Computer Networks

Unit-5: Transport Layer

S.NO.	Course Outcome	Intended Learning Outcomes (ILO)	Knowledge Level	No. of Hours
1	CO 5	Describe Transport Layer – Services, Primitives and Sockets	K1	2
2		Discuss Elements of Transport Protocols	K2	2
3		Discuss The Internet Transport Protocols: TCP Segment Header and Primitives	K2	2
4		Discuss The Internet Transport Protocols: UDP, RPC, RTP, and RTCP Segment Headers and Primitives	K2	1
5		Discuss Congestion Control in TCP	K2	1
6		Outline Quality of Service	K1	1
7		Discuss QoS Improving Techniques: Leaky Bucket and Token Bucket Algorithms	K2	1

Introduction

- The transport layer builds on the network layer to provide data transport from a process on a source machine to a process on a destination machine with a desired level of reliability that is independent of the physical networks currently in use.
- In this chapter, we will study the transport layer in detail, including
  - Services
  - How to tackle issues of reliability, connections
  - Congestion control
  - Protocols such as TCP and UDP.

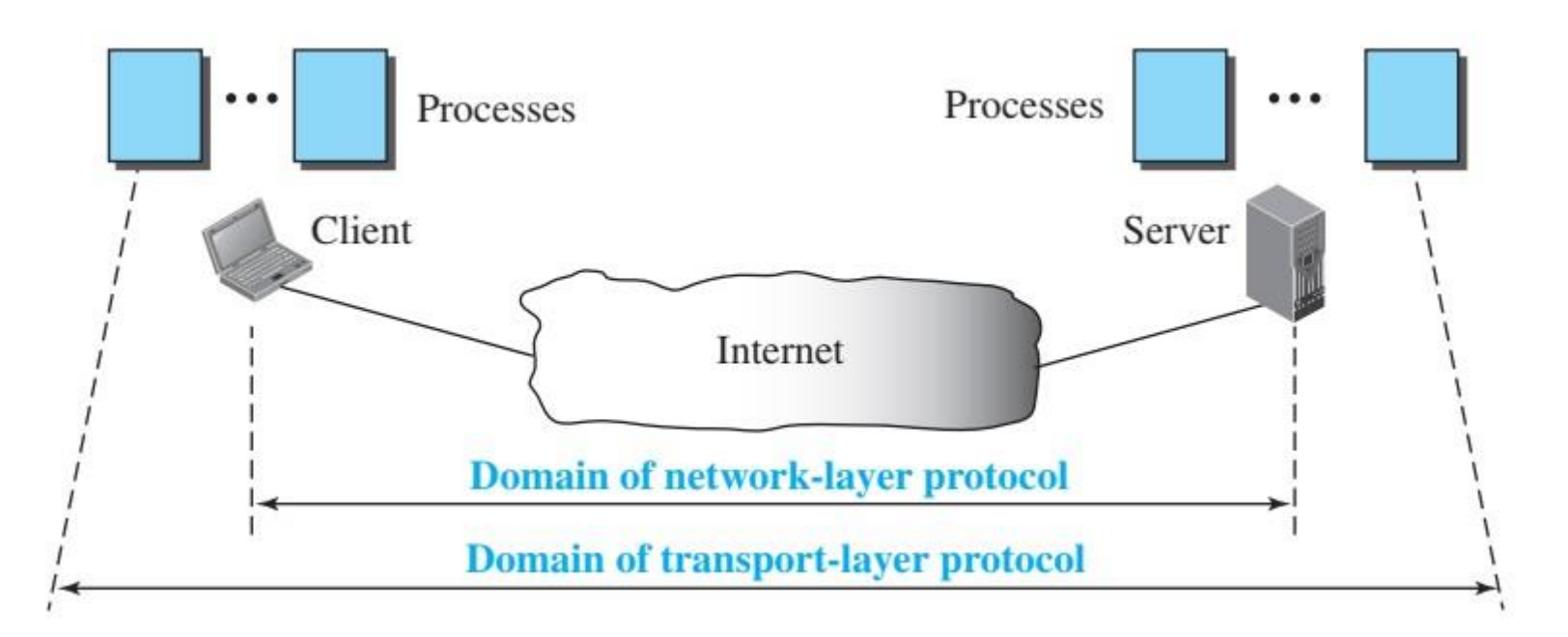
Services Provided to Upper Layer

- The goal of the transport layer is to provide efficient, reliable, and cost-effective data transmission service to its users, normally processes in the application layer.
- Just as there are two types of network service, connection-oriented and connectionless, there are also two types of transport service.
  - Connection-oriented Transport Service (by TCP)
  - Connection-less Transport Service (by UDP)

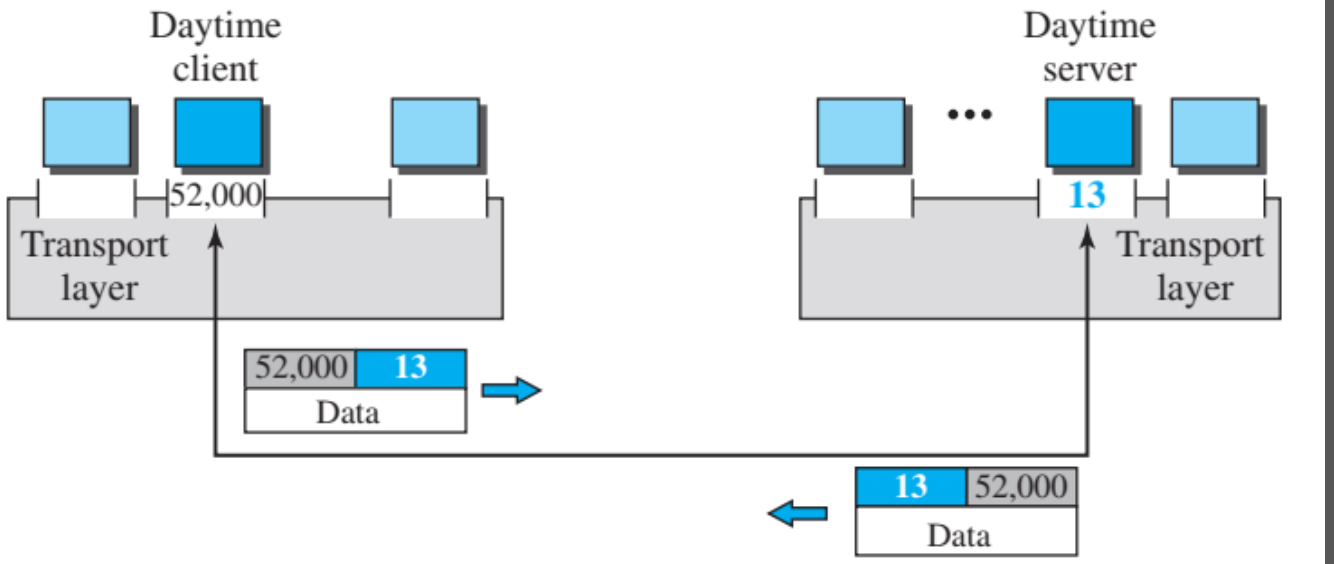
Services Provided to Upper Layer

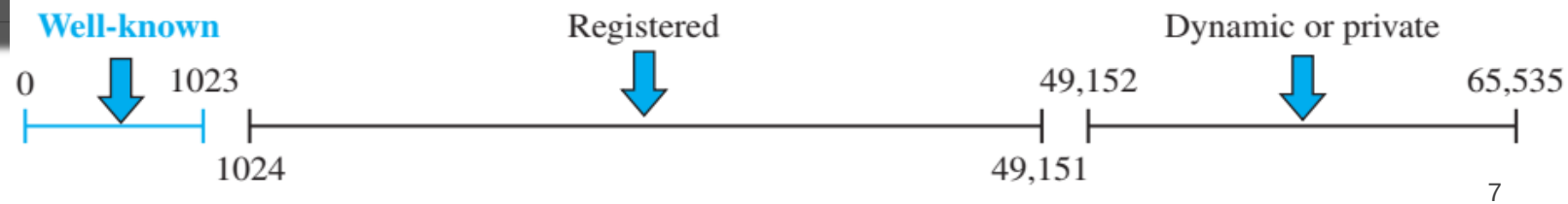
- Functionalities
  - Process-to-Process Communication
  - Addressing: Port Numbers
  - Encapsulation and Decapsulation
  - Multiplexing and Demultiplexing
  - Flow Control
  - Error Control

