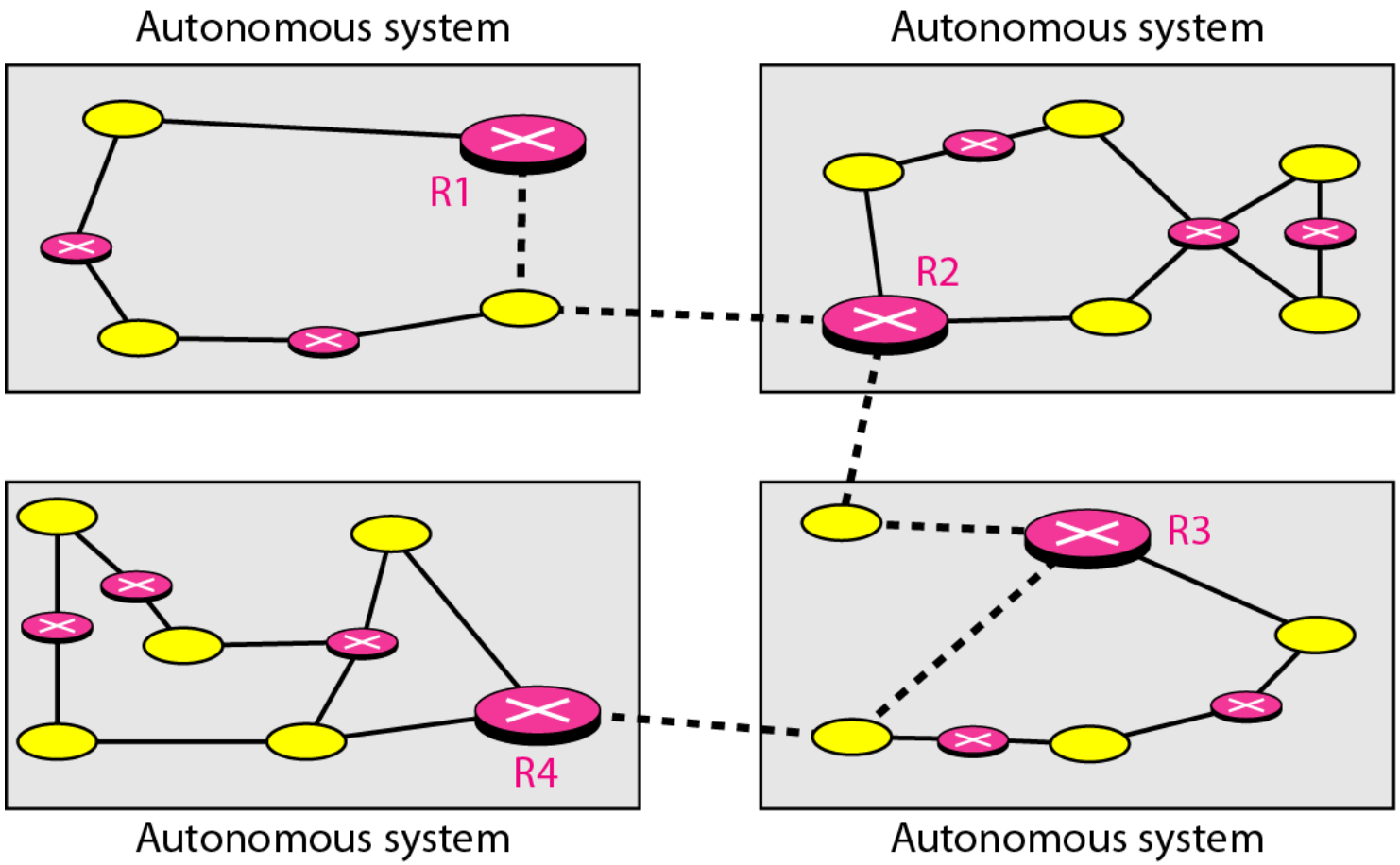
UNICAST ROUTING PROTOCOLS

A routing table can be either static or dynamic. A static table is one with manual entries. A dynamic table is one that is updated automatically when there is a change somewhere in the Internet. A routing protocol is a combination of rules and procedures that lets routers in the Internet inform each other of changes.

There can be several reasons to opt Routing Protocol, and they are as follows:

- It provides ease in the configuration.
- It is beneficial for the large size of network routing.
- It adapts to changes.
- It helps in minimizing and updating traffic.
- It has fast convergence.
- It allows optimal path selection.
- It allows loop-free routing.
- It is compatible with existing routers.

Figure 22.12 *Autonomous systems*



Autonomous system (AS) is a **large network or group of networks that has a unified routing policy**.

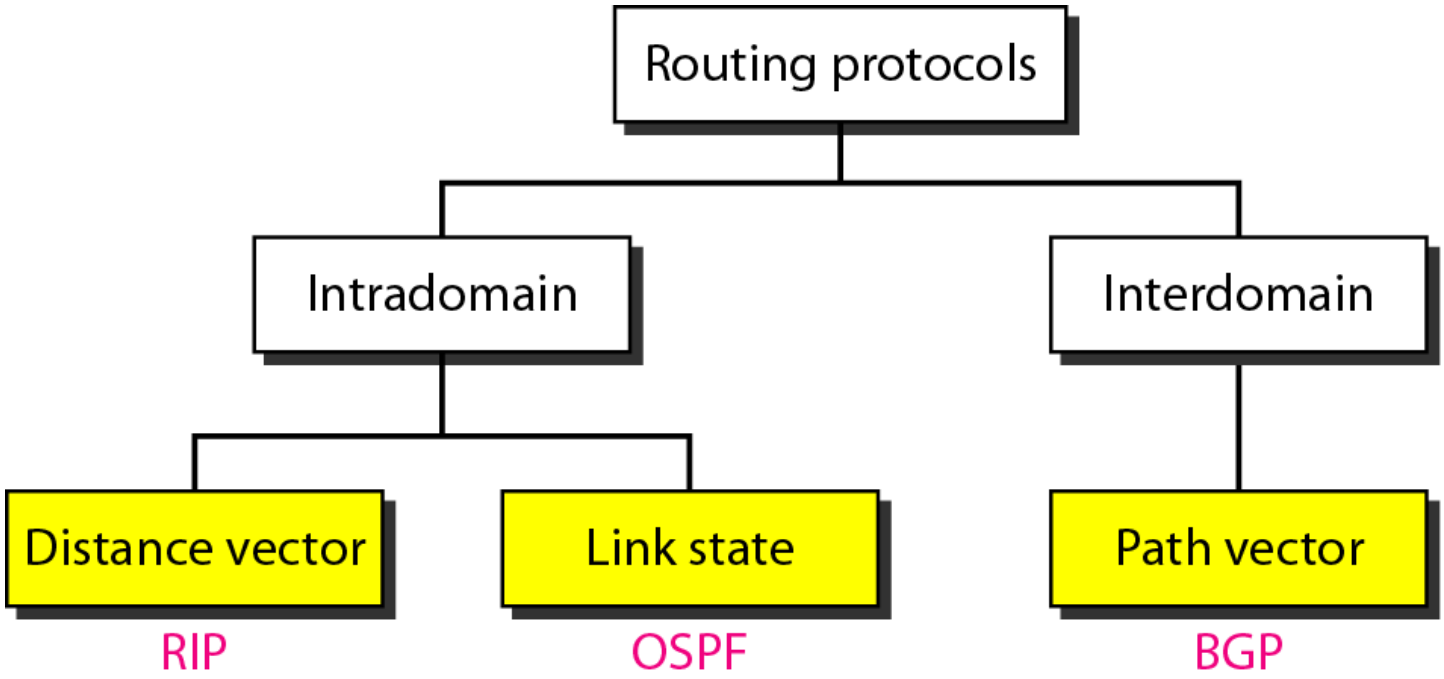
Imagine an AS as being like a town's post office. Mail goes from post office to post office until it reaches the right town, and that town's post office will then deliver the mail within that town. Similarly, data packets cross the Internet by hopping from AS to AS until they reach the AS that contains their destination Internet Protocol (IP) address. Routers within that AS send the packet to the IP address.

- When it receives a packet, to which network should it pass the packet?
- The decision is based on optimization: Which of the available pathways is the optimum pathway?

- One approach is to assign a cost for passing through a network. We call this cost a metric.
- However, the metric assigned to each network depends on the type of protocol.
- Some simple protocols, such as the Routing Information Protocol (RIP), treat all networks as equals. The cost of passing through a network is the same; it is one hop count.

- So if a packet passes through 10 networks to reach the destination, the total cost is 10 hop counts
- Another method can be type of service

Figure 22.13 *Popular routing protocols*



Interdomain Routing is the protocol in which the routing algorithm works both within and between domains. Domains must be connected in some way, for hosts inside one domain to exchange data with hosts in other domains.

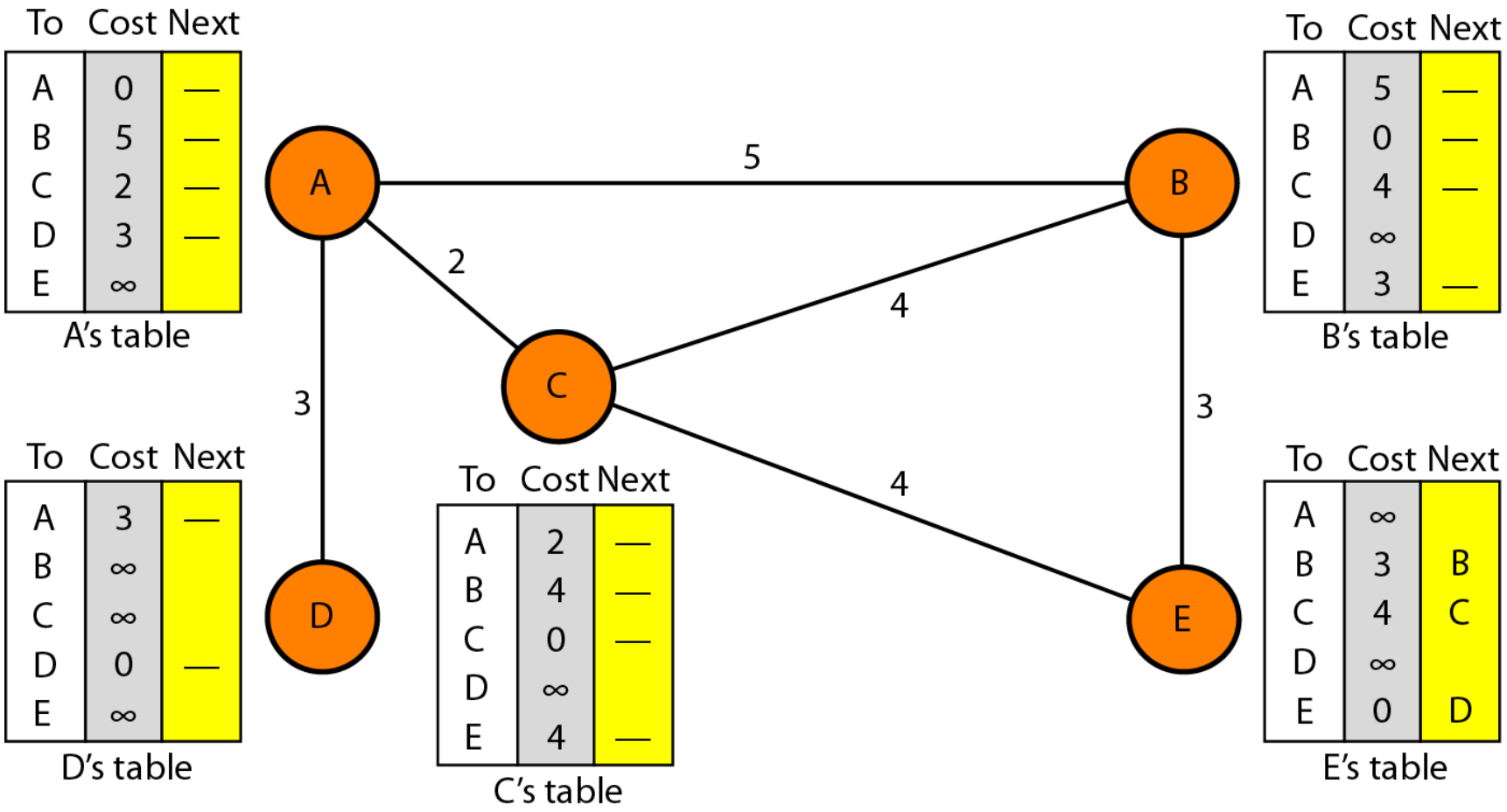
Intradomain Routing is the routing protocol that operates only within a domain. In other words, intradomain routing protocols are used to route packets within a specific domain, such as within an institutional network for e-mail or web browsing. Unlike interdomain routing protocols, it doesn't communicate with other domains.

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

- In this protocol, as the name implies, each node maintains a vector (table) of minimum distances to every node.

- The table at each node also guides the packets to the desired node by showing the next stop in the route (next-hop routing).

Figure 22.15 *Initialization of tables in distance vector routing*



Note

In distance vector routing, each node shares its routing table with its immediate neighbors periodically and when there is a change.

Figure 22.16 *Updating in distance vector routing*

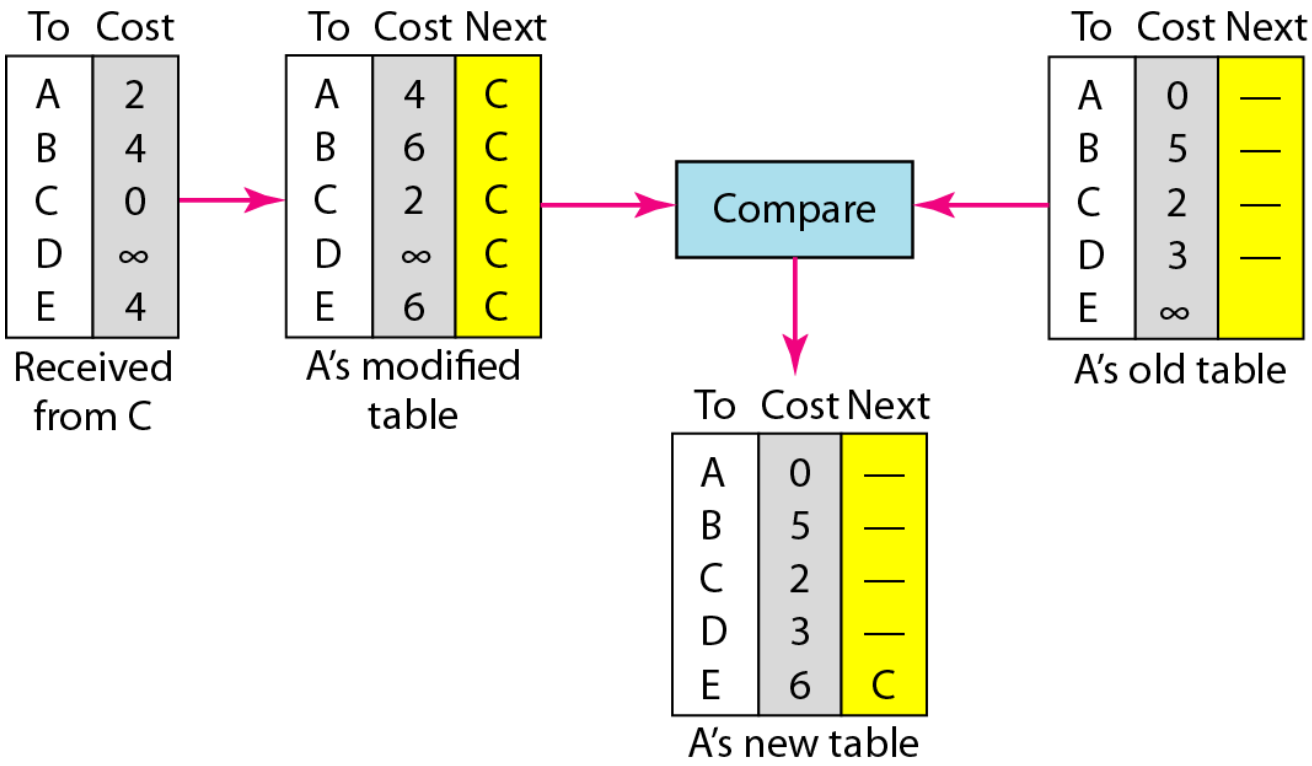
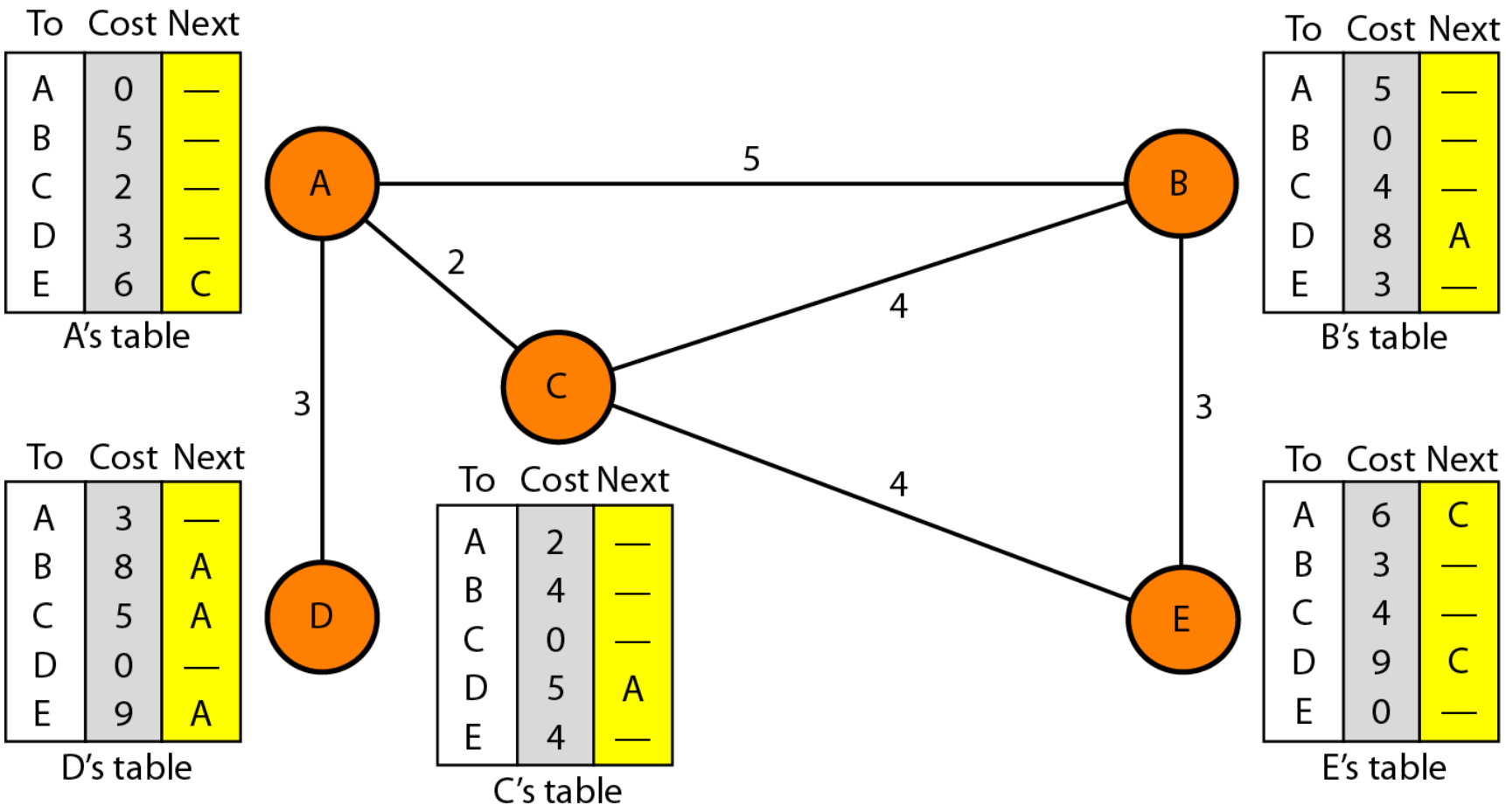
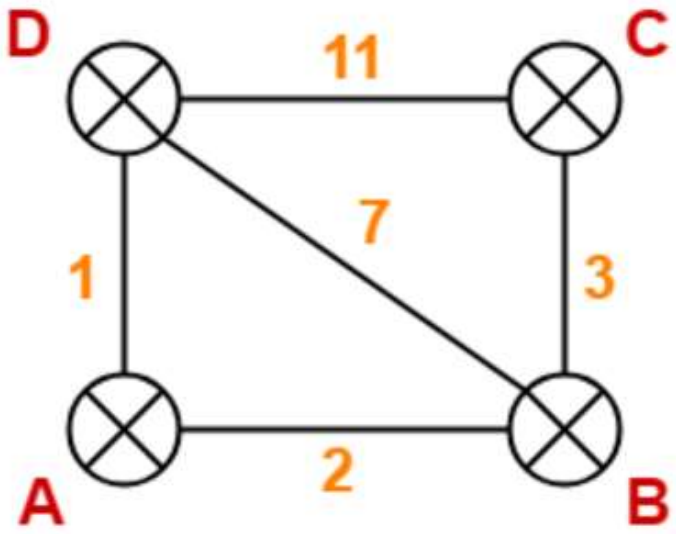


Figure 22.14 *Distance vector routing tables*




D
11
C
7
1
3
2
A
B

Destination	Distance	Next Hop
A	0	A
B	2	B
C	∞	–
D	1	D

Destination	Distance	Next Hop
A	0	A
B	2	B
C	∞	–
D	1	D

Destination	Distance	Next Hop
A	0	A
B	2	B
C	∞	–
D	1	D

Destination	Distance	Next Hop
A	0	A
B	2	B
C	∞	–
D	1	D

Destination	Distance	Next Hop
A	0	A
B	2	B
C	5	B
D	1	D

Destination	Distance	Next Hop
A	0	A
B	2	B
C	5	B
D	1	D

Destination	Distance	Next Hop
A	5	B
B	3	B
C	0	C
D	10	B

Destination	Distance	Next Hop
A	1	A
B	3	A
C	10	B
D	0	D

Destination	Distance	Next Hop
A	0	A
B	2	B
C	5	B
D	1	D

Destination	Distance	Next Hop
A	2	A
B	0	B
C	3	C
D	3	A

Destination	Distance	Next Hop
A	5	B
B	3	B
C	0	C
D	6	B

Destination	Distance	Next Hop
A	1	A
B	3	A
C	6	A
D	0	D

When does a node send its partial routing table (only two columns) to all its immediate neighbors?

Periodic Update: A node sends its routing table, normally every 30 s, in a periodic update. The period depends on the protocol that is using distance vector routing.

Triggered Update: A node sends its two-column routing table to its neighbors anytime there is a change in its routing table. This is called a triggered update.

The change can result from the following:

1. A node receives a table from a neighbor, resulting in changes in its own table after updating.

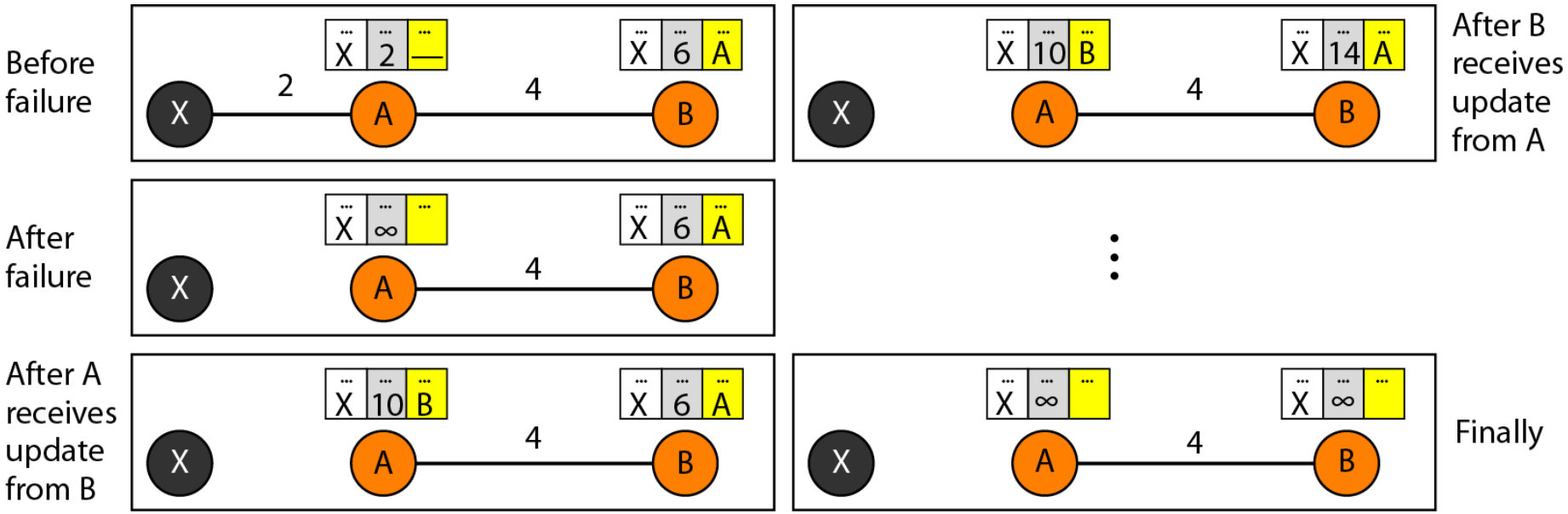
2. A node detects some failure in the neighboring links which results in a distance change to infinity.

Count to Infinity

A problem with distance-vector routing is that any decrease in cost (good news) propa gates quickly, but any increase in cost (bad news) will propagate slowly.

For a routing protocol to work properly, if a link is broken (cost becomes infinity), every other router should be aware of it immediately, but in distance-vector routing, this takes some time. The problem is referred to as **count to infinity**. It sometimes takes several updates before the cost for a broken link is recorded as infinity by all routers.

Figure 22.17 *Two-node instability*



Defining Infinity

- The first obvious solution is to redefine infinity to a smaller number, such as 100.
- For our previous scenario, the system will be stable in less than 20 updates. As a matter of fact, most implementations of the distance vector protocol define the distance between each node to be I and define 16 as infinity.
- However, this means that the distance vector routing cannot be used in large systems.
- The size of the network, in each direction, can not exceed 15 hops.

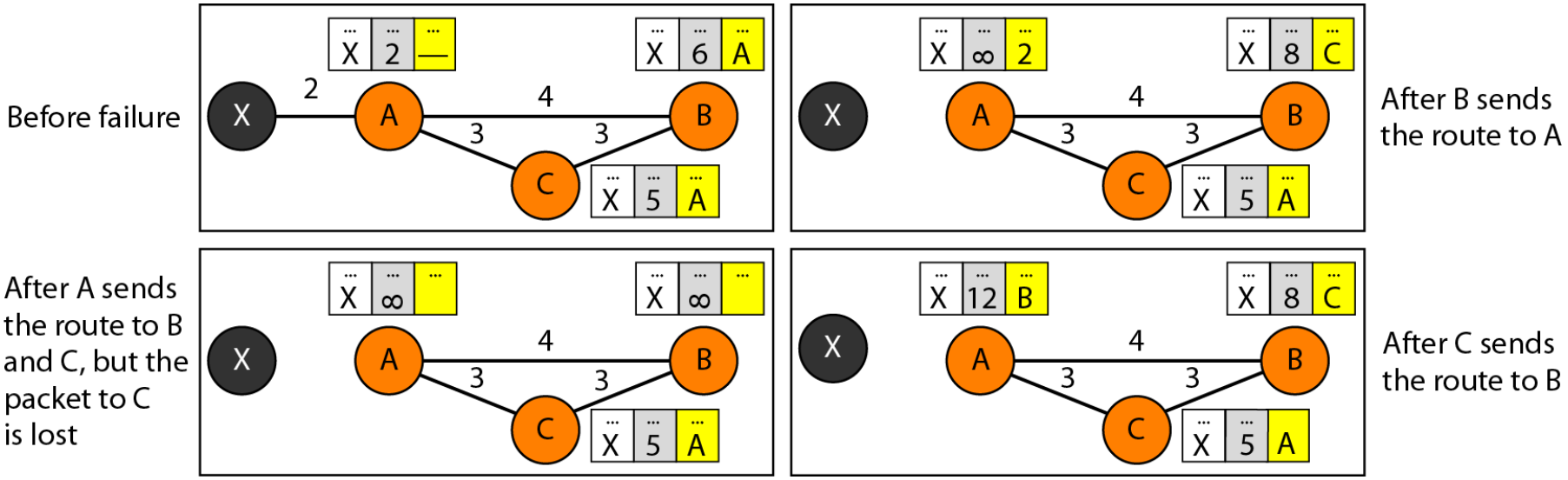
Split Horizon

- If node B thinks that the optimum route to reach X is via A, it does not need to advertise this piece of information to A; the information has come from A (A already knows).
- Taking information from node A, modifying it, and sending it back to node A creates the confusion.

Split Horizon and Poison Reverse

- Normally, the distance vector protocol uses a timer, and if there is no news about a route, the node deletes the route from its table.
- When node B in the previous scenario eliminates the route to X from its advertisement to A, node A cannot guess that this is due to the split horizon strategy (the source of information was A) or because B has not received any news about X recently.
- The split horizon strategy can be combined with the poison reverse strategy.
- Node B can still advertise the value for X, but if the source of information is A, it can replace the distance with infinity as a warning: "Do not use this value; what I know about this route comes from you."

Figure 22.18 *Three-node instability*



Routing Information Protocol (RIP)

It 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.

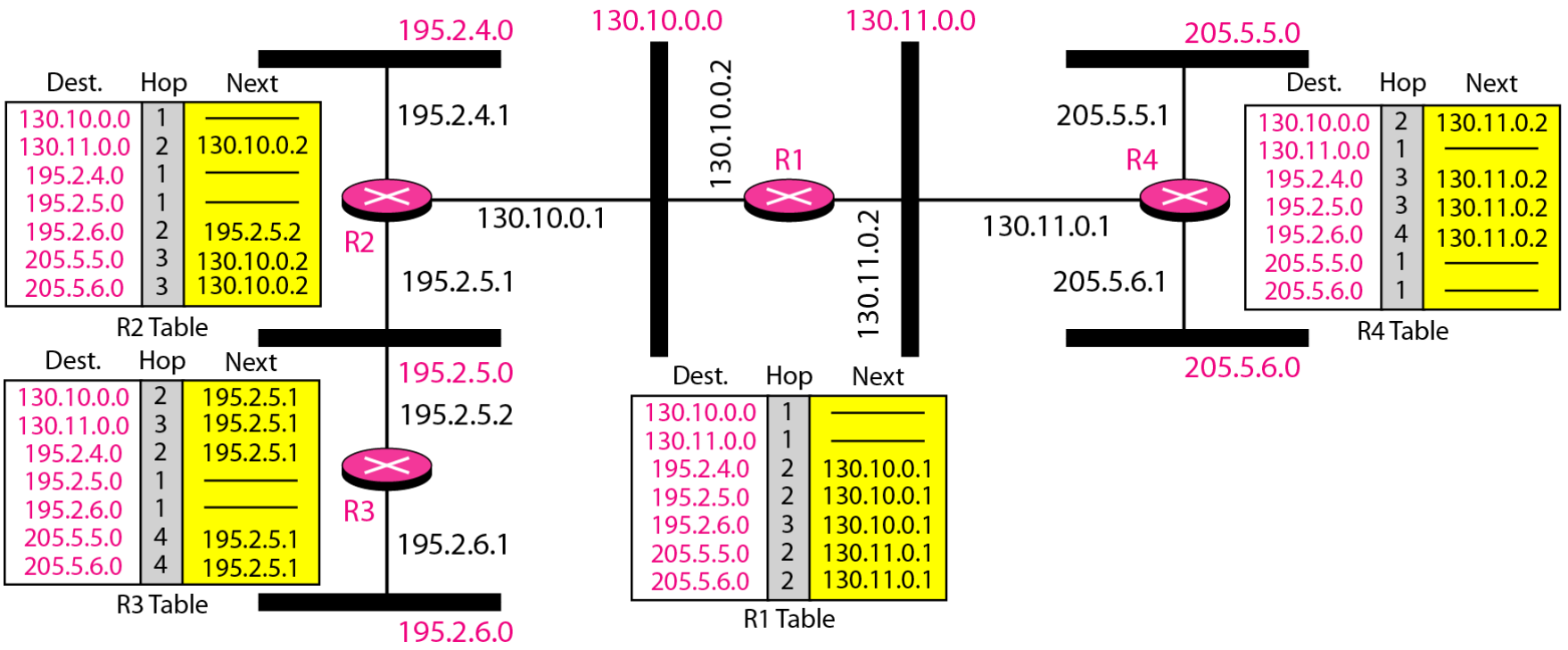
2. The destination in a routing table is a network, which means the first column defines a network address.

