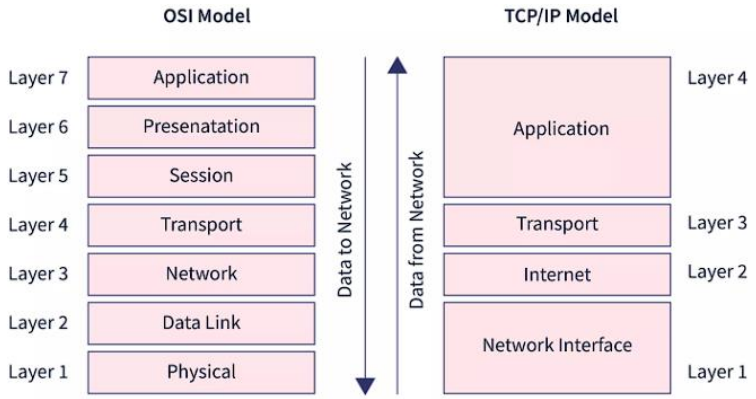
UNIT – 5 : APPLICATION LAYER

Introduction:

The application layer is the topmost layer of the OSI model and the TCP/IP model. In TCP/IP model, the application layer is formed by combining the top three layers, i.e., the application layer, the presentation layer, and the session layer.

An application layer is an abstraction layer that specifies the shared communications protocols and interface methods used by hosts in a communications network. It is the layer closest to the **end-user**, implying that the application layer and the **end-user** can interact directly with the software application.



It does not provide service to other layers because it is the topmost layer. The Application layer uses Transport and any levels below it to communicate with or transfer data to a remote host.

Consumers frequently require protocols from the Application Layer. One of the most often used application protocols is HTTP (HyperText Transfer Protocol), the foundation for the World Wide Web. Domain Name System (DNS), Simple Mail Transfer Protocol (SMTP), File Transfer Protocol (FTP), Trivial File Transfer Protocol (TFTP), and TELNET are some of the protocols used in the application layer.

What are the Services Provided by the Application Layer?

The application provides the following services -

1. The application layer **guarantees** that the receiver is recognized, accessible, and ready to receive data from the sender.
2. It enables authentication between devices for an extra layer of network security.
3. It determines the protocol and data syntax rules at the application level.

4. The protocols of the application layer also define the basic syntax of the message being forwarded or retrieved.
5. It also checks whether the sender's computer has the necessary communication interfaces, such as an Ethernet or Wi-Fi interface.
6. Finally, the data on the receiving end is presented to the user application.

Functions of the Application Layer:

The application layer provides the following functions -

1. The Application Layer provides protocols that allow the software to communicate and receive data and finally present it to users in a meaningful way.
2. This layer allows users to log on as a remote host.
3. The Application Layer provides various facilities for users to forward multiple emails and a storage facility.
4. This layer acts as a window via which users and application processes can access network resources.
5. This layer provides services such as email, file transfer, results distribution, directory services, network resources, etc.
6. The application layer communicates with the operating system and guarantees that data is properly saved.
7. This layer allows users to interact with other software applications.
8. This application layer generally performs host initialization followed by remote login to hosts.
9. This layer visualizes data, allowing individuals to grasp it rather than memorize it or see it in binary format (1s or 0s).

Working of Application Layer in the OSI model:

In the OSI model, this application layer is narrower in scope.

The application layer in the OSI model generally acts only like the interface which is responsible for communicating with host-based and user-facing applications. This is in contrast with TCP/IP protocol, wherein the layers below the application layer, which is Session Layer and Presentation layer, are clubbed together and form a simple single layer which is responsible for performing the functions, which includes controlling the dialogues between computers, establishing as well as maintaining as well as ending a particular session, providing data compression and data encryption and so on.

At first, client sends a command to server and when server receives that command, it allocates port number to client. Thereafter, the client sends an initiation connection request to server and when server receives request, it gives acknowledgement (ACK) to client through client has successfully established a connection with the server and, therefore, now client has access to server through which it may either ask server to send any types of files or other documents or it may upload some files or documents on server itself.

Application Layer Protocols:

Each protocol serves a unique purpose, enabling specific types of data exchange, communication, and network services. Here is a table showing application layer protocol examples.

Protocol	Function
HTTP (Hypertext Transfer Protocol)	Used for transferring data over the web, forming the foundation of data exchange on the web.
HTTPS (Hypertext Transfer Protocol Secure)	HTTP with additional security, it uses SSL/TLS to create a secure connection.
FTP (File Transfer Protocol)	Used for transferring files between a client and a server. It allows you to upload and download files, and manage file systems on a server.
SMTP (Simple Mail Transfer Protocol)	Used for the transmission and delivery of email across IP networks.
DNS (Domain Name System)	Used to translate domain names into IP addresses, making it easier for users to access websites without memorizing numerical IP addresses.
DHCP (Dynamic Host Configuration Protocol)	Automatically assigns IP addresses within a network, eliminating the need for manual configuration.

Difference Between Peer-to-Peer and Client-Server Network:

Peer-to-Peer Model	Client-Server Model
The peer-to-peer network model is distributed and decentralized in nature.	The client-server network model is also distributed in nature but it is centralized.
The main focus of the peer-to-peer network model is on the connectivity among the computer systems.	The main focus of the client-server network model is on data sharing.
In the peer-to-peer network model, each computer system can act as a client and a server.	In the client-server network model, there is a centralized server.
In the P2P network model, each computer system stores individual files and data.	In the client-server network model, there is a central backup of the files and data.
It is more reliable than the client-server model as in case of failure of one system, the entire network does not get affected.	The client-server network model entirely depends upon the central server, so in case of server failure, the entire network gets affected.
The P2P network model is more reliable.	The client-server model is more robust.
In the P2P network model, the response time is low as the computer systems are directly connected.	In the client-server model, the access time may be slow if there are several requests made on the server.
The P2P network model is cheaper as there is no need to implement the centralized server.	The client-server network model is costlier as there is a need for the implementation of the centralized server.
The P2P network model is less secure as there is no need for authentication before the communication.	The client-server network model is more secure as every device needs authentication before communication.
The P2P network model becomes less stable if the number of the peer (computer systems) increases.	The client-server network model is more stable than the P2P network model.
The P2P network model is suitable for small-scale businesses and houses.	The client-server network model is suitable for both small-scale businesses and large networks.
Areas where the P2P network model is used: in Napster (audio streaming platform), in the distribution of games (such as - StarCraft II, Diablo III, and World of Warcraft).	Areas where the client-server network model is used: in web browsers for requesting webpages, in database servers for accessing query results, in mail servers, etc.

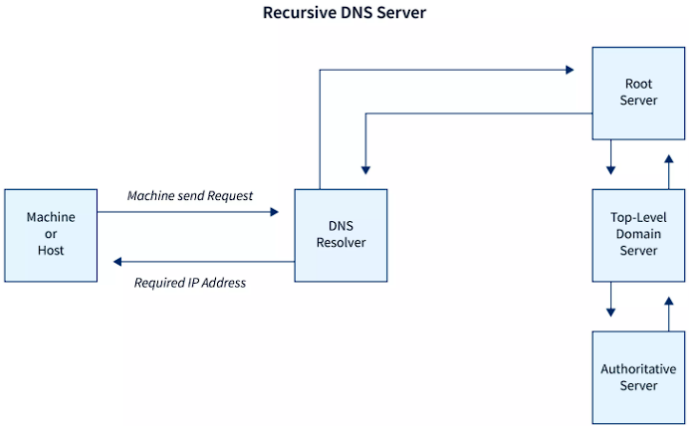
Domain Name System:

The **Domain Name System** (DNS) is a method for mapping alphabetic names to numeric IP addresses on the Internet, similar to how a phone book maps a person's name to a phone number. A DNS query is made when a web **address (URL)** is typed into a browser to obtain the IP address of a web server associated with that name. The DNS directory is distributed worldwide to account for the millions of domain names listed and accessed daily. A domain name can correspond to more than one IP address if various users look for the same website simultaneously. If 100 people search for scaler.com at the same time, each receives a different IP address from separate servers. If a domain name only had one server and one IP address, all **100 individuals** would be waiting in line to access the site.

DNS contains various servers that cover the alphabetic domains to its numeric IP. The resolution process of the **DNS can either** be iterative or recursive.

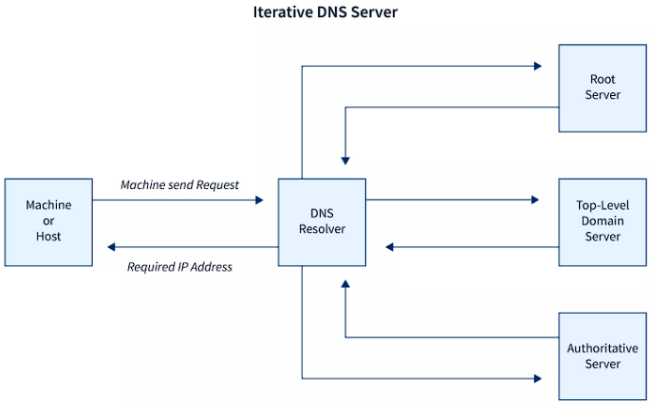
1. Recursive DNS Service:

In this type, if the DNS resolver only communicates to the root servers and the remaining servers were communicated **recursively** by the root server. The root server sends the output (IP in this case) to the DNS resolver.



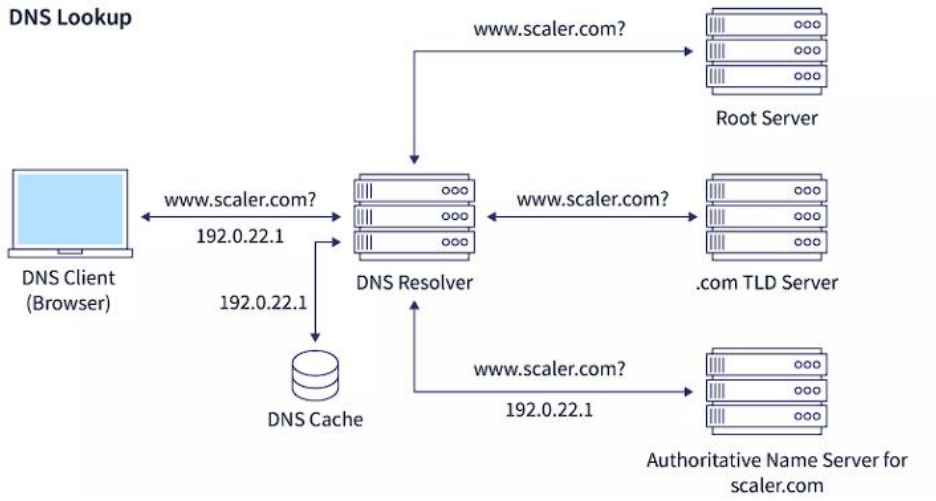
2. Iterative DNS Service:

In this type, the **DNS resolver** can directly communicate and receive input from the servers at different levels.



How Does DNS Work?

DNS is concerned with translating a domain name into an **IP address**. To learn how this process works, look at the following steps.



1. If you type scaler.com into a web browser, the query is transmitted over the Internet and received by a DNS resolver.
2. The DNS resolver then queries a **DNS root** nameserver.
3. After then, the root server responds to the DNS resolver with the address of a **TLD DNS** server (such as .com or .net), which keeps the information for the resolver's domains. Our request for scaler.com is directed to the .com **top-level domain** (TLD).
4. The DNS resolver then requests the .com TLD after receiving the address of the TLD by the root server.
5. The IP address of the domain nameserver, scaler.com, is then returned by the TLD server.
6. Finally, the DNS resolver sends a query to the domain's nameserver.
7. The **nameserver returns** the IP address, for scaler.com, to the resolver.
8. The DNS resolver then returns the IP address of the domain that was requested originally to the web browser.
9. An HTTP request is sent to the IP address by the browser.
10. The server returns the webpage to be **rendered** in the browser at that IP.
11. Finally, after all the processes mentioned above, the user can now view the web page on their machine.