Process-to-Process Communication

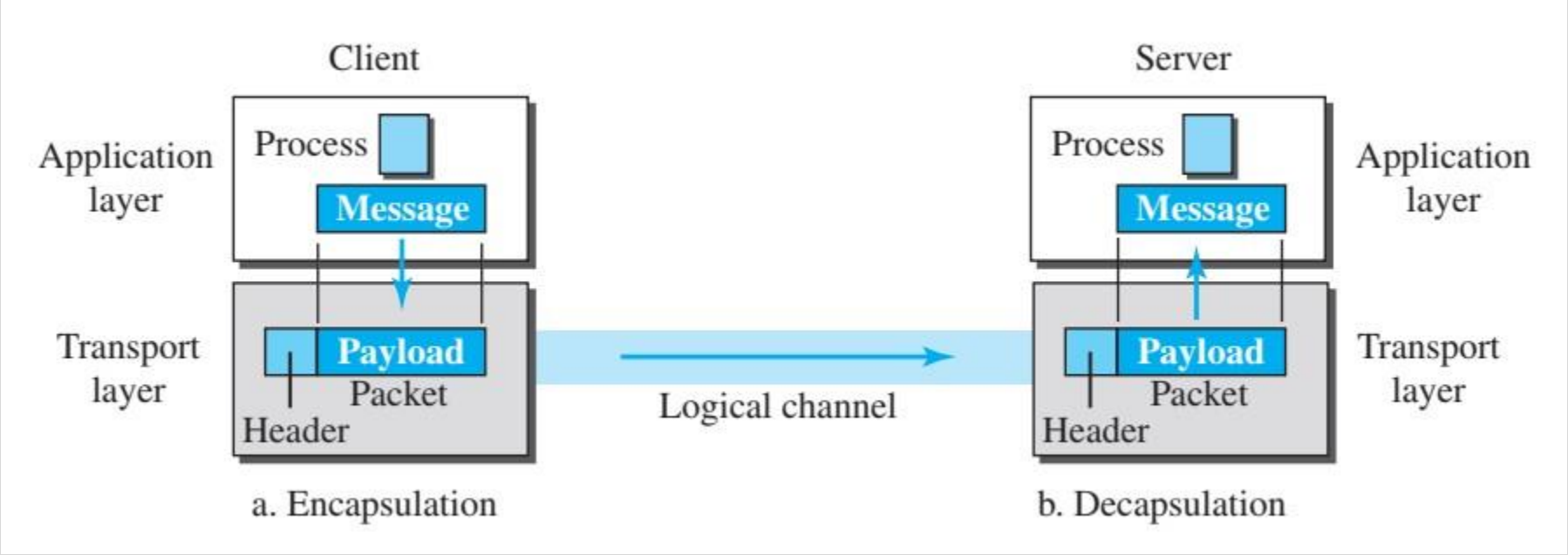


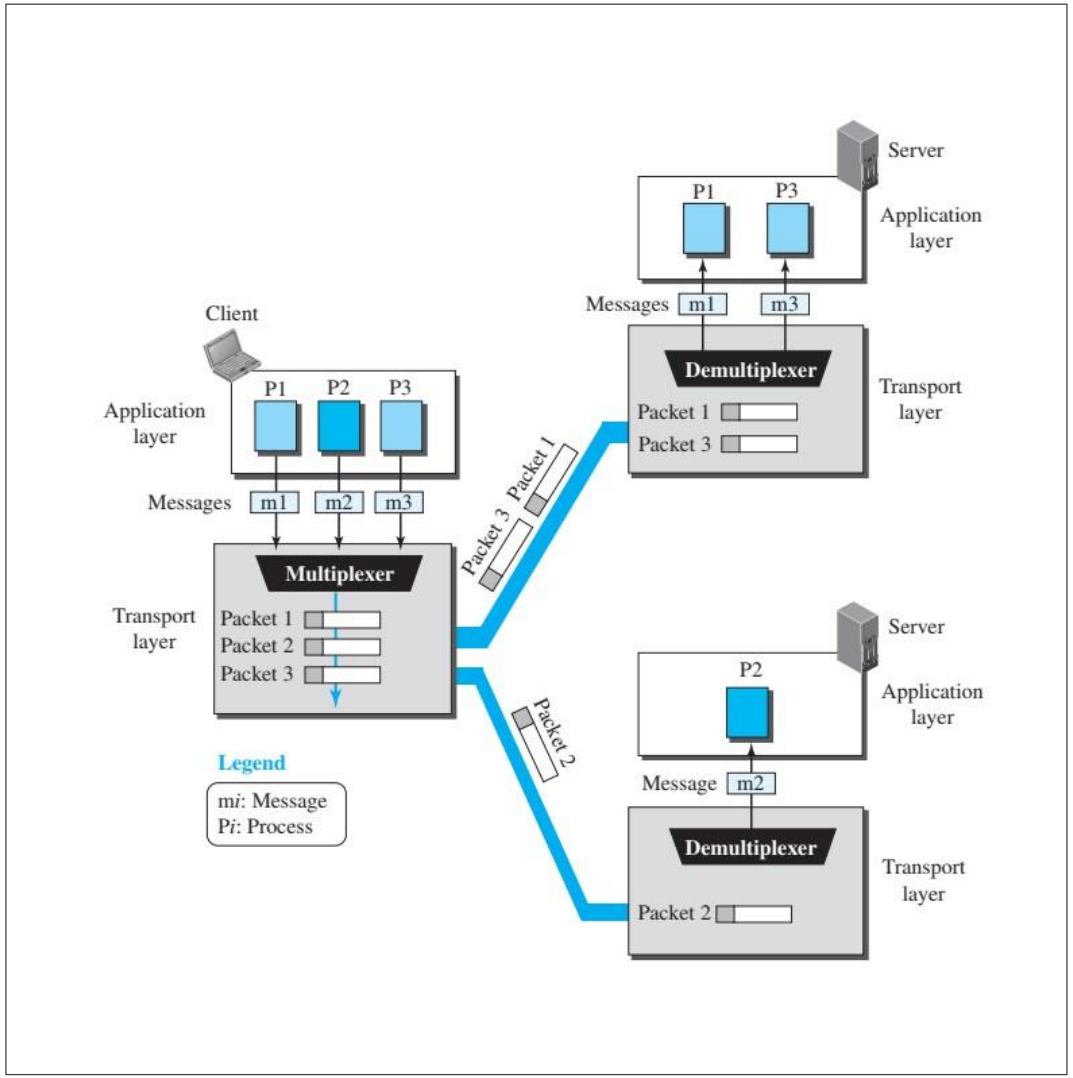
ADDRESSING: PORT NUMBERS





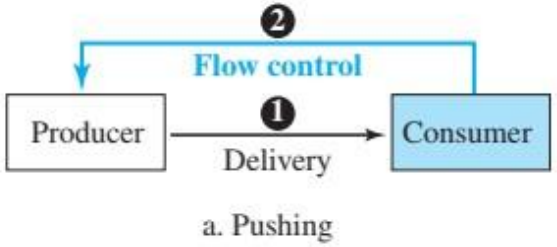
Encapsulation and Decapsulation



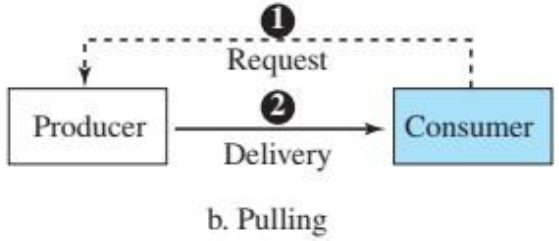


Multiplexing and Demultiplexing

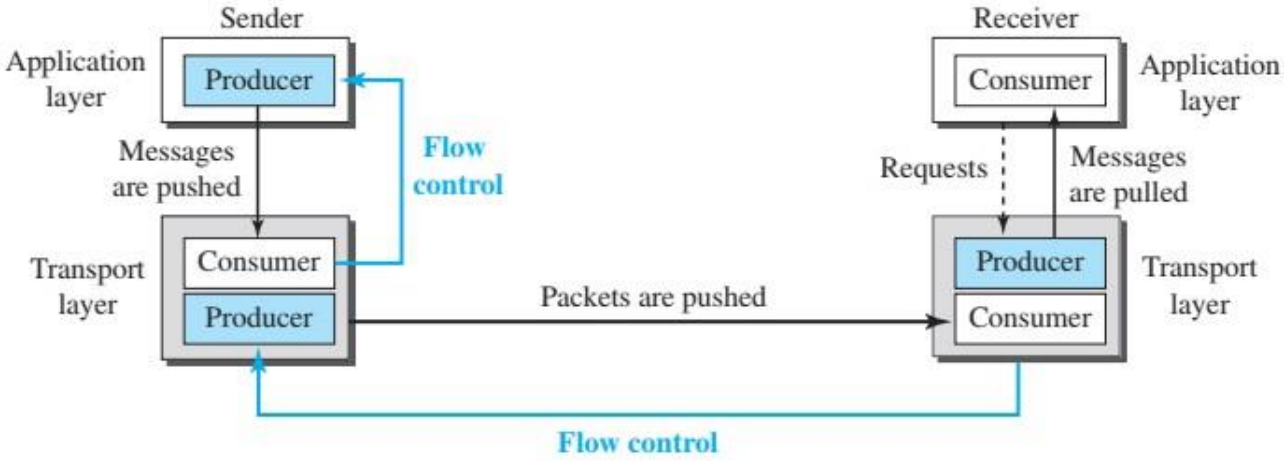
- The packets 1 and 3 use the same logical channel to reach the transport layer of the first server. When they arrive at the server, the transport layer does the job of a demultiplexer and distributes the messages to two different processes.



a. Pushing



b. Pulling



FLOW CONTROL

Error Control

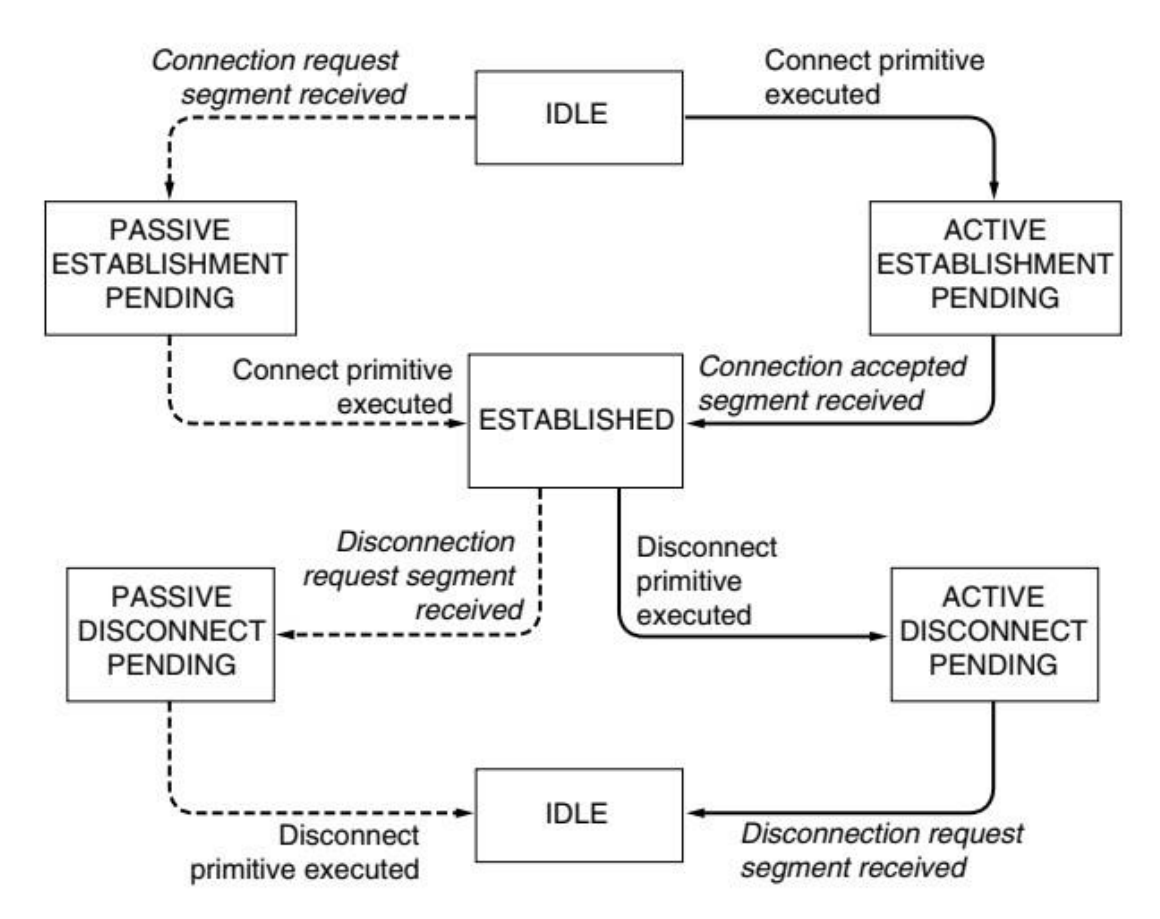
- In the Internet, since the underlying network layer (IP) is unreliable, we need to make the transport layer reliable if the application requires reliability. Reliability can be achieved to add error control services to the transport layer. Error control at the transport layer is responsible for
  - Detecting and discarding corrupted packets.
  - Keeping track of lost and discarded packets and resending them.
  - Recognizing duplicate packets and discarding them.
  - Buffering out-of-order packets until the missing packets arrive.

