Network Layer

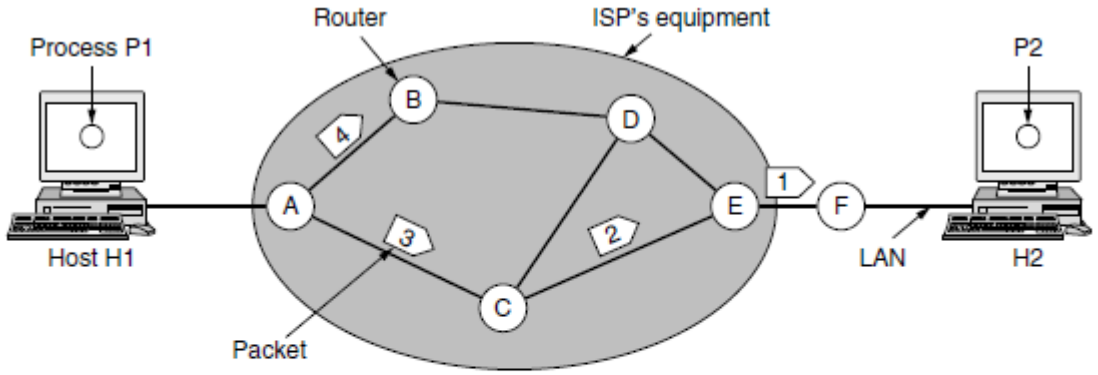
- **network layer** controls the operation of the subnet.
- Main functions: **Routing and Congestion Control**.

- The network layer supports transport layer in providing **connectionless or connection-oriented services**.

- **Connectionless service:** packets/datagrams are injected into the network individually and routed independently of each other. No advance setup is needed.

Network Layer

- Connectionless Service



A's table (initially)

Dest.	Line
A	–
B	B
C	C
D	B
E	C
F	C

A's table (later)

