Network Protocol

Transport Services and Protocols

ISLAMIC UNIVERSITY COLLEGE
Computer Technical Engineering Department

Computer Networks Protocols
Lecture No.6: Transport Services and Protocols

Prepared By: Mr. Karar Al-jawaheri@

Transport layer provide ***logical communication*****between application processes**running on different hosts, Transport protocols run in **end systems**

Network layer*:* logical communication **between hosts**

Transport layer*:* logical communication **between processes**

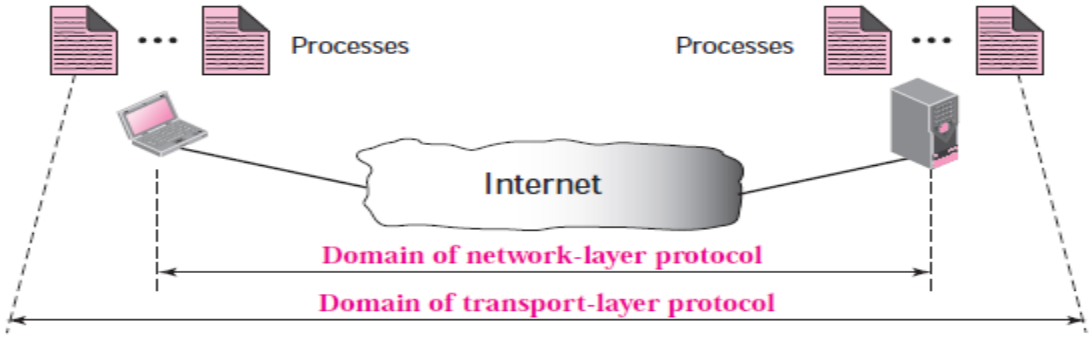
The transport layer deals with:

The quality-of-service issues of reliability, End-to-endflow control, and error correction and recovery, Connection oriented and connectionless.

Services not available in transport layer protocol

–delay guarantees

–bandwidth guarantees



How demultiplexing works

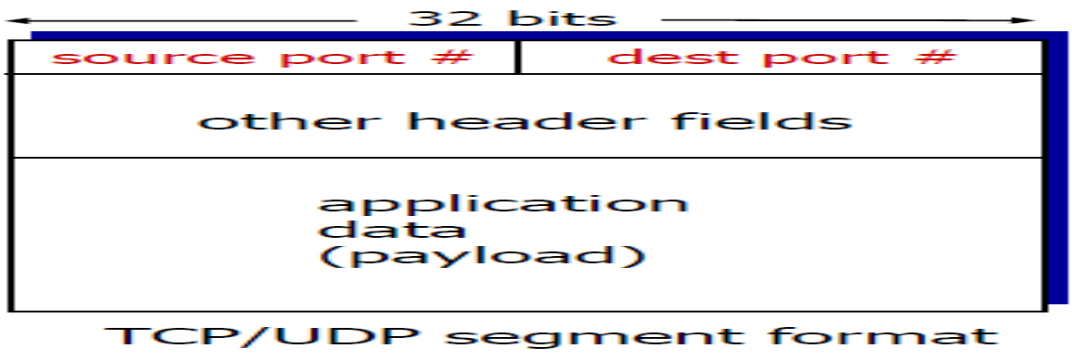
- host receives IP datagrams

  -each datagram has source IP address, destination IP address

  -each datagram carries one transport-layer segment

  -each segment has source, destination port number

- host uses **IP addresses & port numbers to direct segment to appropriate socket**
- **There are two types of demultiplexing: Connection and connectionless**