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.

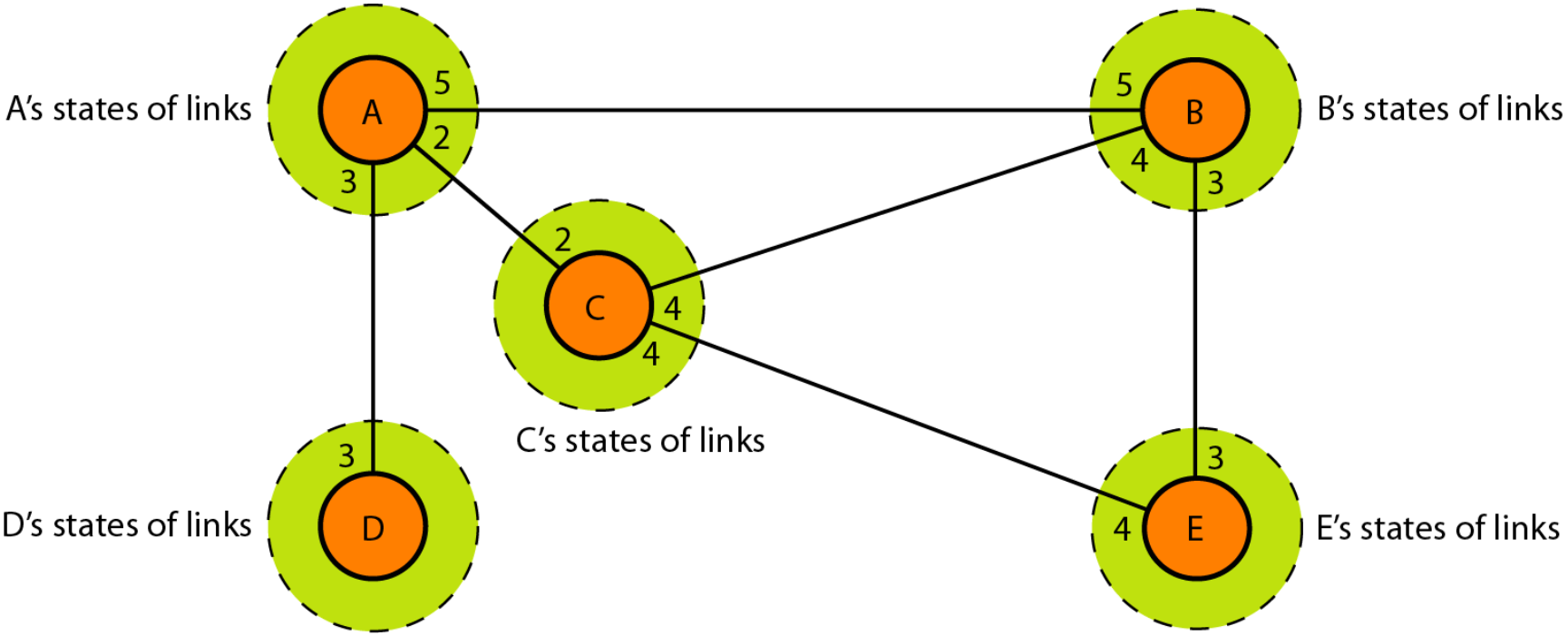
Figure 22.19 *Example of a domain using RIP*



Link state routing

- In link state routing, each node in the domain has the entire topology of the domain
- List of nodes and links, how they are connected including cost (metric), and condition of the links (up or down)

Link state knowledge



Build routing table

In link state routing, four sets of actions are required to ensure that each node has the routing table showing the least-cost node to every other node.

1. Creation of the states of the links by each node, called the link state packet (LSP).
2. Dissemination of LSPs to every other router, called **flooding, in an efficient and** reliable way.
3. Formation of a shortest path tree for each node.
4. Calculation of a routing table based on the shortest path tree.

Creation of Link State Packet (LSP) -

- A link state packet can carry a large amount of information.
- For the moment, however, we assume that it carries a minimum amount of data:
  - node identity
  - list of links
  - sequence number,
  - age.
- The first two, node identity and the list of links, are needed to make the topology. The third, sequence number, facilitates flooding and distinguishes new LSPs from old ones.
- The fourth, age, prevents old LSPs from remaining in the domain for a long time

- LSPs are generated on two occasions:
  - *When there is a change in the topology of the domain*
  - On a periodic basis

Flooding of LSPs-

- The creating node sends a copy of the LSP out of each interface.
- A node that receives an LSP compares it with the copy it may already have.
- If the newly arrived LSP is older than the one it has, it discards the LSP. If it is newer, the node does the following:
- a. It discards the old LSP and keeps the new one.
- b. It sends a copy of it out of each interface except the one from which the packet arrived.

Figure 22.23 *Example of formation of shortest path tree*

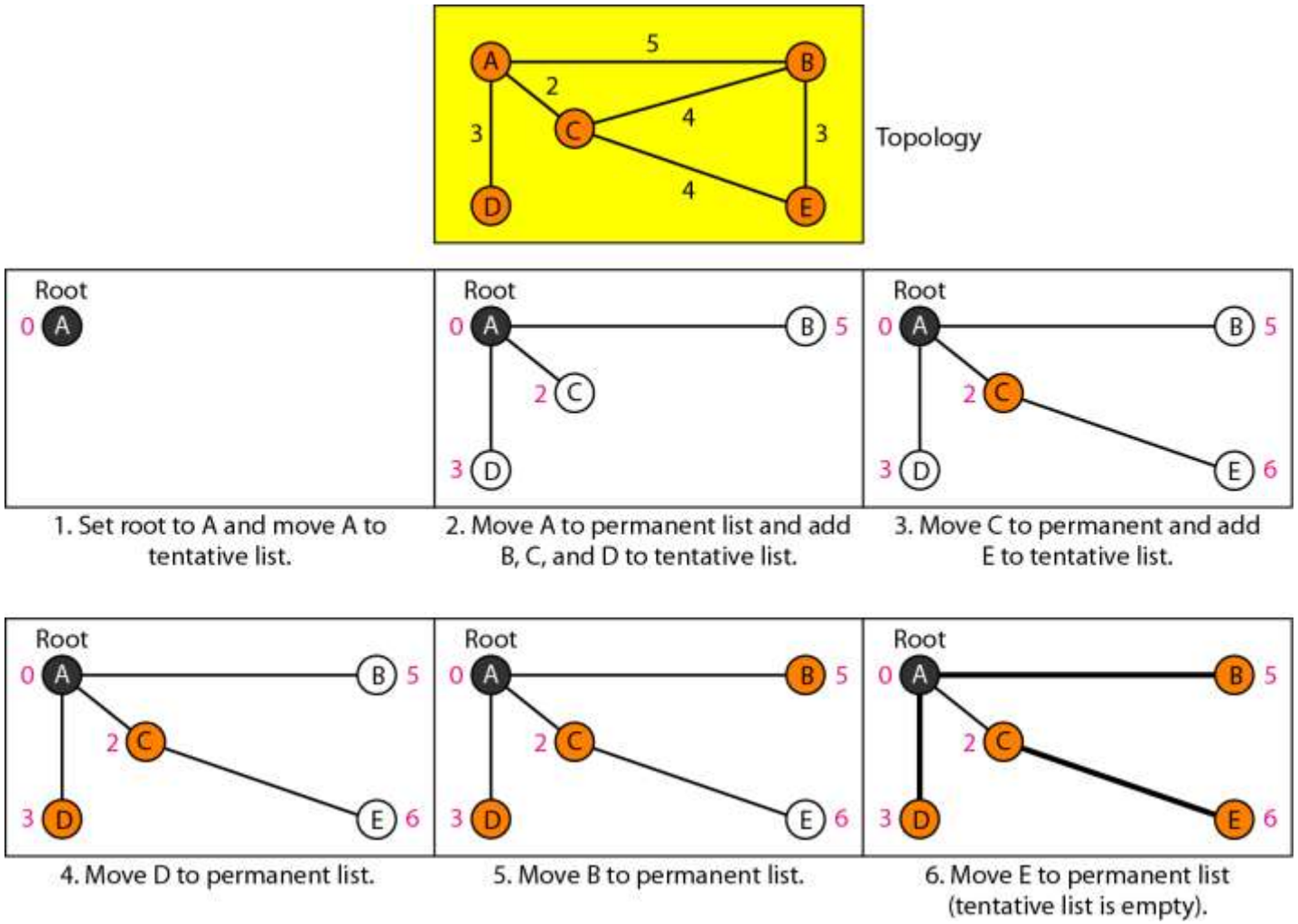


Figure 22.22 *Dijkstra algorithm*

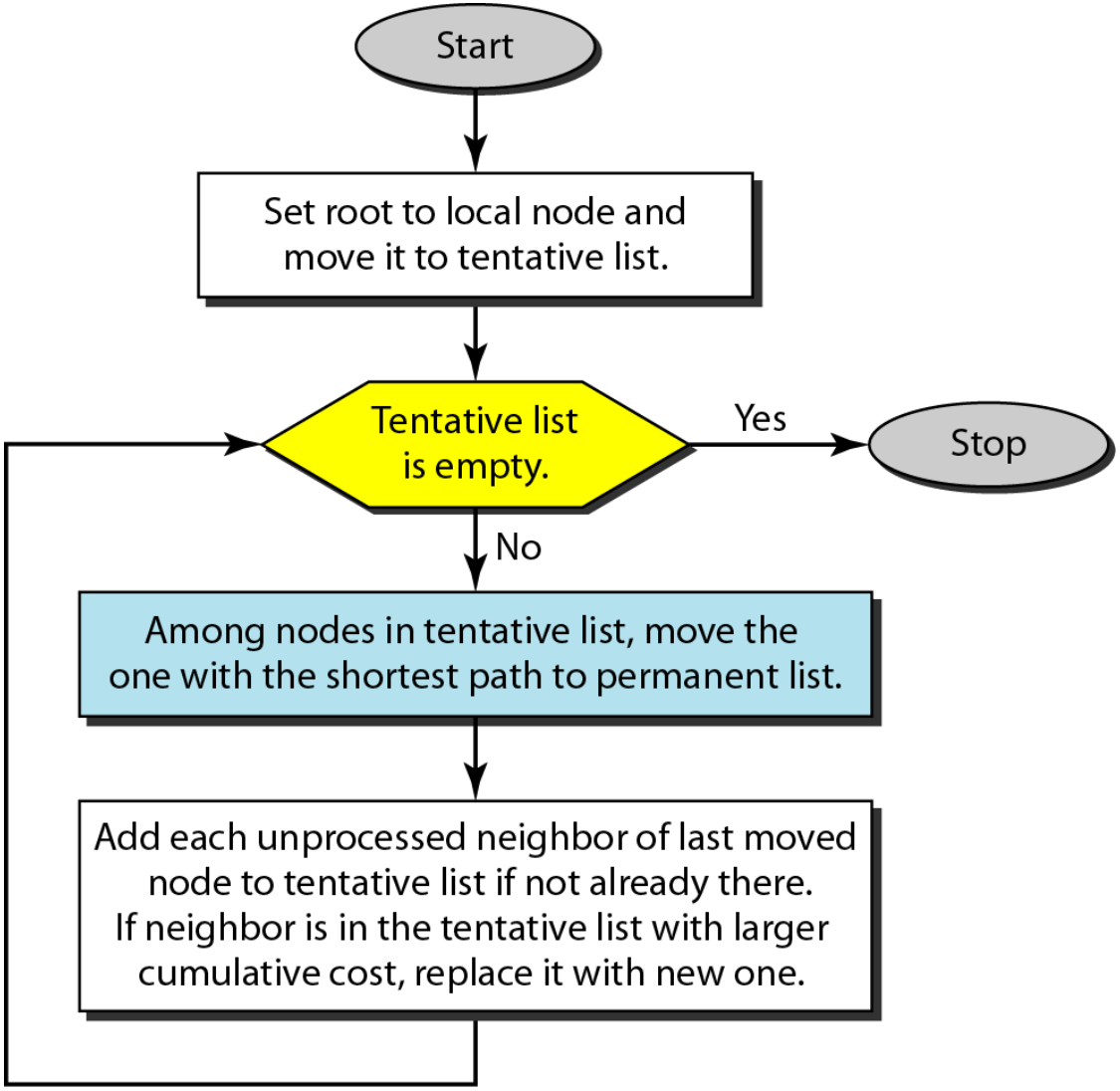


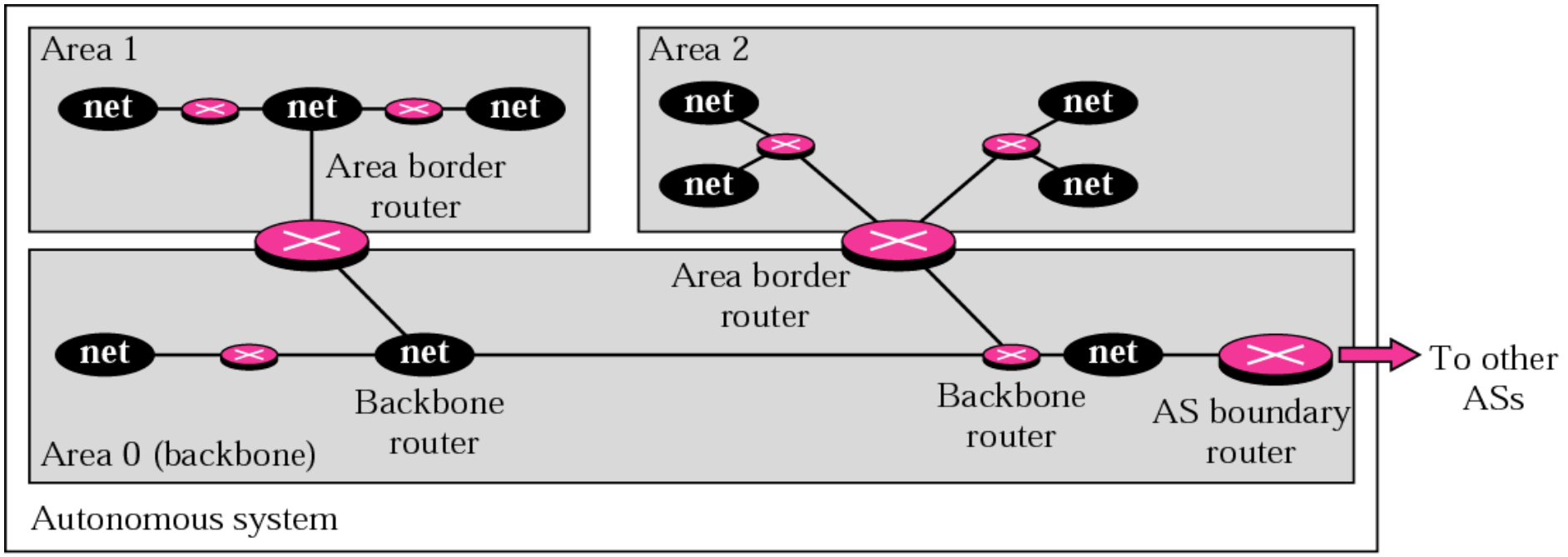
Table 22.2 *Routing table for node A*

Node	*Cost*	*Next Router*
A	0	—
B	5	—
C	2	—
D	3	—
E	6	C

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.

Figure 14.19 *Areas in an autonomous system*



Metric in OSPF

- OSPF allows administrator to assign a cost called metric.
- The metric can be based on type of service- minimum delay, maximum throughput and so on.
- Therefore, a router can have multiple routing tables based on different type of service.

Figure 14.20 *Types of links*

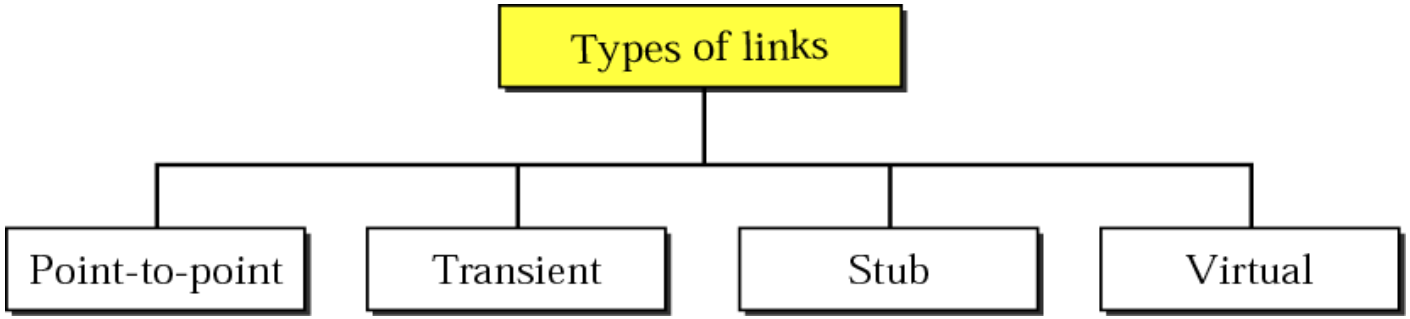


Figure 14.21 *Point-to-point link*

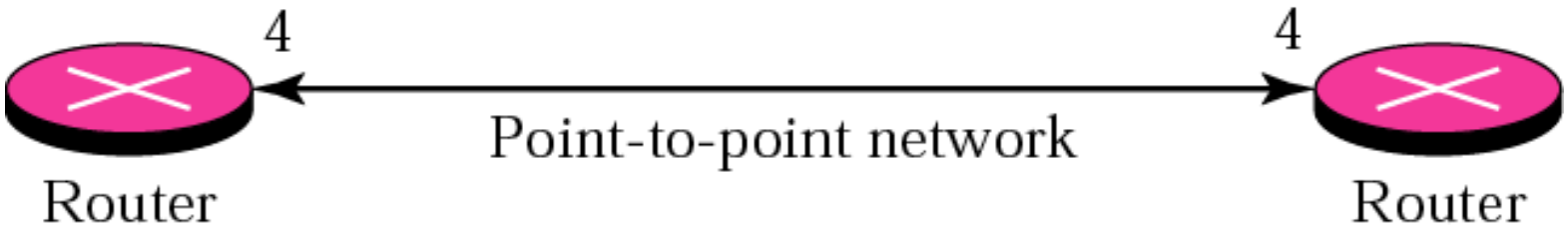
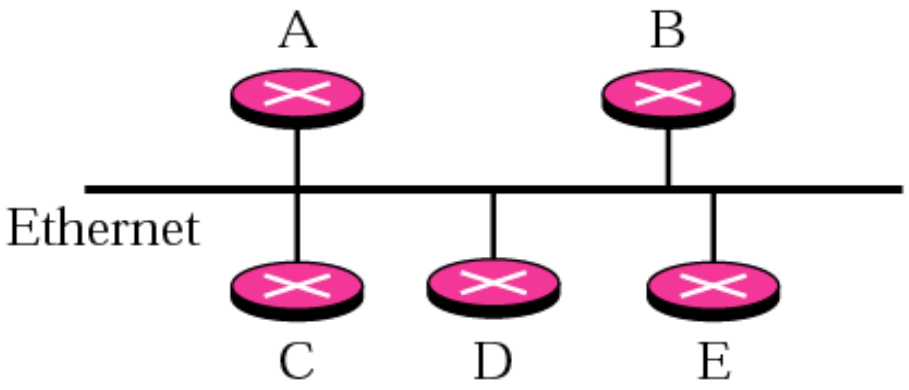
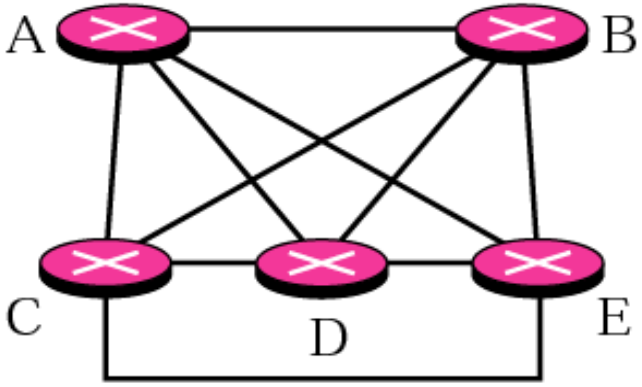


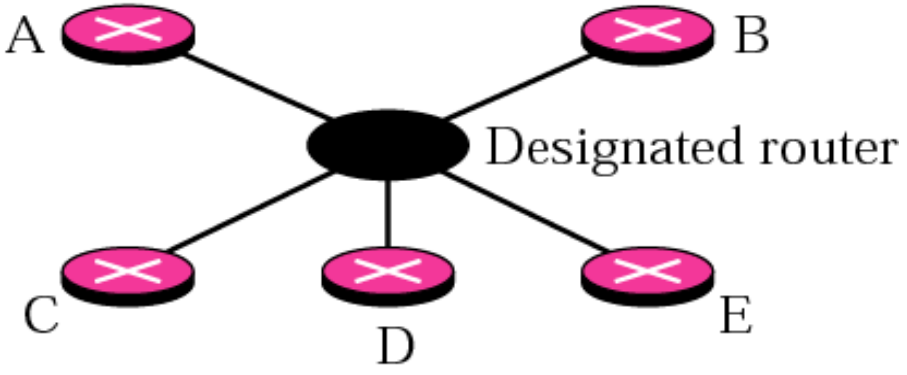
Figure 14.22 *Transient link*



a. Transient network

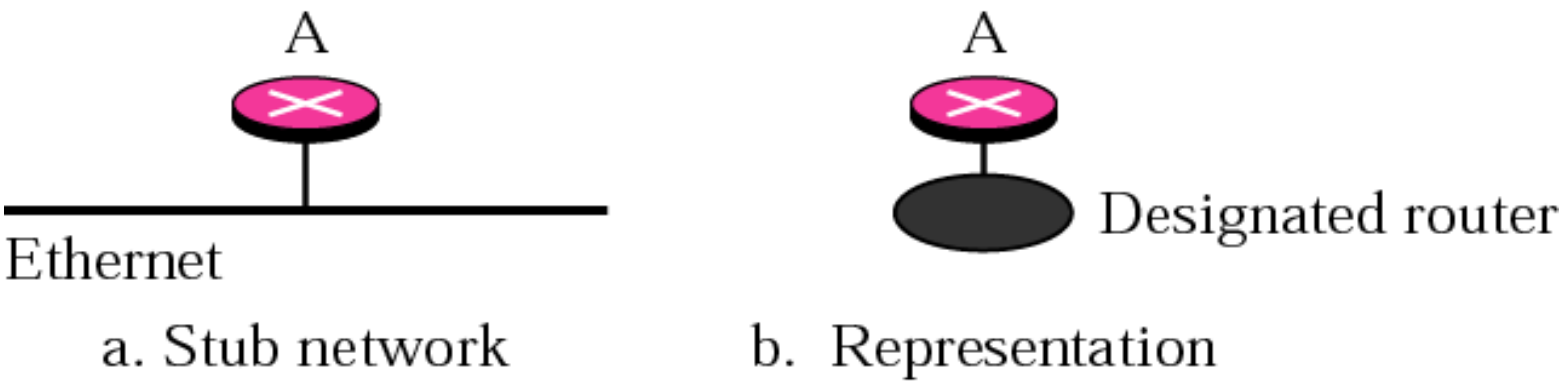


b. Unrealistic representation



c. Realistic representation

Figure 14.23 *Stub link*



a. Stub network b. Representation

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 speaker nodes in the neighboring ASs..

The topics discussed in this section include:

Initialization
Sharing
Updating

- Problem in distance vector is node instability.

- Problem in link state routing is heavy traffic due to flooding and extra resources required for generating routing tables.

Steps in path vector Routing

- Initialization
- Sharing
- Updating– Loop prevention, policy routing, optimum path.

Figure 14.48 *Initial routing tables in path vector routing*

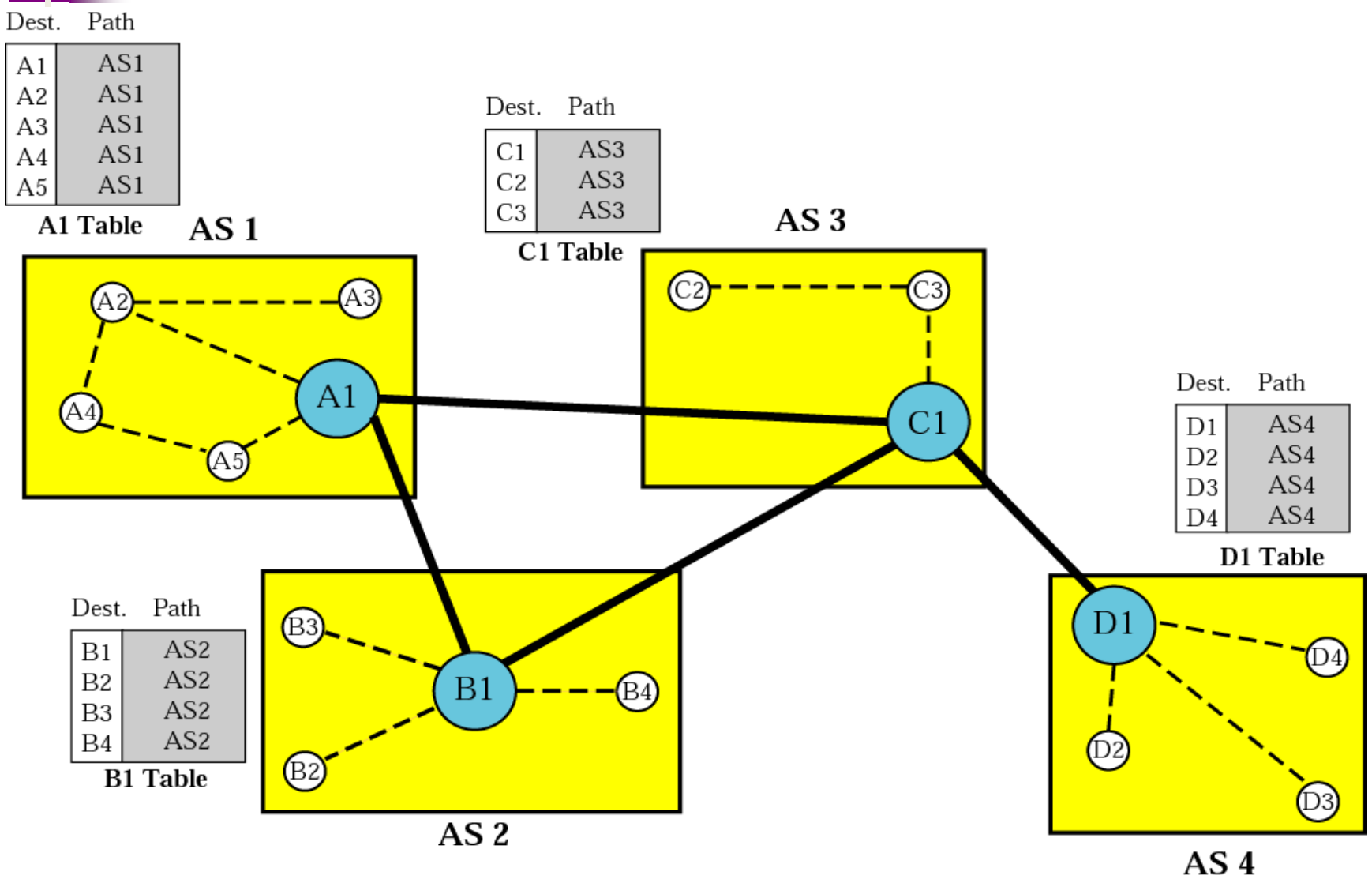


Figure 14.49 *Stabilized tables for four autonomous systems*

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

A1 Table

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

B1 Table

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

C1 Table

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

The topics discussed in this section include:

Types of Autonomous Systems
Path Attributes
BGP Sessions
External and Internal BGP

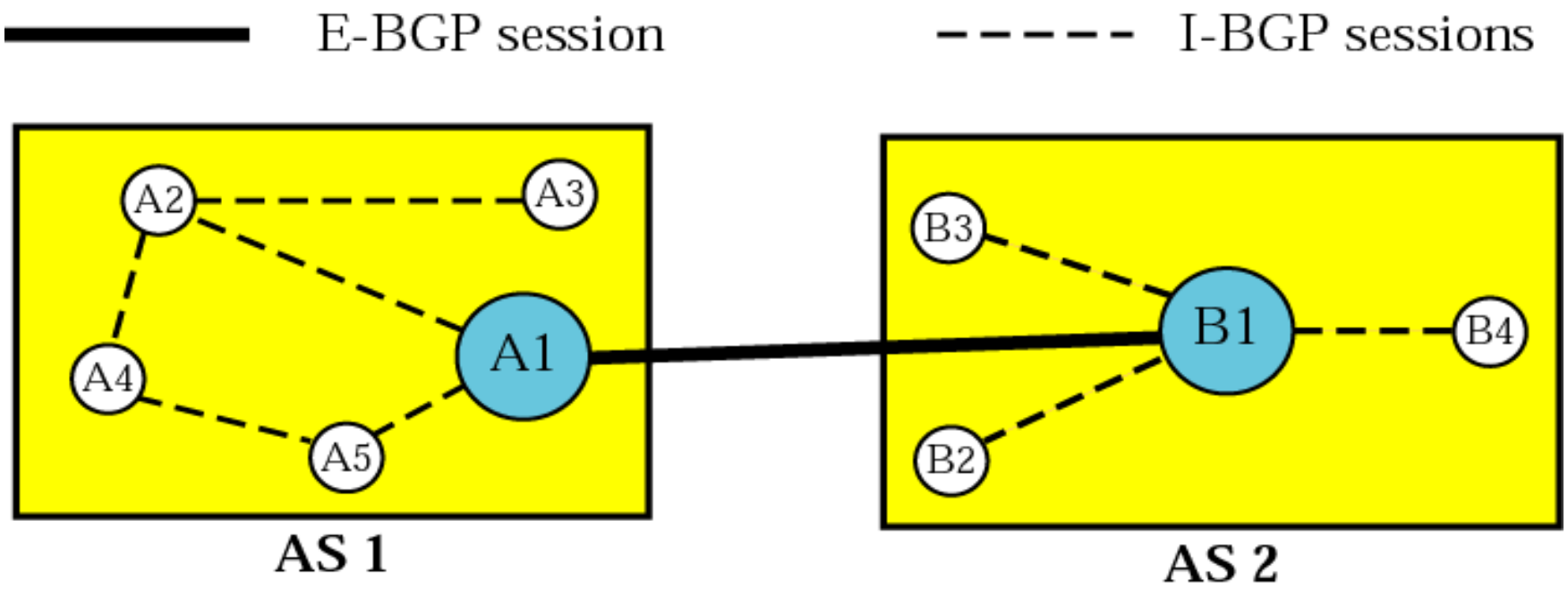
Path Attributes

- Well Known attribute
  - Mandatory
  - Discretionary
- Optional attribute
  - Transitive
  - Non-transitive

BGP Sessions

- Using TCP connections—reliable
- Last for long time so it is called as semi permanent connections.

Figure 14.50 *Internal and external BGP sessions*



24-1 DATA TRAFFIC

The main focus of congestion control and quality of service is data traffic. In congestion control we try to avoid traffic congestion. In quality of service, we try to create an appropriate environment for the traffic. So, before talking about congestion control and quality of service, we discuss the data traffic itself.

