## UNIT – 01

## Internetworking concepts

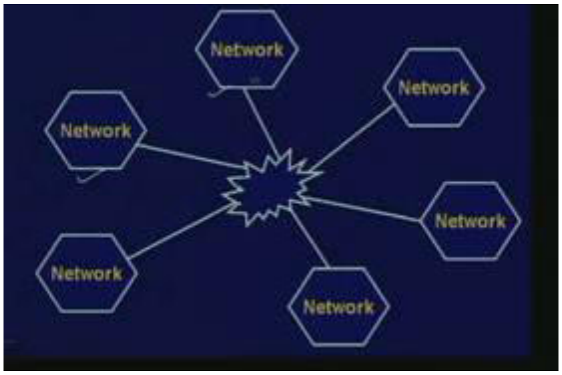
### Unit-01/Lecture-01

**History of the internet**

Now as the definitions goes and internet can be defined as the network which is formed by the co-operative interconnection of a large number of computer networks. Now since internet is formed by the interconnection of a number of computer networks, sometimes it is also known as a network of networks. Now here there are a few interesting things that work with internet. The first and foremost is that there is no single owner of the internet. Now just unlike a network that you can see in your organization, may be your organization owns the network. In contrast internet, you cannot identify a single owner who owns or who administrates or manages the whole network. Suppose you are a member of the internet group which means you are computer is also connected to the internet may be it is so as simple dial up telephone line from your residence. Now in that case you are also a part of the internet.

Every person who makes a connection to the internet becomes a part of the owner. So I repeat there is no central administration or central authority to the internet. Now just talking about the way internet has emerged as a network of network. If you look into the history of computer networks you will find that computer networks initially started in the later 60s. They were mainly some clusters of some computers in different laboratories and an organization for the main purpose was to connect several computers together. So as to achieve a number of goals like exchanging messages, sharing some information, etcetera. Now the networks that were at that time they share some characteristics like they were all proprietary in nature. The network that was there which was connecting a number of computers. All the computers were of the same type or they were for the same vendor. For example we could have had a network which comprised of only IBM computers; a network with only deck computers and so on.

Now with the passage of time people felt, that will these kinds of small networks have emerged in the different labtories and organizations like small islands. As the requirement or need of the people grew with time, they felt the necessity to connect these networks together. So that a user of network A can communicator with a user of network B. There should be a way of communicating between them. But of course in order to that one big problem needed to be solved, that was the problem of compatibility. I told earlier that initially the networks are mostly proprietary. A network of say that had connected a number of IBM computers was totally unknown entity to a HP computer. The HP computer did not understand how the IBM network would work. So there has to be a common binding force or a common standard that would all the computers across these networks to talk or communicated themselves. So this was one of the motivations.



Now talking about a network of networks internet looks something like this. We have a number of such networks which are connected in some way. Of course picture may not look like this. The diagram as I have shown it is a pure star network. But in practice this does not certainly look like this. The purpose of this diagram is to give you a logical picture of the internet. There are a number of different networks which are all connected through some basic back bone network. The central place this portion this is the central back bone network which connects all the networks together.

So the internet, what is it actually in practice? Well internet is not just the network; internet is not just the programs or the applications that the users of the internet use. Internet is not just some document or some resources which are available on some computer in the network. But rather it comprises of a number of different things. Firstly it is a community of people who use and develop the network. So people are also part of the internet. So without people internet would not have existed. Internet also consists of a collection of resources. This is very important because if this resource were not there possible so many people's would not have used the internet. Now by having a connectivity or this network of networks established, what we have is that these resources can be reached from anywhere in the internet.

So we have some kind of network and over the network you can reach any resource you want to from any other place. This is a basic idea and internet also provides a set up to facilitate the collaboration. Now when I say collaboration, this can be simple messaging facilities like electronic mail or it can be some more concerted collaborative efforts like having bulletin board system or somebody can pores some problems or a discussion forum through which you can start an open discussion on some topic, which of course you want to start a discussion on. So particularly this kind of collaboration is very useful in a number of you can say areas in particular among the members of the research and educational communities. They find this kind of collaborative facility invaluable.

Suppose I have a problem I am unable to solve, if I put it in the discussion forum may be somewhere sitting somewhere else would be able to solve my problem and give me a response of the solution to the problem. And talking about the about common standard that bangs on the network together. There is a standard protocol called TCP and IP. TCP stands for Transmission Control Protocol and IP stands for Internet Protocol. So as the scenario is there today, well any computer or any network if it wants to get connected to the internet the computer or the network must understand the language of TCP IP. So all the message exchanges that go on in internet they use the syntax and the format of the TCP IP message packets. So if your computer understands TCP IP then your computer possible can become a part of the internet if you just have connectivity to it.