TCP/UDP segment format

1. Connectionless demultiplexing

when creating datagram to send into **UDP socket**, must specify

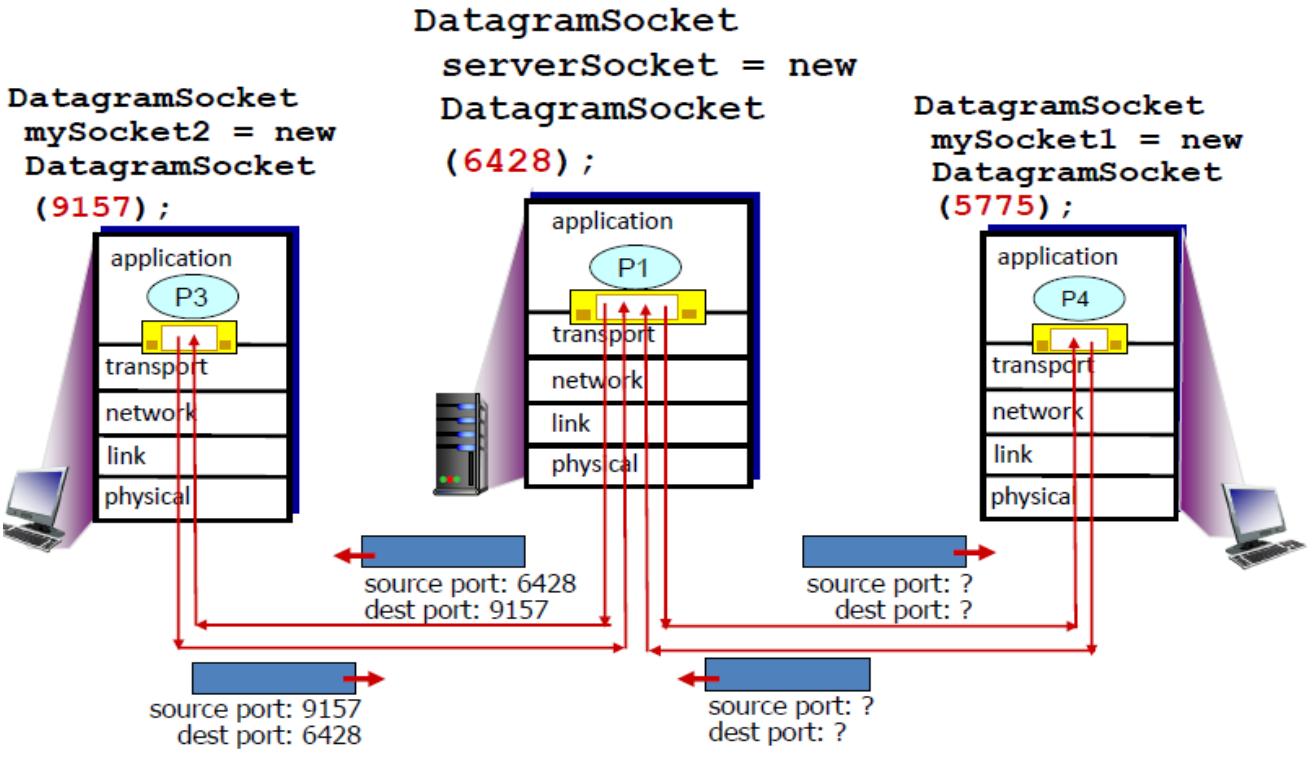
- destination IP address
- destination port #

when host receives UDP segment:

- checks destination port # in segment
- directs UDP segment to socket with that port #

IP datagrams with same dest. port #, but different source IP addresses and/or source port numbers will be directed to same socket at dest

Connectionless demux: example



2. Connection-oriented demux

- **TCP socket** identified by 4-tuple:

source IP address

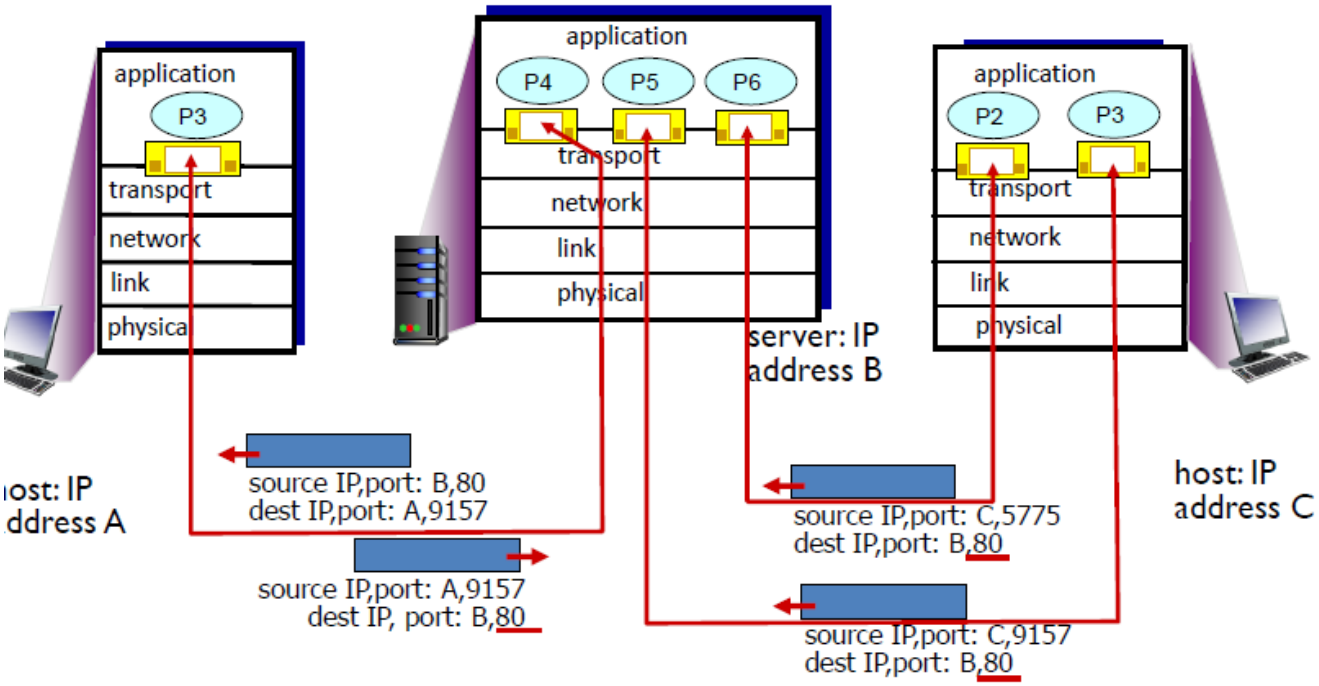
source port number

dest IP address

dest port number

- demux: **receiver** uses all four values to **direct segment to appropriate socket**
- **server host** may support **many** simultaneous TCP sockets(each socket identified by its own 4-tuple)
- web servers have **different sockets** for each connecting client

Connection-oriented demux: example



three segments, all destined to IP address: B,
dest port: 80 are demultiplexed to *different* sockets

Why Flow and Error control

For **reliable** and **efficient** data communication a great deal of coordination is necessary between two machines. Some of these are necessary because of the following :

Constraints:

Both sender and receiver have:

1. limited speed. (receive, send process data).
2. limited memory(storage) .

Requirements:

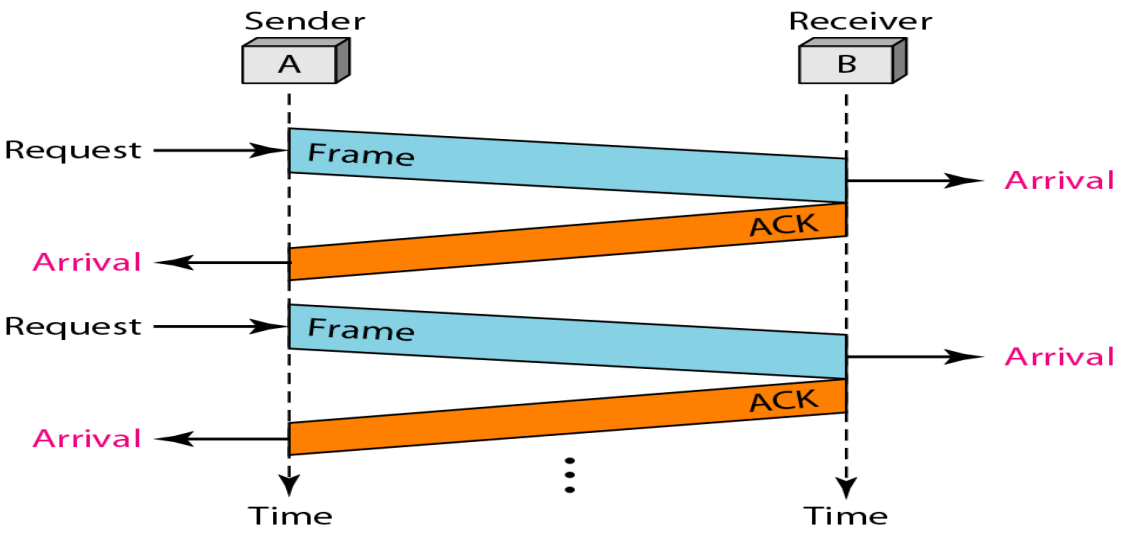
1. A fast sender should not **overwhelm** a slow receiver, which must perform a certain **amount of processing** before passing the data on to the higher-level software.
2. **If error occur** during transmission, it is necessary to create mechanism to correct it.

