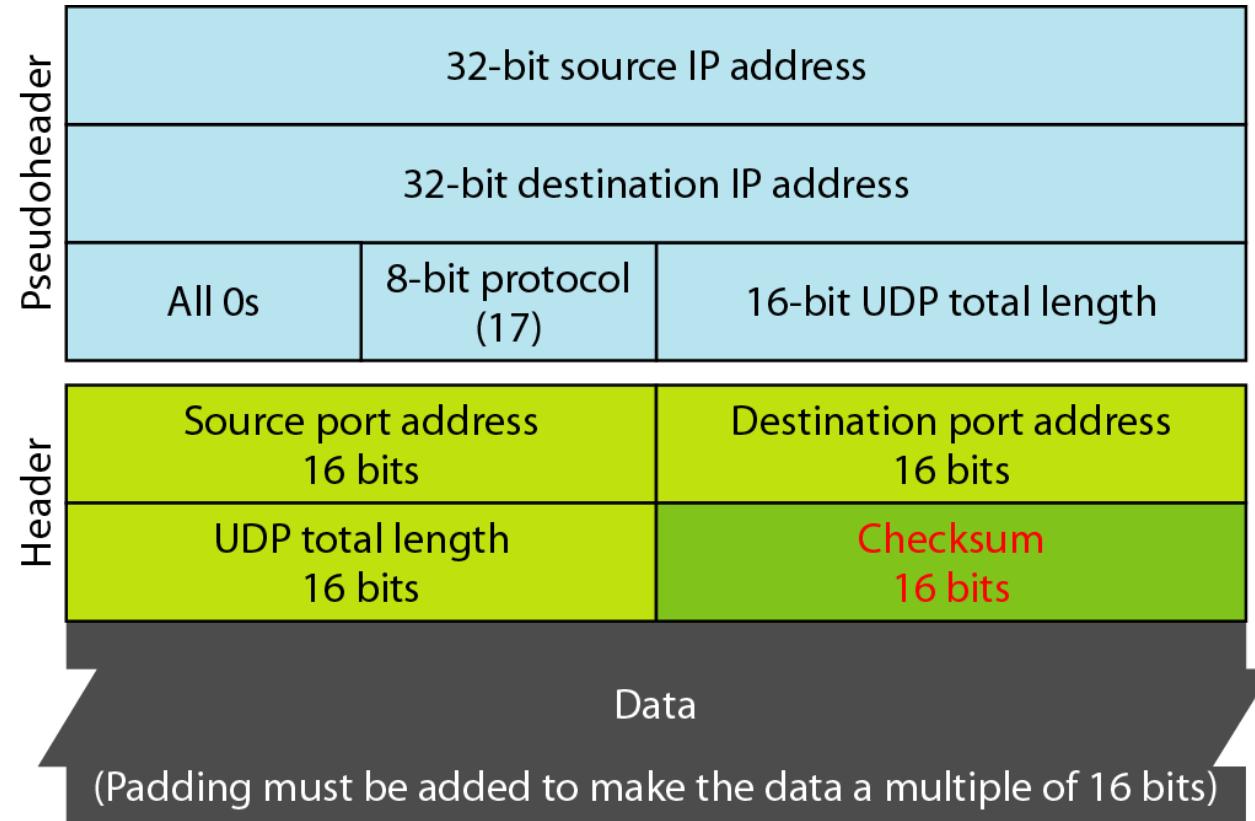
Well-known ports used with UDP

<i>Port</i>	<i>Protocol</i>	<i>Description</i>
7	Echo	Echoes a received datagram back to the sender
9	Discard	Discards any datagram that is received
11	Users	Active users
13	Daytime	Returns the date and the time
17	Quote	Returns a quote of the day
19	Chargen	Returns a string of characters
53	Nameserver	Domain Name Service
67	BOOTPs	Server port to download bootstrap information
68	BOOTPc	Client port to download bootstrap information
69	TFTP	Trivial File Transfer Protocol
111	RPC	Remote Procedure Call
123	NTP	Network Time Protocol
161	SNMP	Simple Network Management Protocol
162	SNMP	Simple Network Management Protocol (trap)

User datagram format



Pseudoheader for checksum calculation



TCP

TCP is a connection-oriented protocol; it creates a virtual connection between two TCPs to send data. In addition, TCP uses flow and error control mechanisms at the transport level.

Stream delivery

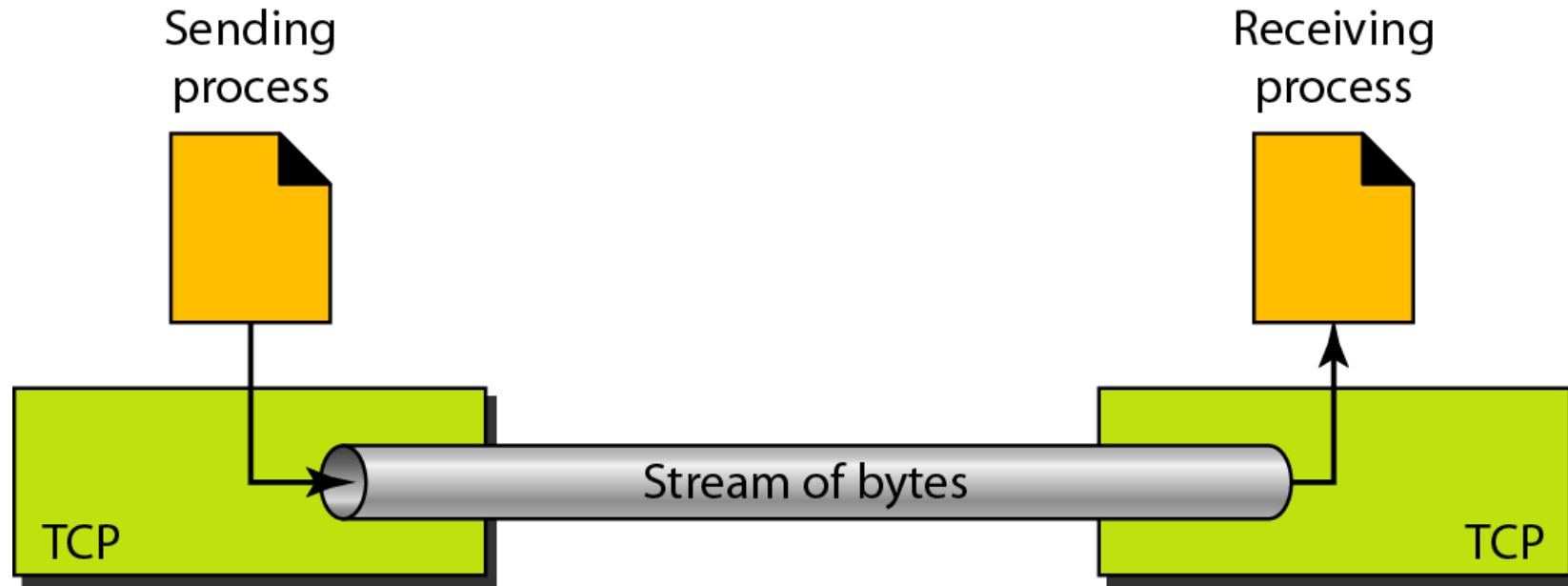
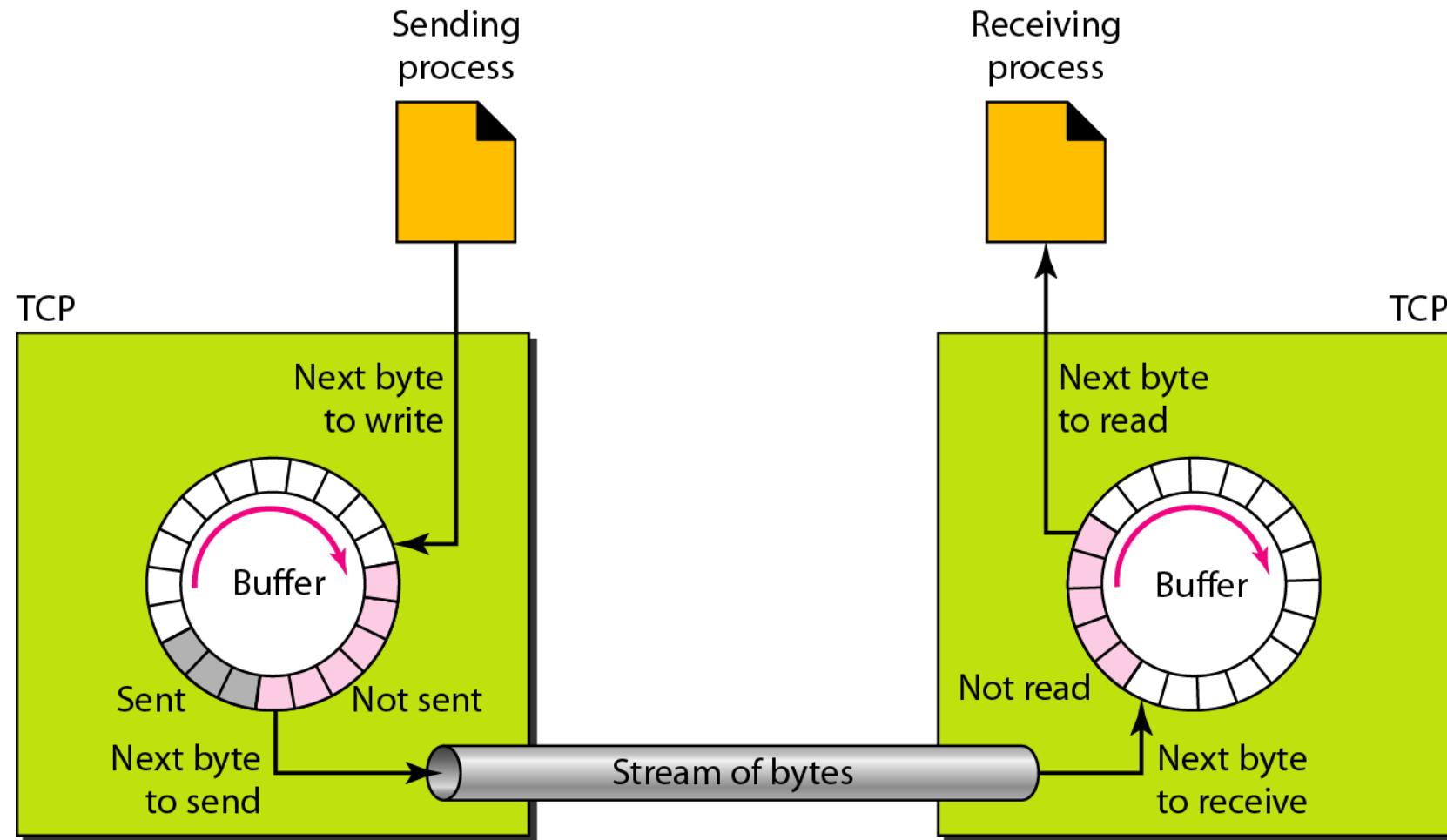
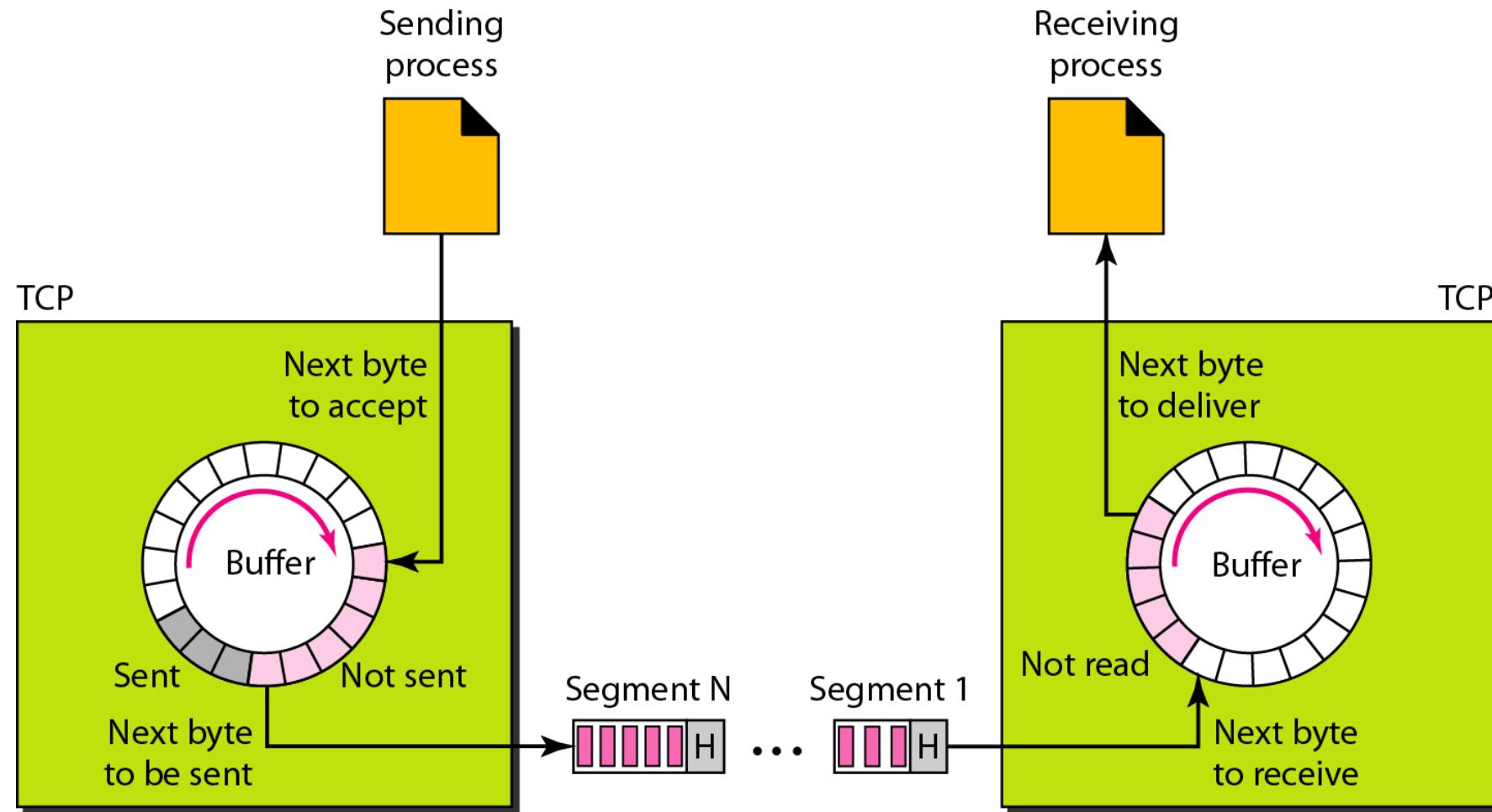
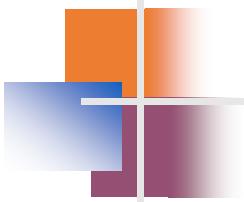


Figure *Sending and receiving buffers*



TCP segments





Note

The bytes of data being transferred in each connection are numbered by TCP.

The numbering starts with an arbitrarily generated number. Uses 32 bits.

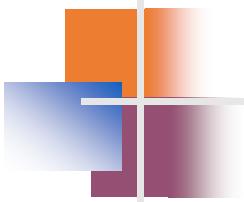
Example 15.1

Suppose a TCP connection is transferring a file of 5,000 bytes. The first byte is numbered 10,001. What are the sequence numbers for each segment if data are sent in five segments, each carrying 1,000 bytes?

Solution

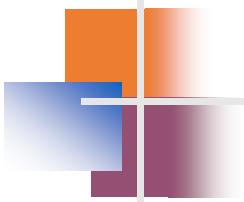
The following shows the sequence number for each segment:

Segment 1	→	Sequence Number:	10,001	Range:	10,001	to	11,000
Segment 2	→	Sequence Number:	11,001	Range:	11,001	to	12,000
Segment 3	→	Sequence Number:	12,001	Range:	12,001	to	13,000
Segment 4	→	Sequence Number:	13,001	Range:	13,001	to	14,000
Segment 5	→	Sequence Number:	14,001	Range:	14,001	to	15,000



Note

The value in the sequence number field of a segment defines the number assigned to the first data byte contained in that segment.



Note

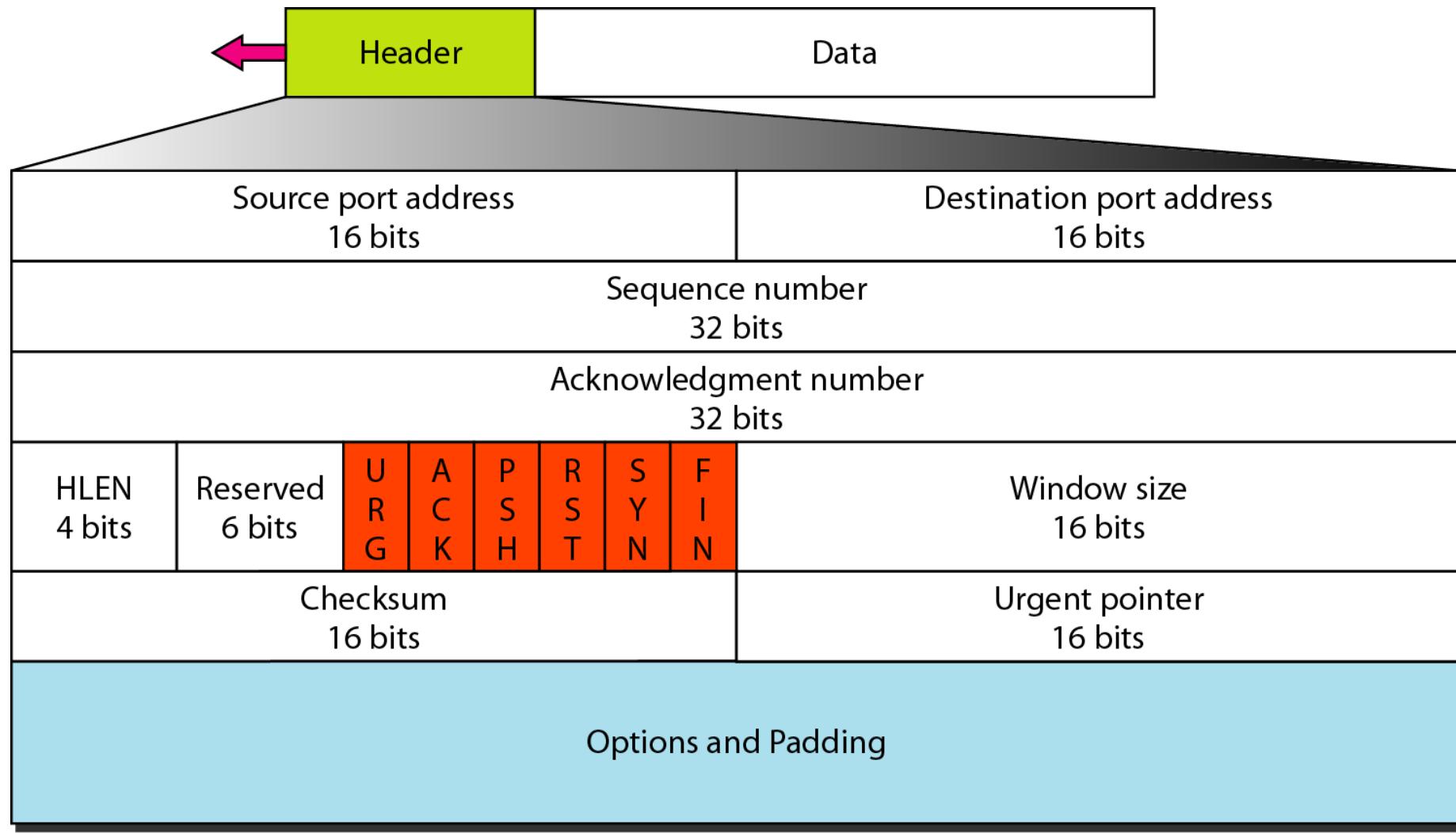
The value of the acknowledgment field in a segment defines the number of the next byte a party expects to receive.

The acknowledgment number is cumulative.

15-3 SEGMENT

Before discussing TCP in more detail, let us discuss the TCP packets themselves. A packet in TCP is called a segment.

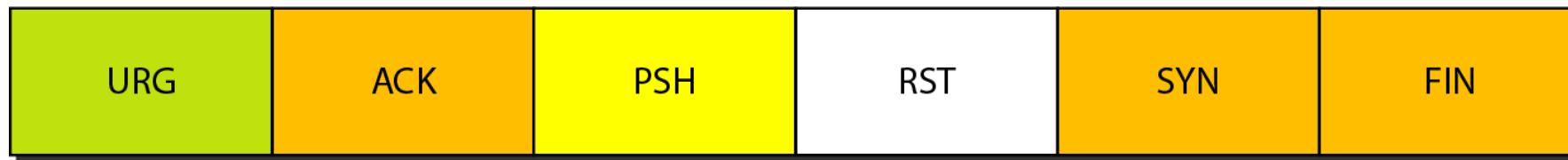
TCP segment format



Control field

URG: Urgent pointer is valid
ACK: Acknowledgment is valid
PSH: Request for push

RST: Reset the connection
SYN: Synchronize sequence numbers
FIN: Terminate the connection



Description of flags in the control field

<i>Flag</i>	<i>Description</i>
URG	The value of the urgent pointer field is valid.
ACK	The value of the acknowledgment field is valid.
PSH	Push the data.
RST	Reset the connection.
SYN	Synchronize sequence numbers during connection.
FIN	Terminate the connection.

15-4 A TCP CONNECTION

TCP is connection-oriented. It establishes a virtual path between the source and destination. All of the segments belonging to a message are then sent over this virtual path. You may wonder how TCP, which uses the services of IP, a connectionless protocol, can be connection-oriented. The point is that a TCP connection is virtual, not physical. TCP operates at a higher level. TCP uses the services of IP to deliver individual segments to the receiver, but it controls the connection itself. If a segment is lost or corrupted, it is retransmitted.

