SREELA CHANDRAN, AP

CST 303 COMPUTER NETWORKS

TRANSPORT LAYER AN APPLICATION LAYER

MODULE 5

Transport service – Services provided to the upper layers, Transport service primitives. User Datagram Protocol (UDP). Transmission Control Protocol (TCP) – Overview of TCP, TCP segment header, Connection establishment &release, Connection management modeling, TCP retransmission policy, TCP congestion control. Application Layer –File Transfer Protocol (FTP), Domain Name System (DNS), Electronic mail, Multipurpose Internet Mail Extension (MIME), Simple Network Management Protocol (SNMP), World Wide Web(WWW) – Architectural overview.

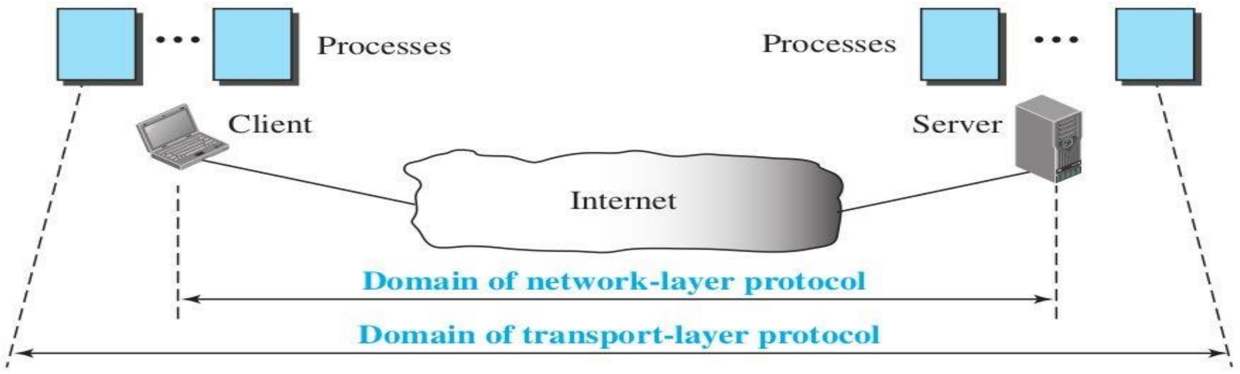
TRANSPORT LAYER

SERVICES

1. Process-to-Process Communication
2. Encapsulation and Decapsulation
3. Multiplexing and Demultiplexing
4. Flow Control
5. Error Control
6. Congestion Control
7. Connectionless and Connection-oriented services.

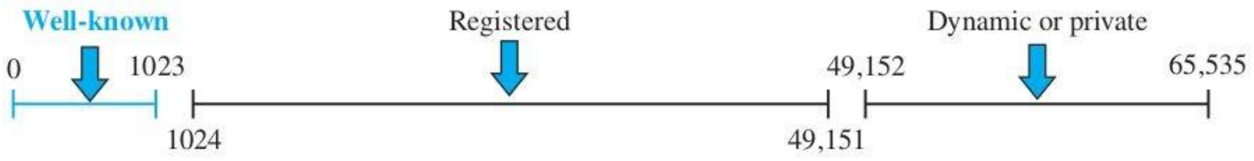
Process-to-Process Communication

- A process is an application-layer entity (running program) that uses the services of the transport layer
- Network layer helps the communication between computers(host-to-host).
- Transport layer helps the communication between processes in computers


Processes
Processes
Client
Server
Internet
Domain of network-layer protocol
Domain of transport-layer protocol

Addressing

- Hosts are identified by IP addresses, and the processes are identified by port numbers.
- Port numbers are of 16 bits (0 - 65535).
- Most process-to-process communication happens through client-server paradigm.
- Client and server processes are identified by the port numbers.
- Ports division by ICANN (Internet Corporation for Assigned Names and Numbers)



- Well-known ports: 0-1023. Assigned and controlled by ICANN.
  - Eg: Port 80 - http server
- Registered ports-1024 - 49151. They can only be registered with ICANN to prevent duplication.
  - Eg: Port 2196 - Apple Push Notification Service, feedback service
- Dynamic ports-49152 - 65535. Can be used as temporary or private port numbers.
- Mostly, clients use ephemeral ports (also called short-lived ports) which will be greater than 1023.
- For some services, clients use well-known ports.

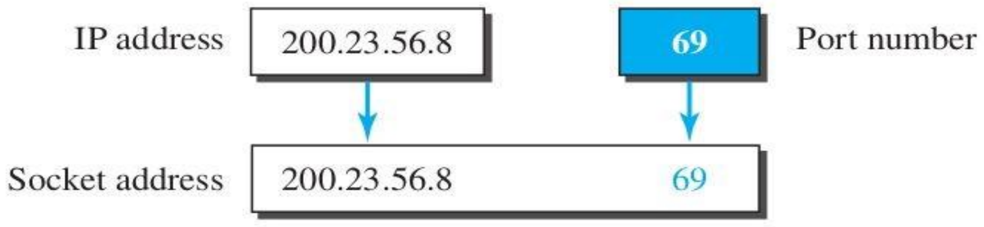
  Eg: DHCP client uses port 68
- Mostly, servers use well-known ports (upto 1023).

  Eg: http server uses port 80, DHCP server use port 67

Socket Addresses:

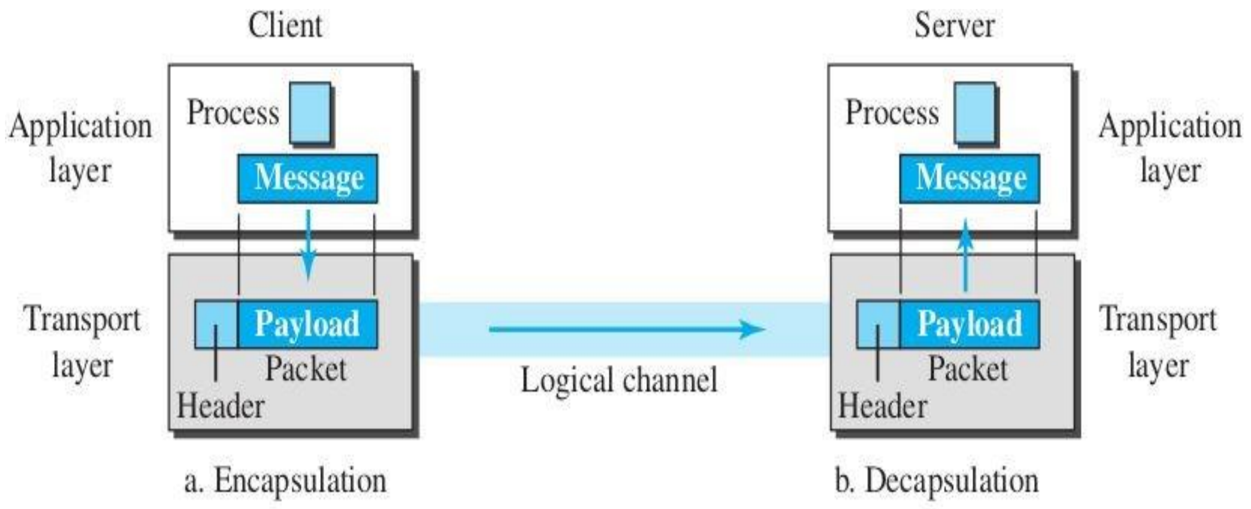
- It is the **combination of an IP address and a port number**

- To uniquely identifies a process.
- To use the services of the transport layer in the Internet, a pair of socket addresses are needed: the client socket address and the server socket address.
- These are part of network layer and transport layer headers



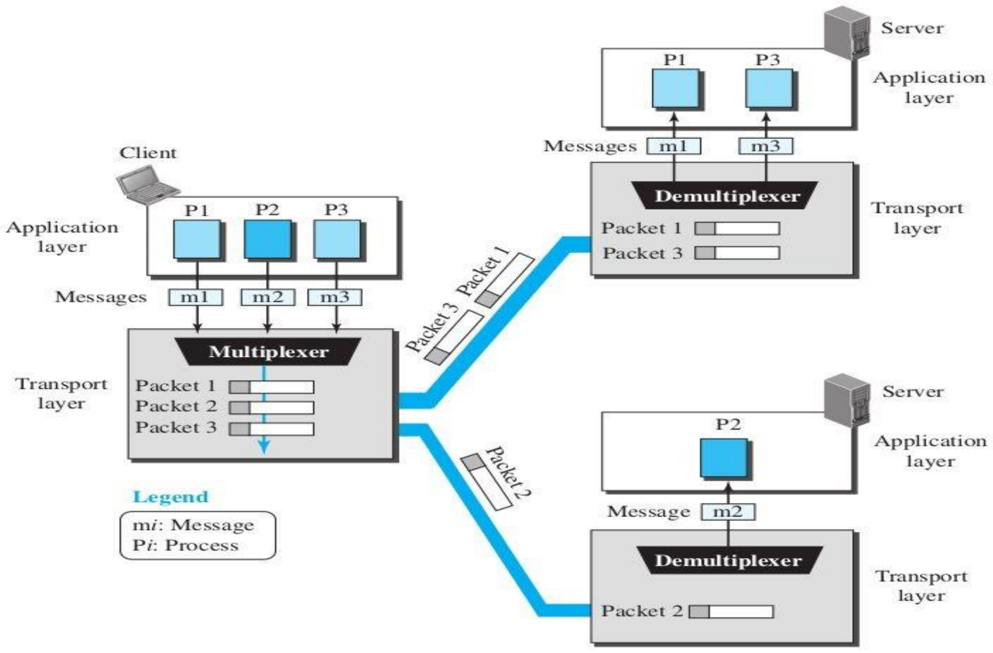
Encapsulation and Decapsulation

- Encapsulation happens at the sender side.
- At the sender the process pass the message to the transport layer along with a pair of socket addresses and some other specific information's.
- With these information, the transport layer makes a header for this message.
- The message with this header is called segment(for TCP) or user datagram(for UDP).
- Decapsulation happens at the receiver side.
- When the packet arrives at the receiver transport layer, the header is separated and the message is passed to the process running at the application layer.
- If needed socket address is also passed.



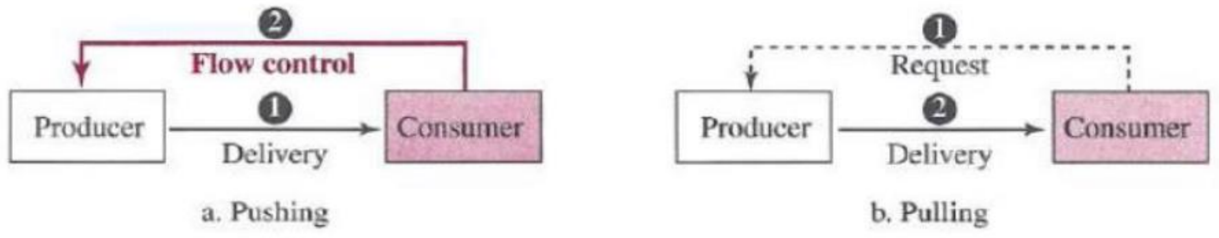
Multiplexing and Demultiplexing

- Whenever an entity accepts items from more than one source, this is referred to as multiplexing(many to one).
- Whenever an entity delivers items to more than one destination, this is referred to as demultiplexing(one to many)
- Transport layer at source performs multiplexing.
- Transport layer at the destination performs demultiplexing



Flow Control

- It is the process of balancing the rate of sending the packets based on the receiver - This is use to prevent data loss.
- Packet delivery can happen in 2 ways.
  1. **Pushing :-** the sender delivers items whenever they are produced- without a prior request from the consumer. The consumer may be overwhelmed and there is a need for flow control, in the opposite direction, to prevent discarding of the items.
  2. **Pulling :-** The producer delivers the items after the consumer has requested them.- then there is no need for flow control

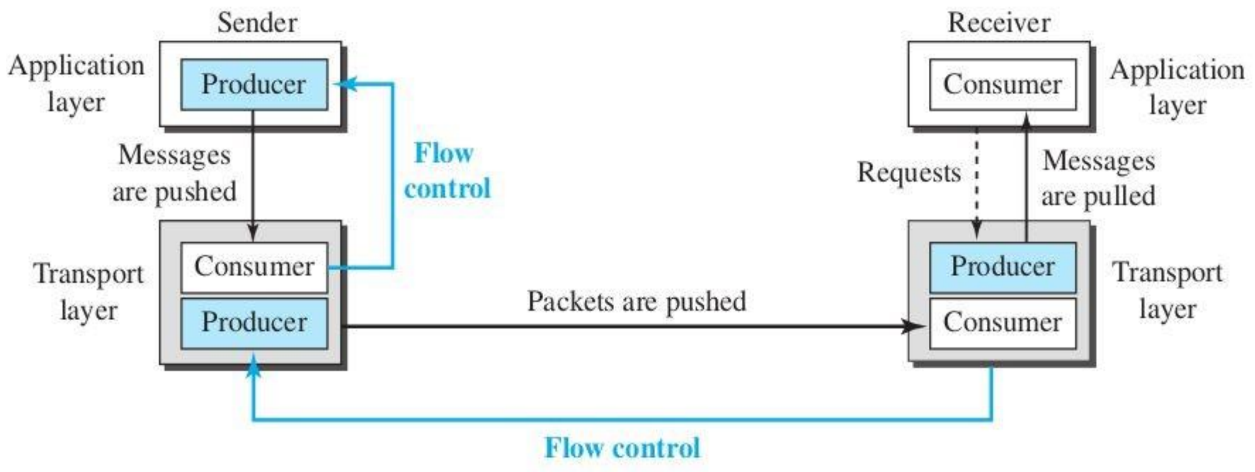


Flow control at Transport Layer

- 4 entities are involved in transport layer communication:

Sender process, sender transport layer, receiver transport layer and receiver process.