DNS Structure:

In Request for Comments **(RFC) 1035**, the **Internet Engineering Task Force** (IETF) lays out the requirements for implementing domain names. In most cases, the domain name is included in the URL. **Labels** are the components that make up a domain name. Each segment of the domain hierarchy denotes a subdivision and is read from right to left.

Each label on the TLD's left side designates a different subdomain of the domain to the right. For example, "xyz" is a subdomain of .com, and "www." is a subdomain of techtarget.com in the URL www.xyz.com. Each label can have up to 63 characters, and there can be up to 127 levels of subdomains. Up to 253 characters can be used in the overall domain character length. Labels do not start with a hyphen and cannot have entirely numeric **TLD names**.

DNS Server Types:

To complete a **DNS** resolution, different types of servers are used. The order in which a query passes through the four name servers is listed below. They can either supply the requested domain name or refer you to alternative name servers.

DNS Resolver: An application, such as a web browser, sends DNS queries to the recursive server. It's the user's **first resource**, and it either provides the response to the query if it has it cached, or it goes to the next-level server if it doesn't. Before answering the client, this server may undergo multiple iterations of querying.

Root Name Server: If the recursive server does not have the answer cached, it first sends a query to this server. The root name server is a directory of all the servers that will hold the requested information. The Internet Corporation for Assigned Names and Numbers **(ICANN)**, specifically the Internet Assigned Numbers Authority, is in charge of these servers.

Top-Level Domain Server: A TLD nameserver keeps track of all domain names with the same domain extension, such as .com, .net, or whatever comes after the last dot in a URL. For example, a **'.com' TLD nameserver** includes information for every website that ends in '.com'. If a user searched for scaler.com, the DNS resolver would submit a query to a .com **TLD nameserver**, which would answer by referring to the authoritative nameserver for that domain.

Authoritative Name Server: A **DNS resolver** will be directed to an authoritative nameserver when it receives a response from a TLD nameserver. The resolver's final stage in the path for an IP address is usually the authoritative nameserver. The server contains information specific to the domain name it serves **(e.g., scaler.com)**. If the authoritative name server has access to the requested record, it will return the requested hostname's IP address to the DNS Recursor (the librarian) who initiated the request.

Types of DNS Queries:

In DNS resolution, generally, three types of queries occur. Combining these queries results in optimal DNS resolution, which reduces the distance traveled to resolve the domain to its respective IP. These queries are described below.

1. Recursive Query: A DNS client expects a DNS server **(usually a DNS resolver)** to respond to a recursive query with either the requested resource record or an error message if the resolver is unable to locate it.

2. Iterative Query: It occurs between the DNS resolver and the nonlocal name servers, like the root, TLD, and authoritative name servers. The root server refers to the recursive server to the TLD, which directs it to an authoritative server. If the authoritative server has the domain name, it passes it on to the **recursive server**. Iterative requests either result in a response or a referral.

3. Non-recursive Query: This usually happens when a DNS resolver client requests a record from a DNS server that it has access to, either because it is authorized for the record or because the record is in its cache. DNS servers typically cache **DNS records** to save unnecessary bandwidth usage and pressure on upstream systems.

Common DNS Record:

DNS records are the data that a query is looking for. Different information is required depending on the **query, client, or application**. Some recordings, such as the A record, are essential. There are various DNS record kinds, each having its unique function in indicating how a query should be handled. The following are some examples of DNS records:

- **A Record** stands for address and holds the IP address of a domain.
- **Ns record** stands for name server records that specify which authoritative server is responsible for maintaining all information for a specific domain.
- **TXT records** enable administrators to enter text into DNS.
- When there is an alias, **canonical name records** are utilized instead of an A record. They're used to retry a query with two different domains from the same IP address.

How Does DNS Increase Web Performance?

We live in an era where speed is everything, and everyone wants speed with the maximum possible accuracy. To increase web performance, DNS uses caching to store the set of records or IP addresses received by the **DNS** queries for a fixed amount of time. Caching improves efficiency by allowing servers to react fast when receiving a request for the same IP address.

To understand better, let us take an example. Suppose you want to revisit scaler topics to read our fantastic articles after reading this article. The IP address of the scaler topics is cached in your browser, and when you search for it the next day, it will get picked by the cache instead of searching in various DNS servers. The time to which the record is held in the cache is called TTL. The **time to live** (TTL) of a record is determined by administrators and is based on a variety of variables. More extended periods reduce server burden, while shorter periods offer the most precise responses.

Domain Name System Caching:

DNS caching aims to shorten the time it takes to receive a response to a DNS query. Caching allows DNS to save past answers to requests closer to clients, allowing them to obtain the same information faster the next time they query it. Caching improves efficiency by allowing servers to react quickly when receiving a request for the same **IP address**. There are various places where caching is done. Some common ones include the following.

1. Operating System: Many operating systems include stub resolvers, which store DNS data and handle queries before sending them to an external server.

2. Web Browsers: Many web browsers store DNS for a fixed time. Browers allow fast resolution of IP addresses.

3. DNS Resolver: The DNS resolver can also cache the result of a DNS query. Some resolvers may already have some of the records needed to respond, allowing them to skip some steps in the DNS resolution process.

Domain Name System Security:

Along with the various features and reliability the DNS provides, it also has some vulnerabilities. Two of the major vulnerabilities are

1. Cache Poisoning: DNS cache poisoning is a misleading assault that diverts traffic away from authorized websites and puts users at risk of malware infestations and data theft. An attacker uses a web server and cache to serve a malicious **Hypertext Transfer Protocol** (HTTP) response to users in web cache poisoning. DNS resolvers cannot validate the data in their caches, which implies that inaccurate data will remain in the cache until the issue is manually fixed or TTL(time to live) expires.

2. Phishing: It is done to get users' data by creating a false website of a well-known website with an utterly **unauthorized backend**. Phishing causes user privacy and financial status as it is intended to harm the end-user.

Domain Name Space:

The name of a host is divided into various pieces called domains. These domains are structured in a hierarchical structure so that top-level domains are listed at the top of the hierarchy and low levels are listed at the bottom. When searching for a host, we start our searching in ascending order, **i.e., from leaf nodes to root nodes**.

Top Level Domains and Country Domains

The TLD servers are divided into two groups by the IANA:

1. Generic Top-level Domains:

Some of the most well-known generic TLDs are .com, .org, .net, .edu, and .gov. Also, these domains are not country-specific.

2. Country code top-level domains:

Any domains specific to a country or state fall under this category. Some examples of country code top-level domains are .uk, .us, .ru, and .jp.

Inverse domain: The inverse domain is used for mapping an address to a name. When the server has received a request from the client, and the server contains the files of only authorized clients. To determine whether the client is on the authorized list or not, it sends a query to the DNS server and ask for mapping an address to the name.

```
DNS = Name service in Internet – A zone is an administrative unit,
and a domain is a subtree.
```

Electronic Mail:

Electronic mail, commonly known as email, is a method of exchanging messages over the internet.

Here are the basics of email:

1. **An email address:** This is a unique identifier for each user, typically in the format of name@domain.com.
2. **An email client:** This is a software program used to send, receive and manage emails, such as Gmail, Outlook, or Apple Mail.
3. **An email server:** This is a computer system responsible for storing and forwarding emails to their intended recipients.

To send an email:

1. Compose a new message in your email client.
2. Enter the recipient's email address in the "To" field.
3. Add a subject line to summarize the content of the message.
4. Write the body of the message.
5. Attach any relevant files if needed.
6. Click "Send" to deliver the message to the recipient's email server.
7. Emails can also include features such as cc (carbon copy) and bcc (blind carbon copy) to send copies of the message to multiple recipients, and reply, reply all, and forward options to manage the conversation.

Electronic Mail (e-mail) is one of most widely used services of Internet. This service allows an Internet user to send a **message in formatted manner (mail)** to the other Internet user in any part of world. Message in mail not only contain text, but it also contains images, audio and videos data. The person who is sending mail is called **sender** and person who receives mail is called **recipient**. It is just like postal mail service.

Components of E-Mail System:

The basic components of an email system are: User Agent (UA), Message Transfer Agent (MTA), Mail Box, and Spool file. These are explained as following below.

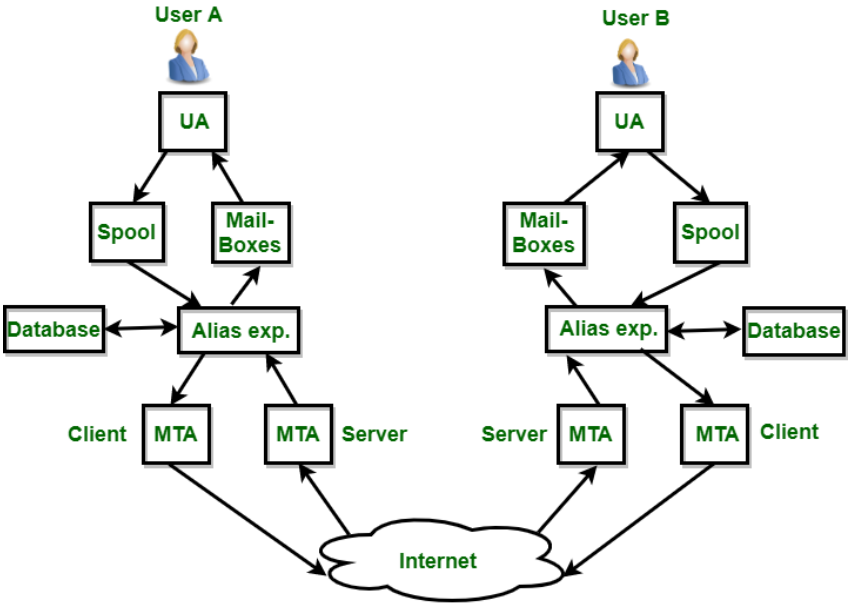
1. **User Agent (UA):** The User-Agent is a simple software that sends and receives mail. It is also known as a mail reader. It supports a wide range of instructions for sending, receiving, and replying to messages and manipulating mailboxes.

   Some of the services supplied by the User-Agent are listed below:

   - Reading a Message
   - Sending a reply to a Message
   - Message Composition
   - Forwarding a Message
   - Handling the Message

2. **Message Transfer Agent (MTA):** MTA is actually responsible for transfer of mail from one system to another. To send a mail, a system must have client MTA and system MTA. It transfers mail to mailboxes of recipients if they are connected in the

same machine. It delivers mail to peer MTA if destination mailbox is in another machine. The delivery from one MTA to another MTA is done by Simple Mail Transfer Protocol.



3. **Message Access Agent:** The Simple Mail Transfer Protocol is used for the first and second stages of e-mail delivery.

   The pull protocol is mainly required at the third stage of e-mail delivery, and the message access agent is used at this point.

   POP and IMAP4 are the two protocols used to access messages.

4. **Mailbox:** It is a file on local hard drive to collect mails. Delivered mails are present in this file. The user can read it delete it according to his/her requirement. To use e-mail system each user must have a mailbox. Access to mailbox is only to owner of mailbox.

5. **Spool file:** This file contains mails that are to be sent. User agent appends outgoing mails in this file using SMTP. MTA extracts pending mail from spool file for their delivery. E-mail allows one name, an **alias**, to represent several different e-mail addresses. It is known as **mailing list**, Whenever user have to sent a message, system checks recipient's name against alias database. If mailing list is present for defined alias, separate messages, one for each entry in the list, must be prepared and handed to MTA. If for defined alias, there is no such mailing list is present, name itself becomes naming address and a single message is delivered to mail transfer entity.