Primitive	Packet sent	Meaning
LISTEN	(none)	Block until some process tries to connect
CONNECT	CONNECTION REQ.	Actively attempt to establish a connection
SEND	DATA	Send information
RECEIVE	(none)	Block until a DATA packet arrives
DISCONNECT	DISCONNECTION REQ.	Request a release of the connection

Transport Service Primitives

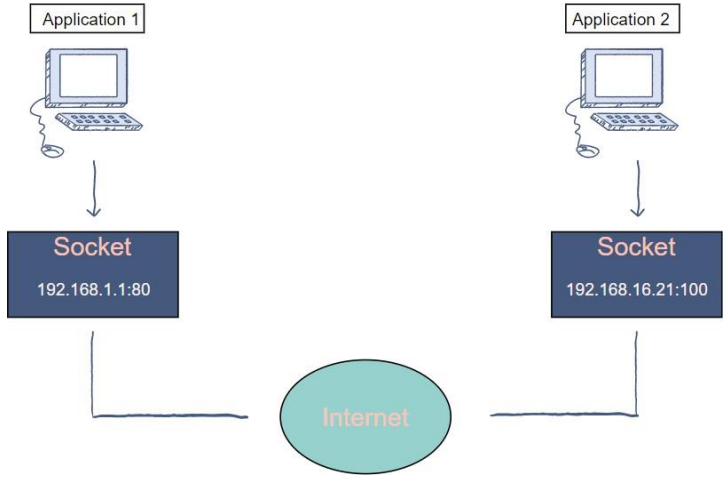
Note: For lack of a better term, we will use the term segment for messages sent from transport entity to transport entity.

Sockets


Connection request segment received
IDLE
Connect primitive executed
PASSIVE ESTABLISHMENT PENDING
ACTIVE ESTABLISHMENT PENDING
Connect primitive executed
ESTABLISHED
Connection accepted segment received
Disconnection request segment received
Disconnect primitive executed
PASSIVE DISCONNECT PENDING
ACTIVE DISCONNECT PENDING
IDLE
Disconnect primitive executed
Disconnection request segment received

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



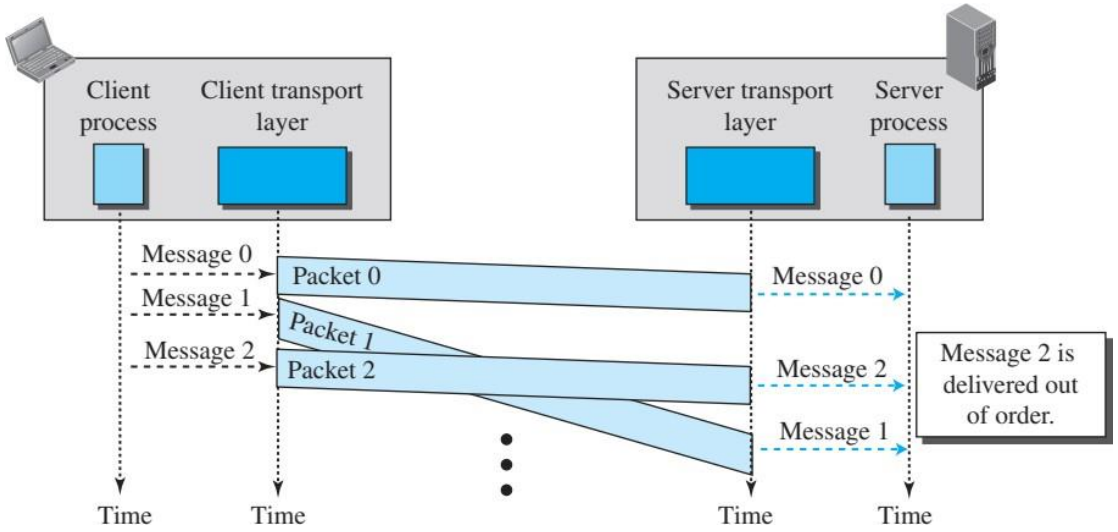
Socket Address

The combination of an IP address and a port number is called a socket address.

The client socket address defines the client process uniquely just as the server socket address defines the server process uniquely.

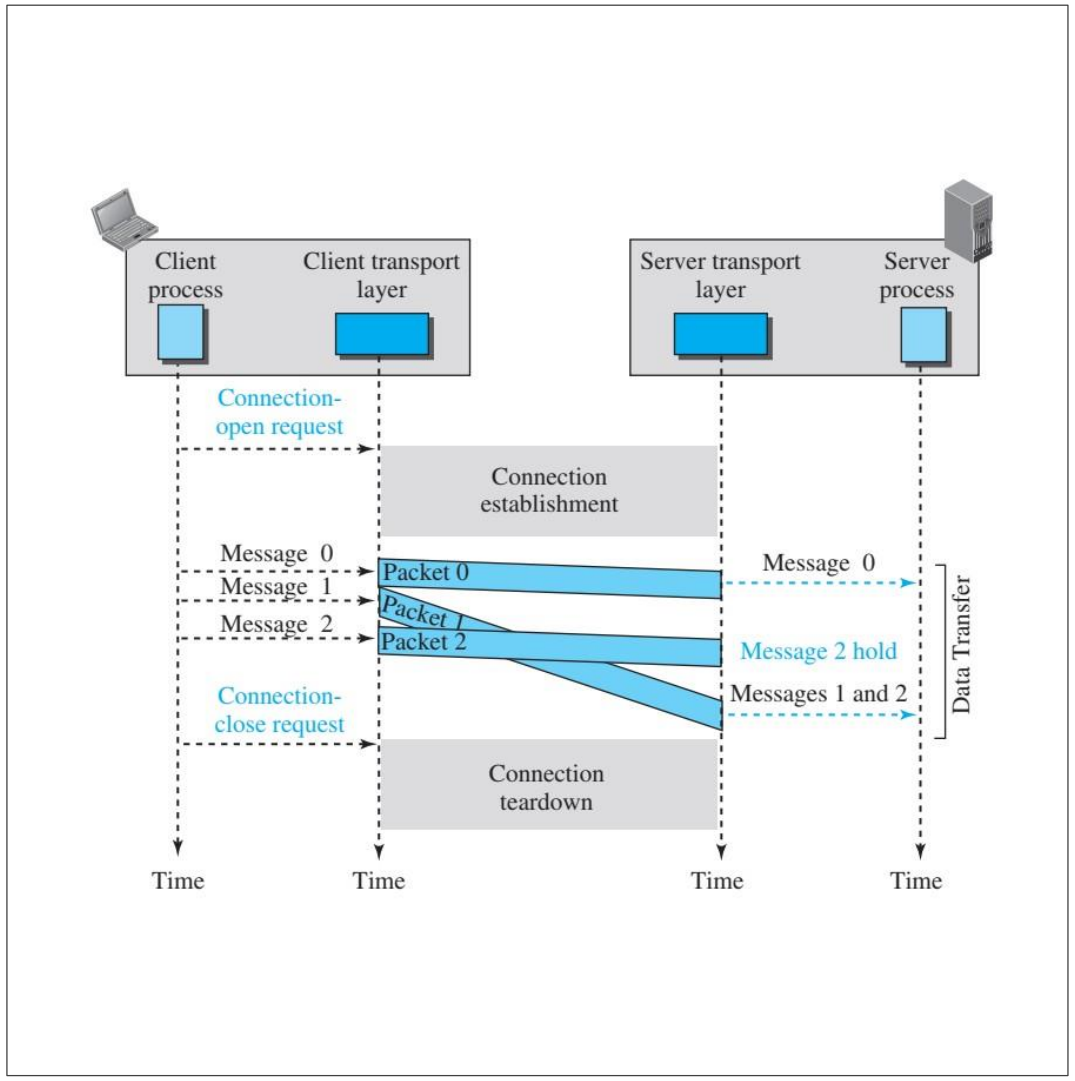
Elements of Transport Layer

- The following is also called asthe elements of transport layer
  - Addressing: Port Numbers
  - Multiplexing and Demultiplexing
  - Flow Control
  - Error Control
  - Connection-Establishment and Release (Will be discussed in TCP)



Connection-less Protocols

- No numbering for each fragment, the two transport layers do not coordinate with each other.
- Fragments may arrive in different order.
- The situation would be worse if one of the fragments were lost.
- We can say that no flow control, error control, or congestion control can be effectively implemented in a connectionless service.



Connection-Oriented Protocols

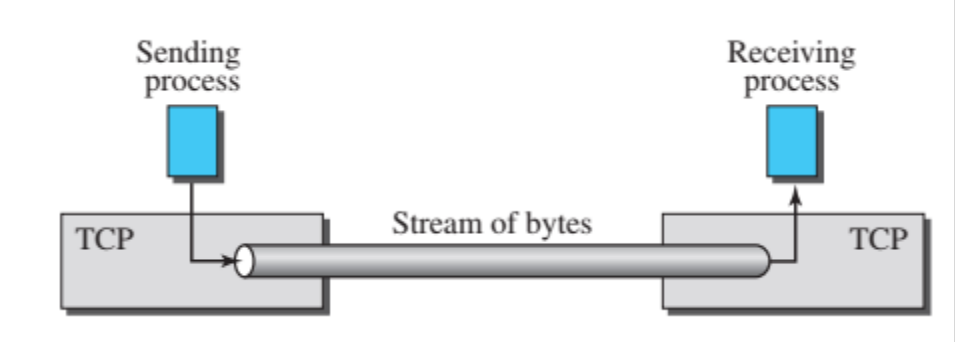
- At the transport layer, connection-oriented service involves only the two hosts; the service is end to end. This means that we should be able to make a connection-oriented protocol at the transport layer over either a connectionless or connection-oriented protocol at the network layer.
- We can implement flow control, error control, and congestion control in a connection-oriented protocol.

Transmission Control Protocol

- The Internet has two main protocols in the transport layer, a connectionless protocol and a connection-oriented one. The protocols complement each other.
- TCP (Transmission Control Protocol) was specifically designed to provide a reliable end-to-end byte stream over an unreliable internetwork.
- An internetwork differs from a single network because different parts may have wildly different topologies, bandwidths, delays, packet sizes, and other parameters.
- TCP was designed to dynamically adapt to properties of the internetwork and to be robust in the face of many kinds of failures.