**The evaluation of the internet**

It started as early as in the 1950s where the US defence organization ARPA, it stands for Advanced Research Projects Agency. This started to network a number of computers that are funded by the in a very small way. So a few computers which are located in different paths of the country where provided with some sort of connectivity. So that they can communicate among themselves, now subsequently while it continued for some time like this in 1970s and beyond this ARPA. ARPA became to know as ARPA network advanced research project agency network. So ARPANET started to create a standard which is basically the predecessor to the TCP standard that we have today. So at that time the standard that was proposed that is not exactly TCP, but it is step in the right direction.

So it was a premiliminary protocol which through subsequent you can say refinents and modification became finally the TCP as we see today. In 1971 the universities were added to the network, the main purpose was that many of the defense funded research used to take place in the universities and ARPANET felt universities should be part of the network. And some basic internet services like telnet and FTP were made available. Now these you will be studying later in more detail. Now using telnet you can start remote session on a different computer sitting on your own computer. And using FTP File Transfer Protocol. You can transfer a file or a group of files between two machines. These were the basic facilities which were provided at that time for communicating between machines. In 1972 the first version of electronic mail came into you can say being coming to be first email message was sent during that time.

In 73, ARPANET spread its reach beyond US it connected to some sites or locations in England or Norway. So it is in 1973 ARPANET started to spread across continents. 1974 TCP was recognized as the standard and it was used for communication across a system of networks. So you can say that in 1974 people actually started to talk about having a number of networks. They all will be speaking the same language and TCP was the vehicle which was used to do or achieve this. Now 1982, the US department of defense it started building their own defense data network based on the same technology that were developed in ARPANET.

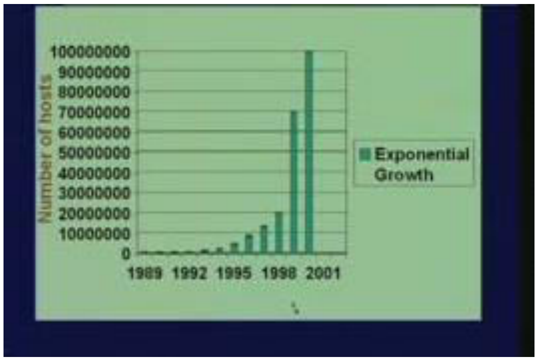
See this ARPANET as it has evolved it also brought along with it a number of different technologies, some protocols, some standards which people used and were actually able to communicate. So the US department of defense saw that here we have a technology which we can use to greet benefit for our case of application also though so they simply borrowed the technology for their own application and they stated using it. In 1983 this ARPANET actually got spit into ARPANET and there was a new network military network MILNET which is created which of course had some additional security requirements. That is why they got split into two.

In 83 we saw the internet which is very familiar to the internet we see today. So in 1983 the internet started to actually take shape as we see it. TCP IP was recognized as a standard. Then in 1986 National Science Foundation this stated another network NFSNET. Now NFSNET, the objective was very simple. The objective was to create very strong back bone network used to connect the regional networks. As you can see here this NFSNET is a system of regional networks which were connected over a back bone network. So here there are some networking devices called routers which were used. And routers belonging to the different regional networks were strongly connected among themselves and this constituted what is called a back bone network. And the main purpose of NFSNET was to create a very powerful back bone network which would provide these. So called back bone for future generation communication systems. In 1991 some new applications like Archie and gopher were released.

Well many of you may not have heard about these application action gophers. See at that time applications like the ftp file transfer protocol become very popular. People stated to keep a large number of resources on the different ftp servers and through ftp you can basically connect to that server and you can download the material whatever you want. It is very similar to the World Wide Web that we see today, I am sure most of you have used the internet through the World Wide Web through the browser and you know how it looks like. But at that time there was no user interface just we have to give command get a file and after you bring the file to a machine you can open it and see what it is. Now the main problem that people used to face that time is suppose I want some resources, say I want a particular document on a subject. So how do I know where that subject or that document is located? While if I know the address at that time some big ftp catalogs were published, I can look at the catalog and find out where these documents are located in this ftp server.

So let me connect there and see if I can find it out. So this Archive was developed as an ftp search engine. Well many of you are familiar with Google, Yahoo; the search engines which people today. So at that time Archie was the search engine through which given a topic you want to search for. Archie return a list of ftp sites where you could possible get that topic. And gopher was you can say a more intelligent version of Archie gopher showed you the documents in a category and sub category few it was like a global view, you can browse through categories and subcategories suppose flower rose, black rose. If you click on black rose you can get a list of place you can get information about black rose. So this gopher allowed you to browse ftp sites through well defined you can say category and subcategory view. Well gopher was a little more general it covered the ftp sites as well as some other non ftp sites also. But I am not going into those details right now.