- Pushing and pulling happens at the transport layer.
- 2 flow controls are needed- buffers are used to control flow.



Error Control

- Involves the **sending** and **receiving transport layers**.
- 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.
- 2 methods used for error control are

  1. Use **sequence number** for error control, which is <u>modulo 2^m</u> if there are m-bits in the sequence number field of the header.

Sequence numbers are used to inform the sender of packet error, to know the order of packets sent and to find duplicate packets- included in header

2. **Acknowledgement with timers** are also used to resend packets, if the packet is lost, corrupted, the acknowledgement itself is lost or delayed.

<u>Combination of Flow and Error Control</u>

- Flow control requires 2 buffers: one at sender and other at the receiver.
- Error control requires the use of sequence numbers and acknowledgment numbers at both sides.
- These two can be combined
- **Sender:** When a packet is prepared to be sent, we use the number of the next free location, x, in the buffer as the sequence number of the packet. When the packet is sent, a copy is stored at memory location x, awaiting the acknowledgment from the other end. When an acknowledgment related to a sent packet arrives, the memory location is freed.
- **Receiver**: When a packet with sequence number y arrives, it is stored at the memory location y until the application layer is ready to receive it. An acknowledgment can be sent to announce the arrival of packet y.

<u>Congestion Control</u>

- Congestion occur if the load on the network is greater than the capacity of the network.
- Congestion at the transport layer is actually the result of congestion at the network layer.
- Congestion control refers to the mechanisms to keep the load below the capacity.

<u>Connectionless and Connection-Oriented Protocols</u>

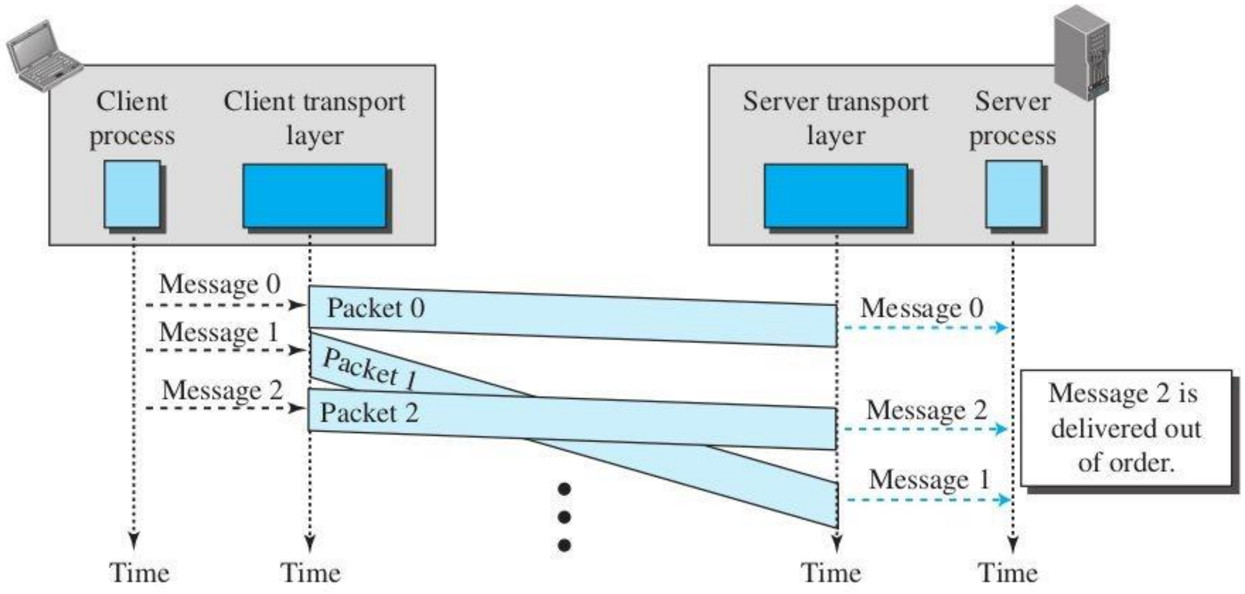
Connectionless

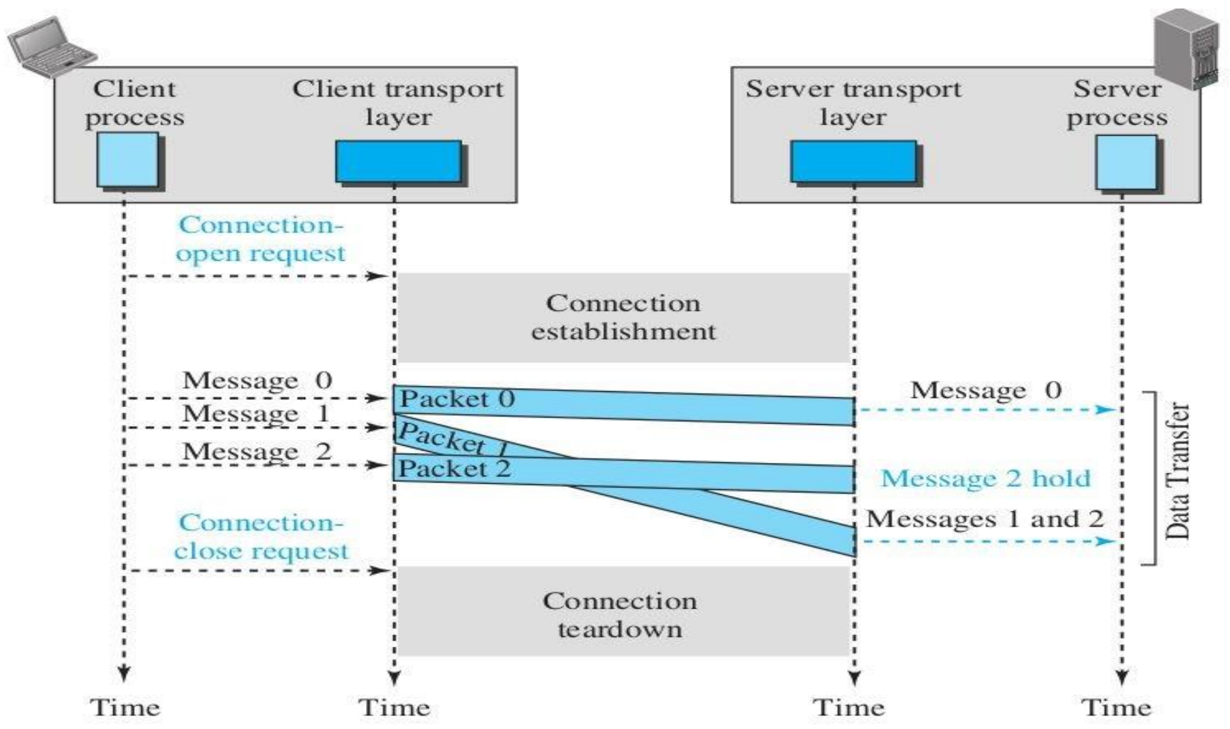
- The source process (application program) needs to divide its message into chunks of data of the size acceptable by the transport layer and deliver them to the transport layer one by one.
- The transport layer treats each chunk as independent unit, encapsulate it and send to the receiver **without any numbering**.
- The receiver may get these packets **out of order**, decapsulates them and deliver the packets in that order only.

- No flow control, error control, or congestion control can be effectively implemented in a connectionless service.

Connection oriented

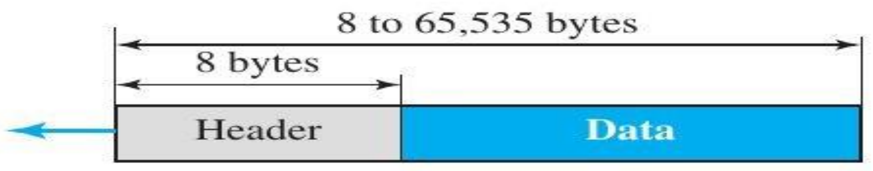
- The client and the server first need to **establish a logical connection** between themselves. Then **data exchange** happens. After this, the connection needs to be **tear down**.
- The connection-oriented protocol at the transport layer happens over either a connectionless or connection-oriented protocol at the network layer.
- Flow control, error control, and congestion control are implemented in a connection-oriented protocol.



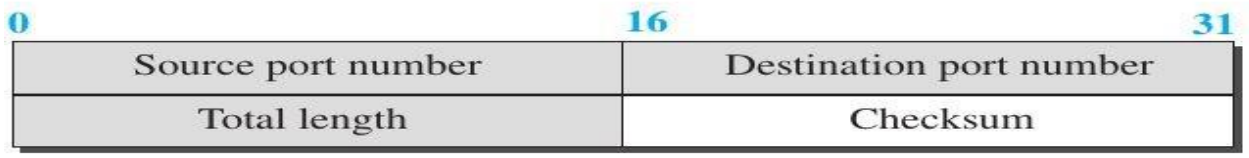


<u>User Datagram Protocol (UDP)</u>

- Connectionless, unreliable protocol.
- No error control and flow control, hence very simple protocol with minimum overhead
- If a process wants to send a small message and does not care much about reliability, it can use UDP
- User Datagrams
- UDP Packets are called user datagrams



a. UDP user datagram



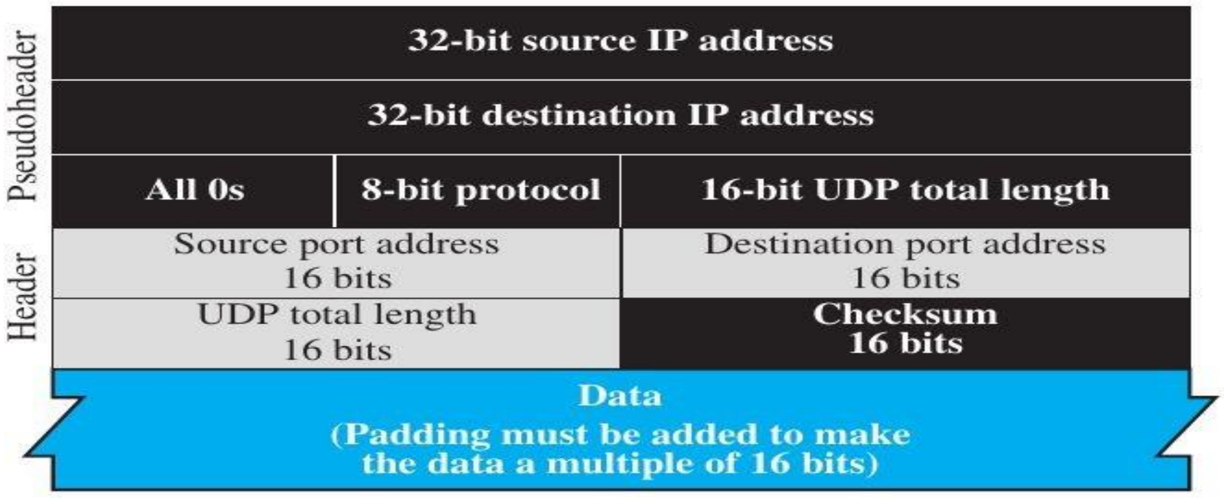
b. Header format

- A fixed-size header of 8 bytes (4 fields, each of 2 bytes)

- First two fields define the source and destination port numbers.
- Total length field (16 bits) specifies the total number of bytes of the datagram (header + payload). Actually it will be smaller than 65535.
- Last 2 bytes - optional checksum

UDP Checksum

- Optional most of the time.
- UDP checksum calculation includes three sections: a pseudoheader, the UDP header, and the data coming from the application layer.
- The pseudoheader is the part of the header of the IP packet.
- If no checksum is used, this field is filled with zeros.