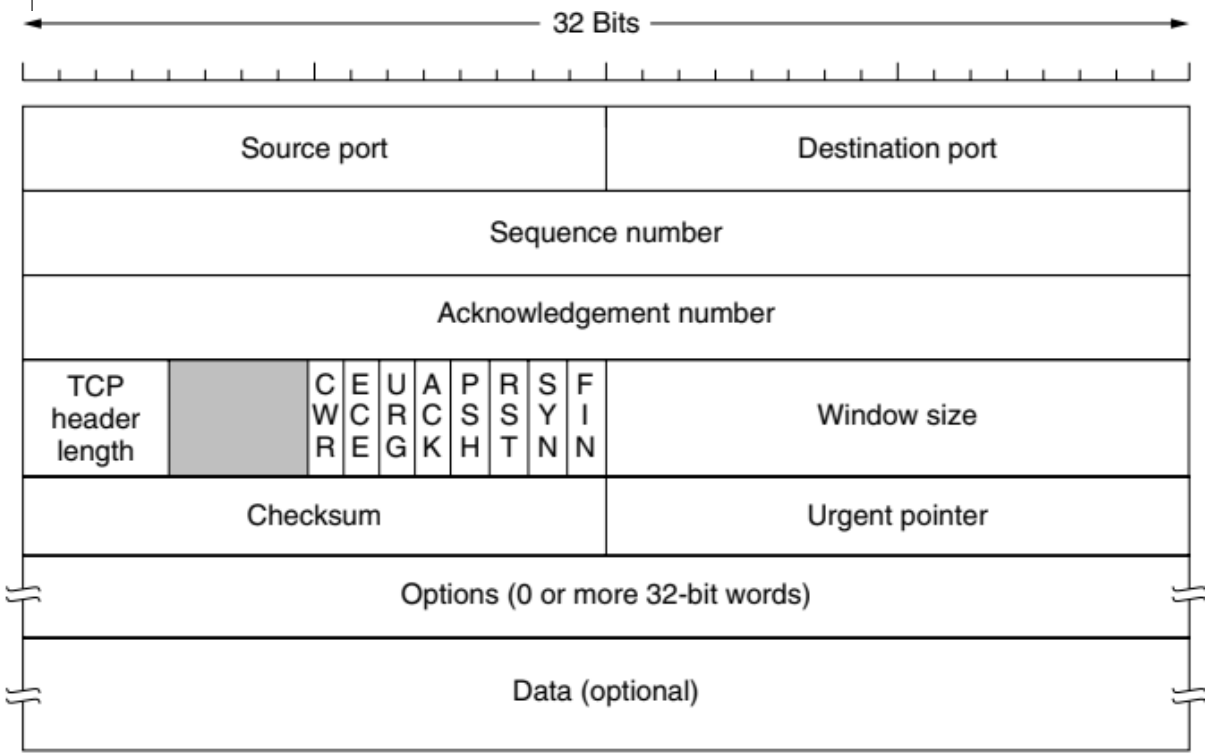
The TCP Service Model

- All TCP connections are full duplex and point-to-point.
- TCP does not support multicasting or broadcasting.
- A TCP connection is a byte stream, not a message stream.

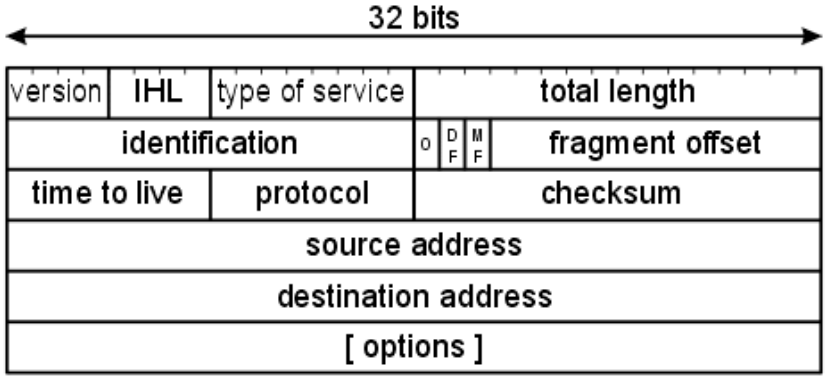


The TCP Segment

- A TCP segment consists of a fixed 20-byte header (plus an optional part) followed by optional headers followed by zero or more data bytes.
- Two limits restrict the segment size.
  - First, each segment, including the TCP header, must fit in the 65,515- byte IP payload.
  - Second, each link has an MTU (Maximum Transfer Unit). Each segment must fit in the MTU at the sender and receiver so that it can be sent and received in a single, unfragmented packet. In practice, the MTU is generally 1500 bytes (the Ethernet payload size) and thus defines the upper bound on segment size.



IP header format



The TCP Segment Header

- Every segment begins with a fixed-format, 20-byte header.
- Optional Headers
- Data of size 65535-20-20=65495.
- Segments without any data are legal and are commonly used for acknowledgements and control messages.

TCP Segment Header

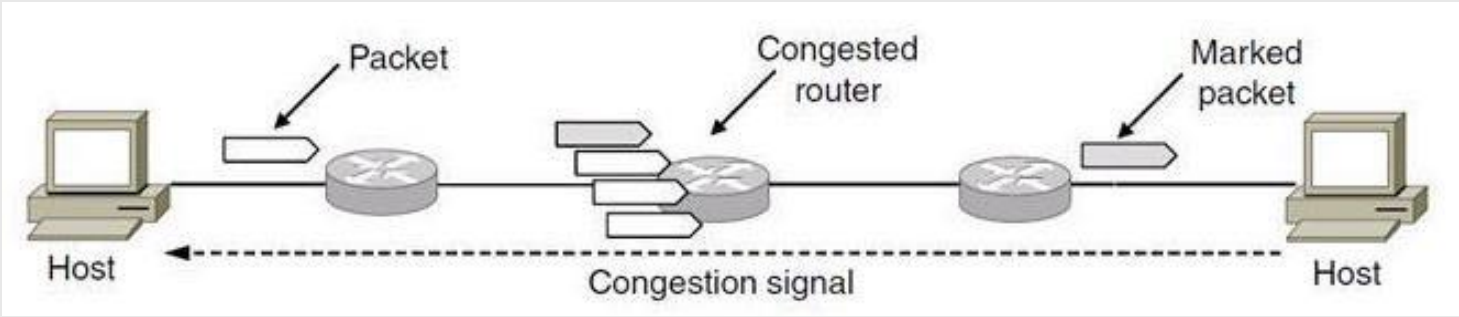
- ***Source port & Destination port:*** identify the local end points of the connection.
- ***Sequence Number:*** Numbering of the segments.
- ***Acknowledgement number:*** the next in-order byte expected, not the last byte correctly received.
- ***TCP header length:*** Number of 32-bit words are contained in the TCP header.
- Next comes a ***4-bit field that is not used***.
- Next 8 bits are used as **FLAGS**.

TCP Segment Header

- ***Window size:*** Tells how many bytes may be sent starting at the byte acknowledged.
- ***Checksum:*** Error Control Information for reliability.
- ***Urgent pointer:*** Indicate a byte offset from the current sequence number at which urgent data are to be found. This facility is in lieu of interrupt messages.
- ***Options:*** Provides a way to add extra facilities not covered by the regular header including negotiations during connection establishment better reliability etc..

FLAGS

- 8 flags of each one bit, set to “0” when not used and set to “1” when used.
- ***CWR & ECE:*** Used to signal congestion when ECN is used.
- ECE is set to signal an ECN-Echo to a TCP sender to tell it to slow down when the TCP receiver gets a congestion indication from the network.
- CWR is set to signal Congestion Window Reduced from the TCP sender to the TCP receiver so that it knows the sender has slowed down and can stop sending the ECN-Echo.



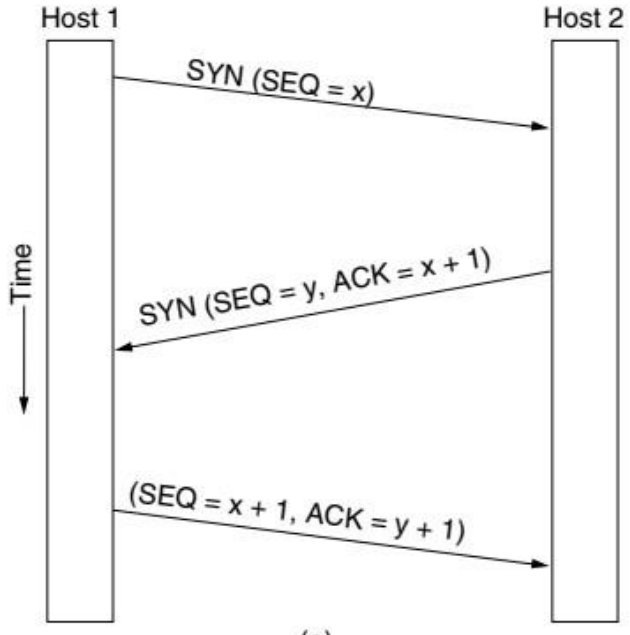
FLAGS:

- ***URG:*** Set to 1 if the Urgent pointer is in use.
- ***ACK:*** Set to 1 to indicate that the Acknowledgement number is valid, else the Acknowledgement field is neglected.
- ***PSH:*** PuSHed data, i.e., The receiver is hereby kindly requested to deliver the data to the application upon arrival and not buffer it until a full buffer has been received.
- ***RST:*** Used to abruptly reset a connection that has become confused due to a host crash or some other reason.
- ***SYN:*** Used to establish a connection.
- ***FIN:*** Used to release a connection.

TCP Communication

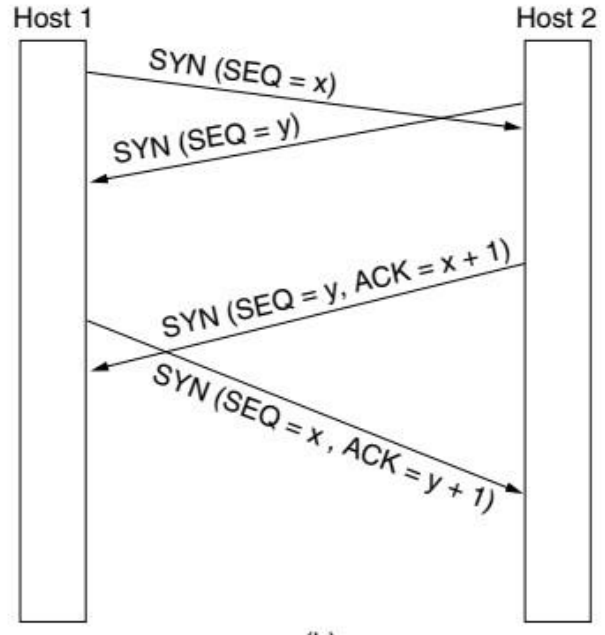
- A Connection-oriented transport service
  - Connection Establishment
  - Data Transfer
  - Connection Release

TCP Connection Establishment:



(a)