Well 1992 the internet linked more than 17000 networks; there were about 3 million hosts. 1993 the World Wide Web application were launched. Today you know this World Wide Web is a defector standard anyone owning a computer. They use World Wide Web; they use either the internet explorer or Mozilla conqueror. Some kind of browser they use to access the World Wide Web 1995, the concept of networks service providers came into be and this network service providers start to offer service. But earlier you had to build your own network and it was your responsibility to get your network connected to the internet back bone. Now there are service providers, well in India there are service providers like VSNL, like Satyam, like Reliance, there are so many today. Now you can approach them. They will provide you a connection and it would be there responsibility to get your network connected to the internet back bone. So in 1995 you can understand you have about 30 million users. So the growth was fantastic. **Growth of Internet**



So this simple file shows you the exponential nature of growth that internet enjoys. You see in 1989, from about 10 million hosts, in 2001 you have 100 million hosts. So over a period of time, the numbers of nodes have increased in a fantastic way. So this exponential growth continues and as of today we have more than a billion hosts which are connected in the internet.

Now as the internet came in place and it became popular with the passage of time, there were a **number of internet applications** that were developed. Some were accepted widely, some they were not accepted and finally they got rejected. But here  listing a few of the internet application which became popular, in fact are still very popular, in fact all of them. The first one TELNET. I told you about this TELNET; allow a user to log into a remote computer and start so called remote session. That means I am sitting on my computer, I am running a program, and I am viewing the file. But actually everything is happening on the other computer. What I am seeing on my screen is the output of that program is coming straight away to my screen and I am having an illusion that I am actually sitting on that other computer; not on my computer.

Similarly you can have file transfer protocol. This I have, this also I have mentioned using this, you can transfer file between machines. Then you have the so called electronic mail. In fact today email is the single largest application which is used by people. While every person who gets connected to the internet invariable uses email. This email has become you can say part and partsell of daily life and is very fast making the so called surface mail which somebody also calls snail. In comparison with the speed so the surface mail is becoming obsolete and redundant in many cases. Many of us today prefer to send documents and reports by email rather than by surface mail. Gopher I have mentioned gopher although today you do not see many gopher application. But there was a time in 80s and mid 90s where gopher was very popular.

It was you cannot say just a previous version of the internet as we see today. So through gopher you could browse through categories and sub categories, you can access the document or the resource you want to access and you can use it in a convenient way. This internet relay chat is another application which is also very widely used by people. Through this internet relay chat you have you can say a medium through which a number of persons can communicate among themselves. Well if I type in some message, that message can be viewed by all members of the group with whom I am participating in that chat. So this chat is a very useful tool and if used in the proper context can prove itself to be extremely useful and beneficial, particularly in education. This Usenet news this is also very important application news. Well broadly this is like a discussion group news group or a discussion forum, say I have a discussion forum through I can start a discussion.

I can float a topic, the other members of the community who are also there in the forum can post, their reviews I can see them while I mentioned. If I have a problem I post the problem on the discussion forum and someone else may be coming up with a solution and posting it there. While the advantage is that I may be having my personal problem which I have posed and someone else has posted a solution to this. But there may be a number of other people who are having similar problem. So if they look through the mails that are getting exchanged through the discussion forum, they can possible also find a solution to their problem automatically by just following the discussion that is going on. And of course lastly the World Wide Web which is the most important application that we are today. In fact today World Wide Web can we treat it in as an umbrella and under the umbrella all other protocols can be used like electronic mail, like ftp, like news groups.

Everything else can be accessed under the same umbrella of World Wide Web. So World Wide Web will itself is an application and it also integrates other application together. So today we have a single common interface, we start a browser and through a browser we can have basically access everything. Now let us talk about something else. See this internet has evolved over a period of time. There are new protocols which have come up many of them have become so popular. Now suppose a new protocol comes up lets take an example say the email; electronic mail. Now you know today all the machines, all the computers, all the operating systems they support electronic mail and somehow they are compatible. If I send a mail from one computer to the other which reaches the destination so you can guess there has to be some kind of a standardization effort that goes on somewhere.