Services provided by E-mail system:

- **Composition –** The composition refers to process that creates messages and answers. For composition any kind of text editor can be used.
- **Transfer –** Transfer means sending procedure of mail i.e. from the sender to recipient.
- **Reporting –** Reporting refers to confirmation for delivery of mail. It helps user to check whether their mail is delivered, lost or rejected.
- **Displaying –** It refers to present mail in form that is understand by the user.
- **Disposition –** This step concern with recipient that what will recipient do after receiving mail i.e save mail, delete before reading or delete after reading.

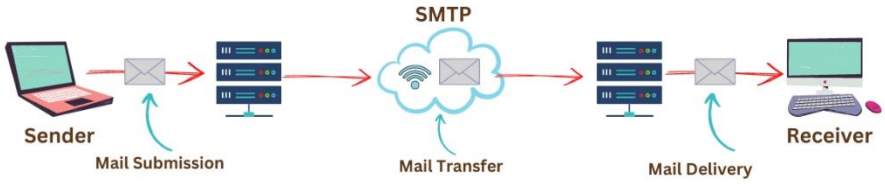
Advantages of email:

1. Convenient and fast communication with individuals or groups globally.
2. Easy to store and search for past messages.
3. Ability to send and receive attachments such as documents, images, and videos.
4. Cost-effective compared to traditional mail and fax.
5. Available 24/7.

Disadvantages of email:

1. Risk of spam and phishing attacks.
2. Overwhelming amount of emails can lead to information overload.
3. Can lead to decreased face-to-face communication and loss of personal touch.
4. Potential for miscommunication due to lack of tone and body language in written messages.
5. Technical issues, such as server outages, can disrupt email service.
6. It is important to use email responsibly and effectively, for example, by keeping the subject line clear and concise, using proper etiquette, and protecting against security threats.

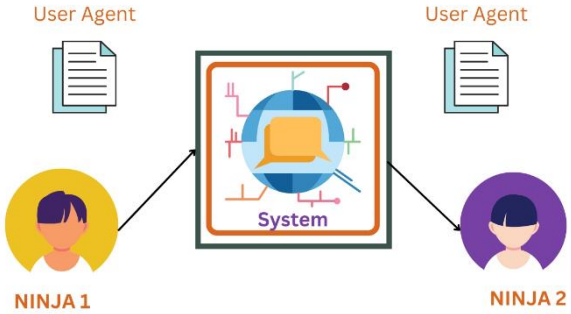
Architecture of Electronic Mail:



First Scenario:

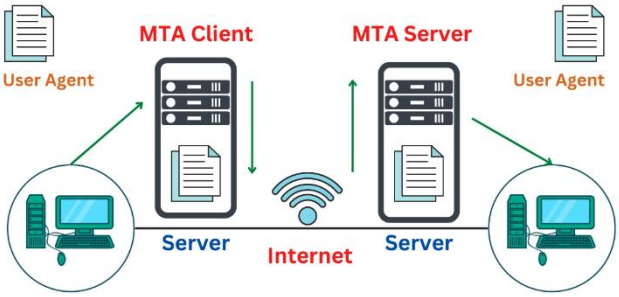
In the first scenario, two user agents are required. The sender and recipient of the e-mail share the same machine directly connected to the server.

For example, let us consider two user agents, Ninja1 and Ninja2. When Ninja1 sends an e-mail to Ninja2, the user agent (UA) programme is used to prepare the message. Following that, this e-mail gets saved in the Ninja2 inbox.



Second Scenario:

In this case, the sender and recipient of an e-mail are essentially users on two different machines over the internet. User-Agents and Message Transfer Agents (MTA) are required in this scenario.

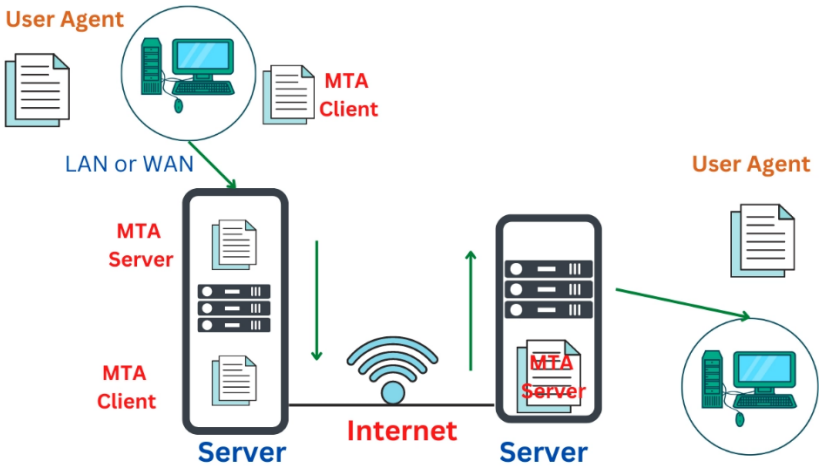


Take, for example, two user agents (Ninja1 and Ninja2), as illustrated in the diagram. When Ninja1 sends an e-mail to Ninja2, the user agent (UA) and message transfer agents (MTAs) programmes prepare the e-mail for transmission over the internet. Following that, this e-mail gets stored in Ninja2's inbox.

Third Scenario:

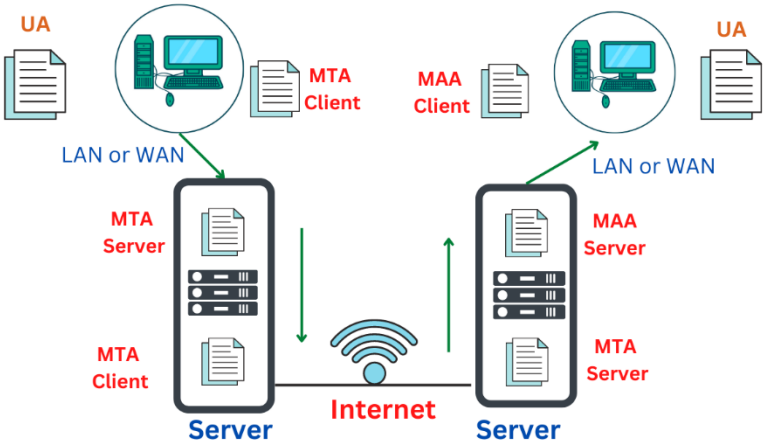
The sender is connected to the system by a point-to-point WAN, which can be a dial-up modem or a cable modem in this case. On the other hand, the receiver is directly attached to the system, as it was in the second scenario.

The sender also needs a User agent (UA) to prepare the message in this situation. After preparing the statement, the sender delivers it over LAN or WAN via a pair of MTAs.



Fourth Scenario:

In this scenario, the recipient is linked to the mail server via WAN or LAN. When the message arrives, the recipient must retrieve it, which needs additional client/server agents. This scenario requires two user agents (UAs), two pairs of message transfer agents (MTAs), and a couple of message access agents (MAAs).



Differences Between Email and Webmail:

Feature	Email	Webmail
Access	Requires a dedicated email client or software (e.g., Outlook, Thunderbird).	Accessed through a web browser, making it platform-independent.
Location	Emails stored on the user's device or mail server, depending on configuration.	Emails are stored on remote servers, accessible from any internet-enabled device.
Accessibility	Accessible only from the device where the email client is installed.	Accessible from any device with a web browser and internet connection.
Storage Limits	Limited by the user's device or email server capacity.	Generally, offers larger storage capacities, often with cloud storage.
User Interface	Dependent on the email client software and may vary.	Interface is standardized, often providing a consistent experience.
Updates	Software updates managed by the user for client-based email.	Updates are server-side, requiring no action from the end user.
Configuration	Requires manual configuration of email client settings.	Webmail services usually require minimal configuration, primarily requiring login credentials.
Integration	May have limited integration with other applications.	Often integrates seamlessly with other online services and apps.
Examples	Microsoft Outlook, Thunderbird, Apple Mail.	Gmail, Yahoo Mail, Outlook.com, and other web-based email services.

Different types of Email (Electronic Mail):

There are various types of email services and protocols catering to different needs and preferences. Here are some common types:

1. Web-Based Email: Web-based email services are accessed through a web browser, eliminating the need for dedicated email client software. Users can access their emails from any internet-enabled device with a browser, making it a convenient and widely-used option. For example, Gmail, Yahoo Mail, Outlook.com.

2. Client-Based Email: Client-based email requires dedicated email client software installed on the user's device for access. These applications offer a more feature-rich and often customizable experience compared to web-based email clients. For example, Microsoft Outlook, Mozilla Thunderbird.

3. Secure Email Services: Secure email services prioritize end-to-end encryption and advanced security features to protect user privacy and sensitive information. They are often favored by individuals who prioritize secure communication. For example, ProtonMail, Tutanota.

4. Business or Corporate Email: Tailored for business use, corporate email services often include collaboration tools, shared calendars, and enhanced security features to meet the specific needs of organizations. For example, Microsoft Exchange, Google Workspace.

5. Disposable Email Services: Disposable email services provide temporary email addresses for short-term use. Users often utilize them for activities like online registrations or verifications, maintaining privacy. For example, Guerrilla Mail, 10 Minute Mail.

6. Encrypted Email Protocols: Encrypted email protocols focus on securing the content of emails. Technologies like PGP and S/MIME employ encryption techniques to ensure confidentiality in email communication. For example, PGP (Pretty Good Privacy), S/MIME (Secure/Multipurpose Internet Mail Extensions).

7. POP3 (Post Office Protocol 3): POP3 retrieves emails from the server to the local device, typically deleting them from the server. It is commonly used when users want to download and store emails locally.

E-Mail Format:

Electronic Mail (e-mail) is one of the most widely used services of the Internet. This service allows an Internet user to send a **message in a formatted manner (mail)** to other Internet users in any part of the world. Message in the mail not only contain text, but it also contains images, audio and videos data.

The person who is sending mail is called **sender** and person who receives mail is called the **recipient**. It is just like postal mail service.

Format of E-mail:

An e-mail consists of three parts that are as follows:

1. Envelope

2. Header

3. Body

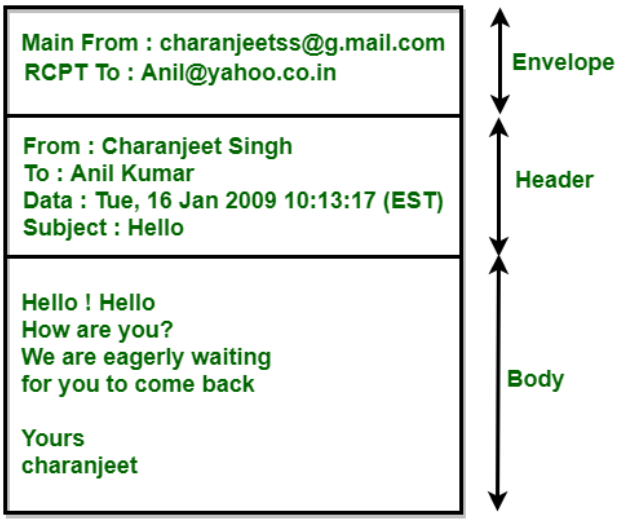
1. Envelope: The envelope part encapsulates the message. It contains all information that is required for sending any e-mail such as destination address, priority and security level. The envelope is used by MTAs for routing message.

2. Header: The header consists of a series of lines. Each header field consists of a single line of ASCII text specifying field name, colon and value. The main header fields related to message transport are:

1. **To:** It specifies the DNS address of the primary recipient(s).
2. **Cc:** It refers to carbon copy. It specifies address of secondary recipient(s).
3. **BCC:** It refers to blind carbon copy. It is very similar to Cc. The only difference between Cc and Bcc is that it allow user to send copy to the third party without primary and secondary recipient knowing about this.
4. **From:** It specifies name of person who wrote message.
5. **Sender:** It specifies e-mail address of person who has sent message.
6. **Received:** It refers to identity of sender's, data and also time message was received. It also contains the information which is used to find bugs in routing system.
7. **Return-Path:** It is added by the message transfer agent. This part is used to specify how to get back to the sender.