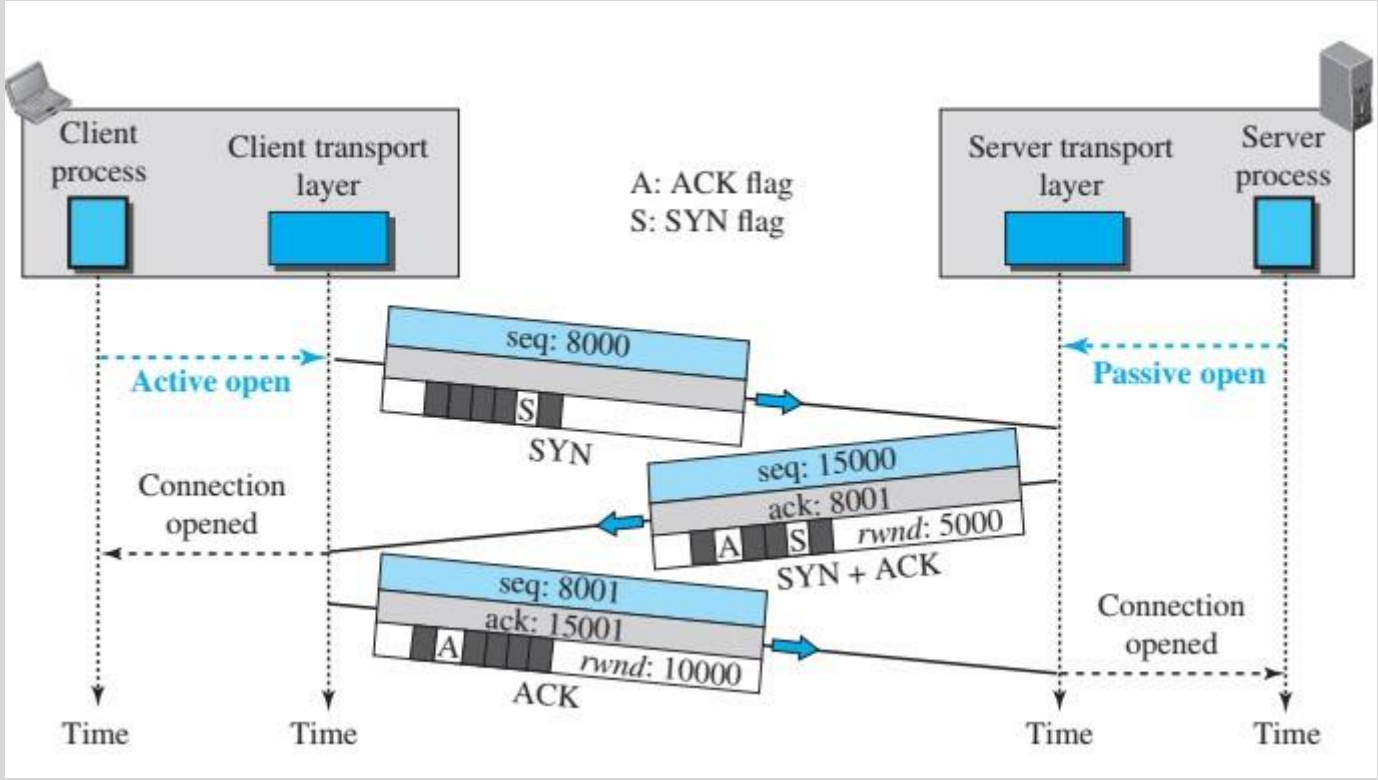
TCP connection establishment in the normal case.



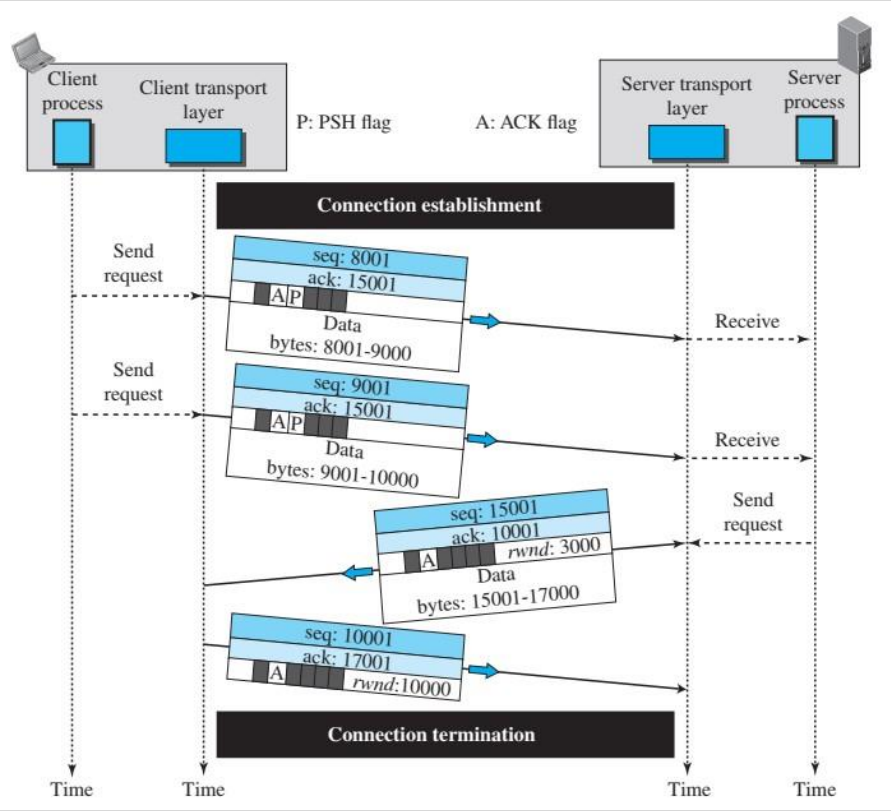
(b)

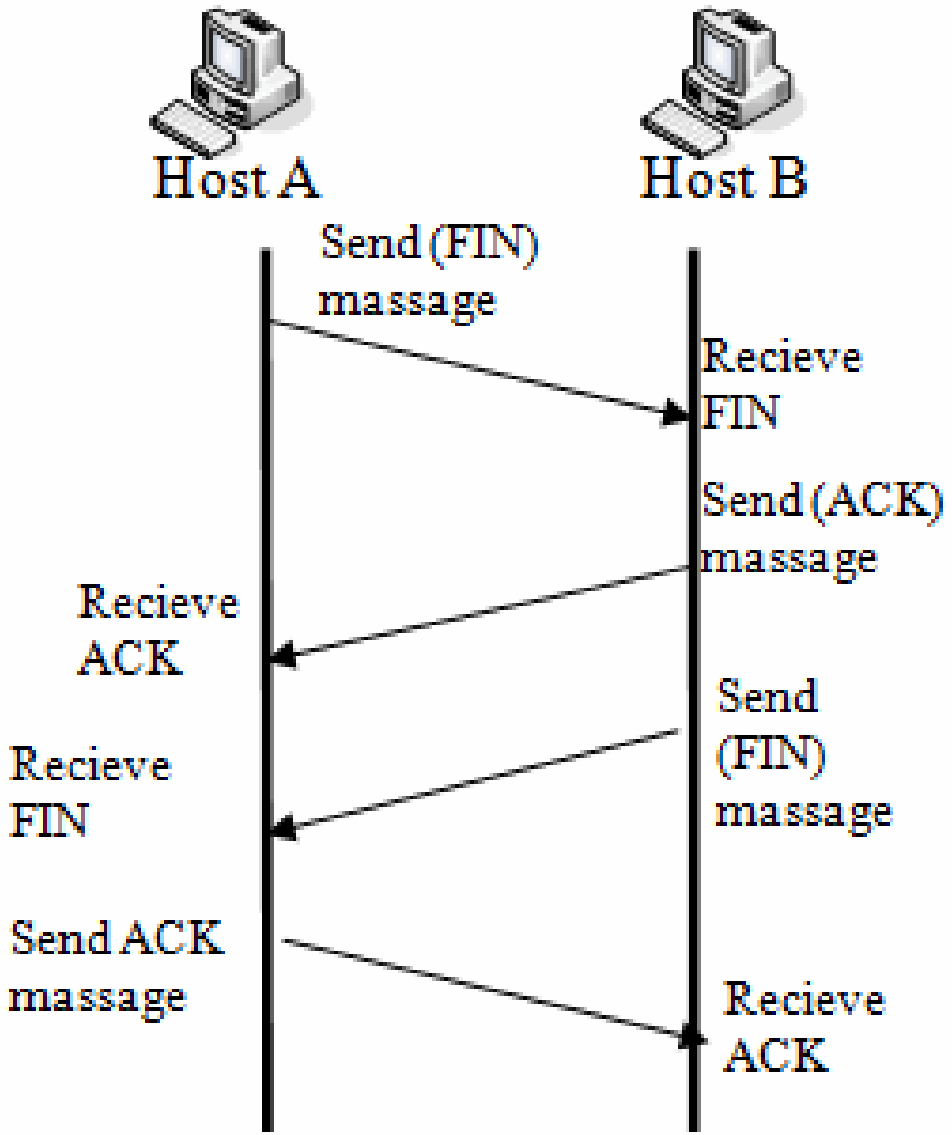
Simultaneous connection establishment on both sides.

TCP Connection Establishment:

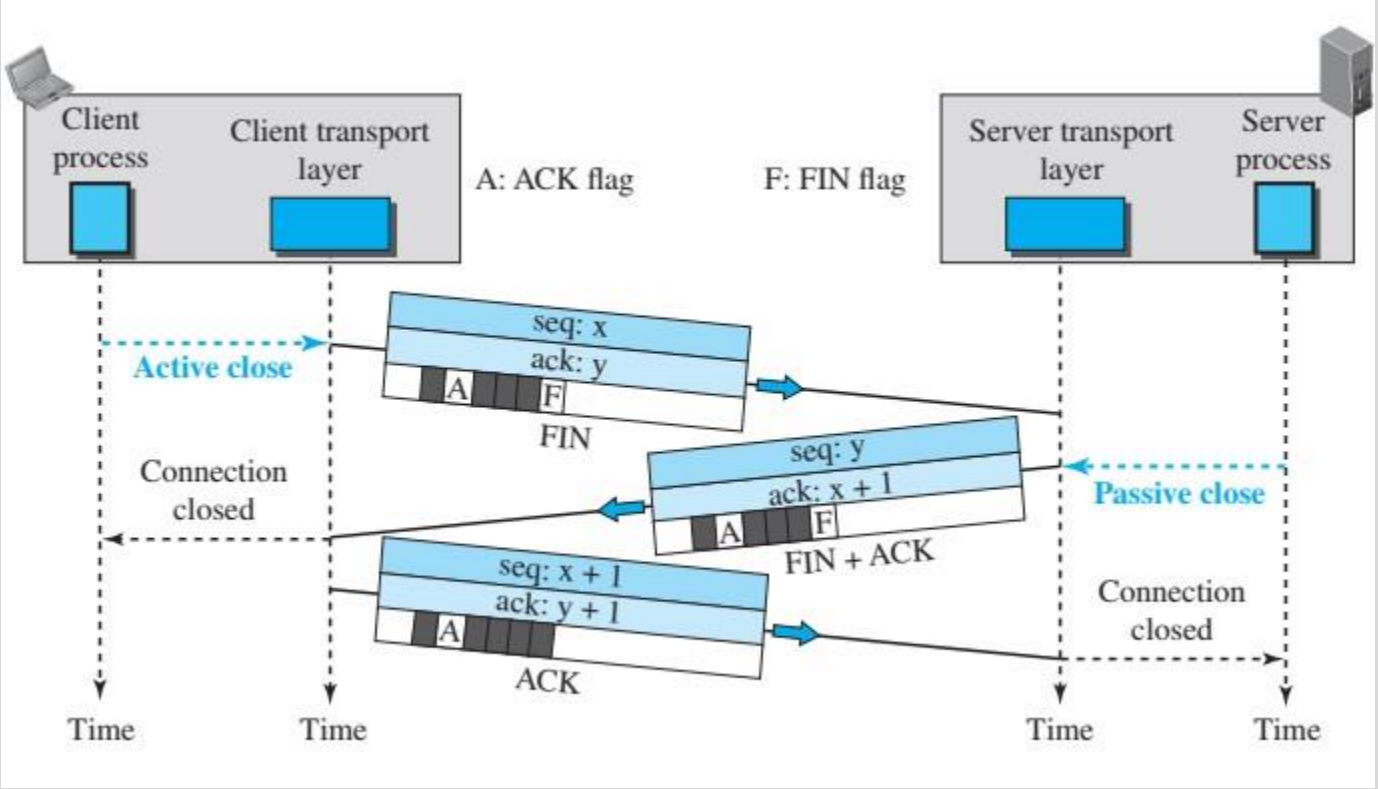


Data Transfer

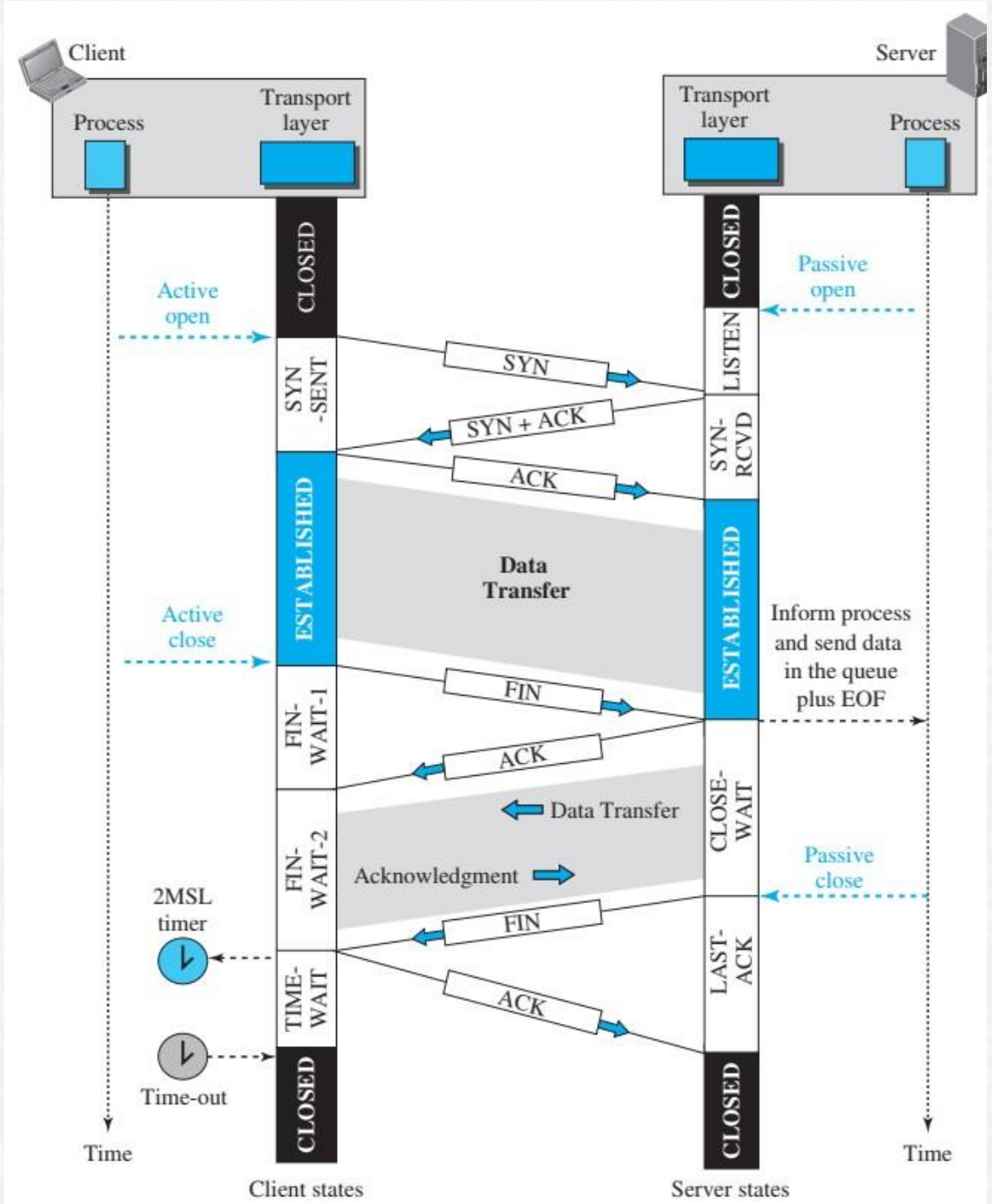




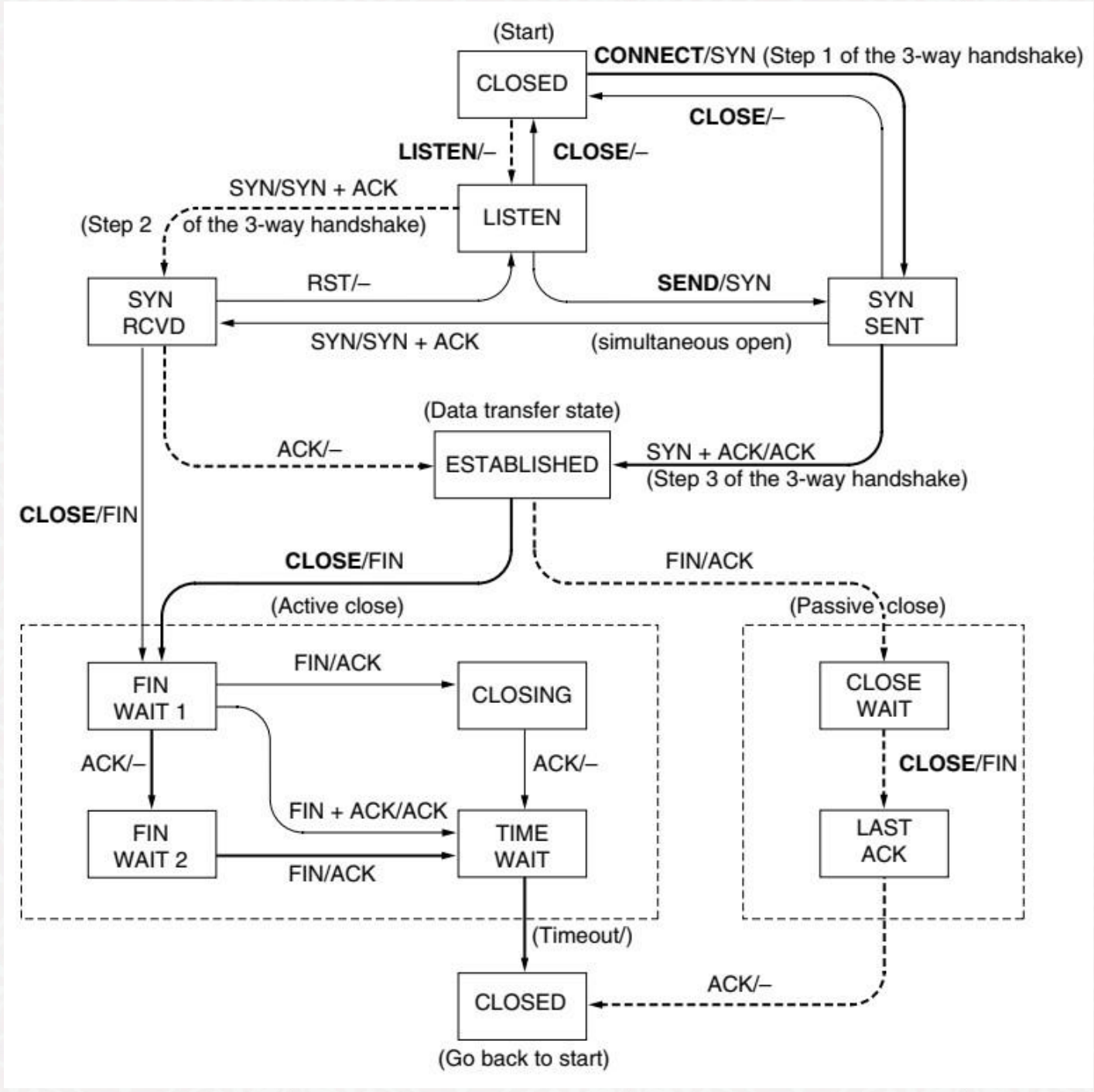
TCP *CONNECTION RELEASE*



TCP *CONNECTION RELEASE*



TCP CONNECTION



TCP CONNECTION MANAGEMENT FINITE STATE MACHINE.

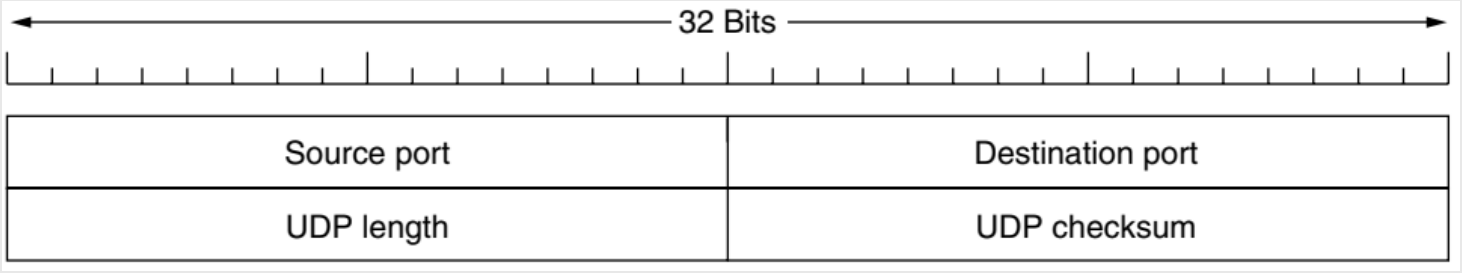
(JUST FOR KNOWING)

User Datagram Protocol

- UDP is a connectionless protocol unreliable protocol.
- It does almost nothing beyond sending packets between applications, letting applications build their own protocols on top as needed.
- UDP provides a way for applications to send encapsulated IP datagrams without having to establish a connection.
- UDP transmits segments consisting of an 8-byte header followed by the payload.

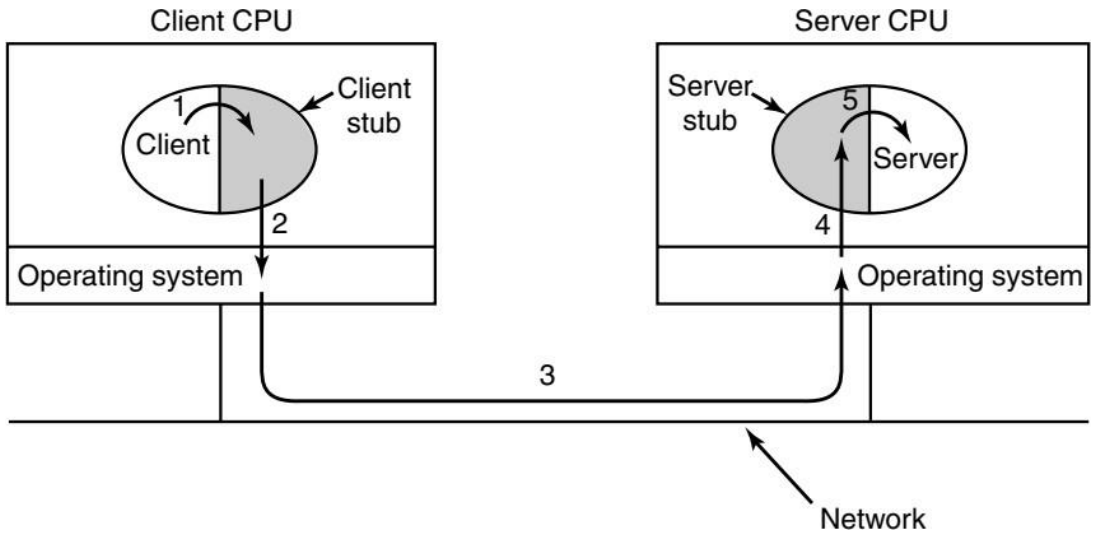
UDP Segment Header

- The two ports serve to identify the endpoints within the source and destination machines.
- The UDP length field includes the 8-byte header and the data. The minimum length is 8 bytes, to cover the header. The maximum length is 65,515 bytes
- An optional Checksum is also provided for extra reliability. It checksums the header, the data, and a conceptual IP pseudo header.



