GRAPHIC ERA HILL UNIVERSITY BHIMTAL
COMPUTER NETWORK-1(TCS604)
CSE 6TH SEM (UNIT-2 NOTES)

BitTorrent(P2P FILE DISTRIBUTION)

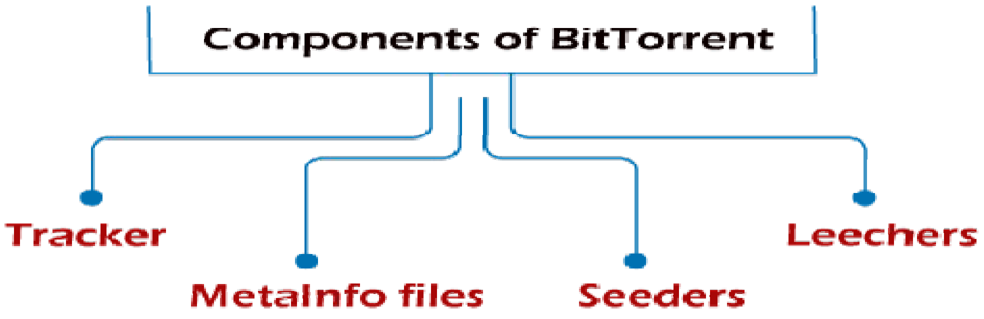
BitTorrent is a communication protocol for ***peer-to-peer*** file sharing (P2P), enabling users to distribute data and electronic files over the Internet in a decentralized manner.

BitTorrent is one of the most common protocols for transferring large files, such as digital video files containing TV shows and video clips or digital audio files containing songs.

The BitTorrent protocol can be used to reduce the server and network impact of distributing large files. Rather than downloading a file from a single source server, the BitTorrent protocol allows users to join a "***swarm***" of hosts to upload or download from each other simultaneously. Several basic computers can replace large servers while efficiently distributing files to many recipients using the BitTorrent protocol.

Components of BitTorrent

BitTorrent distributes a file by separated it into pieces and distributing the pieces amongst its peers. BitTorrent usually consists of the following components:



1. Tracker

A BitTorrent tracker is server software that coordinates the transfer of files among users. The tracker does not contain a copy of the file and helps peers discover each other.

The tracker and the client exchange information using a simple protocol on top of HTTP. The clients inform the tracker regarding the file they want to download, their IP, and port, and the tracker responds with a list of peers downloading the same file and their contact information. This list of peers that all share the same Torrent represents a ***swarm***. The tracker is necessary for peers to find each other and transfer data.

2. Metainfo file

The metainfo file is also called a torrent file and has a .torrent extension. This file mainly contains encoded information regarding the URL of the tracker, the name of the file, and hashes of the pieces of the file for verifying downloaded pieces of the file.

These torrent files are generally created using client software. A list of trackers and the original file are required to create this torrent file.

Once the file is created, it can be shared using regular email, file-sharing websites, etc.

3. Seeders

The original downloader is a peer with the whole file, also known as a seeder. The seeder must keep uploading the file until a complete copy has been distributed among the downloader. As long as there is a complete copy collectively present among the peers, the download will continue for all.

The developers providing the Linux iSO image who have the entire file will be called seeders.

4. Leechers

The peers without a complete copy of the file are known as leechers. Leechers will get the list of peers from the tracker, which has the pieces that the leecher requires. The leecher then downloads the required piece from one of those peers.

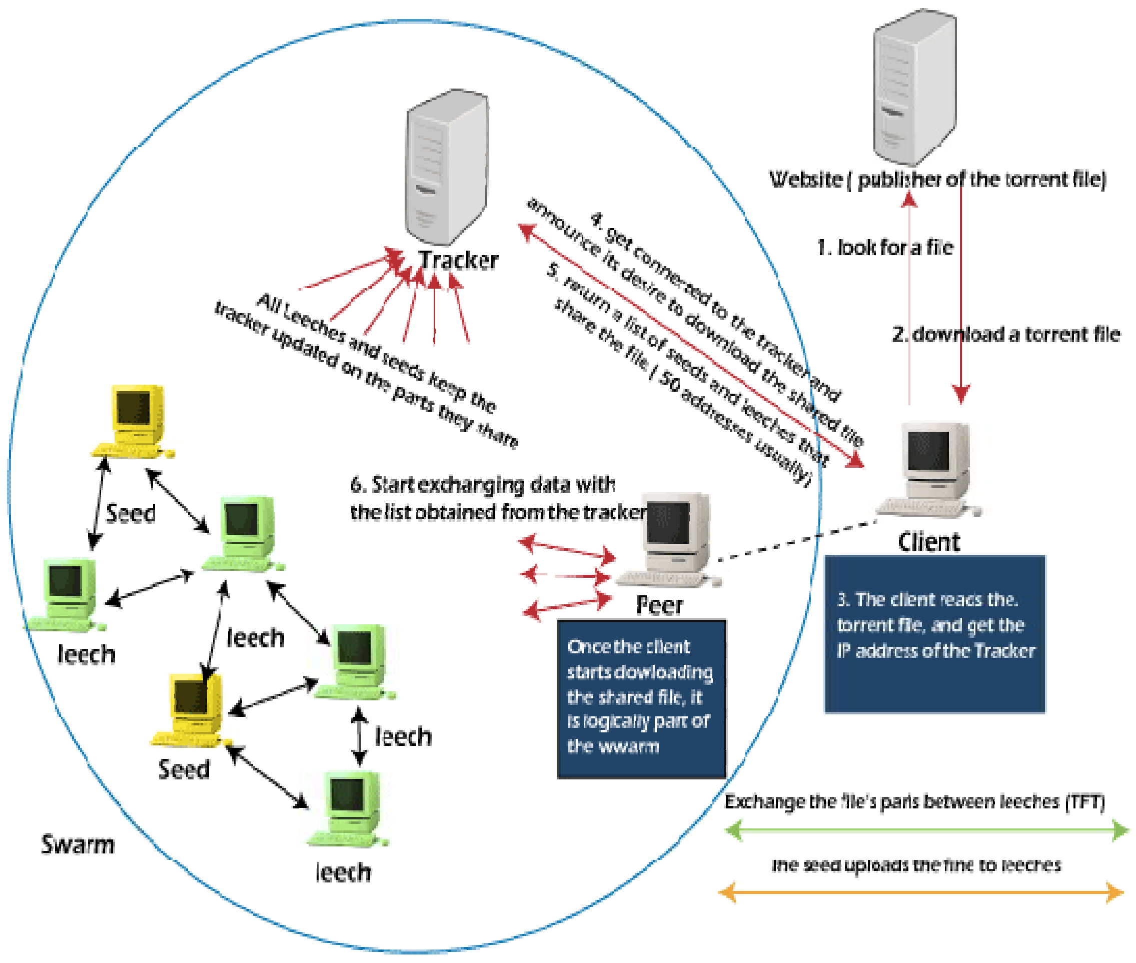
A leecher can also distribute the pieces that it has completed downloading even before it completes downloading the whole file. Once a Leecher has all the pieces, it is called a seeder. As a leecher receives the pieces, it validates them against the hashes present in the metainfo file.