**Internetworking concepts**

Internetworking is a scheme for interconnecting multiple networks of dissimilar technologies

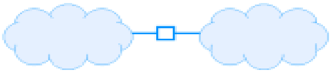
- Uses both hardware and software

- ✓ Extra hardware positioned between networks
- ✓ Software on each attached computer

- System of interconnected networks is called an internetwork or an internet

Routers

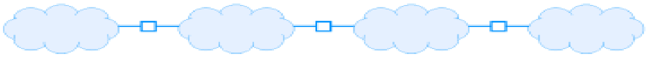
- A router is a hardware component used to interconnect networks
- A router has interfaces on multiple networks



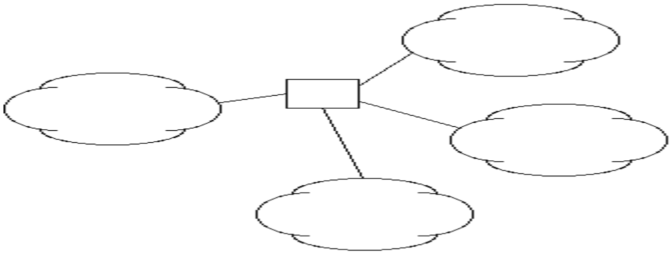
- Networks can use different technologies
- Router forwards packets between networks
- Transforms packets as necessary to meet standards for each network

Internet architecture

- An internetwork is composed of arbitrarily many networks interconnected by routers



- Routers can have more than two interfaces



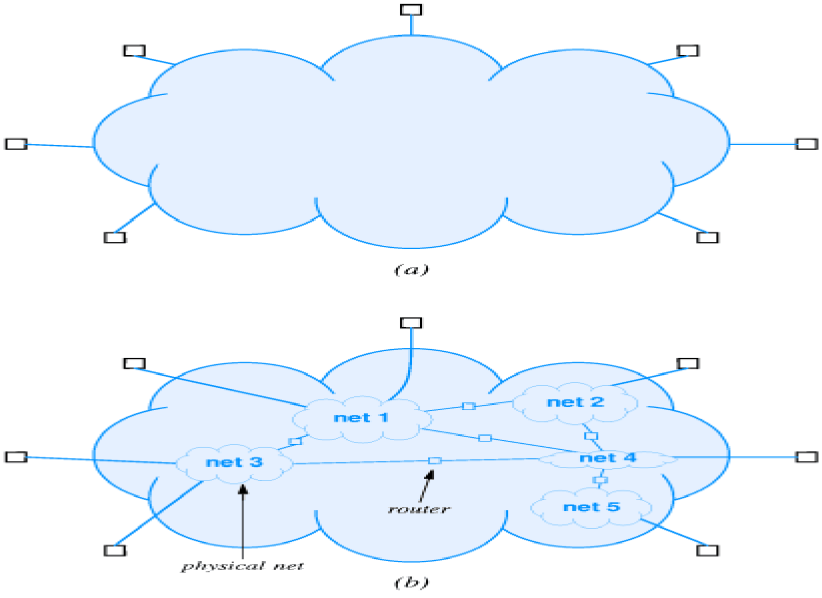
Routers in an organization

- Would be possible to interconnect all networks in an organization with a single router

- Most organizations use multiple routers
  - ✓ Each router has finite capacity; single router would have to handle all traffic across entire organization
  - ✓ Because internetworking technology can automatically route around failed components, using multiple routers increases reliability

A virtual network

- Internetworking software builds a single, seamless virtual network out of multiple physical networks
  - ✓ Universal addressing scheme
  - ✓ Universal service
- All details of physical networks hidden from users and application programs



(a)

(b)

A protocol suite for internetworking

- The TCP/IP Internet Protocols or, simply, TCP/IP is the mostly widely used internetworking protocol suite
- First internetworking protocol suite

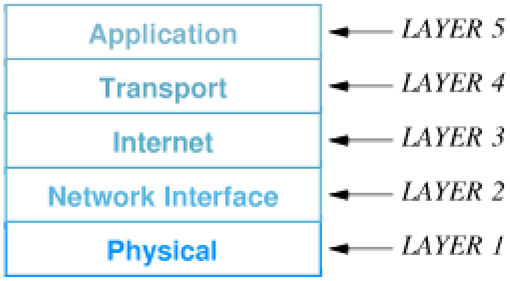
- Internet concept (originally called catenet developed in conjunction with TCP/IP
- Initially funded through ARPA
- Picked up by NSF

Internetworking protocols

- Others include IPX, VINES, AppleTalk
- TCP/IP is by far the most widely used
- Vendor and platform independent
- Used in the Internet - 20 million computers in 82 countries

TCP/IP layering

- OSI 7-layer model does not include internetworking
- TCP/IP layering model includes five layers



Layer 5: Application

Corresponds to ISO model layers 6 and 7; used for communication among applications