A	–
B	B
C	C
D	B
E	B
F	B

C's table

A	A
B	A
C	–
D	E
E	E
F	E

E's table

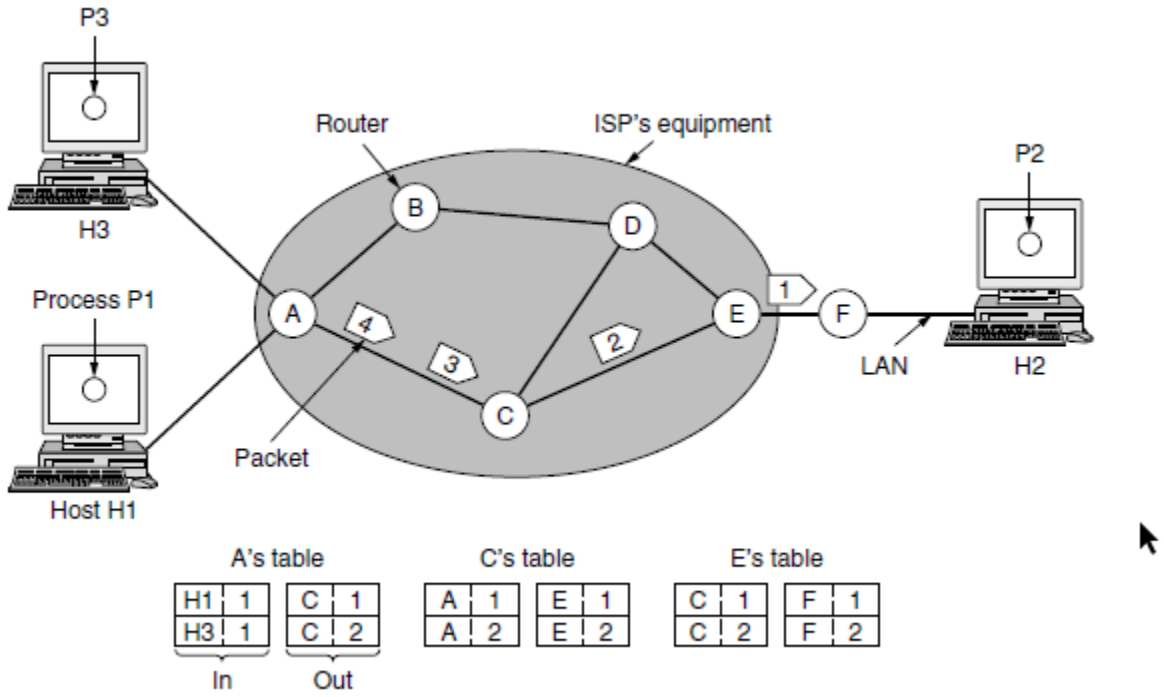
A	C
B	D
C	C
D	D
E	–
F	F

Network Layer

- **connection-oriented service:** A path from the source router all the way to the destination router must be established before any data packets can be sent. This connection is called a **VC** (**virtual circuit**), in analogy with the physical circuits set up by the telephone system, and the network is called a **virtual-circuit network**.

Network Layer

Connection-oriented service (Routing in a virtual circuit network)



Circuit and Packet Switching Reference

- https://www.youtube.com/watch?v=ulKhM0edtDI

- **Routing:** how packets will be routed (transferred) from source to destination. It can be **static (static routing tables-route is fixed for all packets), dynamic, semi-dynamic.**

- **Dynamic Routing Algorithms:** Distance Vector Routing Algorithm, Link State Routing, etc.

Network Layer

- **Congestion Control**

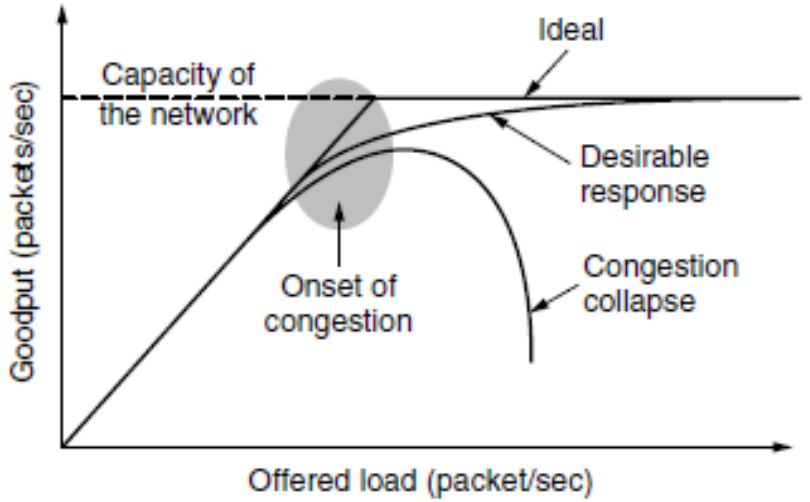
- **Congestion -** If too many packets are present in the subnet at the same time, they will get in one another's way, forming bottlenecks.

- Consequence of congestion – Network Delay, Packet Loss

- Congestion is handled both by Network and Transport Layer

Network Layer

- Goodput vs. Offered Load



With too much traffic, performance drops sharply.

Network Layer

- Approaches to Congestion Control
    - Network Provisioning: Adding spare routers, or adding more links/cables or purchasing more bandwidth/spectrum from market.
    - Traffic-aware routing: Change routes while routing to avoid congested routes
    - Admission Control: New connections are refused
    - Traffic throttling: Inform the source to throttle the traffic
    - Load shedding: when routers are being inundated by packets that they cannot handle, they just throw them away

Network Layer

- Providing the **Quality of Service (QoS) (delay, transmit time, jitter, etc.)** is also a network layer issue.

Application	Bandwidth	Delay	Jitter	Loss
Email	Low	Low	Low	Medium
File sharing	High	Low	Low	Medium
Web access	Medium	Medium	Low	Medium
Remote login	Low	Medium	Medium	Medium
Audio on demand	Low	Low	High	Low
Video on demand	High	Low	High	Low
Telephony	Low	High	High	Low
Videoconferencing	High	High	High	Low

7. Stringency of applications' quality-of-service requirements.

Network Layer

- **Internetworking:** When a packet has to travel from one network to another to get to its destination, many problems can arise. Addressing, Packet size is different, protocols are different. Network layer to overcome all these problems to allow heterogeneous networks to be interconnected.