Any user downloading the file through BitTorrent will be called a leecher. Once they have an entire file, they can be called seeders.

WORKING OF BIT TORRENT

Torrent doesn't depend on a centralized server for storing files. Instead, bits of data from individual large files are saved in participating computers (peers) in a network (swarm) to facilitate the file-sharing process.

A P2P communication protocol like BitTorrent breaks down the files into pieces. Then, it moves them from ***seeders*** to ***leechers*** via a torrent client (a separate program that reads all the information in the .torrent file and connects users to exchange data).


Website (publisher of the torrent file)
1. look for a file
2. download a torrent file
Tracker
4. get connected to the tracker and announce its desire to download the shared file
5. return a list of seeds and leeches that share the file (50 addresses usually)
All Leeches and seeds keep the tracker updated on the parts they share
6. Start excharging data with the list obtained from the tracker
Client
3. The client reads the torrent file, and get the IP address of the Tracker
Peer
Once the client starts dowloading the shared file, it is logically part of the wwarm
Seed
leech
leech
Seed
leech
leech
Swarm
Exchange the file's parts between leeches (TFT)
The seed uploads the fine to leeches

Anyone who wants the file uses a program called a ***BitTorrent client*** to request it from a seed. The client is sent one of the pieces and gets all the remaining pieces, over a while, from other people's computers through P2P communication. At any given moment, each computer is downloading some parts of the file from some of these peers and uploading other parts of the file to other peers. All the computers cooperating in this way at any time are called a ***swarm***.

A system of checks and balances described below is applied to make the torrent process a bit-perfect:

- A torrent file (.torrent) contains information telling users which computers are part of the file-sharing process. It may also provide some details on the files and folders that a user is downloading.
- The torrent client connects to a tracker, which holds the IP addresses of the devices in a swarm. The tracker forwards the IP addresses to all torrent clients to ensure all peers are connected.
- The torrent client starts the download. Once it receives sufficient bits of data, it also begins to upload the file for the benefit of other users.

Uses of BitTorrent

While torrent sites have become a hub for digital pirates and infringers, they can be helpful:

- For syncing large chunks of files and sharing media, you own the rights to.
- Social media giants like ***Facebook*** and ***Twitter*** use a similar protocol to upload large files to their servers to conserve bandwidth.
- A torrent client may also be integrated into a game to deploy software updates, as in Starcraft's case.
- Some government agencies also use torrents to share large images and documents to the public that could otherwise put a strain on their servers.

You can use Torrent in the following easy steps, such as:

- **Step 1: Choose and Download a Torrent Client**
- Before you can start sharing or downloading files, you need to choose and install a torrent client. Choose carefully, as some come with adware that can cause issues with your computer or device.
- Best to download directly from the client's website so you can avoid downloading malware from third-party sites. While there are free-for-download clients, going for a premium client is ideal if you want security features.
- **Step 2: Install a Tracker Site**
- Once you've installed a torrent client, you also need to download a tracker site containing listings of torrent files. However, they are only a repository for torrent files and do not host content on their servers.
- There are two types of trackers sites. One is a public tracker site, accessible to all users. The other is a private tracker site

containing specialized torrent websites that host unique niches of files. Registration to a private tracker site is often exclusive and by invite only. It also requires users to seed torrents after each download.

Step 3: Search Content for Download

Next, you can search for the content you want to download. Search results often return several files. Choose the ones with many seeders, so your download goes faster. Before downloading, check if you can run the file.

Step 4: Download the Content

Once you know if the file is compatible with your installed programs, you can download the content. You can download multiple files as well.

Advantages of BitTorrent

BitTorrent has the following advantages:

- Distributing large files like Linux iSO images.
- Distributing Software patches and updates.
- As being done by Blizzard Entertainment Inc, to distribute updates for the world of Warcraft.
- Distributing popular files which have high traffic for relatively short periods.
- Unlike traditional server or client downloads, high traffic leads to more efficient file sharing via BitTorrent.

Principles of Network Applications

The Principles of Network Applications are fundamental concepts that govern the design and development of applications that run on a computer network. These principles encompass several key aspects of network applications, including:

- Network Application Architectures
- Processes Communicating
- The Interface Between the Process and the Computer Network
- Transport Services Available to Applications
- Transport Services Provided by the Internet
- Application-Layer Protocols

1. Network Application Architectures refer to the overall design and structure of a network application. It encompasses how the application is divided into different components, and how these components interact with each other. There are several commonly used network application architectures, including:

- **Client-Server Architecture:** In this architecture, one component acts as a client and makes requests to a server component, which provides the requested services. This architecture is commonly used in web applications, where the client is a web browser and the server is a web server.
- **Peer-to-Peer Architecture:** In this architecture, every component is both a client and a server, and each component can communicate directly with any other component. This architecture is commonly used in file-sharing applications, where each user's device acts as both a client and a server.

2. **Processes Communicating** refers to the communication between multiple processes in a computer network. Processes can be thought of as individual programs or tasks running on a device, and they may be located on the same device or on different devices connected to the network.

- Communication between processes is facilitated by the use of protocols, which define the rules and formats for exchanging data. The communication between processes can be either synchronous or asynchronous
- In a network application, communication between processes is essential for the application to function correctly. For example, in a client-server architecture, the client process makes requests to the server process, and the server process returns the requested information. In a peer-to-peer architecture, each process can communicate directly with any other process.

3. The Interface between the Process and the Computer Network refers to the connection between a process running on a device and the underlying computer network. This interface determines how the process communicates with other processes and with the network itself. The interface between a process and the computer network is usually provided by a network stack, which is a collection of protocols and services that handle the communication between the process and the network.

4. Transport Services Available to Applications are the services provided by the network stack that enable applications to communicate with each other over a computer network. These services are responsible for ensuring that data is reliably delivered between applications, and they provide the underlying communication infrastructure for the application.
There are several transport services available to applications, including:

1. **TCP (Transmission Control Protocol)**: TCP is a reliable, connection-oriented transport service that provides error-

checking and flows control to ensure that data is delivered accurately. Applications that require reliable data delivery, such as email or file transfer, typically use TCP.

2. **UDP (User Datagram Protocol)**: UDP is an unreliable, connectionless transport service that does not provide error checking or flow control. Applications that require low latency or high speed, such as video streaming or online gaming, typically use UDP.
3. **SCTP (Stream Control Transmission Protocol)**: SCTP is a reliable, multi-homed transport service that provides error checking and flow control. SCTP can handle multiple streams of data between applications, allowing for efficient communication between applications.
4. **DCCP (Datagram Congestion Control Protocol)**: DCCP is a transport service that provides congestion control for applications that do not require reliable data delivery.