Layer 4: Transport

Corresponds to layer 4 in the ISO model; provides reliable delivery of data

Layer 3: Internet

Defines uniform format of packets forwarded across networks of different technologies and rules for forwarding packets in routers

Layer 2: Network

Corresponds to layer 2 in the ISO model; defines formats for carrying packets in hardware frames

Layer 1: Hardware

Corresponds to layer 1 in the ISO model; defines basic networking hardware

Hosts, routers and protocol layers

- A host computer or host is any system attached to an internet that runs applications
- Hosts may be supercomputers or toasters
- TCP/IP allows any pair of hosts on an internet communicate directly
- Both hosts and routers have TCP/IP stacks
  - ✓ Hosts typically have one interface and don't forward packets
  - ✓ Routers don't need layers 4 and 5 for packet forwarding

## Unit-01/Lecture-02

**TCP IP**

TCP IP is a protocol which well you can say it stated as early as in 1970s and it got very quickly accepted by a white community of users. In fact when the internet came into the being, TCP IP was the prime vehicle which was used to connect the computers in the internet and to allow them to communicate over the network. Using TCP IP the computers were able to communicate among each other. And also another very important thing, they were able to share some resources across the network. Some of the resources like dig space or some of the some of the expensive equipments were expensive in those days and over the network it was possible to share those resources. And work on TCP IP stated in the late 60s and in the early 70s it started to take shape. Now in US like most of the innovative developments, the initial research on TCP was funded by the US department of defense the military. So as part of their project it started with a very small network it was called ARPA. It was called advanced research project agency ARPA and the network which evolved in the process it slowly came to know as the ARPANET or the ARPA NETWORK. So this ARPANET was the first network you can say which started to use the first version of TCP.

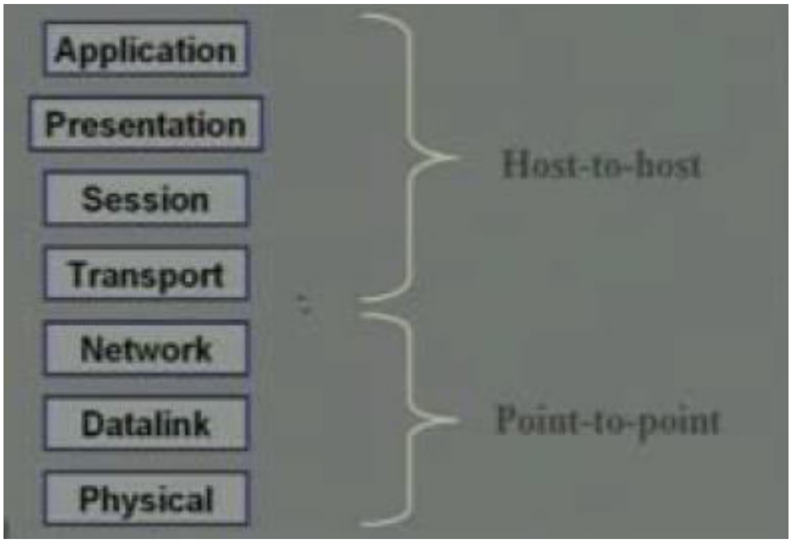
Now this TCP has become so widely accepted technology in modern internet that you can say that in today's world the internet as we see, this is entirely dependent or based on the TCP IP technology which lies under it. So in the internet TCP IP today is treated as a standard. Standard means say when you buy or purchase a new computer and if you want it to get connected to internet the first thing you must ensure that your computer learns or understands the language of TCP IP. Because this is important because all other computers which are connected to the internet. They know and understand TCP IP. So if your computer can talk in the same language then you can communicate with the others. But suppose you have a computer which has a proprietary system, it uses something other than TCP.

Then it will not be possible for your computer to send a message or communicate with some other computer which is connected to the internet somewhere else. So this TCP IP is a standard it has been used to bridge the gap between non compatible platforms. Well you say non compatible platforms, what I mean to say is that some of the machines you are trying to connect may be running the windows operating systems. Some may be running some version of UNIX, Linux; some may be Macintosh machines, macs. So there are wide varieties of types of machines and operating systems which are in the use today. So if all of them have a common layer of software namely the TCP IP then using that common layer they can very easily talk among themselves and work in you can say synchronizing with each other. So the most important point you notice that all computers connected to internet today must understand TCP IP, this is important

**purpose of layering**

Well we develop software in terms of layers. Because number one is that if we make some change in one of the layers as long as the interfaces remain the same, the other layers need not be modified. So if the interface remains the same we can make changes to a layer pretty easily. This is of course one reason; the second reason is that if we have well defined layers your total software can be defined into well-defined modules. So the software development process also becomes much easier.