Remote Procedure Call

- The RPC (Remote Procedure Call) and has become the basis for many networking applications.
- Here, a process on machine 1 calls a procedure on machine 2, the calling process on 1 is suspended and execution of the called procedure takes place on 2.
- Information can be transported from the caller to the callee in the parameters and can come back in the procedure result.
- No message passing is visible to the application programmer.
- Traditionally, the calling procedure is known as the client and the called procedure is known as the server.
- Traditionally, the calling procedure is known as the client and the called procedure is known as the server

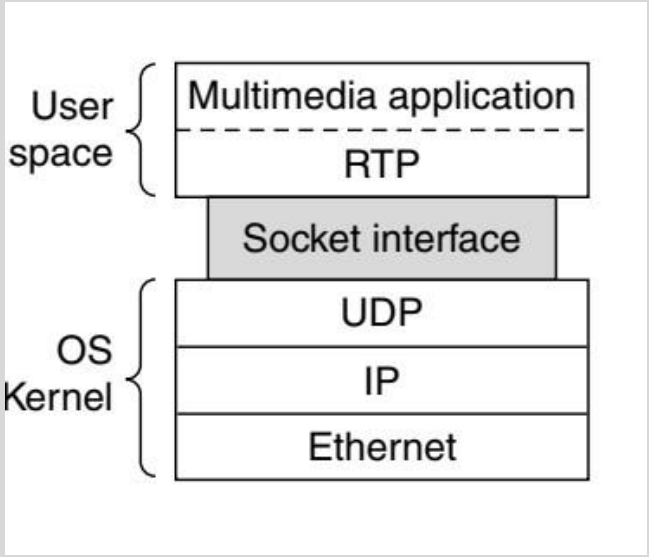


RPC (Cont..)

- Traditionally, the calling procedure is known as the client and the called procedure is known asthe server.
- The key item to note here is that the client procedure, written by the user, just makes a normal (i.e., local) procedure call to the client stub, which has the same name asthe server procedure.

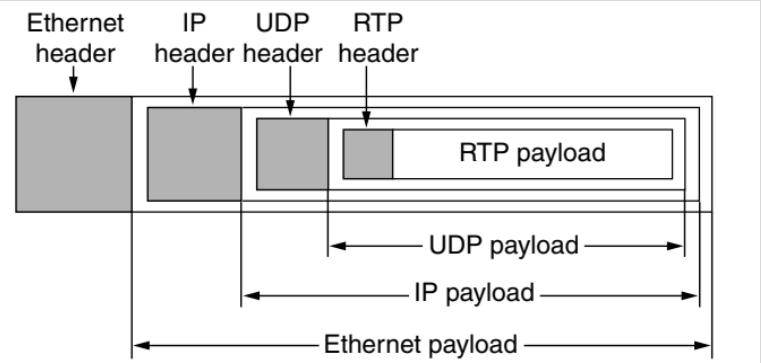
Real-Time Transport Protocol (RTTP/RTP)

- In the initial days UDP is widely used in for real-time multimedia applications.
- It gradually became clear that having a generic real-time transport protocol for multiple applications would be a good idea.
- Thus was RTP (Real-time Transport Protocol) born.



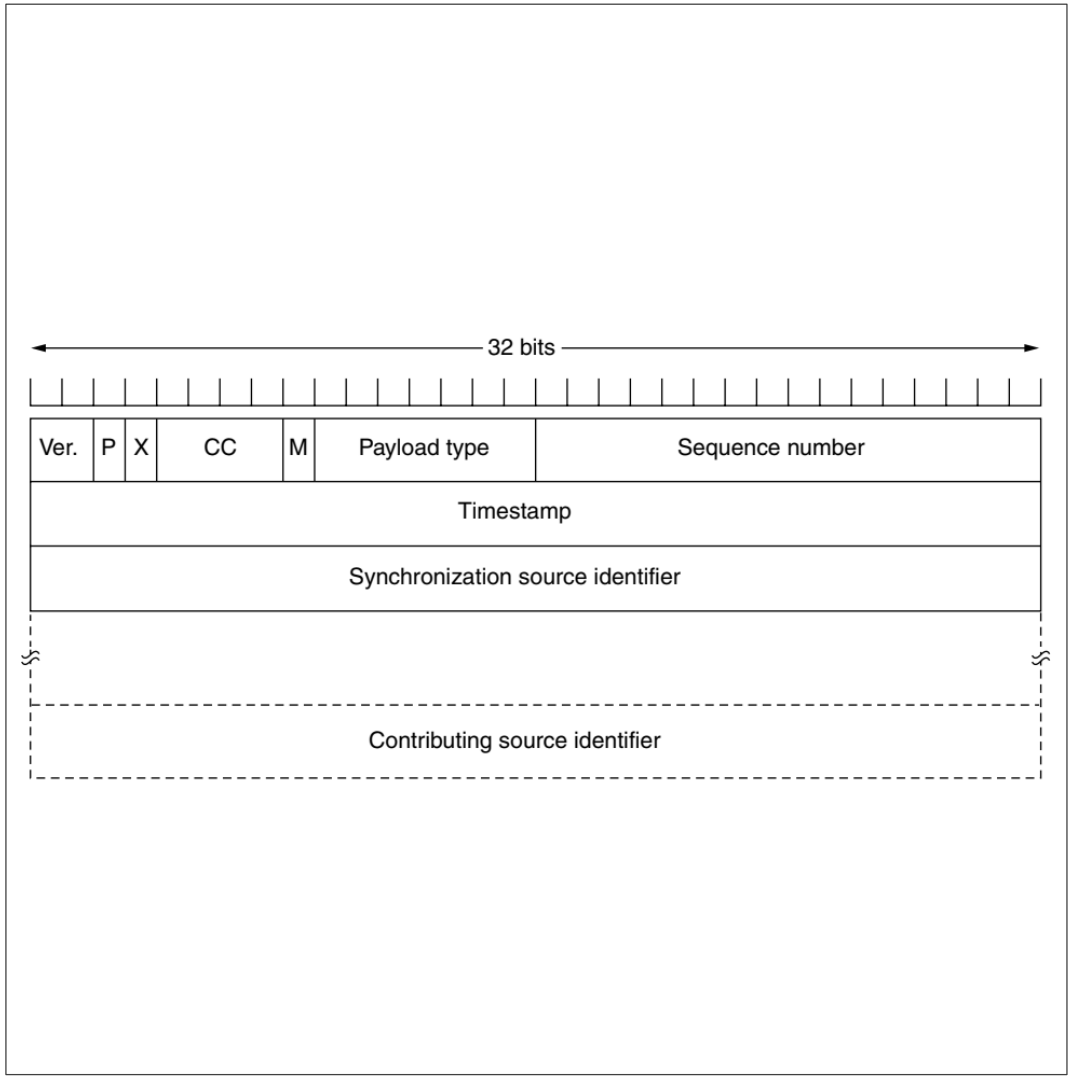
NOTE

It is a little hard to say which layer RTP is in. Since it runs in user space and is linked to the application program, it certainly looks like an application protocol. On the other hand, it is a generic, application independent protocol that just provides transport facilities, so it also looks like a transport protocol.



RTP/RTTP

- RTP normally runs in user space over UDP.
- The multimedia application consists of multiple audio, video, text, and possibly other streams. These are fed into the RTP library, which is in user-space along with the application.
- **The RTP library multiplexes the streams and encodes them in RTP packets, which it stuffs into a socket.**
- On the operating system side of the socket, UDP packets are generated to wrap the RTP packets and handed to IP for transmission over a link such as Ethernet.
- The reverse process happens at the receiver.
- The multimedia application eventually receives multimedia data from the RTP library. It is responsible for playing out the media.



RTP/RTTP

- Because RTP just uses normal UDP, there are no special guarantees about delivery, and packets may be lost, delayed, corrupted, etc.
- For smooth and better communication, the RTP packets are:
  - Numbered.
  - Retransmission is avoided
  - Defines multiple profiles for transmission of various types of multimedia content.
  - Time stamping the packets.
- The RTP header consists of three 32-bit words and potentially some extension.

RTP/RTTP

- *Ver.:* Version, current 2
- *P: Indicates that the packet has been padded to a multiple of 4 bytes.*
  - *The last padding byte tells how many bytes were added.*
- *X: Indicates that an extension header is present.*
- *CC: Number of contributing sources present (CSRC), i.e., the number of CSRC identifiers that follow the fixed header.*
- *M: An application-specific marker bit*

RTP/RTTP

- *Payload type: Type of encoding algorithm used.*
- *Sequence number: A counter that is incremented on each RTP packet.*
- *Timestamp is produced by the stream's source.*
- *Synchronization source identifier: Tells which stream the packet belongs to (Multiplexing and Demultiplexing).*
- *Contributing source identifiers, if any, are*
- *used when mixers are present in the studio*

Real-time Transport Control Protocol (RTCP)

- RTP's little sister protocol.
- RTCP handles
  - Feedback: On delay, variation in delay or jitter, bandwidth, congestion, and other network properties to the sources.
  - Synchronization: Different streams may use different clocks, with different granularities and different drift rates. RTCP can be used to keep them in sync.
  - The user interface: provides away for naming the various sources, like displaying on the receiver's screen to indicate who is talking at the moment.
- It does not transport any media samples.

Congestion Control

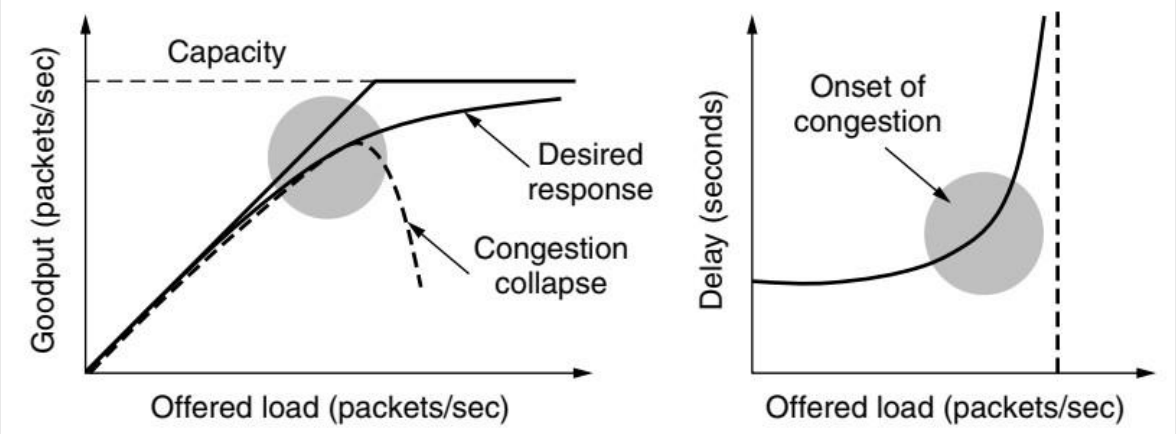
- If the transport entities on many machines send too many packets into the network too quickly, the network will become congested, with performance degraded as packets are delayed and lost.
- Controlling congestion to avoid this problem is the combined responsibility of the network and transport layers.
- Congestion occurs at routers, so it is detected and handled at the network layer.
- However, congestion is ultimately caused by traffic sent into the network by the transport layer.
- The Internet relies heavily on the transport layer for congestion control, and specific algorithms are built into TCP and other protocols.