Transport Services Provided by the Internet The choice of transport service will depend on the requirements of the application, including reliability, performance, and security requirements

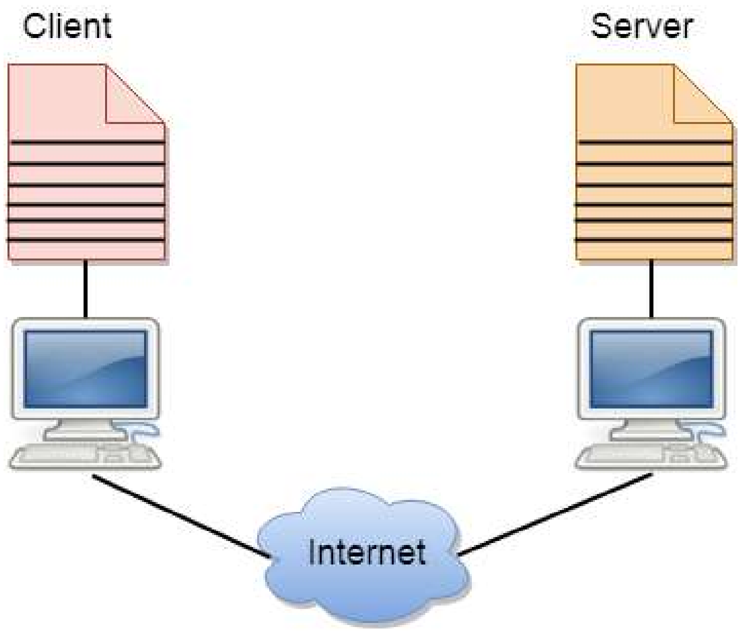
5. Transport Services Provided by the Internet: The Internet provides two primary transport services for applications: TCP (Transmission Control Protocol) and UDP (User Datagram Protocol).

1. **TCP**: TCP is a reliable, connection-oriented transport service that provides error-checking and flows control to ensure that data is delivered accurately. Applications that require reliable data delivery, such as email or file transfer, typically use TCP. TCP establishes a reliable connection between two devices and ensures that data is transmitted in the correct order and without errors.
2. **UDP**: UDP is an unreliable, connectionless transport service that does not provide error checking or flow control. Applications that require low latency or high speed, such as

video streaming or online gaming, typically use UDP. Because UDP does not provide error checking or flow control, it is faster and more efficient than TCP, but it may not be suitable for applications that require reliable data delivery.

Client and Server model

- A client and server networking model is a model in which computers such as servers provide the network services to the other computers such as clients to perform a user based tasks. This model is known as client-server networking model.
- The application programs using the client-server model should follow the given below strategies:



- An application program is known as a client program, running on the local machine that requests for a service from an application program known as a server program, running on the remote machine.

- A client program runs only when it requests for a service from the server while the server program runs all time as it does not know when its service is required.
- A server provides a service for many clients not just for a single client. Therefore, we can say that client-server follows the many-to-one relationship. Many clients can use the service of one server.
- Services are required frequently, and many users have a specific client-server application program. For example, the client-server application program allows the user to access the files, send e-mail, and so on. If the services are more customized, then we should have one generic application program that allows the user to access the services available on the remote computer.

Client

A client is a program that runs on the local machine requesting service from the server. A client program is a finite program means that the service started by the user and terminates when the service is completed.

Server

A server is a program that runs on the remote machine providing services to the clients. When the client requests for a service, then the server opens the door for the incoming requests, but it never initiates the service.

A server program is an infinite program means that when it starts, it runs infinitely unless the problem arises. The server waits for the incoming requests from the clients. When the request arrives at the server, then it responds to the request.

Advantages of Client-server networks:

- **Centralized:** Centralized back-up is possible in client-server networks, i.e., all the data is stored in a server.
- **Security:** These networks are more secure as all the shared resources are centrally administered.
- **Performance:** The use of the dedicated server increases the speed of sharing resources. This increases the performance of the overall system.
- **Scalability:** We can increase the number of clients and servers separately, i.e., the new element can be added, or we can add a new node in a network at any time.

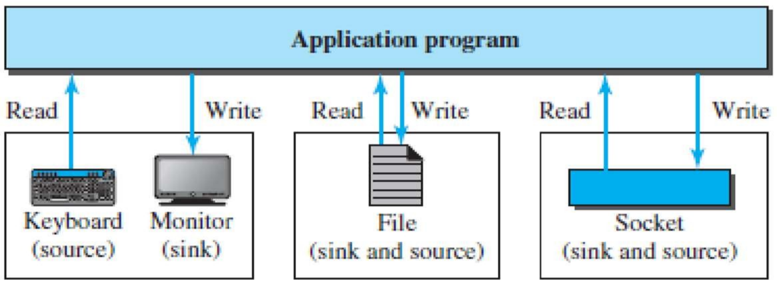
Disadvantages of Client-Server network:

- **Traffic Congestion** is a big problem in Client/Server networks. When a large number of clients send requests to the same server may cause the problem of Traffic congestion.
- It does not have a robustness of a network, i.e., when the server is down, then the client requests cannot be met.
- A client/server network is very decisive. Sometimes, regular computer hardware does not serve a certain number of clients. In such situations, specific hardware is required at the server side to complete the work.

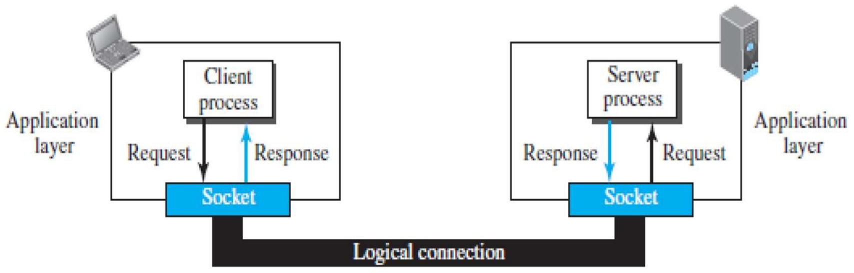
Socket Interface: Socket interface started in the early 1980s at UC Berkeley as part of a UNIX environment. The socket interface is a set of instructions that provide communication between the application layer and the operating system. It is a set of instructions that can be used by a process to communicate with another process.

For example, in most computer languages, like C, C++, or Java, we have several instructions that can read and write data to other sources and sinks such as a keyboard (a source), a monitor (a sink), or a file (source and sink).

In application layer, communication between a client process and a server process is the communication between two sockets. As far as the application layer is concerned, communication between a client process and a server process is communication between two sockets, created at two ends,



The client thinks that the socket is the entity that receives the request and gives the response; the server thinks that the socket is the one that has a request and needs the response. If we create two sockets, one at each end, and define the source and destination addresses correctly, we can use the available instructions to send and receive data. The rest is the responsibility of the operating system and the embedded TCP/IP protocol.