3. Body: The body of a message contains text that is the actual content/message that needs to be sent, such as "Employees who are eligible for the new health care program should contact their supervisors by next Friday if they want to switch." The message body also may include signatures or automatically generated text that is inserted by the sender's email system.

The above-discussed field is represented in tabular form as follows:



In addition to above-discussed fields, the header may also contain a variety of other fields which are as follows:

Header	**Meaning**
Date:	Date and time when the message was sent.
Reply-To:	It contains e-mail address to which replies should be sent.
Message-Id:	It refers to the unique number for referencing this message later.
In-Reply-To:	Message-Id of a message to which this is as a reply.
References:	It contains other relevant message-ids.
Keywords:	User-chosen keywords.
Subject:	It contains short summary of message for one-line display.

World Wide Web (WWW):

The **World Wide Web** (popularly known as the WWW) is a troupe of web pages or websites that is stored within the web servers. This is a collection of web pages or websites connected to local computer systems with the help of the internet. The **World Wide web** is used to connect people through images, text pages, videos, digital content, etc.

The World Wide Web enables a user who is using their computer system or mobile devices to access the content of various sites over the internet network.

The World Wide Web runs on a set of rules that determine how data is transmitted between different devices in the same network. These are known as protocols. There are security risks that affect Web servers, the local area networks that host Web sites, and even innocent users of Web browsers. The risks are most severe from the Webmaster's perspective.

The World Wide Web popularly known as **WWW, W3, or the Web** is an interconnected system of public webpages accessible through the Internet.

The Web is not the same as the Internet: the Web is one of many applications built on top of the Internet.

History of the World Wide Web:

The history of the **World Wide Web** is associated with **Tim Berners Lee**. After graduating from Oxford University, the English computer scientist started working with CERN (Conseil Européen pour la Recherche Nucléaire or European Council for Nuclear Research), the large particle physics laboratory near Geneva, Switzerland, as a software engineer. While his term at CERN he discovered that there was a problem in data sharing among different scientists due to the unavailability of a common network.

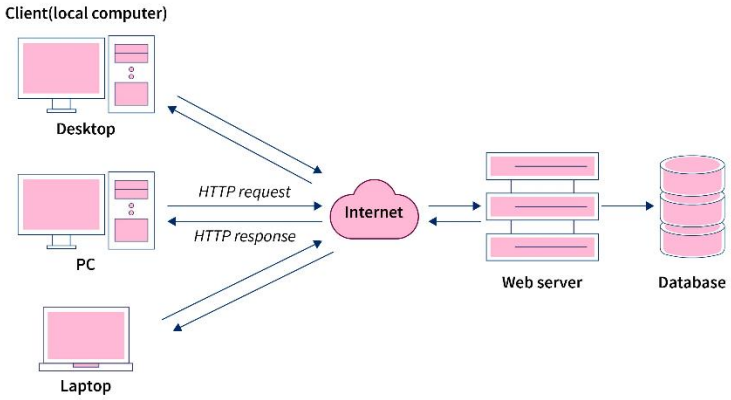
- In **March 1989**, a proposal was laid out by Tim laid, known as *' Information Management: A Proposal, proposing the existence of a common network to share data. This was rejected by the CERN.
- In **September 1990**, the initial work on the web started.
- In **October 1990:** The three fundamental technologies of the web were written. These included HTML, URI, and HTTP.
- By the end of **1990**, the first web page was available on the open internet.
- In **1991**, the web community was opened to people outside CERN.
- In **1994**, the World Wide Web Consortium was founded, it is an international community devoted to developing open web standards.
- At present most of the users are using the **Web 2.0** version. **Web 3.0** has been proposed that focuses on using a machine-based understanding of data to provide a data-driven and Semantic Web.

How Does the World Wide Web Work?

Now that we have an understanding of what the World Wide Web is, in this section, we will see the workings of the World Wide Web.

The working of the World Wide Web is based on the **client-server** model. A web server is software and hardware that uses HTTP (Hypertext Transfer Protocol) and other protocols to respond to client requests made over the World Wide Web.

A **web server** is used to store data or information and this data is transferred to the computer of the user when a user on the same network requests data or information. The computer system of the user that is requesting this data is known as a client. This exchange of data is usually carried out by the browser present on our computer systems. It allows us to access the retrieved data in the form of documents from the web.



Evolution of the World Wide Web:

Over the years the world wide web has evolved a lot. The continuous development work on the World Wide Web has made it vast and contemporary.

The web has primarily seen three generations:

Web 1.0:

Web 1.0 is referred to as the beginning stage of the World Wide Web evolution. It is the first version of the web. This phase included the presence of minimal content over the web with the majority of the users who were consuming the content. Most of the web pages were static and were hosted either on free web hosting providers or on ISP-run web servers.

In **Web 1.0** advertisements on websites while surfing the internet are banned. Also, in **Web 1.0**, Ofoto is an online digital photography website, on which users can store, share, view, and print digital pictures. **Web 1.0** is a content delivery network (CDN) that enables the showcase of the piece of information on the websites. It can be used as a personal website. It costs the user as per pages viewed. It has directories that enable users to retrieve a particular piece of information. The era of **Web 1.0** was roughly from 19911991 to 20042004.

Following are the features of Web 1.0:

- Web 1.0 had static pages.
- The server file system was used to serve content (for example read-only articles, or static web pages).
- The positioning and alignment of the elements on a page were done using tables or frames.

Web 2.0:

Since **Web 1.0** lacked the facilities of user-generated content, usability, and interoperability for end-users, to tackle these issues and provide features like dynamic websites, APIs, etc., **Web 2.0** was introduced.

Web 2.0 is not a change but a transition of **Web 1.0** from static and riveted content to dynamic content. With this transition, it became relatively easier for the user of the web to create their content on the web and share it with others. The web browser technologies are used in **Web 2.0** development and it includes AJAX and JavaScript frameworks. Recently, **AJAX** and **JavaScript frameworks** have become a very popular means of creating **Web 2.0** sites.

Following are the features of Web 1.0 :

- It allows the users to retrieve and classify the information collectively.
- The content is user-responsive and dynamic thus making the website device friendly in terms of alignment and sizing.
- **APIs** were introduced to allow self-usage, such as by a software application.
- Allowed users to put content over the web. Social media etc. was introduced.

Web 3.0:

Web 3.0 is the latest advancement in the World Wide Web.

Web 3.0 refers to the evolution of web utilization and interaction which includes altering the Web into a database, with the integration of DLT (Distributed Ledger Technology blockchain is an example), and that data can help to make Smart Contracts based on the needs of the individual.

Web 3.0 has brought multiple evolutions to web interaction and usage. The main idea behind **Web 3.0** is that the data is not owned by a certain authority but is shared among the web.

Features of WWW:

The World Wide Web provides the following features:

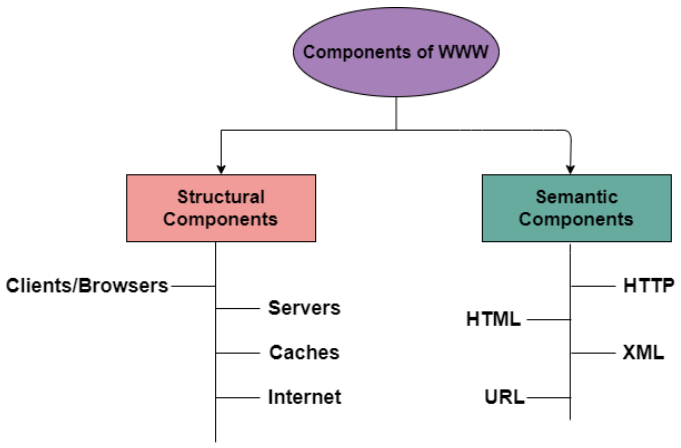
1. **HyperText Information System:** Hypertext is a system for linking related text documents that allow the participation of multiple users. In a hypertext document, any word or phrase can be **"hyperlinked"** to information related to that word or phrase residing in the same document or another document.
2. **Cross-Platform:** **Cross-platform** apps are the ones with built-in web languages (like JavaScript) that can be later pulled (f.e. through React Native) as native apps able to work on any operating system and device. Cross-platform apps are great when you want to: Build your app 50% faster. Build one app for both iOS and Android.
3. **Distributed:** WWW is a **distributed client-server** service. In this, a client can access the services from a server using a browser. These services are usually distributed over many locations called sites or websites. From the user's point of view, the web consists of a vast worldwide collection of documents called web pages.
4. **Open Standards and Open Source:** WWW provides the features of Open standards and Open source. An open standard is a standard that is freely available for adoption, implementation, and updates. A few famous examples of open standards are XML, SQL, and HTML. Businesses within an industry share open standards because this allows them to bring huge value to both themselves and their customers.

   Open-source software is computer software that is released under a license in which the copyright holder grants users the rights to use, study, change, and distribute the software and its source code to anyone and for any purpose.

Components of WWW:

The Components of WWW mainly falls into two categories:

1. Structural Components 2. Semantic Components

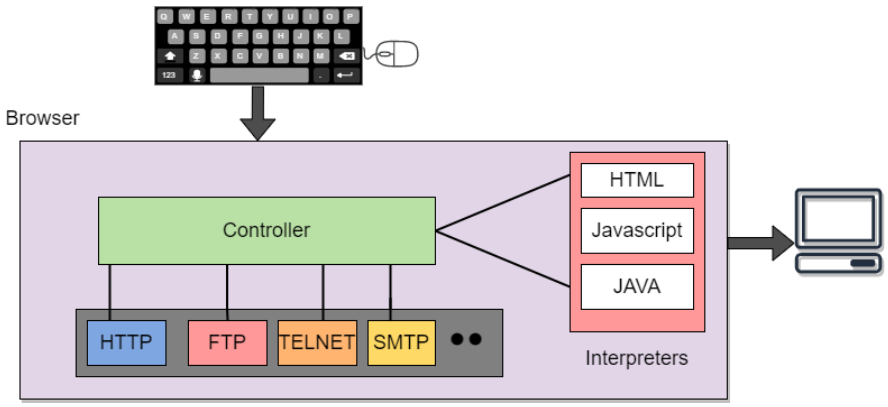


Now we will cover the components of WWW in detail -

1. Client/Browser:

The Client/Web browser is basically a program that is used to communicate with the webserver on the Internet.

- Each browser mainly comprises of three components and these are:
  - Controller
  - Interpreter
  - Client Protocols
- The Controller mainly receives the input from the input device, after that it uses the client programs in order to access the documents.
- After accessing the document, the controller makes use of an interpreter in order to display the document on the screen.
- An interpreter can be Java, HTML, javascript mainly depending upon the type of the document.
- The Client protocol can be FTP, HTTP, TELNET.



2. Server:

The computer that is mainly available for the network resources and in order to provide services to the other computer upon request is generally known as the **server.**

- The Web pages are mainly stored on the server.
- Whenever the request of the client arrives then the corresponding document is sent to the client.
- The connection between the client and the server is TCP.
- It can become more efficient through multithreading or multiprocessing. Because in this case, the server can answer more than one request at a time.

3. URL:

URL is an abbreviation of **the Uniform resource locator.**

- It is basically a standard used for specifying any kind of information on the Internet.
- In order to access any page, the client generally needs an address.
- To facilitate the access of the documents throughout the world HTTP generally makes use of Locators.