Figure 24.1 *Traffic descriptors*

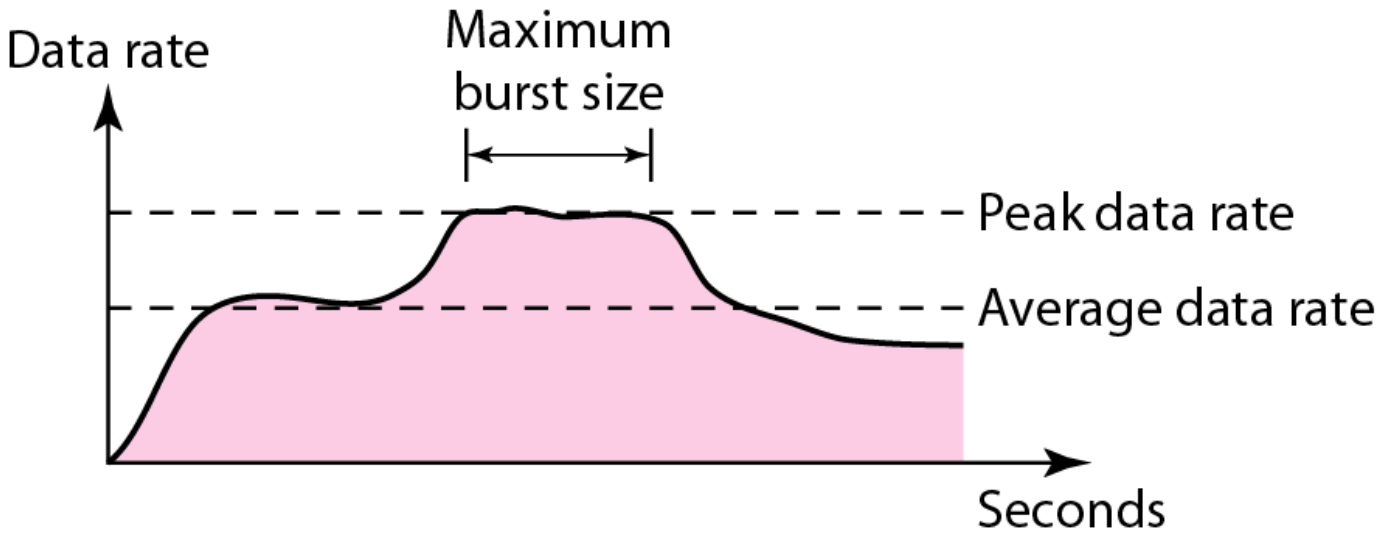
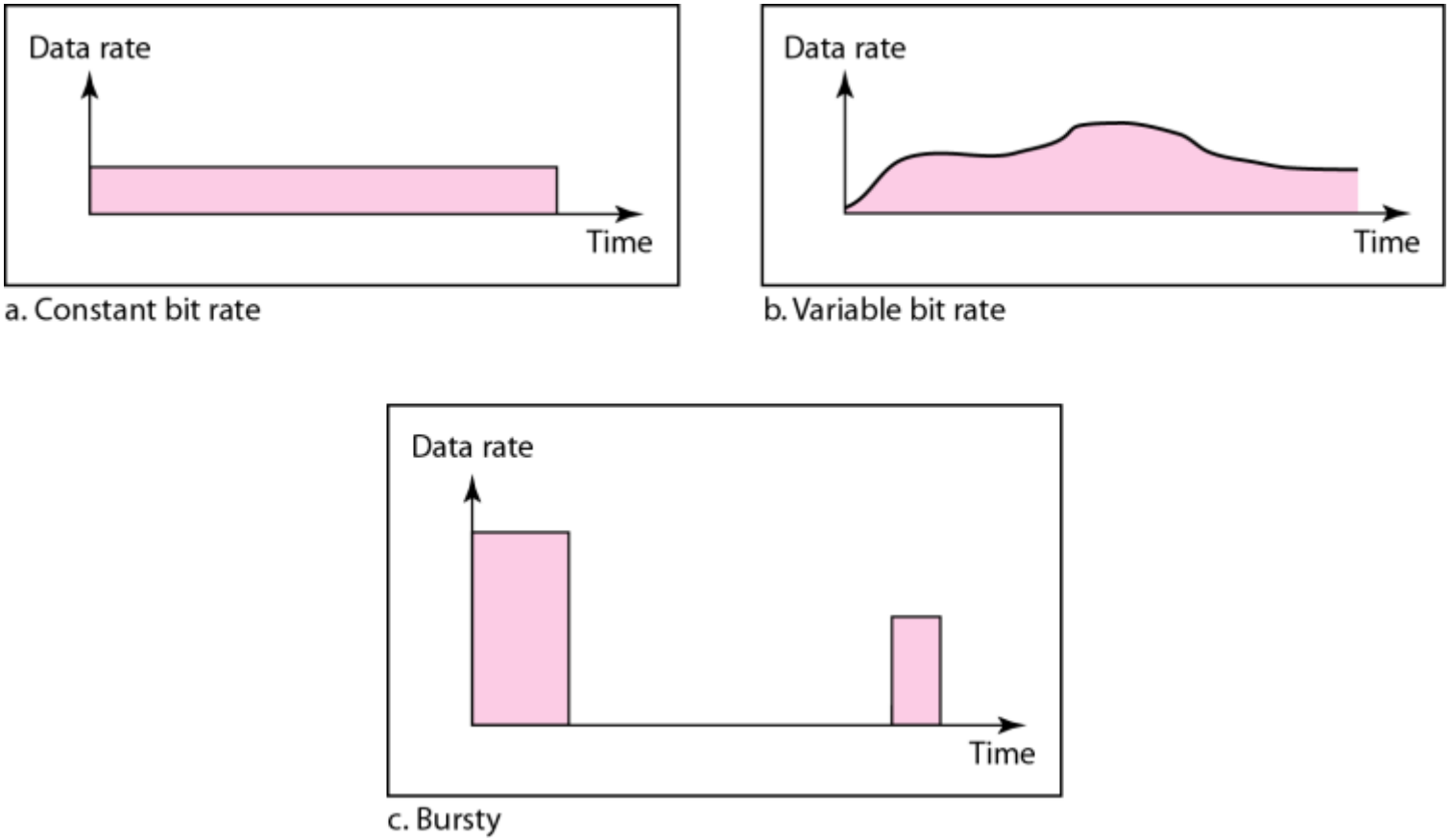


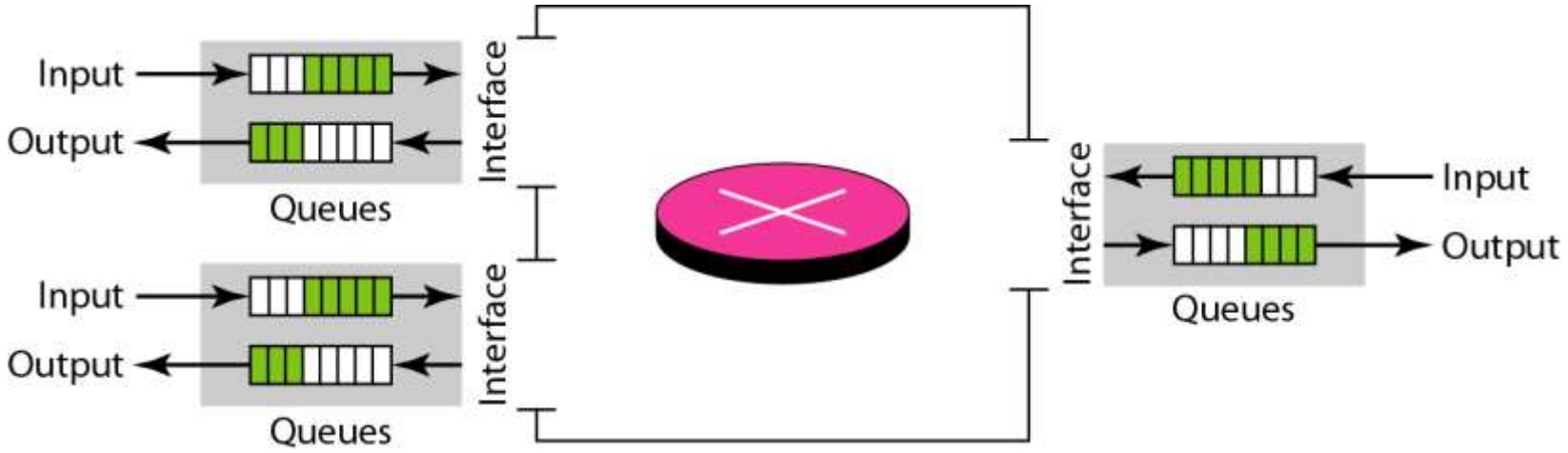
Figure 24.2 *Three traffic profiles*



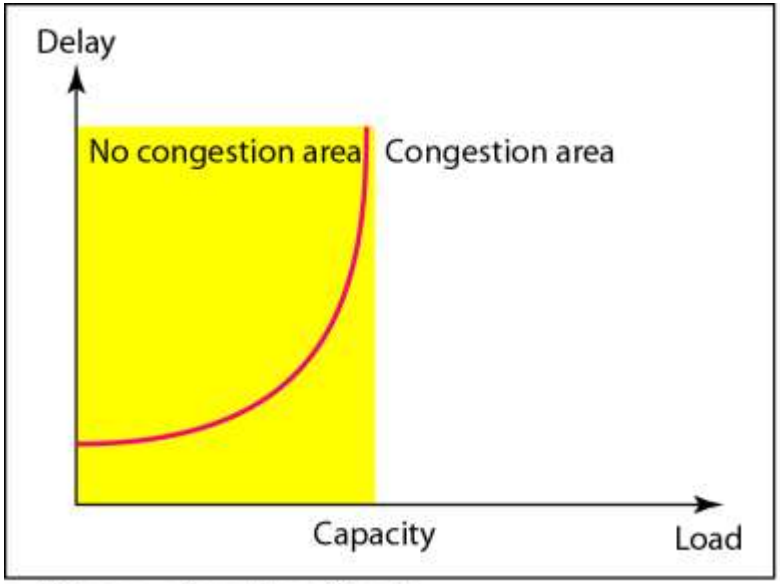
24-2 CONGESTION

Congestion in a network may occur if the load on the network—the number of packets sent to the network—is greater than the capacity of the network—the number of packets a network can handle. Congestion control refers to the mechanisms and techniques to control the congestion and keep the load below the capacity.

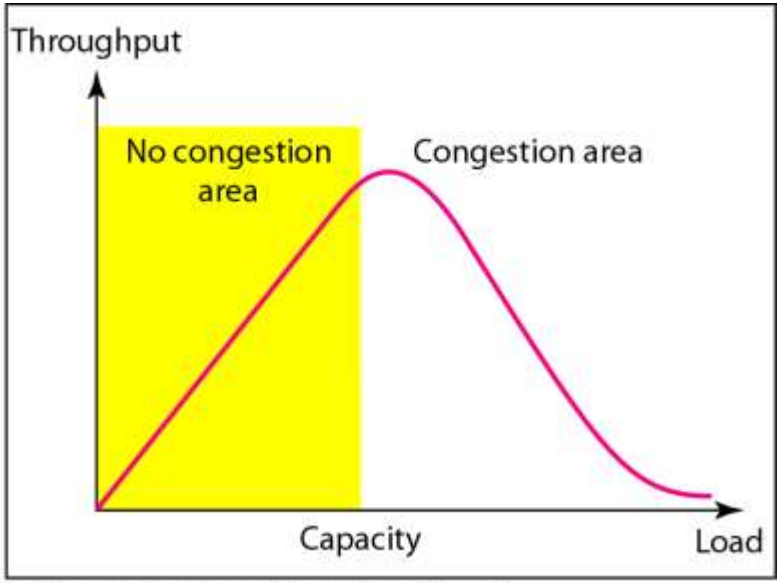
Figure 24.3 *Queues in a router*



Network Performance: *Delay and throughput as functions of load*



a. Delay as a function of load



b. Throughput as a function of load

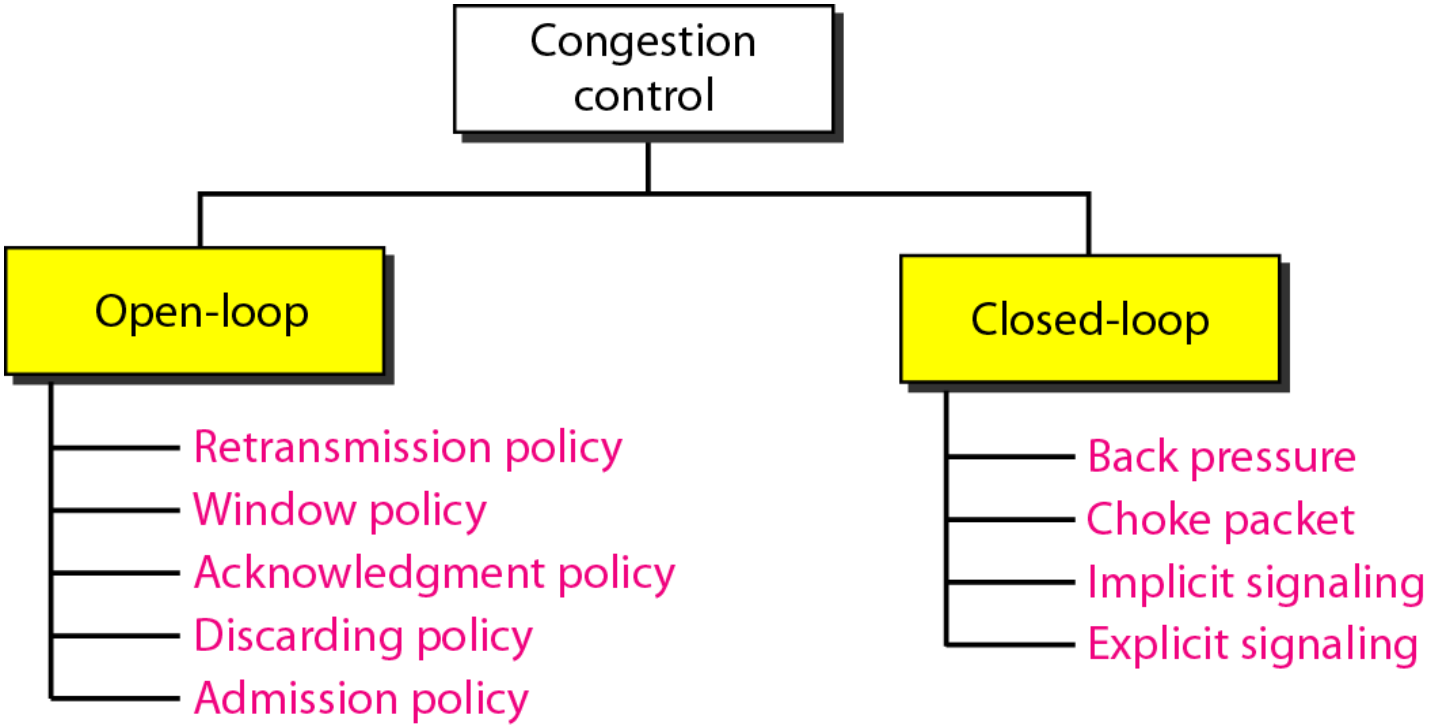
24-3 CONGESTION CONTROL

*Congestion control refers to techniques and mechanisms that can either **prevent congestion**, before it happens, or **remove congestion**, after it has happened. In general, we can divide congestion control mechanisms into two broad categories: **open-loop congestion control (prevention)** and **closed-loop congestion control (removal).***

Topics discussed in this section:

Open-Loop Congestion Control
Closed-Loop Congestion Control

Figure 24.5 *Congestion control categories*



Retransmission Policy

- When to retransmit
  - Lost Frame
  - Lost Ack
  - Damaged Frame

After Timer expiration

timer value too small– more retransmissions

timer value too large– delay in response to lost segment.

Window policy

- What will be maximum size of window- in case of sliding window protocol

  Either you implement

  - Go-Back N ARQ
  - Selective Repeat ARQ

Acknowledgement Policy

- Selective Ack
- Cumlative Ack

Discarding policy

- In case of congestion
  - Which packet to discard
    - Low priority
    - Newer Packet
    - Discard the packet which is NOT nearest to destination
    - When to discard
      - Make sure , you are NOT discarding those packets which are just about to enter the destination's network.

Admission Policy

- If there are insufficient resources to handle the transmission, then do not accept any more packets from sender.

- A router can deny establishing a virtual circuit connection if there is a congestion.

Solutions to Open Loop Congestion

Selective repeat window is better

Should send cumulative acknowledgement

Higher priority packets should not be discard

Closed Loop Congestion Control

- Backpressure
- Choke packet
- Implicit Signaling
- Explicit Signaling

Figure 24.6 *Backpressure method for alleviating congestion*

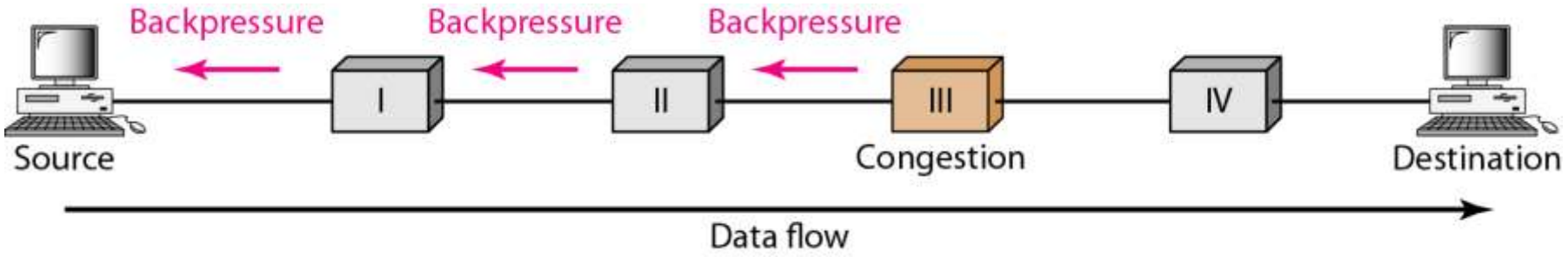
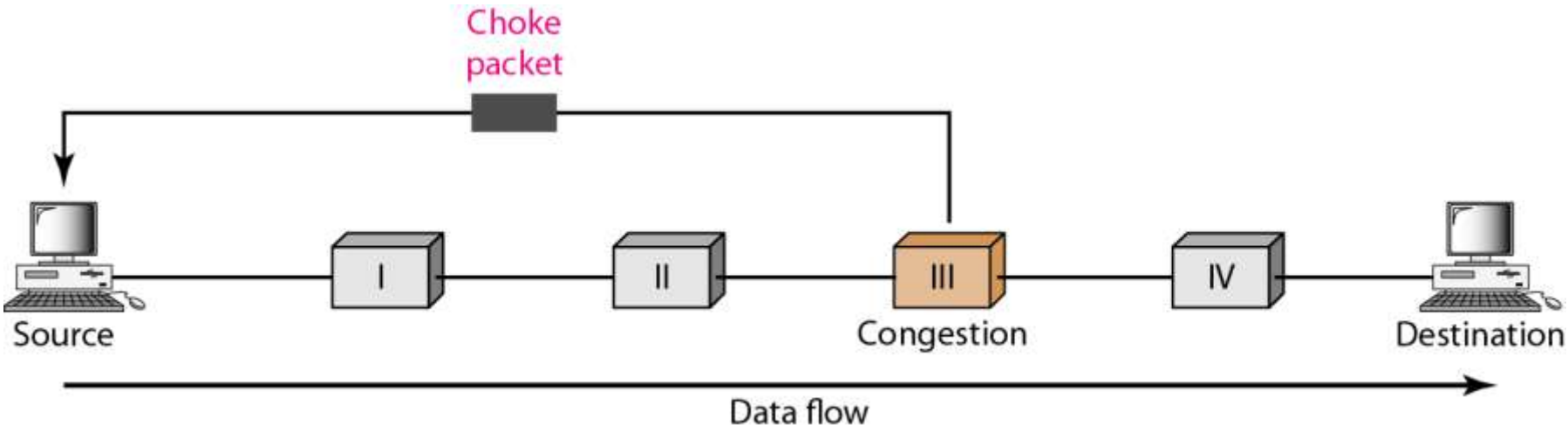


Figure 24.7 *Choke packet*



In backpressure, the warning is from one node to its upstream node, although the warn ing may eventually reach the source station.

In the choke-packet method, the warning is from the router, which has encountered congestion, directly to the source station. The intermediate nodes through which the packet has traveled are not warned.

Congestion leads to –

discard of segments

transmission delay increase

Implicit Signaling- by symptom ***source*** will aware about congestion in network. No communication is send to source from congested node. For example, through delayed acknowledgments.

Explicit signal- network alerts sender or destination to slow down the rate of transmission by sending signal that is included in packet that carry data. That is why its different from choke packet.

- Backward signaling- warning source (opposite direction to congestion) and take appropriate congestion corrective measures

- Forward signaling- warning destination (same direction to congestion) and take appropriate congestion avoidance measures. Such as receiver can slow down the acknowledgment

Quality of service (QoS)

- To provide good quality of service is to build a network with enough capacity for whatever traffic will be thrown at it. The name for this solution is **over-provisioning**.

- Quality of service mechanisms let a network with less capacity meet application requirements just as well at a lower cost.

Flow Characteristics

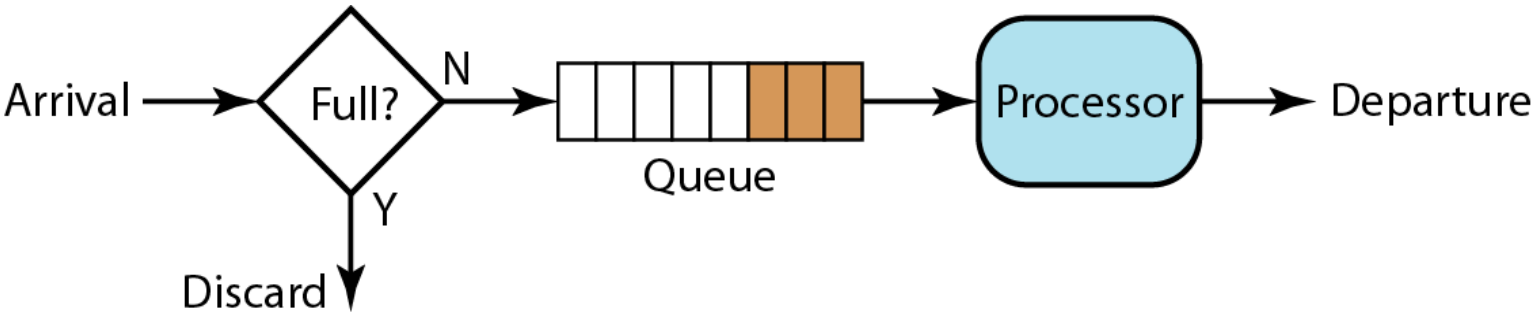
- Reliability
- Delay
- Jitter– variation in delay
- Bandwidth

TECHNIQUES TO IMPROVE QoS

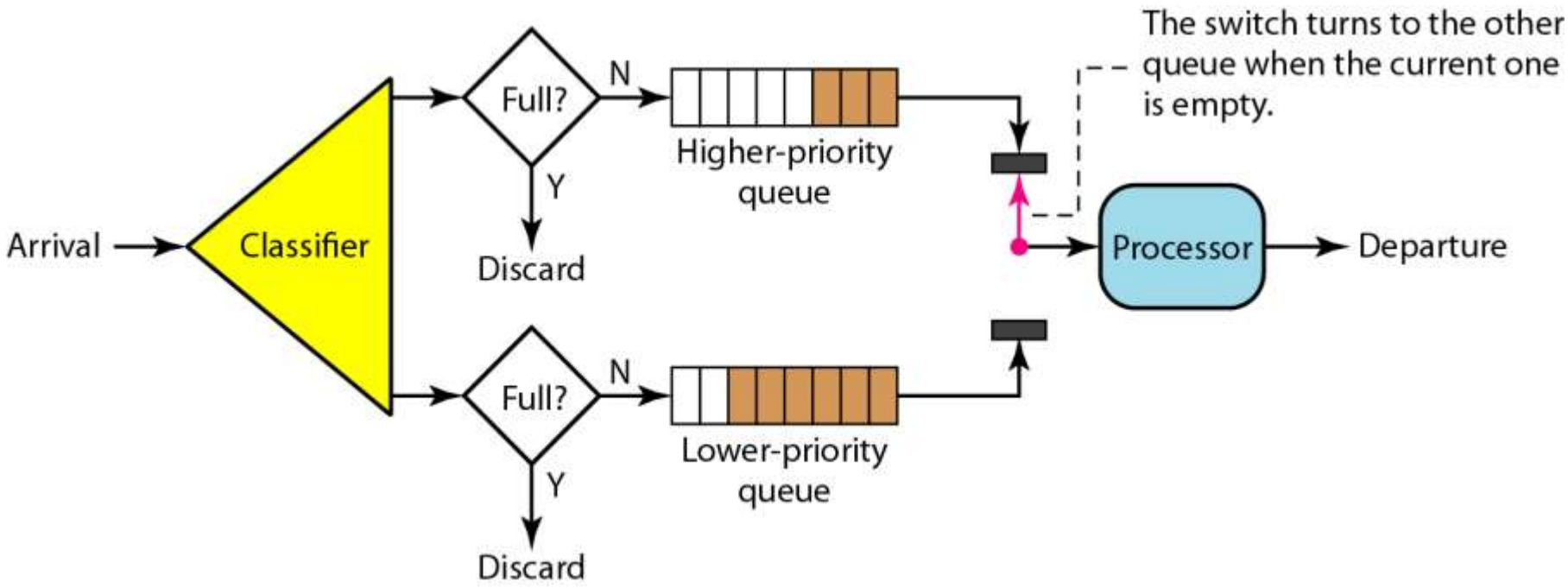
- Scheduling
  - FIFO Queuing
  - Priority Queuing
  - Weighted Fair Queuing
- Traffic Shaping
  - Leaky Bucket
  - Token Bucket
- Resource Reservation
- Admission Control

Scheduling : *FIFO queue*

In first-in, first-out (FIFO) queuing, packets wait in a buffer (queue) until the node (router or switch) is ready to process them. If the average arrival rate is higher than the average processing rate, the queue will fill up and new packets will be discarded. A FIFO queue is familiar to those who have had to wait for a bus at a bus stop.



Scheduling : *Priority queuing*

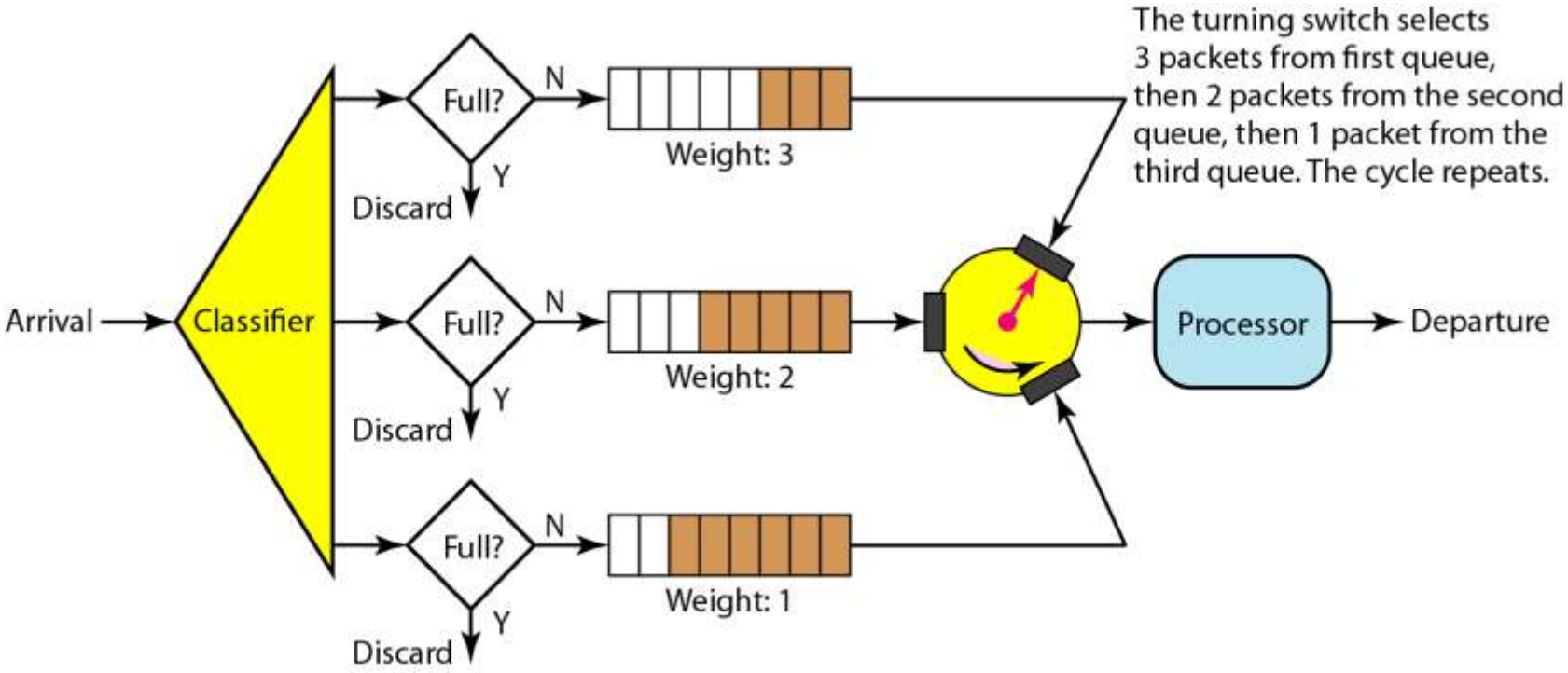


- A priority queue can provide better QoS than the FIFO queue because higher-priority traffic, such as multimedia, can reach the destination with less delay. However, there is a potential drawback.
- If there is a continuous flow in a high-priority queue, the packets in the lower-priority queues will never have a chance to be processed. This is a condition called starvation. Severe starvation may result in dropping of some packets of lower priority.

Starvation

- A priority queue can provide better QoS than the FIFO queue because higher priority traffic, such as multimedia, can reach the destination with less delay. However, there is a potential drawback. If there is a continuous flow in a high-priority queue, the packets in the lower-priority queues will never have a chance to be processed. This is a condition called starvation

Scheduling : *Weighted fair queuing*



- A better scheduling method is weighted fair queuing. In this technique, the packets are still assigned to different classes and admitted to different queues. The queues, however, are weighted based on the priority of the queues; higher priority means a higher weight. The system processes packets in each queue in a round-robin fashion with the number of packets selected from each queue based on the corresponding weight. For example, if the weights are 3, 2, and 1, three packets are processed from the first queue, two from the second queue, and one from the third queue. If the system does not impose priority on the classes, all weights can be equal. In this way, we have fair queuing with priority.