APPLICATION LAYER PROTOCOL

The Hypertext Transfer Protocol (HTTP) is an application-level protocol that uses TCP as an underlying transport and typically runs on port 80. HTTP is a stateless protocol i.e. server maintains no information about past client requests.

HTTP Connections

1. Non-Persistent
2. Persistent

Basic Pre-Requisite

The terminology which we must know before going deep into Persistent & Non-Persistent Connections is

1. RTT(Round Trip Time)
2. TCP 3-Way Handshake

1. **RTT:** Time for a small packet to travel from client to server and back.

```
RTT = 2 X propagation time
```

1. For a connection Persistent or Non-persistent it is sure that to initiate a TCP connection one RTT is used.

2. One RTT is used for the HTTP request and the first few bytes to the HTTP response to return. So to know the total file transmission time.

```
Total = 2RTT + transmit time
```

2. TCP 3-Way Handshake: TCP Connection establishes in 3 ways, that's why it is called a 3-way Handshake.

- Requesting the server for the connection.
- The server responds to whether the connection can be established or not.
- Acknowledgment by the client on the response sent by the server.

Difference between Persistent and Non-Persistent Connections

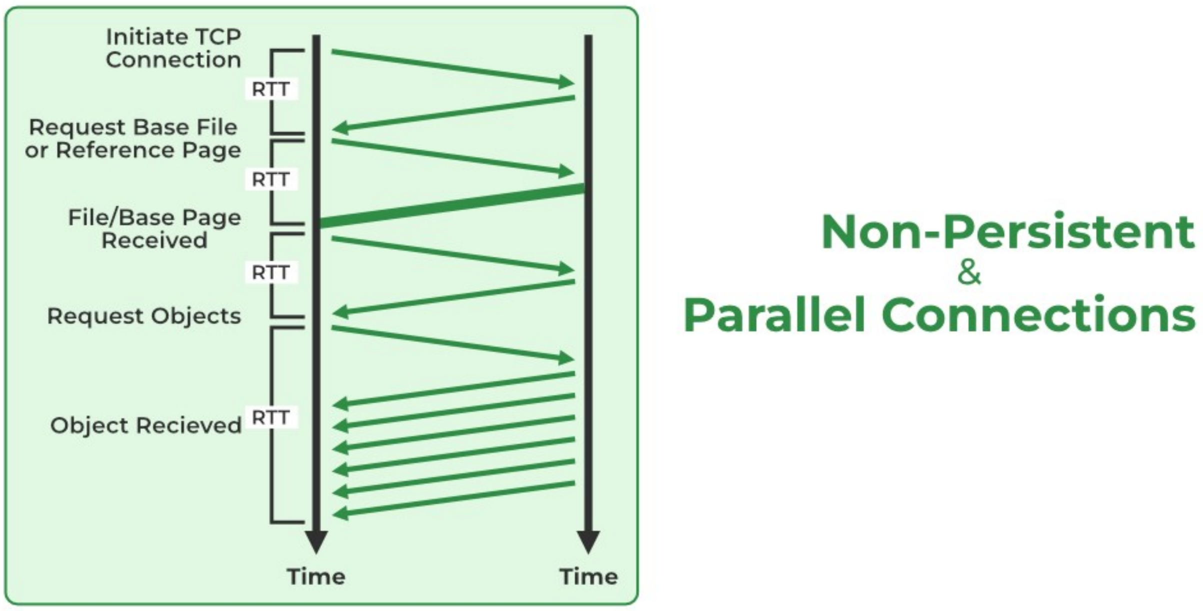
Persistent HTTP	Non-Persistent HTTP
The server leaves the connection open after sending a response.	Requires 2 RTTs per object.

Persistent HTTP	Non-Persistent HTTP
Subsequent HTTP messages between the same client/server are sent over an open connection.	OS overhead for each TCP connection
The client sends requests as soon as it encounters a referenced object.	Browsers often open parallel TCP connections to fetch referenced objects.
As little as one RTT for all the referenced objects.	Here, at most one object can be sent over one TCP Connection.

Non-Persistent Connection

Non-Persistent Connections are those connections in which for each object we have to create a new connection for sending that object from source to destination. Here, we can send a maximum of one object from one TCP connection.
There are two types:

1. **Non-Persistent-Without parallel connection:** Each objection takes two RTTs (assuming no window limit) one for TCP connection and the other for HTTP image/text file.
2. **Non-Persistent-With parallel connection:** Non-Persistent with a parallel connection requires extra overhead in transferring data.



Non-Persistent & Parallel Connection

Advantages of Non-Persistent Connection

1. Wastage of Resources is very less because the connection opens only when there is some data to be sent.
2. Non-Persistent Connection is more secure because after sending the data, the connection gets terminated and nothing can be shared thereafter.

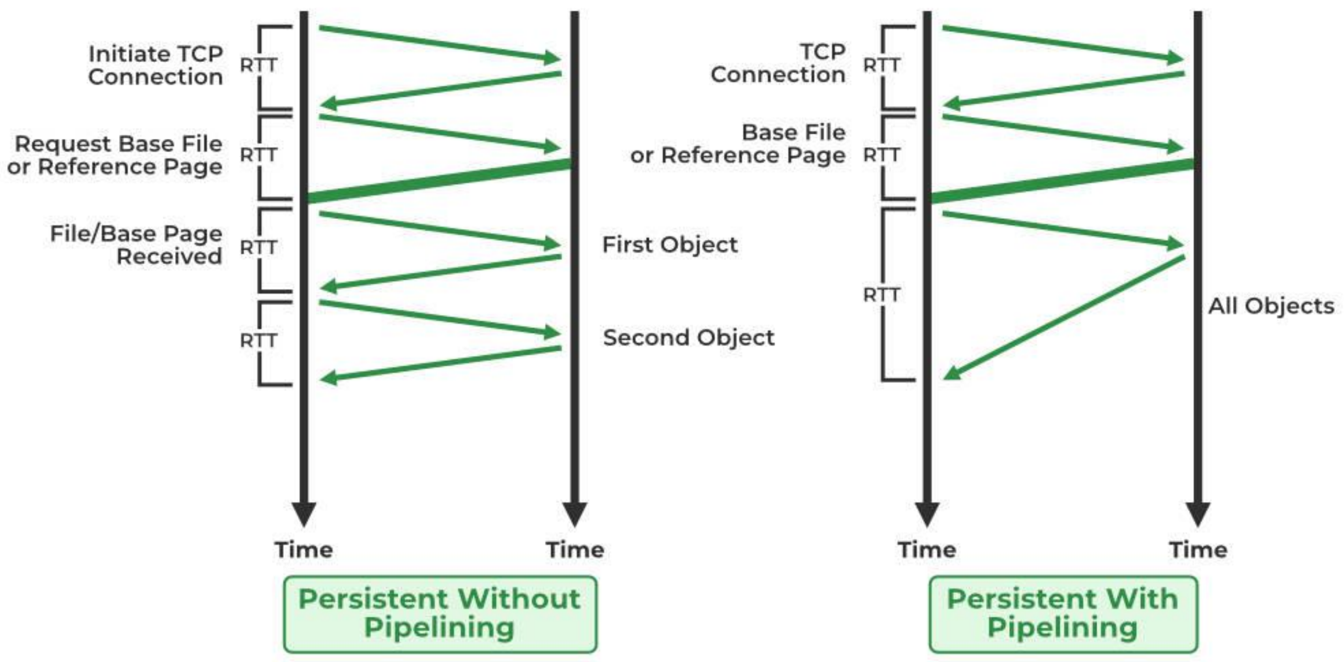
Disadvantages of Non-Persistent Connection