Pseudoheader
32-bit source IP address
32-bit destination IP address
All 0s
8-bit protocol
16-bit UDP total length
Header
Source port address 16 bits
Destination port address 16 bits
UDP total length 16 bits
Checksum 16 bits
Data (Padding must be added to make the data a multiple of 16 bits)

<u>**UDP Services**</u>

- **Process-to-Process Communication**: Using socket addresses
- **Connectionless Services**: No connection establishment and termination, no numbering of packets.
- **Flow Control**: No flow control. If needed, the process above will take care of it.
- **Error Control**: Only checksum. Nothing else. If needed, the process above will do it.
- **Congestion Control**: Not provided
- **Encapsulation and Decapsulation**: Provided.
- **Queuing**: Some processes have incoming and outgoing queues. Some have only incoming queue.
- **Multiplexing and Demultiplexing**: Provided

UDP Features

- ❖ Connectionless Service - Independent packets

  Advantage:

  - ■ If short or single messages are to be sent.
  - ■ Less overhead due to connectionless feature.
  - ■ Less delay
- ❖ Lack of Error Control
  - ➢ Error control cause packet resending; receiver application layer has to wait for transport layer for all packets to come error-free.
  - ➢ Here it does not happen.
- ❖ Lack of Congestion Control
  - ➢ Causes very less congestion or no congestion, as packets are simple and no resending is there.

UDP Applications

- Simple request-response communication with little concern for flow and error control.
- Process with internal flow- and error-control mechanisms - For example, the Trivial File Transfer Protocol (TFTP) process includes flow and error control - It can easily use UDP.
- For multicasting: Multicasting capability is embedded in the UDP software but not in the TCP software.
- For management processes such as SNMP.
- For some route updating protocols such as Routing Information Protocol (RIP).
- For interactive real-time applications such as multimedia streaming.

TRANSMISSION CONTROL PROTOCOL (TCP)

- ➢ It is most widely used protocol for data transmission in communication network such as internet.
- ➢ A connection-oriented, reliable protocol.
- ➢ Defines connection establishment, data transfer, and connection tear-down phases.
- ➢ Uses a combination of Go-Back-N and Selective Repeat protocols to provide reliability.
- ➢ Uses checksum (for error detection), retransmission of lost or corrupted packets, cumulative and selective acknowledgments, and timers.
- ➢ TCP data packets are called **segments**.

TCP Services

- **Process-to-Process Communication**:Using port numbers.
- **Full-Duplex Communication**: Each TCP endpoint then has its own sending and receiving buffer,
- **Multiplexing and Demultiplexing**:Multiplexing at the sender and demultiplexing at the receiver.
- **Connection-Oriented Service**:Creates logical connections before sending the data.
- **Reliable Service**: Use acknowledgements for safe arrival of data.
- **Stream Delivery Service**.
  - TCP 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 uses sending buffer and receiving buffer in each direction

Numbering System

TCP keeps track of the order of segments by using the following numbers;

1. Byte number
2. Sequence number
3. Acknowledgement number

Byte Number:

- On receiving the data bytes, TCP stores them in the sending buffer and numbers them.
- The first number will be randomly selected from 0 to 2^{32}-1.
- The numbering is independent in each direction.

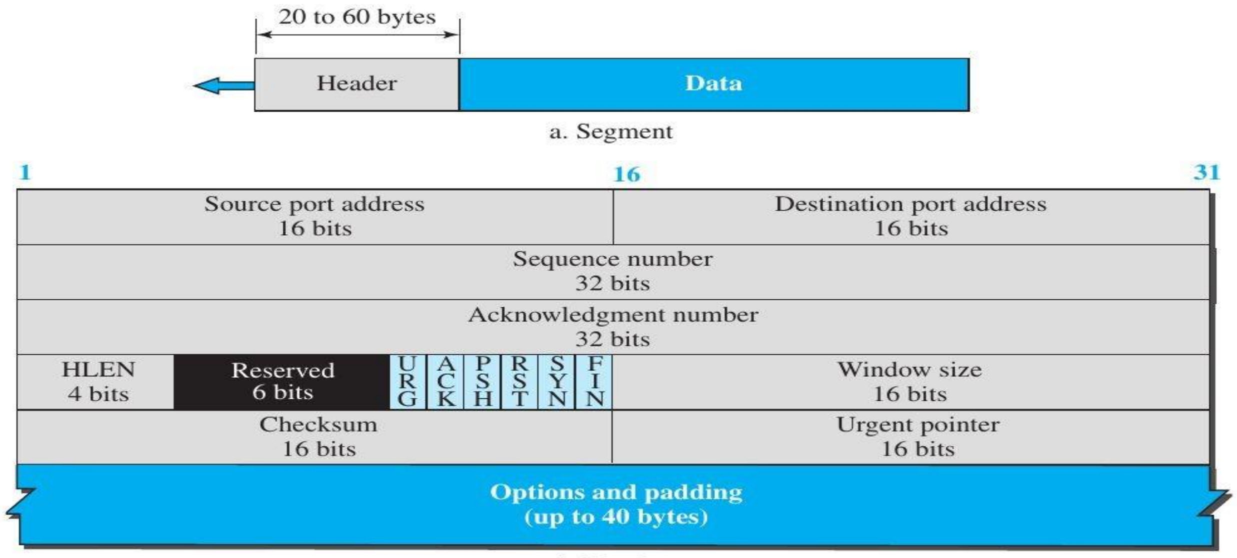
Sequence number:

- The value in the sequence number field of a segment defines the number assigned to the first data byte contained in that segment.
- The sequence number of any other segment is the sequence number of the previous segment plus the number of bytes carried by the previous segment
- The sequence number of the first segment is the ISN (Initial Sequence Number).

Acknowledgement number:

- ➢ In most cases, the value of the acknowledgment field in a segment defines the number of the next byte to be received.
  The acknowledgment number can be **cumulative**.

TCP Segment format

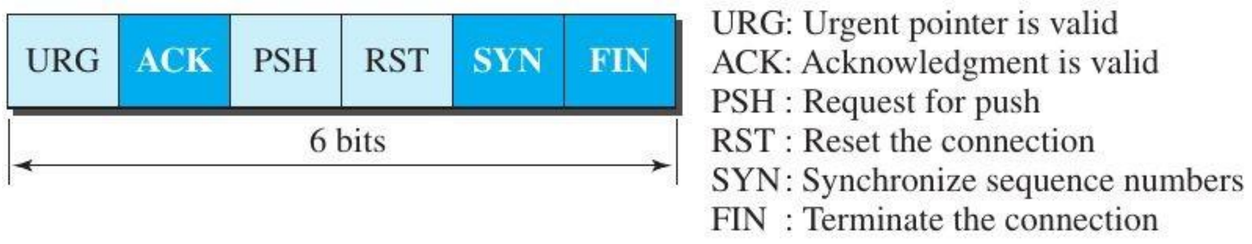


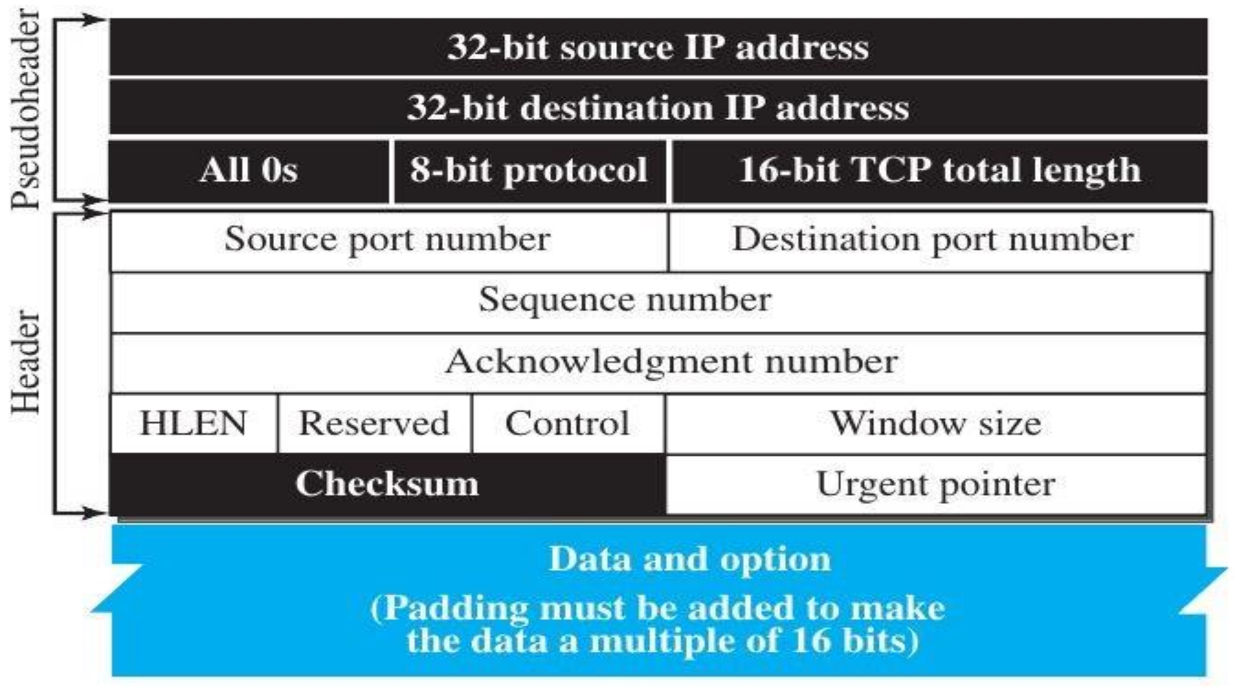
b. Header

TCP Segment format

- **Source port address**: 16-bit source port number
- **Destination port address**: 16-bit destination port number
- **Sequence number**: 32-bit. Byte number of the first data byte in the segment.
- **Acknowledgment number**: 32-bit. Added for piggybacking.
- **Header length**: Length of TCP header in 4-bit words. Values 5 to 15.
- **Control**: 6 flag bits for flow control, connection establishment and termination, connection abortion, and the mode of data transfer in TCP.
- **Window size**: 16-bit. Size of of the sending TCP in bytes. Denoted as *rwnd*.
- **Checksum**: 16-bit. Mandatory field.
- **Urgent pointer**: 16-bit. Valid only if the urgent flag is set.
- **Options**: Can be upto 40 bytes.

TCP Segment format: Control Flags


URG
ACK
PSH
RST
SYN
FIN
6 bits
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



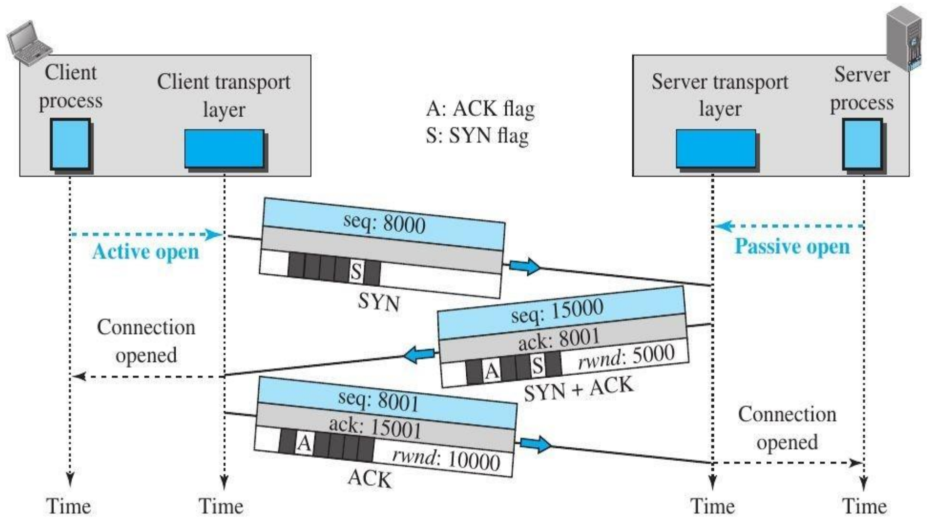
<u>A TCP Connection</u>

A TCP data transfer consists of three phases:

1. Connection establishment
2. Data transfer
3. Connection termination

<u>Connection Establishment</u>

- TCP works in full duplex mode. For this, each party must initialize communication and get approval from the other party before any data are transferred.
- The connection establishment in TCP is called **three-way handshaking**.
- Consider a client-server data transfer. Here the server TCP waits at passive open state to accept connections from clients.