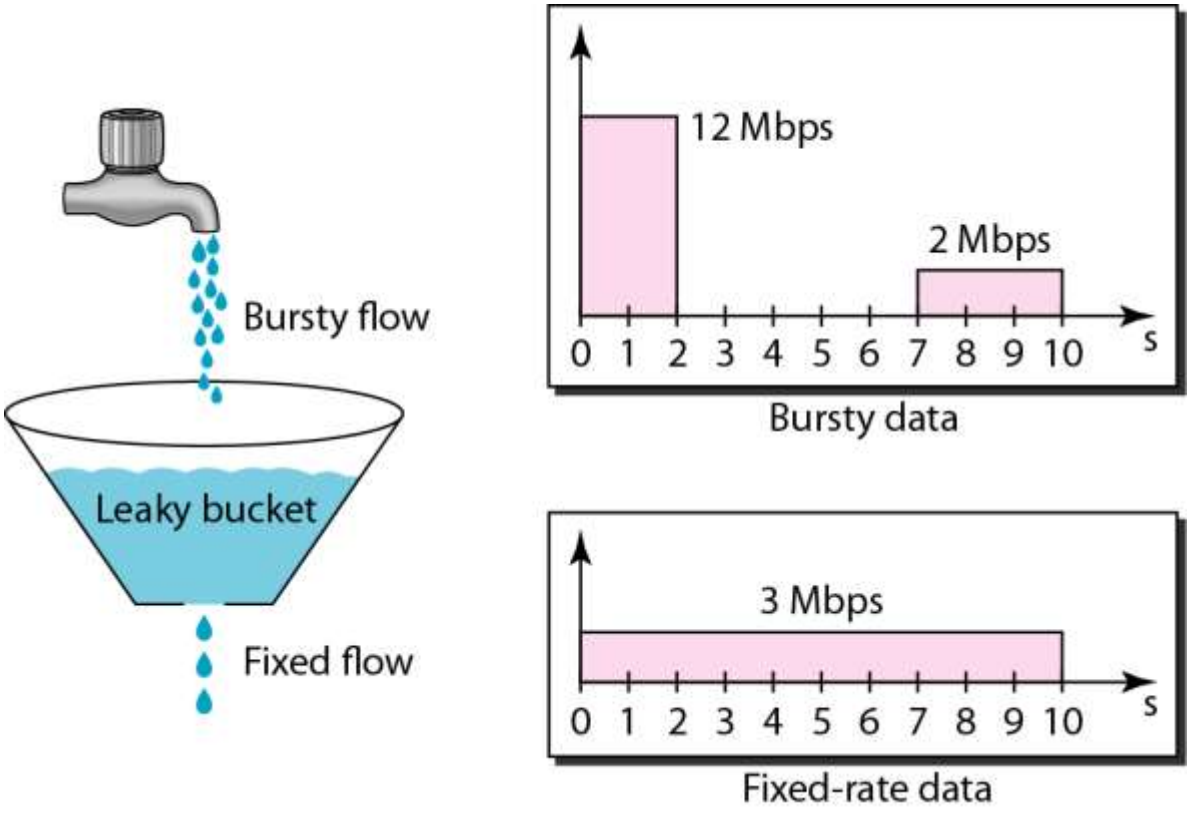
Traffic Shaping

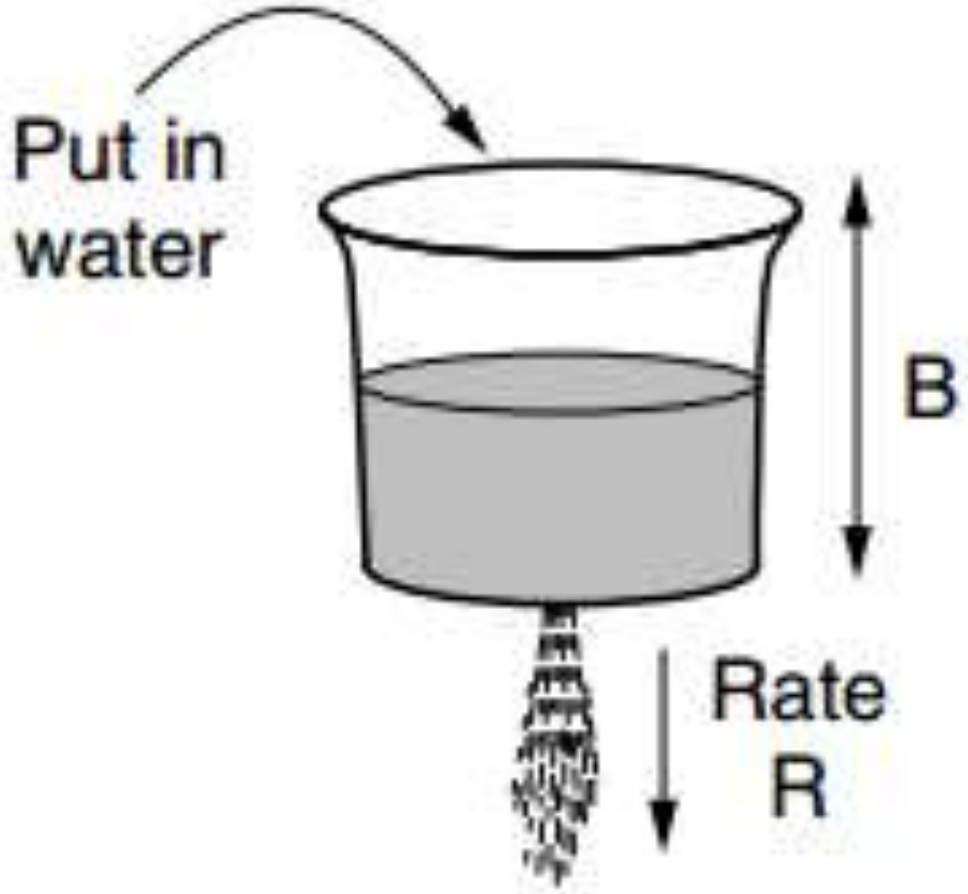
- **Traffic shaping** is a technique for regulating/ controling the average rate and burst-ness of a flow of data that enters the network.
- The goal is to allow applications to transmit a wide variety of traffic that suits their needs, including some bursts.
- **SLA** (**Service Level Agreement**)

Traffic Policing

- Traffic shaping reduces congestion and thus helps the network live up to its promise.
- There is also the issue of how the provider can tell if the customer is following the agreement and what to do if the customer is not.
- Packets in excess of the agreed pattern might be dropped by the network, or they might be marked as having lower priority.
- Monitoring a traffic flow is called **traffic policing**.

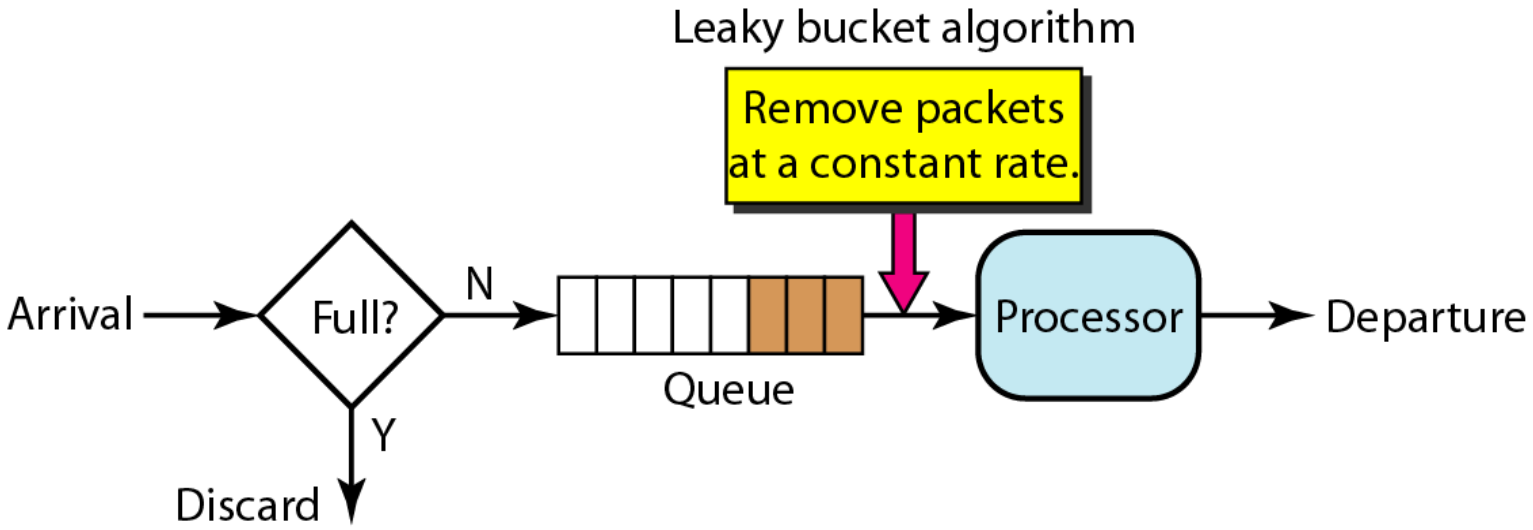
Traffic Shaping : *Leaky bucket*




Put in
water
B
Rate
R

- we assume that the network has committed a bandwidth of 3 Mbps for a host. The use of the leaky bucket shapes the input traffic to make it conform to this commitment. In Figure 4.34 the host sends a burst of data at a rate of 12 Mbps for 2 s, for a total of 24 Mbits of data. The host is silent for 5 s and then sends data at a rate of 2 Mbps for 3 s, for a total of 6 Mbits of data. In all, the host has sent 30 Mbits of data in lO s. The leaky bucket smooth’s the traffic by sending out data at a rate of 3 Mbps during the same 10 s.

Traffic Shaping : *Leaky bucket implementation*



Note

A leaky bucket algorithm shapes bursty traffic into fixed-rate traffic by averaging the data rate. It may drop the packets if the bucket is full.

- Suppose we have a bucket in which we are pouring water, at random points in time, but we have to get water at a fixed rate, to achieve this we will make a hole at the bottom of the bucket. This will ensure that the water coming out is at some fixed rate, and also if the bucket gets full, then we will stop pouring water into it. The input rate can vary, but the output rate remains constant. Similarly, in networking, a technique called leaky bucket can smooth out bursty traffic. Bursty chunks are stored in the bucket and sent out at an average rate.

Note

The token bucket allows bursty traffic at a regulated maximum rate.

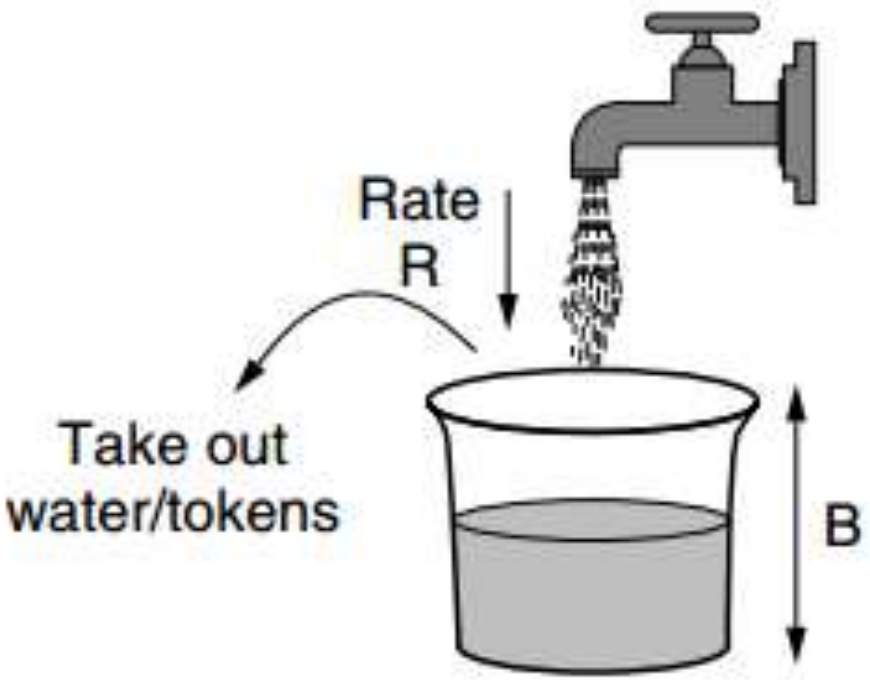
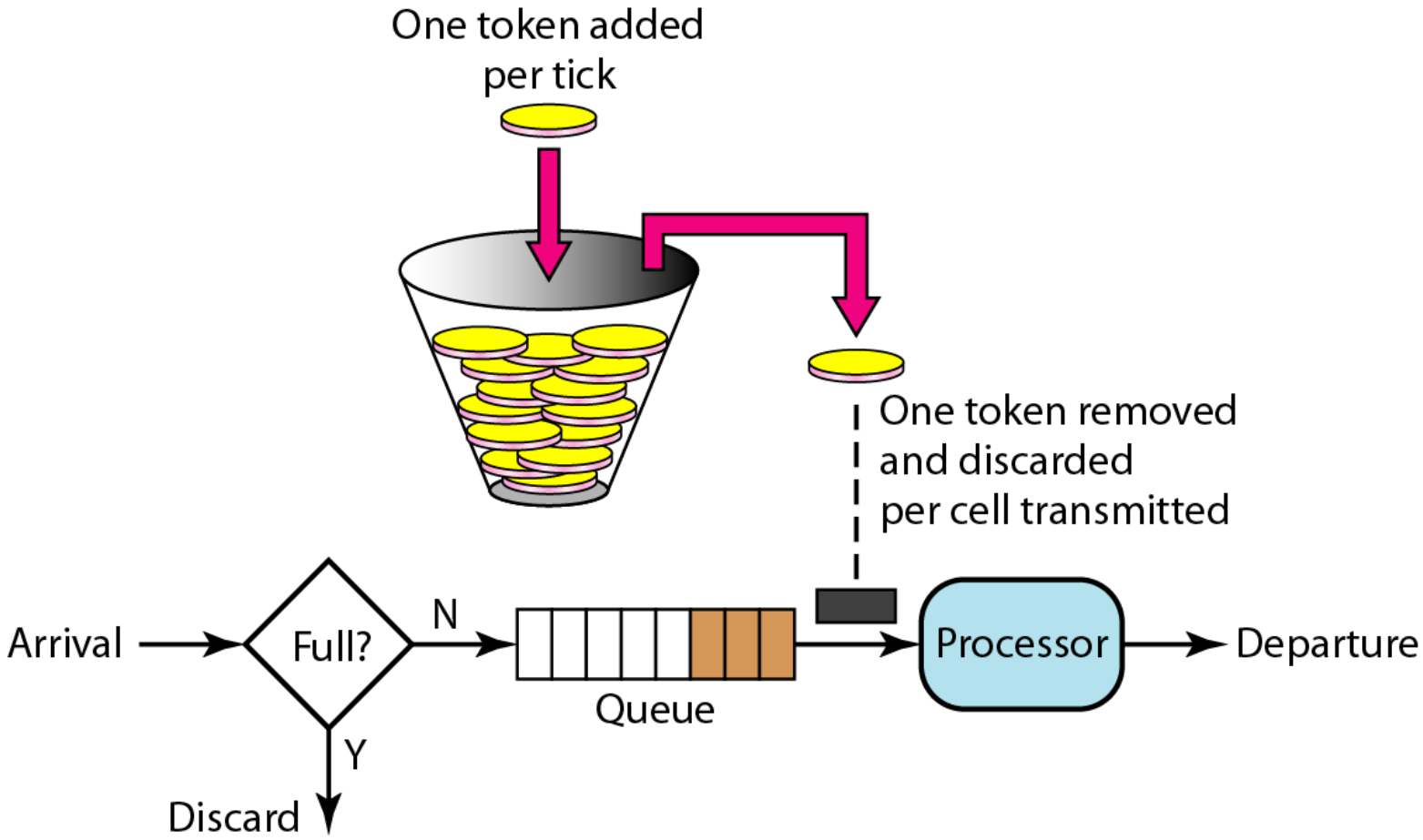

Rate
R
Take out
water/tokens
B

Figure 24.21 *Token bucket*



- The leaky bucket is very restrictive. It does not credit an idle host. For example, if a host is not sending for a while, its bucket becomes empty. Now if the host has bursty data, the leaky bucket allows only an average rate. The time when the host was idle is not taken into account. On the other hand, the token bucket algorithm allows idle hosts to accumulate credit for the future in the form of tokens. For each tick of the clock, the system sends n tokens to the bucket. The system removes one token for every cell (or byte) of data sent. For example, if n is 100 and the host is idle for 100 ticks, the bucket collects 10,000 tokens.

- **Some advantage of token Bucket over leaky bucket**
- If a bucket is full in tokens Bucket, tokens are discard not packets. While in leaky bucket, packets are discarded.
- Token Bucket can send large bursts at a faster rate while leaky bucket always sends packets at constant rate.
- **Predictable Traffic Shaping:** Token Bucket offers more predictable traffic shaping compared to leaky bucket. With token bucket, the network administrator can set the rate at which tokens are added to the bucket, and the maximum number of tokens that the bucket can hold. This allows for better control over the network traffic and can help prevent congestion.
- **Better Quality of Service (QoS):** Token Bucket provides better QoS compared to leaky bucket. This is because token bucket can prioritize certain types of traffic by assigning different token arrival rates to different classes of packets. This ensures that important packets are sent first, while less important packets are sent later, helping to ensure that the network runs smoothly.
- **More efficient use of network bandwidth:** Token Bucket allows for more efficient use of network bandwidth as it allows for larger bursts of data to be sent at once. This can be useful for applications that require high-speed data transfer or for streaming video content.

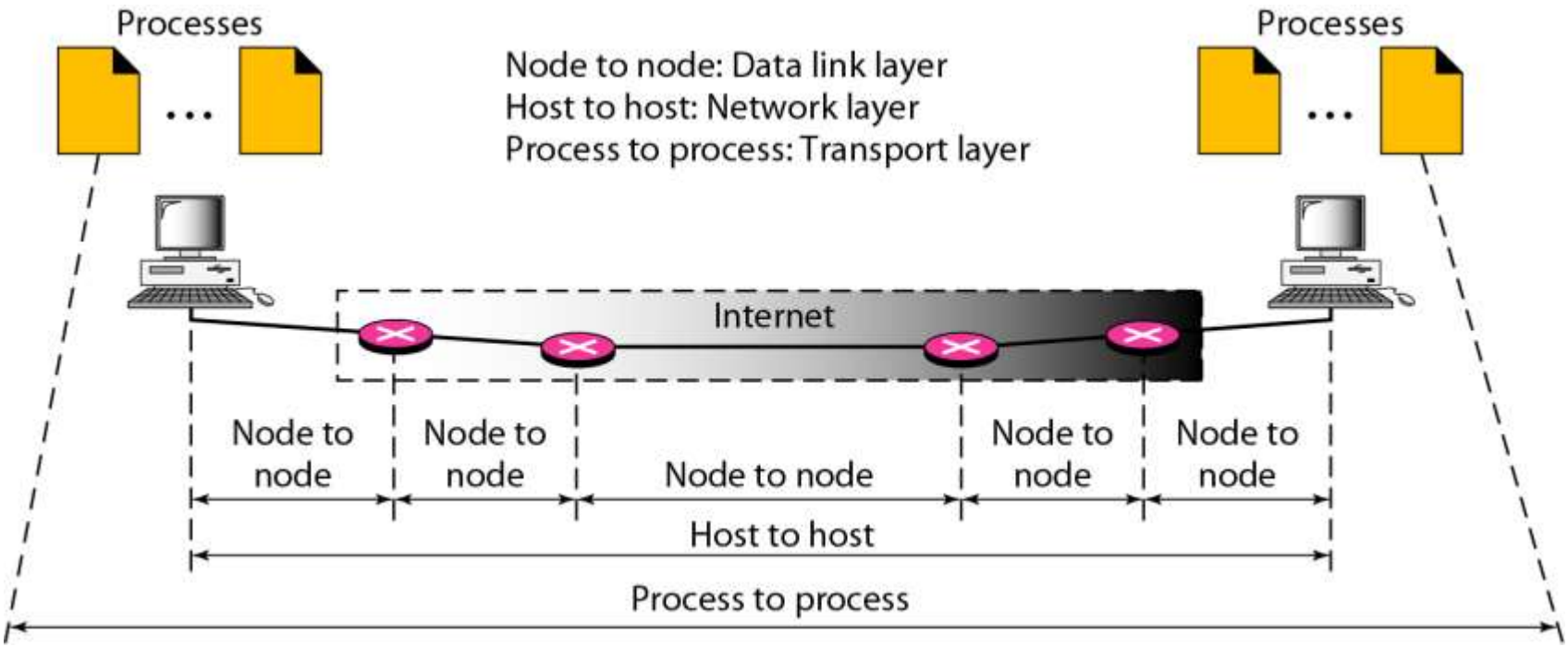
PROCESS-TO-PROCESS DELIVERY

The transport layer is responsible for process-to-process delivery—the delivery of a packet, part of a message, from one process to another. Two processes communicate in a client/server relationship, as we will see later.

Note

The transport layer is responsible for process-to-process delivery.

Figure 23.1 *Types of data deliveries*



- Client, server
- 1. Local host
- 2. Local process
- 3. Remote host
- 4. Remote process

- lANA Ranges
- The lANA (Internet Assigned Number Authority) has divided the port numbers into
- three ranges: well known, registered, and dynamic (or private)
- o Well-known ports.

- The ports ranging from 0 to 1023 are assigned and controlled
- by lANA. These are the well-known ports.

- Registered ports. The ports ranging from 1024 to 49,151 are not assigned or controlled
- by lANA.

- Dynamic ports. The ports ranging from 49,152 to 65,535 are neither controlled
- nor registered. They can be used by any process. These are the ephemeral ports

Figure 23.2 *Port numbers*

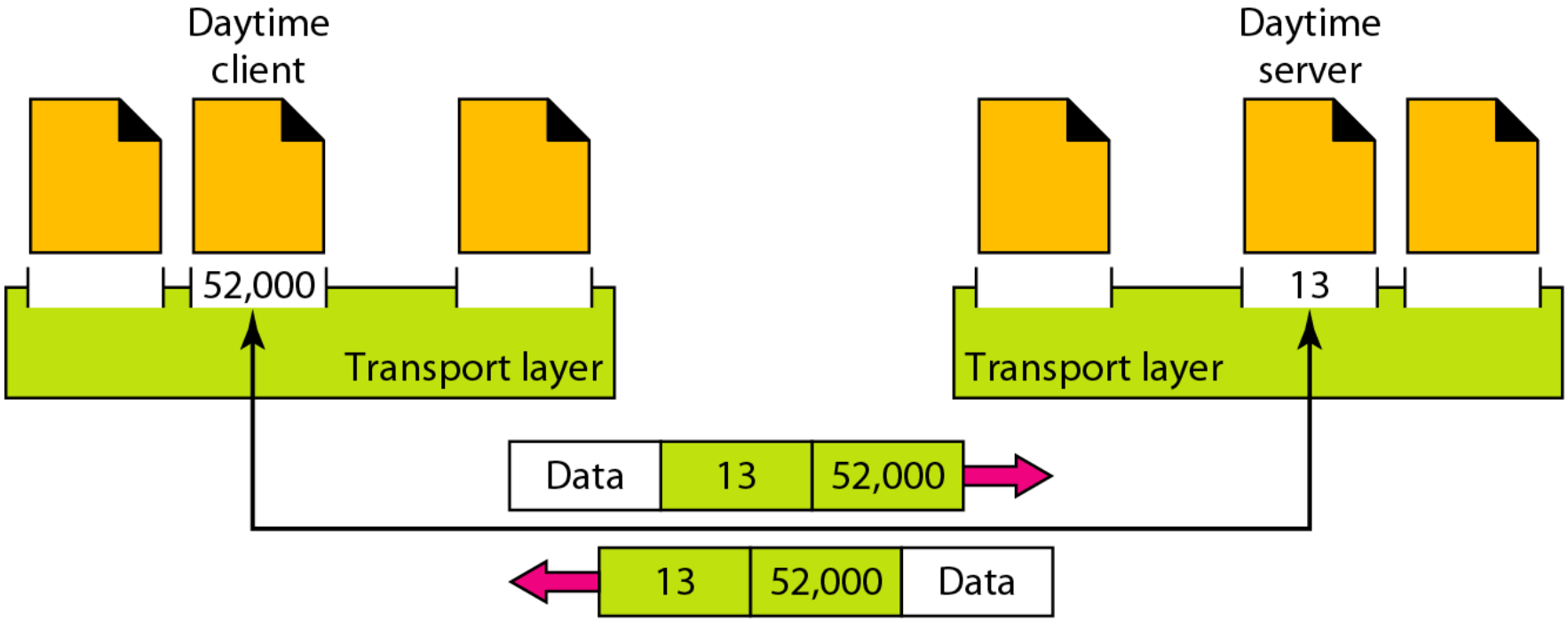


Figure 23.3 *IP addresses versus port numbers*

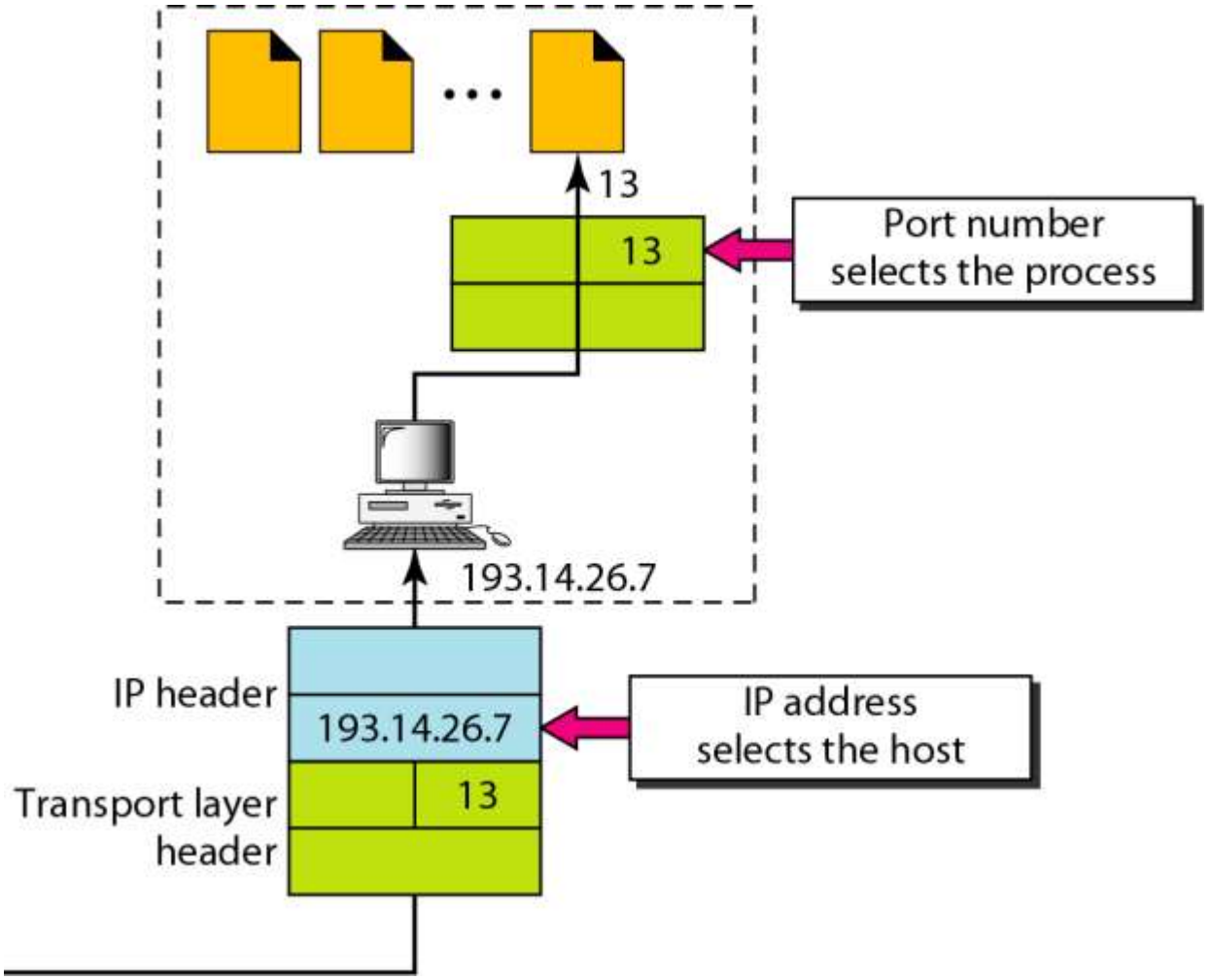


Figure 23.4 *IANA ranges*

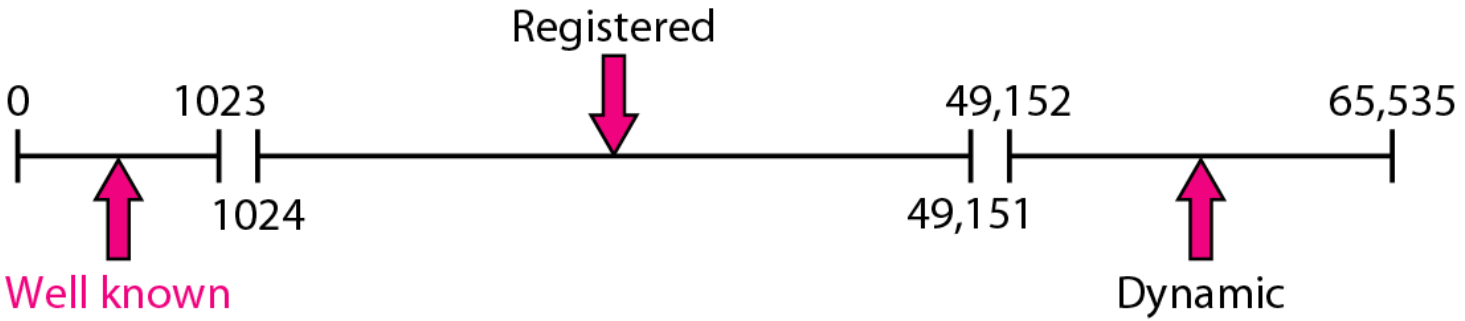


Figure 23.5 *Socket address*

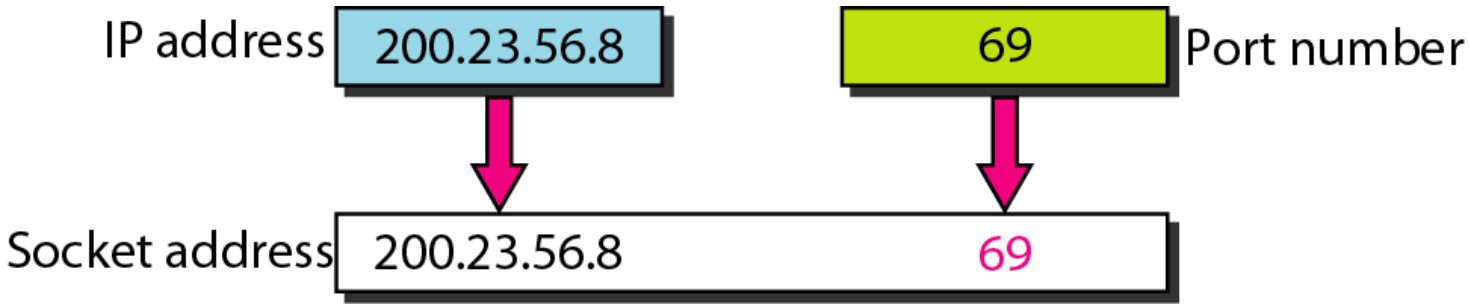
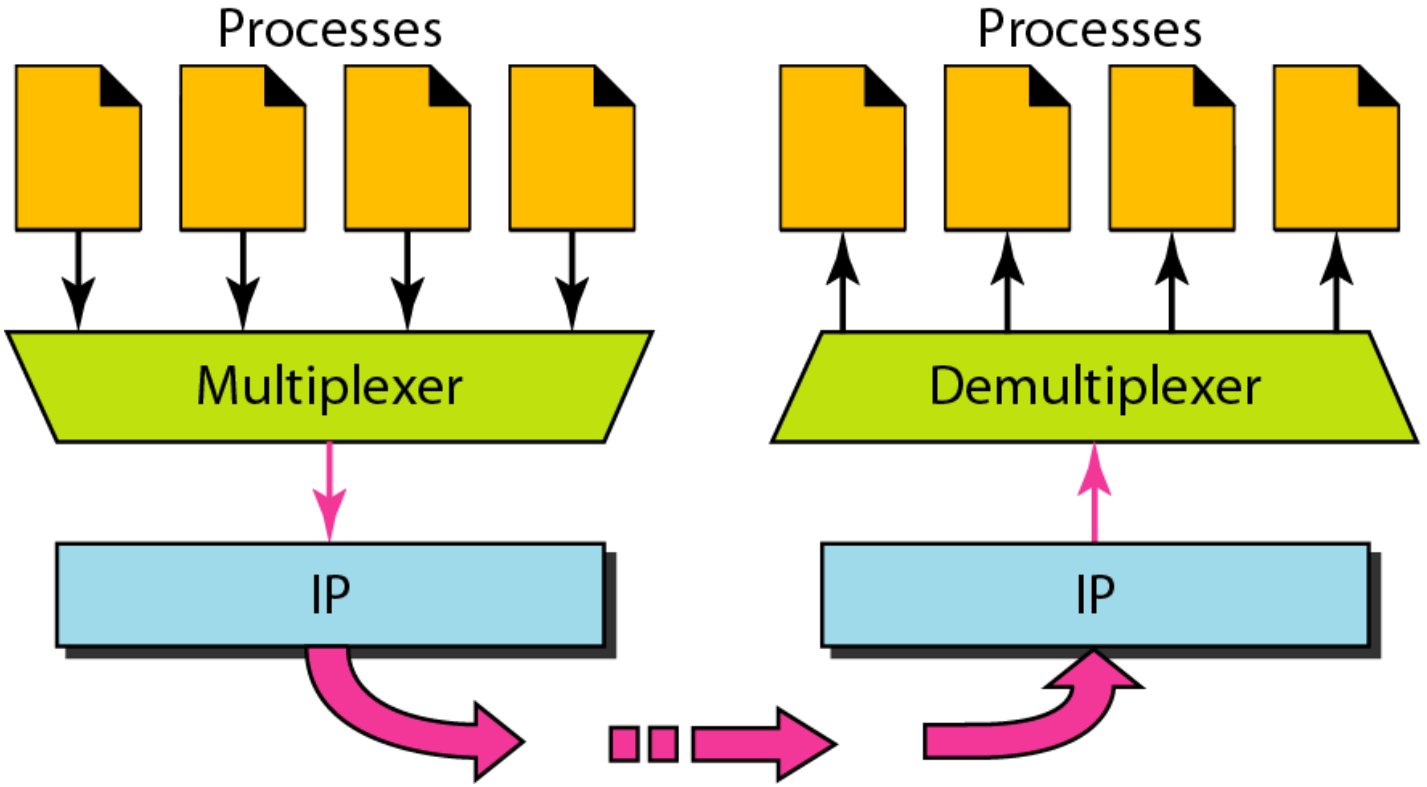


Figure 23.6 *Multiplexing and demultiplexing*



In a host running a TCP/IP protocol suite, there is only one UDP but possibly several processes that may want to use the services of UDP.