1. In Non-Persistent Connection, it requires a greater CPU overhead for the transmission of data

Persistent Connection

1. Non-Pipelined Persistent Connection: In a Non-pipeline connection, we first establish a connection that takes two RTTs

then we send all the object’s images/text files which take 1 RTT each (TCP for each object is not required).
2. Pipelined Persistent Connection: In Pipelined connection, 2RTT is for connection establishment and then 1RTT(assuming no window limit) for all the objects i.e. images/text.



Persistent Without Pipelining and with Pipelining

Advantages of Persistent Connections

- Lower CPU and memory usage because there is less number of connections.
- Allows HTTP pipelining of requests and responses.
- Reduced network congestion (fewer TCP connections).
- Reduced latency in subsequent requests (no handshaking).
- Errors can be reported without the penalty of closing the TCP connection.

Disadvantages of Persistent Connections

- Resources may be kept occupied even when not needed and may not be available to others.

- Most modern browsers like Chrome, Firefox, and Internet Explorer use persistent connections.

HTTP Message

HTTP Message is used to show how data is exchanged between the client and the server. It is based on client-server architecture. An **HTTP** client is a program that establishes a connection to a server to send one or more HTTP request messages. An **HTTP server** is a program that accepts connections to serve HTTP requests by sending an HTTP response messages.

The HTTP Messages can be classified as follows:

Message Type

HTTP message consists of an initial request line and an initial response line.

Format:

1. HTTP-message = Request | Response ; HTTP/1.1 messages

1) Initial Request Line

The initial line is different for the request and for the response. A request-line consists of three parts: a **method name, requested resource's local path**, and the **HTTP version** being used. All these parts are separated by spaces.

Syntax:

1. GET /path/to/file/index.html HTTP/1.0

Here,

- **GET** is the most common HTTP method.
- The **path** shows the part of the URL after the host name. It is also called a request URI.
- The **version** of HTTP always takes the form "**HTTP/x.x**", uppercase.

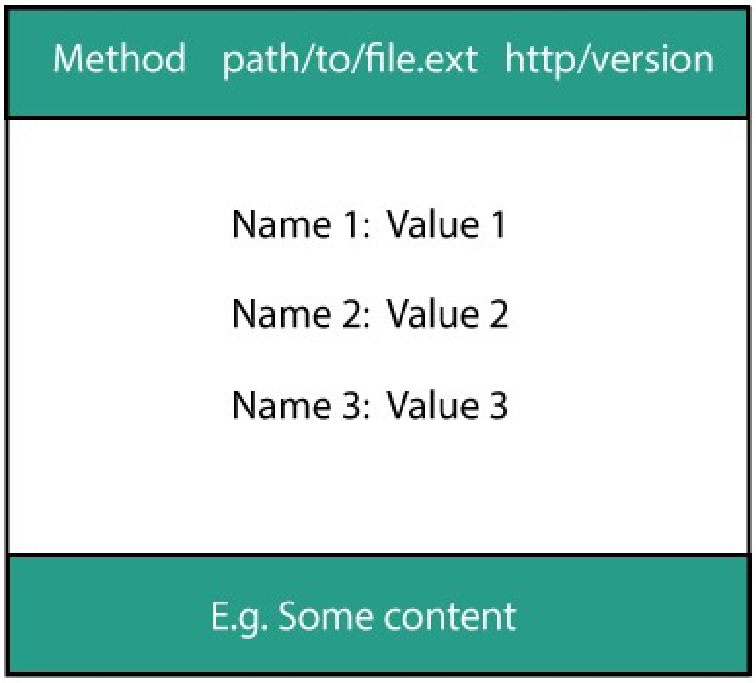


Fig:- Request HTTP Message

2) Initial Response Line

The initial Response line is also known as the status line. It also has three parts: the HTTP version, a response status code that gives the result of the request, and the English reason phrase describing the status code.

he HTTP version of the response line and request line are the same as "HTTP/x.x".

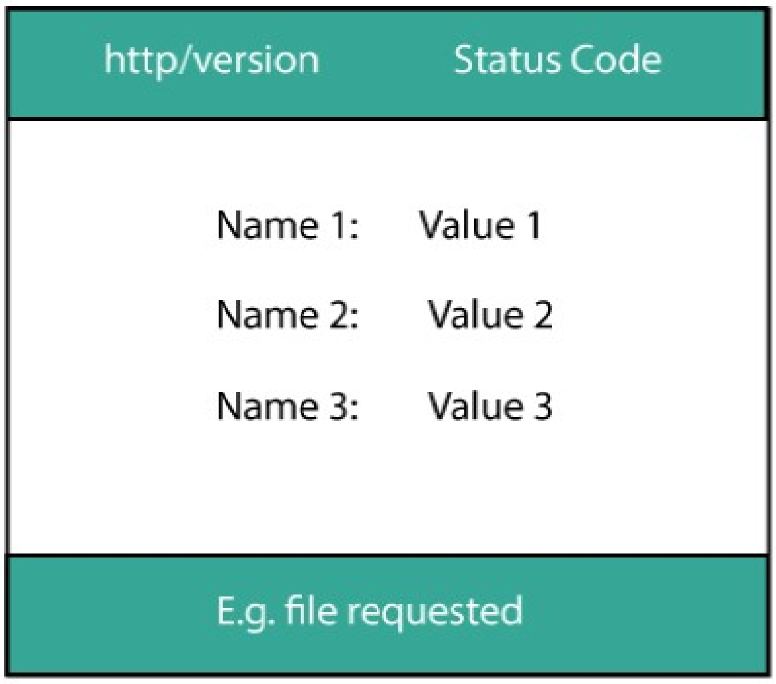


Fig:- Response HTTP Message

Message Headers

The Message header provides information about the request and response. It also provides information about the object which is sent in the message body. Message Headers are of four types:

1. **General Header:** It has general applicability for both request messages and response messages.
2. **Request Header:** It has applicability only for the request messages.
3. **Response Header:** It has applicability only for the response messages.

4. **Entity Header:** It defines meta-information about the entity-body, and about the resource identified by request.