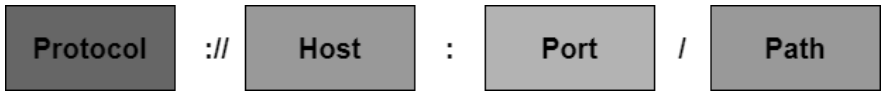
URL mainly defines the four things:

i. **Protocol**: It is a client/server program that is mainly used to retrieve the document. A commonly used protocol is HTTP.

ii. **Host Computer**: It is the computer on which the information is located. It is not mandatory because it is the name given to any computer that hosts the web page.

iii. **Port**: The URL can optionally contain the port number of the server. If the port number is included then it is generally inserted in between the host and path and is generally separated from the host by the colon.

iv. **Path**: It indicates the pathname of the file where the information is located.



4. HTML:

HTML is an abbreviation of Hypertext Markup Language.

- It is generally used for creating web pages.
- It is mainly used to define the contents, structure, and organization of the web page.

5. XML:

XML is an abbreviation of Extensible Markup Language. It mainly helps in order to define the common syntax in the semantic web.

Advantages of WWW:

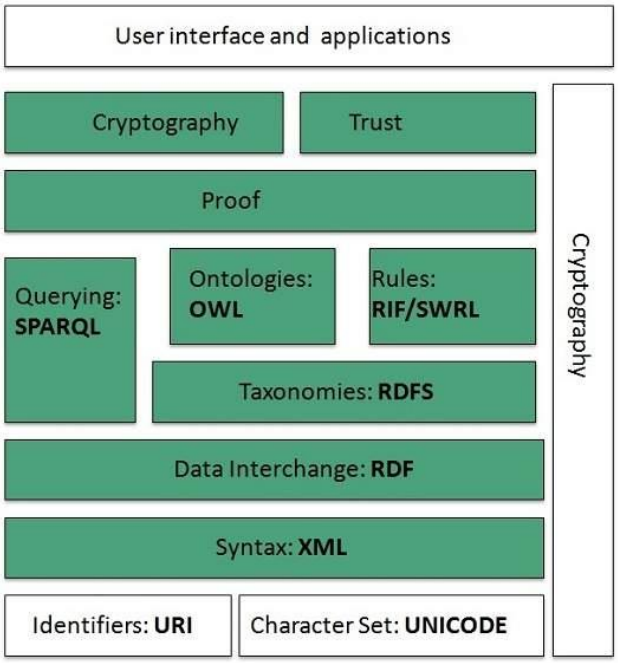
- It mainly provides all the information for Free.
- Provides rapid Interactive way of Communication.
- It is accessible from anywhere.
- It has become the Global source of media.
- It mainly facilitates the exchange of a huge volume of data.

Disadvantages of WWW:

- It is difficult to prioritize and filter some information.
- There is no guarantee of finding what one person is looking for.
- There occurs some danger in case of overload of Information.
- There is no quality control over the available data.
- There is no regulation.

WWW Architecture:

WWW architecture is divided into several layers as shown in the following diagram:



Identifiers and Character Set: **Uniform Resource Identifier (URI)** is used to uniquely identify resources on the web and **UNICODE** makes it possible to built web pages that can be read and write in human languages.

Syntax: **XML (Extensible Markup Language)** helps to define common syntax in semantic web.

Data Interchange: **Resource Description Framework (RDF)** framework helps in defining core representation of data for web. RDF represents data about resource in graph form.

Taxonomies: **RDF Schema (RDFS)** allows more standardized description of **taxonomies** and other **ontological** constructs.

Ontologies: **Web Ontology Language (OWL)** offers more constructs over RDFS. It comes in following three versions:

- OWL Lite for taxonomies and simple constraints.
- OWL DL for full description logic support.
- OWL for more syntactic freedom of RDF

Rules: **RIF** and **SWRL** offers rules beyond the constructs that are available from **RDFs** and **OWL.** Simple Protocol and **RDF Query Language (SPARQL)** is SQL like language used for querying RDF data and OWL Ontologies.

Proof: All semantic and rules that are executed at layers below Proof and their result will be used to prove deductions.

Cryptography: **Cryptography** means such as digital signature for verification of the origin of sources is used.

User Interface and Applications: On the top of layer **User interface and Applications** layer is built for user interaction.

Difference Between WWW and Internet:

Parameters of Comparison	**Internet**	**World Wide Web**
Founded	It was established in the later 1960s.	It was established in 1989.
Meaning	It is a massive global network made up of millions of small subnetworks.	It is a system of information where data is kept for public access.
Nature	There is a whole infrastructure there.	It is a specific service contained within an infrastructure.
Type	It emphasizes hardware.	The emphasis is on software.
Dependency	It is independent of the world wide web.	Here, the internet is a need.
Use	It may be used for a variety of things, including banking, entertainment, research, education, and navigation.	It is employed to gain access to resources all across the world.

HTTP:

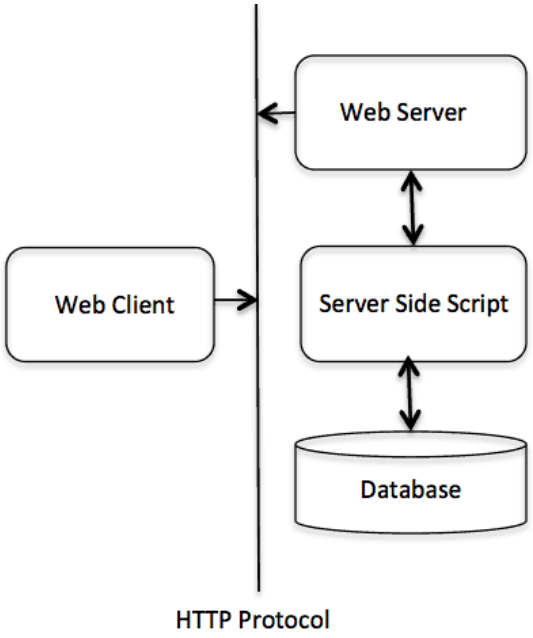
- HTTP stands for **HyperText Transfer Protocol**.
- **Tim Berner** invents it.
- It is a protocol used to access the data on the World Wide Web (www).
- The HTTP protocol can be used to transfer the data in the form of plain text, hypertext, audio, video, and so on.
- This protocol is known as HyperText Transfer Protocol because of its efficiency that allows us to use in a hypertext environment where there are rapid jumps from one document to another document.
- HTTP is similar to the FTP as it also transfers the files from one host to another host. But, HTTP is simpler than FTP as HTTP uses only one connection, i.e., no control connection to transfer the files.
- HTTP is used to carry the data in the form of MIME-like format.
- HTTP is similar to SMTP as the data is transferred between client and server. The HTTP differs from the SMTP in the way the messages are sent from the client to the server and from server to the client. SMTP messages are stored and forwarded while HTTP messages are delivered immediately.
- **HTTP/2** is the new version of HTTP. HTTP/3 is the latest version of HTTP, which is published in 2022.

Features of HTTP:

- **Connectionless protocol:** HTTP is a connectionless protocol. HTTP client initiates a request and waits for a response from the server. When the server receives the request, the server processes the request and sends back the response to the HTTP client after which the client disconnects the connection. The connection between client and server exist only during the current request and response time only.
- **Media independent:** HTTP protocol is a media independent as data can be sent as long as both the client and server know how to handle the data content. It is required for both the client and server to specify the content type in MIME-type header.
- **Stateless:** HTTP is a stateless protocol as both the client and server know each other only during the current request. Due to this nature of the protocol, both the client and server do not retain the information between various requests of the web pages.

Basic Architecture:

The following diagram shows a very basic architecture of a web application and depicts where HTTP sits:



The HTTP protocol is a request/response protocol based on the client/server based architecture where web browsers, robots and search engines, etc. act like HTTP clients, and the Web server acts as a server.

Client: The HTTP client sends a request to the server in the form of a request method, URI, and protocol version, followed by a MIME-like message containing request modifiers, client information, and possible body content over a TCP/IP connection.

Server: The HTTP server responds with a status line, including the message's protocol version and a success or error code, followed by a MIME-like message containing server information, entity meta information, and possible entity-body content.

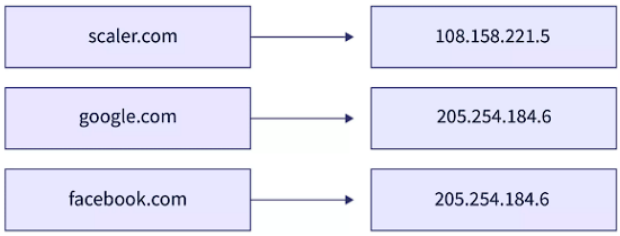
How Does Hypertext Transfer Protocol Work?

Below is a step-by-step discussion of how the Hypertext transfer protocol works.

1. Extracting the IP address of a URL (Uniform Resource Locator):

In our browser, we enter the URL address of the website we want to visit. Once we enter the URL address of the website, the browser with the help of the DNS extracts the IP address corresponding to the URL address that is entered.

The DNS contains the mapping of the URLs along with their corresponding IP addresses.



Domain Name System (DNS)

2. Sending request to the server to access the webpage and receiving response:

Once we get the IP address of the website, the browser sends an HTTP request to the server to extract the HTML webpage corresponding to the IP address. This request is sent over PORT 80 using TCP. Once the server receives this HTTP request, it responds back with an HTTP response. This HTTP response consists of the information related to the HTML page corresponding to the IP address of the website.

HTTP Request and Response

3. Receiving HTTP response and displaying the webpage:

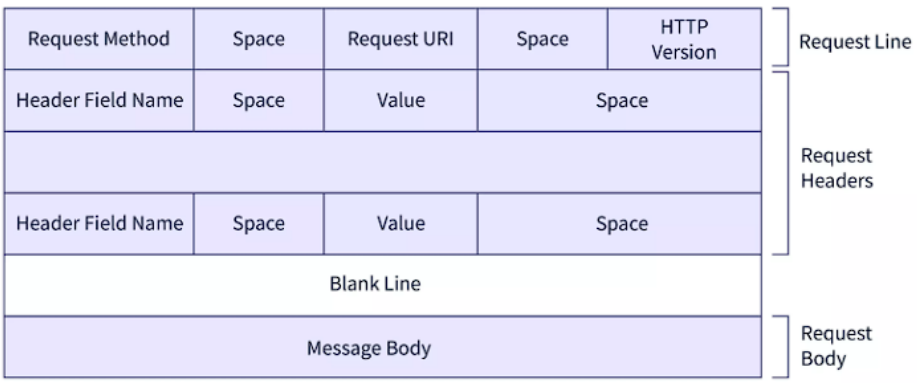
The browser receives the HTML information for the website along with the response and hence, it processes and displays the HTML page on the browser. Finally, the users can see an HTML page for the URL that they entered.

What is in an HTTP Request?

In order to fetch the webpage corresponding to a given IP address, the browser sends a request (called as HTTP request) to the web server.

An HTTP request consists of 3 parts i.e. a request line, request headers, and request body.

Given below is the structure of an HTTP request.



Request Line:

An HTTP request-line consists of 3 parts which are discussed below.

1. **Request Method** A request method defines what type of request is needed to send to the web server. For example, if we want to fetch something from the web server, we use a GET request method. If we want to send something from the client to the web server, we use POST request method.
2. **Request URI** It is the the URI of the website/destination we want to reach. For example "http://scaler.com/".
3. **HTTP Version** It states the version of the HTTP we are using. For example HTTP 1.0, HTTP 1.1.

Request Headers:

Request headers contain additional information about the resource/data that is to be fetched from the web server.

For example, if we want the data in the plaintext format, we can specify that information in the HTTP headers.

We can also specify the information related to the client using headers like the browser it is using etc.

We can send more than one HTTP header along with an HTTP request to specify the additional information that is to be sent along with an HTTP request.

Request Body:

It is an optional part of an HTTP request. If there is a requirement to send some data along with an HTTP request, we send that data along with the HTTP Body.

For example: If we want to send the details of a user which is taken through a form at the client side to the web server, we can send this data in the HTTP body.

What is in an HTTP Request Body?