What is Flow Control

- **Flow Control** is a technique so that transmitter and receiver with different speed characteristics can communicate with each other.
- To control the flow of data, the receiver needs to send some **feedback** to the sender to inform the latter it is overwhelmed with data.
- **ACK** is a packet sent by one host in response to a packet its has received **Time out** .
- **Propagation delay** is define as delay between transmission and receipt.
- **Propagation time** can be used to estimate time out period.

A. Stop-and-Wait

- This is the **simplest** form of flow control.
- After receiving the frame, the receiver indicates its willingness to accept another frame by sending back an **ACK** frame acknowledging the frame just received.
- The sender must **wait** until it receives the ACK frame **before** sending the next data frame.

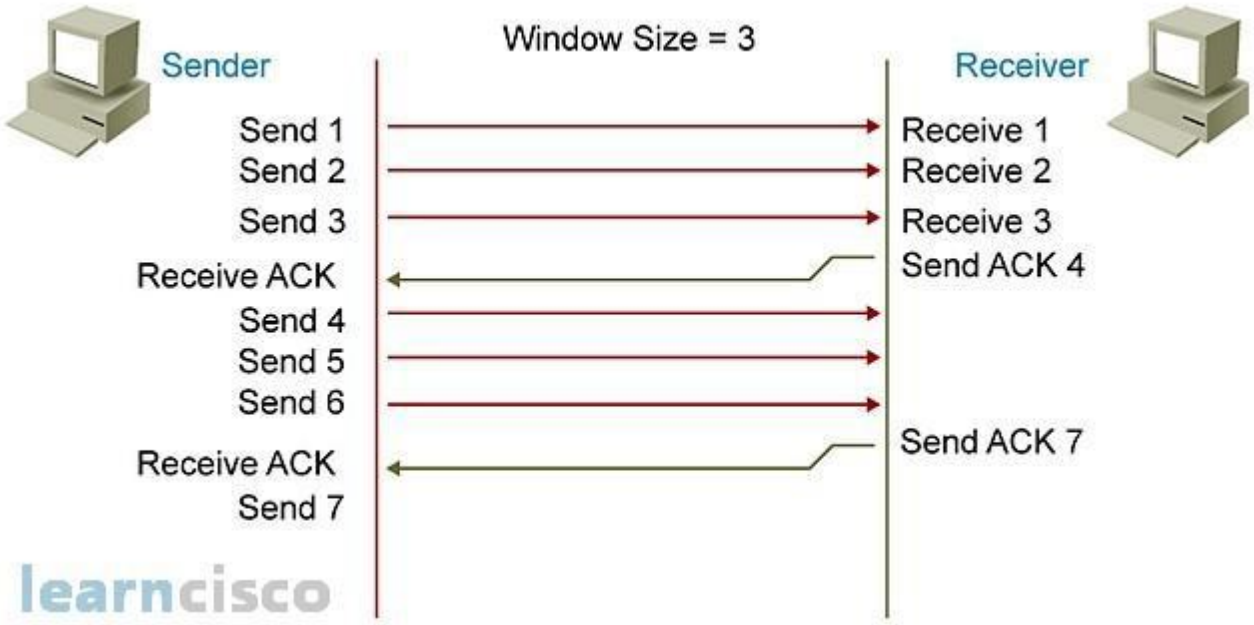


Disadvantage of stop and wait

- At any time there is only **one frame** that sent and waiting to be ACK1
- This is **not** good for transmission **media.**
- To improve efficiency **multiple frames** should be transmit while send is wait to one ACK this called sliding window

B. Sliding Window flow control

- With the use of multiple frames for a **single** message, the stop-and-wait protocol does not perform well.
- Efficiency can be greatly improved by allowing **multiple frames** to be in transit at the **same time.**
- To keep track of the frames, sender station sends **sequentially** numbered frames .
- **Window** meaning number of frames that sender can transmit in same time.
- Size of window can be variable.