HTTP Request

HTTP Requests are messages which are sent by the client or user to initiate an action on the server.

The first line of the message includes the request message from the client to the server, the method which is applied to the resource, identifier of the resource, and the protocol version.

Syntax

```
1. Request      = Request-Line
2.              *(( general-header
3.              | request-header
4.              | entity-header ) CRLF)
5.               CRLF
6.              [ message-body ]
```

HTTP Response

HTTP Response sent by a server to the client. The response is used to provide the client with the resource it requested. It is also used to inform the client that the action requested has been carried out. It can also inform the client that an error occurred in processing its request.

An HTTP response contains the following things:

1. Status Line
2. Response Header Fields or a series of HTTP headers

3. Message Body

Http Cookies

An Http cookie, also known as a web cookie, is a small piece of data that the server sends to the web browser. The browser may store the cookies and send them back to the server with other requests. The typical use of the cookies is to identify multiple requests from the same resource (browser). It helps the user to keep logged in and trace his data, such as user name and session. It is also useful to remember the stateful information for the stateless HTTP protocol.

When the user visits any site, he requests the application's home page from the server, and when He selects any specific action, he again requests any other page of the application. So if the user visits the 100th time every time, the request will be unique.

Usage of Cookies

The following are some common usage of the cookie:

Session Management:

The primary use of the cookie is to manage user logins, shopping cart details, game scores, or anything else the server should remember when the user logs in next time.

Tracking

Tracking involves recording the user activity and analyzing their behaviours to personalize the content for them. It records and analyzes their habits and interests and finds the pages that they visit. The tracking details may include the time spent on a page or across the website during a session. There is some sensitive information in the user tracking, so the user should be aware of the vulnerabilities of visiting insecure websites and avoid if possible visiting such websites.

Personalization

Cookies help in personalizing user preferences, themes, and other settings. Most of the time, these settings are synchronized with a central database to personalize the things for the users when the user logs in for the first time. The user personalization keeps the information of the user preferences and settings for reapplying when the user logs in or restarts the application.

Authentication

Cookies are also very important in user authentication. When the user signs in, the server uses the **Set-Cookie HTTP header** in response to set a cookie with a unique "session identifier".

How Cookies Works

When a user interacts with a website, a new request to the server is sent, and when he performs any action within the application each time, a new request is sent to the server. The server considered every new request sent by a different user.

To recognize the requests by the same user, the developers need to add the cookie with the server's response, which is stored in the system's local memory. So if the same user sends a request to the server, the server identifies the user using cookies.

From the client side communication, the web browser is the medium to send the request from the client to the server.

Once the developers add the cookie, the requests sent by the users are identified, and the server sends the response as per the personalization.

Types of Cookies

First-Party cookies

The first-party cookies are the cookies which are being sent to the system straight from the server of the site that you are visiting. These cookies are stored directly on your computer by the visited website. Most websites send cookies to the client to collect helpful information and analytics to improve the user experience.

The best example of a first-party cookie is authentication. When the user logs in to an application, the server sends the cookies to the client and stores them in their local system to identify the user. Next time, when you log in to the application, you do not need to submit the login information again.

Third-Party cookies

The third-party cookies are created by the third-party domains to which you are redirected. It is not created by the same domain in which you have logged in. These cookies are generally created for tracking purposes and can be stored even after the browser is closed.

One great example of a third-party cookie is ad tracking from websites.

Session cookies

The session cookies are also temporary cookies, which will expire when the user closes the browser tab. Usually, session cookies are used when the user enters their login credentials and logs in to the application. When the session cookies are used, the users have to provide their credentials every time when they try to log in to the application.

Secure cookies

The cookies provide a Secure attribute to prevent unauthorized access from the observing cookies sent to a user within an HTTP response. When the cookies are used with a Secure attribute, all the HTTP requests will only include the cookie if it is transmitted over a secure medium.

Initially, Microsoft Internet Explorer implemented the HttpOnly cookies in 2002. The cookies flagged with HttpOnly can be included in a Set-Cookie HTTP header.

Zombies cookies can store themselves in another place rather than the browser's dedicated storage. Generally, the Zombie cookie can create a replica of the original cookie, store it elsewhere, and attach it to the user's cookie storage again.

Proxy Server

A proxy server refers to a server that acts as an intermediary between the request made by clients, and a particular server for some services or requests for some resources. There are different types of proxy servers available that are put into use according to the purpose of a request made by the clients to the servers. The basic purpose of Proxy servers is to protect the direct connection of Internet clients and Internet resources

The proxy server also prevents the identification of the client's IP address when the client makes any request to any other servers.

- **Internet Client and Internet resources:** For Internet clients, Proxy servers also act as a shield for an internal network against the request coming from a client to access the data stored on the server. It makes the original IP address of the node remain hidden while accessing data from that server.

Protects true host identity: In this method, outgoing traffic appears to come from the proxy server rather than internet navigation. It must be configured to a specific application such as HTTP or FTP

Types Of Proxy Server

- **Reverse Proxy Server:** The job of a reverse proxy server to listen to the request made by the client and redirect to the particular web server which is present on different servers. Example – Listen for TCP port 80 website connections which are normally placed in a demilitarized zone (DMZ) zone for publicly accessible services but it also protects the true identity of the host.
- **Web Proxy Server:** Web Proxy forwards the HTTP requests, only URL is passed instead of a path. The request is sent to particular the proxy server responds. Examples, Apache, HAP Proxy.
- **Anonymous Proxy Server:** This type of proxy server does not make an original IP address instead these servers are

detectable still provides rational anonymity to the client device.

- **Highly Anonymity Proxy:** This proxy server does not allow the original IP address and it as a proxy server to be detected.
- **Transparent Proxy:** This type of proxy server is unable to provide any anonymity to the client, instead, the original IP address can be easily detected using this proxy. But it is put into use to act as a cache for the websites.

CGI Proxy: CGI proxy server developed to make the websites more accessible. It accepts the requests to target URLs using a web form and after processing its result will be returned to the web browser.