Step 1:Client sends a SYN segment (ie, with only SYN flag set to 1).

- This segment is for synchronization of sequence numbers.
- The client chooses a random number as the first sequence number (ISN - initial sequence number) and sends this number to the server.
- A SYN segment is a **control segment** and **cannot carry data**, but it is supposed to carry an **imaginary byte** and hence consumes **one sequence number**

Step 2:The server sends a SYN + ACK segment (ie, with SYN and ACK flags set) ○ Dual purpose:

- It is a SYN segment for communication in the other direction (ie, sending the ISN).
- The server also acknowledges the receipt of the SYN segment from the client by setting the ACK flag and displaying the next sequence number it expects to receive from the client.
- Also send the receiving window size rwnd to be used by the client.
- This segment also does not carry data but consumes one sequence number.

Step 3:The client sends an ACK segment (ie, with ACK flags set)

- To acknowledge the receipt of the second segment.
- If there is no data in the ACK segment, it will not consume any sequence number.

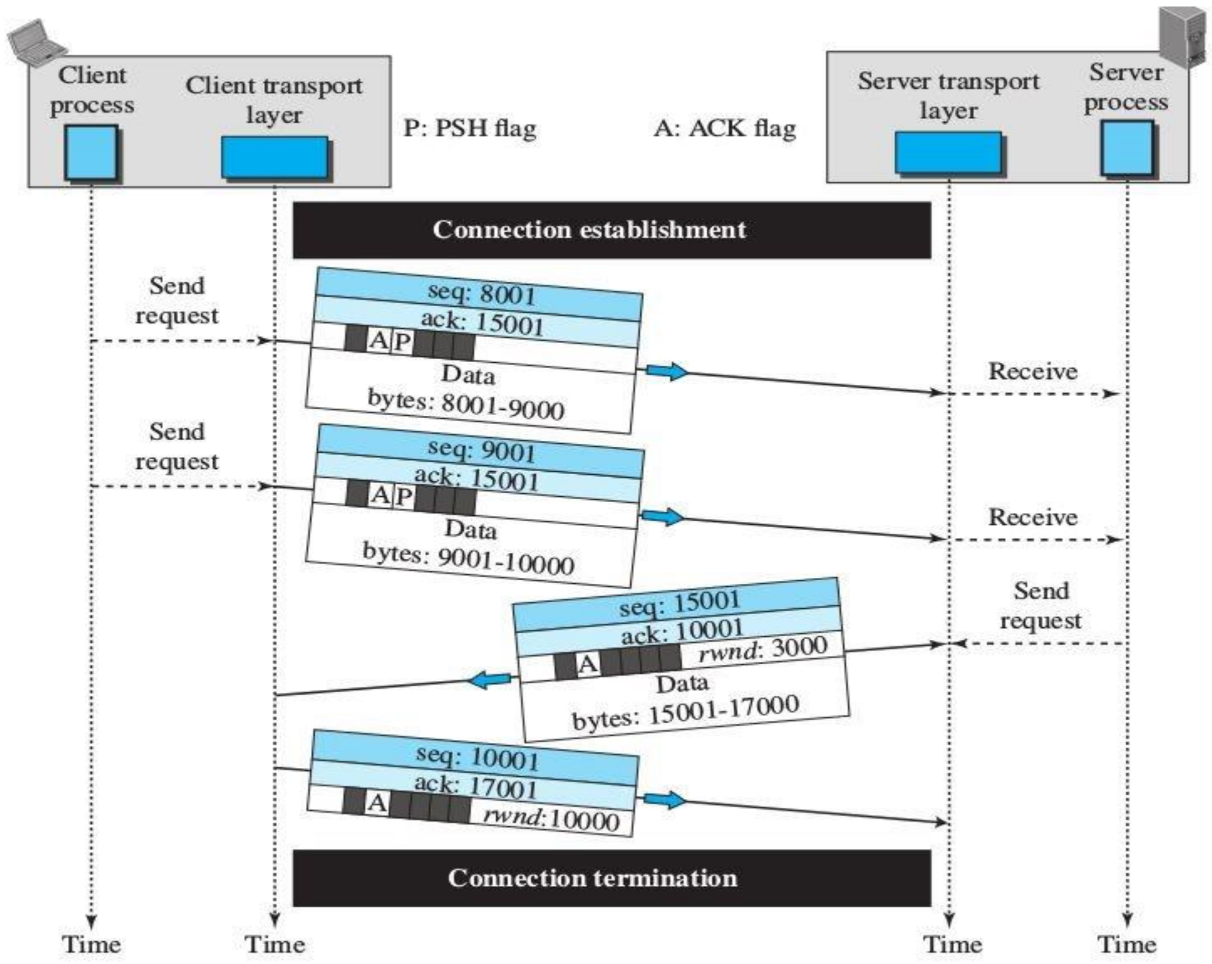
- In most cases, the ACK segment is transmitted along with the first chunk of data from the client. In this case, the segment consumes as many sequence numbers as the number of data bytes.

SYN Flooding Attack

- A type of Denial-of-Service (DoS) attack.
- Generally, when a server receives an active open request from a client, it allocates resources for a connection; such as setting TCP connection tables ('Transfer Control Blocks'- TCB) and setting the timers.
- The server then sends SYN + ACK segments to the clients.
- The attackers use this property.
- They send multiple SYN requests to the server using fake client IPs.
- The server arranges resources and sends SYN + ACK to the fake client IPs and waits for the ACK which will never arrive.
- Some servers resends the SYN + ACK again and again after waiting for some time.
- The table becomes exhausted for some time rejecting the connection requests from original clients.

Data Transfer

➢ After connection establishment, bidirectional data transfer with piggybacking happens between the client and the server.



Pushing Data – Use of PSH flag

- Generally, the stream of data from the application layer process is stored in the transport layer buffer and the TCP decides the segment size to send.
- The receiver TCP also stores the segments in buffers and delivers them to the application program when the application program is ready **or** when it is convenient for the receiving TCP.
- This type of flexibility increases the efficiency of TCP.
- But many times the application layer on one side communicates interactively with another one on the other side and thus immediate responses are required. Delayed transmission or delivery may not be admissible. PSH is used in such situation.
- The application layer process request the *push* operation; the sender TCP makes segments from available data without waiting for the buffer to be filled.
- Then set the PSH flag.

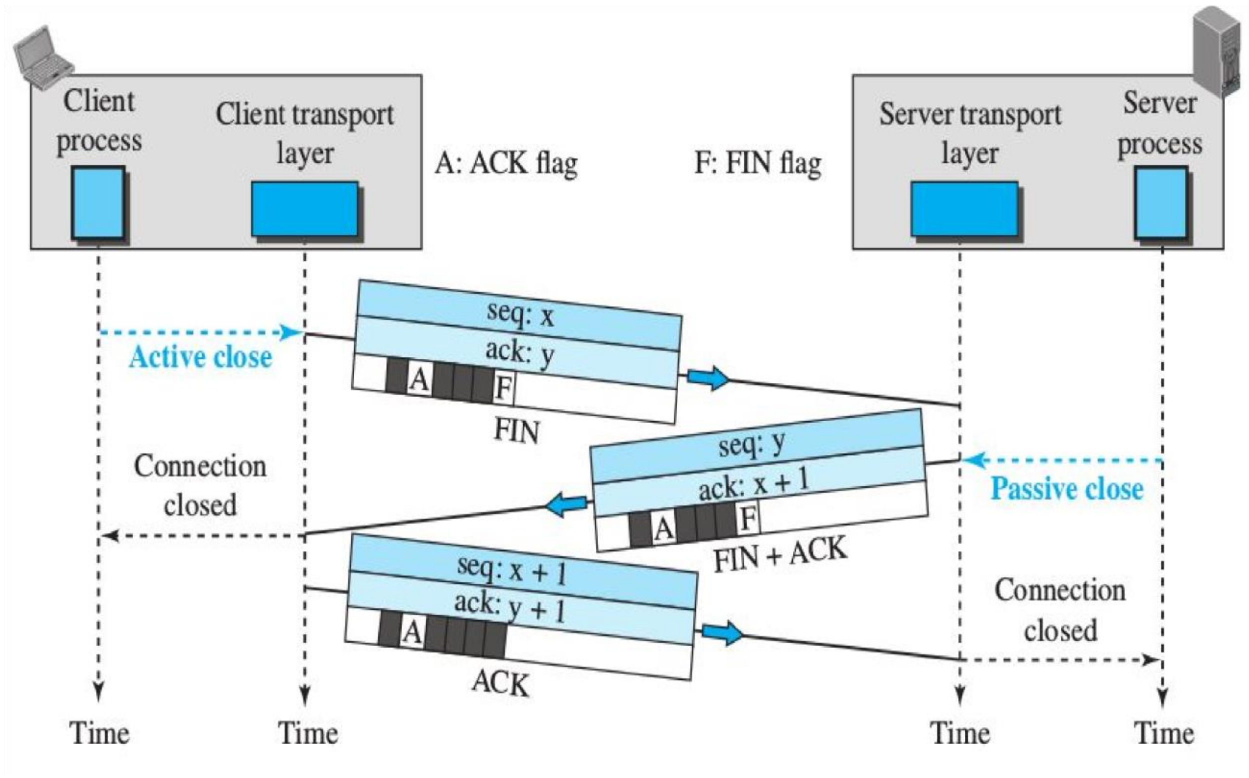
- At the receiver, the PSH flag tells the TCP that this segment need to be transferred to the application layer without waiting for the buffer to be filled.

Urgent Data

- Sometimes some bytes in the segment are to be treated as most urgent or to be in a special way by the receiving application process and thus to be processed first.
- The URG bit and the Urgent Pointer (16-bits) are used for this.
- If n bytes in the segment are to be treated urgently, they are placed at the start of the payload (data) and the Urgent Pointer is set to the position of the last urgent data.
- TCP's urgent data is neither a priority service nor an out-of-band data service. It is to inform the application layer program that some part of the data is important and to be treated specially.

Connection Termination

- Either client or server can close the connection even though the connection establishment is initiated by the client.
- There are two options for connection termination:
    - *Three-way handshaking* (most common)
    - *Four-way handshaking* with a half-close option.



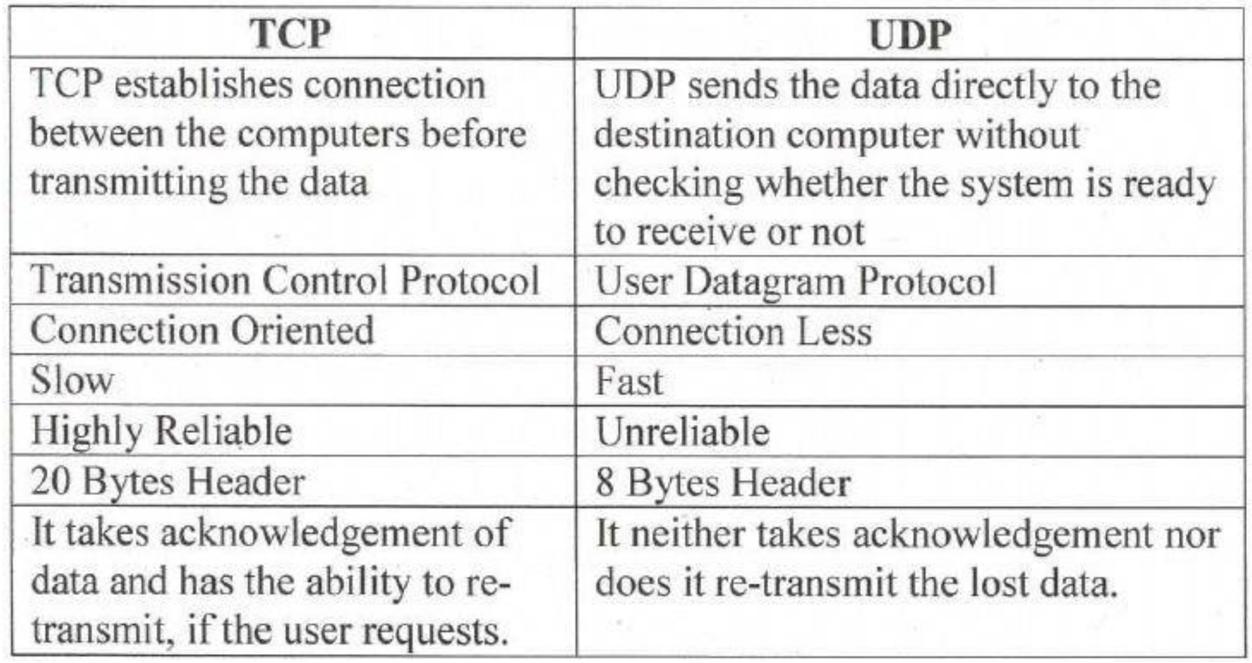
<u>Three-way handshaking</u> (Say, client starts termination)

- Client sends a FIN segment. It can be along with the last data segment. If it is not, it consumes one sequence number.
- Server sends a FIN+ACK segment containing Ack for the first message along with its FIN request. This segment can also contain the last chunk of data from the server. If it does not carry data, it consumes only one sequence number.
- Client sends the last segment, an ACK segment, for the second segment (FIN). It does not consume any sequence number.