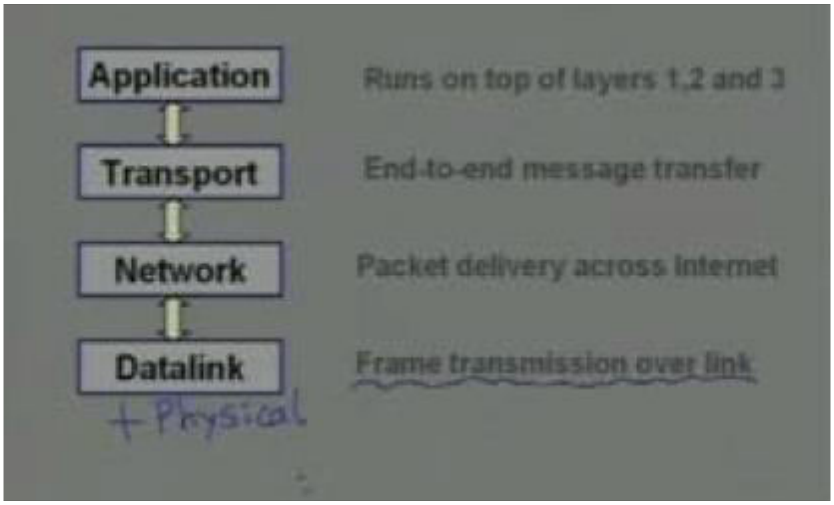
**OSI 7 layer model**



Now just to recall the OSI 7 layer model, had these 7 layers starting from the lowest level physical, data link, network, transport, session, presentation and application. Now you recall one thing. Suppose you have a network like this, there are several nodes connected. Right these are several networks nodes connected and there are some links which are joining them. Suppose this is my source and this is my destination. I want to send some message from the source to this destination from the source to this destination. Now there are some of the layers, in fact the lower 3 layers physical, data link and network. They work on a point-to-point link means they work or they cooperate in sending the data over this link, over this link and so on.

Whereas the higher 4 layers for them the intermediate thing is more like a black box. Only the 2 end systems are important and whatever this source is sending as if the destination is receiving directly. This is the advantage you have in layering the lower 3 layers of the software. These are working on the point -to-point to links. But they ensure that the packets which are starting from S will finally reach D. In contrast the higher 4 layers, they are not worried about how the packets are delivered to D. They are more worried about what is reaching D. Finally these are therefore called host-to-host layers because for them the internal network is a black box. Only the two communicating entities are the 2 ends the source and the destination. They are important as if the source is directly talking to the destination.

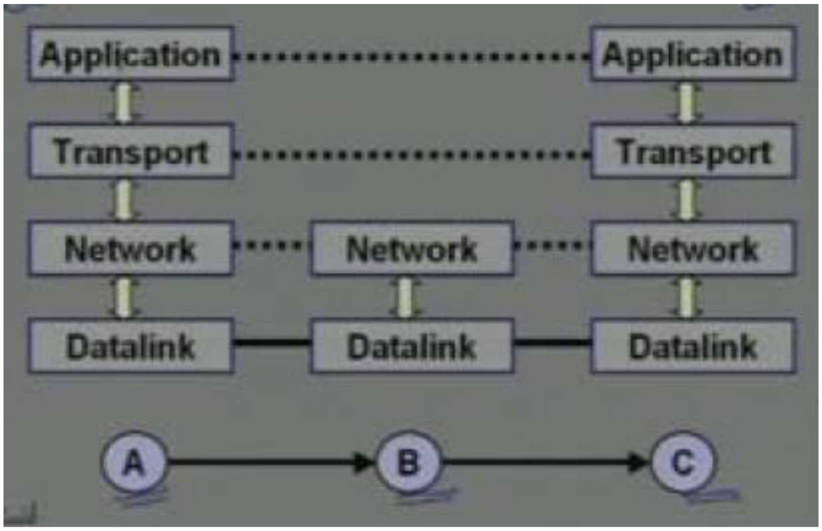
**TCP IP the simplified 4 layered model**


Application
Runs on top of layers 1,2 and 3
Transport
End-to-end message transfer
Network
Packet delivery across Internet
Datalink
Frame transmission over link
+ Physical

TCP IP the simplified 4 layered model which is used, well if you draw an analogy from the OSI model approximately the layering would look like this. At the lowest level this is sometimes called the data link layer. This layer will be responsible for the transmission of frames over link. But one thing you understand all thought the lower level link is called the data link layer. This must also include the physical layer. Now the reason they are put together in a single block or layer is that in most of the modern systems what happens is that we have also called network interface card or a nic which we put inside your computer and through a networking cable we connect that nic. Now that nic that piece of hardware it works both as a physical layer and as the data link layer. So both the two layers get embedded into the hardware interface card. That is why we really cannot distinguish the two layers in modern day systems.