Figure 15.9 Connection establishment using three-way handshake

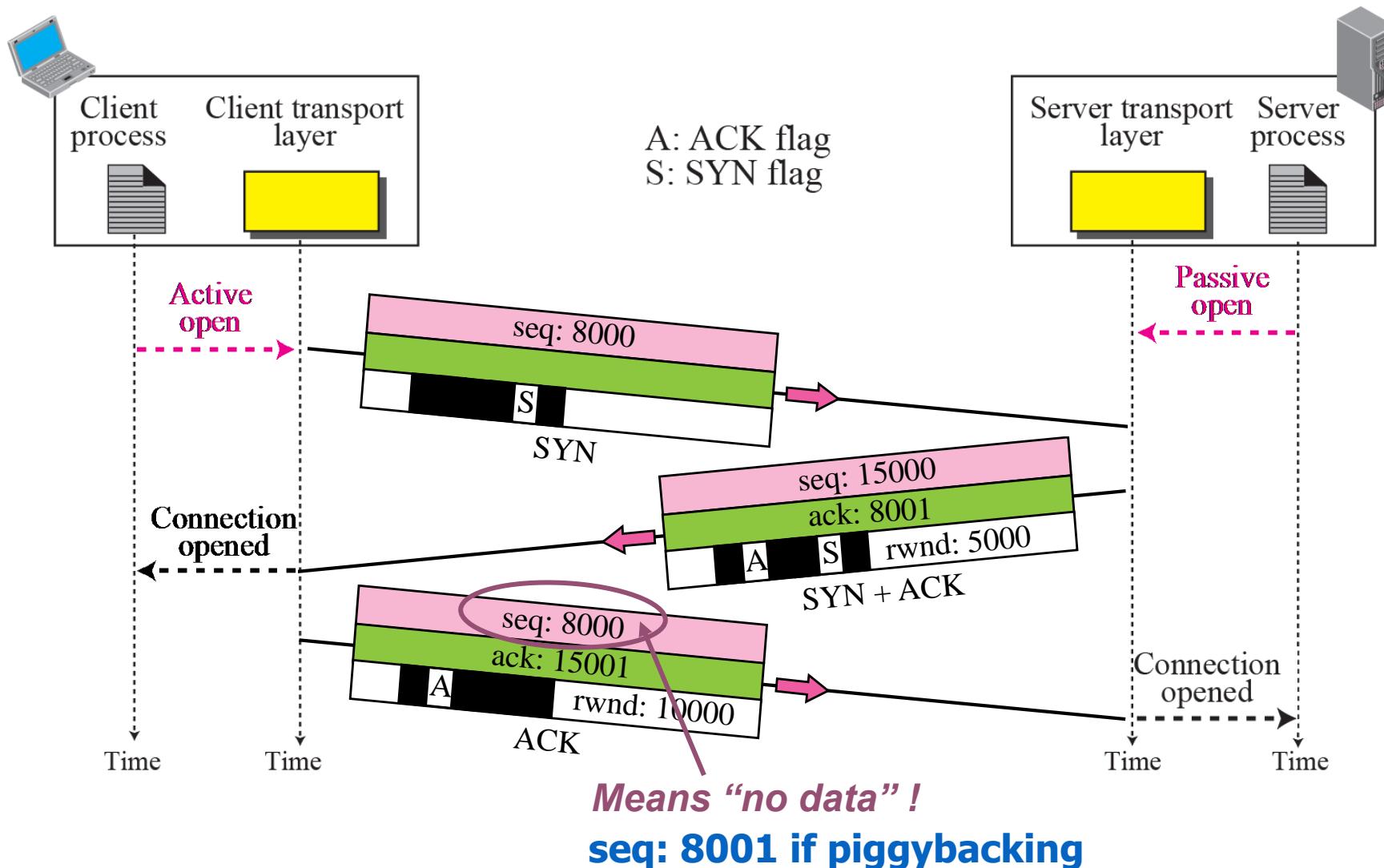
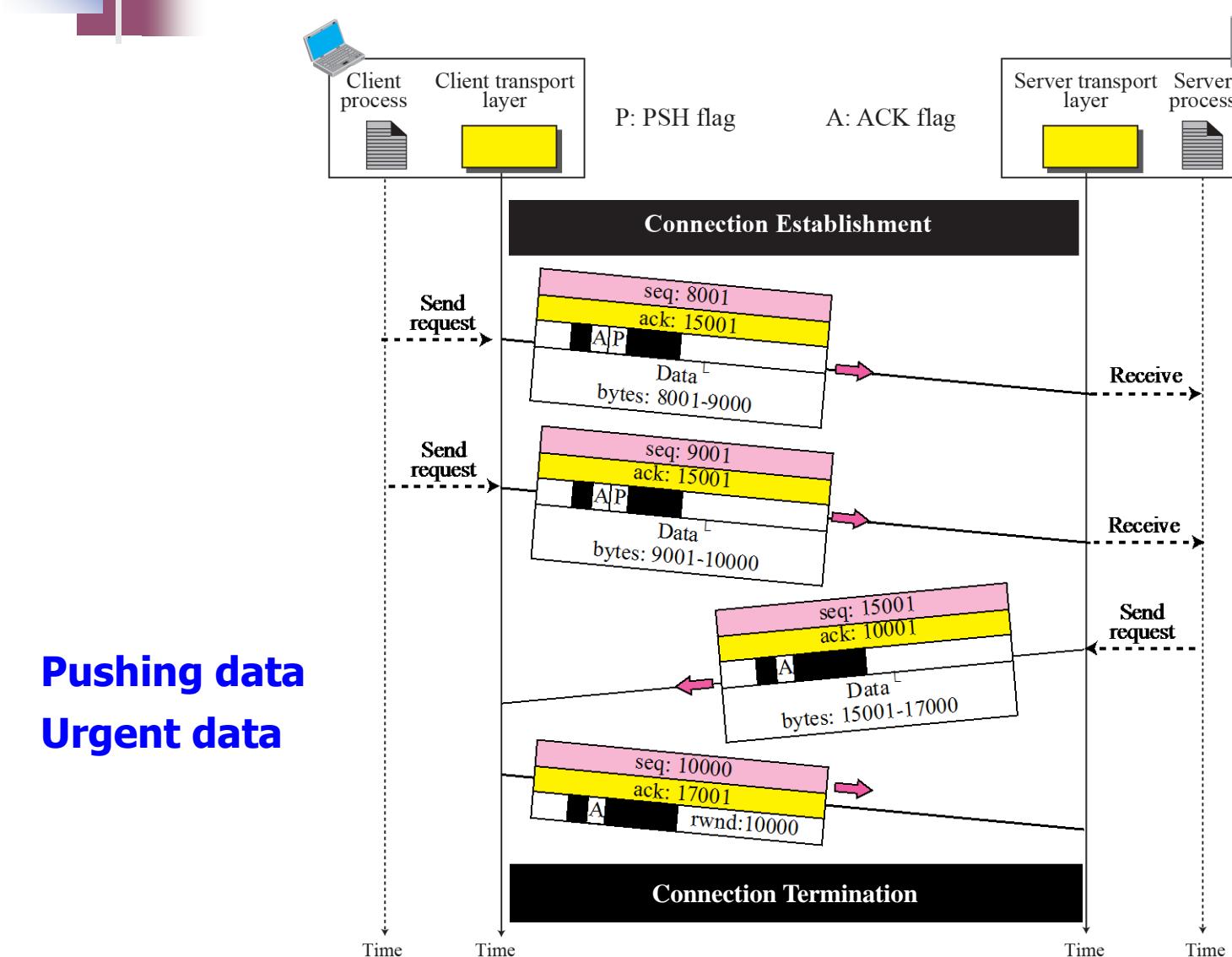
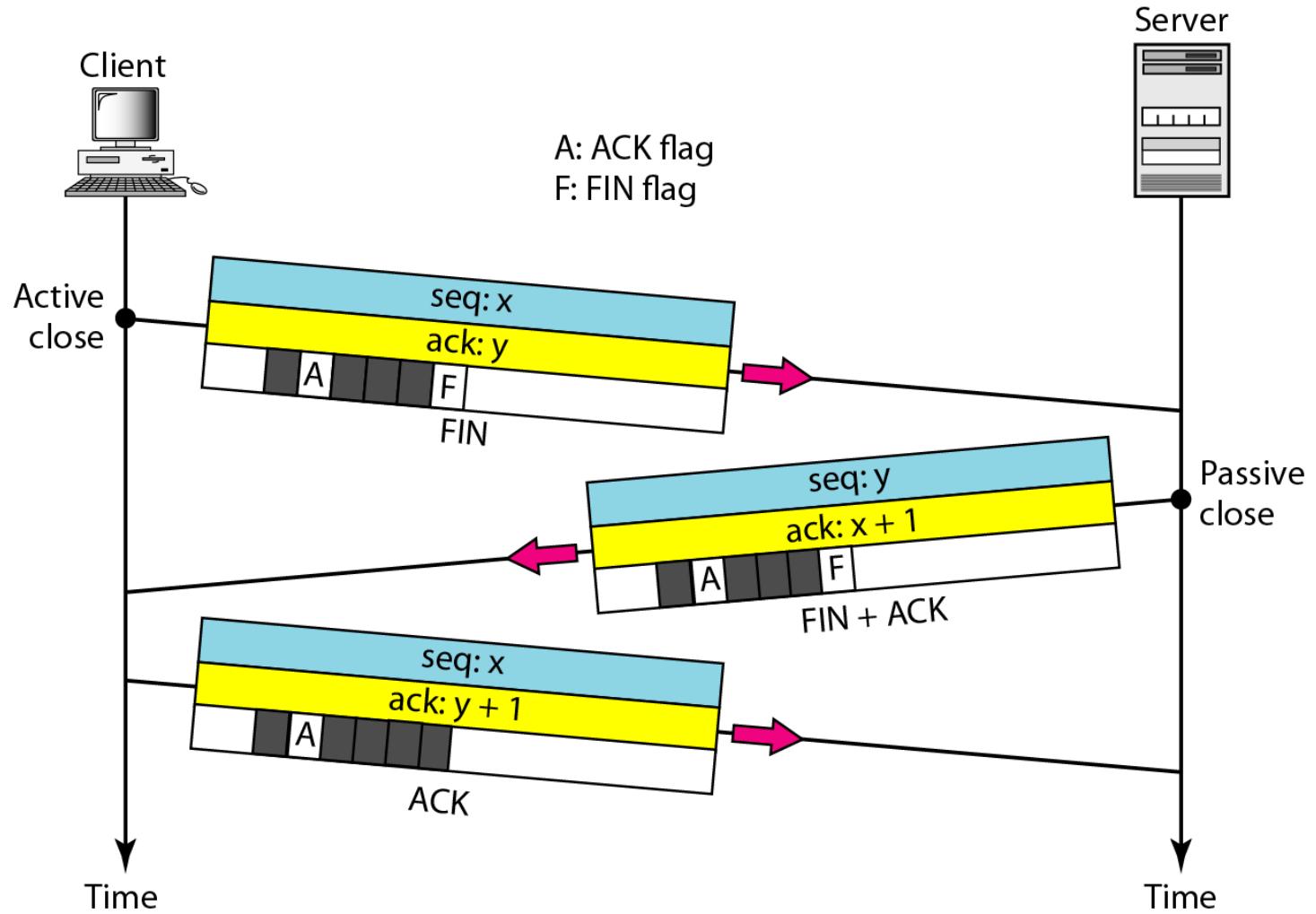


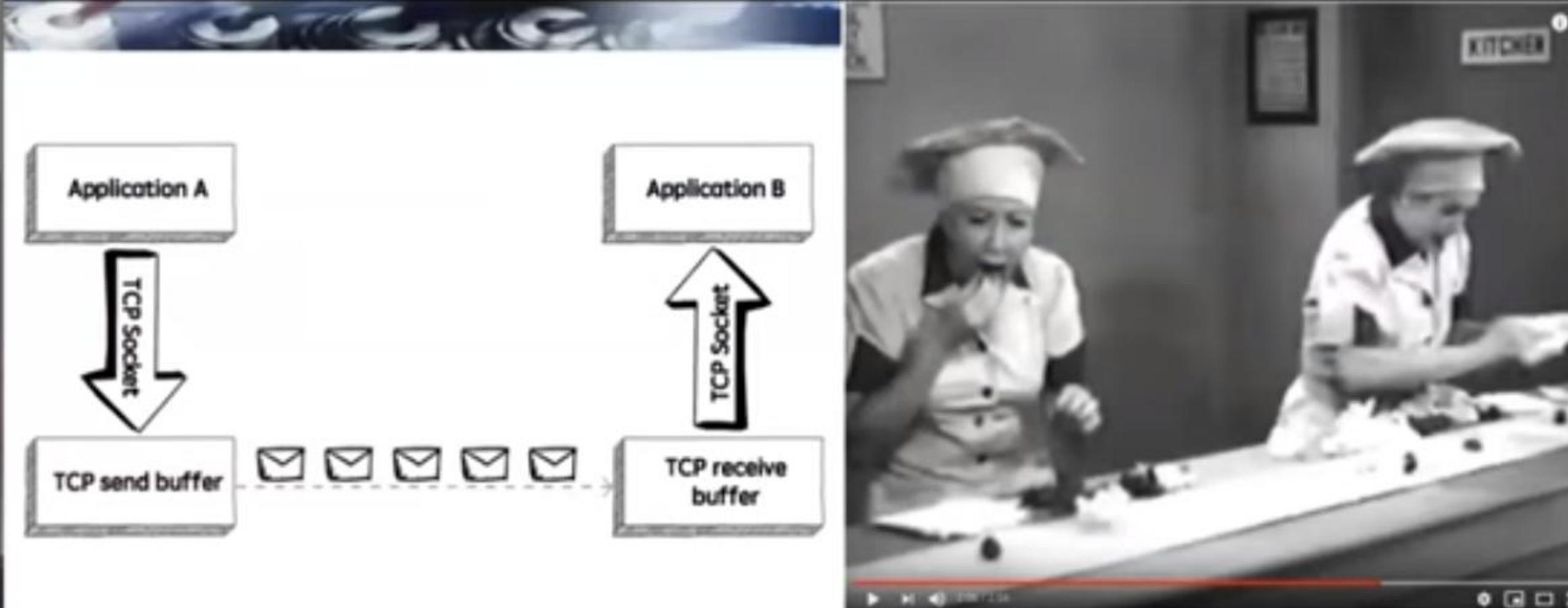
Figure 15.10 Data Transfer



Connection termination using three-way handshaking



TCP Flow Control & Window Size



Lucy and the Chocolate Factory high res

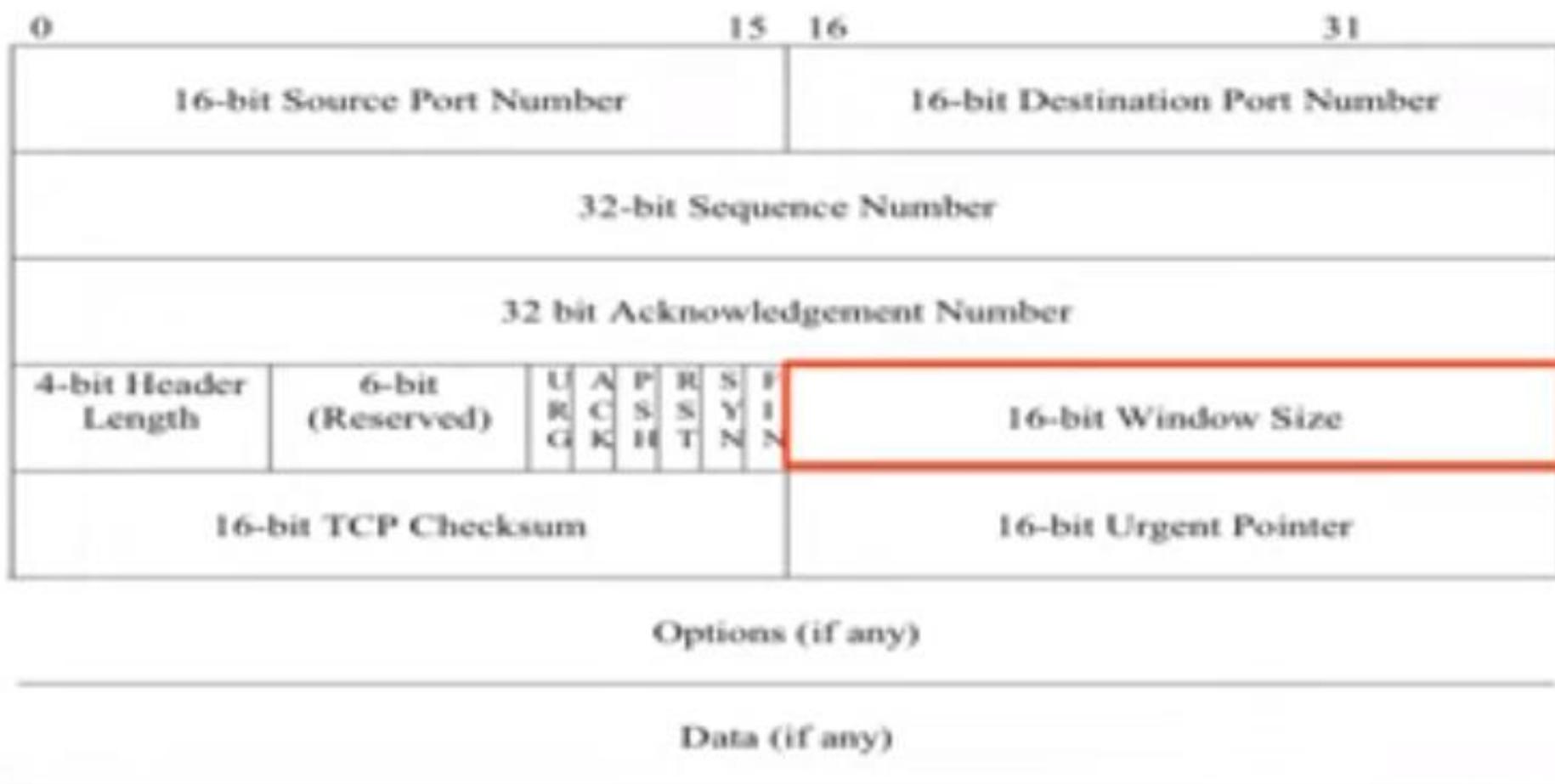
203,663 views

0 0% 0P 20 4K 1080P 1080i

- TCP Flow Control means that TCP will help ensure that the sender is not overwhelming the receiver by sending more packets (segments) than it can handle.

Windowing and Flow Control

- TCP implements flow control by increasing / decreasing window sizes as required.
 - Window sizes are variable during the lifetime of a connection.



Client has a Window Size of 5,000 bytes

Client



Client Window Size=5,000

Server Window Size=10,000

...

ACK=5,001

Client Window Size=5,000

Server Window Size=10,000

...

ACK=10,001

Client Window Size=5,000

Server Window Size=10,000

...

Web Server

MSS of 1,000 bytes

Send Window=5,000

SEQ=1 (to 1,000)

SEQ=1,001 (to 2,000)

SEQ=2,001 (to 3,000)

SEQ=3,001 (to 4,000)

SEQ=4,001 (to 5,000)

Send Window: Byte 10,000

SEQ=5,001 (to 6,000)

SEQ=6,001 (to 7,000)

SEQ=7,001 (to 8,000)

SEQ=8,001 (to 9,000)

SEQ=9,001 (to 10,000)

Send Window: Byte 15,000

SEQ=10,001 (to 11,000)

- **Send Window Byte:** This is the last byte that can be sent before receiving an ACK
- This is known as a **Stop-and-Wait** windowing protocol.
- Server must wait for acknowledgment before continuing to send data.
- A better method?

TCP Window Management

Before discussing data transfer in TCP and the issues such as flow, error, and congestion control, we describe the windows used in TCP.

- TCP uses two windows (send window and receive window) for each direction of data transfer, which means four windows for a bidirectional communication.
- To make the discussion simple, we make an assumption that communication is only unidirectional;
- The bidirectional communication can be inferred using two unidirectional communications with piggybacking.

TCP Window Management

Figure 15.22 Send window in TCP

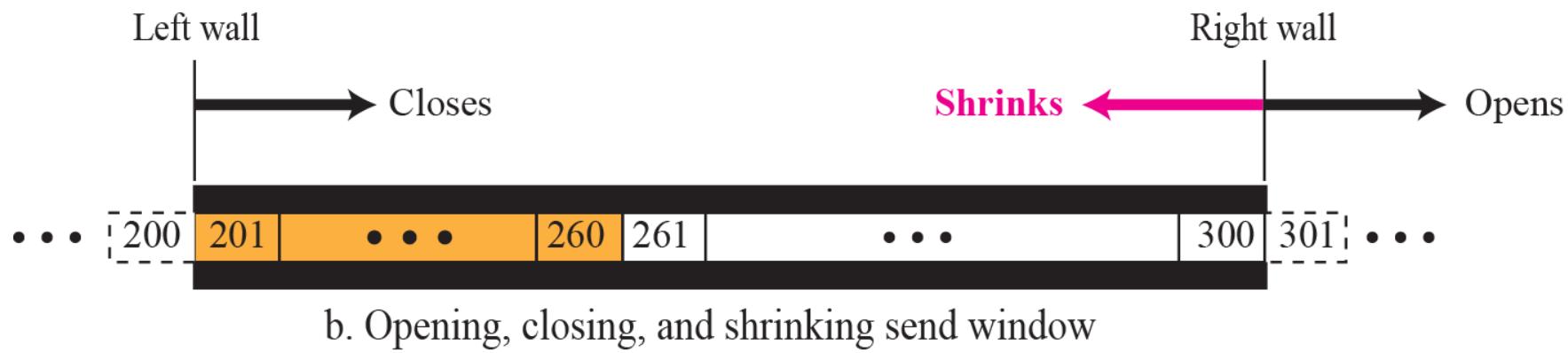
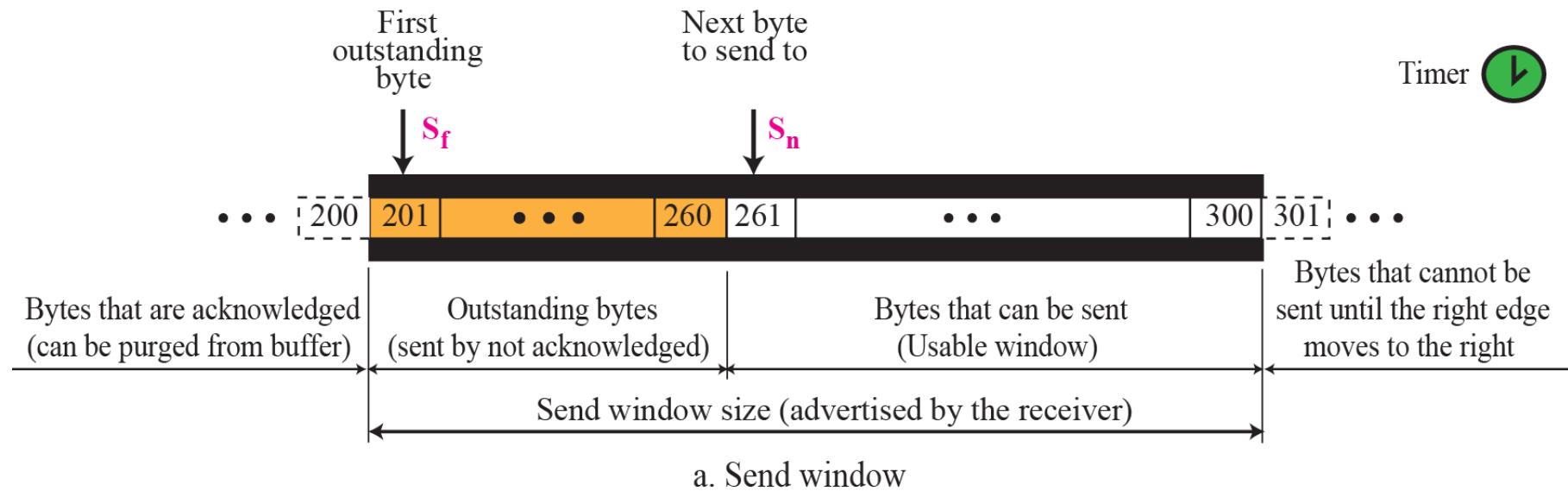


Figure 15.23 Receive window in TCP

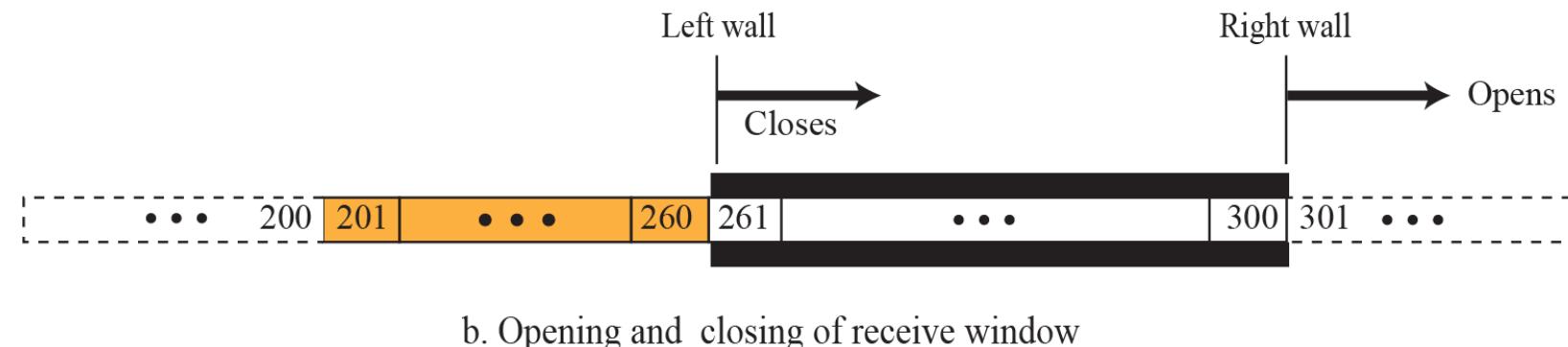
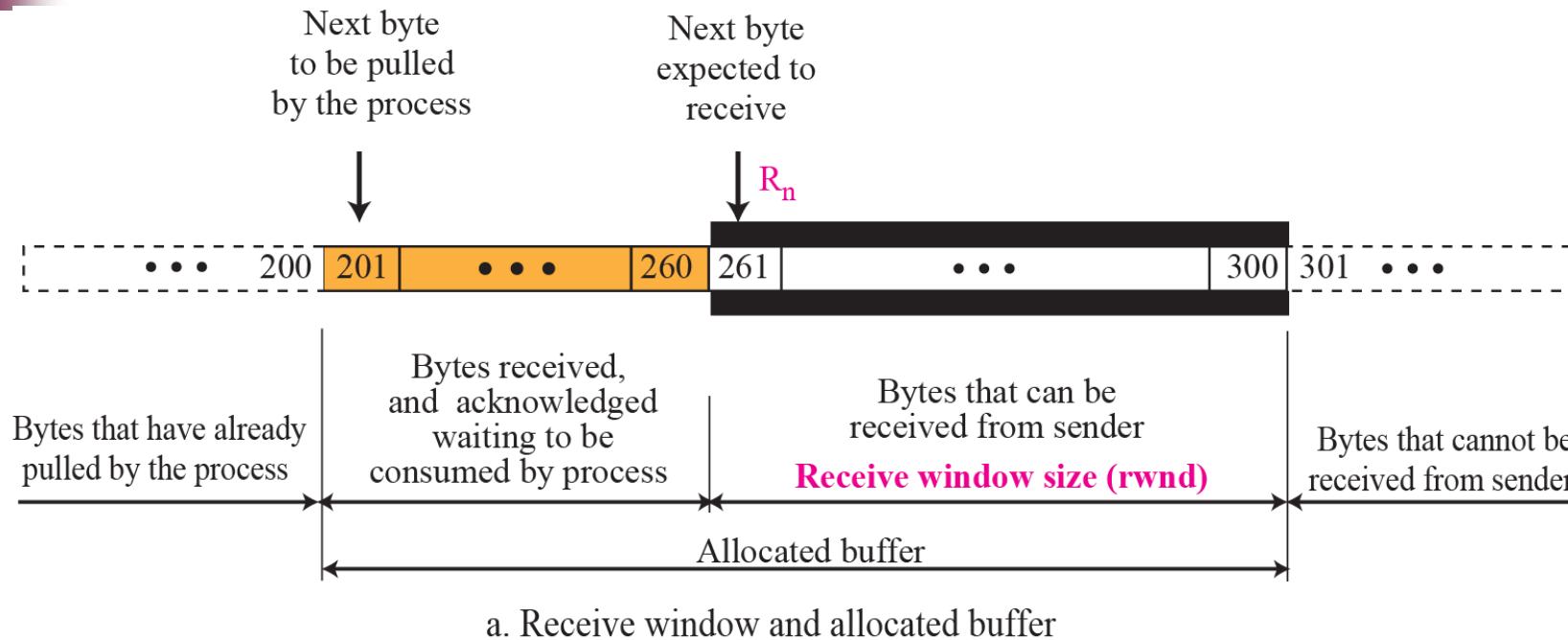
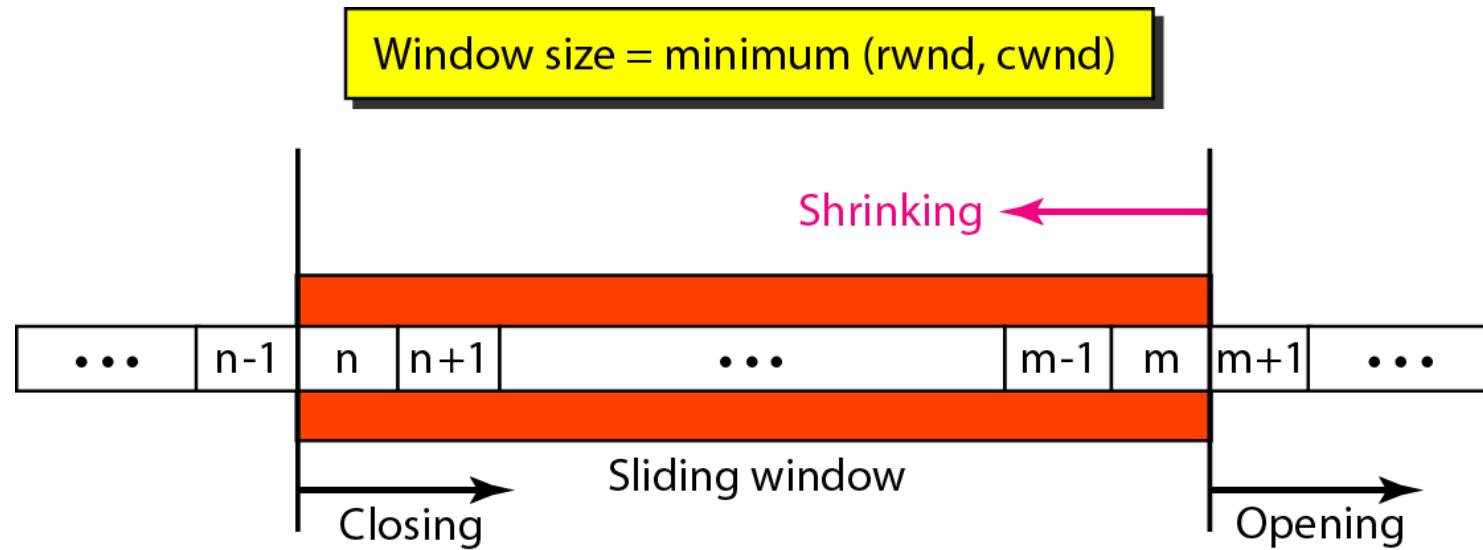
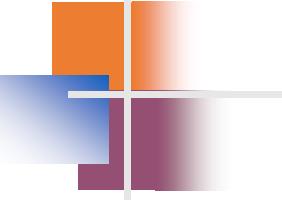


Figure 23.22 Sliding window

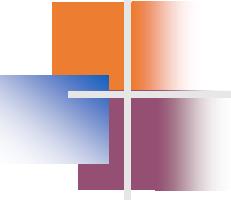




Note

A sliding window is used to make transmission more efficient as well as to control the flow of data so that the destination does not become overwhelmed with data.

TCP sliding windows are byte-oriented.

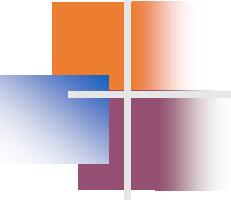


Example 23.4

What is the value of the receiver window (rwnd) for host A if the receiver, host B, has a buffer size of 5000 bytes and 1000 bytes of received and unprocessed data?

Solution

The value of rwnd = 5000 – 1000 = 4000. Host B can receive only 4000 bytes of data before overflowing its buffer. Host B advertises this value in its next segment to A.

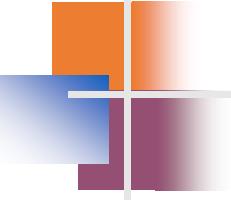


Example 23.5

What is the size of the window for host A if the value of rwnd is 3000 bytes and the value of cwnd is 3500 bytes?