Congestion Control

- Congestion control can be achieved by
  - Desirable Bandwidth Allocation.
  - Regulating the Sending Rate.

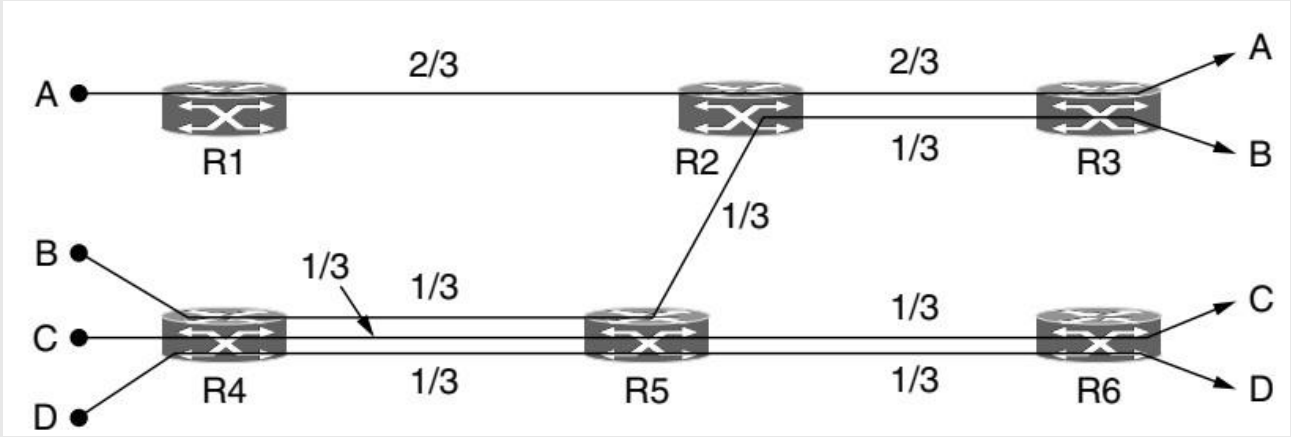
Desirable Bandwidth Allocation

- The goal is more than to simply avoid congestion.
- It is to find agood allocation of bandwidth to the transport entities that are using the network.
- For both goodput and delay, performance begins to degrade at the onset of congestion. Intuitively, we will obtain the best performance from the network if we allocate bandwidth up until the delay starts to climb rapidly



Desirable Bandwidth Allocation

- How to divide bandwidth between different transport senders?
- Ans: give all the senders an equal fraction of the bandwidth. But it is not right.
- It needs several considerations.
- Hence, the right answer is **"max-min fairness".**
- The max-min fairness is said to be achieved by an allocation if and only if the allocation is feasible and an attempt to increase the allocation of any participant necessarily results in the decrease in the allocation of some other participant with an equal or smaller allocation



Desirable Bandwidth Allocation

- The bandwidth needed by a given connection will vary over time as the connections are always coming and going.
- A good congestion control algorithm should rapidly converge to the ideal operating point, and it should track that point as it changes over time.
- If the convergence is too slow, the algorithm will never be close to the changing operating point.
- If the algorithm is not stable, it may fail to converge to the right point in some cases, or even oscillate around the right point.

Regulating the Sending Rate

- How do we regulate the sending rates to obtain a desirable bandwidth allocation?
- The sending rate may be limited by two factors.
  - The first is flow control, in the case that there is insufficient buffering at the receiver.
  - The second is congestion, in the case that there is insufficient capacity in the network.
- The way that a transport protocol should regulate the sending rate depends on the form of the feedback returned by the network.
- The feedback may be
  - Explicit or Implicit
  - Precise or Imprecise.

Protocol	Signal	Explicit?	Precise?
XCP	Rate to use	Yes	Yes
TCP with ECN	Congestion warning	Yes	No
FAST TCP	End-to-end delay	No	Yes
Compound TCP	Packet loss & end-to-end delay	No	Yes
CUBIC TCP	Packet loss	No	No
TCP	Packet loss	No	No

Regulating the Sending Rate

Chiu and Jain (1989) studied the caseof binary congestion feedback and concluded that AIMD (Additive Increase Multiplicative Decrease) is the appropriate control law to arrive at the efficient and fair operating point.

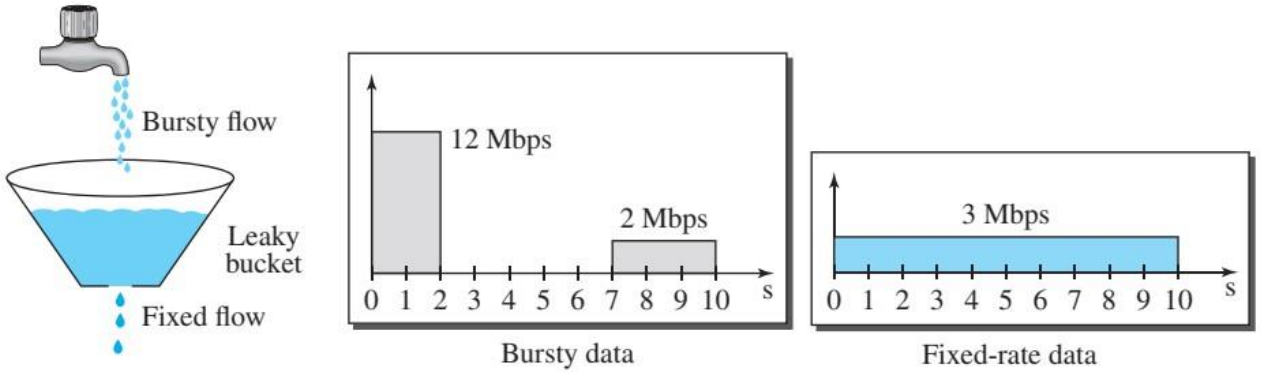
Quality of Service

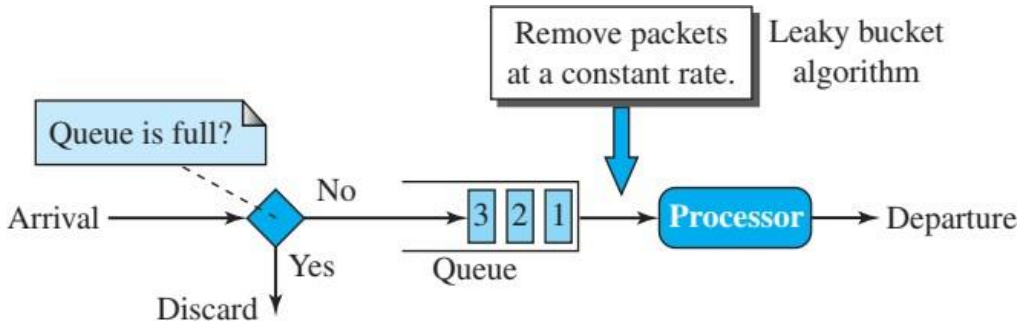
- Quality of service (QoS) is an internetworking issue that refers to a set of techniques and mechanisms that guarantee the performance of the network to deliver predictable service to an application program.
- For better QoS the following must be handled.
  - Data-flow characteristics: Reliability, Delay, Jitter, & Bandwidth.
  - Flow Control: Scheduling, **Traffic Shaping.**
  - Integrated (BW)and Differentiated Services (Priority).

Traffic Shaping or Policing

- To control the amount and the rate of traffic is called traffic shaping or traffic policing.
- Two techniques can shape or police the traffic:
  - leaky bucket
  - token bucket.

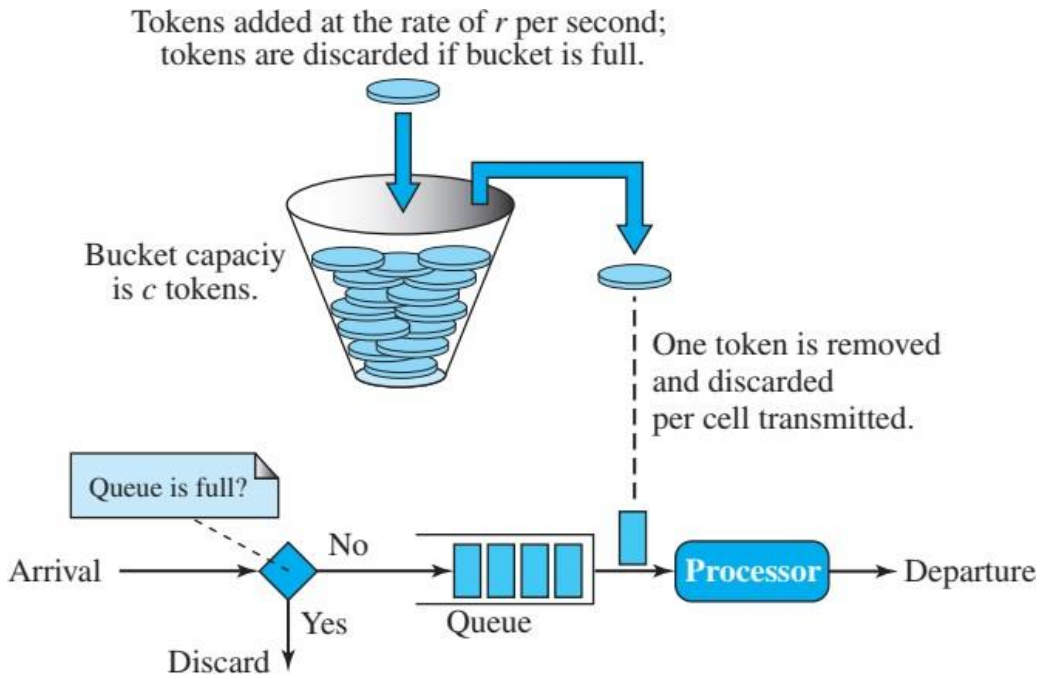
Leaky Bucket





- The input rate can vary, but the output rate remains constant hence smoothing out the bursty traffic.
- Bursty chunks are stored in the bucket and sent out at an average rate.

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



$$\text{Maximum number of packets} = r \times t + c$$

$$\text{Maximum average rate} = (r \times t + c)/t \text{ packets per second}$$

Token Bucket

- The leaky bucket is very restrictive. It does not credit an idle host.
- On the other hand, the token bucket algorithm allows idle hosts to accumulate credit for the future in the form of tokens.

Assume the capacity of the bucket is ***c*** tokens and tokens enter the bucket at the rate of ***r*** tokens per second. The system removes one token for every cell of data sent. The maximum number of cells that can enter the network during any time interval of length ***t*** is shown below.