<u>Four-way handshaking with Half-Close</u>

- One end can stop sending data while still receiving data. This is called a half-close.
- Either the server or the client can issue a half-close request.
- This can occur when one device need to get a lot of data and does not have anything to send.
- Eg: sorting.
- It can occur when the server needs all the data before processing can begin.

- After half-closing the connection, data can travel from the server to the client and acknowledgments can travel from the client to the server. The client cannot send any more data to the server.



TCP	UDP
TCP establishes connection between the computers before transmitting the data	UDP sends the data directly to the destination computer without checking whether the system is ready to receive or not
Transmission Control Protocol	User Datagram Protocol
Connection Oriented	Connection Less
Slow	Fast
Highly Reliable	Unreliable
20 Bytes Header	8 Bytes Header
It takes acknowledgement of data and has the ability to re-transmit, if the user requests.	It neither takes acknowledgement nor does it re-transmit the lost data.

<u>TCP RETRANSMISSION</u>

TCP Retransmission refers to the process of resending a packet of data after it is not acknowledged by the receiving end within a specific timeframe. It happens when there's packet loss or corruption in a network connection, which can occur due to various reasons such as congestion, faulty router or switch settings, poor signal quality, etc. When this happens, TCP protocol sends duplicate copies of packets until all the missing data has been received by the receiver.

The Transmission Control Protocol (TCP) uses several mechanisms to recover lost packets during data transmission such as Retransmission Time Out(RTO), Fast Recovery(FR) etc. When an acknowledgement of receipt is not received within an expected time frame (RTO), TCP initiates a retransmission of the missing packet(s).

<u>Types of TCP Retransmission</u>

Fast Retransmit: The Speedy Solution

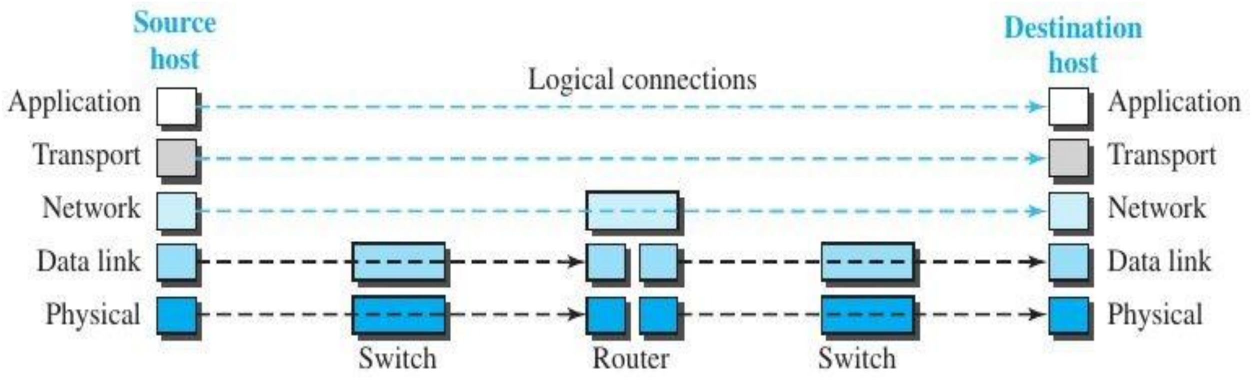
One type of TCP retransmission is known as fast retransmit. This technique involves quickly resending packets that are suspected to be lost in transmission before waiting for the retransmission timer to expire. Fast retransmit works by identifying the missing packet based on three duplicate acknowledgments from the receiving end. Once a sender receives three duplicate acknowledgments, it assumes that the packet has been lost and promptly resends it without waiting for a timeout.

Timeout–based Retransmission: Safety First

Another type of TCP retransmission is timeout–based retransmission, which waits for a predetermined amount of time before assuming that a packet has been lost and initiating a resend. With this technique, when an acknowledgment is not received within a certain time limit (usually set by the sender), then TCP assumes that either the network connection has failed or one or more packets have been dropped along the way. It then triggers a resend of all data sent since the last acknowledged packet.

APPLICATION LAYER

- Provides services to the user.
- Communication is provided using a logical connection between the source and destination application layers



- Highest layer in the protocol suite.
- Do not provide service to any layers, but receive service from transport layer.
- Hence new protocols can be easily added.
- All protocols in the lower four layers are standardized, but application layer protocols can be standardized or non-standardized.
- Standardized:
  - can be used anywhere in the internet.
  - provides a standard set of services.
- Non-standardized:
  - created for proprietary use in some offices for private use.
  - use the standard services provided by transport layer.

Application-Layer Paradigms

- Two types of paradigms: client-server (traditional) and peer-to-peer.

Client-Server

- An application program, called the server process, runs continuously, waiting for another application program, called the client process, to make a connection through the Internet and ask for service.
- Server processes do some specific services; but clients can ask for a number of services to different servers.
- Many clients can ask for a service to a single server. Hence the server will be loaded heavily and thus need a powerful computer to work
- Eg: WWW and e.mail

Peer-to-Peer (P2P)

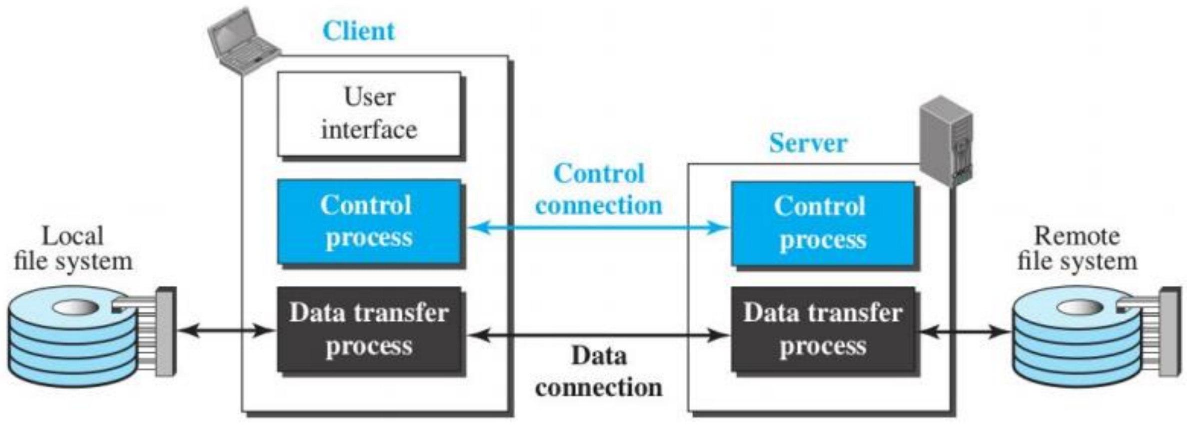
- No server process running all the time and waiting for the client processes to connect.
- One device provide service to the other, and vice versa, and both can be at the same time. ie, *responsibility is shared between peers*.
- Eg: BitTorrent, IPTV, Internet Telephony, Skype

Mixed Paradigm

- Combining the advantages of both.

FILE TRANSFER PROTOCOL (FTP)

- Used to transfer files from one host to another.
- File transfer means retrieving a file (server to client), storing a file (client to server), or directory listing (server to client).
- Uses TCP/IP.
- Transfers files even though both hosts have different directory structures, naming conventions, data representation.
- The basic model for FTP is;



- The client has three components: the user interface, the client control process, and the client data transfer process.
- The server has two components: the server control process and the server data transfer process.
- The control connection is made between the control processes. The data connection is made between the data transfer processes.
- When a user starts an FTP session, the control connection opens.
- Control connection remains connected during the entire interactive FTP session. Data connection is opened and then closed for each file transfer activity.
- TCP port 21 is used for the control connection, and port 20 is used for the data connection.

<u>Control Connection</u>

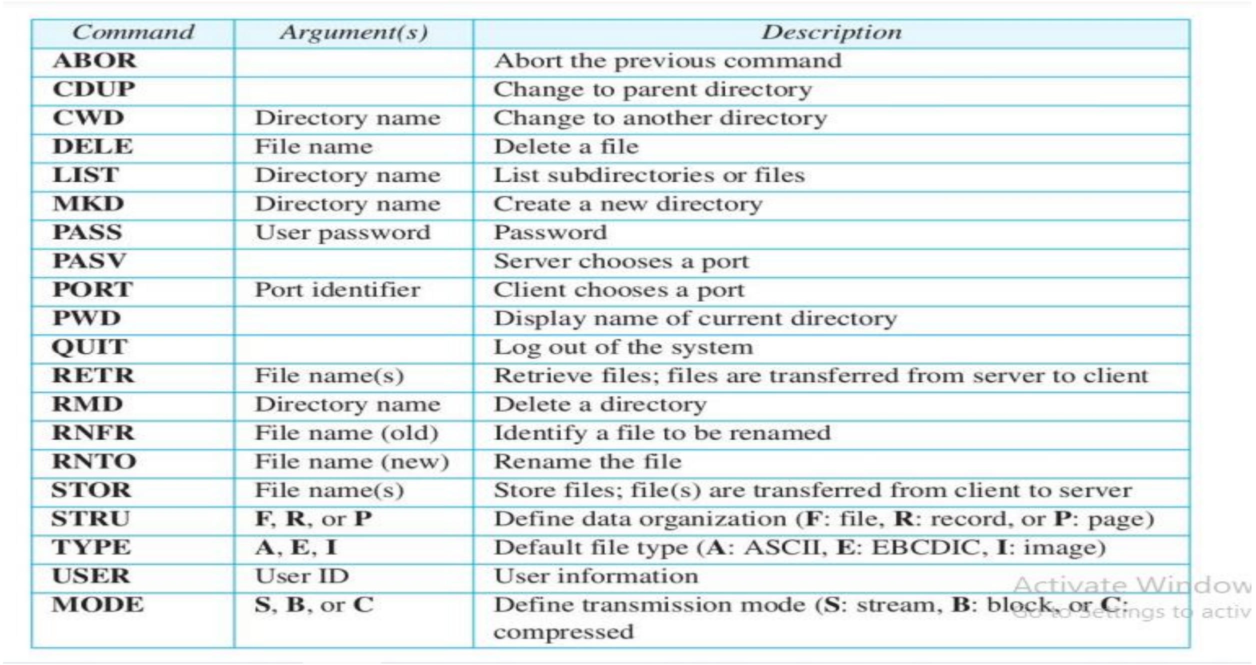
- Communication through commands and responses as ASCII characters.
- Each line is terminated with a two-character (carriage return and line feed) end-of-line token.
- Commands are sent from client to server as uppercase letters and responses from server to client as digits and letters.
- Some commands are:
- ■ ALLO (Allocate sufficient disk space to receive a file), DELE filename (delete a file), QUIT (log out) ○
    - Every FTP command generates one or more responses.
- ■ Response format:
- ■ Eg: 125 Data connection open, 230 User login OK

<u>Data Connection</u>

- To transfer data.
- Server uses its port 20.
- The client defines the type of file to be transferred, the structure of the data, and the transmission mode through control connection before every data connection is made.