Solution

The size of the window is the smaller of rwnd and cwnd, which is 3000 bytes.



Example 23.6

Figure 23.23 shows an unrealistic example of a sliding window. The sender has sent bytes up to 202. We assume that cwnd is 20 (in reality this value is thousands of bytes). The receiver has sent an acknowledgment number of 200 with an rwnd of 9 bytes (in reality this value is thousands of bytes). The size of the sender window is the minimum of rwnd and cwnd, or 9 bytes. Bytes 200 to 202 are sent, but not acknowledged. Bytes 203 to 208 can be sent without worrying about acknowledgment. Bytes 209 and above cannot be sent.

Figure 23.23 Example 23.6

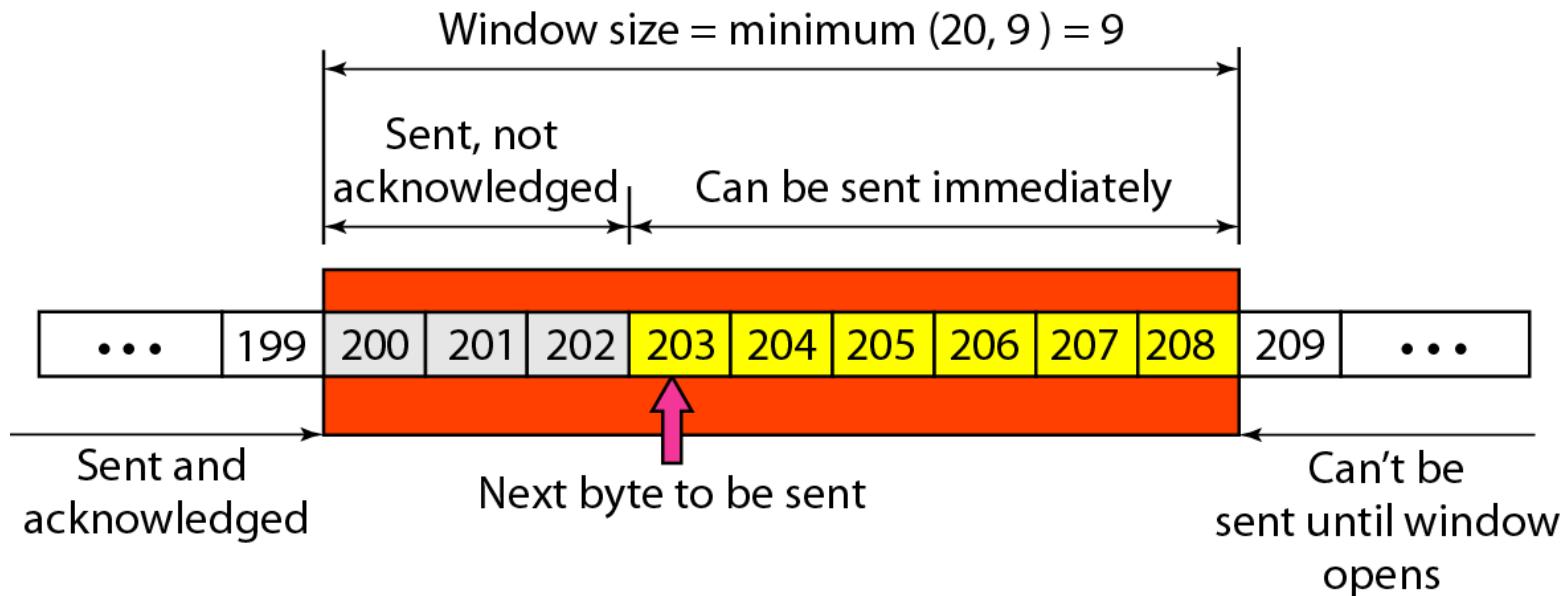


Figure 23.24 Normal operation

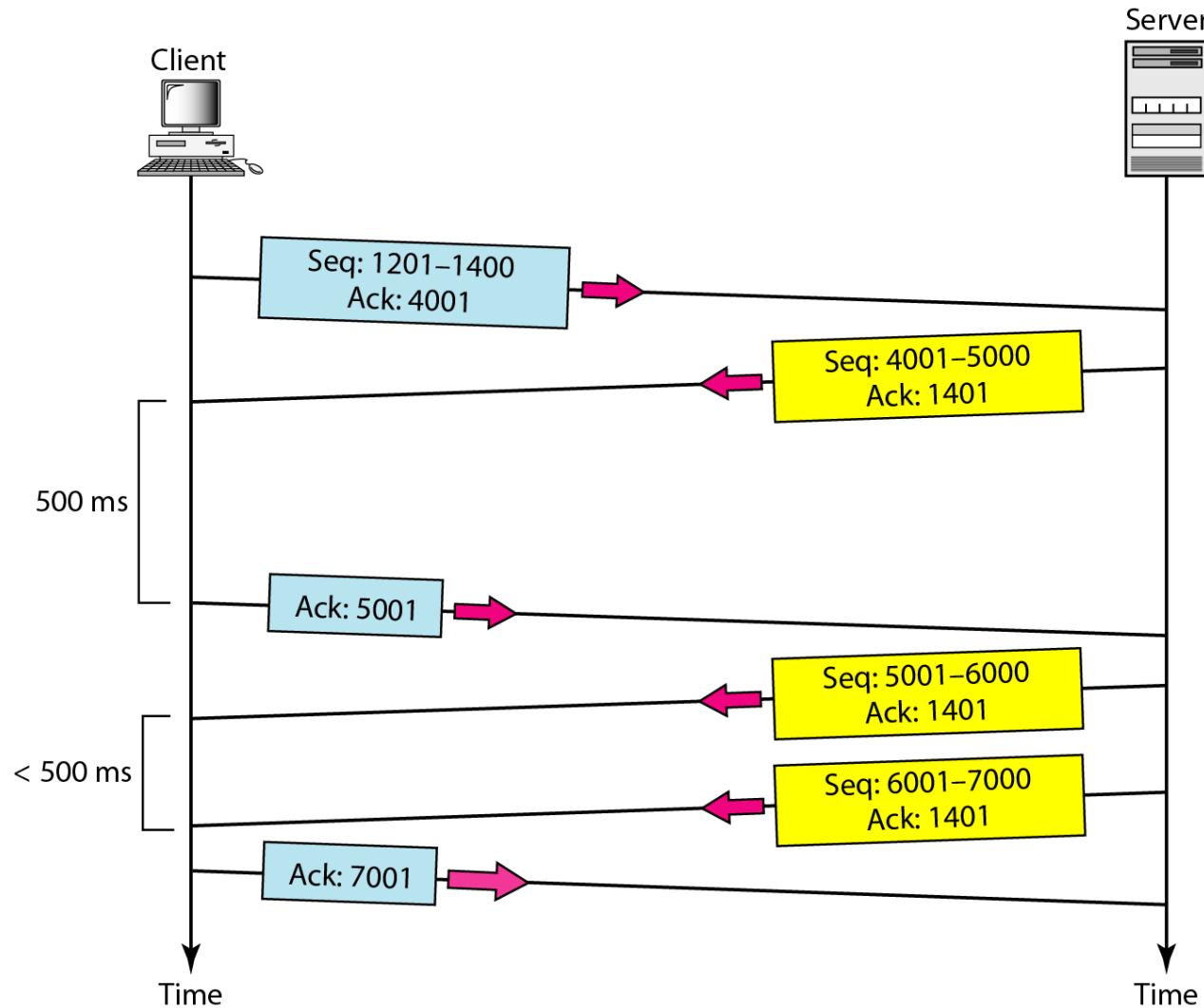
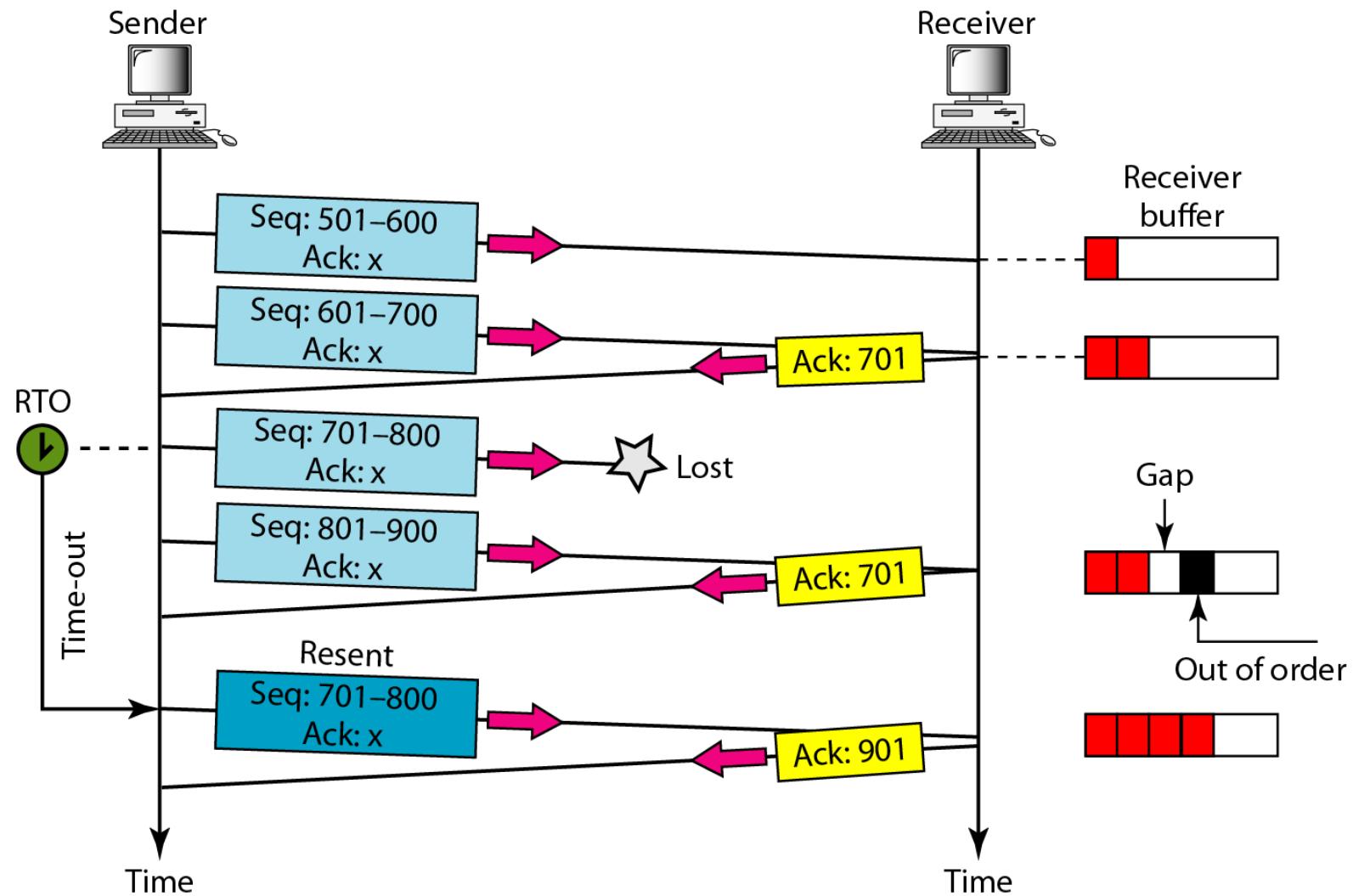
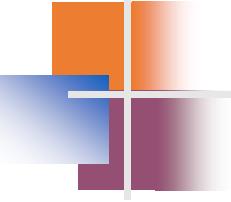


Figure 23.25 Lost segment

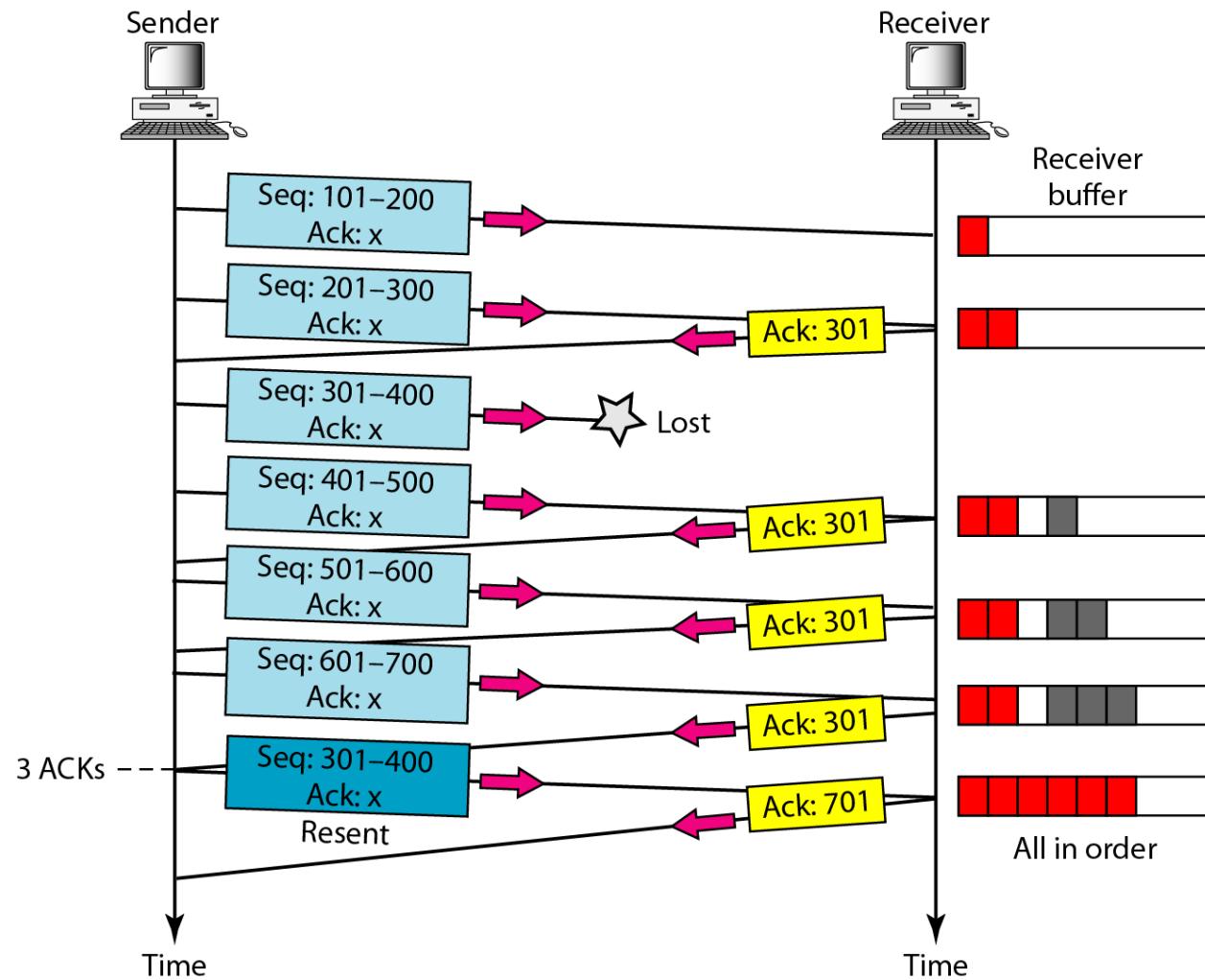




Note

The receiver TCP delivers only ordered data to the process.

Figure 23.26 *Fast retransmission*



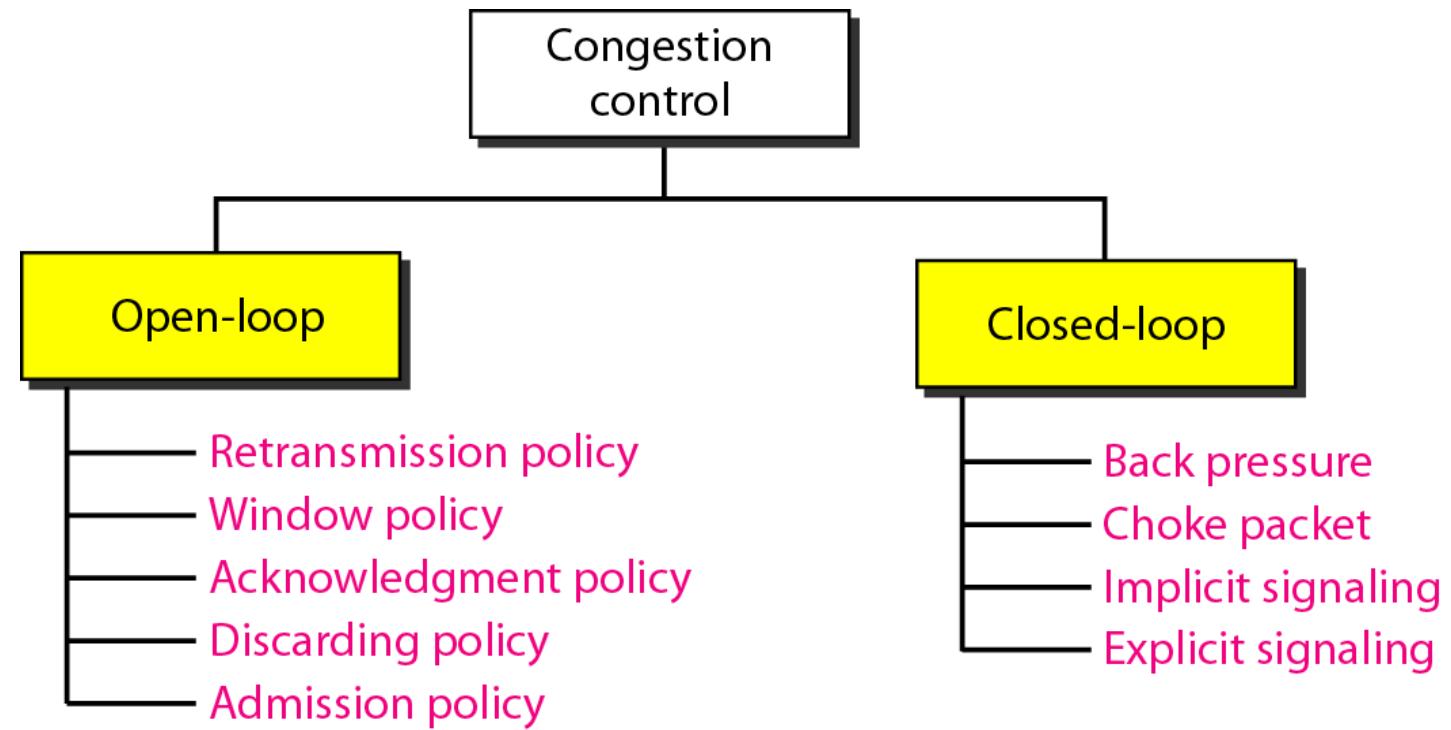
15-9 CONGESTION CONTROL

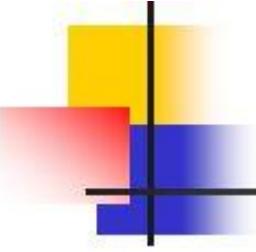
Congestion control in TCP is based on both open loop and closed-loop mechanisms. TCP uses a congestion window and a congestion policy that avoid congestion and detect and alleviate congestion after it has occurred.

Congestion Control Introduction:

- When too many packets are present in (a part of) the subnet, performance degrades. This situation is called **congestion**.
- As traffic increases too far, the routers are no longer able to cope and they begin losing packets.
- At very high traffic, performance collapses completely and almost no packets are delivered.
- **Reasons of Congestion:**
 - Slow Processors.
 - High stream of packets sent from one of the sender.
 - Insufficient memory.
 - High memory of Routers also add to congestion as becomes un manageable and un accessible. (Nagle, 1987).
 - Low bandwidth lines.

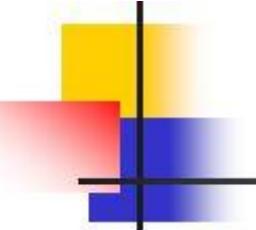
Figure 24.5 Congestion control categories





Congestion Control

- Open-loop congestion control [Prevention]
 - Windows policy: Type of window at sender may also affect congestion. Selective repeat is better than Go-Back-N.
 - Acknowledgement policy: Policy set by receiver may also affect congestion. If receiver does not acknowledge every packet it receives, it may slow down the sender and help prevent congestion.
 - Discard policy: Discard less sensitive packets [in audio transmission] at routers.
 - Admission policy: Switches in a flow first check the resource requirement of a flow before admitting it to the network.



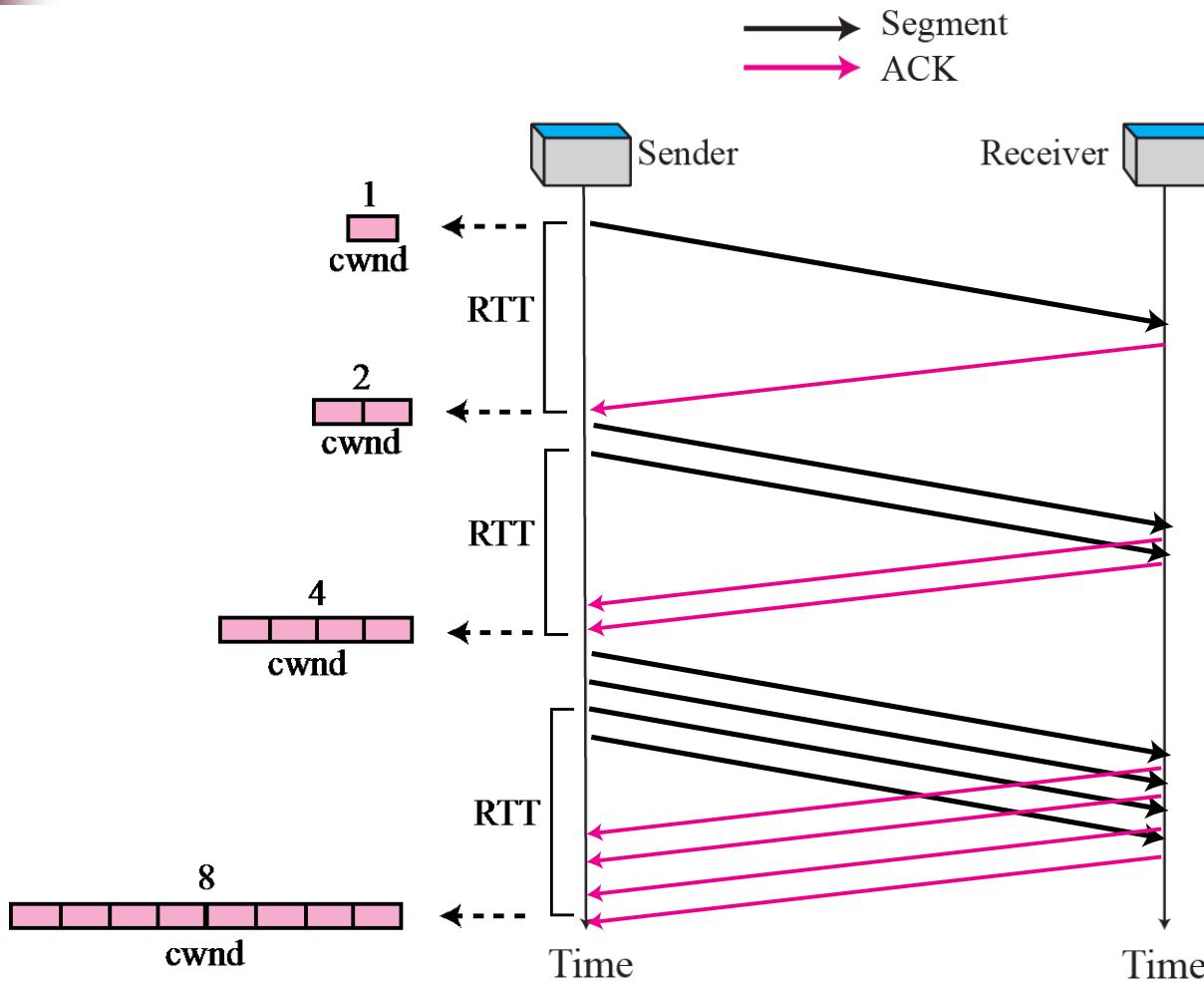
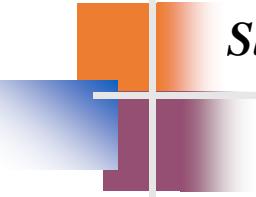
Closed-loop congestion control [Removal]

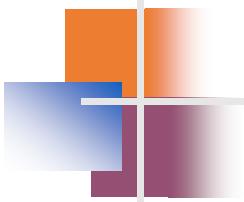
- Back Pressure: When a router is congested, it can inform the previous upstream router to reduce the rate of outgoing packets. The action can be recursive all the way to the router before the source.
- Choke Point: A packet sent by a router to the source to inform it of congestion. This type of control is similar to ICMP's source quench packet.
- Implicit Signaling: Source can detect an implicit signal concerning congestion and slow down its sending rate. For example, the mere delay in receiving an acknowledgement can be a signal that the network is congested.
- Explicit Signaling: Routers that experience congestion can send an explicit signal, the setting of a bit in a packet, for example, to inform the sender or the receiver of congestion.
 - Backward Signaling: Bit can be set in a packet moving in the direction opposite to the congestion; indicate the source.
 - Forward Signaling: Bit can be set in a packet moving in the direction of the congestion; indicate the destination.