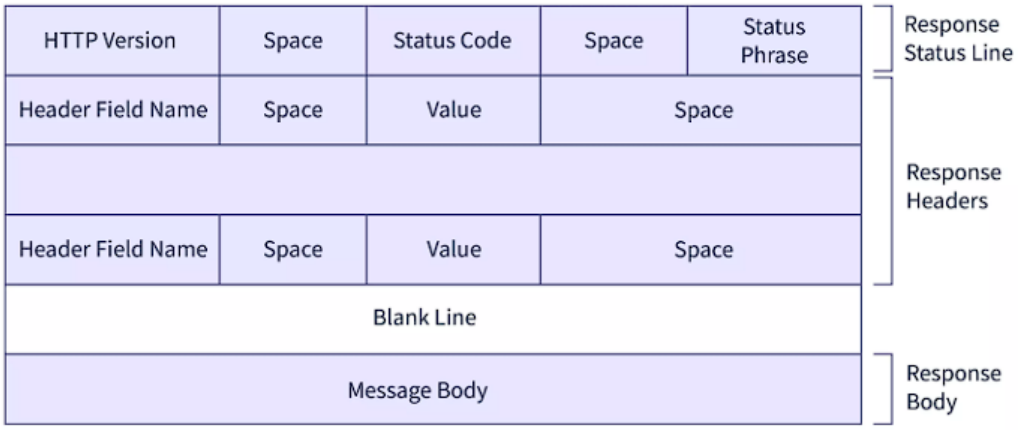
As discussed above, an HTTP body is an optional part of an HTTP request. If there is a requirement to send some data along with an HTTP request, we send that data along with the HTTP Body.

For example: If we want to send details of the user which is taken through a form at the client side to the web server. We can send this data in the HTTP body.

What is in an HTTP Response?

After the client/browser sends an HTTP request to the web server, the server responds back to the browser with an HTTP response. This response consists of all the data that is requested by the client in the HTTP request.

Below is the structure of an HTTP response.



An HTTP response consists of 3 parts:

1. Response Status Line:

An HTTP request line consists of 3 parts which are discussed below.

1. **HTTP Version** It states the version of theHTTPwe are using. For example HTTP 1.0, HTTP 1.1.
2. **Status Code** It is a three-digit code that tells the status of the HTTP response. For example code 200 stands for a successful HTTP response.
3. **Status Phrase** It is a short description of the status code.

2. Response Headers:

The response headers are used to specify the additional information related to the response data and the server.

3. Response Body:

It is an optional part of the HTTP response. The data which is requested by the client using the HTTP request is sent from server to client by keeping it inside the response body.

What is an HTTP Status Code?

An HTTP status code is a three-digit code that specifies the status of the HTTP response. Some of the commonly seen HTTP status codes along with their description are given below.

Status Code	Phrase	Description
200	OK	A successful request.
404	Page not found	The requested document is not present.
500	Internal server error	An error such as a server crash has occurred.
400	Bad request	There is some syntax error in the code.

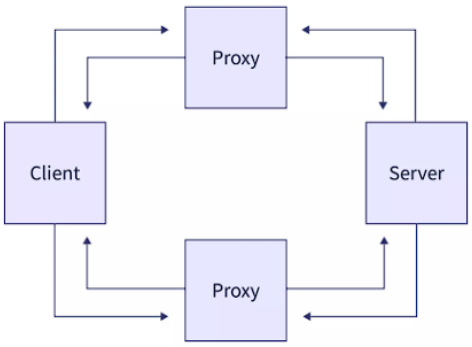
What are HTTP Response Headers?

As discussed in the above section as well, HTTP response headers are used to specify the information related to the data that is to be sent to the client through the HTTP response.

We can also specify the information related to the server like the server's local clock time, server name, etc. using the response headers.

Components of HTTP-based Systems:

An HTTP-based system consists of 3 parts i.e. a client, a server, and numerous proxies lying in between the client and the server.



1. **Client** A client is the one that sends the HTTP request so as to fetch information from the server at the backend. **For example:** The web browser you are using to surf the internet is the client that is actually requesting the webpages from the web server.
2. **Server** A server is present at the backend which receives requests from the client. It sends an HTTP response back to the client along with the data which is requested by the client.

3. **Proxy** In order to make the process of the request and response faster, we use proxy servers. Proxy servers are generally smaller servers that contain some of the information that is present in the main server.

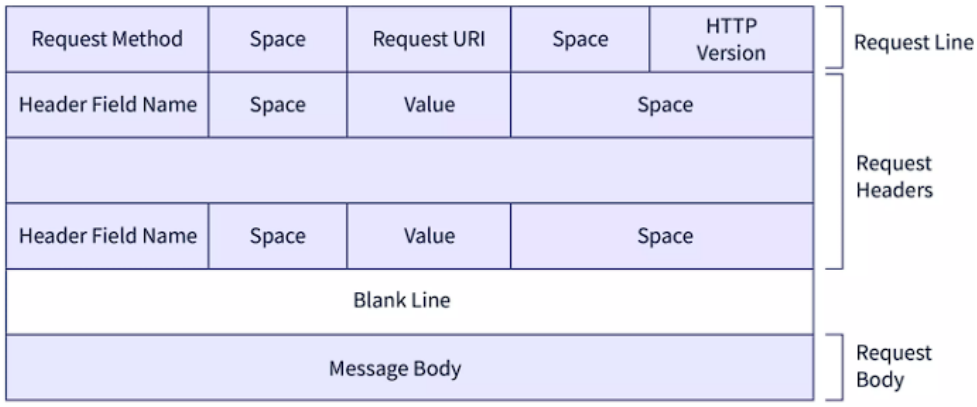
The **first request** from the client is always sent to a proxy server that is actually closest to the web client. If the requested data is present in the proxy server, the proxy server responds back to the client along with the requested data. But if the requested data is not present in the proxy server, the client sends the HTTP request to the main server.

HTTP Messages:

We discussed the parts of the HTTP request (Request Line, Request Headers, Request Body) and the parts of the HTTP response (Response Status-Line, Response Headers, Response Body) in the above sections. This combined information in the HTTP request and theHTTPresponse are called a request message and response message respectively.

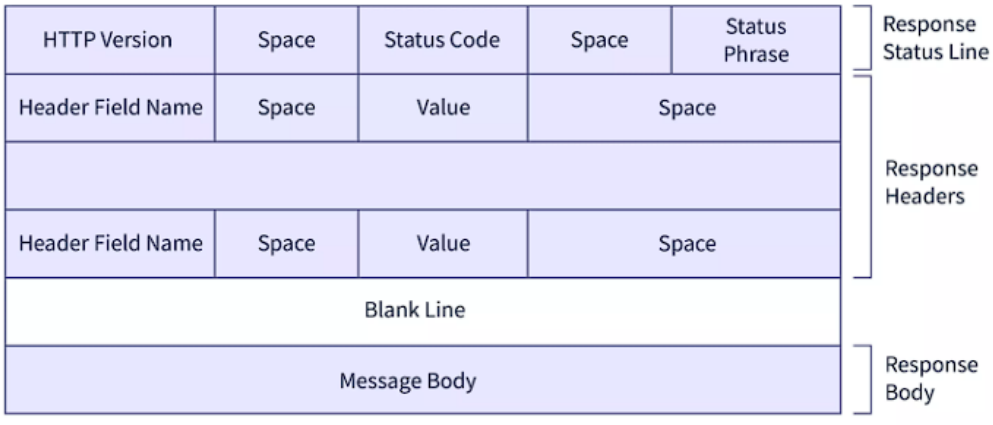
Below is the structure of a request message and a response message.

Request Message:



Request Message

Response Message:



Response Message

HTTP vs HTTPS:

Hypertext Transfer Protocol (HTTP)	Hypertext Transfer Protocol Secure (HTTPS)
1. HTTP is a protocol used to transfer data over the internet.	1. HTTPS is also used to transfer data over the internet but it is more secure than HTTP.
2. It does not use cryptographic algorithms to encrypt the transmitted data.	2. It uses cryptographic algorithms like SSL (Secure Socket Layer) and TLS (Transport Layer Security) which makes it more secure than HTTP
3. The connection of HTTP takes place on PORT 80.	3. The connection of HTTP takes place on PORT 443.
4. HTTP is faster than HTTPS since it does not include any additional steps for the cryptographic algorithms.	4. HTTPS is slower than HTTP since it includes additional steps for the cryptographic algorithms.

History of HTTP:

Tim Berners Lee and his team at CERN get credit for inventing original HTTP and associated technologies.

- **HTTP version 0.9:** This was the first version of HTTP which was introduced in 1991.
- **HTTP version 1.0:** In 1996, RFC 1945 (Request For Comments) was introduced in HTTP version 1.0.
- **HTTP version 1.1:** In January 1997, RFC 2068 was introduced in HTTP version 1.1. Improvements and updates to the HTTP version 1.1 standard were released under RFC 2616 in June 1999.
- **HTTP version 2.0:** The HTTP version 2.0 specification was published as RFC 7540 on May 14, 2015.
- **HTTP version 3.0:** HTTP version 3.0 is based on the previous RFC draft. It is renamed as Hyper-Text Transfer Protocol QUIC which is a transport layer network protocol developed by Google.

Cookies in HTTP: An HTTP cookie (web cookie, browser cookie) is a little piece of data that a server transmits to a user's web browser. When making subsequent queries, the browser may keep the cookie and transmit it back to the same server. An HTTP cookie is typically used, for example, to maintain a user's login state, to determine whether two requests originate from the same browser. For the stateless HTTP protocol, it retains stateful information.

Advantages of HTTP:

- Memory usage and CPU usage are low because of fewer simultaneous connections.
- Since there are few TCP connections hence network congestion is less.
- Since handshaking is done at the initial connection stage, then latency is reduced because there is no further need for handshaking for subsequent requests.
- The error can be reported without closing the connection.
- HTTP allows HTTP pipe-lining of requests or responses.

Disadvantages of HTTP:

- HTTP requires high power to establish communication and transfer data.
- HTTP is less secure because it does not use any encryption method like HTTPS and use TLS to encrypt regular HTTP requests and response.
- HTTP is not optimized for cellular phones and it is too gabby.
- HTTP does not offer a genuine exchange of data because it is less secure.
- The client does not close the connection until it receives complete data from the server; hence, the server needs to wait for data completion and cannot be available for other clients during this time.

FTP:

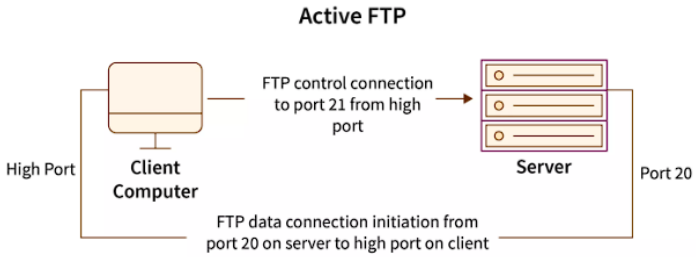
- FTP stands for **File transfer protocol**.
- FTP is a standard internet protocol provided by TCP/IP used for transmitting the files from one host to another.
- It is mainly used for transferring the web page files from their creator to the computer that acts as a server for other computers on the internet.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

Objectives of FTP:

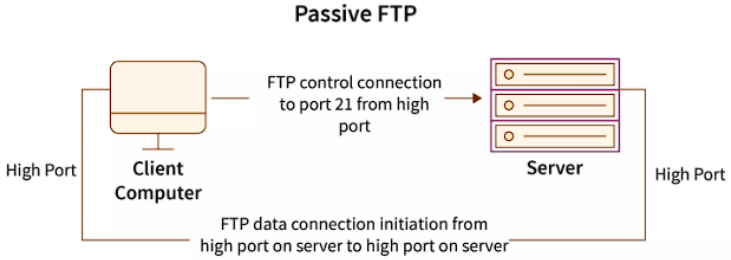
- It provides the sharing of files.
- It is used to encourage the use of remote computers.
- It transfers the data more reliably and efficiently.