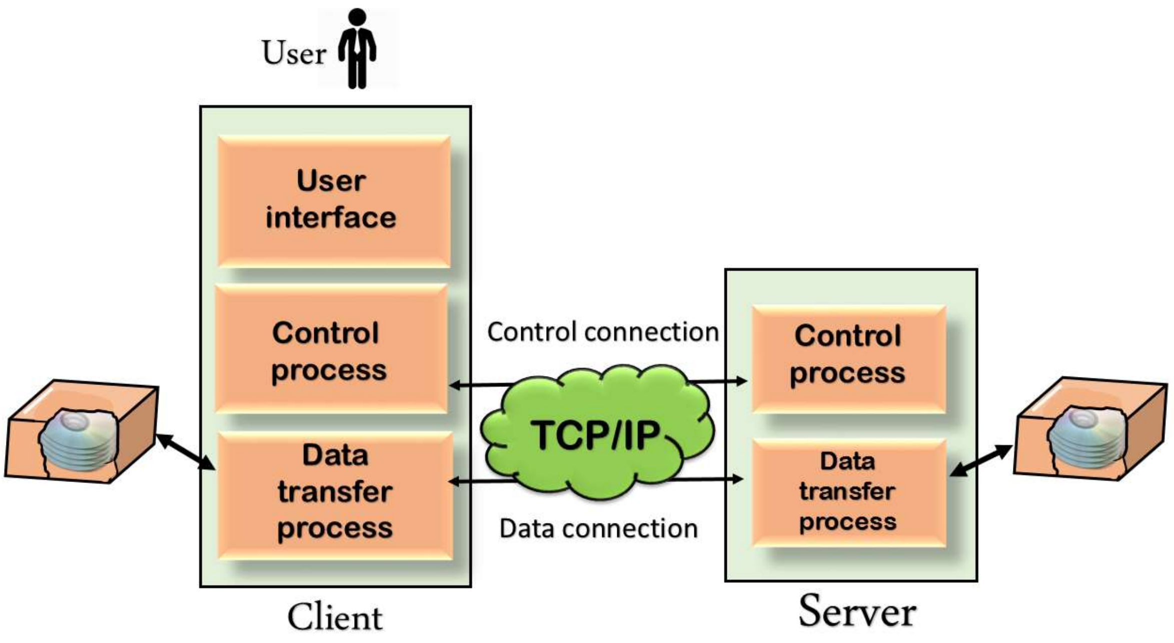
FTP:

- FTP stands for File transfer protocol.
- FTP is a standard internet protocol provided by TCP/IP used for transmitting the files from one host to another.
- It is mainly used for transferring the web page files from their creator to the computer that acts as a server for other computers on the internet.
- It is also used for downloading the files to computer from other servers.

Objectives of FTP

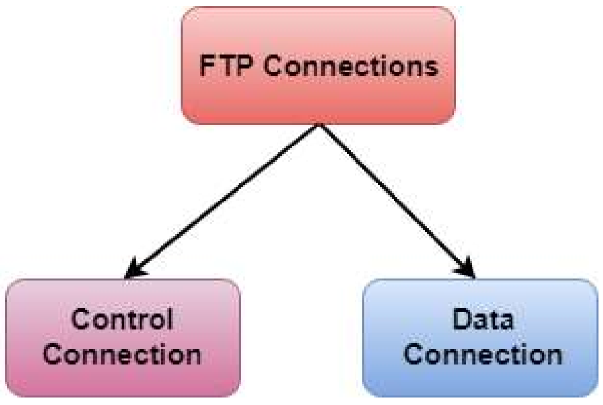
- It provides the sharing of files.
- It is used to encourage the use of remote computers.
- It transfers the data more reliably and efficiently.

Mechanism of FTP



The above figure shows the basic model of the FTP. The FTP client has three components: the user interface, control process, and data transfer process. The server has two components: the server control process and the server data transfer process.

There are two types of connections in FTP:



- **Control Connection:** The control connection uses very simple rules for communication. Through control connection, we can transfer a line of command or line of response at a time. The control connection is made between the control processes. The control connection remains connected during the entire interactive FTP session.

- **Data Connection:** The Data Connection uses very complex rules as data types may vary. The data connection is made between data transfer processes. The data connection opens when a command comes for transferring the files and closes when the file is transferred.

FTP Clients

- FTP client is a program that implements a file transfer protocol which allows you to transfer files between two hosts on the internet.
- It allows a user to connect to a remote host and upload or download the files.
- It has a set of commands that we can use to connect to a host, transfer the files between you and your host and close the connection.
- The FTP program is also available as a built-in component in a Web browser. This GUI based FTP client makes the file transfer very easy and also does not require to remember the FTP commands.

Advantages of FTP:

- **Speed:** One of the biggest advantages of FTP is speed. The FTP is one of the fastest way to transfer the files from one computer to another computer.
- **Efficient:** It is more efficient as we do not need to complete all the operations to get the entire file.
- **Security:** To access the FTP server, we need to login with the username and password. Therefore, we can say that FTP is more secure.
- **Back & forth movement:** FTP allows us to transfer the files back and forth. Suppose you are a manager of the company, you send some information to all the employees, and they all send information back on the same server.

Disadvantages of FTP:

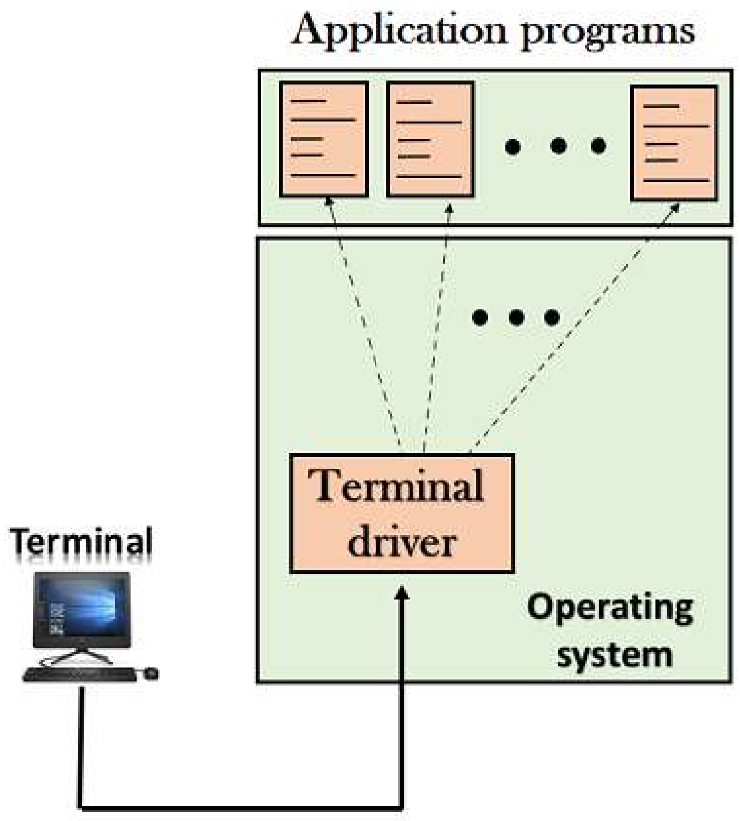
- The standard requirement of the industry is that all the FTP transmissions should be encrypted. However, not all the FTP providers are equal and not all the providers offer encryption. So, we will have to look out for the FTP providers that provides encryption.
- FTP serves two operations, i.e., to send and receive large files on a network. However, the size limit of the file is 2GB that can be sent. It also doesn't allow you to run simultaneous transfers to multiple receivers.