Congestion Control in TCP

- Slow Start
- Additive Increase (Congestion Avoidance)
- Multiplicative decrease

Slow start, exponential increase





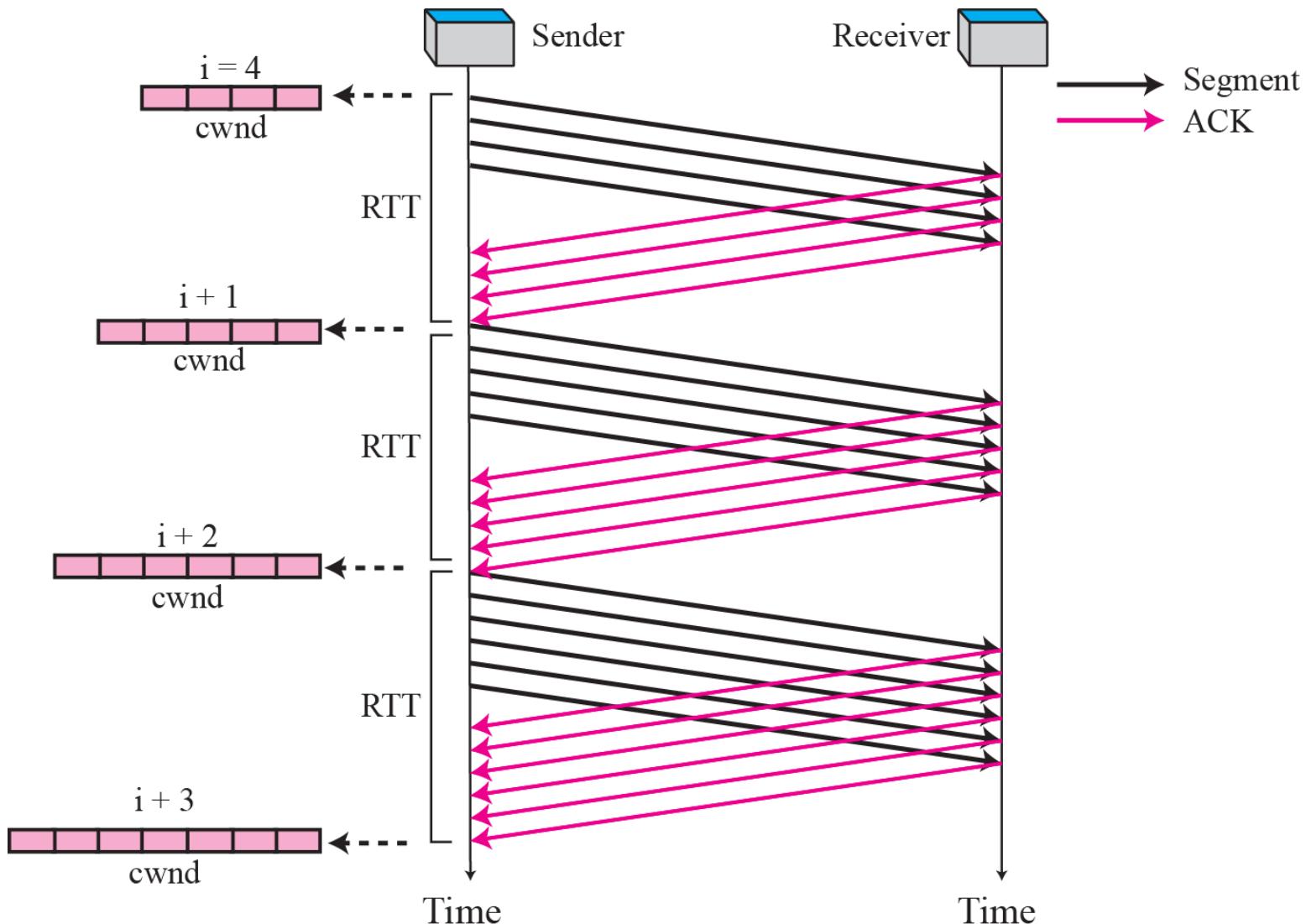
Note

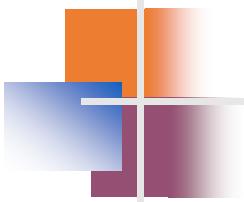
In the slow start algorithm, the size of the congestion window increases exponentially until it reaches a threshold (ssthresh).

What will happen reaches to threshold

- $\text{ssthresh} = \text{window}/2$
- $\text{Cwnd} = \text{ssthresh}$

**Figure Congestion avoidance, additive increase
multiplicative decrease (AIMD)**





Note

***In the congestion avoidance algorithm
the size of the congestion window
increases additively until
congestion is detected.***

What will happen reaches to threshold

- Congestion is detected by timeout
- $ssthresh=window/2$
- $cwnd=1$

Multiplicative Decrease

- Congestion is detected by 3 dup Acks
- $ssthresh = window/2$
- $cwnd = ssthresh$

- Packet =50
- Lost Packets = 10,25,34,45



Congestion threshold =

$$\frac{\text{Window size}}{2} = \frac{8}{2} = 4$$

5 | 10

6 | 11 12

7 | 13 14 15 16

Congestion window = Congestion threshold

Now Linear increment will happen

8 | 17 18 19 20 21

9 | 22 23 24 25 26 27

Window size=6

Congestion threshold =

$\frac{\text{Window size}}{2} = \frac{6}{2} = 3$

10

25

11

26

27

12

28

29

30

threshold \geq Size of Window
Now linear increment will happen

13

31

32

33

34

Window size=4

Congestion threshold =

Window size = $\frac{4}{2}$ = 2

14

34

15

35

36

Congestion window = Congestion threshold

Now Linear increment will happen

16

37

38

39

17

40

41

42

43

18

44

45

46

47

48

Window size=5

Congestion threshold =

$\frac{\text{Window size}}{2} = \frac{5}{2} = 2.5$

19

45

20

46

47

Congestion window = Congestion threshold

Now Linear increment will happen

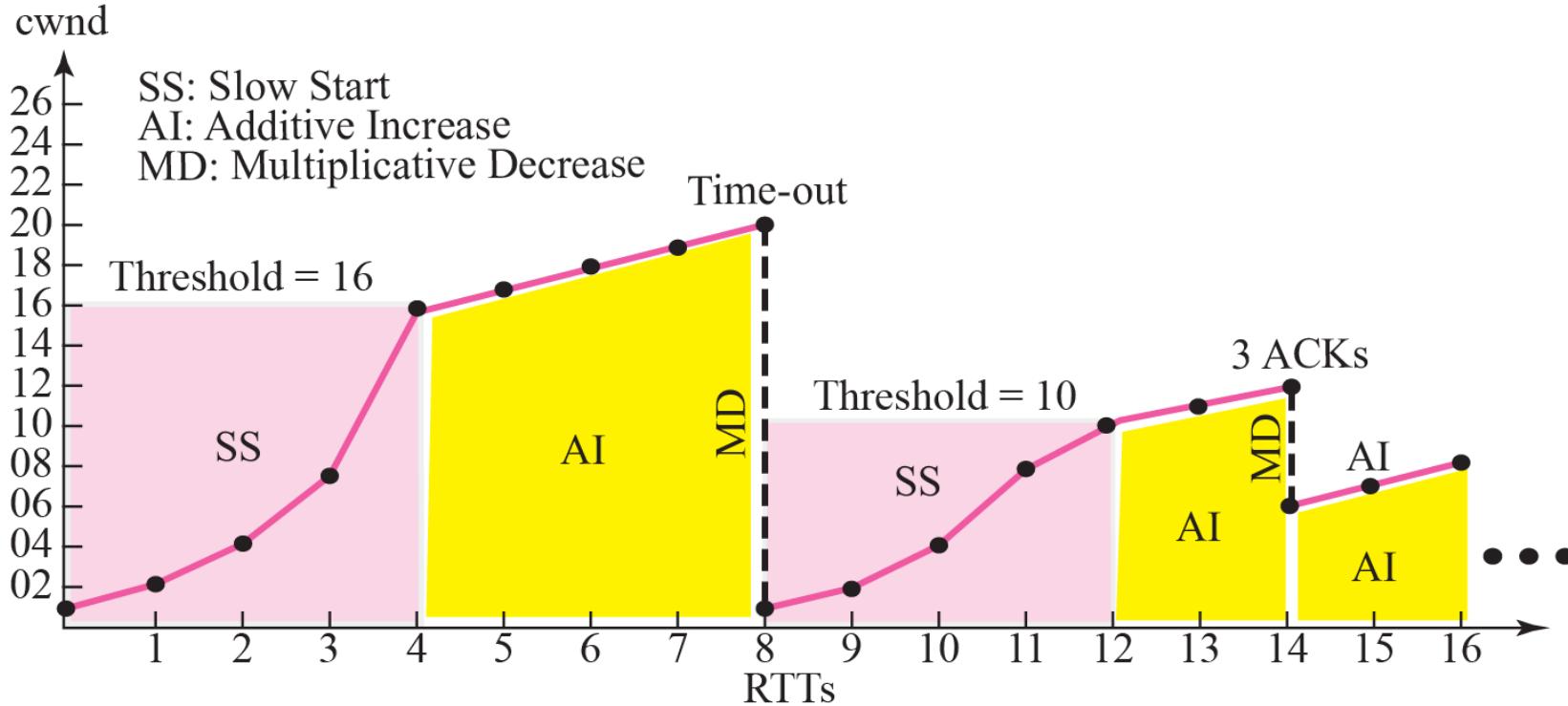
21

48

49

50

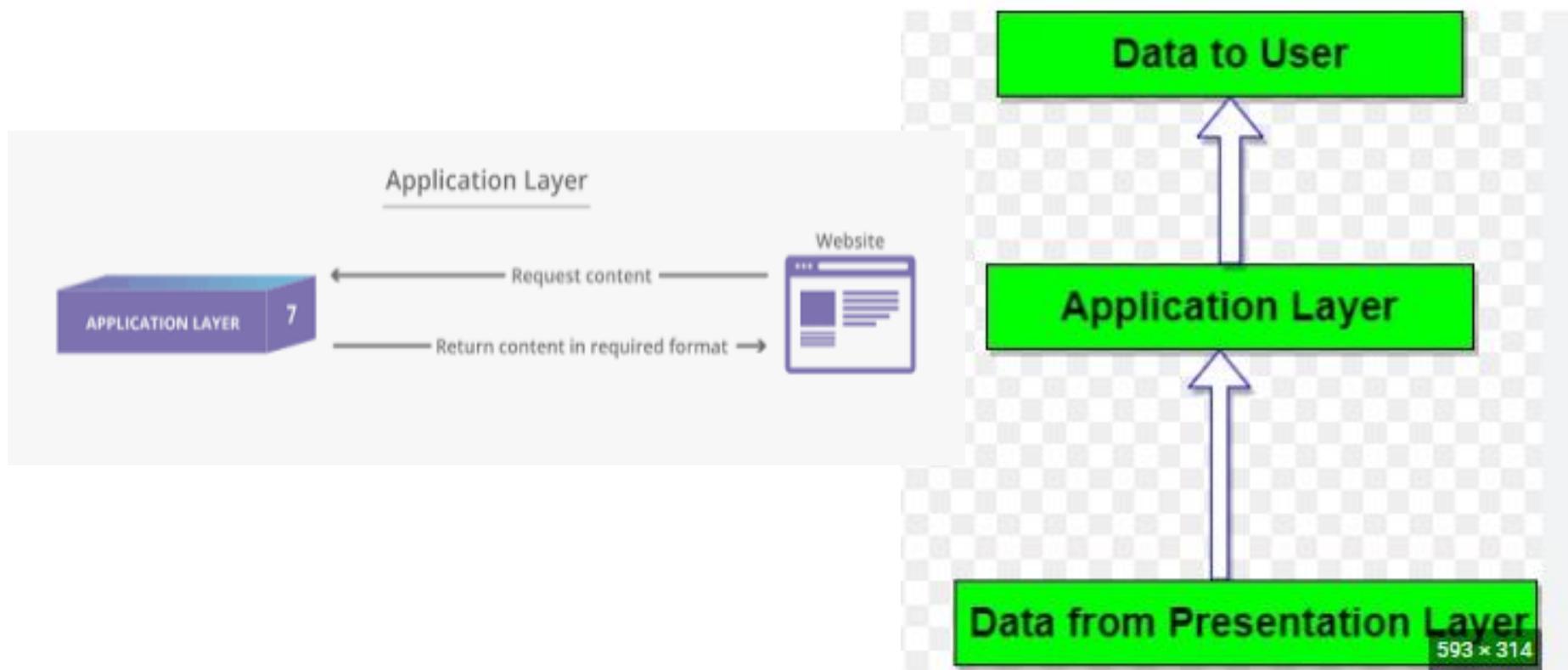
Figure 15.37 Congestion example



**APPLICATION
LAYER**

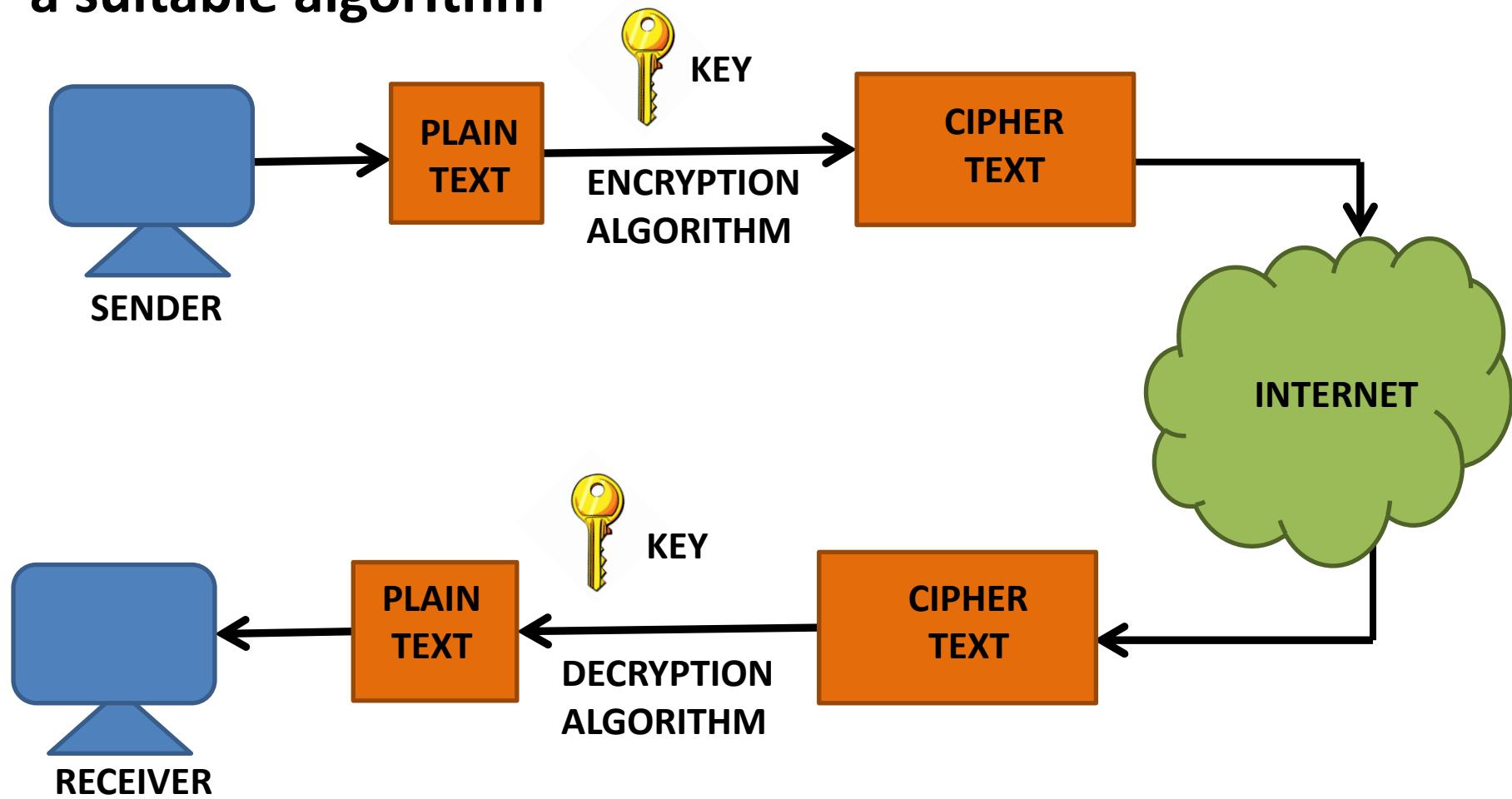
Application Layer

- Topmost layer in OSI model that deals different application running over the system using different application layer protocols such as HTTP, FTP, TELNET etc.
- It ensures an application can effectively communicate with other applications on different computer systems and networks.



Data Encryption

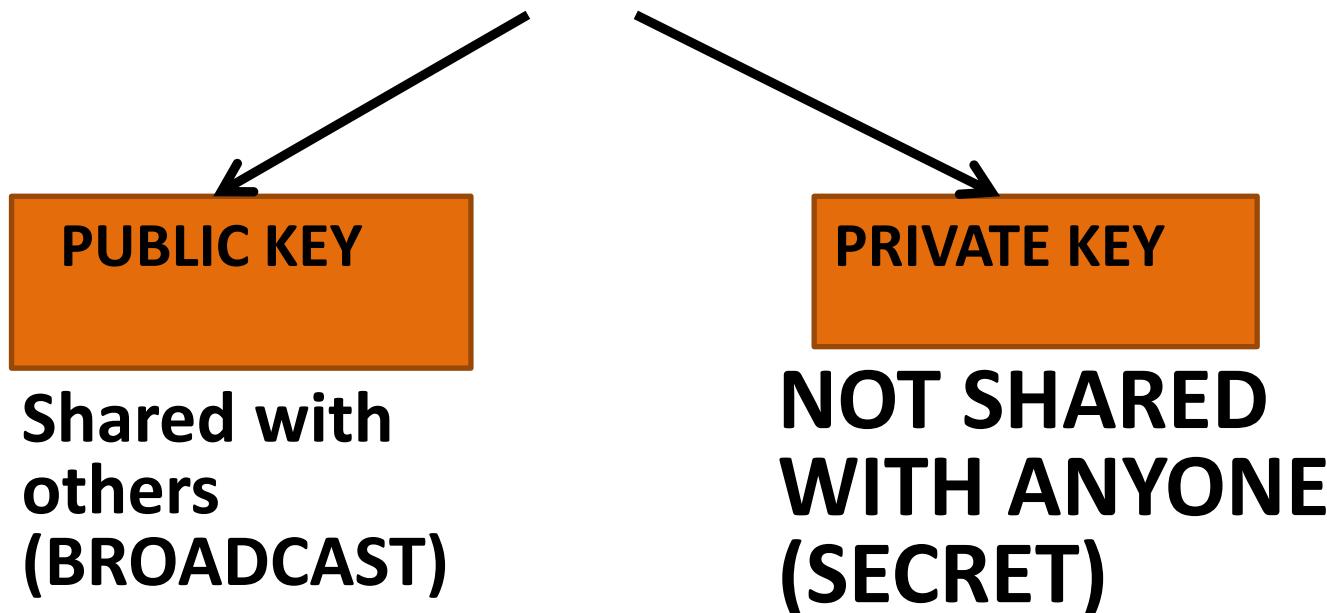
- Conversion of our Readable form data (**Plain Text**) into non Readable form(**Cipher Text**) are known as Data Encryption
- Data Encryption can be performed by using keys and a suitable algorithm



Data Encryption Keys

Data Encryption keys are value or property by which we can perform encryption and decryption.