Network Layer

- Network Layer **does not** guarantee that the packet will reach its intended destination. There are no reliability guarantees.

- The **transport layer is a true end-to-end** layer; it carries data all the way from the source to the destination.

Transport Layer

- **Process-to-Process Communication**
  - Network Layer is responsible for Host to Host Communication
  - Transport Layer on the other hand is responsible for process to process communication
  - Client process initiates connection to Server Process listening for connections.
  - The connection establishment is done with the help of TCP protocol.
  - TCP protocol requires IP address and **port number** to establish connection. **Sockets** have the capability to provide interface for TCP to establish connection.
  - A port number is a 16-bit address used to uniquely identify a Client/Server Process.
  - Using port number to address a process – **Service Point Addressing**
- Basic unit of information is called segment.

Transport Layer- Sockets

- Sockets: A socket can also be referred to as an endpoint for inter process communication(IPC).

- A *socket* is the interface through which a process (application) communicates with the transport layer

- Sockets were first introduced in 2.1BSD and subsequently refined into their current form with 4.2BSD

Transport Layer - Sockets

- Socket Types
  - **Stream Sockets -** sockets use TCP (Transmission Control Protocol) for data transmission. Delivery in a networked environment is guaranteed.
  - **Datagram Sockets -** Delivery in a networked environment is not guaranteed. They're connectionless because you don't need to have an open connection as in Stream Sockets − you build a packet with the destination information and send it out. They use UDP (User Datagram Protocol).

Client Server Model using Sockets

- Client Server Architecture translates to two processes or two applications that communicate with each other to exchange some information.
- **Client Process:**
  - It makes a request for information.
  - Example: Browser Application
- **Server Process:**
  - This process takes request from the clients. After getting a request from the client, this process will perform the required processing, gather the requested information, and send it to the requestor client.
  - Example: Web Server Application

Sockets – Ports

- To identify server host, 32 bit Internet Address (IP Address is used).
- TO identify a particular server process (web server app, ftp server app, etc.), ports are used.
- Port will be defined as an integer number between 1024 and 65535. This is because all port numbers smaller than 1024 are considered *well-known* -- for example, telnet uses port 23, http uses 80, ftp uses 21, and so on.

Sockets - Ports

- Port will be defined as an integer number between 1024 and 65535. This is because all port numbers smaller than 1024 are considered *well-known* -- for example, telnet uses port 23, http uses 80, ftp uses 21, and so on.

Service	Port Number	Service Description
echo	7	UDP/TCP sends back what it receives.
discard	9	UDP/TCP throws away input.
daytime	13	UDP/TCP returns ASCII time.
chargen	19	UDP/TCP returns characters.
ftp	21	TCP file transfer.
telnet	23	TCP remote login.
smtp	25	TCP email.
daytime	37	UDP/TCP returns binary time.
tftp	69	UDP trivial file transfer.
finger	79	TCP info on users.
http	80	TCP World Wide Web.