How Does File Transfer Protocol (FTP) Work?

- FTP works by connecting two computers that are attempting to interact with each other. One connection handles the commands and answers that get sent between the two clients, while the other channel handles data transfer. During an FTP transmission, the computers, servers, or proxy servers communicate using four commands. These are *send*, *get*, *change directory*, and *transfer*.
- FTP transfers files in three separate modes: **block**, **stream**, and **compressed**. The stream mode allows FTP to manage information in a string of data without any boundaries between them. The block mode divides the data into blocks, and in the compressed mode, FTP uses an algorithm called the *Lempel-Ziv* to compress the data.
- **A typical FTP transfer works like this:**
    1. Firstly, a user must log in to the FTP server. There are some servers that make some or all of their content available without requiring a login; they are referred to as anonymous FTP.
    2. When a user raises a request to download a file, the client starts a discussion with the server.
    3. A client can use FTP to download, move, delete, upload, rename, and copy files on a server.
- **FTP sessions work in both active and passive modes:**
    - **Active mode:** When a client requests a session over a command channel, the server establishes a data connection with the client and starts sending data.

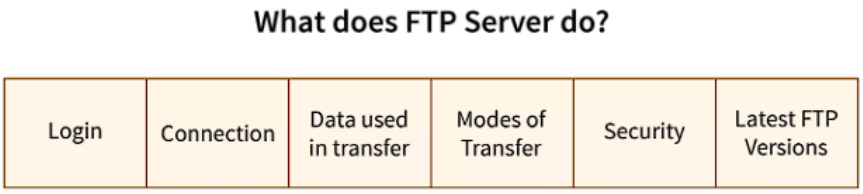


    - **Passive mode:** The command channel is used by the server to transfer the information needed by the client to start a data channel. Because the client initiates all connections in passive mode, it can easily traverse firewalls and network address translation gateways.



FTP can be accessed via a basic command-line interface (from a console or terminal window in Microsoft Windows, Apple macOS, or Linux) or with a dedicated graphical user interface. FTP clients can also be used with web browsers.

What is the Purpose of FTP, and Why is It Important?



File Transfer Protocol is a type of standard network protocol that allows for large-scale file transfers through IP networks. Other systems, such as email or an HTTP web service, can manage file and data transmission in the absence of FTP, but they lack the clarity of focus, accuracy, and control that FTP provides.

FTP is a file transfer protocol that is used to send files between systems and has a variety of uses.

Some of the uses of FTP are:

- **Backup:** Individual users can use FTP to store their data by uploading it onto a server. Also, backup services can use FTP to backup data from one location to a secure backup server running FTP services.
- **Replication:** FTP can also be used for replication and is similar to backup. It involves duplicating data from one computer to another but takes a more extensive approach to increase availability and resilience.
- **Access and data loading:** FTP is also often used to access shared web hosting and cloud services to load data onto a remote server.

Types of FTP:

There are different ways through which a server and a client do a file transfer using FTP. Some of them are mentioned below:

- **Anonymous FTP:** Anonymous FTP is enabled on some sites whose files are available for public access. A user can access these files without having any username or password. Instead, the username is set to anonymous, and the password is to the guest by default. Here, user access is very limited. For example, the user can be allowed to copy the files but not to navigate through directories.
- **Password Protected FTP:** This type of FTP is similar to the previous one, but the change in it is the use of username and password.

- **FTP Secure (FTPS):** It is also called as FTP Secure Sockets Layer (FTP SSL). It is a more secure version of FTP data transfer. Whenever FTP connection is established, Transport Layer Security (TLS) is enabled.
- **FTP over Explicit SSL/TLS (FTPES):** FTPES helps by upgrading FTP Connection from port 21 to an encrypted connection.
- **Secure FTP (SFTP):** SFTP is not a FTP Protocol, but it is a subset of Secure Shell Protocol, as it works on port 22.

Types of Connection in FTP:

1. Control Connection: For sending control information like user identification, password, commands to change the remote directory, commands to retrieve and store files, etc., FTP makes use of a control connection. The control connection is initiated on port number 21.

2. Data connection: For sending the actual file, FTP makes use of a data connection. A data connection is initiated on port number 20. FTP sends the control information out-of-band as it uses a separate control connection. Some protocols send their request and response header lines and the data in the same TCP connection. For this reason, they are said to send their control information in-band. HTTP and SMTP are such examples.

FTP Session: When an FTP session is started between a client and a server, the client initiates a control TCP connection with the server side. The client sends control information over this. When the server receives this, it initiates a data connection to the client side. But the control connection remains active throughout the user session. As we know HTTP is stateless. But FTP needs to maintain a state about its user throughout the session.

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.

Some of the following FTP clients that are available:

Client	Description
FileZilla	It is a free FTP client that supports FTP, FTPS, and SFTP on Windows, macOS, and Linux.
Transmit	It is a macOS FTP client that supports FTP and SSH.
WinSCP	It is a free (open-source) SSH File Transfer Protocol for Windows FTP clients that supports FTP, SSH, and SFTP.
WS_FTP	It is yet another FTP client for Windows that supports SSH.

FTP Commands:

FTP commands	Description
!	It is used to switch between the operating system and FTP.
?	It is used to print the FTP command's details.
append	It is used to append the two files together.
ascii	It is used to change the file transfer mode to ASCII. It is the default mode for most FTP programs.
binary	It sets the file transfer mode to binary.
bell	This command sounds a bell when it completes any command.
bye	It is used to exit and end the ftp session.
cd	This command alters the remote system's directory.
close	It is used to put an end to a remote system's FTP connection.
dir	It is used to get a list of contents in the remote directory.
delete	It's used to remove a file from the currently open remote directory.
debug	It is used to turn on and off debugging mode. This command does not necessitate a remote system connection.
disconnect	It is used to end the FTP session.
ftp	It is used to access the FTP interpreter.
get	It is used to copy data (a file) from a server to a client device.
hash	It enables the client to request a file's cryptographic hash from a server-FTP process.

help	It is used to show local assistance information. This command does not necessitate a remote system connection.
lcd	It is used to modify the directory on your local system.
literal	It sends the argument to the remote system.
ls	It is used to display a list of the names of the files in the current remote directory.
mdelete	It is used to delete a large number of files at once.
mdir	It is used to list the contents of multiple remote directories.
mget	It is used to copy multiple data files from a remote computer to a local computer.
mkdir	It is used to create a new directory within the current remote directory.
mls	This command displays a list of the file names in the various server directories.
mput	This command copies a large number of files from a user device to a server device.
open	This command is used to establish a link with another computer system.
put	This command copies a file from a user device to a server device.
pwd	This command is used to find the path name of the remote system's local directory.
quit	It is used to end an FTP session (same as bye).
quote	It sends the argument to the remote system.
recv	This command retrieves a file from a remote machine.
remotehelp	It is used to display the information of the remote system.
rename	This command renames the system's file.
rmdir	This command is used to delete a directory from the local remote directory.
send	This command is used to send a file.
status	This command displays the system's current status.
type	It is used to specify the file transfer type.
user	This command transmits the new user's information.
verbose	It is used to turn verbose on/off.

FTP Security:

- FTP was not designed to be secure. It is widely regarded as an insecure protocol since it uses clear-text usernames and passwords for authentication and does not use encryption. Sniffing, spoofing, and brute force attacks, among other basic attack methods, are all possible with data sent via FTP.
- There are numerous ways that are commonly used to address these challenges and secure FTP usage. FTPS is an FTP extension that can encrypt connections at the request of the client. **Transport Layer Security (TLS)**, **Secure Socket Layer (SSL)**, and **SSH File Transfer Protocol (also known as Secure File Transfer Protocol or SFTP)** are frequently used as more secure alternatives to FTP because they use encrypted connections.

FTP Data Types:

The data type of a file, which determines how the file is represented overall, is the first piece of information that can be provided about it. The FTP standard specifies the following four categories of data:

- **ASCII:** Describes an ASCII text file in which each line is indicated by the previously mentioned type of end-of-line marker.
- **EBCDIC:** For files that use IBM's EBCDIC character set, this type is conceptually identical to ASCII.
- **Image:** This is the "black box" mode I described earlier; the file has no formal internal structure and is transferred one byte at a time without any processing.
- **Local:** Files containing data in logical bytes with a bit count other than eight can be handled by this data type.

FTP Replies:

Some of the FTP replies are:

- 200 – Command okay.
- 530 – Not logged in.
- 331 – User name okay, need a password.
- 221 – Service closing control connection.
- 551 – Requested action aborted: page type unknown.
- 502 – Command not implemented.
- 503 – Bad sequence of commands.
- 504 – Command not implemented for that parameter.

FTP Transmission Modes:

FTP can transfer a file across the data connection using one of the three given modes:

1. Stream Mode: Stream Mode is the default mode of transmission used by FTP. In this mode, the File is transmitted as a continuous stream of bytes to TCP.

If the data is simply in the form of the stream of bytes, then there is no need for End-of-File, Closing of data connection by the sender is considered as EOF or end-of-file. If the data is divided into records (that is the record structure), each record has an I-byte of EOR (end-of-record).

2. Block Mode: Block mode is used to deliver the data from FTP to TCP in the form of blocks of data. Each block of data is preceded by 3 bytes of the header where the first byte represents the block descriptor while the second and third byte represents the size of the block.

3. Compressed Mode: In this mode, if the file to be transmitted is very big then the data can be compressed. This method is normally used in Run-length encoding. In the case of a text file, usually, spaces/blanks are removed. While in the case of the binary file, null characters are compressed.

Advantages of FTP:

- File sharing also comes in the category of advantages of FTP in this between two machines files can be shared on the network.
- Speed is one of the main benefits of FTP.
- Since we don't have to finish every operation to obtain the entire file, it is more efficient.
- Using the username and password, we must log in to the FTP server. As a result, FTP might be considered more secure.
- We can move the files back and forth via FTP. Let's say you are the firm manager and you provide information to every employee, and they all reply on the same server.

Disadvantages of FTP:

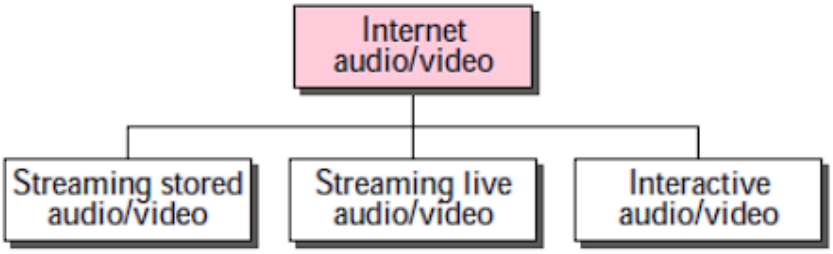
- File size limit is the drawback of FTP only 2 GB size files can be transferred.
- More then one receivers are not supported by FTP.
- FTP does not encrypt the data this is one of the biggest drawbacks of FTP.
- FTP is unsecured we use login IDs and passwords making it secure but they can be attacked by hackers.

HTTP vs FTP: Comparison Chart:

Comparison	HTTP	FTP
Full form	HTTP stands for Hypertext Transfer Protocol.	FTP stands for File Transfer Protocol.
Use case	HTTP works on client-server architecture and is used to transfer web pages between a client and a server.	FTTP works on client-server architecture and is used to transmit files among different hosts.
Connection Type	HTTP connection is built upon TCP (Transmission Control Protocol)	FTP connection is built upon TCP (Transmission Control Protocol)
Types of connections	HTTP only establishes a data connection.	FTP establishes two types of connections, command connection and data connection.
PORT used	HTTP establishes a TCP connection on PORT 80.	FTP establishes command connection on PORT 21 and data connection on PORT 20.
Authentication	HTTP does not support authentication.	FTP supports authentication. The client needs to log in through the command connection and after this only, the clients can access the files.
TCP connection types	HTTP supports both persistent and non-persistent types of TCP connections.	FTP only supports a non-persistent type of TCP connection.
Protocol state	HTTP is a stateless protocol. But it can maintain states with the help of cookies.	FTP can maintain state.