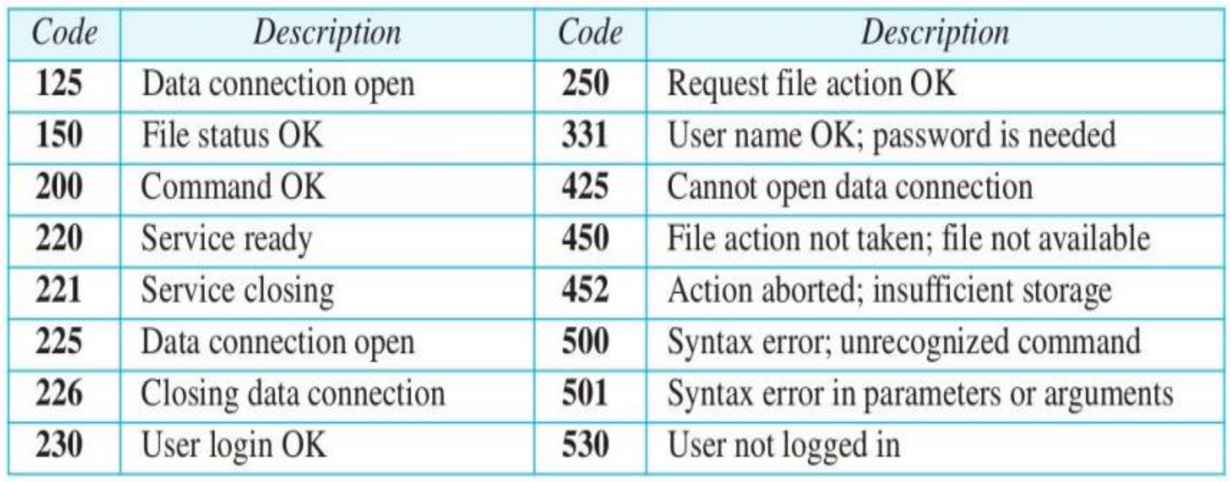
- The steps for data connection are;
- ■ The client issues a passive open request to the server from an ephemeral port.
- ■ Using the PORT command the client sends this port number to the server.
- ■ The server receives the port number and issues an active open using the well-known port 20 and the received ephemeral port number.

Some FTP commands are



Command	*Argument(s)*	*Description*
ABOR		Abort the previous command
CDUP		Change to parent directory
CWD	Directory name	Change to another directory
DELE	File name	Delete a file
LIST	Directory name	List subdirectories or files
MKD	Directory name	Create a new directory
PASS	User password	Password
PASV		Server chooses a port
PORT	Port identifier	Client chooses a port
PWD		Display name of current directory
QUIT		Log out of the system
RETR	File name(s)	Retrieve files; files are transferred from server to client
RMD	Directory name	Delete a directory
RNFR	File name (old)	Identify a file to be renamed
RNTO	File name (new)	Rename the file
STOR	File name(s)	Store files; file(s) are transferred from client to server
STRU	**F**, **R**, or **P**	Define data organization (**F**: file, **R**: record, or **P**: page)
TYPE	**A**, **E**, **I**	Default file type (**A**: ASCII, **E**: EBCDIC, **I**: image)
USER	User ID	User information
MODE	**S**, **B**, or **C**	Define transmission mode (**S**: stream, **B**: block, or **C**: compressed

Some FTP responses are



Code	*Description*	*Code*	*Description*
125	Data connection open	**250**	Request file action OK
150	File status OK	**331**	User name OK; password is needed
200	Command OK	**425**	Cannot open data connection
220	Service ready	**450**	File action not taken; file not available
221	Service closing	**452**	Action aborted; insufficient storage
225	Data connection open	**500**	Syntax error; unrecognized command
226	Closing data connection	**501**	Syntax error in parameters or arguments
230	User login OK	**530**	User not logged in

<u>**WORLD WIDE WEB (ABBREVIATED WWW OR WEB)**</u>

- A standard client-server protocol.

- Proposed by Tim Berners-Lee.
- Repository of information in which the documents, called web pages, are distributed all over the world and related documents are linked together.
- Linking of web pages was achieved using a concept called hypertext, now hypermedia.
- Distributed client-server service, in which a client using a browser can access a service using a server.
- Service is distributed over many locations called sites.
- Each site holds one or more web pages. Each web page can contain some links to other web pages in the same or other sites.
- Thus a web page can be **simple** (no links) or **composite** (contains at least a link).
- Each web page is a file with a name and address.
- Accessing a web page is done by a web client from a web server

Web Client (Browser)

- Consists of three parts: **a controller, client protocols,** and **interpreters.**
- Controller receives input from the keyboard or the mouse and uses the client programs (protocols) to access the document.
- Controller uses one of the interpreters to display the document on the screen.
- Eg: Firefox, Chrome, Opera

Web Server

- The web page is stored at the server.
- Each time a request arrives, the corresponding document is sent to the client.
- Efficiency can be improved by caching, multithreading or multiprocessing.
- Eg: Apache and Microsoft Internet Information Server (IIS).

Uniform Resource Locator (URL)

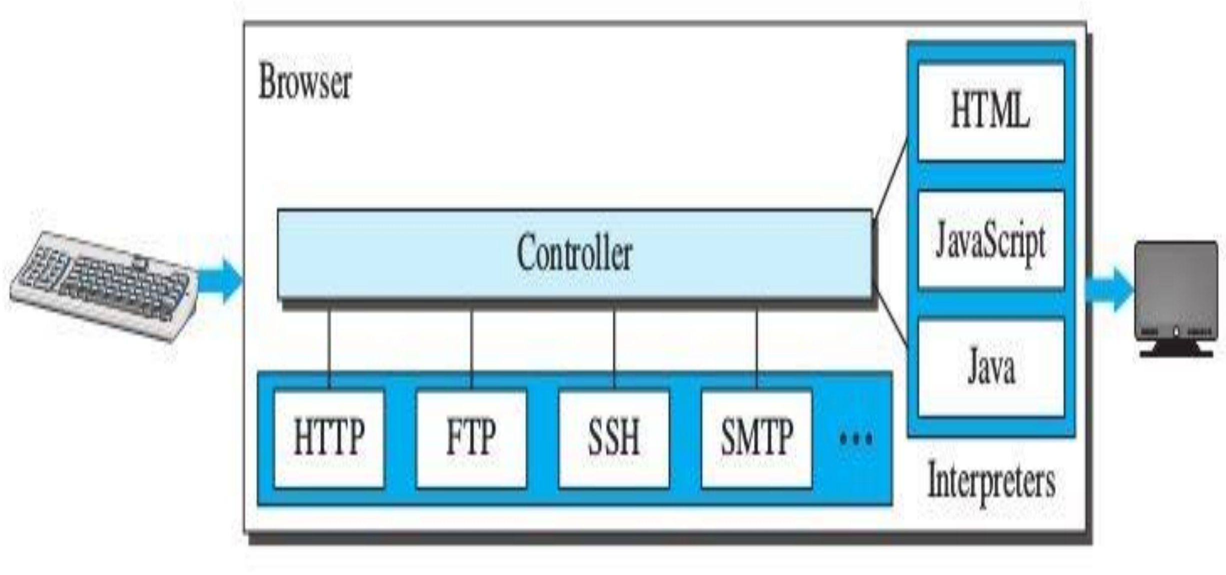
- A unique identifier for a web page.
- Combination of the **Protocol** to be used by the browser, **Host** address of the server, **Port** to be accessed and the actual **Path** of the file to be accessed.
  - Protocol: http, ftp
  - Host: IP address (eg: 93.184.216.34) or domain name (eg: example.com)

■ Port: A 16-bit integer. Normally pre-defined for client-server application. Eg. HTTP port number is 80.

■ Path: The location and the name of the file in the underlying operating system.

○ protocol://host/path Eg: http://www.tekerala.org/staff_login.php

Eg: http://www.example.com:8080/samples/index.html



WWW Architecture

WWW - Web Documents

Classified into three: static, dynamic and active.

Static Documents:

- Fixed-content documents that are created and stored in a server.
- Client can get a copy only.
- The document content in the server can be changed, but the user cannot change them.
- The user can see the document copy in the browser.
- Some languages used for static document creation are: HyperText Markup Language (HTML), Extensible Markup Language (XML), Extensible Style Language (XSL), and Extensible Hypertext Markup Language (XHTML).

Dynamic Documents:

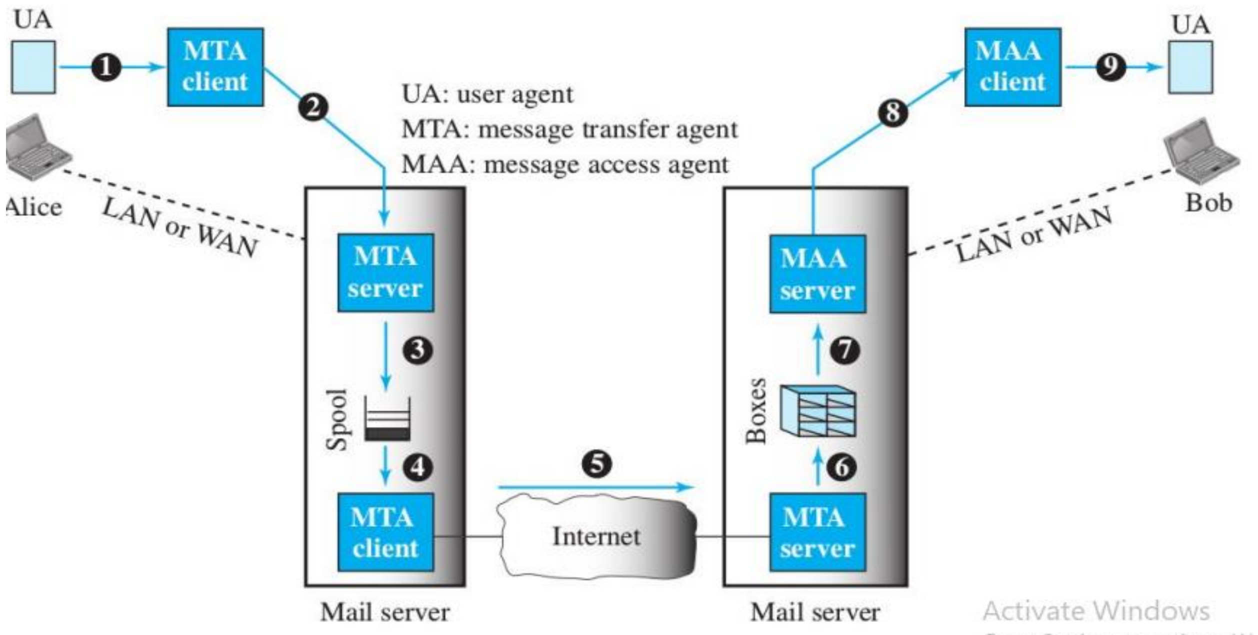
- Created by a web server whenever a browser requests the document.
- Whenever a request arrives from the client, the web server runs an application program or a script that freshly creates the dynamic document every time.
- Some languages for this are Java Server Pages (JSP), Active Server Pages (ASP), etc

Active Documents:

- Documents containing scripts that works at the client side.
- Such as a program that creates animated graphics on the screen or a program that interacts with the user.
- When a browser requests an active document, the server sends a copy of the document or a script.
- This can be done using Java applet, javascript, etc

ELECTRONIC MAIL

- E-mail is a one-way transaction.
- If A sends an e-mail to B, it is a one-way transaction. If B replies to A that is another one-way transaction.
- Intermediary servers always keep running to transfer the emails; the users are just clients
- Consider a common scenario as in the figure below. Here Alice is sending a mail to Bob, both belong to two different mail domains.



- In the common scenario, the sender and the receiver of the e-mail, Alice and Bob respectively, are connected via a LAN or a WAN to two mail servers.
- The administrator has created one mailbox for each user where the received messages are stored.
- A mailbox is part of a server hard drive, a special file with permission restrictions. Only the owner of the mailbox has access to it.
- The administrator has also created a queue (spool) to store messages waiting to be sent.
- To send an email, Alice and Bob use three different agents: a user agent (UA), a message transfer agent (MTA), and a message access agent (MAA).
    1. UA: program to prepare the message and send it to the sender's mail server. Receiver UA interprets the mail. It can be CLI (Eg: elm, pine) or GUI (Eg: outlook) based.
    2. MTA: A push program to transfer the email from one point to the other.
    3. MAA: A pull program to get the email from one point to the other.
- The given electronic mail system needs two UAs, two pairs of MTAs (2 client and 2 server), and a pair of MAAs (client and server).