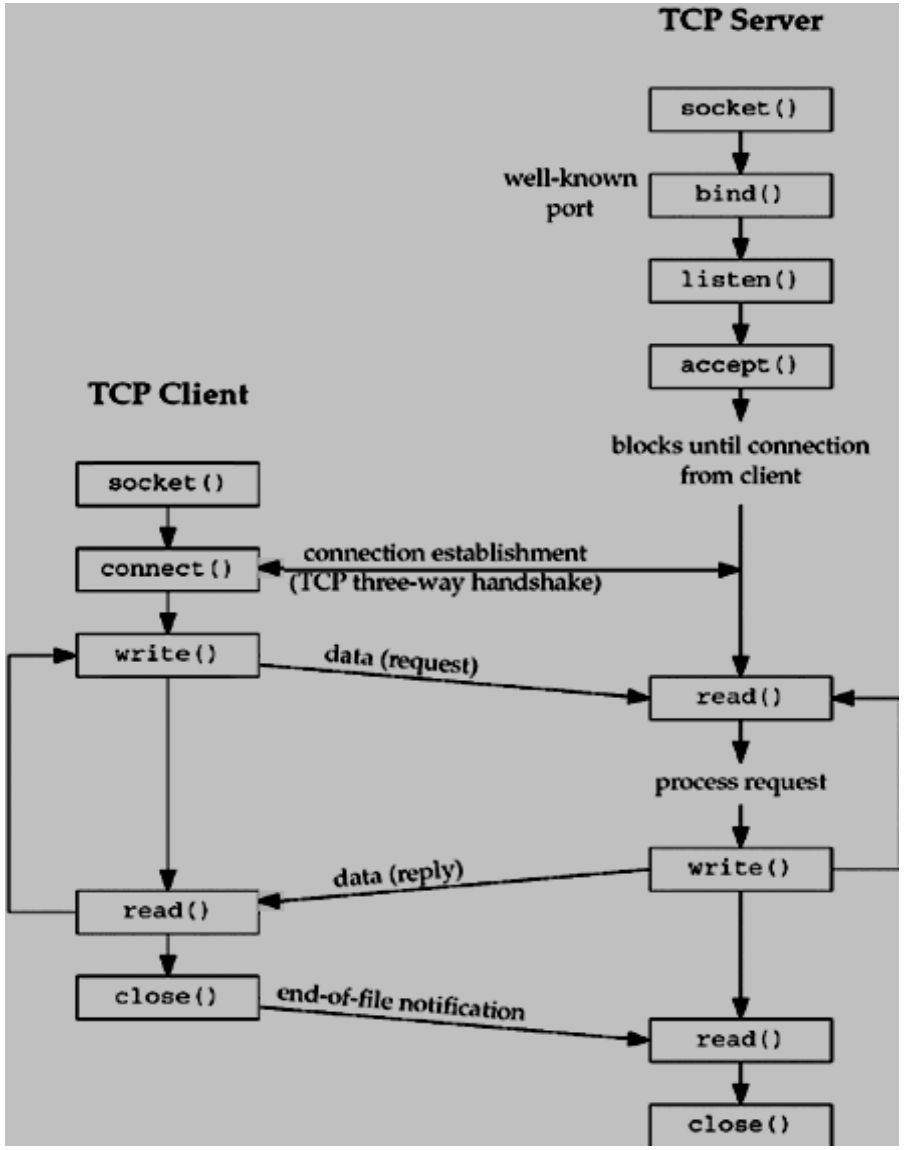
Client Server Model using Sockets

- Types of Server
  - **Iterative**: a server process serves one client at a time
  - **Concurrent**: Server handles multiple client requests (concurrently) at the same time. To design a concurrent server, *fork* a child process to handle each client separately.

Transport Layer - Sockets

Primitive	Meaning
SOCKET	Create a new communication endpoint
BIND	Associate a local address with a socket
LISTEN	Announce willingness to accept connections; give queue size
ACCEPT	Passively establish an incoming connection
CONNECT	Actively attempt to establish a connection
SEND	Send some data over the connection
RECEIVE	Receive some data from the connection
CLOSE	Release the connection

Sockets continued….