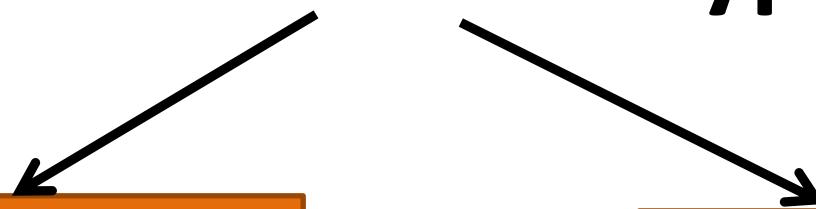
Types of Encryption Keys



Types of Data Encryption

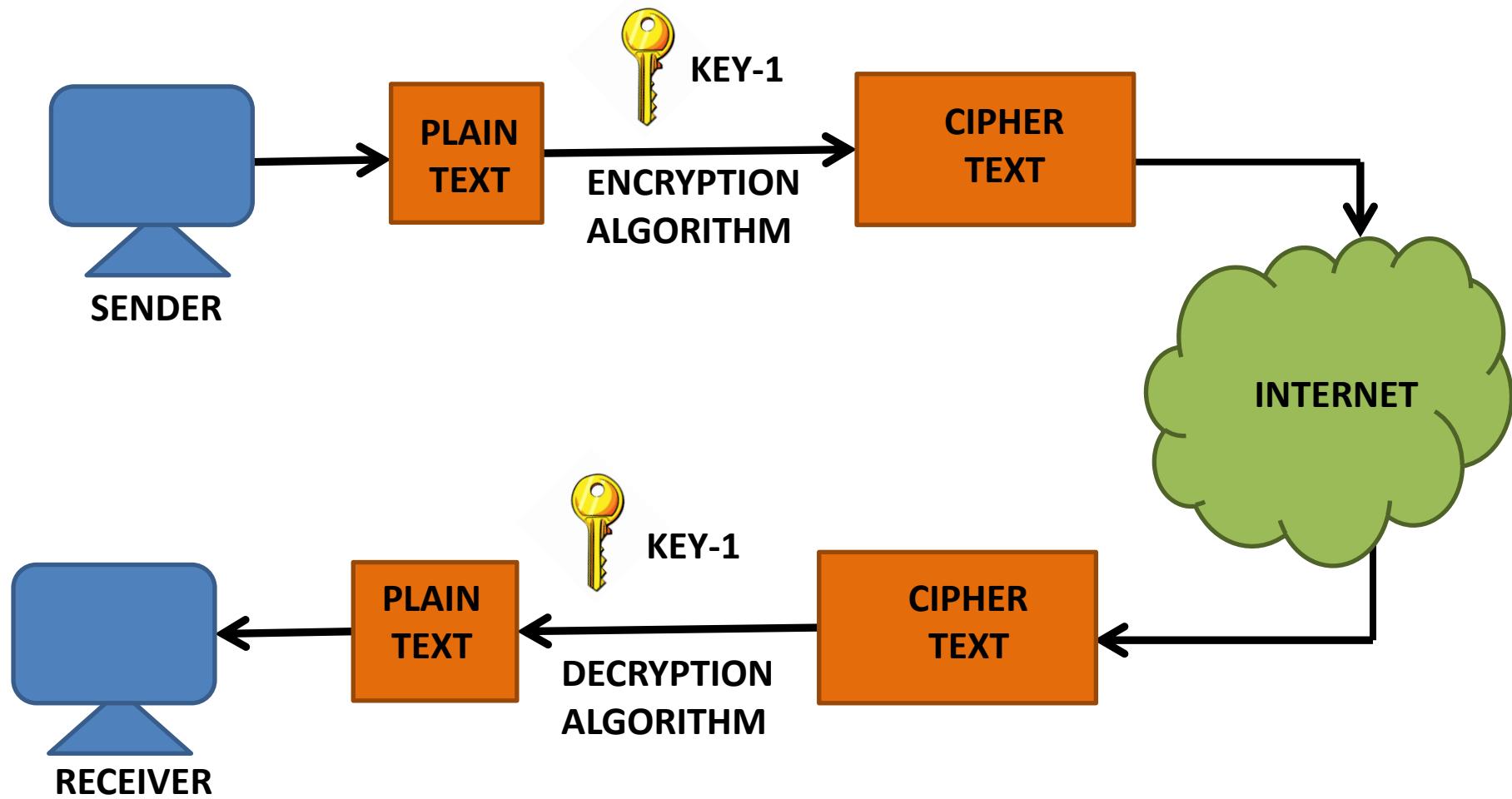
SYMMETRIC
For Eg. DES

ASYMMETRIC
For Eg. RSA

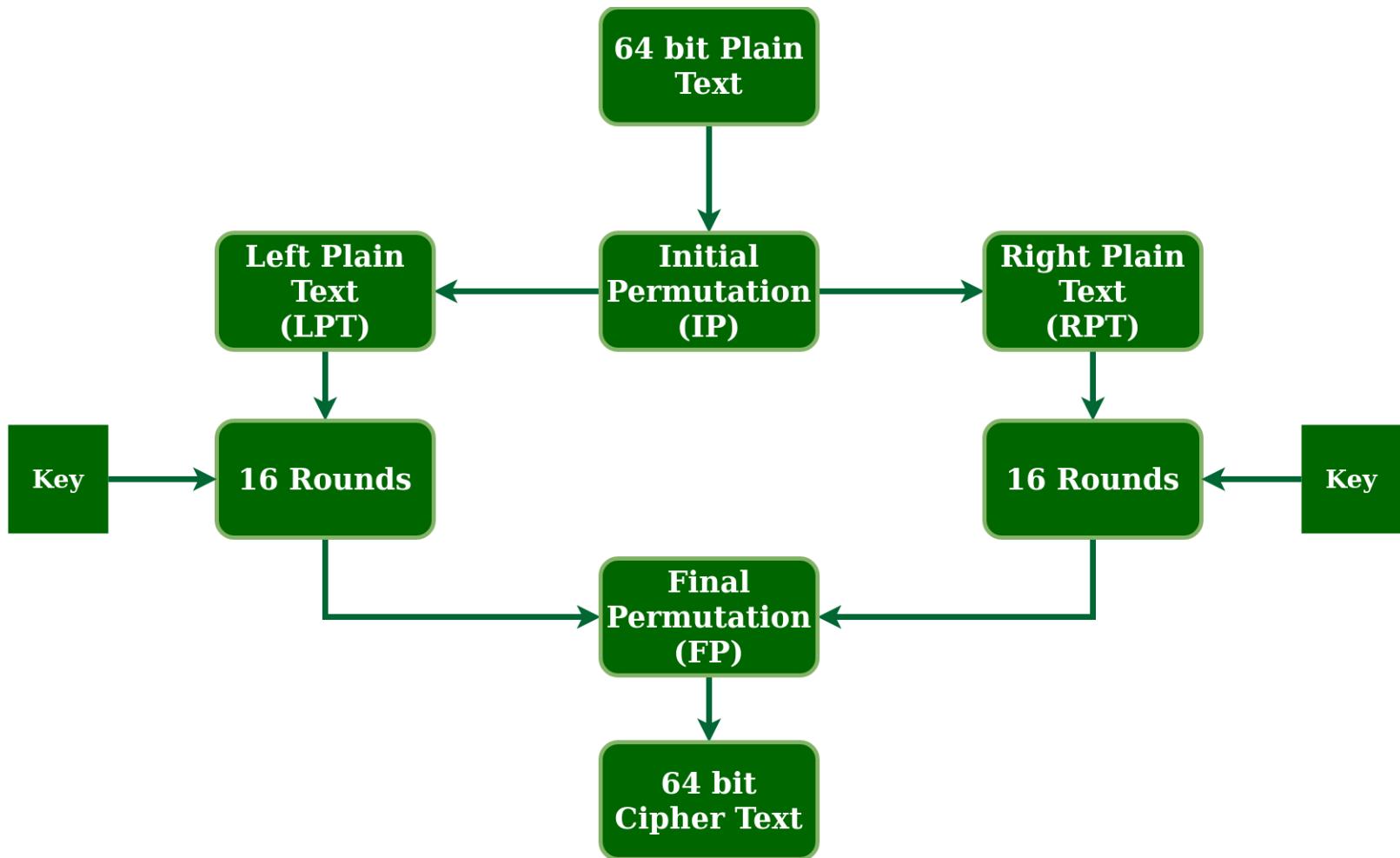


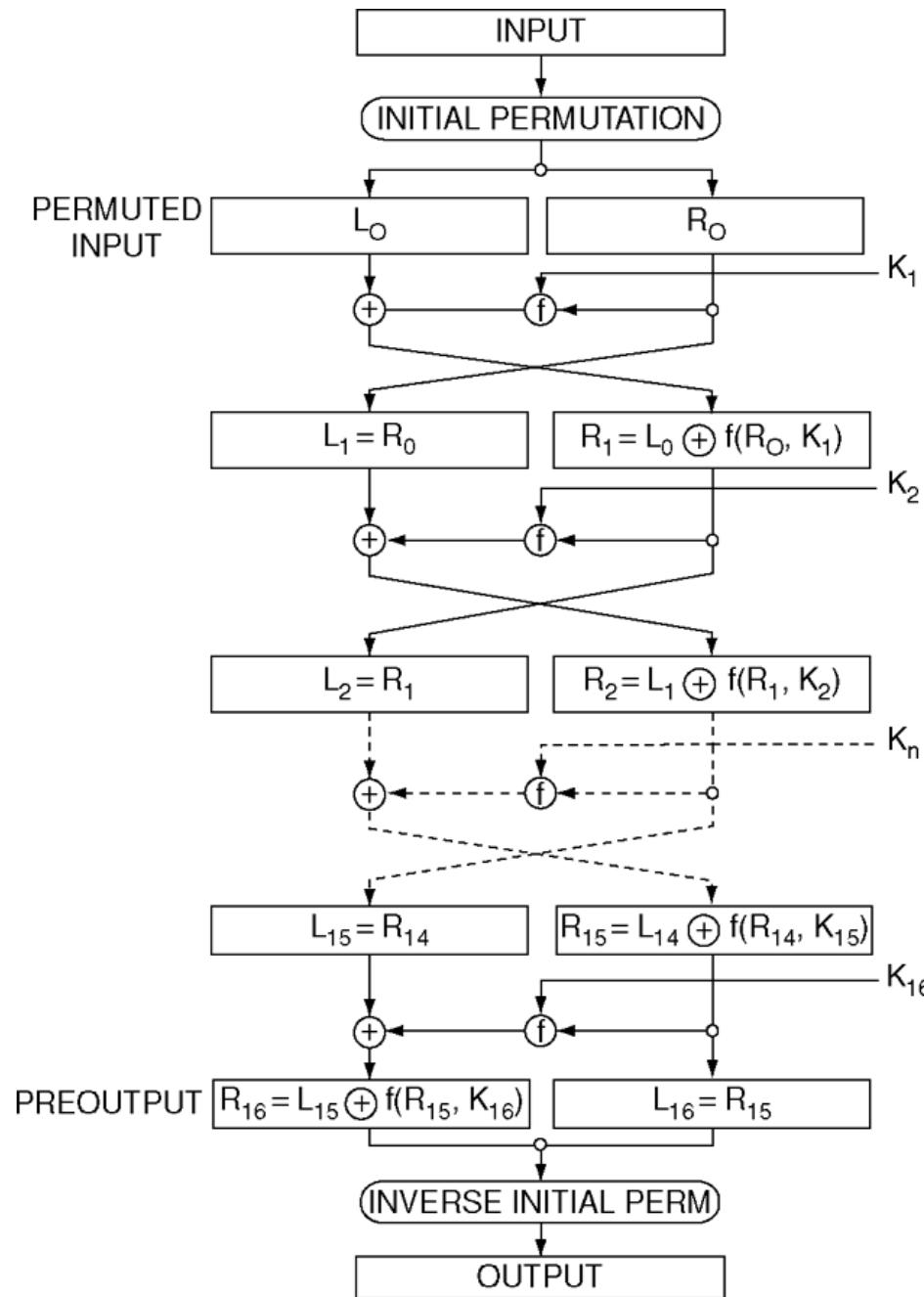
Symmetric Data Encryption

When Encryption and Decryption can be performed by using only same type of key



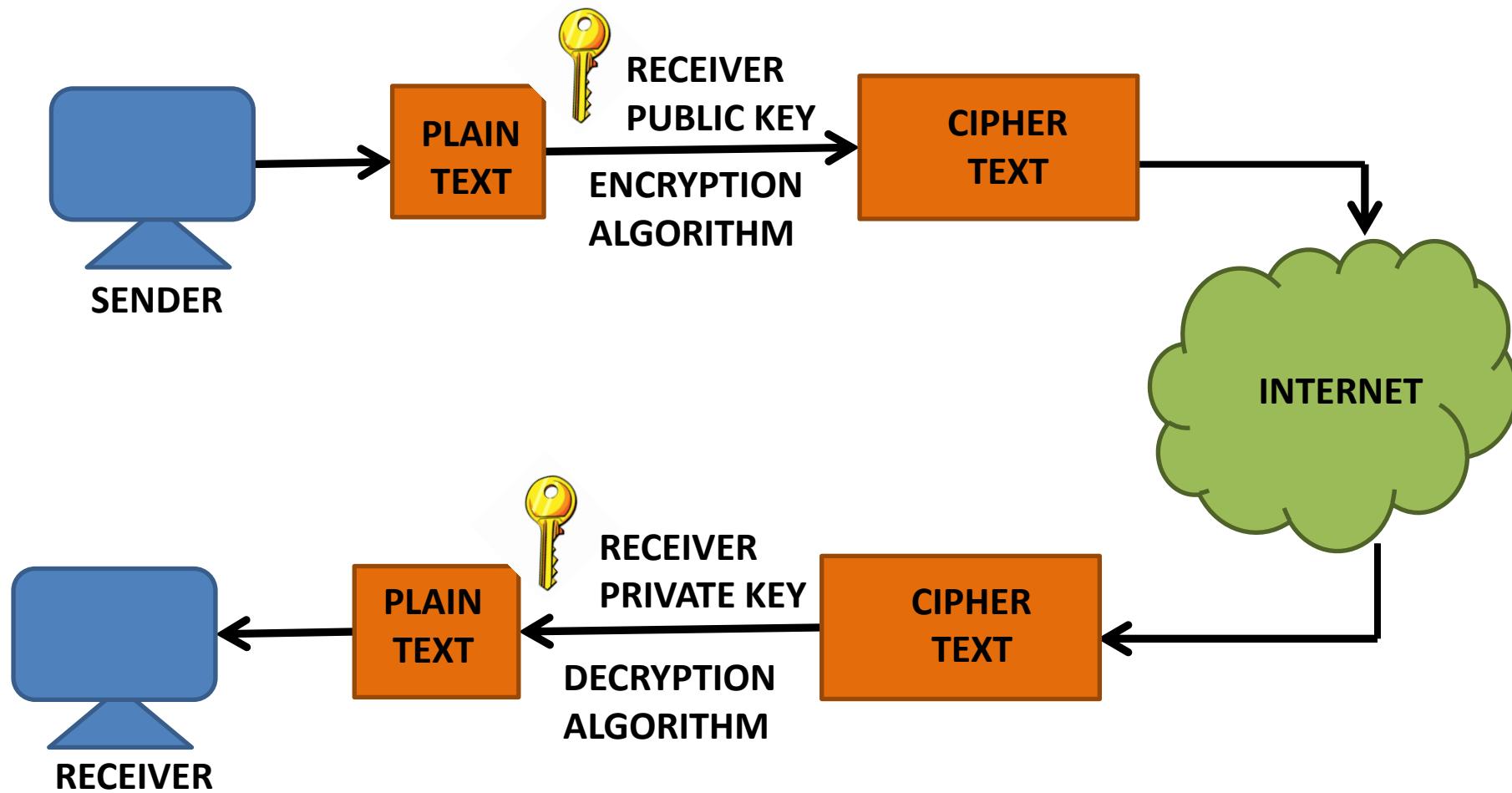
DES Data Encryption





Asymmetric Data Encryption

When Encryption and Decryption can be performed by using two different type of key(Public or Private)



RSA Algorithm

1. Choose two large primes p and q (typically around 256 bits)
2. Compute: $n = p \times q$
3. Calculate totient function $\phi(n) = (p-1)(q-1)$
4. Choose a number 'd' relatively prime to 'z'
 $1 < e < \phi(n)$, e is co-prime to $\phi(n)$, $\text{gcd}(e, \phi(n)) = 1$
5. Find e^{-1} such that $e^{-1}d \text{ mod } (p-1)(q-1) = 1$
6. for encryption:
$$C = P^e \pmod{n}$$

for decryption:
$$D = C^d \pmod{n}$$

Q. Let p=3 and q=11 and plain text message length M=31. Encrypt and Decrypt it using RSA and also calculate its Public Key 'e' and Private Key 'd'.

1. Calculate $n=p \times q$ So, $n= 3 \times 11= 33$
2. Calculate $\phi(n) = (p-1)x(q-1)$. So, $\phi(n)=2x10=20$
3. Calculate 'e', Let $e=7$ As $\gcd(7,20)=1$ & $1<7<20$.
4. Calculate 'd'. As $d \equiv e^{-1} \pmod{\phi(n)}$

$$d \times 7 \pmod{20}=1$$

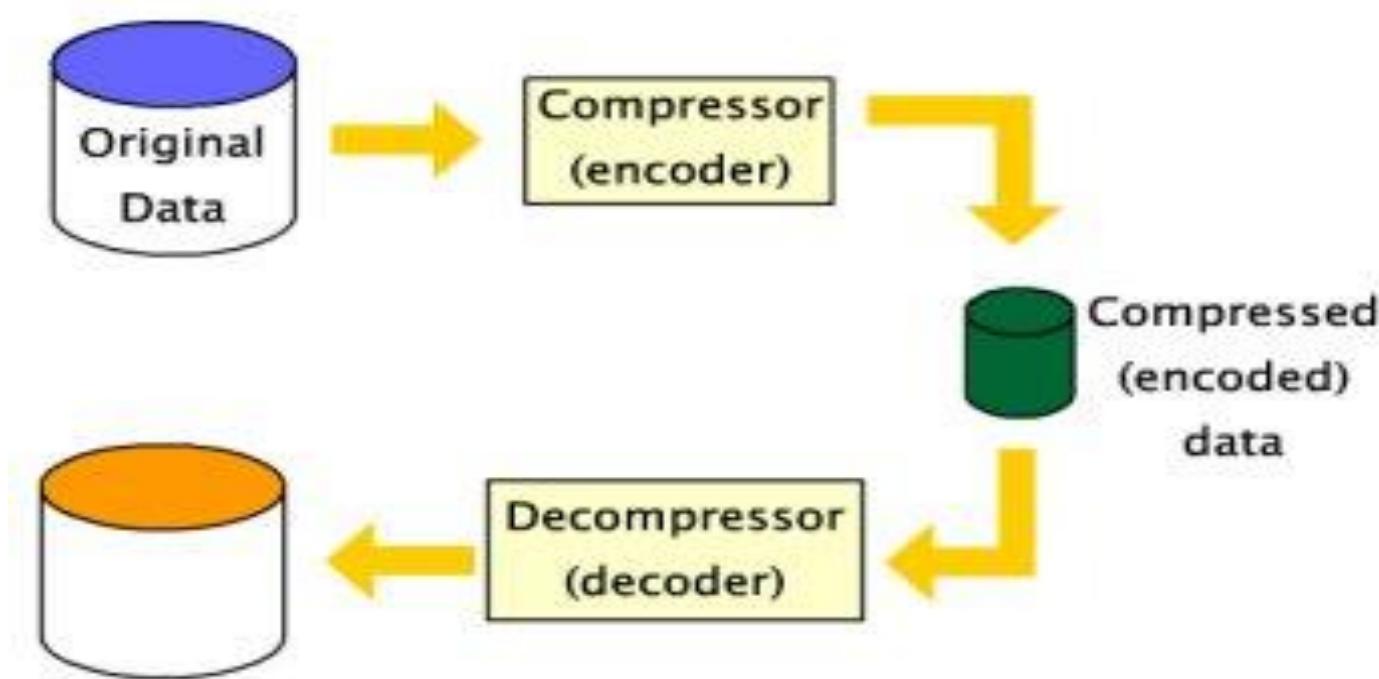
$$3 \times 7 \pmod{20}=1$$

So, $d=3$

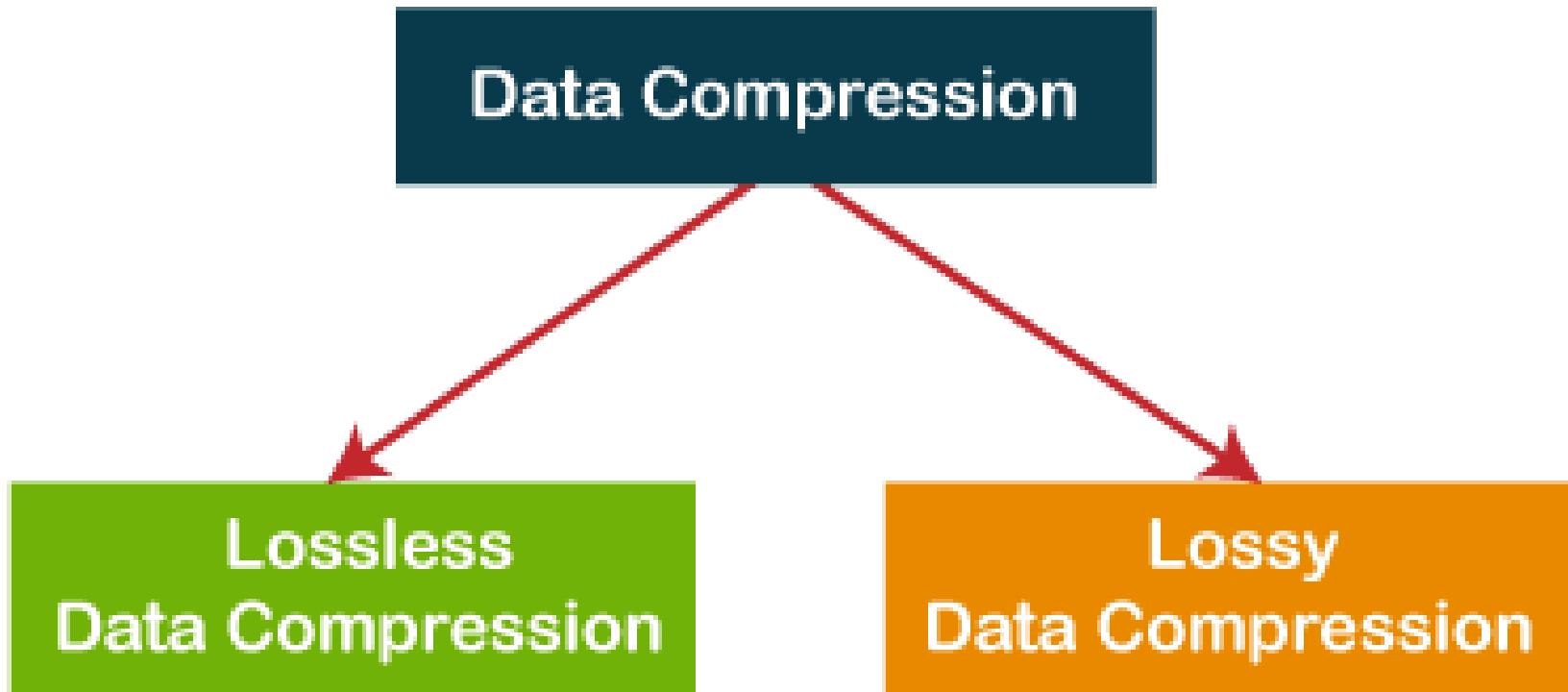
5. Encrypt Cipher $C= M^e \pmod{n}$, $C=31^7 \pmod{33}=4$
6. Decrypt Plain $M= C^d \pmod{n}$, $M=4^3 \pmod{33}=31$

Data Compression

- Data compression means conversion of Large Data into Small reduced form (Reduction in the number of bits needed to represent data)
- Compressing data can save storage capacity, speed up file transfer, and decrease costs for storage hardware and network bandwidth.



Data Compression Techniques



Lossless Data Compression

- Lossless compression is a class of data compression that allows the original data to be perfectly reconstructed from the compressed data with no loss of information.

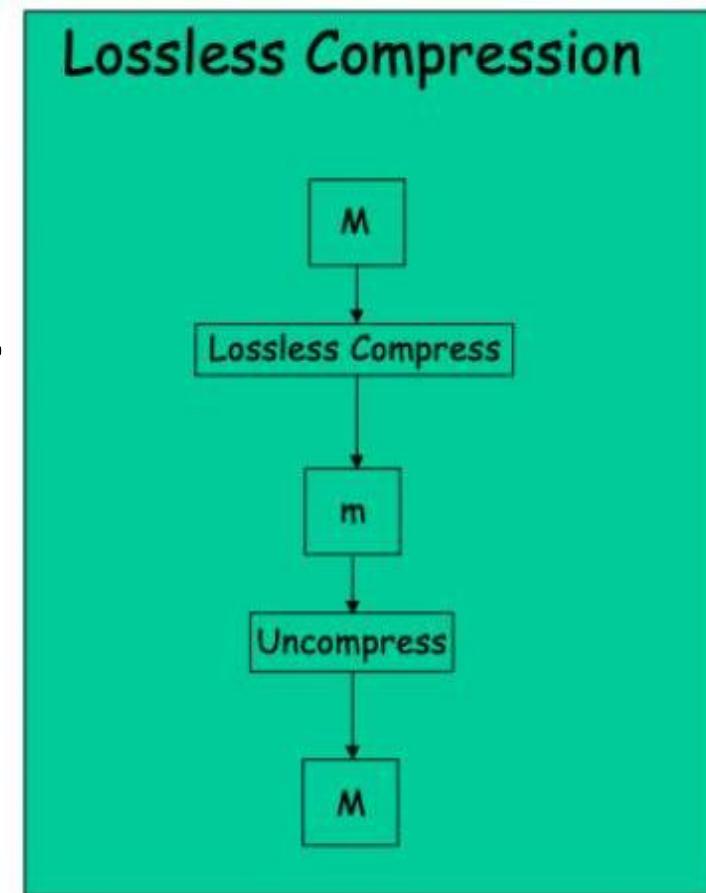
Let M = Original Data (Before Compression)

M' =Data Recovered (After Compression)

$$m=M-M'$$

where m =data lost

Here $m=0$



**Lossless methods
(text or program)**



Lossy Data Compression

- In such type of compression, some data is lost during recovery. Original data may not be recovered as it is.

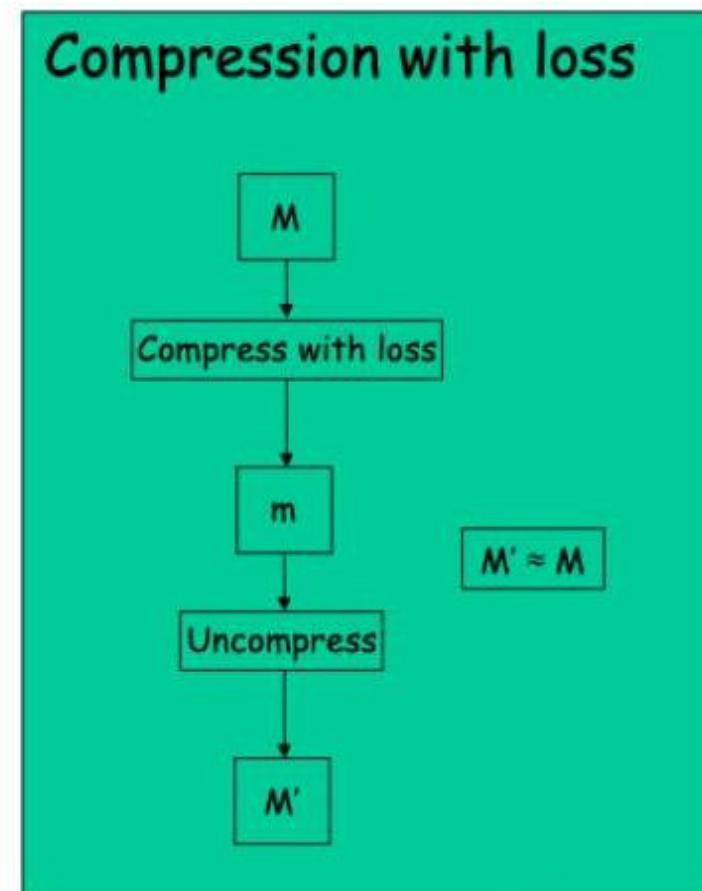
Let M = Original Data (Before Compression)

M' =Data Recovered (After Compression)

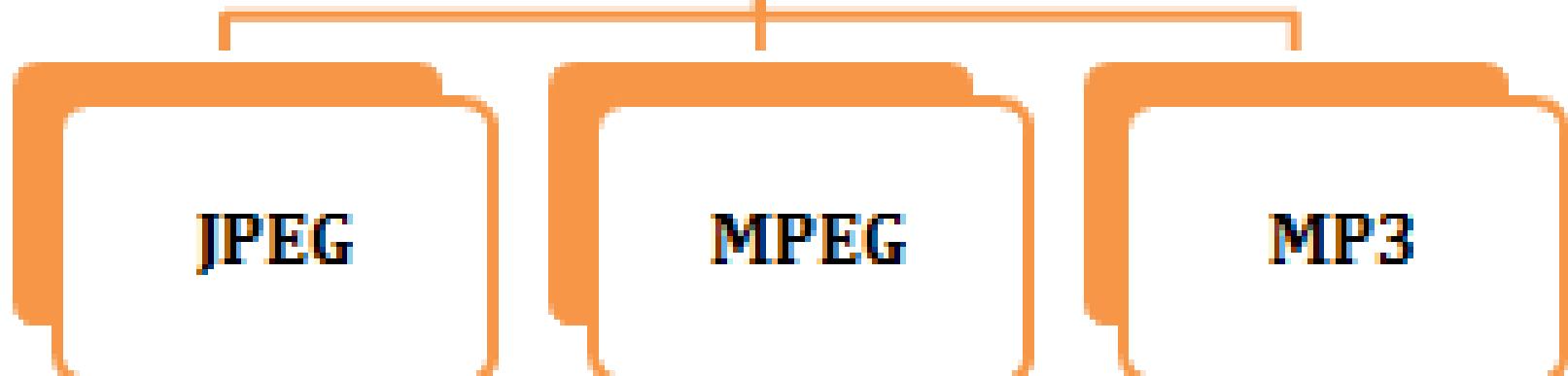
$$m=M-M'$$

where m =data lost

Here $m>0$



**Lossy methods
(image, video,
audio)**



Data Compression Terms

Compression Ratio=

Compressed Data

Original Data

Compression Factor=

1

Compression Ratio

Compression Time: Total Time Taken by Encoder to compress the data.

Decompression Time: Total Time Taken by Decoder to uncompress the data.

Fixed Length Encoding:-

ASCII

a	\rightarrow 97	\rightarrow 7 bits (1100001)	(3 bit)	a \rightarrow 000	7x3 bits = 21 bits
b	:			b \rightarrow 001	
c	:			c \rightarrow 010	
d	:			d \rightarrow 011	
e	:			e \rightarrow 100	
f	:			f \rightarrow 101	
g	:			g \rightarrow 110	

RUN LENGTH ENCODING

- Pick the first character from the source string.
- Append the picked character to the destination string.
- Count the number of subsequent occurrences of the picked character and append the count to the destination string.
- Pick the next character and repeat steps 2, 3 and 4 if the end of the string is NOT reached.

RUN LENGTH ENCODING

Example!

Original bit stream:

00000001111111111100000000000011111111
6 0's 14 1-bits 13 0-bits 9 1-bits

Size \Rightarrow 42 bits

① 0:6, 1:14, 0:13, 1:9

② resulting 5-bit bytes

$\Rightarrow 00110, 11110, 01101, 11001$

Huffman Encoding

- Calculate the frequency of each character in the string.

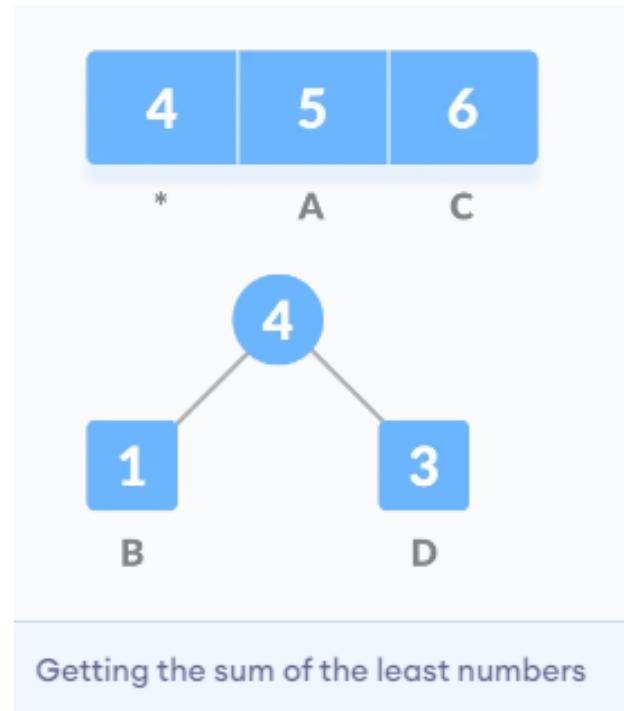
1	6	5	3
B	C	A	D
Frequency of string			

- Sort the characters in increasing order of the frequency. These are stored in a priority queue Q.

1	3	5	6
B	D	A	C
Characters sorted according to the frequency			

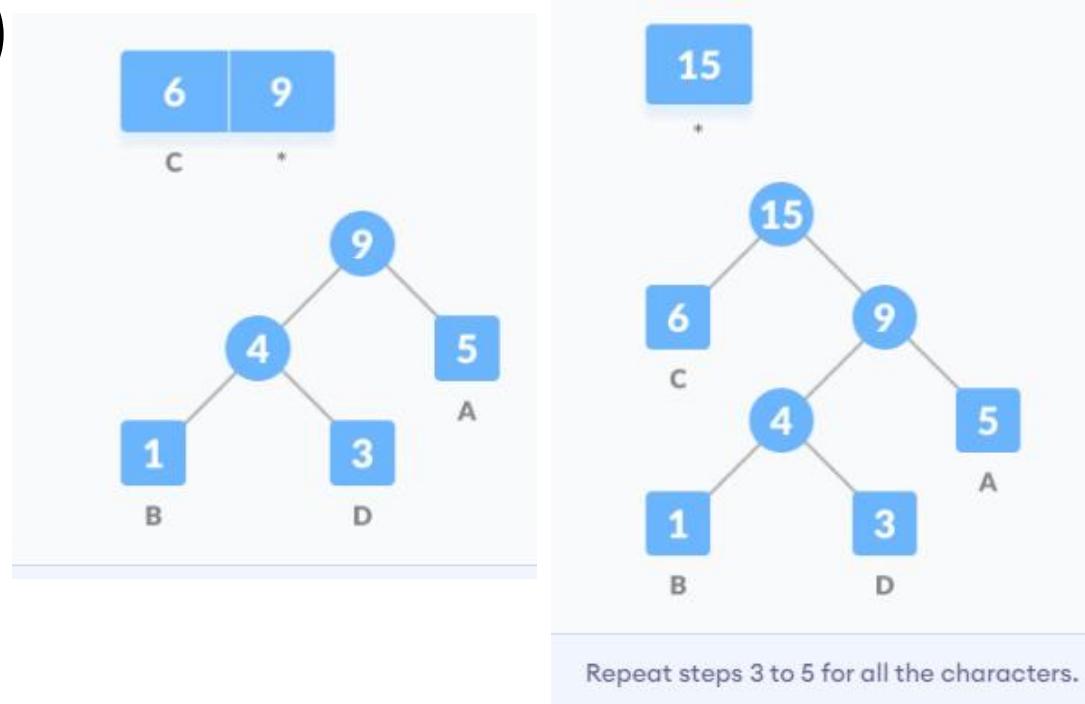
Huffman Encoding

- Make each unique character as a leaf node.
- Create an empty node z. Assign the minimum frequency to the left child of z and assign the second minimum frequency to the right child of z. Set the value of the z as the sum of the above two minimum frequencies.
Getting the sum of the least numbers



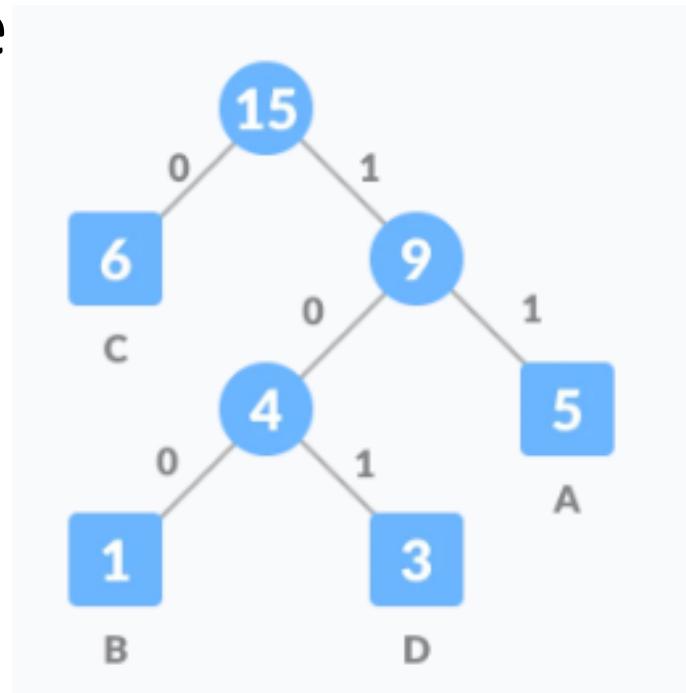
Huffman Encoding

- Remove these two minimum frequencies from Q and add the sum into the list of frequencies (* denote the internal nodes in the figure above)
- Insert node z into the tree.