To handle this situation, UDP mul tiplexes and demultiplexes.

Connectionless Versus Connection-Oriented Service

Connectionless Service

- In a connectionless service, the packets are sent from one party to another with no need for connection establishment or connection release. The packets are not numbered; they may be delayed or lost or may arrive out of sequence. There is no acknowledgment either. We will see shortly that one of the transport layer protocols in the Internet model, UDP, is connectionless.

Connection-Oriented *Service*

- In a connection-oriented service, a connection is first established between the sender and the receiver. Data are transferred. At the end, the connection is released. TCP and SCTP are connection-oriented protocols.

Reliable Versus Unreliable

- The transport layer service can be reliable or unreliable. If the application layer program needs reliability, we use a reliable transport layer protocol by implementing flow and error control at the transport layer. This means a slower and more complex service.
- On the other hand, if the application program does not need reliability because it uses its own flow and error control mechanism or it needs fast service or the nature of the service does not demand flow and error control (real-time applications), then an unreliable protocol can be used.
- UDP is connectionless and unreliable; TCP and SCTP are connection oriented and reliable.

- These three can respond to the demands of the application layer programs.

Figure 23.7 *Error control*

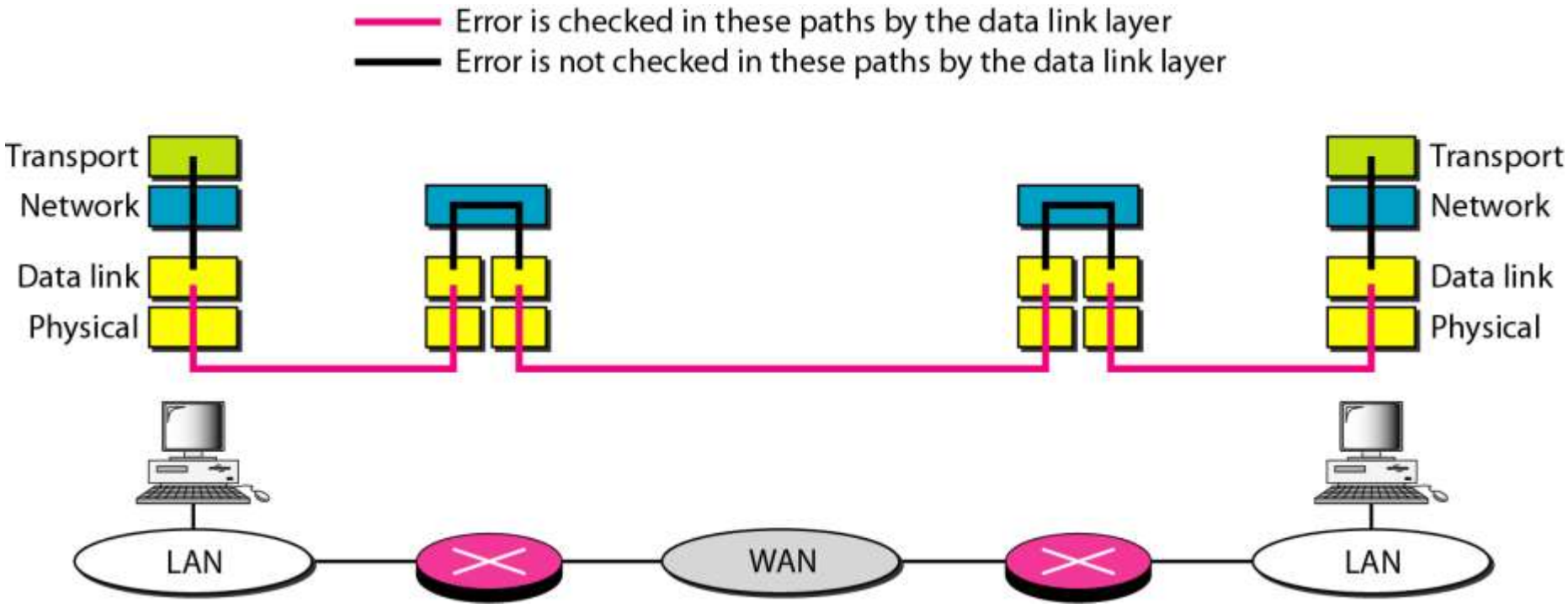
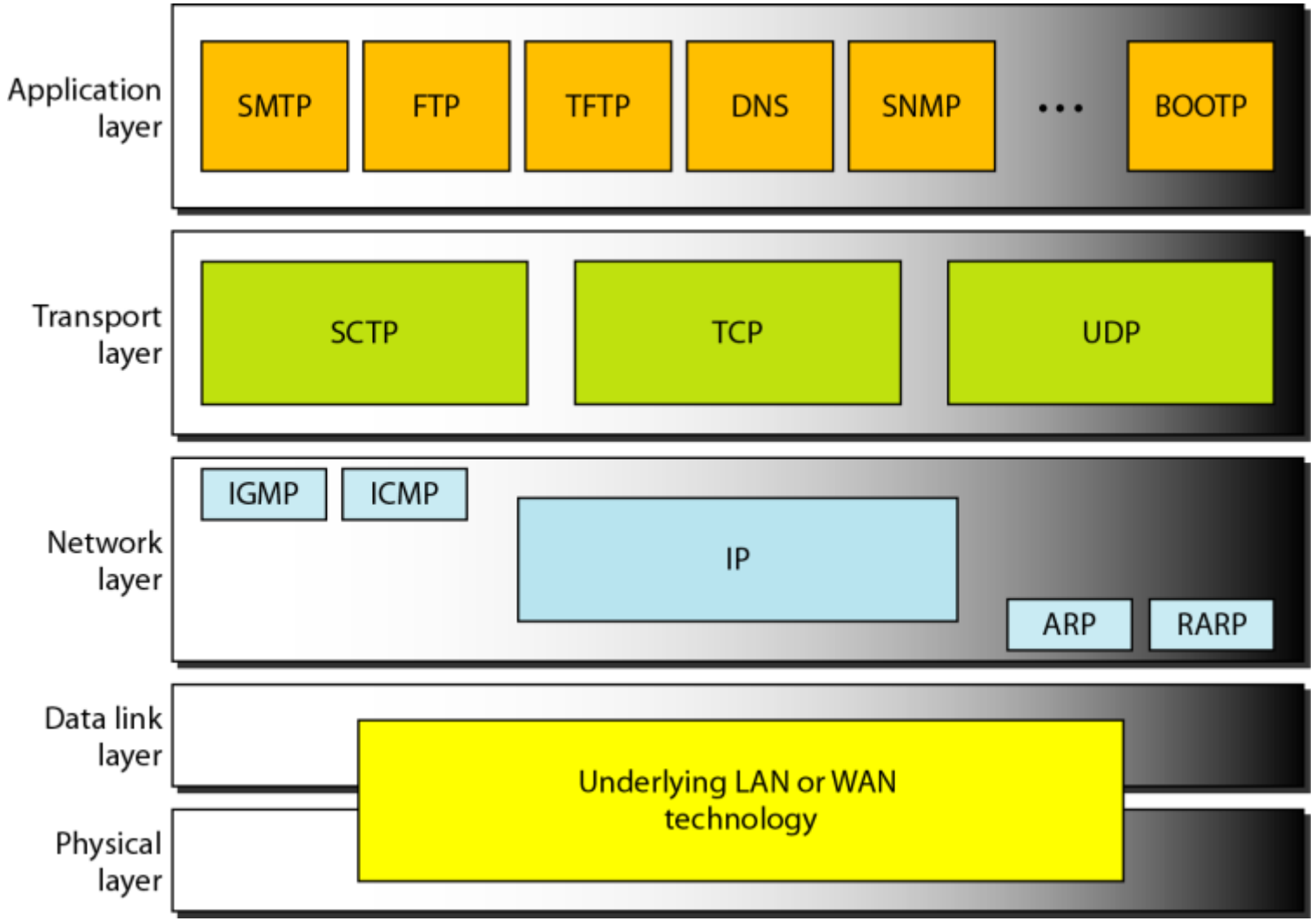


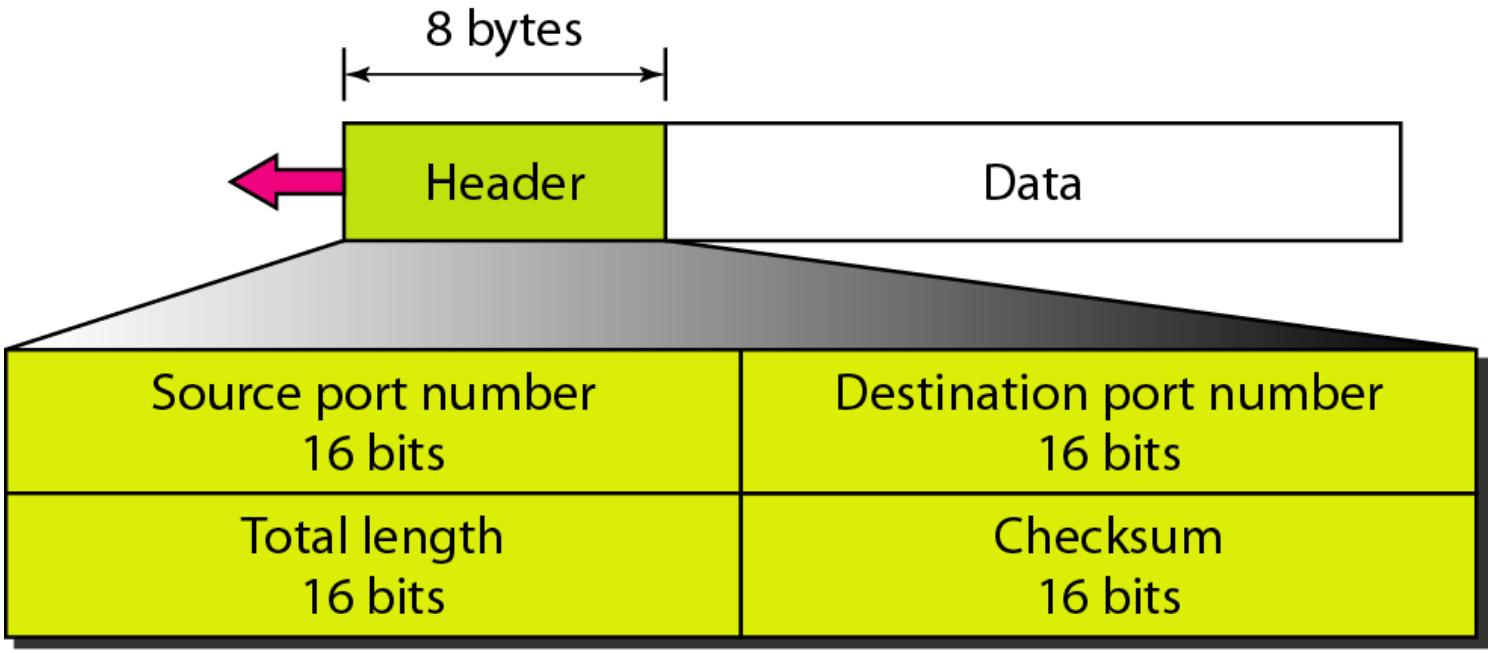
Figure 23.8 *Position of UDP, TCP, and SCTP in TCP/IP suite*



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.

Figure 23.9 *User datagram format*



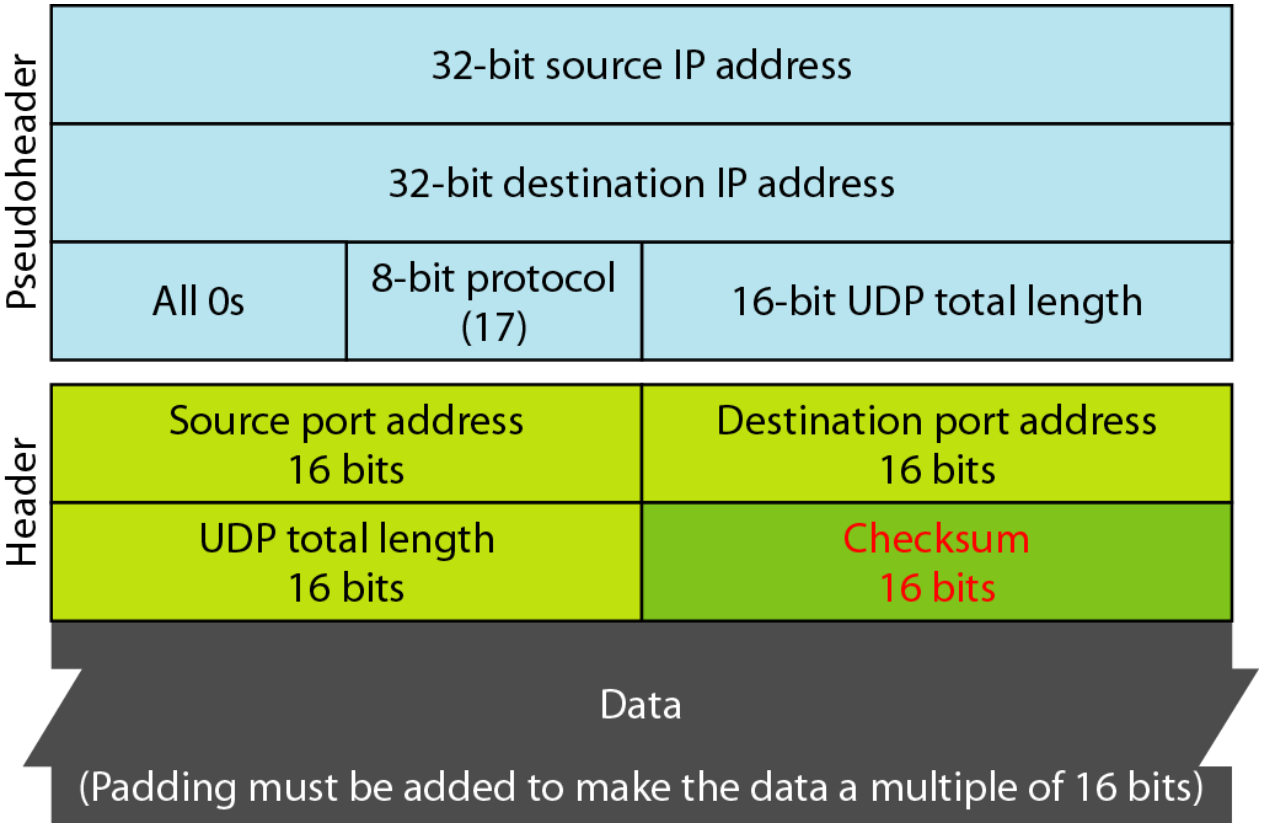
CB84000D001C001C

The source port number is the first four hexadecimal digits (CB84)16, which means that the source port number is 52100.
b. The destination port number is the second four hexadecimal digits (000D)16, which means that the destination port number is 13.
c. The third four hexadecimal digits (001C)16 define the length of the whole UDP packet as 28 bytes.
d. The length of the data is the length of the whole packet minus the length of the header, or 28 − 8 = 20 bytes.
e. Since the destination port number is 13 (well-known port), the packet is from the client to the server.
f. The client process is the Daytime

Note

UDP length
= IP length – IP header's length

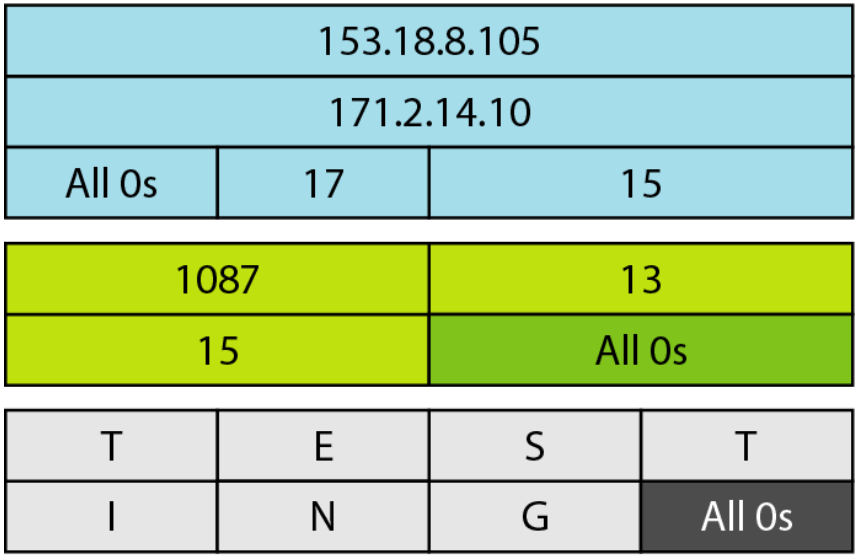
Figure 23.10 *Pseudoheader for checksum calculation*

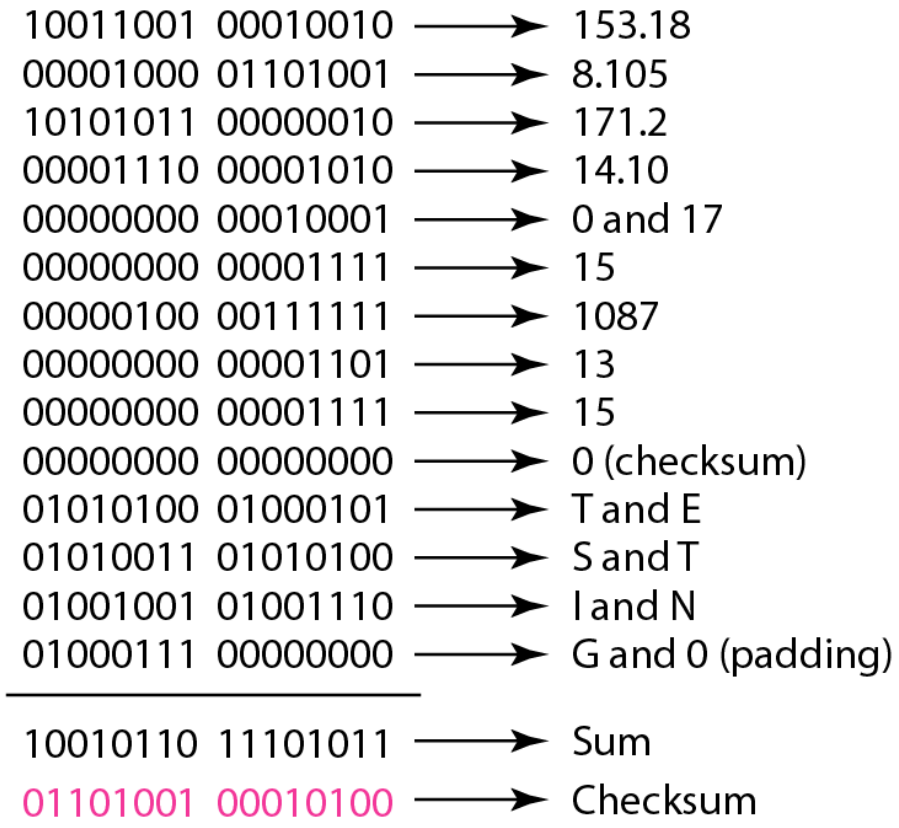


Example 23.2

Figure 23.11 shows the checksum calculation for a very small user datagram with only 7 bytes of data. Because the number of bytes of data is odd, padding is added for checksum calculation. The pseudoheader as well as the padding will be dropped when the user datagram is delivered to IP.

Figure 23.11 *Checksum calculation of a simple UDP user datagram*





UDP Operation

- Connectionless service- no relation between datagram, not numbered
- No Flow and error control- no flow control so no window mechanics.
- No error control except checksum (silently discard packet)
- Encapsulation and decapsulation – IP Datagrams - To send a message from one process to another, the UDP protocol encapsulates and decapsulates messages.

Example 23.2.3

A UDP header in hexadecimal format

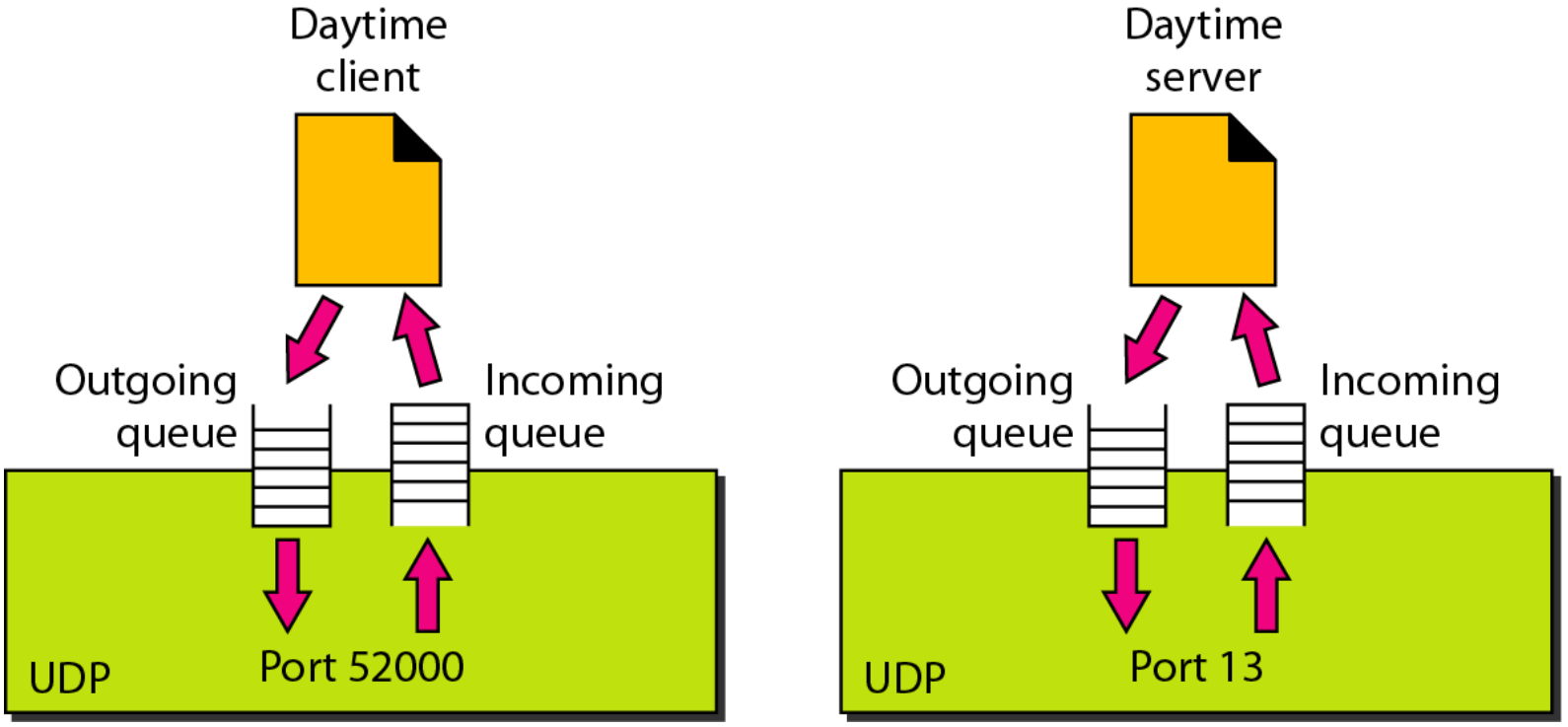
06 32 00 0D 00 1C E2 17

What is the source port number?
What is the destination port number?
What is the total length of the user datagram?
What is the length of the data?

Queuing

- In UDP, queues are associated with ports.
- At the client site, when a process starts, it requests a port number from the operat□ing system. Some implementations create both an incoming and an outgoing queue associated with each process. Other implementations create only an incoming queue associated with each process.
- Incoming and outgoing queue
- It will obtain only one port number
- Port unreachable icmp message (if queue is not created)

Figure 23.12 *Queues in UDP*



Uses of UDP

- Suitable for process that require simple request response communication with little concern for flow and error control.
- Suitable for multicasting
- Used for management process such as SNMP
- Used for routing updating protocol : RIP

23-3 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. *TCP uses checksum (for error detection), retransmission of lost or corrupted packets, cumulative and selective acknowledgments, and timers*

TCP Services
TCP Features
Segment
A TCP Connection
Flow Control
Error Control

Figure 23.13 *Stream delivery*

TCP, unlike UDP, is a stream-oriented protocol. In UDP, a process sends messages with predefined boundaries to UDP for delivery.
UDP adds its own header to each of these messages and delivers it to IP for transmission. Each message from the process is called a user datagram, and becomes, eventually, one IP datagram. Neither IP nor UDP recognizes any relationship between the datagrams.
TCP, on the other hand, allows the sending process to deliver data as a stream of bytes and allows the receiving process to obtain data as a stream of bytes.
TCP creates an environment in which the two processes seem to be connected by an imaginary “tube” that carries their bytes across the Internet.

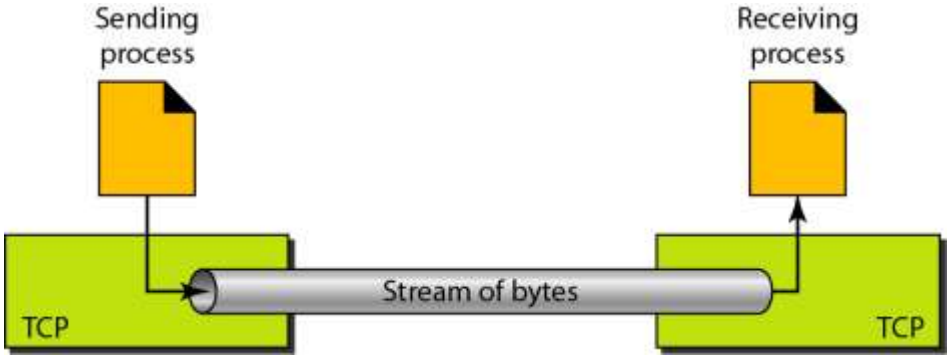


Figure 23.14 *Sending and receiving buffers*

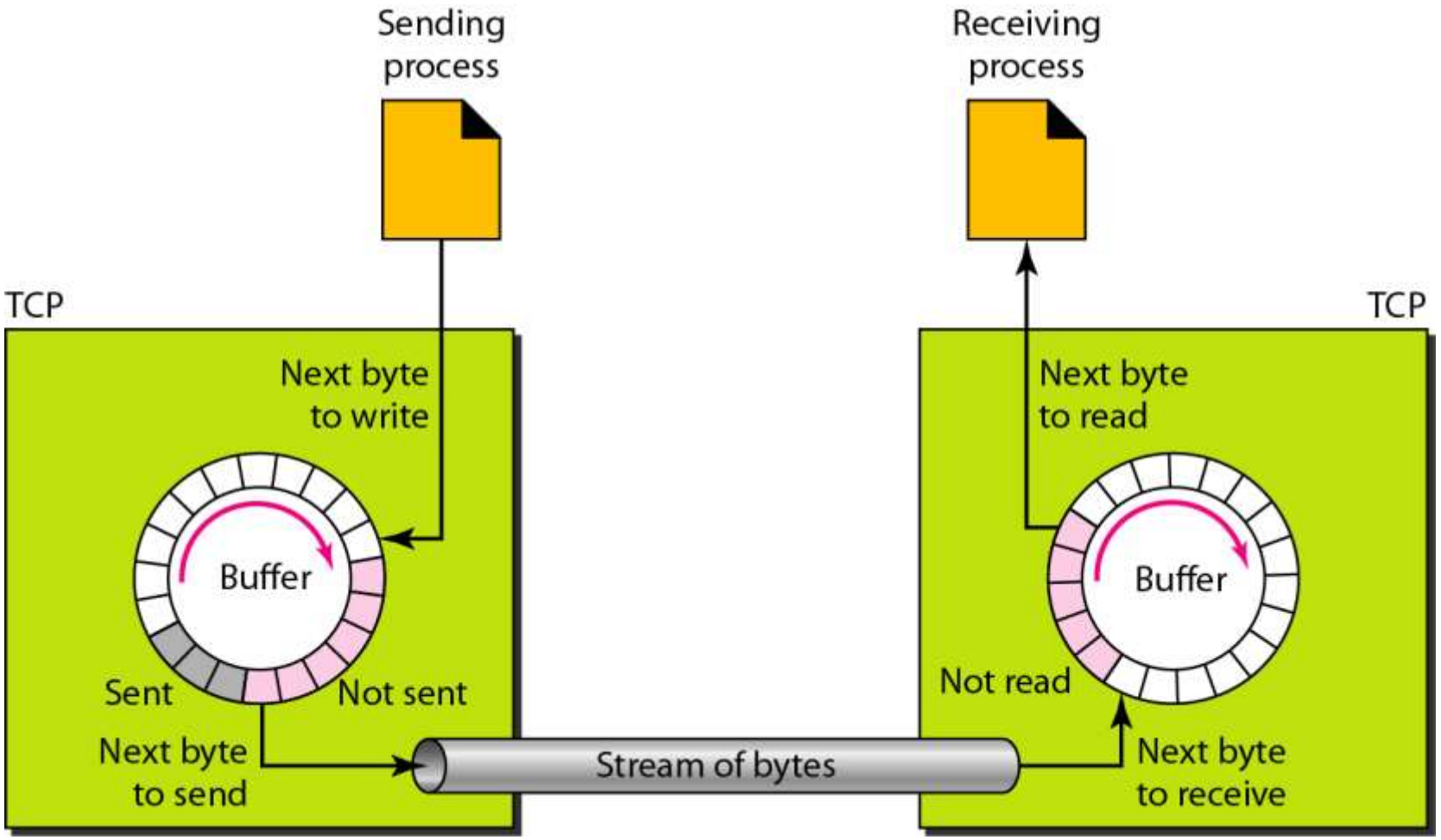
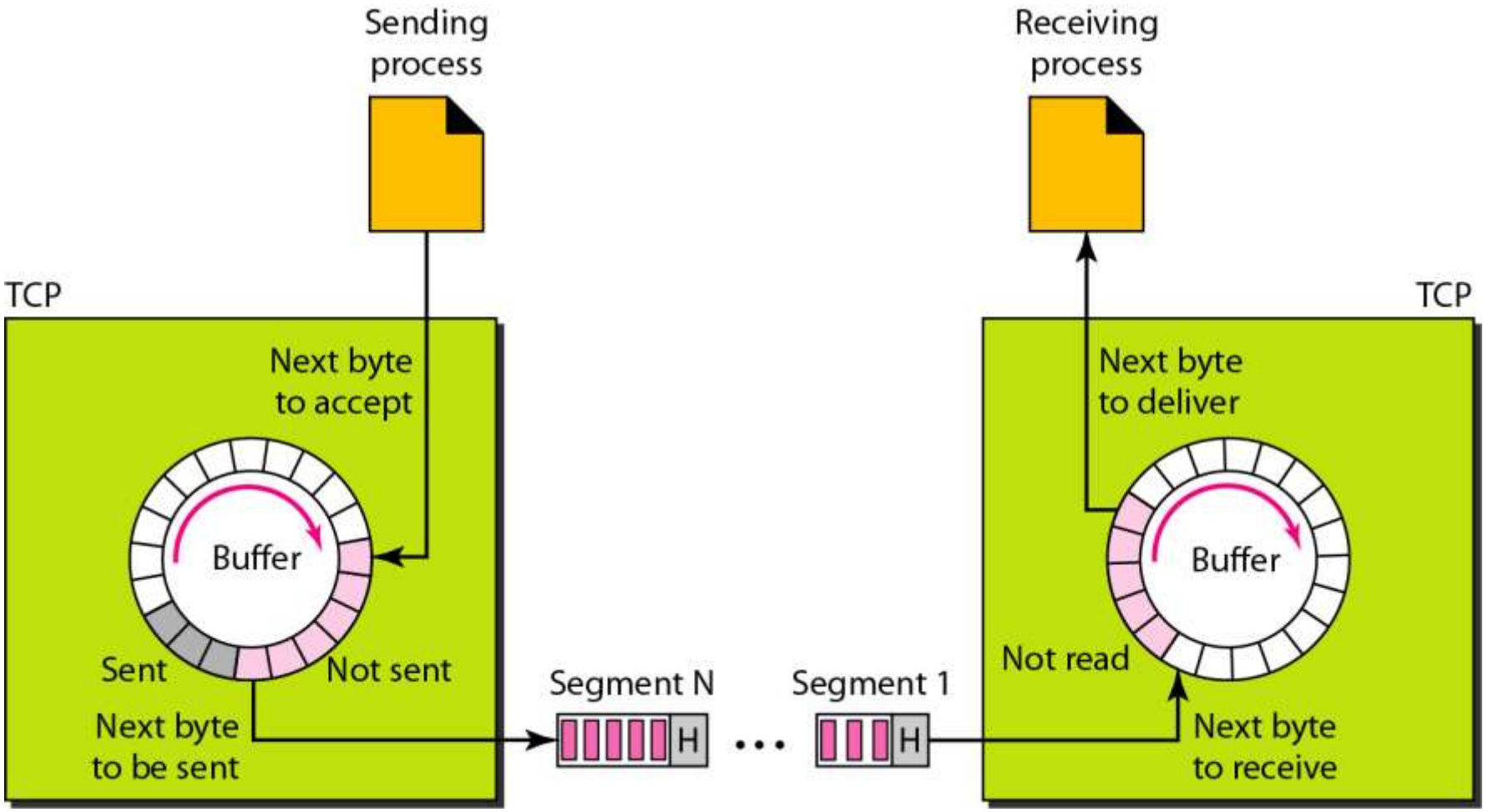


Figure 23.15 *TCP segments*



TCP

- Connection oriented phase-
- Reliable
- Features
  - Numbering system
    - No segment number – use byte number – sequence number, ack number
    - 0- 2^32 -1
- Flow control
- Error control
- Congestion control

Note

The bytes of data being transferred in each connection are numbered by TCP. The numbering starts with a randomly generated number.