Telnet

- The main task of the internet is to provide services to users. For example, users want to run different application programs at the remote site and transfers a result to the local site. This requires a client-server program such as FTP, SMTP. But this would not allow us to create a specific program for each demand.
- The better solution is to provide a general client-server program that lets the user access any application program on a remote computer. Therefore, a program that allows a user to log on to a remote computer. A popular client-server program Telnet is used to meet such demands. Telnet is an abbreviation for **Terminal Network**.
- Telnet provides a connection to the remote computer in such a way that a local terminal appears to be at the remote side.

There are two types of login:

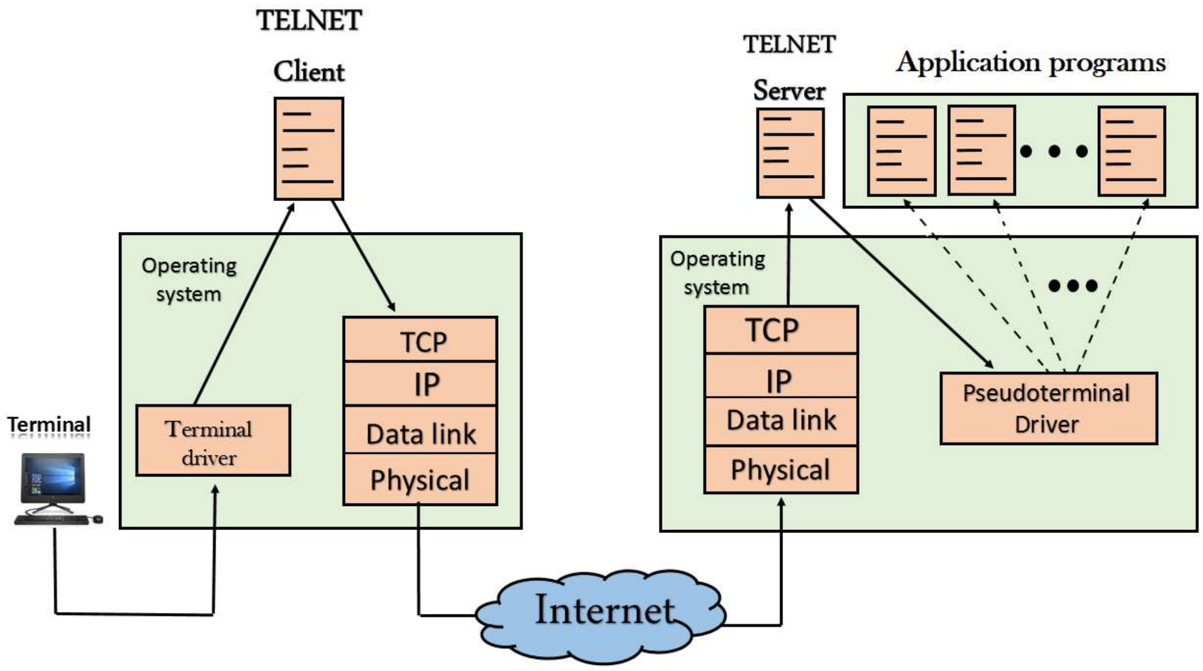
Local Login



- When a user logs into a local computer, then it is known as local login.
- When the workstation running terminal emulator, the keystrokes entered by the user are accepted by the terminal driver. The terminal driver then passes these characters to the operating system which in turn, invokes the desired application program.
- However, the operating system has special meaning to special characters. For example, in UNIX some

combination of characters have special meanings such as control character with "z" means suspend. Such situations do not create any problem as the terminal driver knows the meaning of such characters. But, it can cause the problems in remote login.

Remote login



- When the user wants to access an application program on a remote computer, then the user must perform remote login.

How remote login occurs

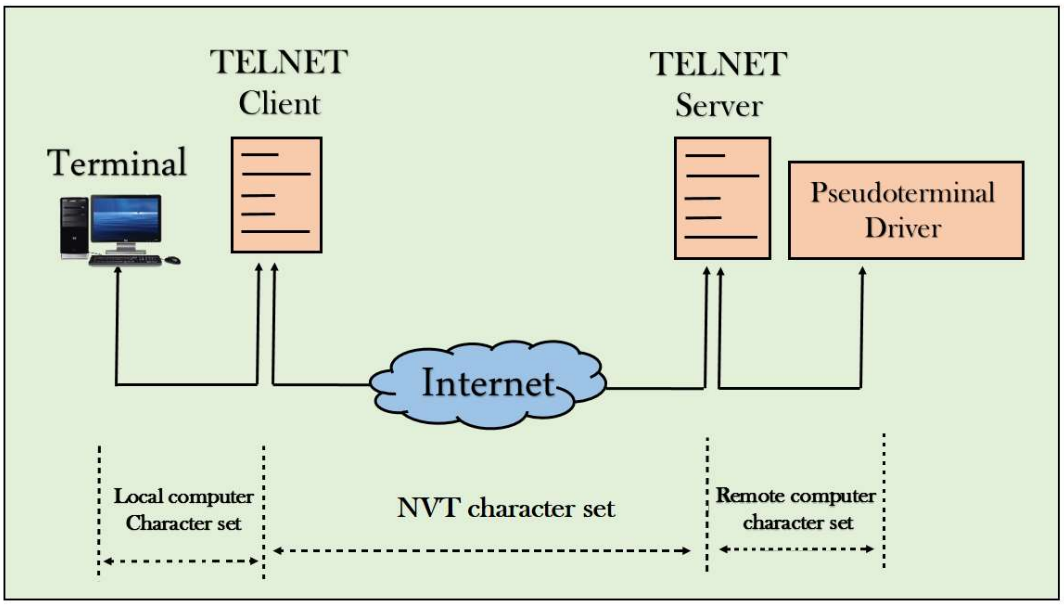
At the local site

The user sends the keystrokes to the terminal driver, the characters are then sent to the TELNET client. The TELNET client which in turn, transforms the characters to a universal character set known as network virtual terminal characters and delivers them to the local TCP/IP stack

At the remote site

The commands in NVT forms are transmitted to the TCP/IP at the remote machine. Here, the characters are delivered to the operating system and then pass to the TELNET server. The TELNET server transforms the characters which can be understandable by a remote computer. However, the characters cannot be directly passed to the operating system as a remote operating system does not receive the characters from the TELNET server. Therefore it requires some piece of software that can accept the characters from the TELNET server. The operating system then passes these characters to the appropriate application program.

Network Virtual Terminal (NVT)



- The network virtual terminal is an interface that defines how data and commands are sent across the network.
- In today's world, systems are heterogeneous. For example, the operating system accepts a special combination of characters such as end-of-file token running a DOS operating system *ctrl+z* while the token running a UNIX operating system is *ctrl+d*.
- TELNET solves this issue by defining a universal interface known as network virtual interface.
- The TELNET client translates the characters that come from the local terminal into NVT form and then delivers them to the network. The Telnet server then translates the data from NVT form into a form which can be understandable by a remote computer.

DNS