Sending Mail

- A user creates a mail through a UA.
- An email has an envelope and a message.

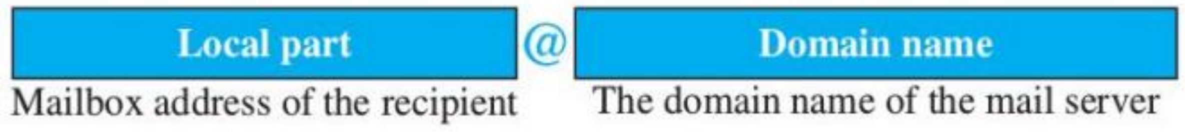
- The envelope usually contains the sender address, the receiver address, and other information.
- The message contains the header and the body.
- The header of the message defines the sender, the receiver, the subject of the message, and some other information.
- The body of the message contains the actual information to be read by the recipient.

<u>Receiving Mail</u>

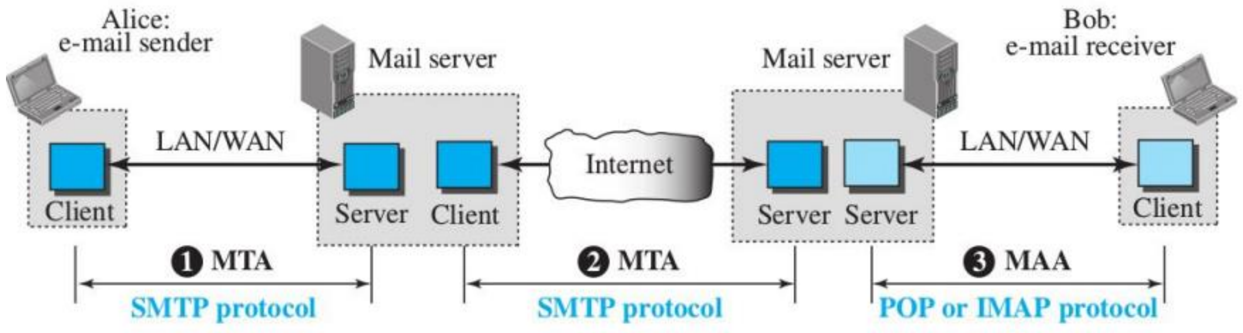
- The user agent is triggered by the user (or a timer).
- If a user has mail, the UA informs the user with a notice. The user can select any of the messages and display its contents on the screen.

<u>Addresses</u>

- Consists of two parts: a local part (called user mailbox) and a domain name (called mail servers or exchangers), separated by an @ sign.
- The local part defines the name of a special file, called the user mailbox, where all the mail received for a user is stored for retrieval by the message access agent

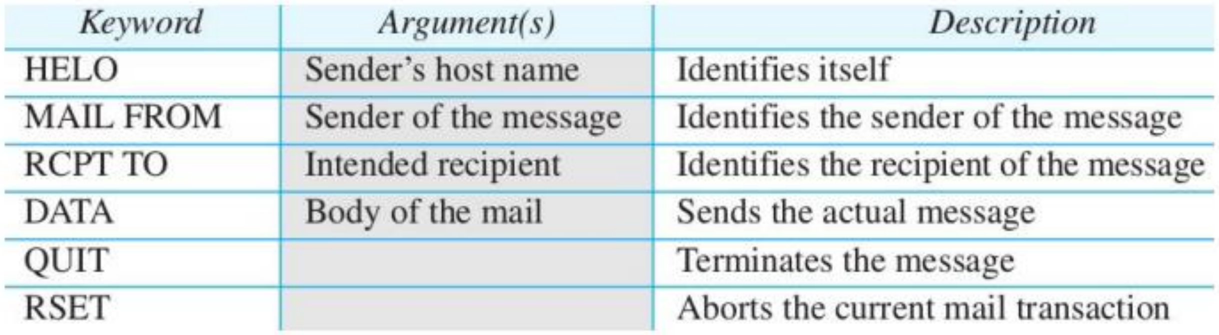

Local part
@
Domain name
Mailbox address of the recipient
The domain name of the mail server

<u>Protocols used in electronic mail</u>



Message Transfer Agent: SMTP (Simple Mail Transfer Protocol)

- The formal protocol that defines the MTA client and server in the Internet.
- SMTP is used two times, between the sender and the sender's mail server and between the two mail servers.
- A push protocol, it pushes the message from the client to the server.
- Done by commands and responses.
- **Command format**: Keyword: argument(s). Sent from client to server.



Keyword	*Argument(s)*	*Description*
HELO	Sender's host name	Identifies itself
MAIL FROM	Sender of the message	Identifies the sender of the message
RCPT TO	Intended recipient	Identifies the recipient of the message
DATA	Body of the mail	Sends the actual message
QUIT		Terminates the message
RSET		Aborts the current mail transaction

<u>Responses:</u>

- ▶ A three-digit code that may be followed by additional textual information. Sent from the server to the client.
- ▶ Eg: 220 Service ready
  250 Request command completed
  450 Mailbox not available

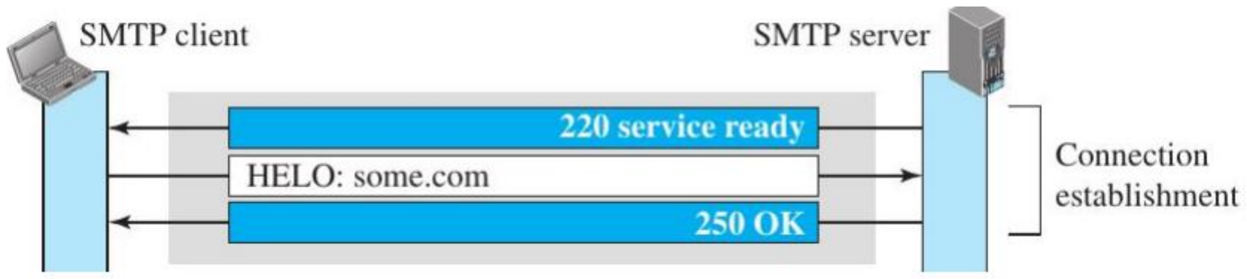
Mail Transfer Phases -Three phases:

1) Connection establishment
2) Mail transfer
3) Connection termination.

<u>Step1:-Connection Establishment:</u>

Client TCP connect to well-known port 25 of the server and does the following three steps:

1) If the server is ready it sends a code 220 (service ready), otherwise 421 (service not available).
2) The client sends the HELO message to identify itself, using its domain name address.
3) The server responds with code 250 (request command completed) or some other code depending on the situation.

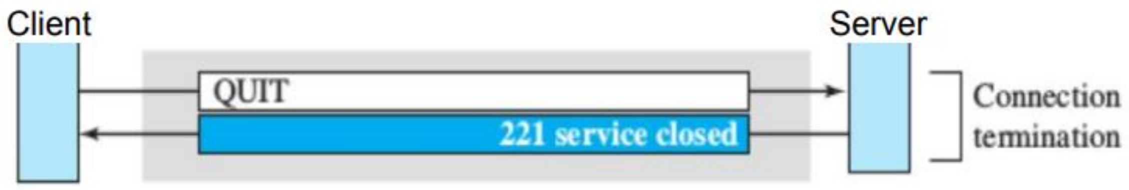


Step2:-Message Transfer: Eight steps

1) The client sends the MAIL FROM message to introduce the sender of the message. It includes the mail address of the sender (mailbox and domain name). This step is needed to give the server the return mail address for returning errors and reporting messages.
2) The server responds with code 250 or some other appropriate code.
3) The client sends the RCPT TO (recipient) message, which includes the mail address of the recipient.
4) The server responds with code 250 or some other appropriate code
5) The client sends the DATA message to initialize the message transfer.
6) The server responds with code 354 (start mail input) or some other appropriate message.
7) The client sends the contents of the message in consecutive lines. Each line is terminated by a two-character end-of-line token (carriage return and line feed). The message is terminated by a line containing just one period (.).
8) The server responds with code 250 (OK) or some other appropriate code.

Connection Termination: Two steps:

1) The client sends the QUIT command.
2) The server responds with code 221 or some other appropriate code.



Message Access Agent: POP and IMAP

- The third stage of the mail delivery needs a pull protocol; the client must pull messages from the server.
- Two such protocols are Post Office Protocol, version 3 (POP3) and Internet Mail Access Protocol, version 4 (IMAP4).

Post Office Protocol, version 3 (POP3)

- Simple but limited in functionality.
- The client POP3 software is installed on the recipient computer; the server POP3 software is installed on the mail server.
- Mail access starts with the client when the user needs to download its e-mail from the mailbox on the mail server.
- The client opens a connection to the server on TCP port 110.
- It then sends its username and password to access the mailbox. The user can then list and retrieve the mail messages, one by one.
- POP3 has two modes: the delete mode and the keep mode.
- In the delete mode, the mail is deleted from the mailbox after each retrieval.
- Used when the user is working at her permanent computer and can save and organize the received mail after reading or replying.
- In the keep mode, the mail remains in the mailbox after retrieval.
- Used when the user accesses her mail away from her primary computer

Internet Mail Access Protocol, version 4 (IMAP4)

- More feature rich, powerful and complex than POP3.
- Extra functions;
  - A user can check the email header prior to downloading.
  - A user can search the contents of the email for a specific string of characters prior to downloading.
  - A user can partially download e-mail.
- This is especially useful if bandwidth is limited and the email contains multimedia with high bandwidth requirements.
- A user can create, delete, or rename mailboxes on the mail server.
- A user can create a hierarchy of mailboxes in a folder for email storage.

MULTIPURPOSE INTERNET MAIL EXTENSION (MIME) PROTOCOL

MIME stands for Multipurpose Internet Mail Extensions. It is used to extend the capabilities of Internet e-mail protocols such as SMTP. The MIME protocol allows the users to exchange various types of digital content such as pictures, audio, video, and various types of documents and files in the e-mail. MIME was created in 1991 by a computer scientist named Nathan Borenstein at a company called Bell Communications.

MIME is an e-mail extension protocol, i.e., it does not operate independently, but it helps to extend the capabilities of e-mail in collaboration with other protocols such as SMTP. Since MIME was able to transfer only text written file in a limited size English language with the help of the internet. At present, it is used by almost all e-mail related service companies such as Gmail, Yahoo-mail, Hotmail.

MIME protocol is used to transfer e-mail in the computer network for the following reasons:

1. The MIME protocol supports multiple languages in e-mail, such as Hindi, French, Japanese, Chinese, etc.
2. Simple protocols can reject mail that exceeds a certain size, but there is no word limit in MIME.
3. Images, audio, and video cannot be sent using simple e-mail protocols such as SMTP. These require MIME protocol.
4. Many times, emails are designed using code such as HTML and CSS, they are mainly used by companies for marketing their product. This type of code uses MIME to send email created from HTML and CSS.

MIME Header

MIME adds five additional fields to the header portion of the actual e-mail to extend the properties of the simple email protocol. These fields are as follows:

1. MIME Version
2. Content Type
3. Content Type Encoding
4. Content Id
5. Content description

1. MIME Version

It defines the version of the MIME protocol. This header usually has a parameter value 1.0, indicating that the message is formatted using MIME.

2. Content Type

It describes the type and subtype of information to be sent in the message. These messages can be of many types such as Text, Image, Audio, Video, and they also have many subtypes such that the subtype of the image can be png or jpeg. Similarly, the subtype of Video can be WEBM, MP4 etc.

3. Content Type Encoding

In this field, it is told which method has been used to convert mail information into ASCII or Binary number, such as 7-bit encoding, 8-bit encoding, etc.