That is why they are often embedded into a single layer. Now the next higher layer the network layer. This is responsible for packet delivery across the internet or source sends a packet. It is the network layer in all the intermediate nodes as well as the source and destination. It will be responsible for routing or forwarding the packet systematically from the source to destination along the correct path transport layer is the end-to-end link where the source and destination can talk directly and application is any program which can run on top of these. Now examples of applications may be electronic mail. Email may be any other program you are running which is communicating with any other client server programming, in fact which is communicating with another program on the other machine. They can be treated as application.

**Dataflow in 4 layer model**



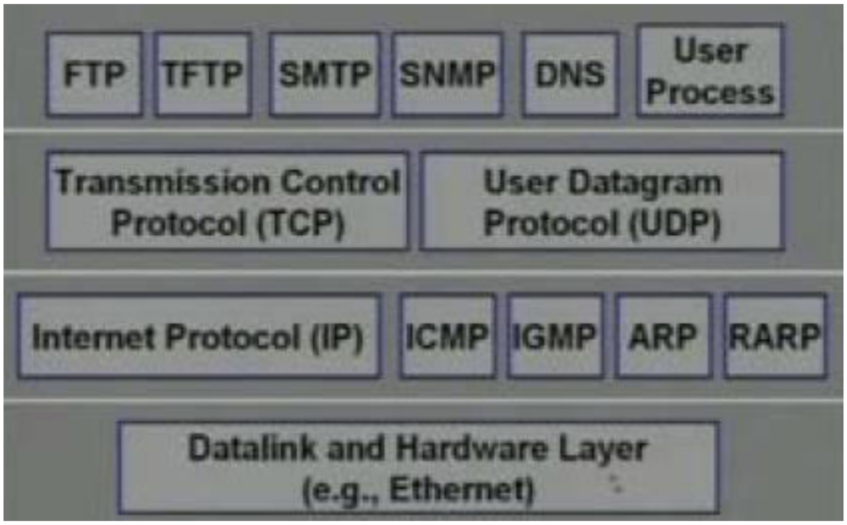
Now in a 4 layer model, let us look into this diagram in a slightly more detail. Now what we are showing here is that there are 3 nodes A, B and C. Suppose A is the source, C is the destination and the packets are the data from A, go to C via an intermediate node B. So when I say that A is the source and C is the destination I mean is that on machine A, there is some application program which is running. Similarly on machine C there is some other application which is running which will be receiving the data send by the application running on A. So what the application program running on machine A will do is that, it will send some information down to the transport layer. Transport layer will be sending it down to the network layer.

Network layer will take some decision that will it will be looking at the destination address. The address where the packet has to be delivered and in case this node A has more than one outgoing links. It will take the decision through which outgoing link the packet has to be forwarded. So the network layer will be sending back the information to the data link layer corresponding to this selected link. For example I have selected this link, so over this physical link the data packets or the frames will go to machine B. Machine B is not the final destination. Now here again there can be several outgoing links. So what this machine B will do, it will again send back the packet to its network layer. And the network layer will again take a decision depending upon the destination address that over which link we have to forward it.

Suppose it selects this link, so now from the network layer it again comes down to the datalink of that corresponding selective link. So finally the packet reaches C. So from C again it traces the reverse route datalink to network network to transport and transport back to the application. So the point to note is that in all the intermediate nodes only the networks layers up to the network layer, datalink and network these two layers are coming into the picture. One is at the two ends where the end to end layers like transport and application are being used. So this diagram shows that how typically data flows in a four layer model network model. So now let us come to TCP IP specifically what the TCP IP suite of protocols really mean?

Now as I had mentioned TCP IP does not refer to a single protocol or a few two or three protocols. In fact this refers to a family of protocols. All these protocols are built on top of a so called connection less technology. Now connectionless technology as you had seen earlier, these means datagrams. There is a basic mechanism of transporting datagrams over the network. So in TCP IP the basic idea is that data will be sent from one node to the next as a sequence of datagrams. Now exactly the way datagrams work each datagram is independent of the others. So the datagrams will be sent independently. Now again it is a characteristic of datagrams. Since they are send independently there is no guarantee that the datagrams that comprise of the same message will be following the same path. Say suppose I have a message I break up the message into 5 datagrams and I send them independently, each of these 5 datagrams may be following different paths. So there may be reaching the destination following different paths. And accordingly the order of arrival may be different. Some of the datagram may get lost in transit and if there is some error checking time out retransmission then some duplicate datagrams may also get generated. So variable delay arrival order at destination are possible.