Example 23.3

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 of 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.

Figure 23.16 *TCP segment format*

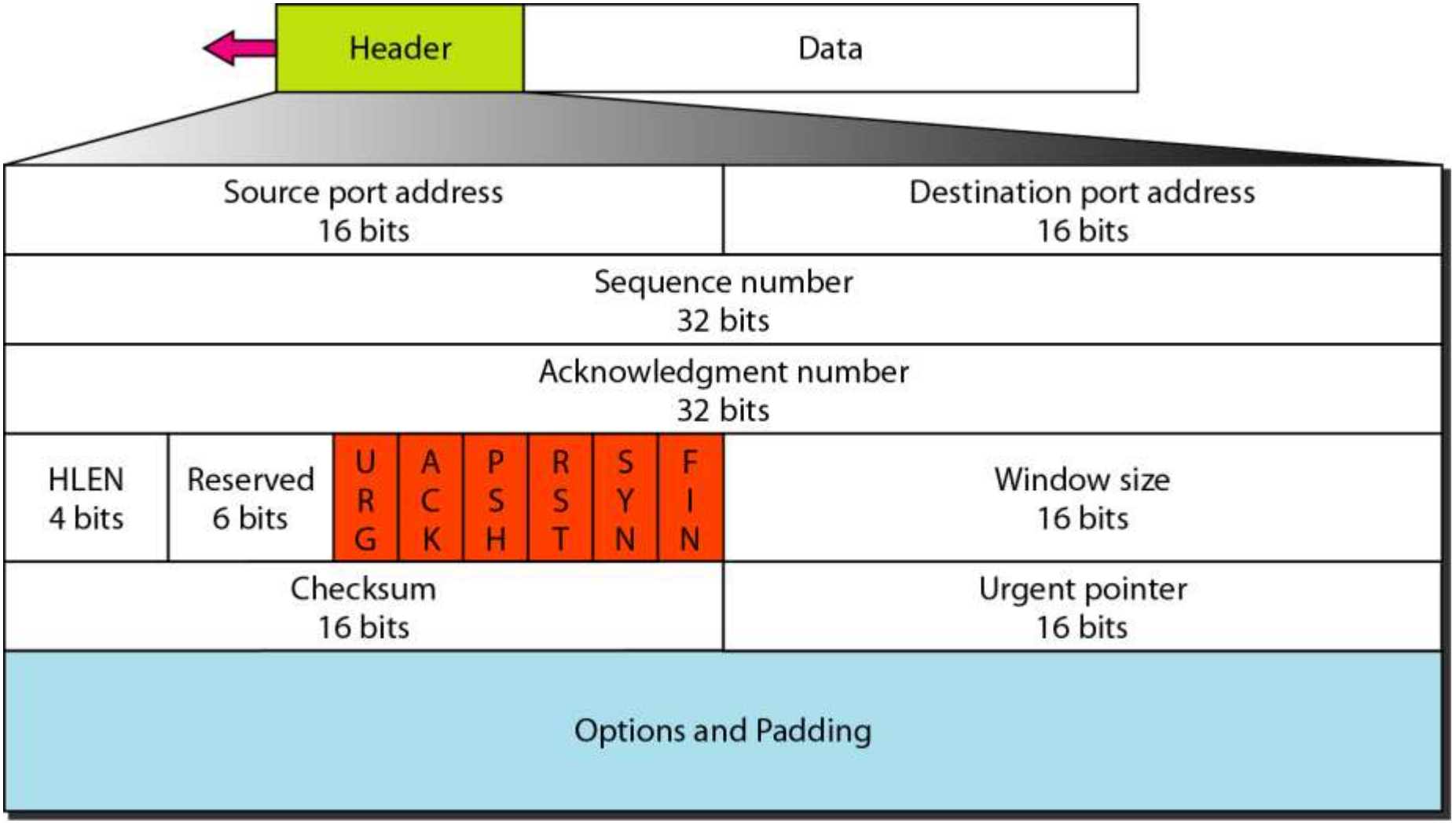


Figure 23.17 *Control field*

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

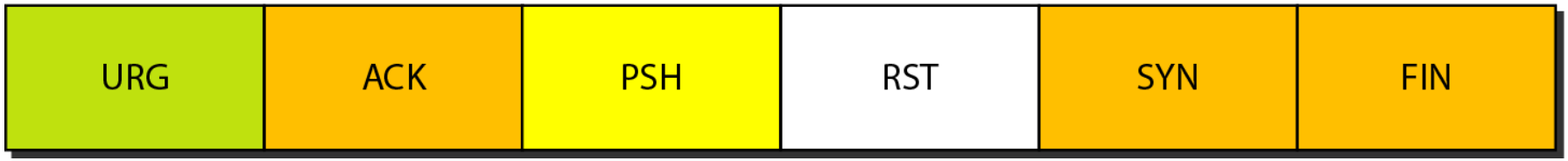


Table 23.3 ***Description of flags in the control field***

Flag	*Description*
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.

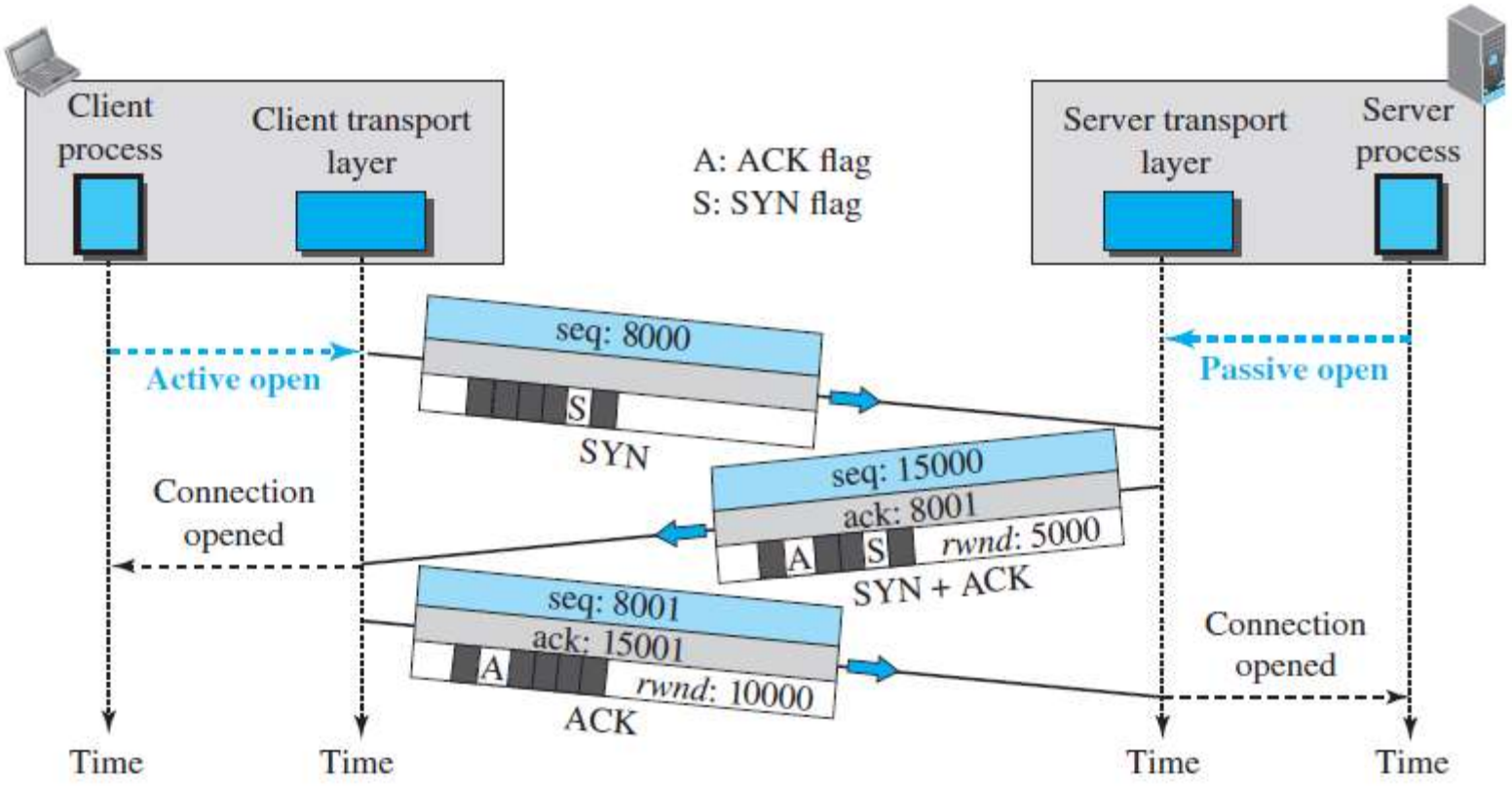
Example 23.2.4

The following is a dump of a TCP header in hexadecimal format

05320017 00000001 00000000 500207FF 00000000

What is the source port number?
What is the destination port number?
What is sequence number?
What is the acknowledgment number?
What is the length of the header?
What is the type of the segment?
What is the window size?

Figure 23.18 *Connection establishment using three-way handshaking*



Note

A SYN segment cannot carry data, but it consumes one sequence number.

Note

A SYN + ACK segment cannot carry data, but does consume one sequence number.

Note

An ACK segment, if carrying no data, consumes no sequence number.

- Syn flooding attack
- Simultaneous open

Figure 23.19 *Data transfer*

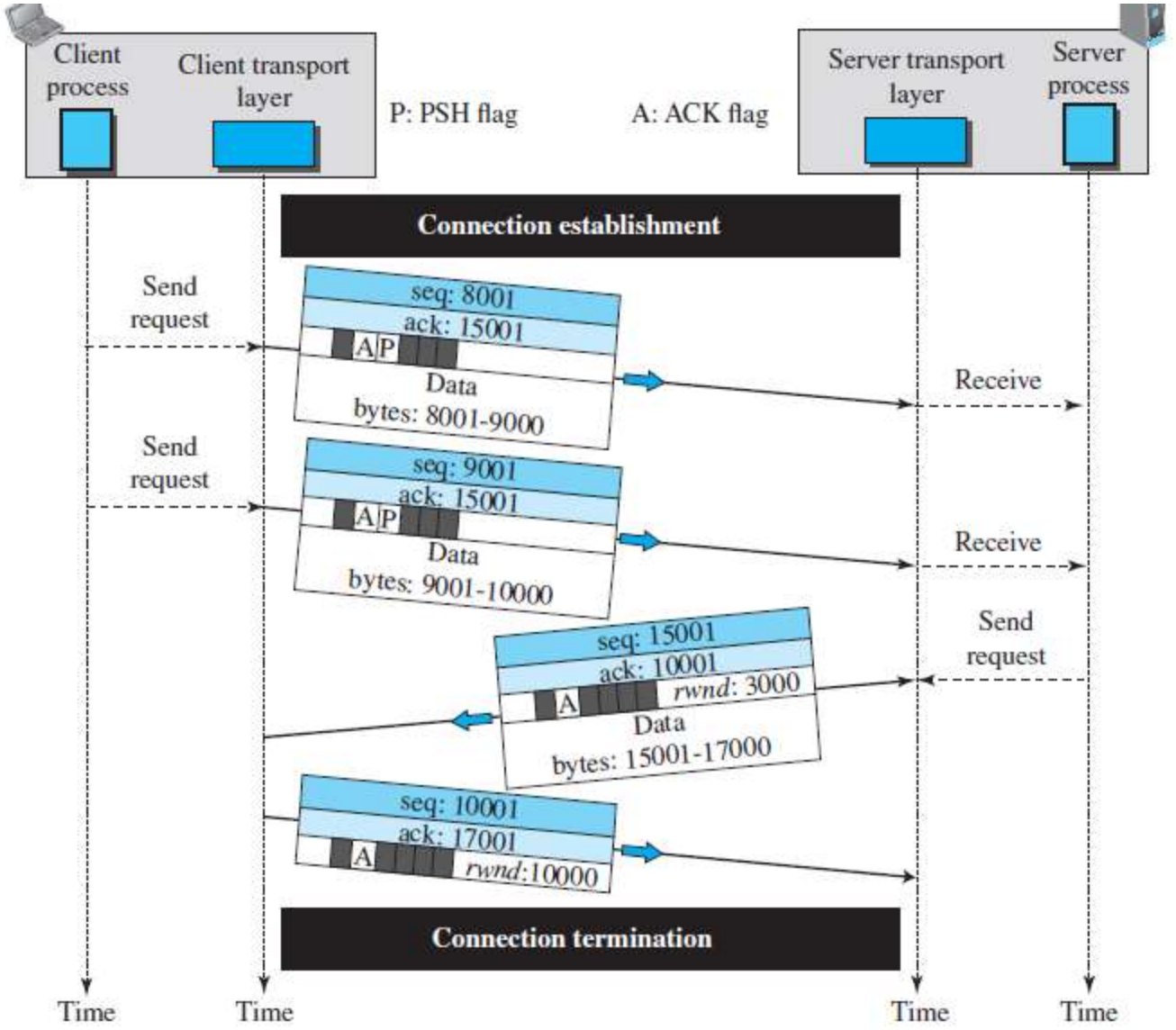
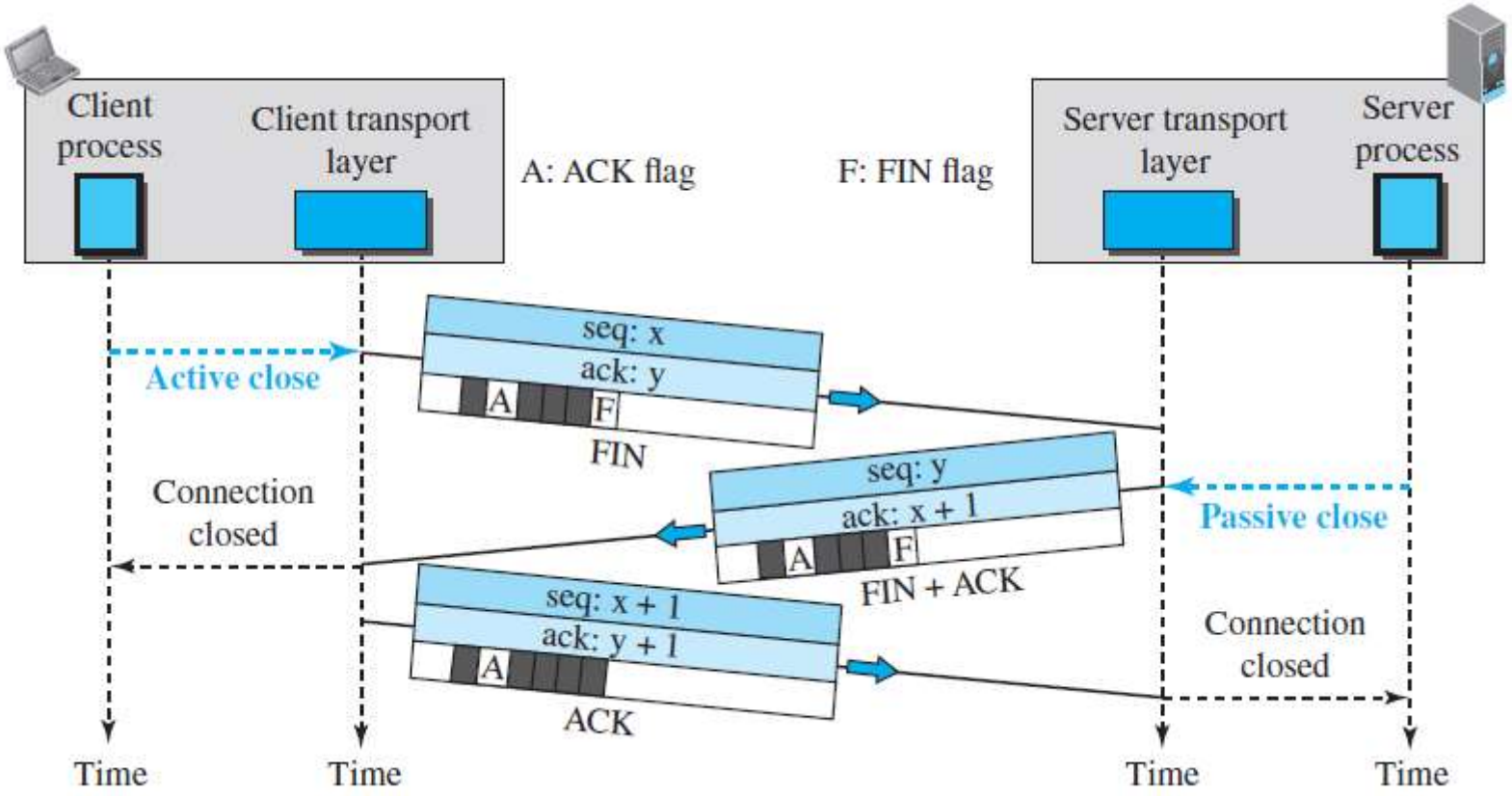


Figure 23.20 *Connection termination using three-way handshaking*



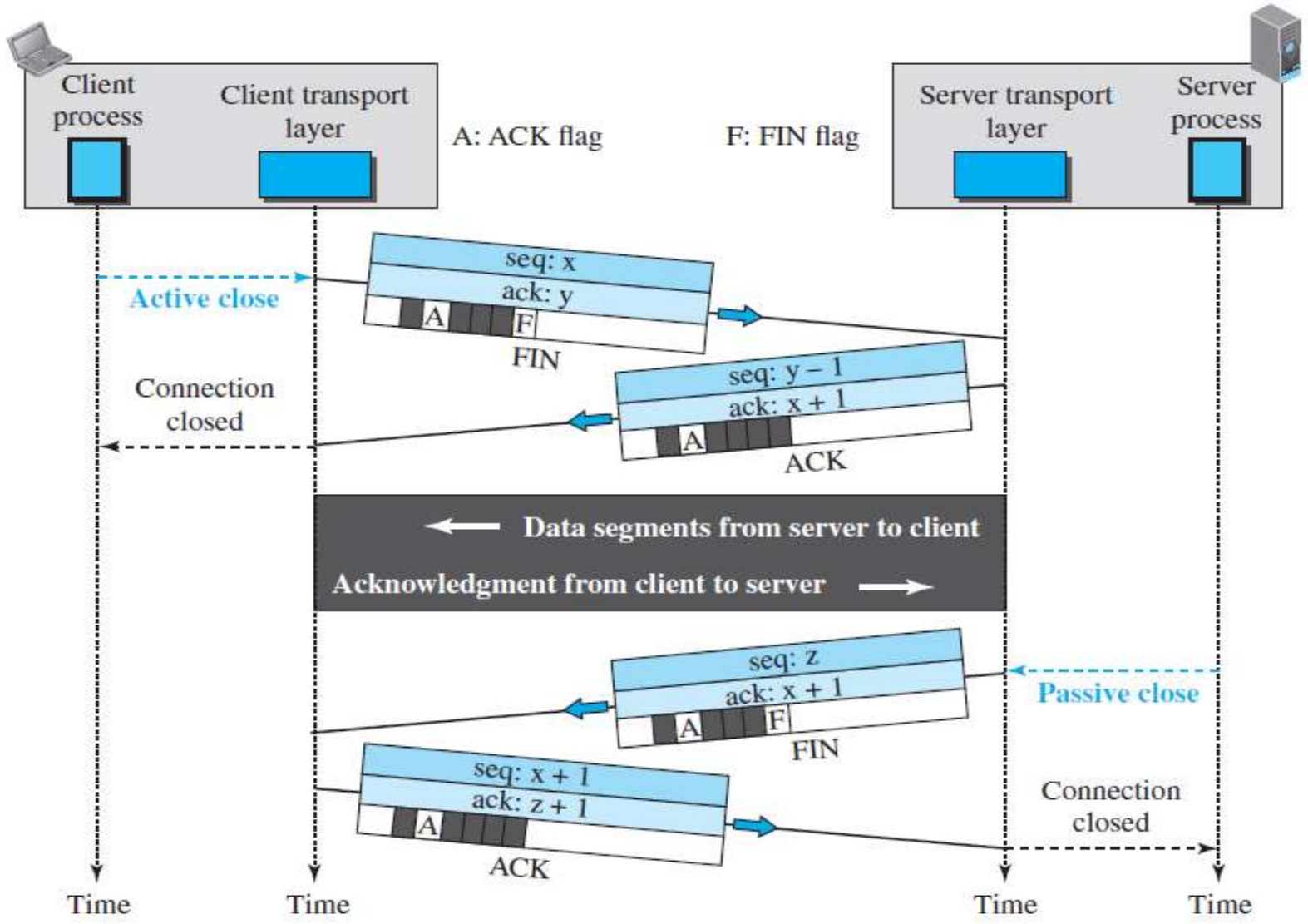
Note

The FIN segment consumes one sequence number if it does not carry data.

Note

The FIN + ACK segment consumes one sequence number if it does not carry data.

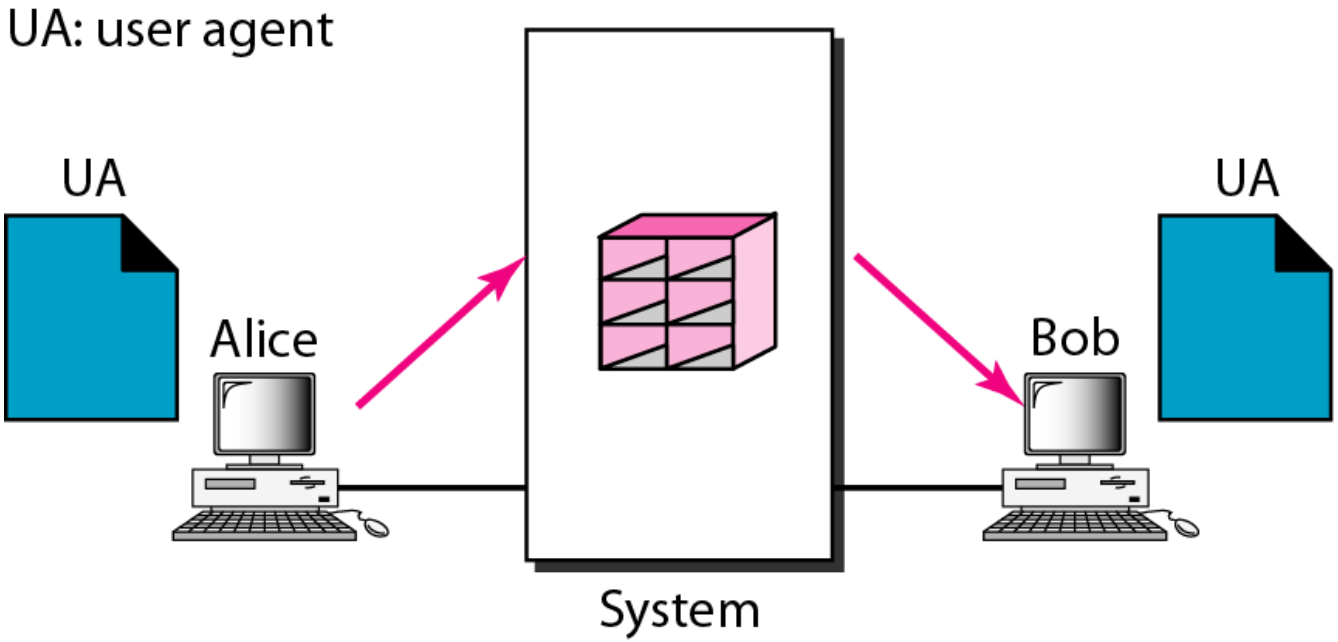
Figure 23.21 *Half-close*



ELECTRONIC MAIL

One of the most popular Internet services is electronic mail (e-mail). The designers of the Internet probably never imagined the popularity of this application program. Its architecture consists of several components that we discuss in this chapter.

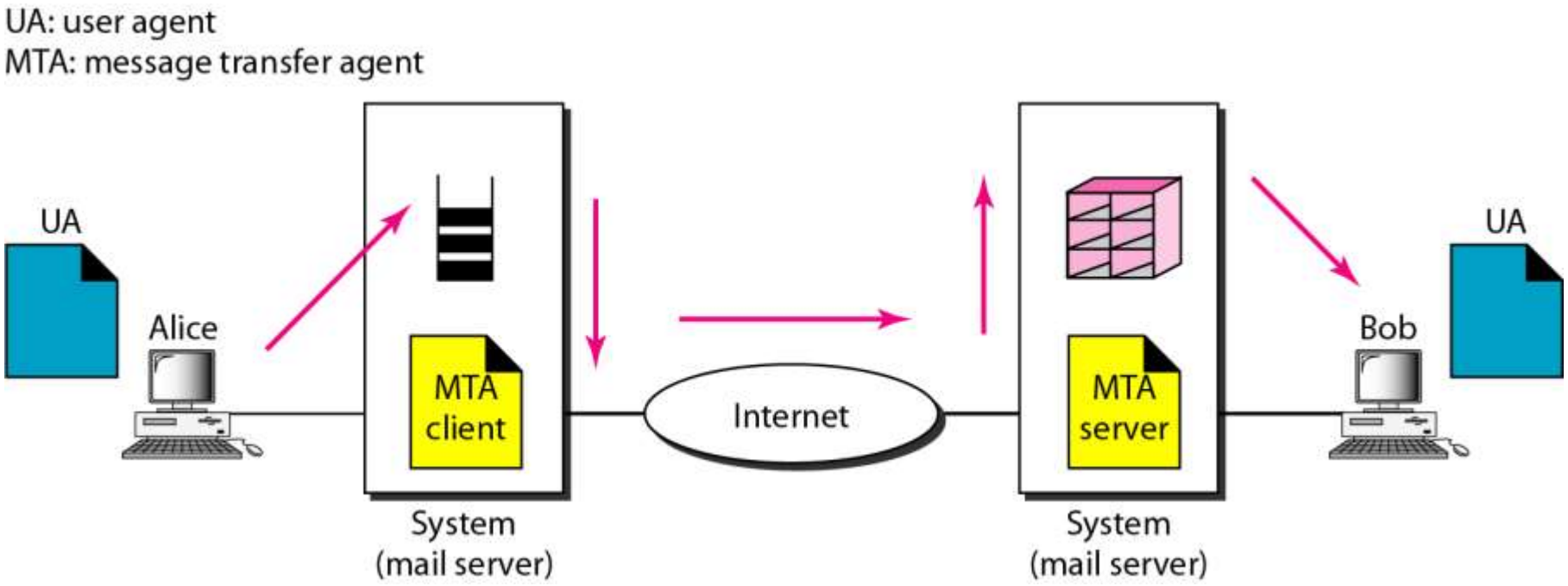
Figure 26.6 *First scenario in electronic mail*



Note

When the sender and the receiver of an e-mail are on the same system, we need only two user agents.