Congestion Control: Reasons of congestions:

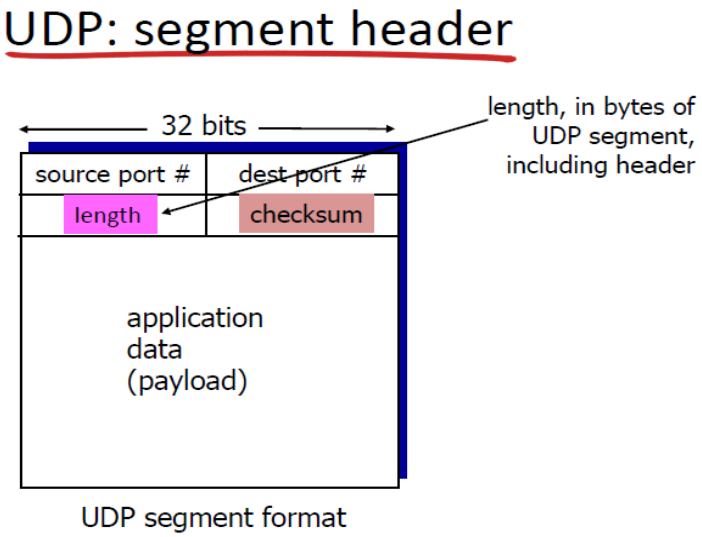
1. **Load** on the network.
2. The number of packets sent is greater than the **capacity** of the network.
3. **Low** Bandwidth.
4. **Slow** Processor.

USER DATAGRAM PROTOCOL (UDP)

- **Connectionless (unreliable transportprotocol)**(best-effort)
- This means that each user datagram sent by UDP is an **independent datagram** and no handshaking between UDP sender, receiver.
- UDP segments may be:(lost, delivered out-of-order to app)
- The user datagrams **are not numbered**.
- Also, there is **no connection establishment and no connection termination**, This means that each user datagram **can travel on a different path**.
- UDP is a very **simple protocol using a minimum of overhead**.
- There is **no flow control** and hence **no window mechanism**.
- Unreliable but **fast**

Q/ why is there a UDP?

- no connection establishment (which can add delay)
- simple: no connection state at sender, receiver
- small header size
- no congestion control



Use of UDP

The following lists some uses of the UDP protocol:

- Routing Information Protocol(**RIP)**
- Domain Name System (**DNS)**
- streaming **multimedia apps** (loss tolerant, rate sensitive)
- **SNMP**

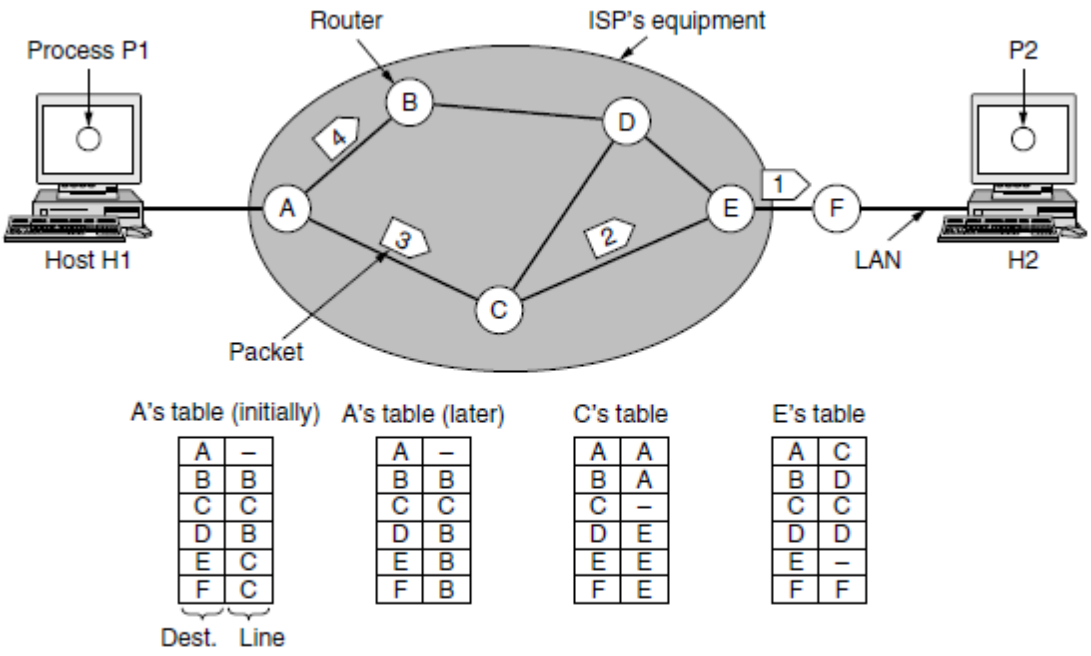
TCP(Transmission Control Protocol)