Sockets continued…

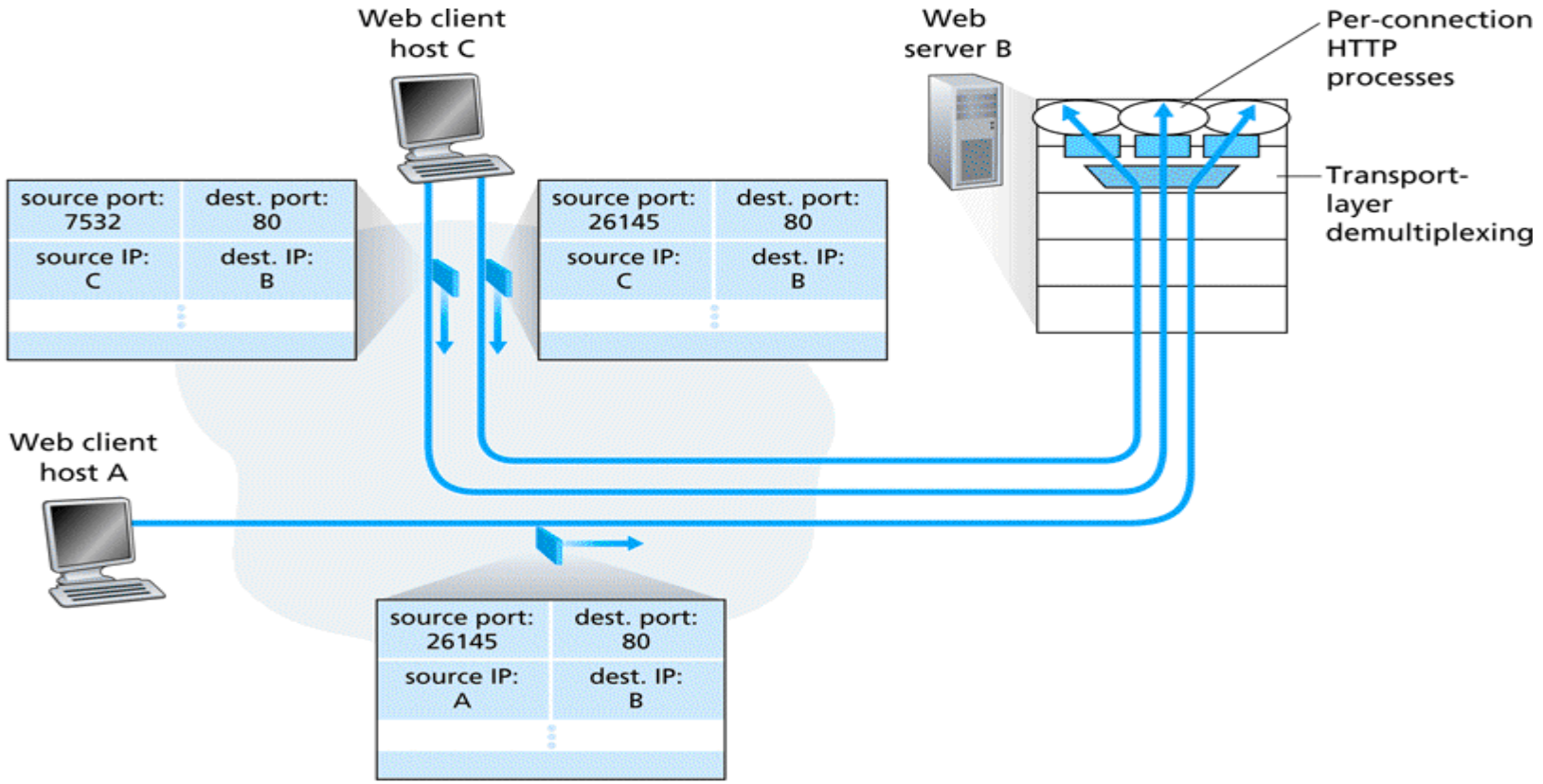
- Steps to implement a client process using socket
  - Create a socket with the **socket()** system call.
  - Connect the socket to the address of the server using the **connect()** system call.
  - Send and receive data. There are a number of ways to do this, but the simplest way is to use the **read()** and **write()** system calls.
- Steps to implement a server process using socket
  - Create a socket with the **socket()** system call.
  - Bind the socket to an address using the **bind()** system call. For a server socket on the Internet, an address consists of a port number on the host machine.
  - Listen for connections with the **listen()** system call.
  - Accept a connection with the **accept()** system call. This call typically blocks the connection until a client connects with the server.
  - Send and receive data using the **read()** and **write()** system calls.

Transport Layer

- **Multiplexing and Demultiplexing:**
  - The transport layer in a receiving machine receives a sequence of segments from its network layer. Delivering segments to the correct socket is called *demultiplexing.*
  - Assembling segments with the necessary information and passing them to the network layer is called *multiplexing.*