Streaming Audio and Video:

We can divide audio and video services into three broad categories: streaming stored audio/video, streaming live audio/video, and interactive audio/video. Streaming means a user can listen (or watch) the file after the downloading has started.



In the first category, streaming stored audio/video, the files are compressed and stored on a server. A client downloads the files through the Internet. This is sometimes referred to as on-demand audio/video.

In the second category, streaming live audio/video refers to the broadcasting of radio and TV programs through the Internet.

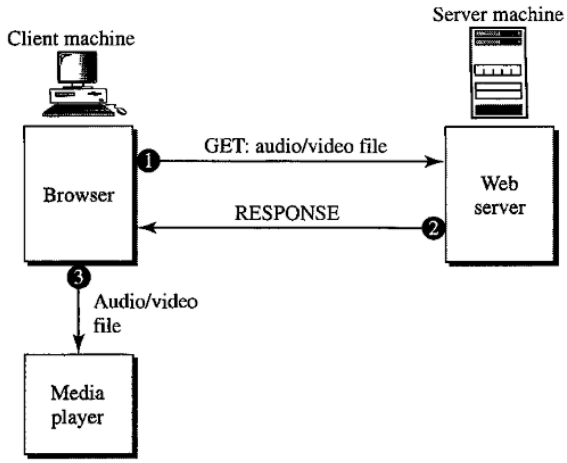
In the third category, interactive audio/video refers to the use of the Internet for interactive audio/video applications. A good example of this application is Internet telephony and Internet teleconferencing.

1. Streaming Stored Audio/Video:

Downloading these types of files from a Web server can be different from downloading other types of files.

First Approach: Using a Web Server

A compressed audio/video file can be downloaded as a text file. The client (browser) can use the services of HTTP and send a GET message to download the file. The Web server can send the compressed file to the browser. The browser can then use a help application, normally called a media player, to play the file. Fig. shows this approach.



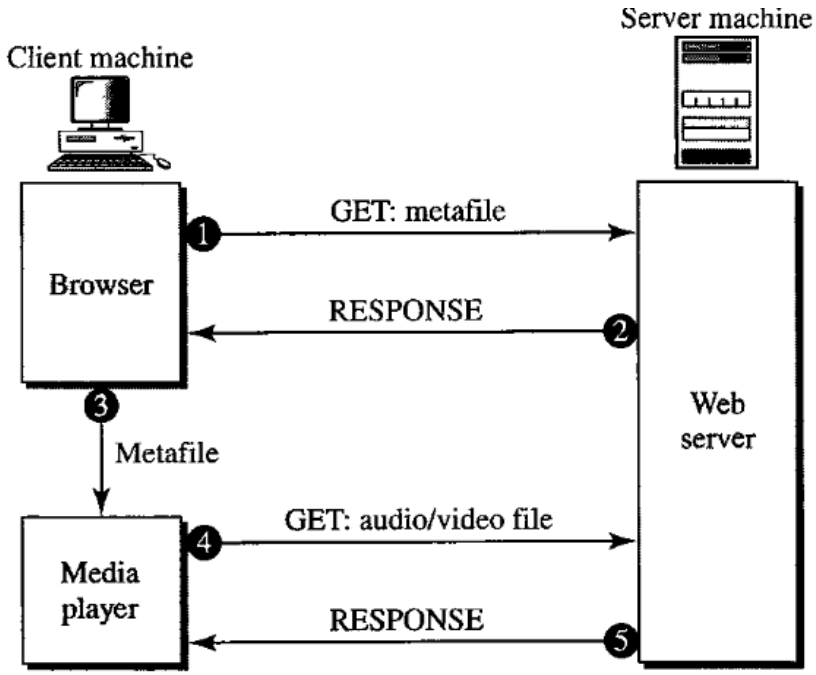
This approach is very simple and does not involve streaming. However, it has a drawback. An audio/video file is usually large even after compression. An audio file may contain tens of megabits, and a video file may contain hundreds of megabits. In this approach, the file needs to download completely before it can be played. Using contemporary data rates, the user needs some seconds or tens of seconds before the file can be played.

Second Approach: Using a Web Server with Metafile

In another approach, the media player is directly connected to the Web server for downloading the audio/video file. The Web server stores two files: the actual audio/video file and a metafile that holds information about the audio/video file.

Below Fig. shows the steps in this approach.

1. The HTTP client accesses the Web server by using the GET message.
2. The information about the metafile comes in the response.
3. The metafile is passed to the media player.
4. The media player uses the URL in the metafile to access the audio/video file.
5. The Web server responds.

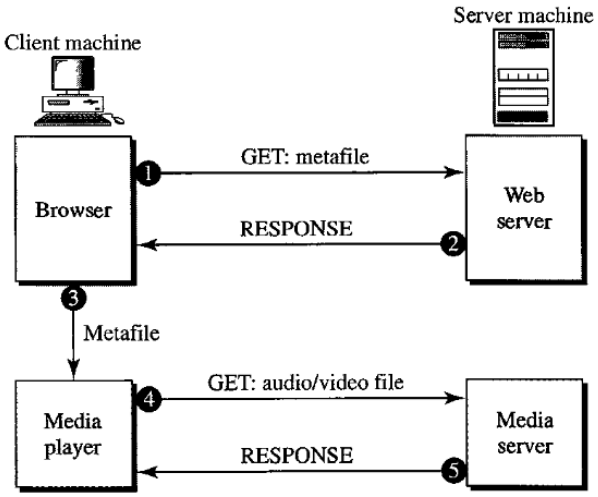


Third Approach: Using a Media Server

The problem with the second approach is that the browser and the media player both use the services of HTTP. HTTP is designed to run over TCP.

This is appropriate for retrieving the metafile, but not for retrieving the audio/video file. The reason is that TCP retransmits a lost or damaged segment, which is counter to the philosophy of streaming. We need to dismiss TCP and its error control; we need to use UDP. However, HTTP, which accesses the Web server, and the Web server itself are designed for TCP; we need another server, a media server.

Below Fig. shows the concept.



1. The HTTP client accesses the Web server by using a GET message.

2. The information about the metafile comes in the response.

3. The metafile is passed to the media player.

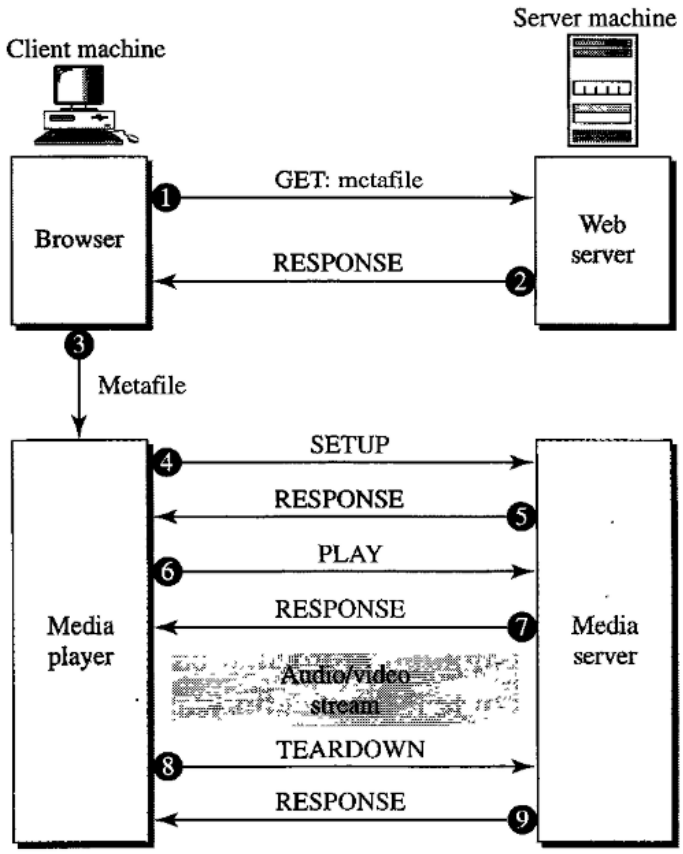
4. The media player uses the URL in the metafile to access the media server to download the file. Downloading can take place by any protocol that uses UDP

5. The media server responds.

Fourth Approach: Using a Media Server and RTSP

The Real-Time Streaming Protocol (RTSP) is a control protocol designed to add more functionalities to the streaming process. Using RTSP, we can control the playing of audio/video. RTSP is an out-of-band control protocol that is similar to the second connection in FTP.

Below Fig. shows a media server and RTSP.



1. The HTTP client accesses the Web server by using a GET message.

2. The information about the metafile comes in the response.

3. The metafile is passed to the media player.

4. The media player sends a SETUP message to create a connection with the media server.

5. The media server responds.

6. The media player sends a PLAY message to start playing (downloading).

7. The audio/video file is downloaded by using another protocol that runs over UDP.

8. The connection is broken by using the TEARDOWN message.

9. The media server responds.

The media player can send other types of messages. For example, a PAUSE message temporarily stops the downloading; downloading can be resumed with a PLAY message.

2. Streaming Live Audio/Video:

Streaming live audio/video is similar to the broadcasting of audio and video by radio and TV stations. Instead of broadcasting to the air, the stations broadcast through the Internet. There are several similarities between streaming stored audio/video and streaming live audio/video. They are both sensitive to delay; neither can accept retransmission. However, there is a difference. In the first application, the communication is unicast and on-demand. In the second, the communication is multicast and live. Live streaming is better suited to the multicast services of IP and the use of protocols such as UDP and RTP (discussed later). However, presently, live streaming is still using TCP and multiple unicasting instead of multicasting. There is still much progress to be made in this area.

3. Real-Time Interactive Audio/Video:

In real-time interactive audio/video, people communicate with one another in real time, The Internet phone or voice over IP is an example of this type of application. Video conferencing is another example that allows people to communicate visually and orally.

Characteristics of Real-Time Audio/Video Communication:

1. **Time Relationship**: Real-time audio/video communication systems maintain a synchronous relationship between the transmission and reception of audio/video data. This ensures that participants experience minimal delay, enabling natural interactions and conversations.
2. **Timestamp**: Timestamps are used to mark the timing of audio and video packets. These timestamps help synchronize playback on the receiving end and ensure that audio and video streams remain aligned.
3. **Playback Buffer**: A playback buffer is employed on the receiving end to store incoming audio/video packets temporarily before playback. The size of the buffer is dynamically adjusted based on network conditions to mitigate jitter and maintain smooth playback.
4. **Ordering**: Audio and video packets are transmitted and received in the correct order to reconstruct the original stream. Maintaining packet order is essential for coherent playback and synchronization between audio and video components.
5. **Multicasting**: Real-time communication systems may utilize multicasting to efficiently deliver audio/video streams to multiple recipients simultaneously. Multicasting minimizes network bandwidth usage by transmitting data only once to reach multiple recipients.

6. **Translation**: Real-time communication systems may support translation services to facilitate communication between participants who speak different languages. Translation features can automatically convert spoken or written content from one language to another in real-time.
7. **Mixing**: In multi-party communication scenarios, mixing capabilities allow audio streams from multiple participants to be combined into a single stream for transmission. Mixing enables group conversations and conference calls where all participants can hear each other simultaneously.
8. **Support from Transport Layer Protocol**: Real-time audio/video communication relies on transport layer protocols such as User Datagram Protocol (UDP) or Real-time Transport Protocol (RTP) for efficient and timely delivery of multimedia data. These protocols provide mechanisms for packetization, transmission, and error handling tailored to real-time communication requirements.