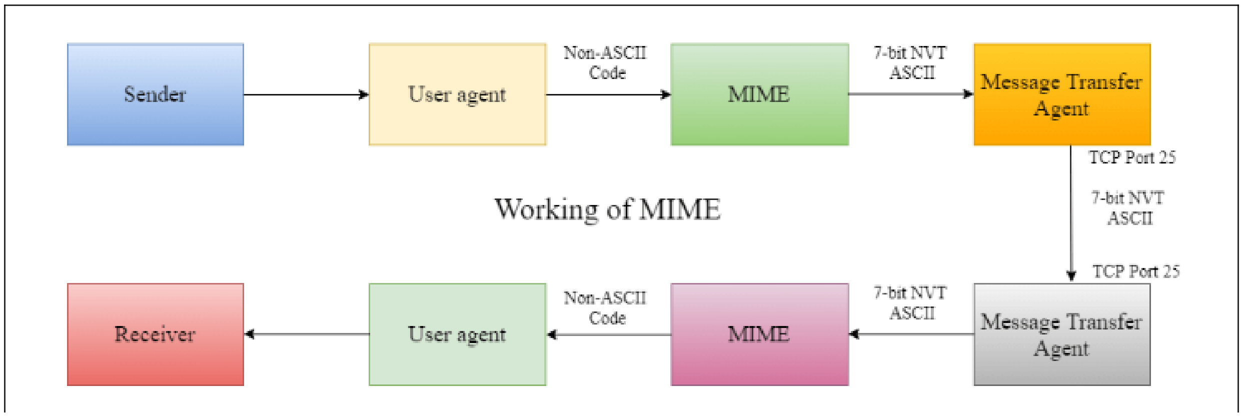
4. Content Id

In this field, a unique "Content Id" number is appended to all email messages so that they can be uniquely identified.

5. Content description

This field contains a brief description of the content within the email. This means that information about whatever is being sent in the mail is clearly in the "Content Description". This field also provides the information of name, creation date, and modification date of the file.

Working diagram of MIME Protocol

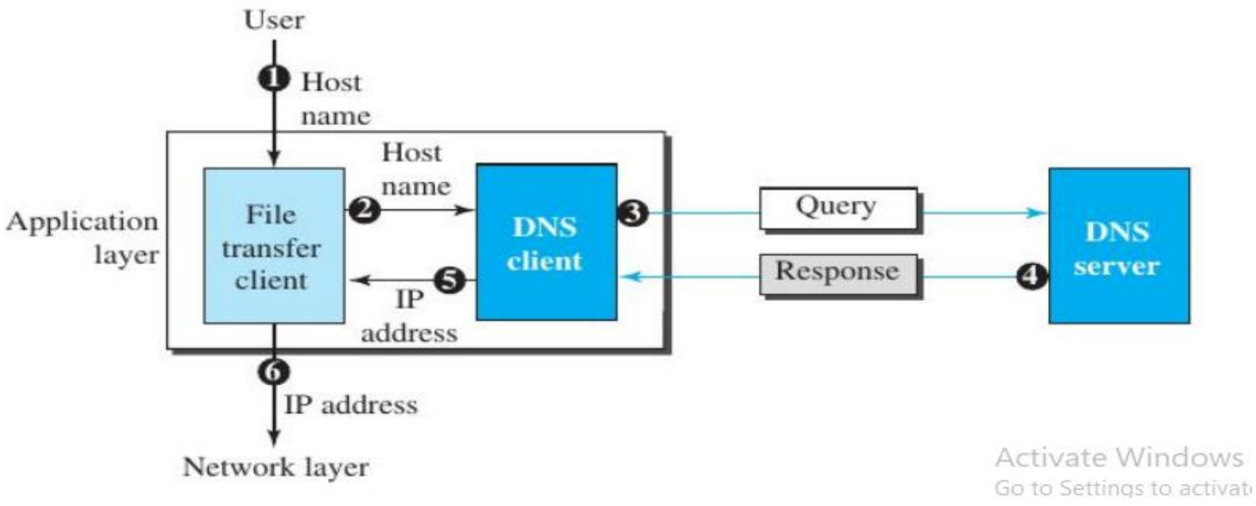


<u>DOMAIN NAME SYSTEM (DNS)</u>

- Machines can understand only IP (numeric) address while people can remember only names.
- Hence IP addresses are mapped to some names and the whole such information in the network are distributed among many computers in the world.

- Any host that needs the mapping from names to IP address can contact the nearby computer that holds this information.
- This method is called the DOMAIN NAME SYSTEM (DNS) and the computer that holds such mapping information is called a DNS server.
- The DNS client sends the name to the DNS server which returns the corresponding IP address.

Consider a file transfer scenario where the user gives a host name instead of IP address



<u>Name Space</u>

- It is a collection of names and addresses; both are unique.
- Names can be organized in two ways: flat or hierarchical.
- Flat Name Space: A name is assigned to an address, one by one. It can't be used for large number of names-address combinations.
- Hierarchical Name Space: Each name is made of several parts.
- ■ Eg: kerala.bsnl.co.in, <u>www.indiana.edu</u>

<u>A domain</u> is a subtree of the domain name space

- For efficiency, domains are stored in different places called domain servers.
- Since the complete domain hierarchy can't be stored on a single server, it is divided among different servers; each one is said to have an authority of a zone.
- A root server is a server whose zone consists of the entire tree.

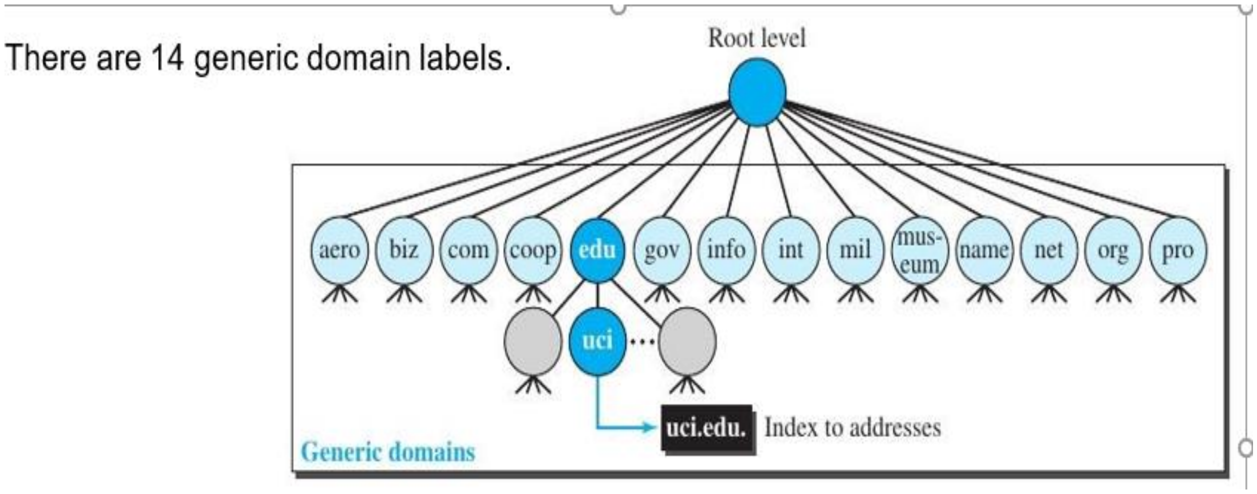
<u>DNS defines two types of servers:</u>

- primary and secondary.

- A primary server is a server that stores a file about the zone for which it is an authority and is responsible for creating, maintaining, and updating the zone file. It stores the zone file on a local disk.
- A secondary server is a server that transfers the complete information about a zone from another server (primary or secondary) and stores the file on its local disk. The secondary server neither creates nor updates the zone files

Generic Domains

- Based on generic behaviour.
- Each node in the tree defines a domain, which is an index to the domain name space database.
- There are 14 generic domain labels.



SIMPLE NETWORK MANAGEMENT PROTOCOL (SNMP)

Simple Network Management Protocol (SNMP) is a widely used protocol for network management that provides a standardized framework for monitoring and managing network devices such as routers, switches, servers, and printers. It operates within the application layer of the Internet protocol suite and allows network administrators to manage network performance, find and solve network problems, and plan for network growth. NMP is an application layer protocol that uses UDP port number 161/162

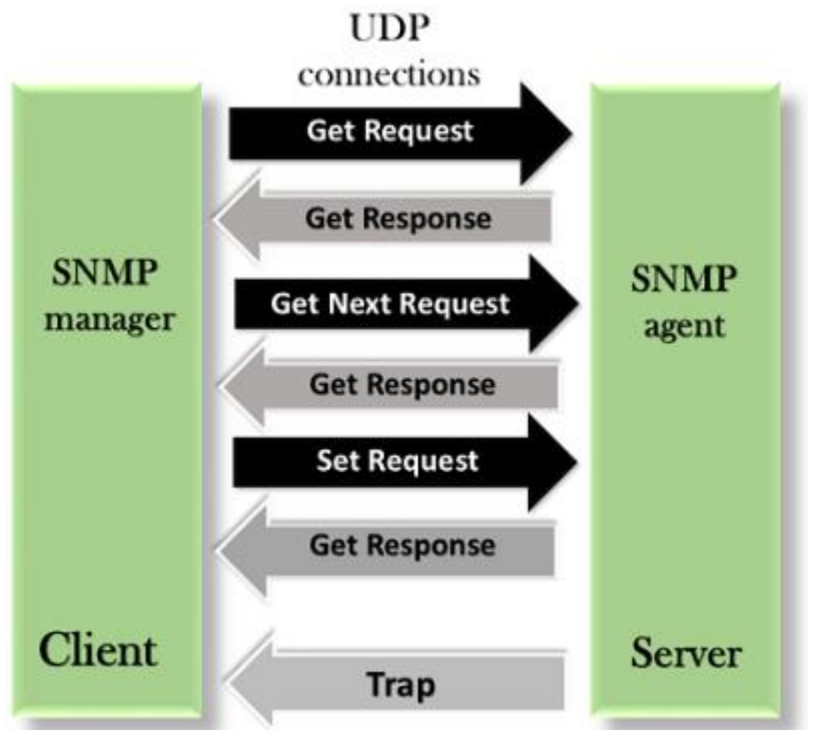
Components of SNMP

There are mainly three main components of SNMP

- **SNMP Manager:** It is a centralized system used to monitor the network. It is also known as a Network Management Station (NMS). A router that runs the SNMP server program is called an agent, while a host that runs the SNMP client program is called a manager.
- **SNMP agent:** It is a software management software module installed on a managed device. The manager accesses the values stored in the database, whereas the agent maintains the information in the database. To ascertain if the router is congested or not, for instance, a manager can examine the relevant variables that a router stores, such as the quantity of packets received and transmitted.
- **Management Information Base:** MIB consists of information on resources that are to be managed. This information is organized hierarchically. It consists of objects instances which are essentially variables. A MIB, or collection of all the objects under management by the manager, is unique to each agent. System, interface, address translation, IP, udp, and egp , icmp, tcp are the eight categories that make up MIB. The mib object is home to these groups.

SNMP Messages

- **GetRequest** : It is simply used to retrieve data from SNMP agents. In response to this, the SNMP agent responds with the requested value through a response message.
- **GetNextRequest :** To get the value of a variable, the manager sends the agent the GetNextRequest message. The values of the entries in a table are retrieved using this kind of communication. The manager won’t be able to access the values if it doesn’t know the entries’ indices. The GetNextRequest message is used to define an object in certain circumstances.
- **SetRequest :** It is used by the SNMP manager to set the value of an object instance on the SNMP agent.
- **Response :** When sent in response to the Set message, it will contain the newly set value as confirmation that the value has been set.
- **Trap :** These are the message sent by the agent without being requested by the manager. It is sent when a fault has occurred.
- **InformRequest :** It was added to SNMPv2c and is used to determine if the manager has received the trap message or not. It is the same as a trap but adds an acknowledgement that the trap doesn’t provide.



How SNMP works

SNMP software agents on network devices and services communicate with a network management system to relay status information and configuration changes. The NMS provides a single interface from which administrators can issue batch commands and receive automatic alerts.

SNMP relies on the concept of an MIB to organize how information about device metrics gets exchanged. The MIB is a formal description of a network device's components and status information.

MIBs can be created for any network device in the Internet of Things (IoT), including IP video cameras, vehicles, industrial equipment and medical equipment. In addition to hardware, SNMP can be used to monitor services such as Dynamic Host Configuration Protocol (DHCP).

SNMP uses a blend of pull and push communications between network devices and the network management system. The SNMP agent, which resides with the MIB on a network device, constantly collects status information but will only push information to the NMS upon request or when some aspect of the network crosses a pre-defined threshold known as a trap.

Trap messages are typically sent to the management server when something significant, such as a serious error condition, occurs.

SNMP also includes an inform message type that enables a network monitoring tool to acknowledge messages from a device. Inform messages enable the agent to reset a triggered alert. Network management tools can also use a set message to make changes to a network device through the SNMP agent. This capability enables the network manager to make change device configurations in response to new network events.