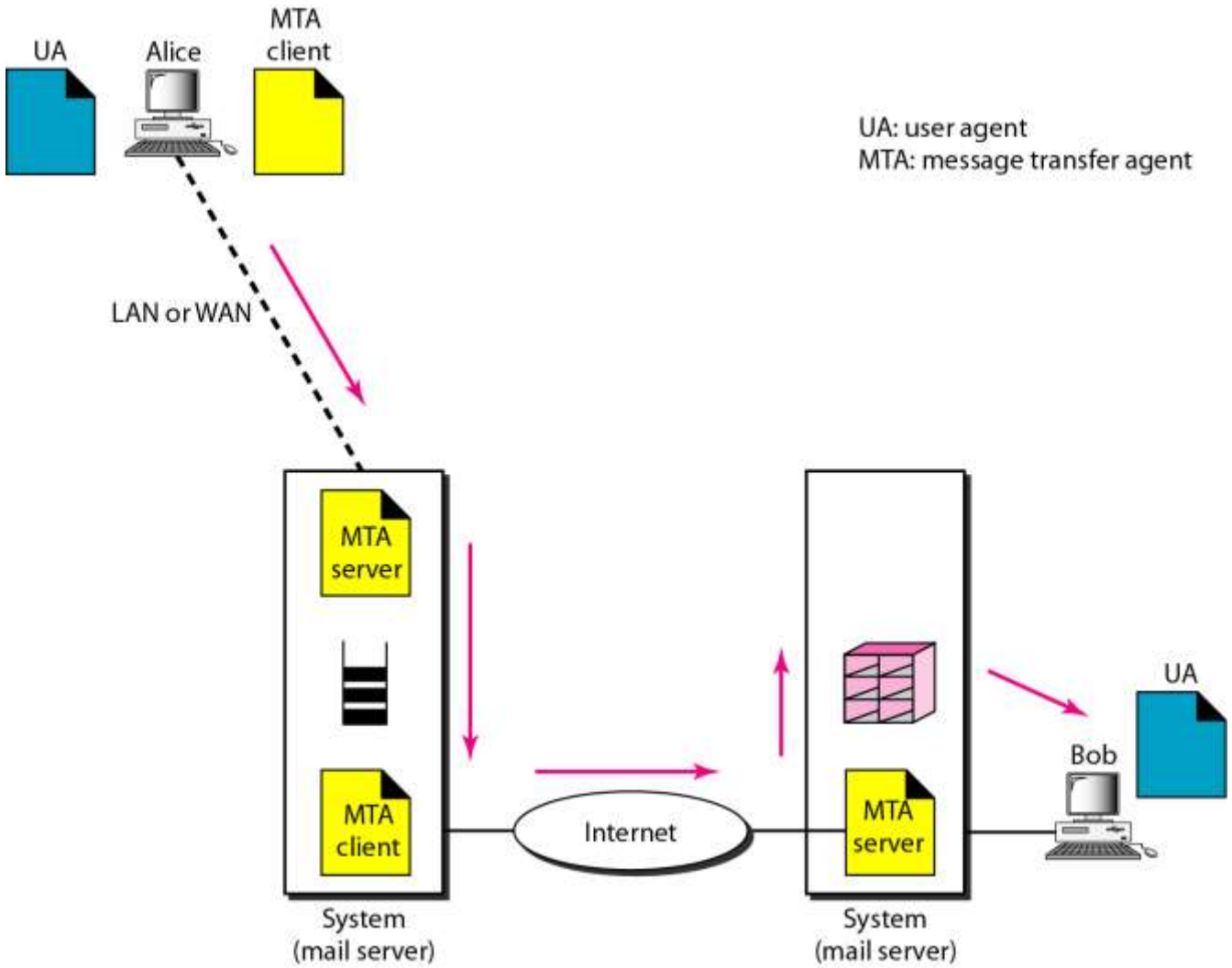
Figure 26.7 *Second scenario in electronic mail*



Note

When the sender and the receiver of an e-mail are on different systems, we need two UAs and a pair of MTAs (client and server).

Figure 26.8 *Third scenario in electronic mail*



Note

When the sender is connected to the mail server via a LAN or a WAN, we need two UAs and two pairs of MTAs (client and server).

Figure 26.9 *Fourth scenario in electronic mail*

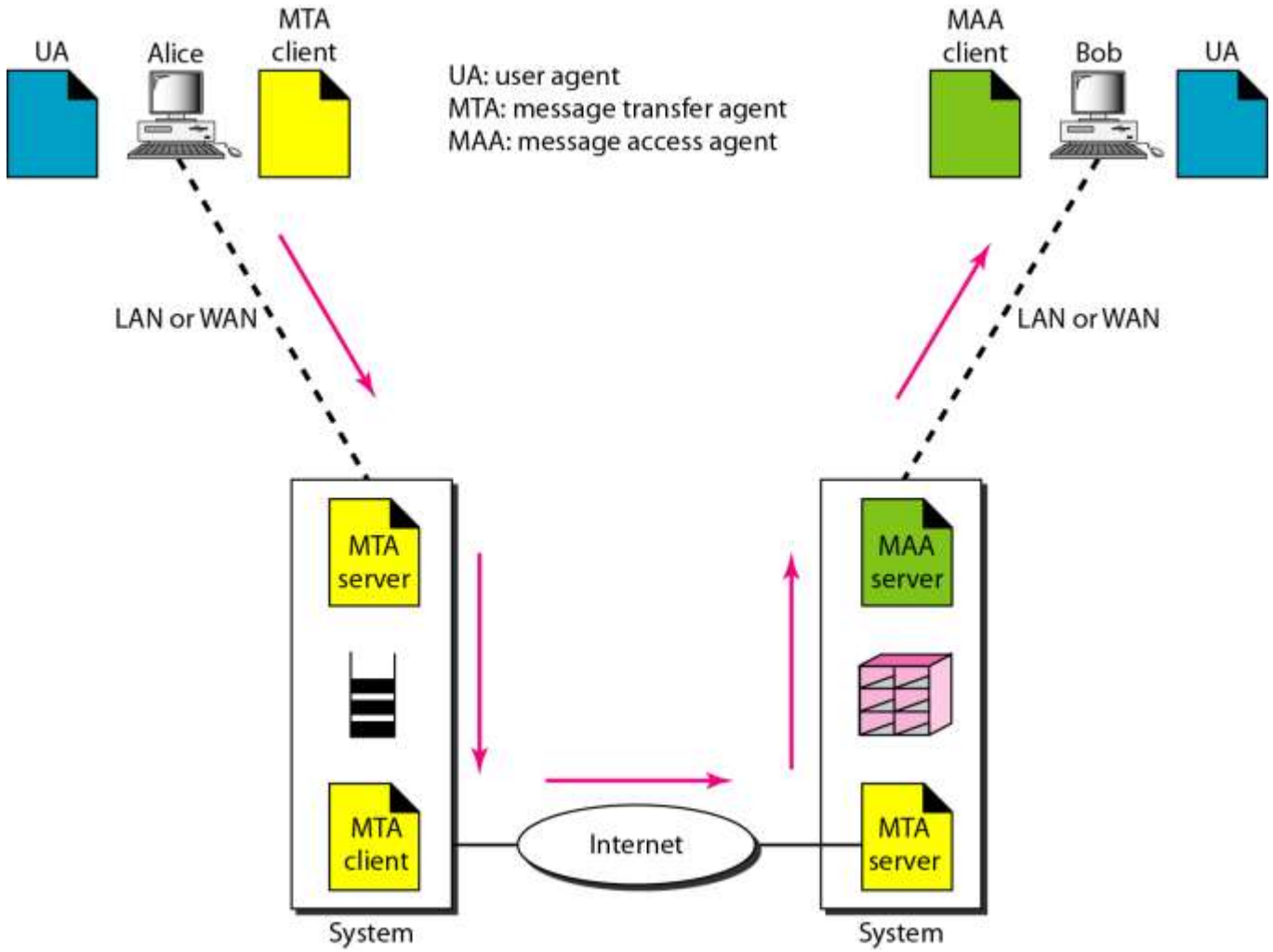
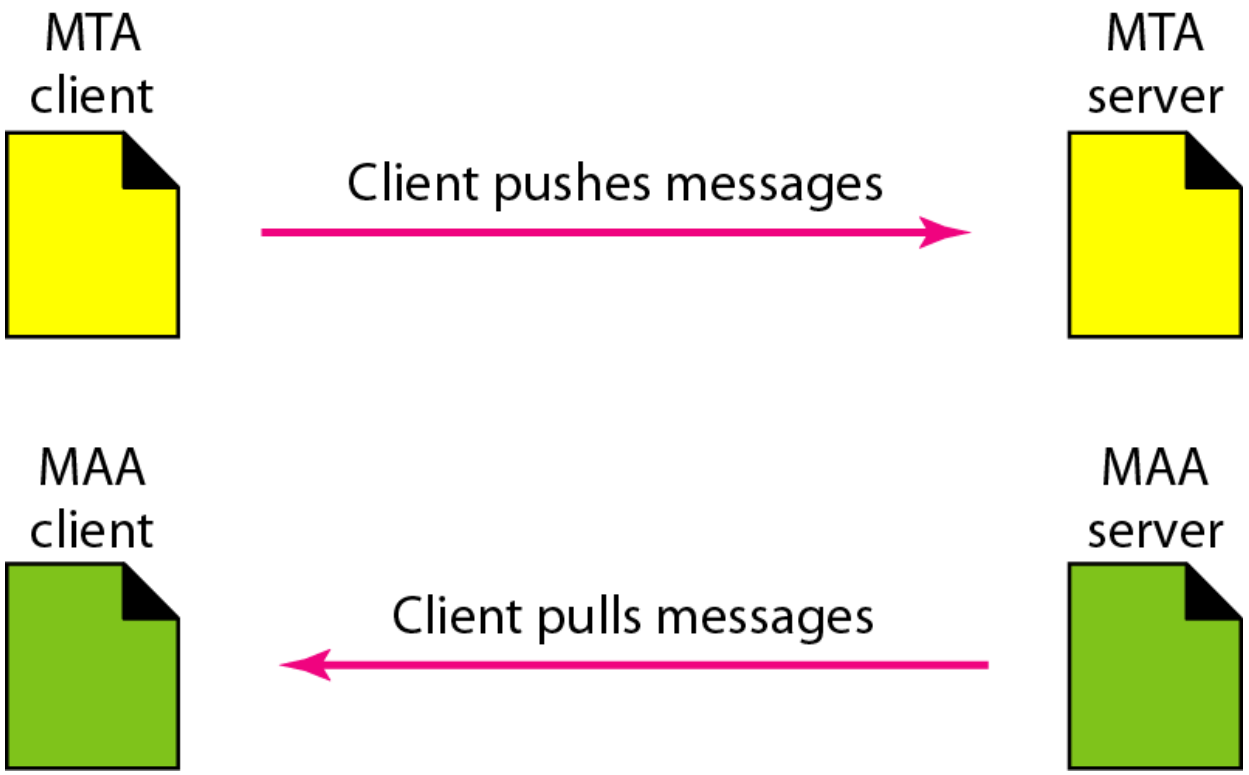


Figure 26.10 *Push versus pull in electronic email*

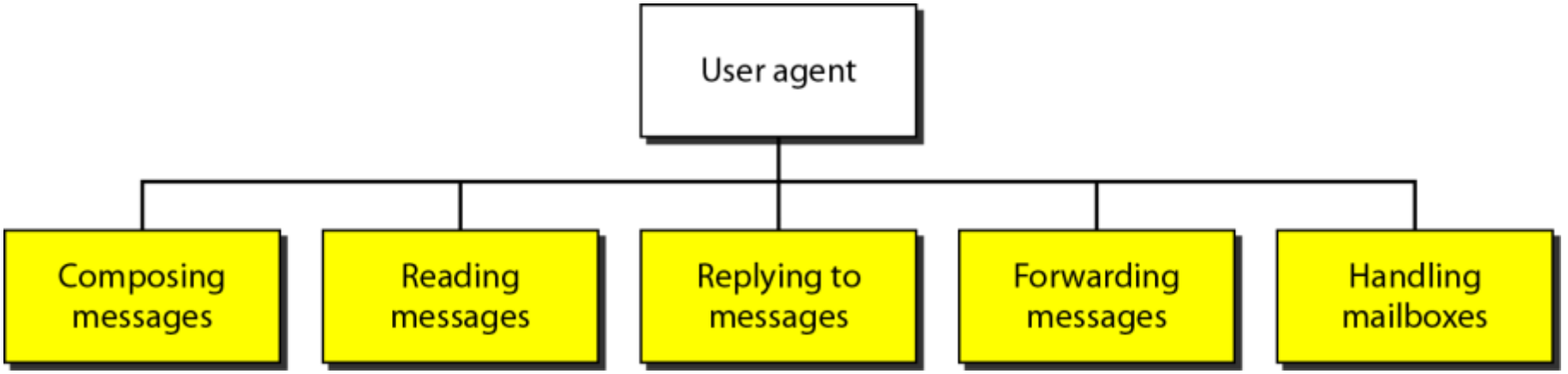


Note

When both sender and receiver are connected to the mail server via a LAN or a WAN, we need two UAs, two pairs of MTAs and a pair of MAAs.

This is the most common situation today.

Figure 26.11 *Services of user agent*



Note

Some examples of command-driven user agents are *mail*, *pine*, and *elm*.

Note

Some examples of GUI-based user agents are *Eudora*, *Outlook*, and *Netscape*.

Figure 26.12 *Format of an e-mail*

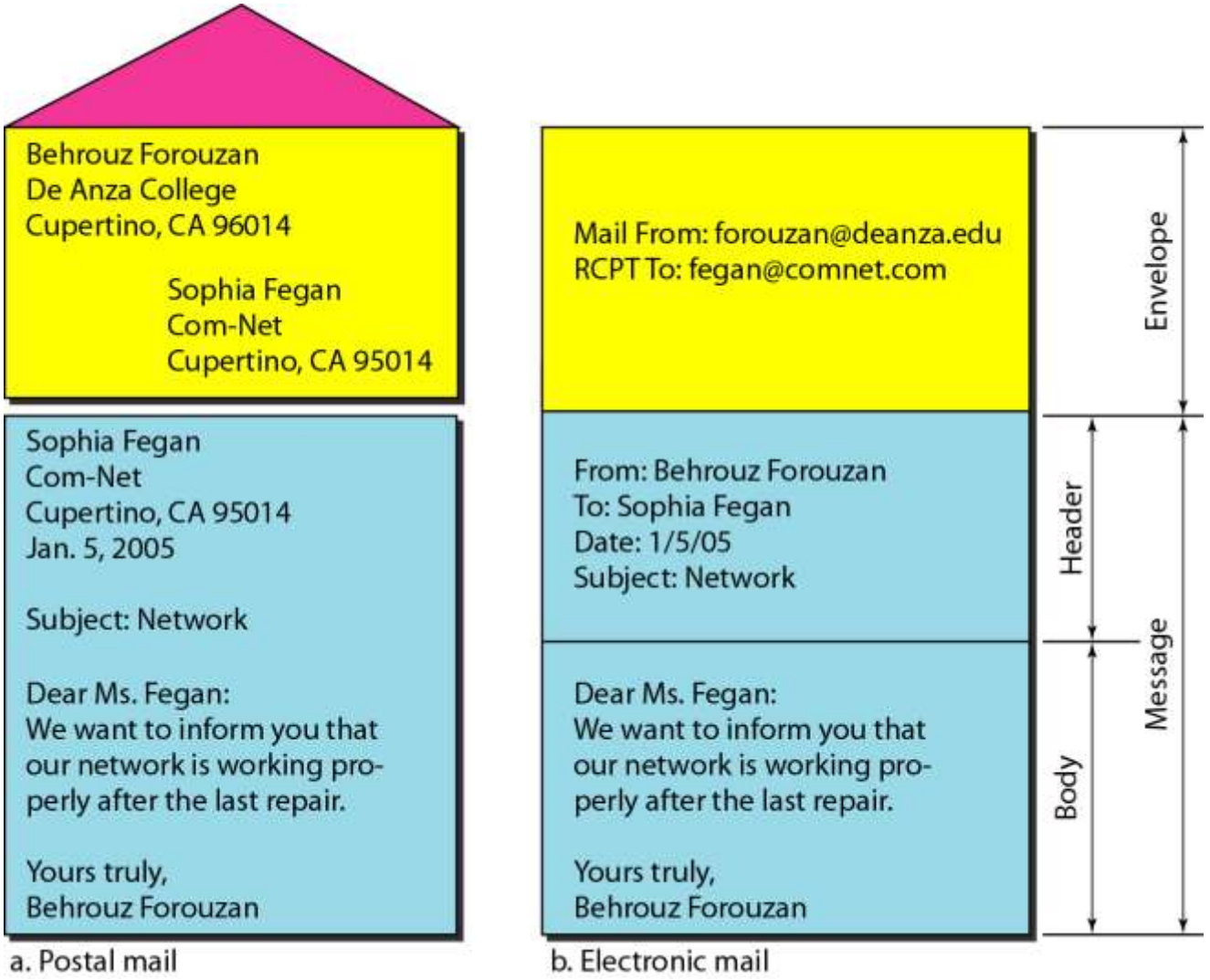


Figure 26.13 *E-mail address*

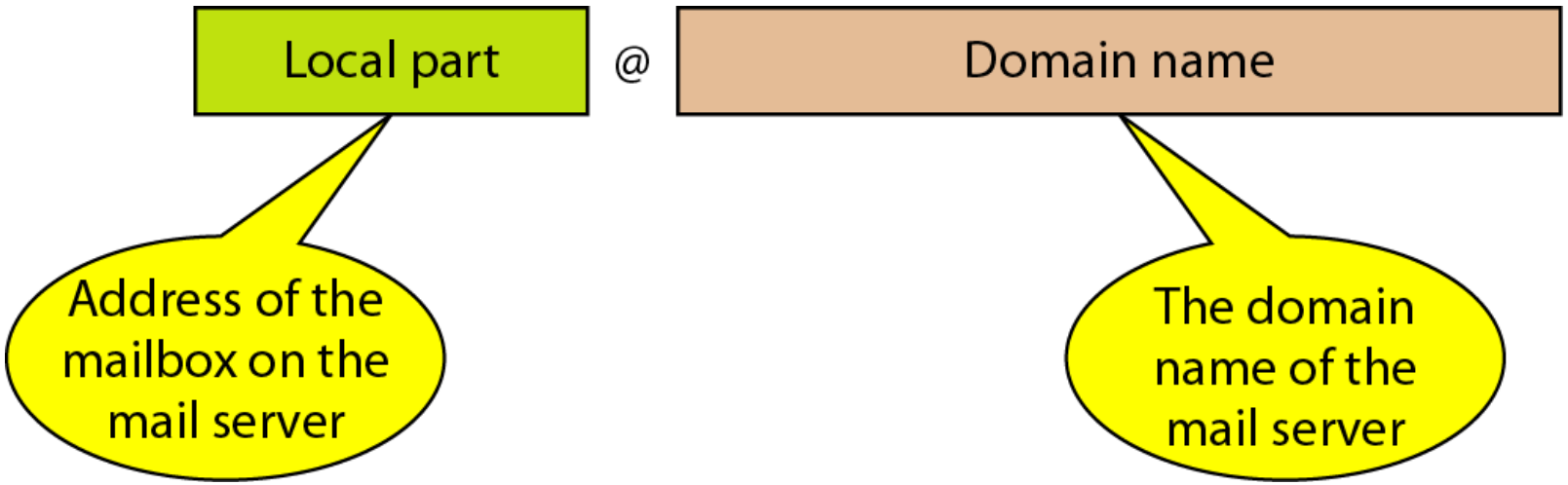


Figure 26.17 *Commands and responses*

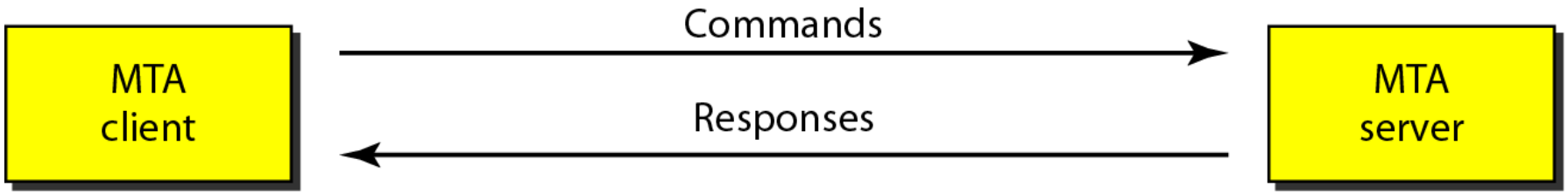


Figure 26.18 *Command format*

Table 26.7 *Commands*

Keyword	*Argument(s)*
HELO	Sender's host name
MAIL FROM	Sender of the message
RCPT TO	Intended recipient of the message
DATA	Body of the mail
QUIT	
RSET	
VRFY	Name of recipient to be verified
NOOP	
TURN	
EXPN	Mailing list to be expanded
HELP	Command name
SEND FROM	Intended recipient of the message
SMOL FROM	Intended recipient of the message
SMAL FROM	Intended recipient of the message

NAME SPACE

To be unambiguous, the names assigned to machines must be carefully selected from a name space with complete control over the binding between the names and IP addresses.

DOMAIN NAME SPACE

- In the postal system
  - the country, state or province, city, street address and name of the addressee.

- Hierarchical addressing ensures that there is NO confusion between the
  - Street no. 2 in Azad Nagar in Jalandhar, Punjab and the Street no. 2 in Azad Nagar in Ambala, Haryana.

- DNS works the same way.

Figure 25.1 *Example of using the DNS service*

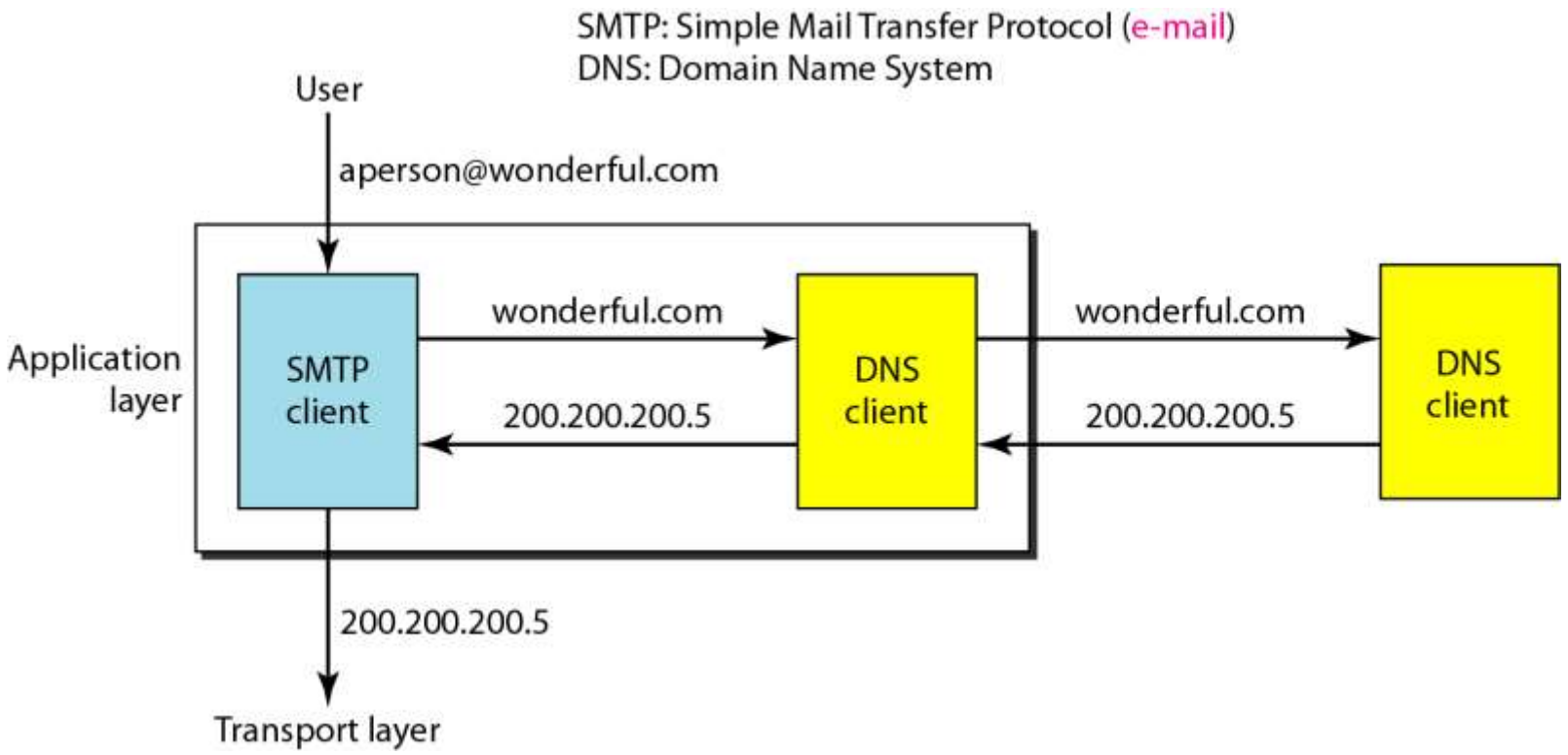


Figure 25.2 *Domain name space*

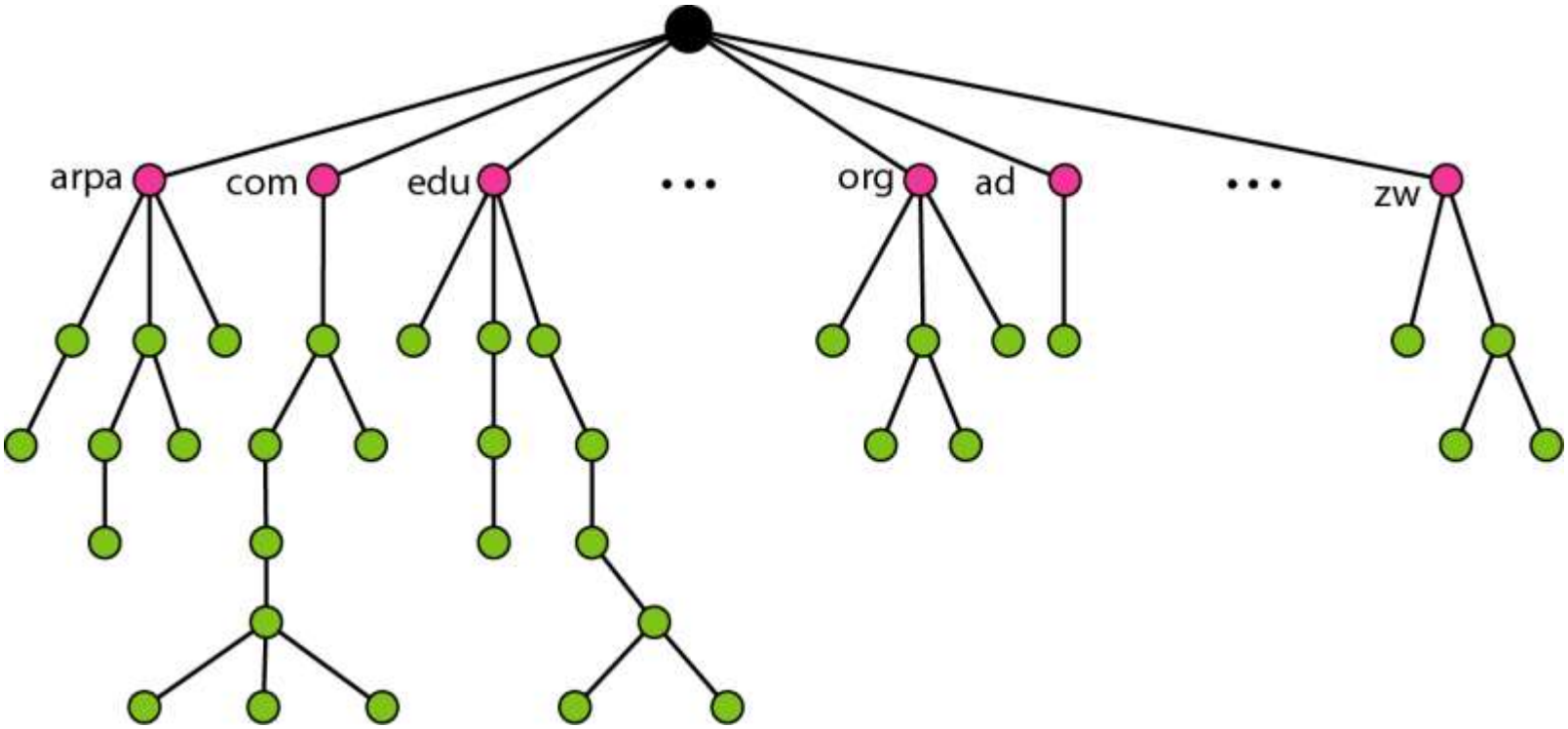
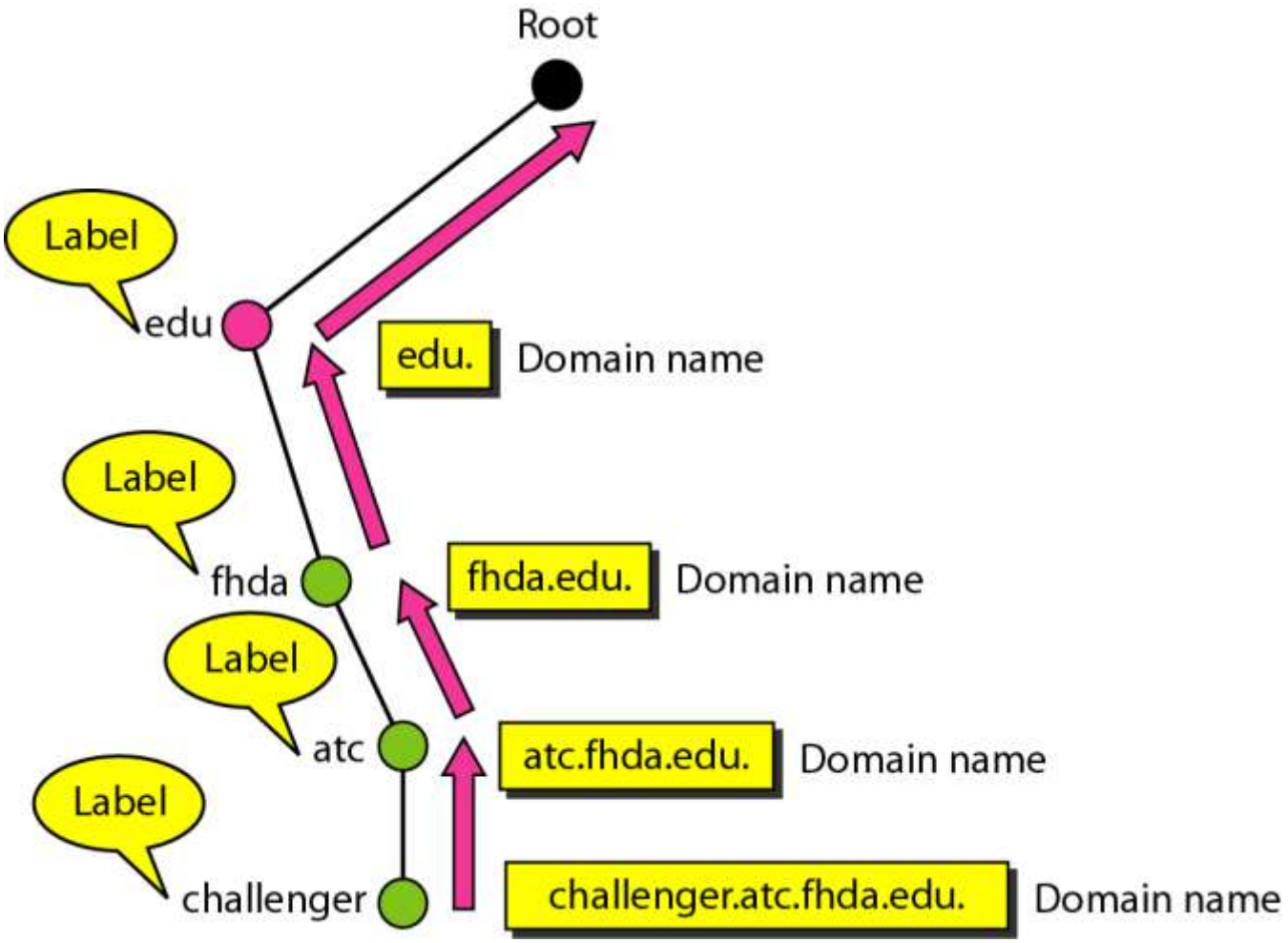


Figure 25.3 *Domain names and labels*



Flat Name Space vs Hierarchical Name Space

- Flat Name Space: A name is assigned to an address. A name in this space is a sequence of characters without structure.

- Disadvantage: It cannot be used in a large system such as the Internet because it must be centrally controlled to avoid ambiguity and duplication.