Huffman Encoding

- For each non-leaf node, assign 0 to the left edge and 1 to the right edge
- Assign 0 to the left edge and 1 to the right edge



Character	Frequency	Code	Size
A	5	11	$5*2 = 10$
B	1	100	$1*3 = 3$
C	6	0	$6*1 = 6$
D	3	101	$3*3 = 9$
$4 * 8 = 32$ bits	15 bits		28 bits

Without encoding, the total size of the string was 120 bits. After encoding the size is reduced to $32 + 15 + 28 = 75$.

Huffman Coding

Example:

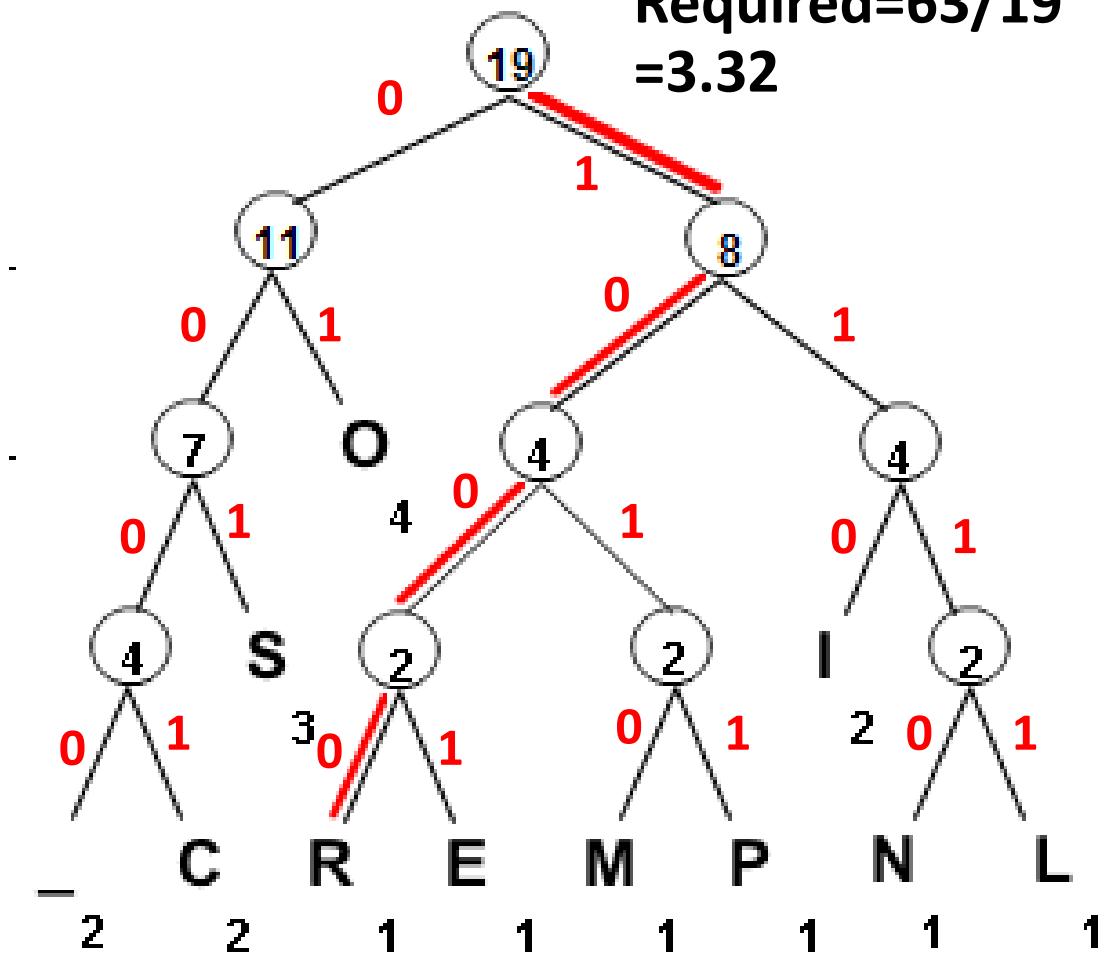
"COMPRESSION_IS_COOL"

Here,
 $M=1, P=1, R=1, N=1, L=1$
 $E=1, C=2, _ =2, I=2, S=3, O=4,$
Total Characters=19
 $19 \times 7 = 133$ bits

Here,
 $C=0001=4 \quad 4 \times 2 = 8$
 $O=01=2 \quad 2 \times 4 = 8$
 $M=1010=4 \quad 4 \times 1 = 4$
 $P=1011=4 \quad 4 \times 1 = 4$
 $R=1000=4 \quad 4 \times 1 = 4$
 $E=1001=4 \quad 4 \times 1 = 4$
 $S=001=3 \quad 3 \times 3 = 9$
 $I=110=3 \quad 3 \times 2 = 6$
 $N=1110=4 \quad 4 \times 1 = 4$
 $_ =0000=4 \quad 4 \times 2 = 8$
 $L=1111=4 \quad 4 \times 1 = 4$

Total Bits Required 63 bits

Average bits
Required = $63/19$
= 3.32



Lempel ZIV(LZ) Algorithm

Let we have data

101011011010101010

1,0,10,11,01,101,010,1010

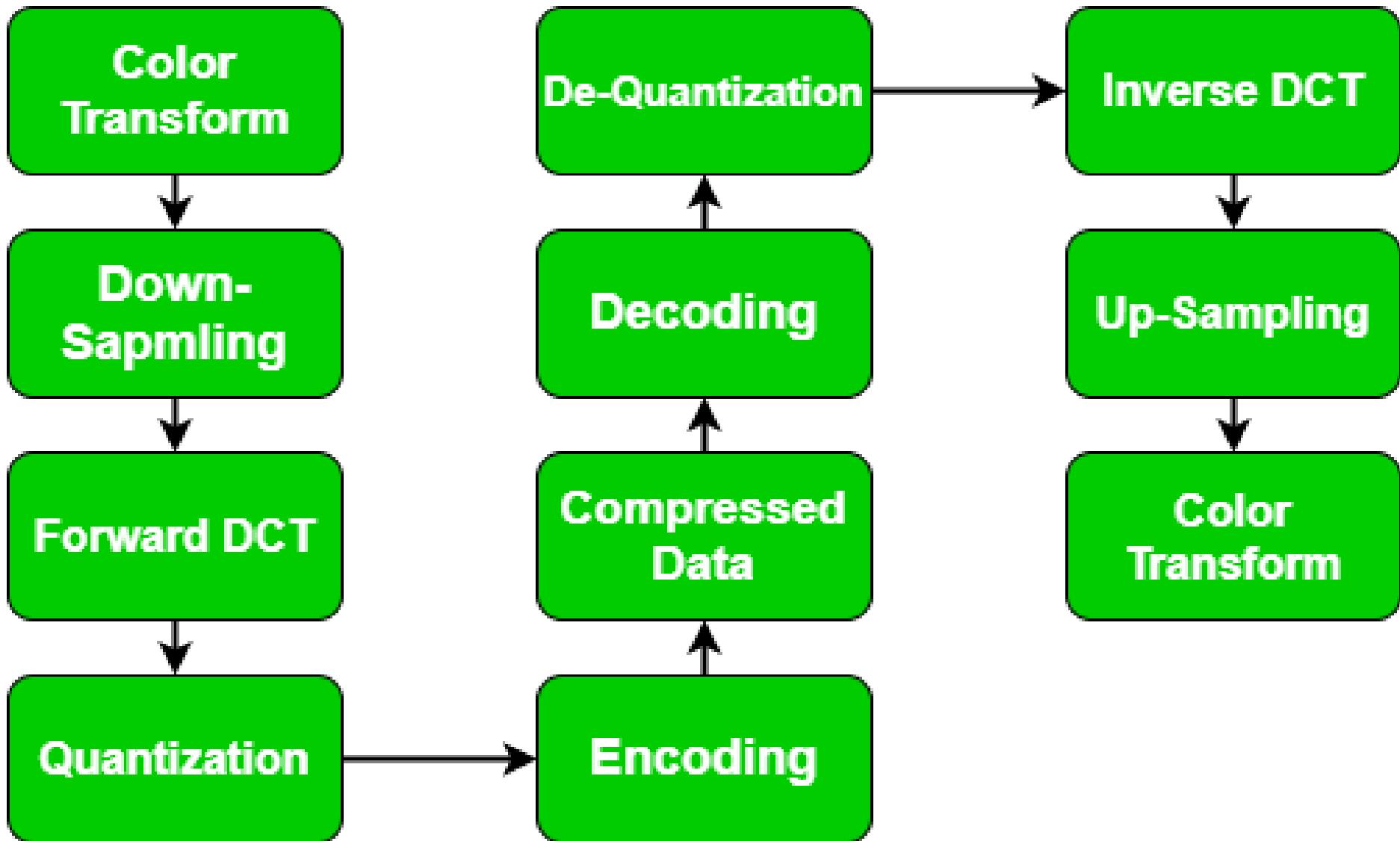
Dictionary Location	Content	Code word
001	1	0001
010	0	0000
011	10	0010
100	11	0011
101	01	0101
110	101	0111
111	010	1010
	1010	1100

Required Digital Code:

00010000001000110101011110101100

Image Compression (JPEG)

- JPEG, GIF and MPEG are more than compression algorithms, they also define the format for image or video data in the same way as XDR, NDR and ANS.1 define the format for numeric and string data.
- **JPEG** -Named after the Joint Photographic Experts Group that designed it.
- Used for digital imagery (ISO standard)
- Compression takes place in 3 phases: a discrete cosine transformation (DCT), quantization and encoding



Algorithm of JPEG Data Compression :

1. Splitting -

We split our image into the blocks of 8×8 blocks. It forms 64 blocks in which each block is referred to as 1 pixel.

2. Color Space Transform -

In this phase, we convert R, G, B to Y, Cb, Cr model. Here Y is for brightness, Cb is color blueness and Cr stands for Color redness. We transform it into chromium colors as these are less sensitive to human eyes thus can be removed.

3. Apply DCT -

We apply Direct cosine transform on each block. The discrete cosine transform (DCT) represents an image as a sum of sinusoids of varying magnitudes and frequencies.

4. Quantization -

In the Quantization process, we quantize our data using the quantization table.

5. Serialization -

In serialization, we perform the zig-zag scanning pattern to exploit redundancy.

6. Vectoring -

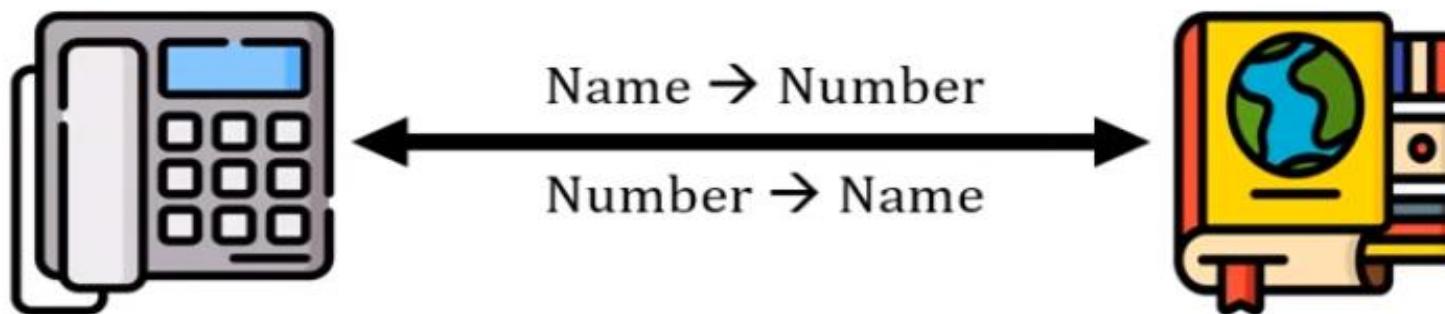
We apply DPCM (differential pulse code modeling) on DC elements. DC elements are used to define the strength of colors.

7. Encoding -

In the last stage, we apply to encode either run-length encoding or Huffman encoding. The main aim is to convert the image into text and by applying any encoding we convert it into binary form (0, 1) to compress the data.

DNS Introduction

- **Domain or Domain name** is the location of a website.
- **DNS** is directory(phonebook) of internet.

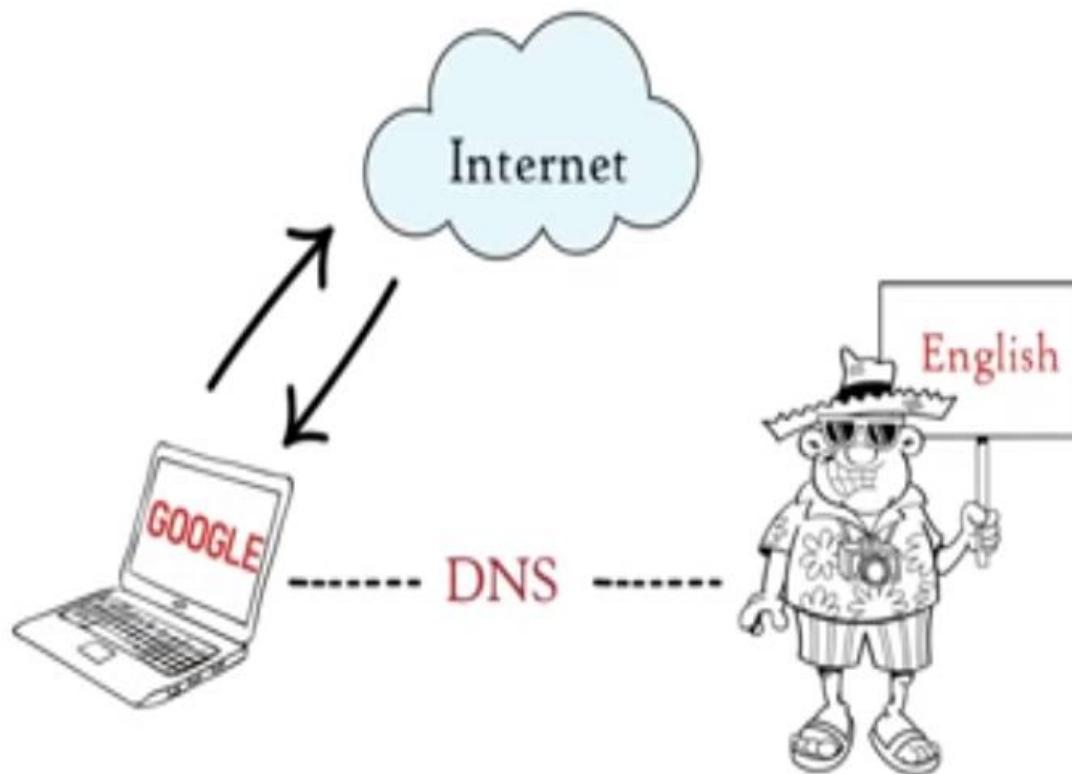


- Connecting web browsers with web servers.



- IP Address → Naming || Naming → IP Address





Domain Name System



Types of DNS Servers:



DNS recursive resolver/DNS resolver



Root name server



Top Level Domain/TLD name server



Authoritative name server



Root name server

13 sets ←

letter.root-servers.net

www.root-servers.org

operated by: 12 organizations

info page: letter.root-servers.org

letter: 'a' to 'm'