- reliable, in-order delivery(congestion control,flow control,connection setup)
- TCP, **like** UDP, is a **process-to-process** (program-to-program) protocol. TCP, therefore, like UDP, **uses port numbers**.
- **Unlike** UDP, 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.
- **Reliable delivery**: no packet loss, error, duplication, disorder
- In brief, TCP is called a ***connection-oriented***, ***reliable***transport protocol. It adds connection-oriented and reliability features to the services of IP.
- Most of the user applicationprotocols, such as **Telnet,SMTP,HTTP and FTP**, use TCP

Comparison between UDP and TCP

UDP	TCP
1. **Connectionless** service, UDP datagrams are delivered independently.	**1.** **Connection-oriented**: connection establishment & termination
2. **Unreliable** delivery: packet loss, corruption, duplication, disorder.	**2.** **Reliable** delivery: no loss, error, duplication, disorder.
3. **fast** as compared with TCP	**3.** **slow**: retransmission, flow control
4. No sequencing	**4.** Segment sequencing
5. No acknowledge	**5.** Acknowledge segment
6. Simple request-response communication without internal flow and error control	**6.** Connection overhead
7. RIP, DNS use UDP	**7.** **Telnet,SMTP,HTTP and FTP**, use TCP

Implementation of Connectionless Service



Dest.	Line
A	–
B	B
C	C
D	B
E	C
F	C

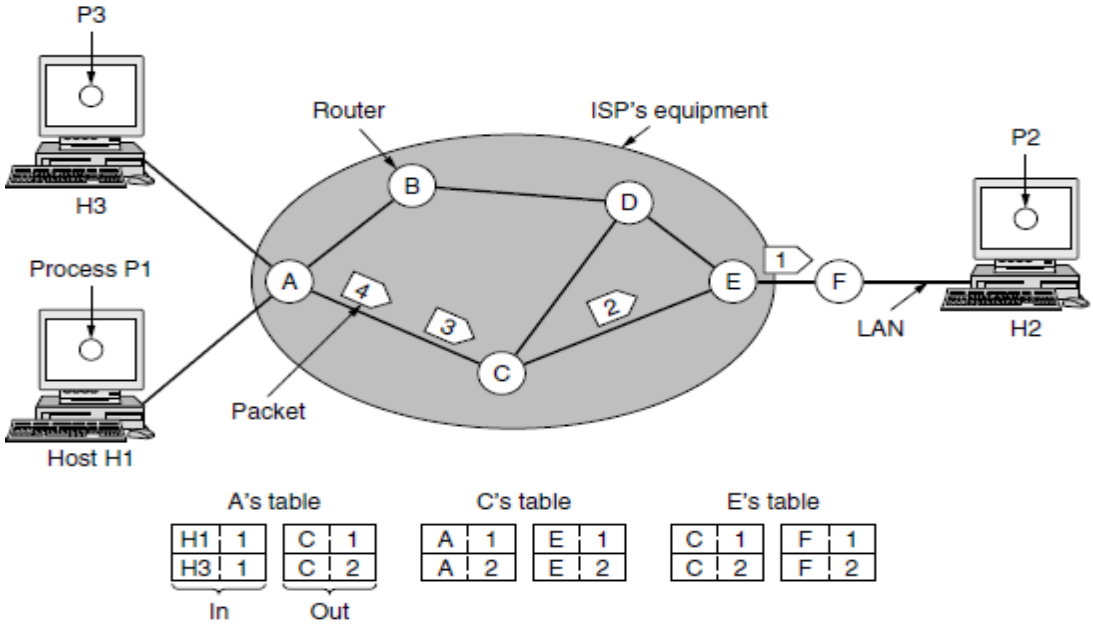
Dest.	Line
A	–
B	B
C	C
D	B
E	B
F	B

Dest.	Line
A	A
B	A
C	–
D	E
E	E
F	E

Dest.	Line
A	C
B	D
C	C
D	D
E	–
F	F

Figure 5-2. Routing within a datagram network.

Implementation of Connection-Oriented Service



In		Out	
H1	1	C	1
H3	1	C	2

A	1	E	1
A	2	E	2

C	1	F	1
C	2	F	2

Figure 5-3. Routing within a virtual-circuit network.