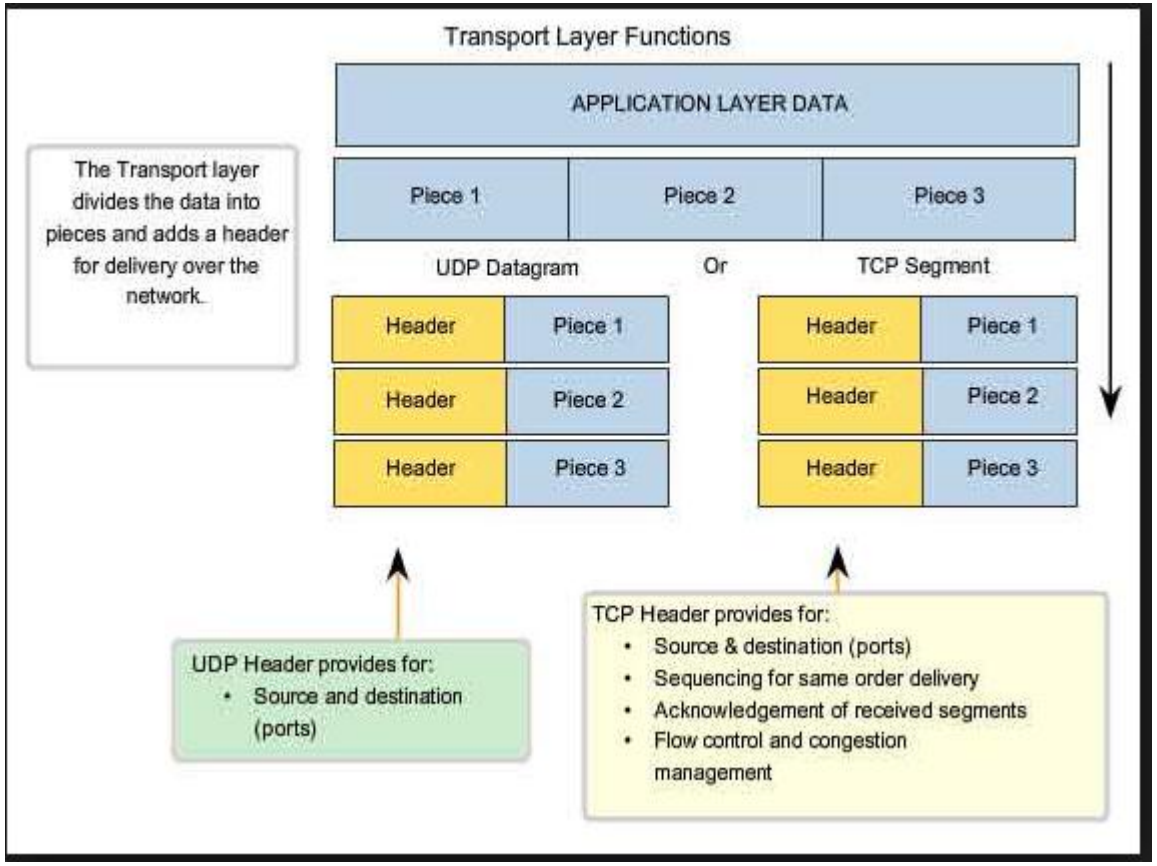
Transport Layer Functions

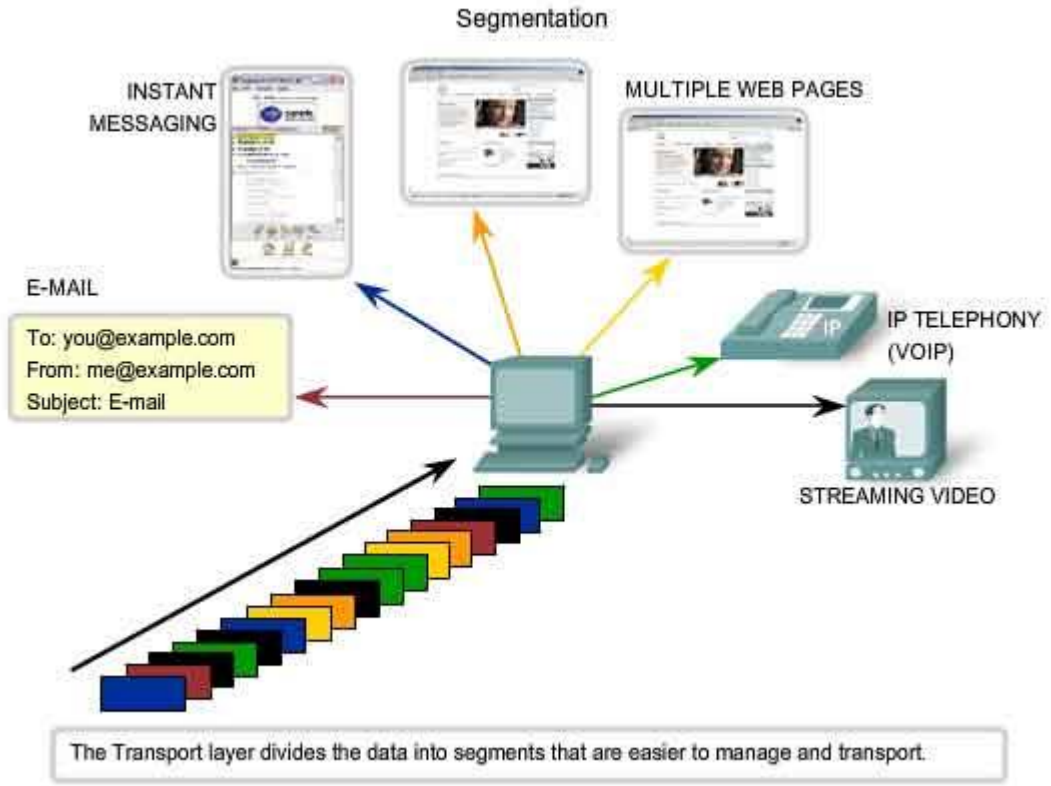
- **Multiplexing and Demultiplexing**



Transport Layer Functions

Segmentation





Transport Layer Functions

- Why segmentation is required?

- The data packet is larger than the maximum transmission unit supported by the network. So, dividing a data packet into smaller units for efficient transmission over the network.

- The network is unreliable and it is desirable to divide the information into smaller segments to maximize the probability that each one of them can be delivered correctly to the destination. Error detection and correction techniques can be applied.
- With the help of sequence numbers, destination can ask for retransmission of a segment from the host.

Transport Layer

- **Segmentation and Reassembly:**
  - A message is divided into segments;
  - each segment contains sequence number, which enables this layer in reassembling the message.
  - Message is reassembled correctly upon arrival at the destination and replaces packets which were lost in transmission.

Transport Layer

- **Congestion Control**
  - Too many sources over a network attempt to send data and the router buffers start overflowing due to which loss of packets occur.
  - **Open loop Congestion Control:** Includes policies before the congestion actually occurs.
    - Retransmission policy: retransmission timers are so designed that it prevents congestion
    - Window policy: Sliding window protocol – prefer selective repeat version instead of Go Back N
    - Discarding policy: A good discarding policy by the routers may prevent congestion and at the same time may not harm the integrity of the transmission.
    - Acknowledgment policy: Acknowledgments are also part of the load in a network. Sending fewer acknowledgments means imposing fewer loads on the network.
    - Admission policy: A router can deny establishing a virtual circuit connection if there is congestion in the network or if there is a possibility of future congestion.
  - **Closed loop Congestion Control:**

Transport Layer

- **Closed loop Congestion Control:** Techniques aim to alleviate congestion once it has occurred.
  - Backpressure: is a node-to-node congestion control that starts with a node and propagates, in the opposite direction of data flow, to the source.