for 'g': <https://disa.mil/g-root>



TLD name server

.com TLD name server

.net TLD name server



domains: .com, .net, .in, .edu



websites: .com extension



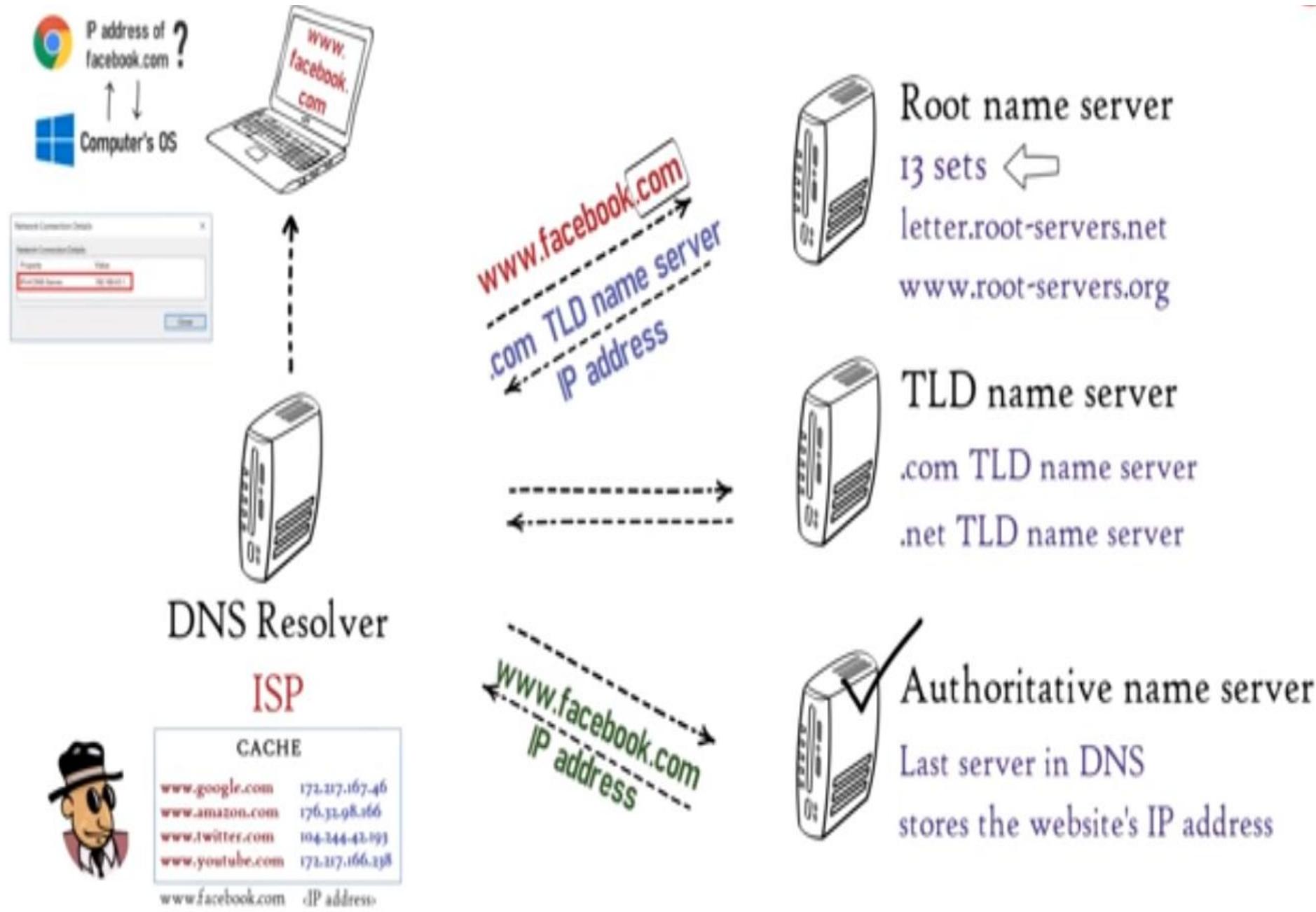
websites: .net extension



Authoritative name server

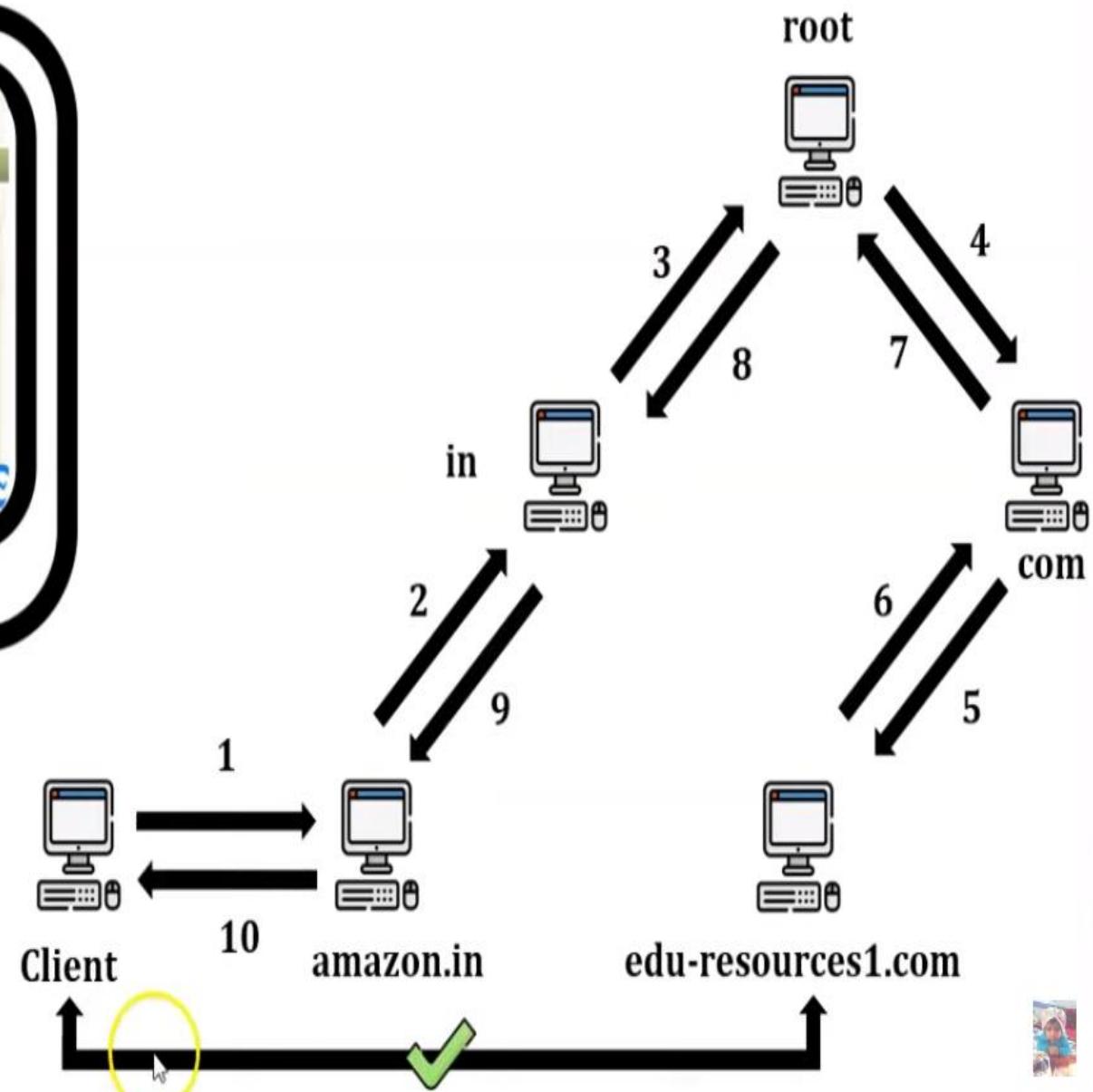
Last server in DNS

stores the website's IP address



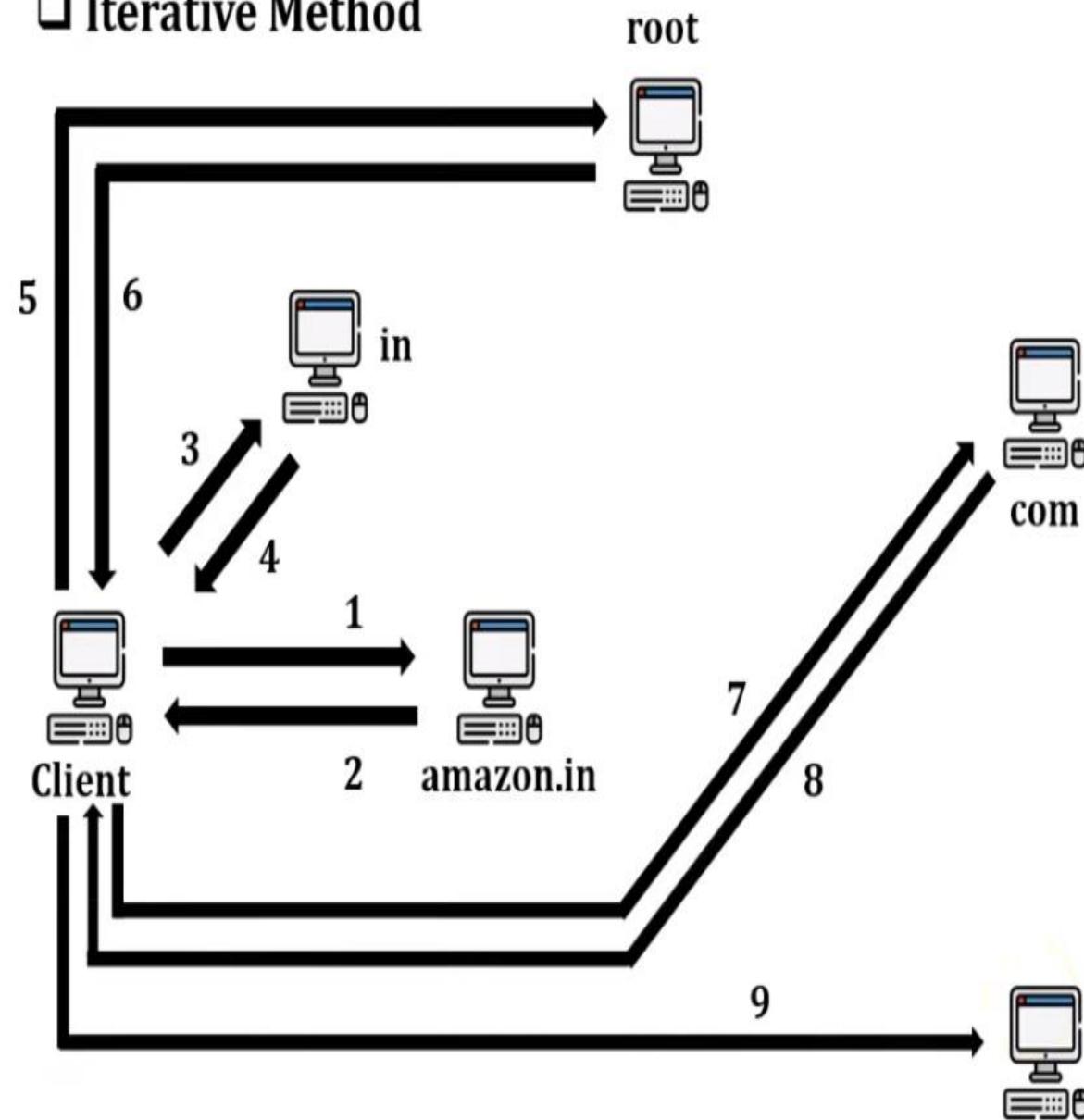
Name Resolver

□ Recursive Method

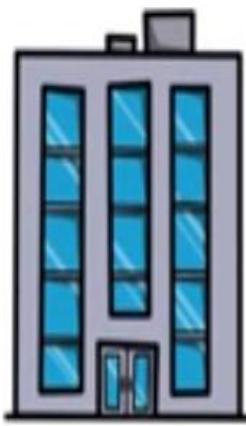


Name Resolver

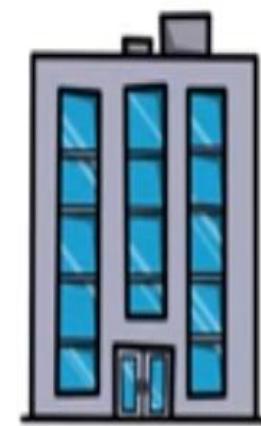
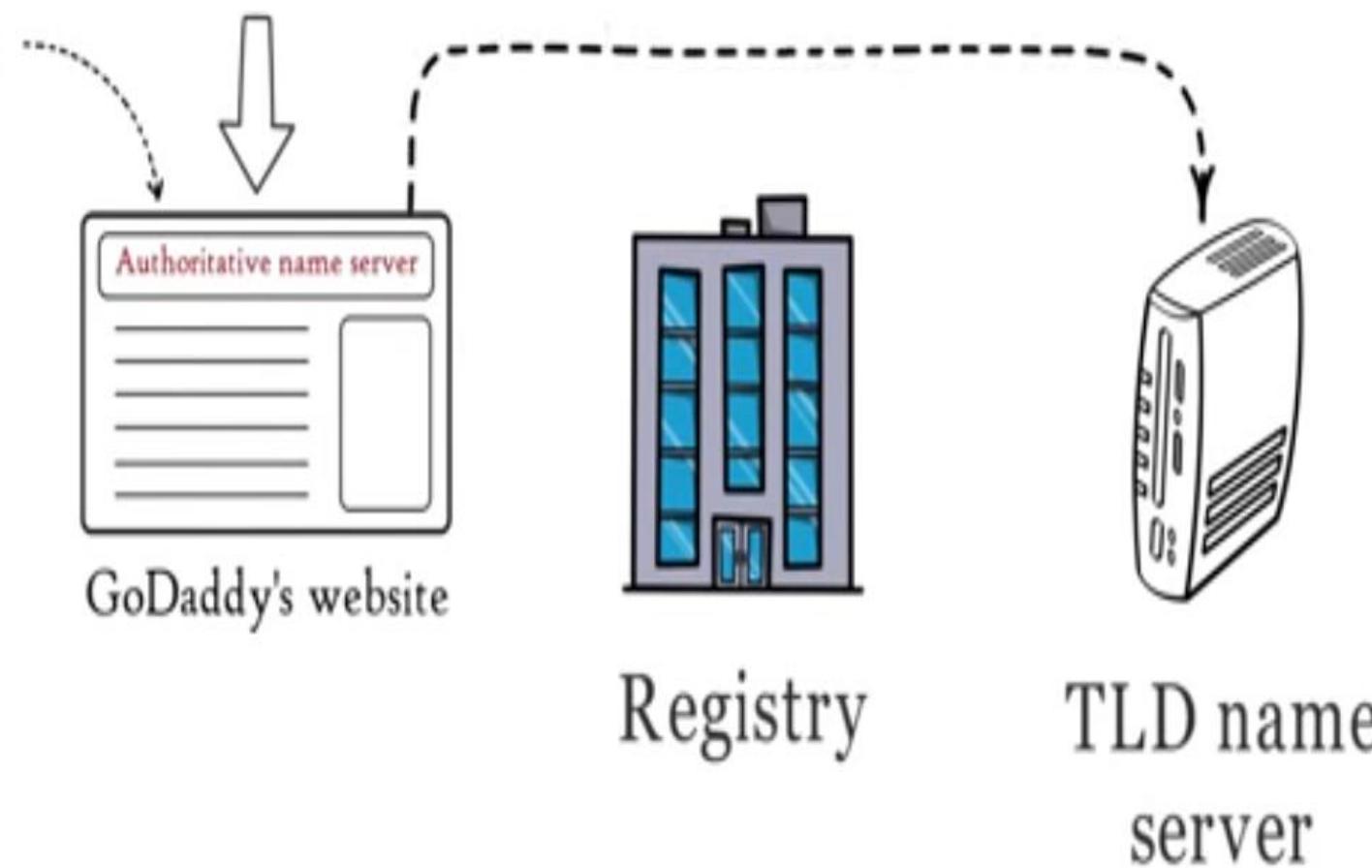
□ Iterative Method



www.example.com



GoDaddy
(Registrar)



TLD name
server

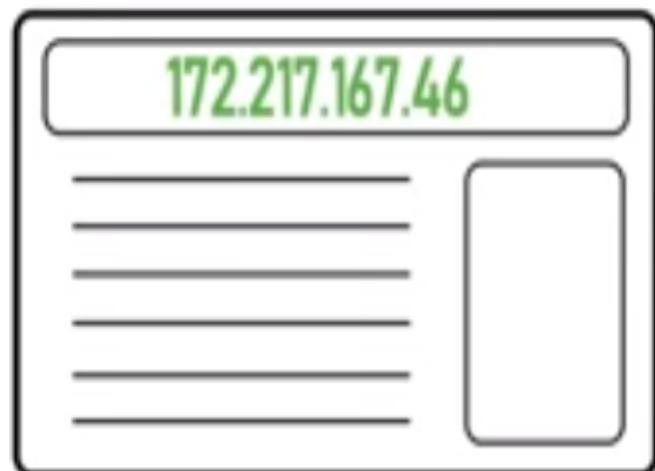
Is DNS necessary?

www.google.com



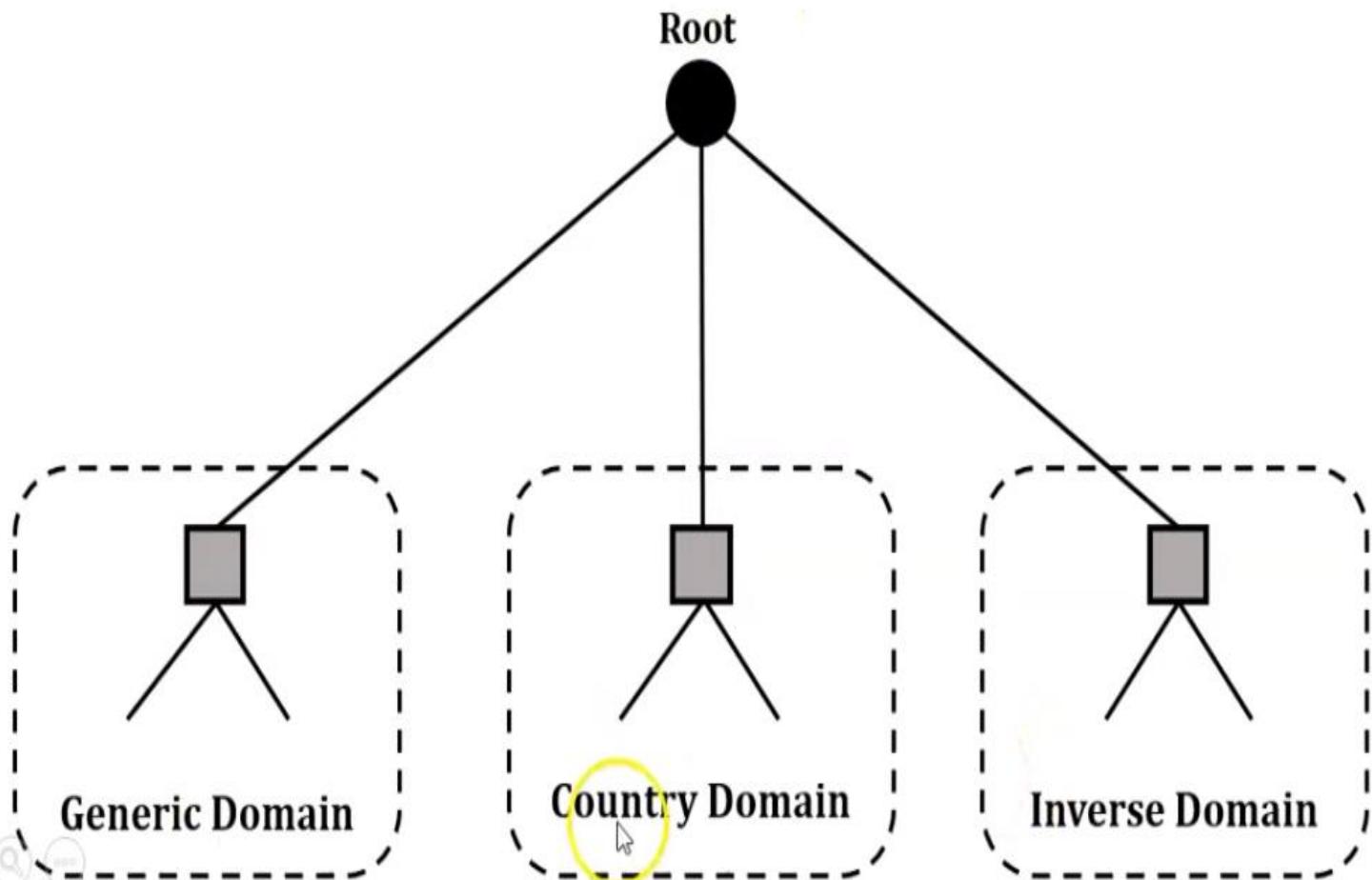
172.217.167.46

Web Browser



Domain Name

- Domain name is divided into main three different categories in the internet.
- i.e., Generic domain, Country domain and Inverse domain.
- Each node in tree defines a domain, which is an index to the domain name space database.



Domain Name

□ Generic Domain

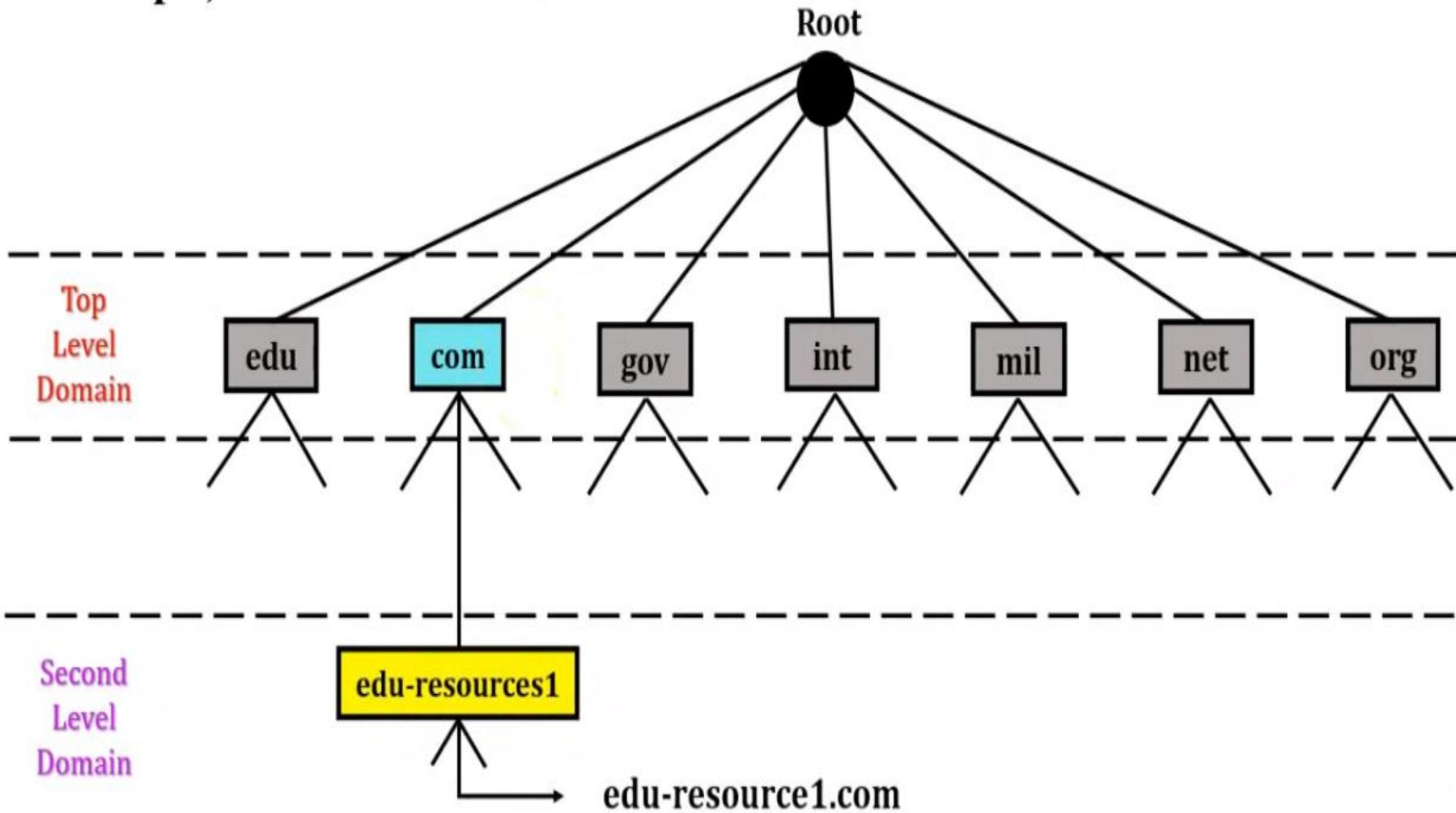
- The 3 character domains are called the *generic domains*.
- Generic domain labels are as follows

Sr. No.	Label	Description
1.	.com	Commercial Organization
2.	.edu	Educational Organization
3.	.gov	Government Organization
4.	.int	International Organization
5.	.mil	Military Group
6.	.net	Network Support Centers
7.	.org	Nonprofit Organization

Domain Name

❑ Generic Domain

- For example, edu-resource1.com



Domain Name

□ Country Domain

- The 2 character domains are called the *country domains*.

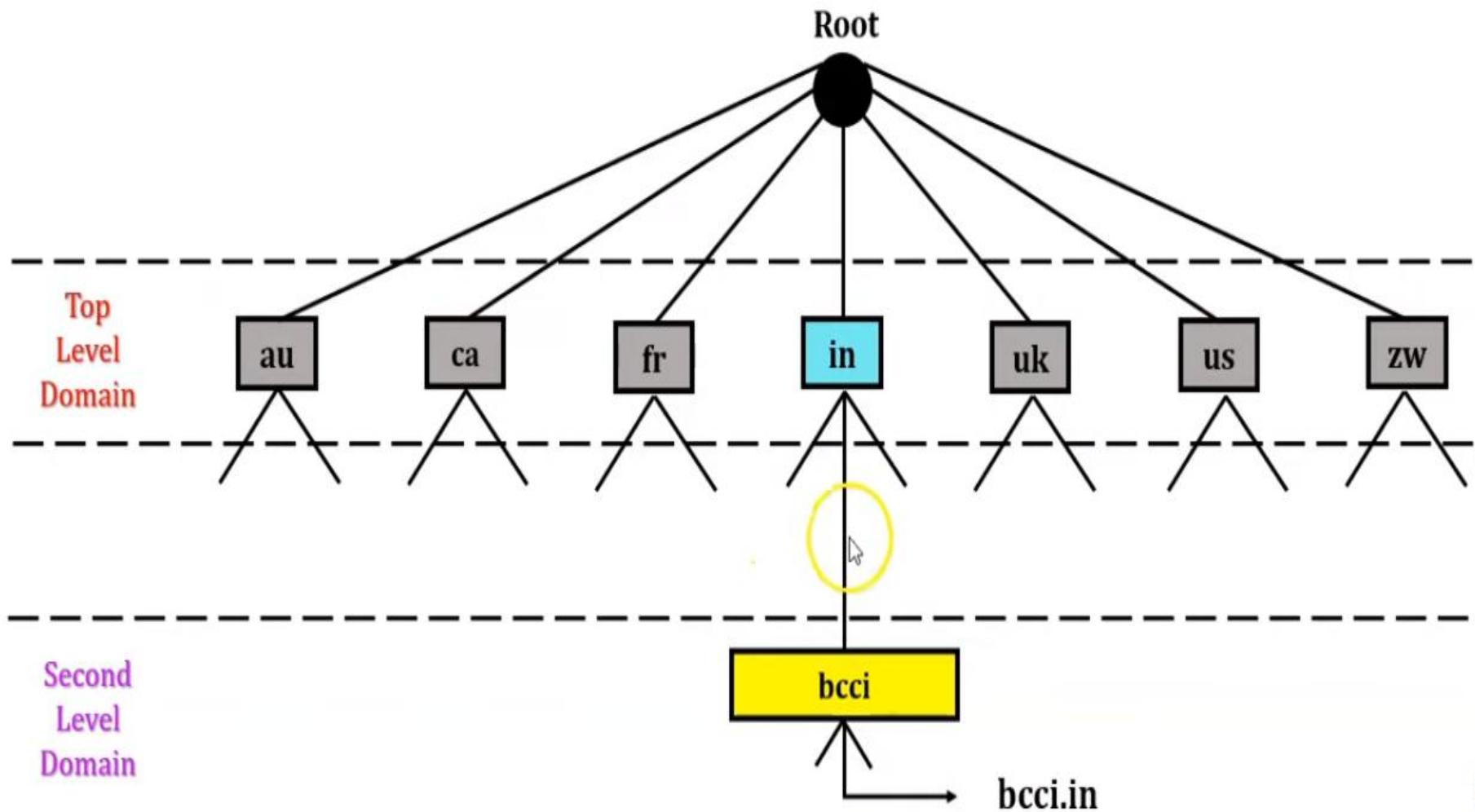
Sr. No.	Label	Country Name
1.	.at	Austria
2.	.au	Australia
3.	.bb	Barbados
4.	.be	Belgium
5.	.br	Brazil
6.	.ca	Canada
7.	.ch	Switzerland
8.	.cn	China
9.	.de	Germany
10.	.es	Spain
11.	.fr	France
12.	.hk	Hong Kong

Sr. No.	Label	Country Name
13.	.in	India
14.	.il	Israel
15.	.it	Italy
16.	.jp	Japan
17.	.kr	South Korea
18.	.nz	New Zealand
19.	.pt	Portugal
20.	.qa	Qatar
21.	.ru	Russia
22.	.us	United States
23.	.uk	United Kingdom
24.	.zw	Zimbabwe

Domain Name

□ Country Domain

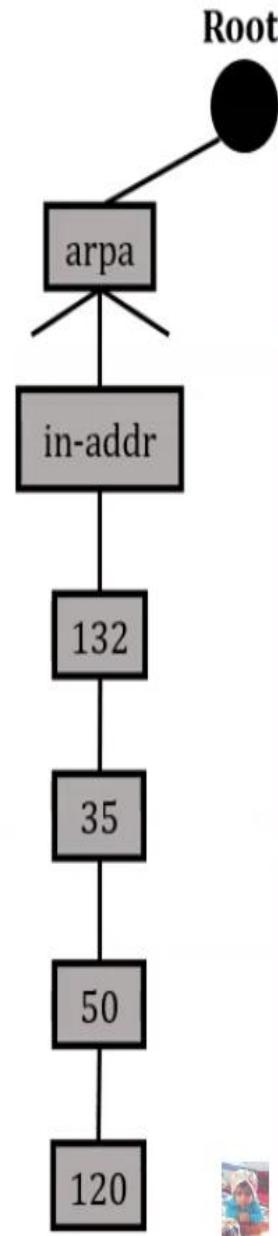
- For example, bcci.in



Domain Name

□ Inverse Domain

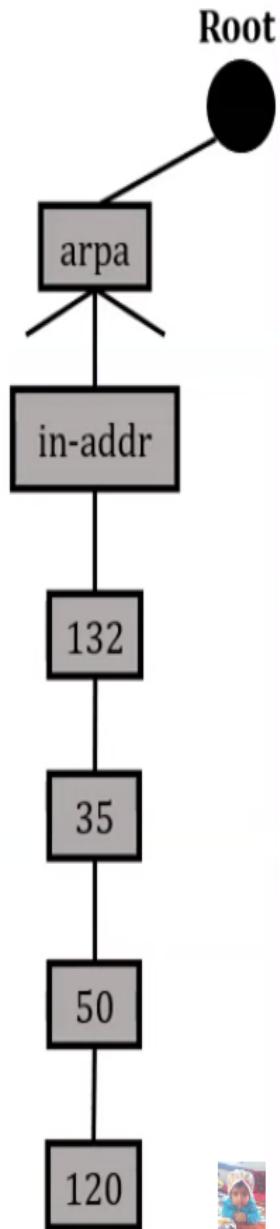
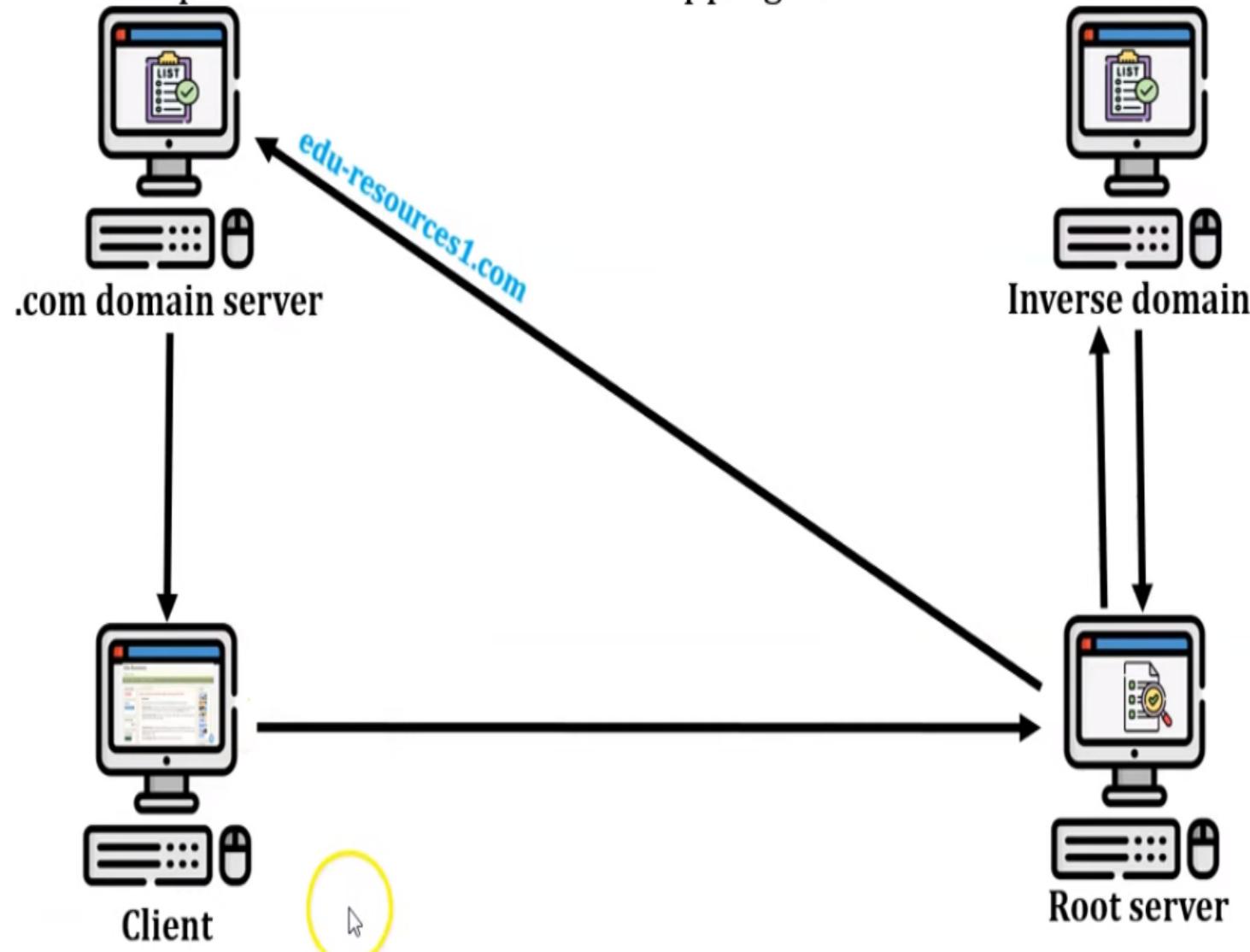
- Purpose of inverse domain is mapping an address to a name.
- Example: When a client send a request to the server for doing particular task, server finds the list of authorized client.
- The list contains only IP address of the client.
- Server send a query to the inverse DNS server and ask for a mapping address to name for authorized client list.
- The above query is called an inverse or pointer query.
- The pointer query is handled by the first level node called arpa. The second level is also one single node named in-addr. The rest of the domain defines IP addresses.



Domain Name

□ Inverse Domain

- Purpose of inverse domain is mapping an address to a name.



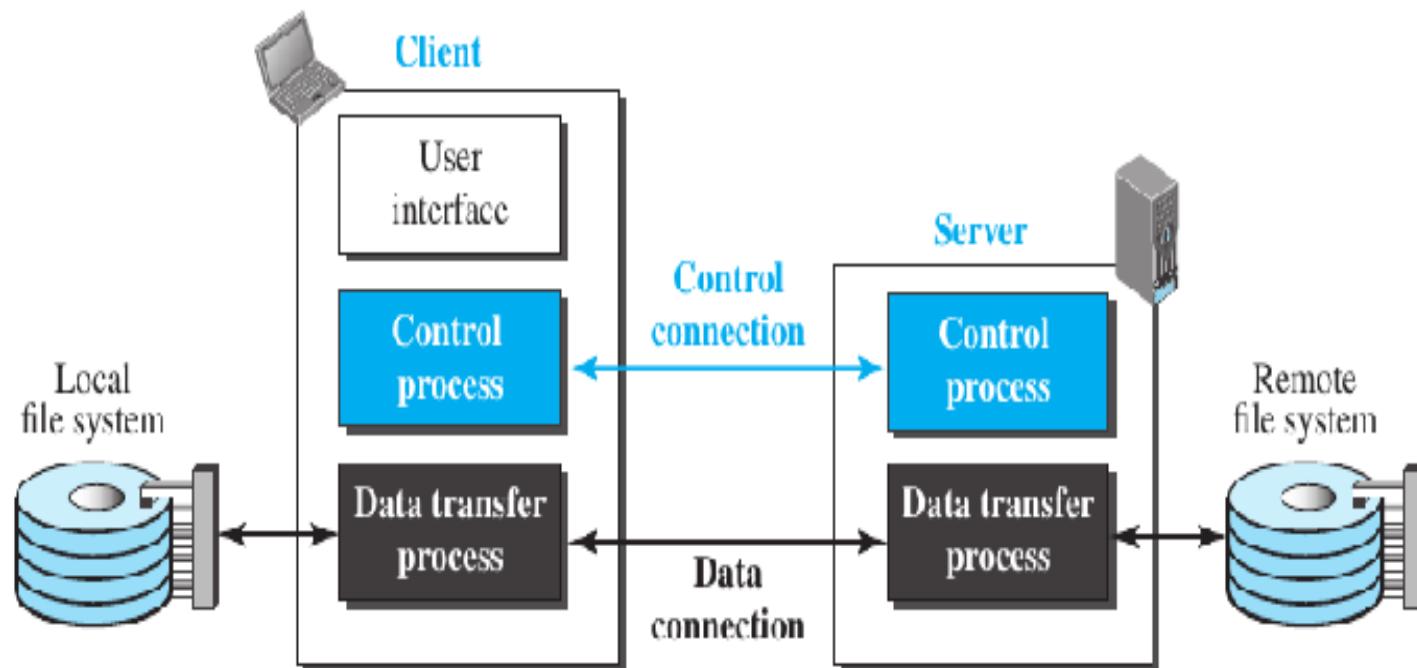
File Transfer

Transfer of data from Sender to Receiver in OSI Model is possible using FTP Protocol

FILE TRANSFER PROTOCOL (FTP)

File Transfer Protocol (FTP) is the standard mechanism provided by TCP/IP for copying a file from one host to another.