- Hierarchical Name Space: Each name is made of several parts. The first part can define the **nature of the organization**, the second part can define the **name of an organization**, the third part can define **departments in the organization**, and so on. In this case, the authority to assign and control the name spaces can be decentralized.

Fully Qualified Domain Name

- If a label is terminated by a null string, it is called a FQDN.
- An FQDN is a domain name that contains the full name of a host.
- It contains all labels, from the most specific to the most general, that uniquely define the name of the host.
- For example, the domain name challenger.ate.tbda.edu

Partially Qualified Domain Name

- If a label is not terminated by a null string, it is called a PQDN.
- A PQDN starts from a node, but it does not reach the root.
- It is used when the name to be resolved belongs to the same site as the client.
- Here the resolver can supply the missing part, called the suffix, to create an FQDN. For example, if a user at the *jhda.edu.* site wants to get the IP address of the challenger computer, he or she can define the partial name
  challenger The DNS client adds the suffix *atc.jhda.edu.* before passing the address to the DNS server.

Figure 25.4 *FQDN and PQDN*

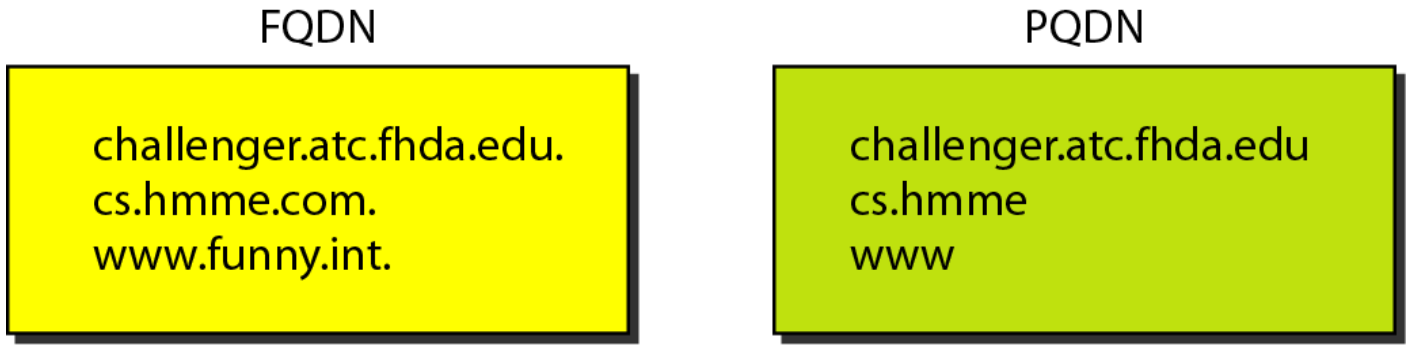
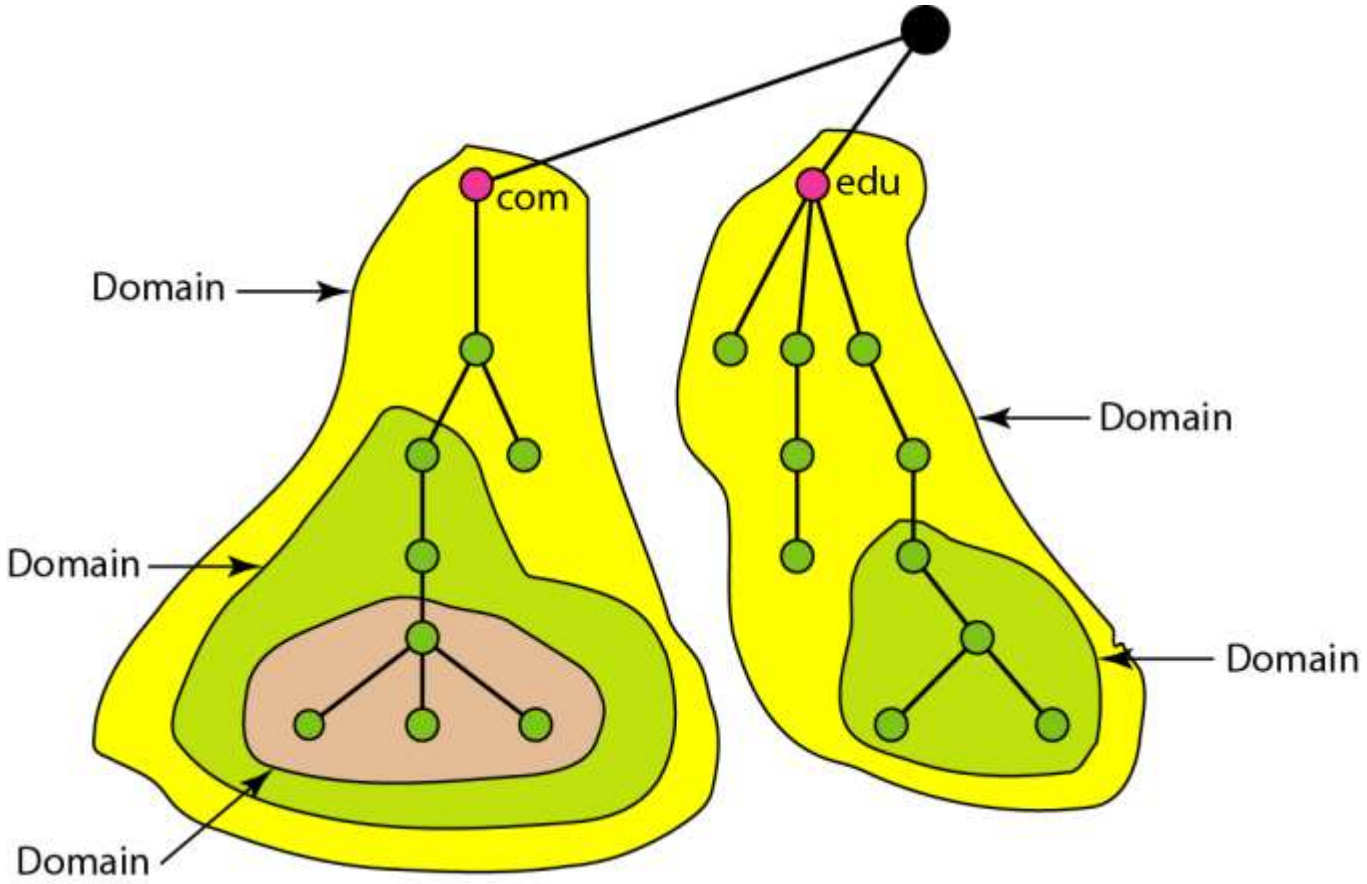


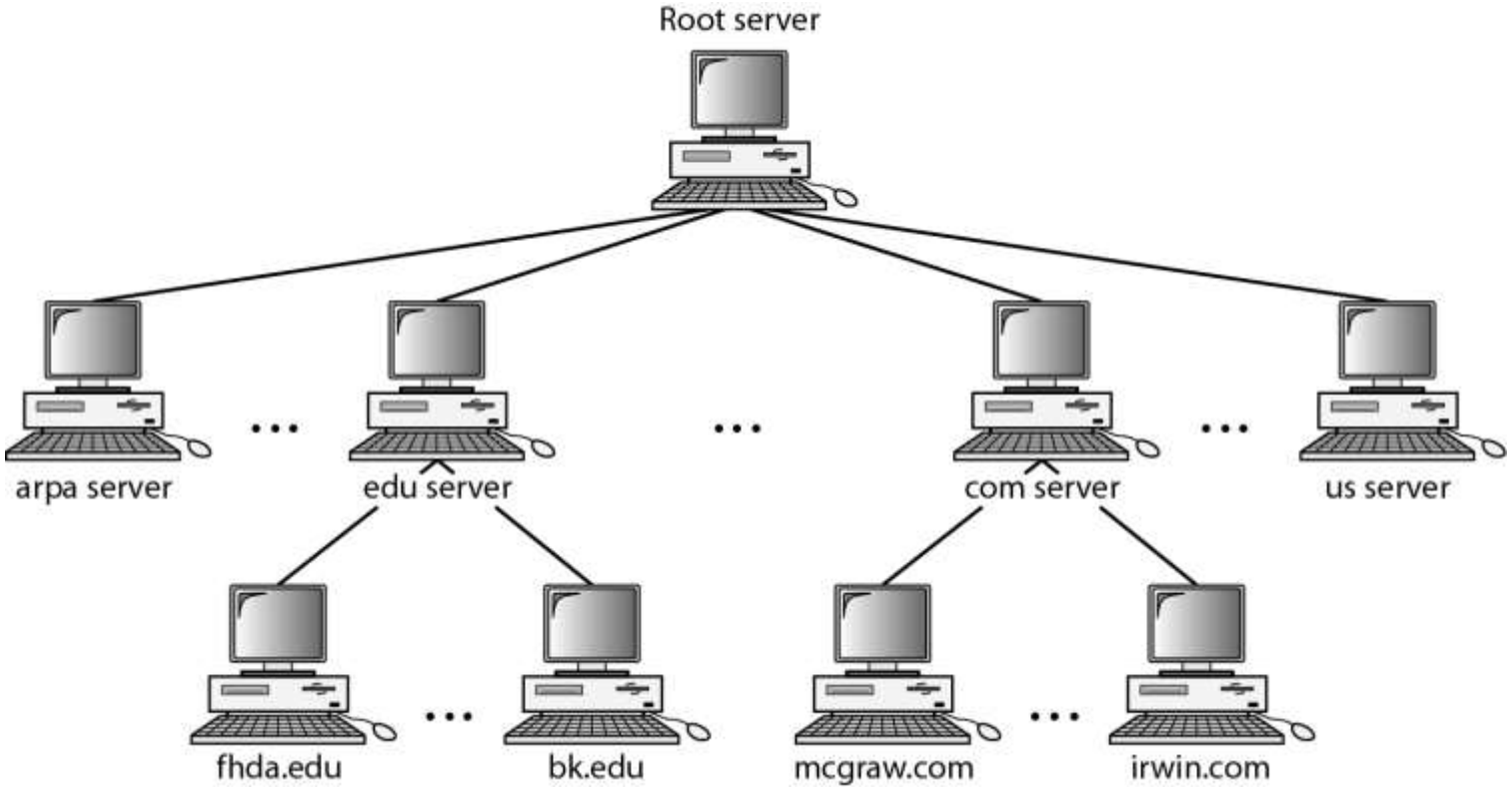
Figure 25.5 *Domains*



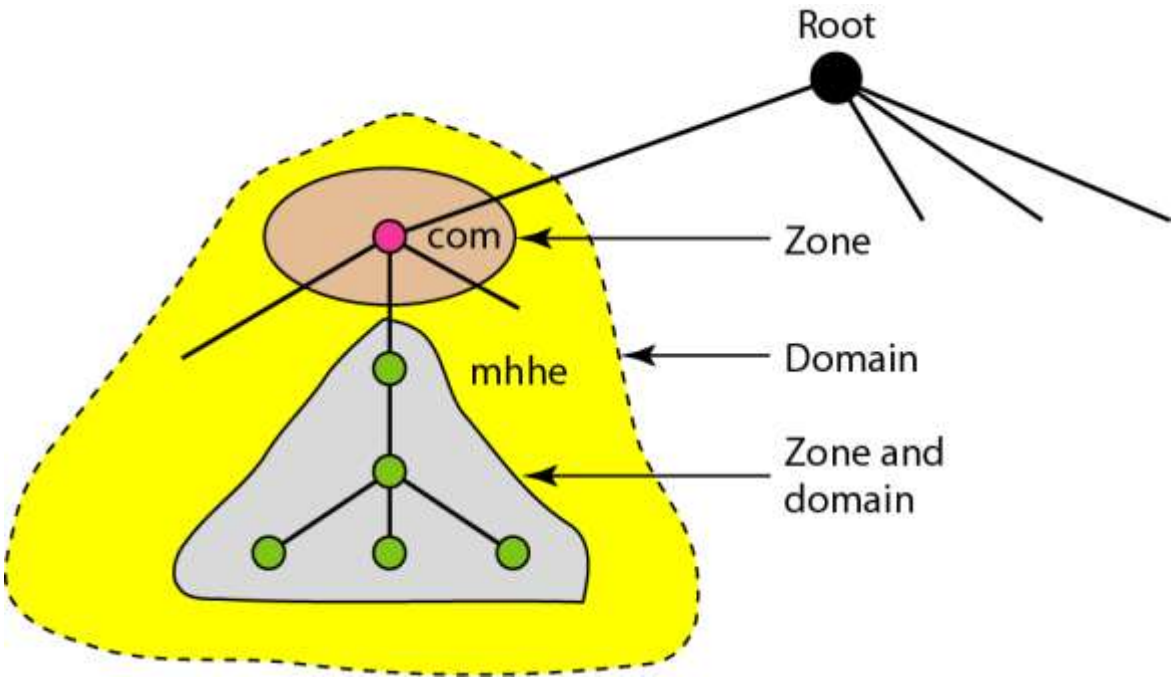
DISTRIBUTION OF NAME SPACE

The information contained in the domain name space must be stored. However, it is very inefficient and also unreliable to have just one computer store such a huge amount of information. In this section, we discuss the distribution of the domain name space.

Hierarchy of name servers



Zones and domains



Note

A primary server loads all information from the disk file; the secondary server loads all information from the primary server.

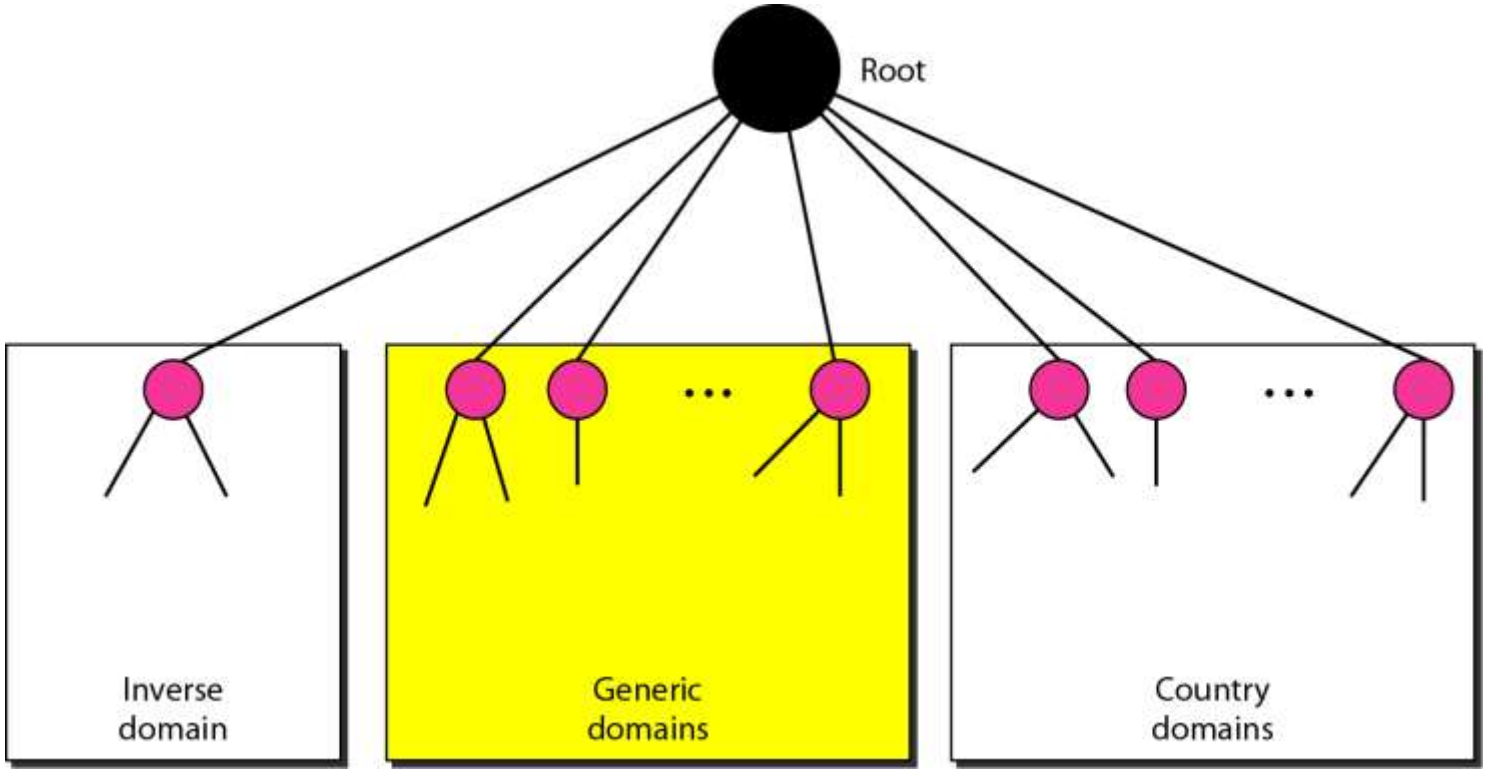
When the secondary downloads information from the primary, it is called zone transfer.

DNS IN THE INTERNET

DNS is a protocol that can be used in different platforms. In the Internet, the domain name space (tree) is divided into three different sections:

generic domains
country domains
inverse domain.

DNS IN THE INTERNET



Generic domains

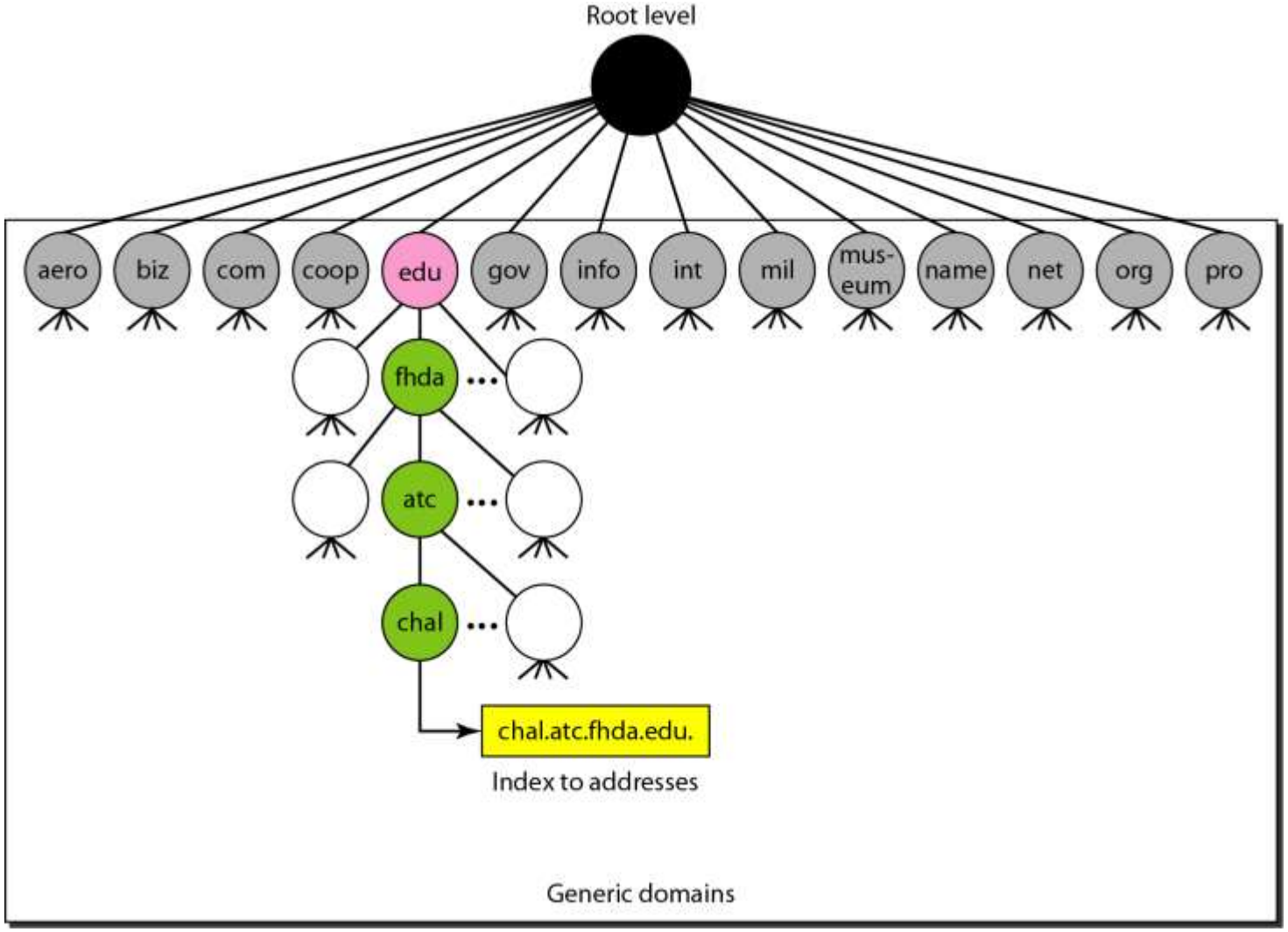


Table 25.1 *Generic domain labels*

Label	*Description*
aero	Airlines and aerospace companies
biz	Businesses or firms (similar to "com")
com	Commercial organizations
coop	Cooperative business organizations
edu	Educational institutions
gov	Government institutions
info	Information service providers
int	International organizations
mil	Military groups
museum	Museums and other nonprofit organizations
name	Personal names (individuals)
net	Network support centers
org	Nonprofit organizations
pro	Professional individual organizations

Figure 25.10 *Country domains*

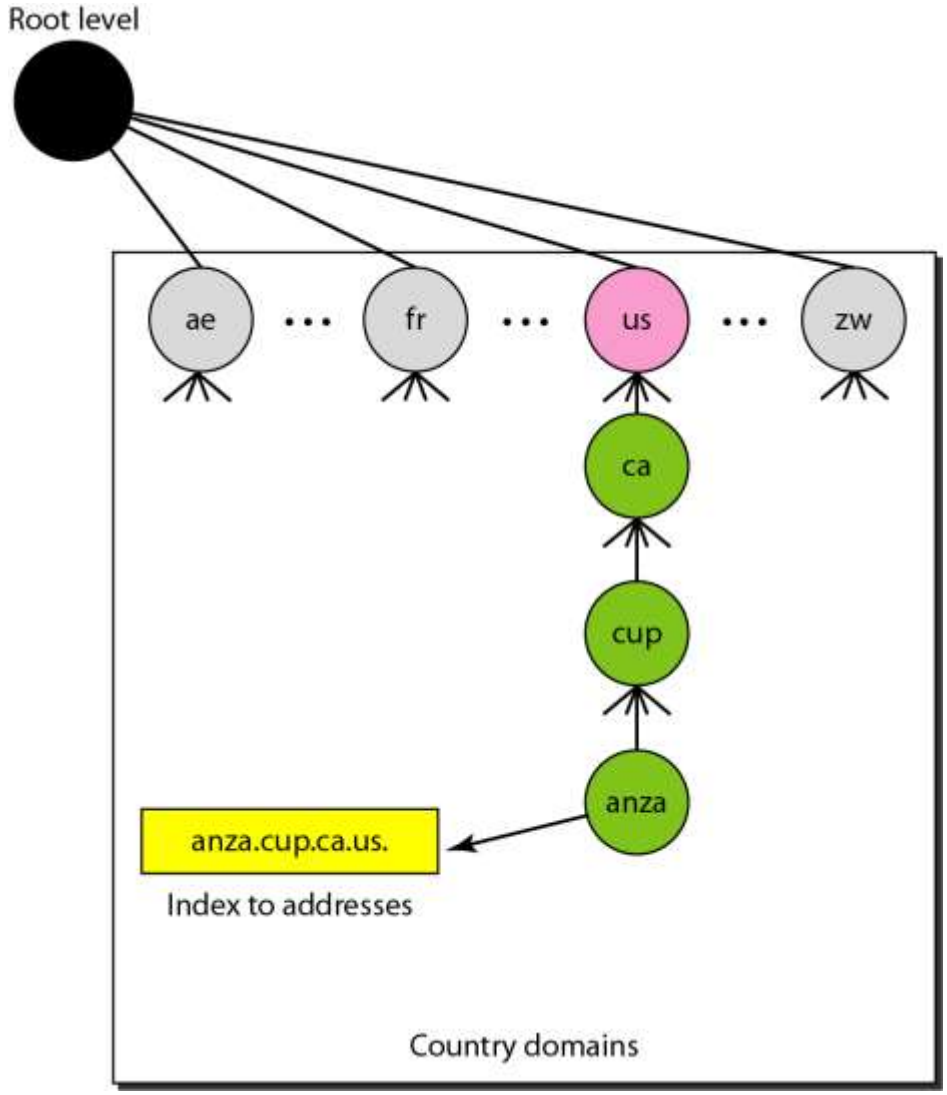
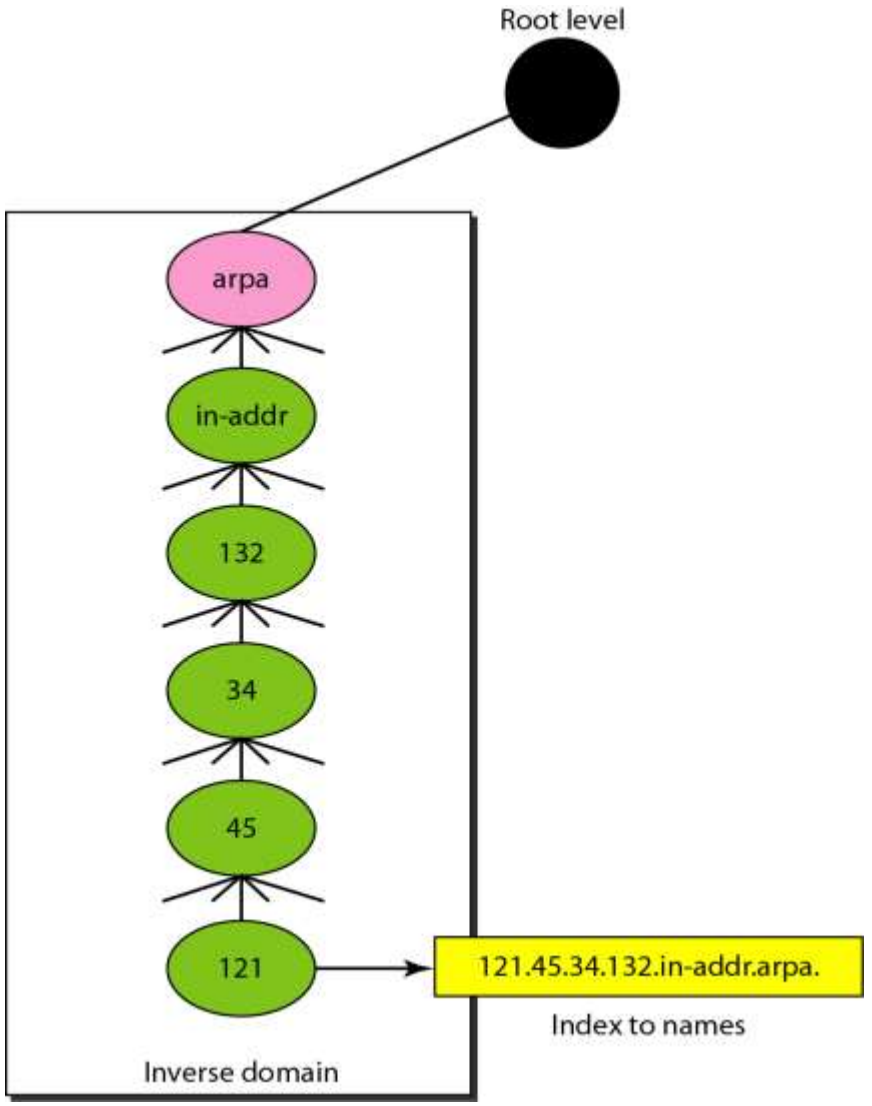


Figure 25.11 *Inverse domain*

- The inverse domain is used to map an address to a name.
- When a server has received a request from a client to do a task.
- Server has a file that contains a list of authorized clients, only the IP address of the client (extracted from the received IP packet) is listed.
- Server asks its resolver to send a query to the DNS server to map an address to a name to determine if the client is on the authorized list.
- This type of query is called an inverse or pointer (PTR) query.



RESOLUTION

Mapping a name to an address or an address to a name is called name-address resolution.

Figure 25.12 *Recursive resolution*

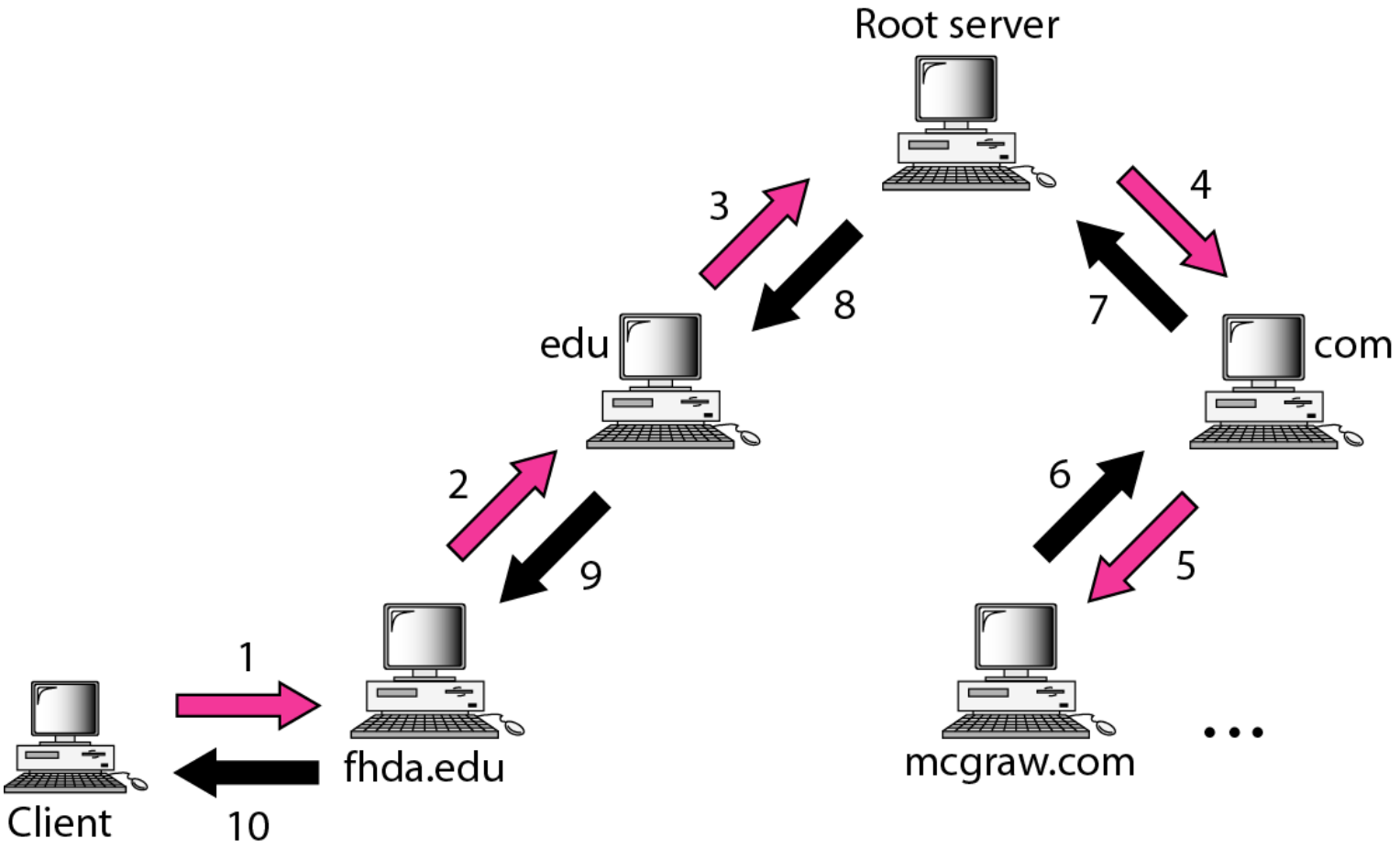


Figure 25.13 *Iterative resolution*