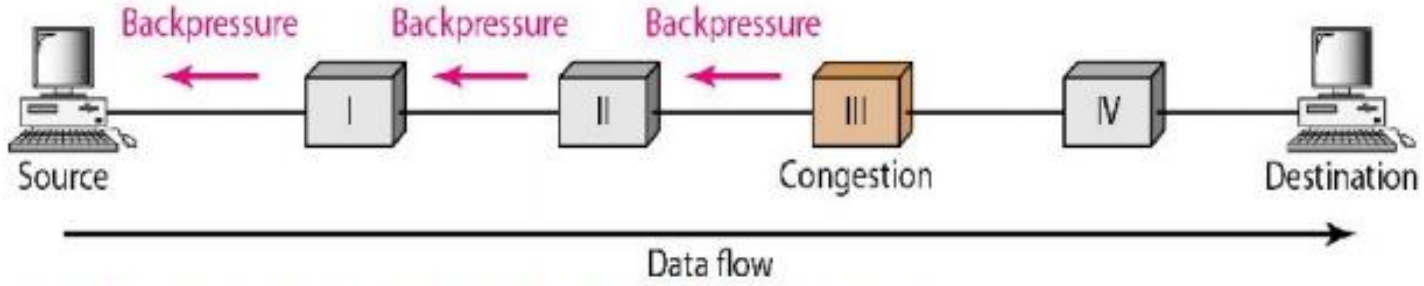


Figure 4.28 Backpressure method for alleviating congestion

Transport Layer

- **Closed loop Congestion Control:** Techniques aim to alleviate congestion once it has occurred.
  - Choke Packet: A choke packet is a packet sent by a node to the source to inform it of congestion.

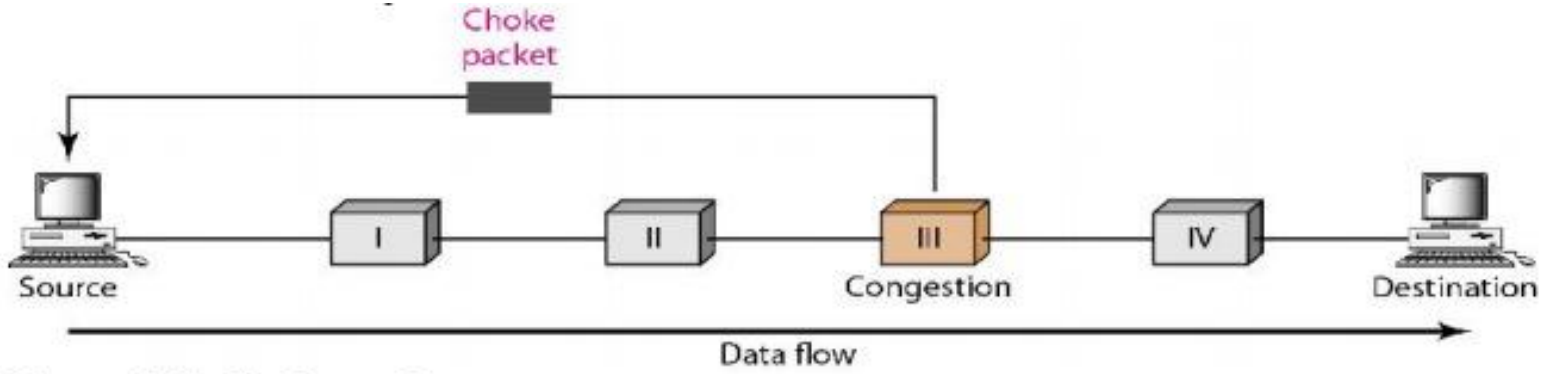


Figure 4.29 Choke packet

Transport Layer

- **Closed loop Congestion Control:** Techniques aim to alleviate congestion once it has occurred.
  - Implicit Signaling: The source guesses that there is congestion in a network.
  - Explicit Signaling: a node explicitly sends a packet to the source or destination to inform about congestion.
    - Forward Signaling
    - Backward Signaling

Transport Layer

- **Error Control:**
  - Use of error detection codes, computing checksums, and use of ACKs/NACKs for informing the sender
- **Flow Control:**
  - End to end flow control. Sliding window protocols used to prevent data loss due to a fast sender and slow receiver

Transport Layer

- **Type of Service:** for creating **end-to-end** Connection between hosts using TCP and UDP.
  - TCP is a secure, connection- orientated protocol which uses a handshake protocol to establish a robust connection between two end- hosts.
  - TCP ensures reliable delivery of messages and is used in various applications.
  - UDP, on the other hand, is a stateless and unreliable protocol which ensures best-effort delivery.

- **Connection Establishment and Release**

Session Layer

- Allows users on different machines to establish **sessions** between them. Sessions offer various services, including
  - **dialog control** (keeping track of whose turn it is to transmit),
  - **token management** (preventing two parties from attempting the same critical operation simultaneously), and
  - **Synchronization** (checkpointing long transmissions to allow them to pick up from where they left off in the event of a crash and subsequent recovery).