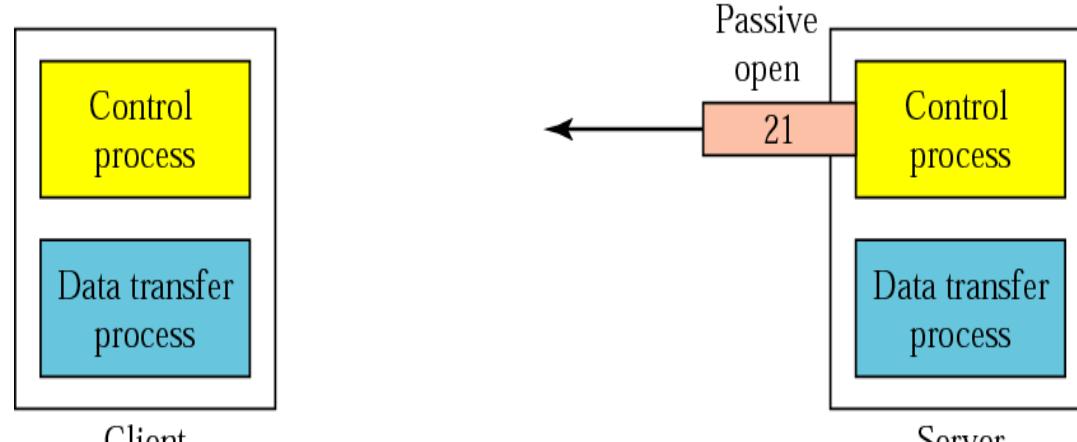
FTP



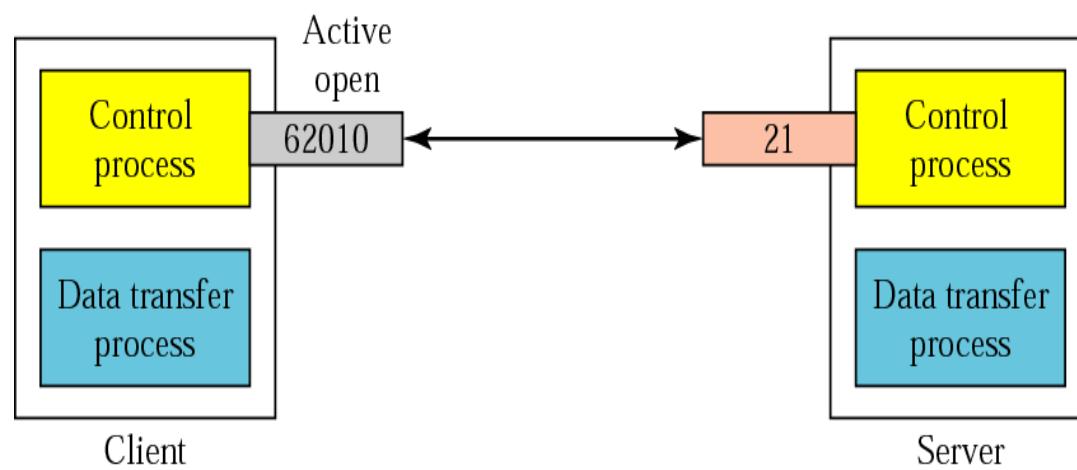
*FTP uses the services of TCP.
It needs two TCP
connections.*

*The well-known port 21 is
used for the control
connection and the well-
known port 20 for the data
connection.*

Opening the control connection

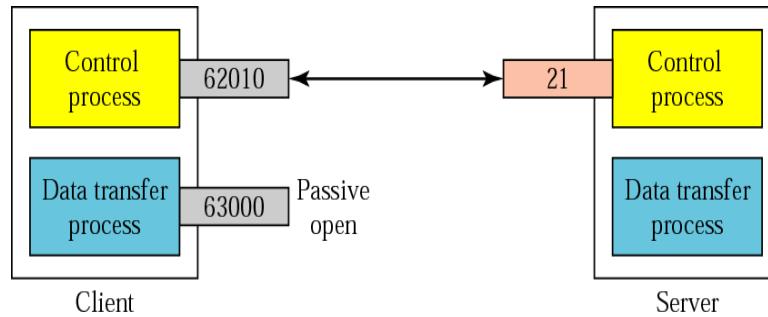


a. Passive open by server

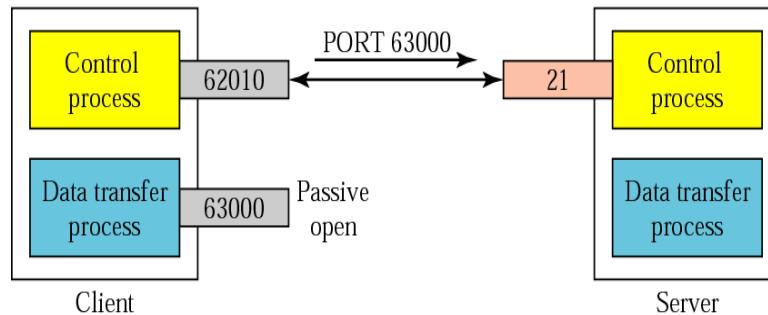


b. Active open by client

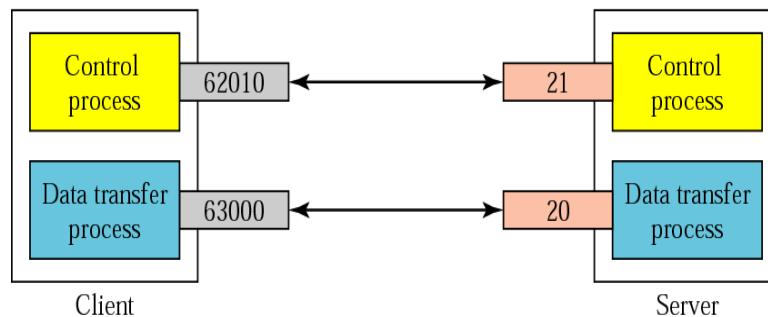
Creating the data connection



a. Passive open by client

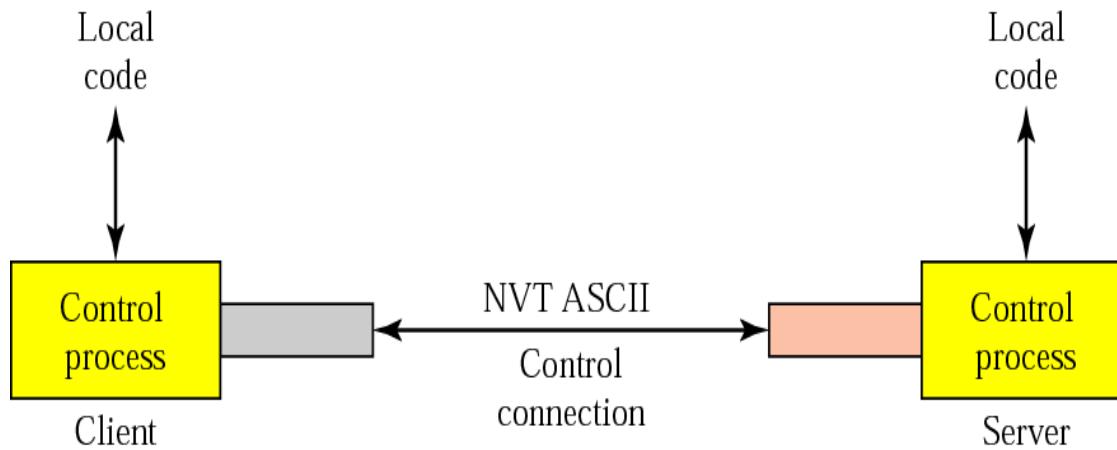


b. Sending ephemeral port number to server

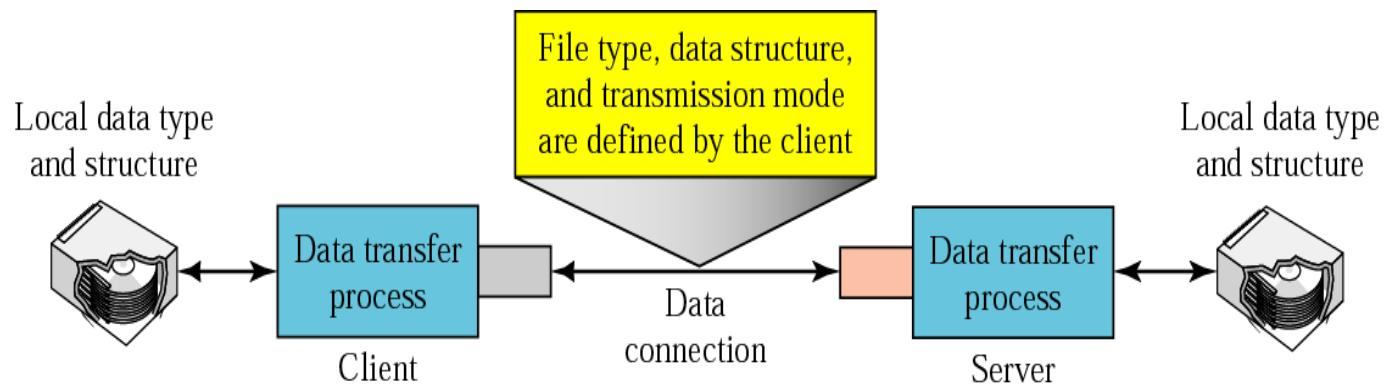


c. Active open by server

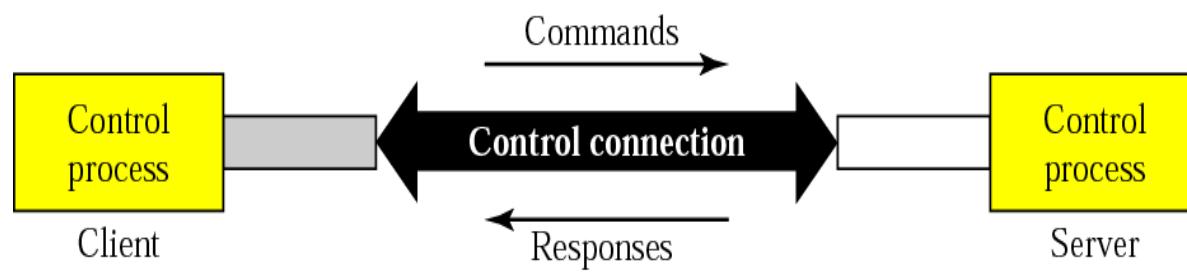
Using the control connection



Using the data connection



Command processing



Access commands

<i>Command</i>	<i>Argument(s)</i>	<i>Description</i>
USER	User id	User information
PASS	User password	Password
ACCT	Account to be charged	Account information
REIN		Reinitialize
QUIT		Log out of the system
ABOR		Abort the previous command

File management commands

<i>Command</i>	<i>Argument(s)</i>	<i>Description</i>
CWD	Directory name	Change to another directory
CDUP		Change to the parent directory
DELE	File name	Delete a file
LIST	Directory name	List subdirectories or files
NLIST	Directory name	List the names of subdirectories or files without other attributes
MKD	Directory name	Create a new directory
PWD		Display name of current directory
RMD	Directory name	Delete a directory
RNFR	File name (old file name)	Identify a file to be renamed
RNTO	File name (new file name)	Rename the file
SMNT	File system name	Mount a file system

Data formatting commands

<i>Command</i>	<i>Argument(s)</i>	<i>Description</i>
TYPE	A (ASCII), E (EBCDIC), I (Image), N (Nonprint), or T (TELNET)	Define the file type and if necessary the print format
STRU	F (File), R (Record), or P (Page)	Define the organization of the data
MODE	S (Stream), B (Block), or C (Compressed)	Define the transmission mode

Port defining commands

<i>Command</i>	<i>Argument(s)</i>	<i>Description</i>
PORT	6-digit identifier	Client chooses a port
PASV		Server chooses a port

File transfer commands

<i>Command</i>	<i>Argument(s)</i>	<i>Description</i>
RETR	File name(s)	Retrieve files; file(s) are transferred from server to the client
STOR	File name(s)	Store files; file(s) are transferred from the client to the server
APPE	File name(s)	Similar to STOR except if the file exists, data must be appended to it
STOU	File name(s)	Same as STOR except that the file name will be unique in the directory; however, the existing file should not be overwritten

File transfer commands (continued)

<i>Command</i>	<i>Argument(s)</i>	<i>Description</i>
ALLO	File name(s)	Allocate storage space for the files at the server
REST	File name(s)	Position the file marker at a specified data point
STAT	File name(s)	Return the status of files

Miscellaneous commands

<i>Command</i>	<i>Argument(s)</i>	<i>Description</i>
HELP		Ask information about the server
NOOP		Check if server is alive
SITE	Commands	Specify the site-specific commands
SYST		Ask about operating system used by the server

Responses

<i>Code</i>	<i>Description</i>
Positive Preliminary Reply	
120	Service will be ready shortly
125	Data connection open; data transfer will start shortly
150	File status is OK; data connection will be open shortly

Responses (continued)

<i>Code</i>	<i>Description</i>
Positive Completion Reply	
200	Command OK
211	System status or help reply
212	Directory status
213	File status
214	Help message
215	Naming the system type (operating system)
220	Service ready
221	Service closing
225	Data connection open
226	Closing data connection
227	Entering passive mode; server sends its IP address and port number
230	User login OK
250	Request file action OK

Responses (continued)

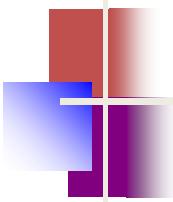
<i>Code</i>	<i>Description</i>
Positive Intermediate Reply	
331	User name OK; password is needed
332	Need account for logging
350	The file action is pending; more information needed

Responses (continued)

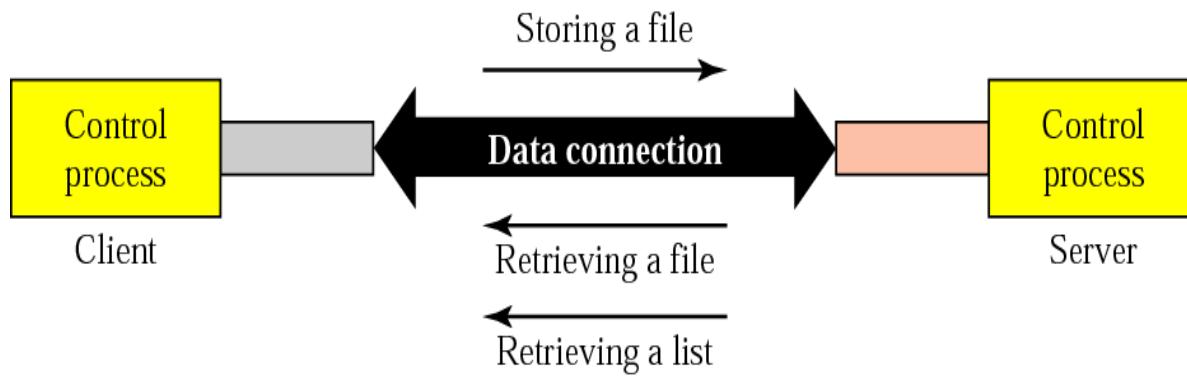
<i>Code</i>	<i>Description</i>
Transient Negative Completion Reply	
425	Cannot open data connection
426	Connection closed; transfer aborted
450	File action not taken; file not available
451	Action aborted; local error
452	Action aborted; insufficient storage
Permanent Negative Completion Reply	
500	Syntax error; unrecognized command

Responses (continued)

<i>Code</i>	<i>Description</i>
501	Syntax error in parameters or arguments
502	Command not implemented
503	Bad sequence of commands
504	Command parameter not implemented
530	User not logged in
532	Need account for storing file
550	Action is not done; file unavailable
552	Requested action aborted; exceeded storage allocation
553	Requested action not taken; file name not allowed



File transfer



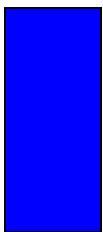


EXAMPLE

1

Next Figure shows an example of using FTP for retrieving a list of items in a directory.

- 1. After the control connection to port 21 is created, the FTP server sends the 220 (service ready) response on the control connection.*
- 2. The client sends the USER command.*
- 3. The server responds with 331 (user name is OK, password is required).*
- 4. The client sends the PASS command.*
- 5. The server responds with 230 (user login is OK)*



EXAMPLE 1 (CONTINUED)

- 6. The client issues a passive open on an ephemeral port for the data connection and sends the PORT command (over the control connection) to give this port number to the server.*
- 7. The server does not open the connection at this time, but it prepares itself for issuing an active open on the data connection between port 20 (server side) and the ephemeral port received from the client. It sends response 150 (data*



EXAMPLE 1 (CONTINUED)

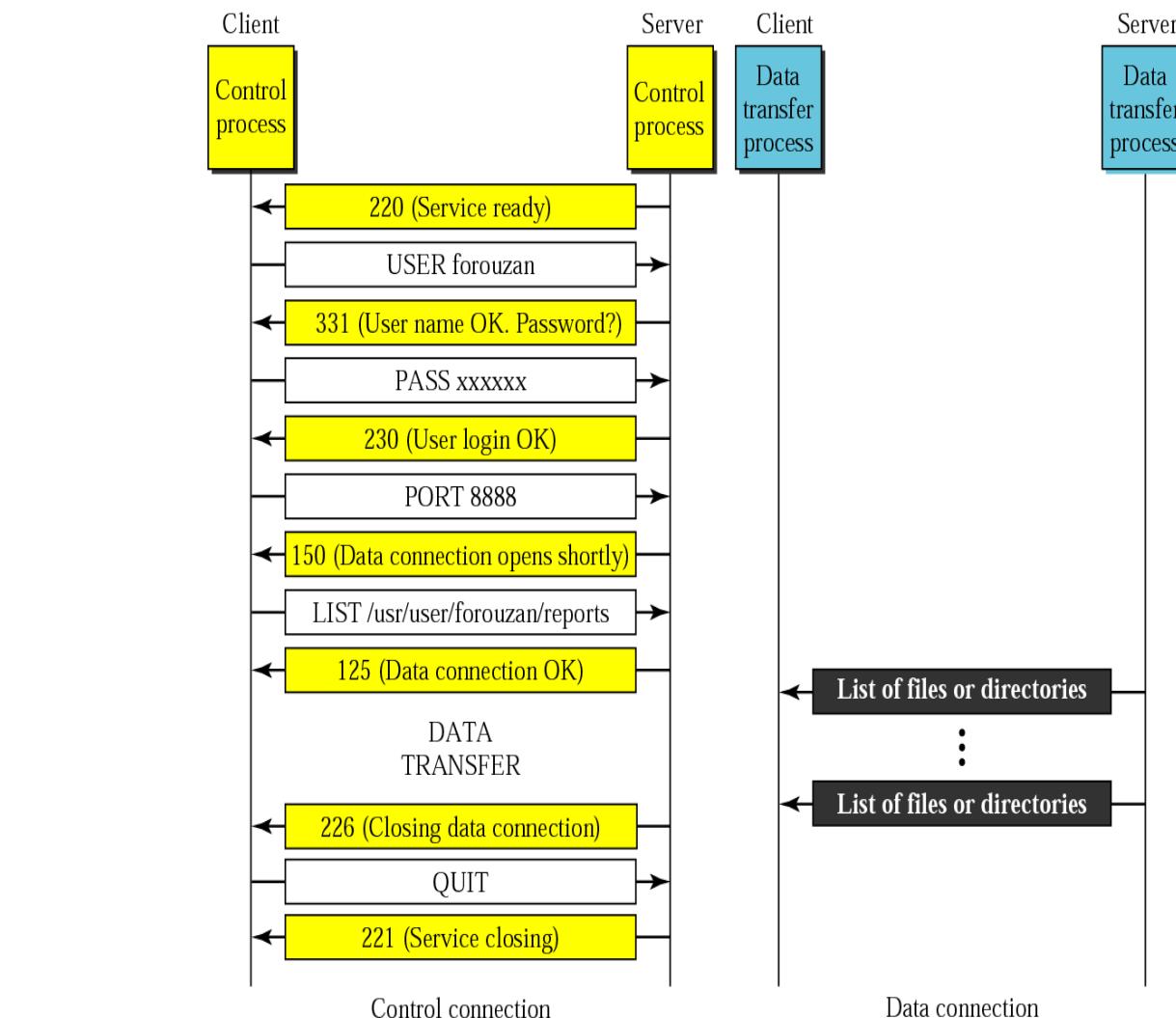
10. The server then sends the list of the files or directories (as a file) on the data connection. When the whole list (file) is sent, the server responds with 226 (closing data connection) over the control connection.

11. The client now has two choices. It can use the QUIT

command to request the closing of the control connection or it can send another command to

start another activity (and eventually open another data connection). In our example, the client has sent the QUIT

Figure 19.8 Example 1



EXAMPLE 2

The following shows an actual FTP session that parallels Example 1. The colored lines show the responses from the server control connection; the black lines show the commands sent by the client. The lines in white with black background shows data transfer

```
$ ftp voyager.deanza.fhda.edu
Connected to voyager.deanza.fhda.edu.
220 (vsFTPd 1.2.1)
530 Please login with USER and PASS.
Name (voyager.deanza.fhda.edu:forouzan): forouzan
331 Please specify the password.
```

EXAMPLE 2

Password:

230 Login successful.

Remote system type is UNIX.

Using binary mode to transfer files.

ftp> ls reports

227 Entering Passive Mode (153,18,17,11,238,169)

150 Here comes the directory listing.

```
drwxr-xr-x 2 3027 411 4096 Sep 24 2002 business
drwxr-xr-x 2 3027 411 4096 Sep 24 2002 personal
drwxr-xr-x 2 3027 411 4096 Sep 24 2002 school
```

226 Directory send OK.

ftp> quit

221 Goodbye.



EXAMPLE

3

Next Figure shows an example of how an image (binary) file is stored.

1. After the control connection to port 21 is created, the FTP

server sends the 220 (service ready) response on the control connection.

2. The client sends the USER command.

3. The server responds with 331 (user name is OK, a password is required).

4. The client sends the PASS command.

5. The server responds with 230 (user login is OK).

6. The client issues a passive open on an ephemeral port for the data connection and sends the PORT

command (over



EXAMPLE 3 (CONTINUED)

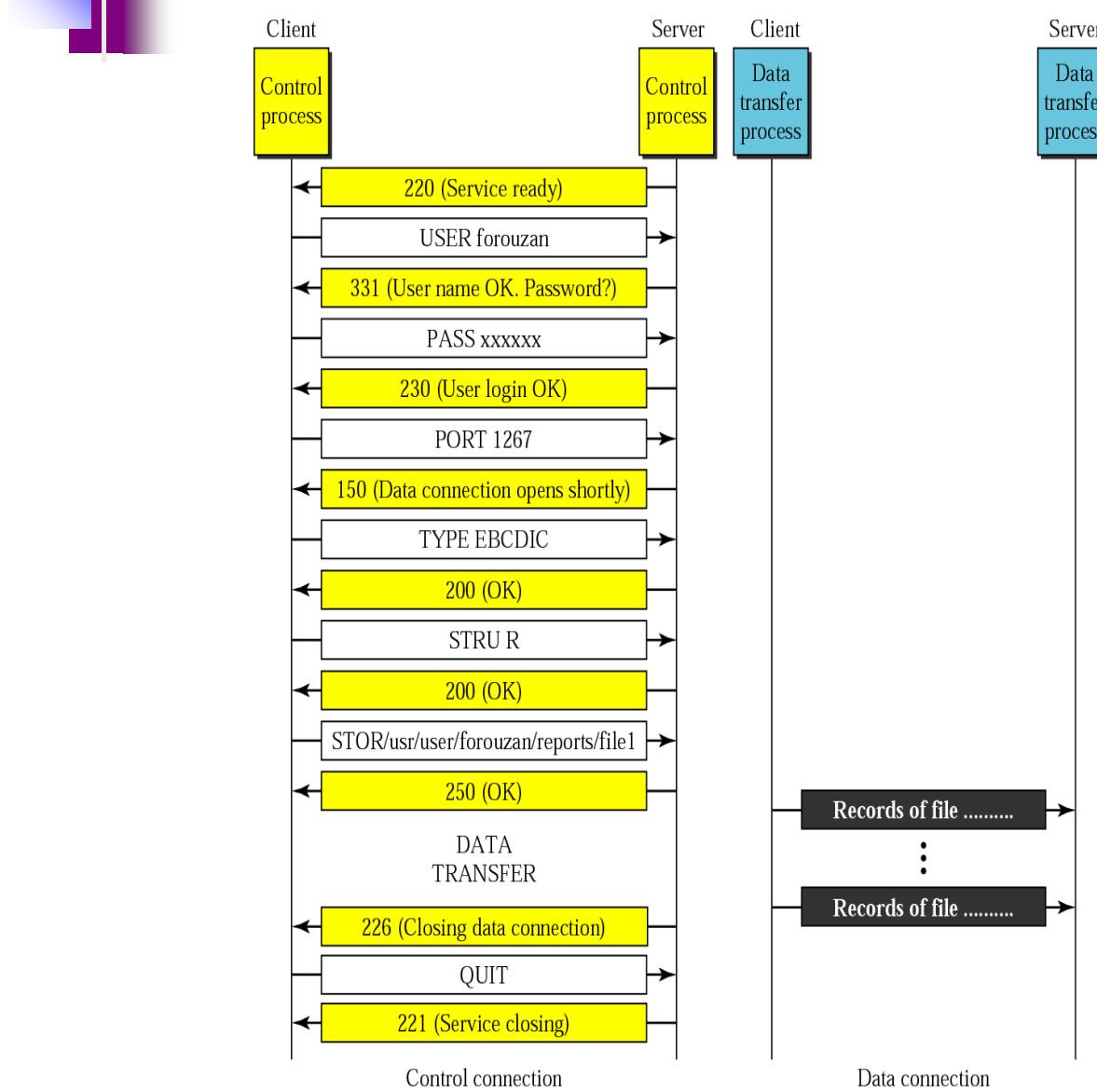
7. *The server does not open the connection at this time, but prepares itself for issuing an active open on the data connection between port 20 (server side) and the ephemeral port received from the client. It sends the response 150 (data connection will open shortly).*
8. *The client sends the TYPE command.*
9. *The server responds with the response 200 (command OK).*
10. *The client sends the STRU command.*
11. *The server responds with 200 (command OK).*
12. *The client sends the STOR command.*
13. *The server responds with 200 (command OK).*



EXAMPLE 3 (CONTINUED)

- 14. The client sends the file on the data connection. After the entire file is sent, the data connection is closed. Closing the data connection means end-of-file.*
- 15. The server sends the response 226 on the control connection.*
- 16. The client sends the QUIT command or uses other commands to open another data connection for transferring another file. In our example, the QUIT command is sent.*

Figure 19.9 Example 3



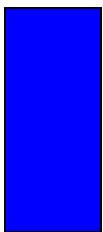


EXAMPLE

4

We show an example of anonymous FTP. We assume that some public data are available at internic.net.

```
$ ftp internic.net
Connected to internic.net
220 Server ready
Name: anonymous
331 Guest login OK, send "guest" as password
Password: guest
ftp > pwd
257 '/' is current directory
```



EXAMPLE

4

bin

... .

... .

... .

ftp > close

221 Goodbye

ftp > quit

Figure 17.19 *Resource record format*