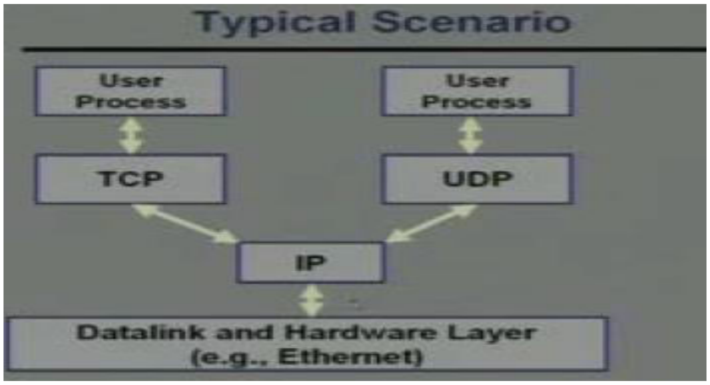
**TCP IP family Members**


FTP
TFTP
SMTP
SNMP
DNS
User Process
Transmission Control Protocol (TCP)
User Datagram Protocol (UDP)
Internet Protocol (IP)
ICMP
IGMP
ARP
RARP
Datalink and Hardware Layer (e.g., Ethernet)

Now let us have a very quick look at the important members of the TCP IP family. This diagram summarizes the different important family members in TCP IP. Of course at the lower level we have the data link and the hardware layers this refers to the network interface card as I told you. Typically we use Ethernet at this level at the next higher level, this is the network layer level in TCP IP the protocols that works at the network layer level is called internet protocol or in short IP. So IP is the main protocol which is working at the network layer level. But in addition there are several auxiliary protocols like internet control message protocol, group message protocol, address resolution protocol and reverse address resolution protocol which also works at the network layer level.

But they have specific functions like ICMP is used to generate and send some error messages, like for example address resolution protocol is used to translate from an IP address into an Ethernet address. For example, so IGMP and RARP also have some specific purposes. But in a typical application you will be using internet protocols. These protocols are typically hidden from the applications. These are invoked and used in a transparent way. As a user you will not be able to see that they are been used or as a programmer also. You will not be using them directly. Now at the transport layer level you have two options. You can either use transmission control protocol or TCP or you can use user datagram protocol or UDP. So you see although we call it TCP IP actually you can also use UDP and IP. TCP IP is just a name for this family moving up to the application layer level.

So at the application layer level there are a number of well-defined applications. Only a few of them I have listed here we have file transfer protocol, we have trivial file transfer protocol, we have simple mail transfer protocol, we have simple network management protocol, domain name server or you can have any general user application which has been developed. This is only a short listing, general you can have any number of applications at this level. So typically an application running on this level. For example this user process. This can interact with either TCP or UDP depending on what you want. TCP or UDP in turn will interact with IP then IP will interact with the lowest layer. This diagram shows that simplified typical data flow model.



This is the typical scenario for at the application layer level the user process they will be using either TCP or UDP. There is a choice, there is a characteristic, they will be in turn using IP and IP will be interacting with the lowest layer. So the most important members of TCP IP family are TCP, UDP and IP; these three. So first let us try to say briefly what the basic functionalities of three models are? First let us look at IP the protocols that works at the network layer level has. I had mentioned that a network layer protocol is responsible for the correct routing of a packet from the source to the destination. So it will take some decision that where to forward or send the packet next and the packet will find its way from the source to the destination taking help of the IP layer. IP layer software that is running on each of the intermediate nodes

**What does IP do?**

Basically IP is a transport system for datagrams. So at the level of IP it is only datagrams which flow on the network. So IP is basically a datagram packet delivery system. It is responsible for routing the packets. Of course moreover if it sees that a packet is too large it cannot be handled by a network, then it can break a packet into smaller pieces which are called fragments which will be talking about later. That how it is done? So a packet may be broken into smaller fragments of packets which later will need to be reassembled again to get back the original packet. Now this reassembling is done at the final destination. But the point to notice that IP does not do any kind of error control. It is an unreliable service, it is basically a datagram service and all the unreliability that comes with datagram it stays with IP also like a packet may get lost.

Packet may get arriving, may arrive out of order because the individual packets may follow different paths to the network. And in case of time out and retransmission duplicate copies of the same packet may be flowing through the network. So the final destination the IP layer at the final destination has to manage all these things packets arriving out of order. Some packets not coming at all and multiple copies of the same packet coming. So these issues have to be handled there. Now looking at TCP which works at the transport layer level and you recall the transport layer level is an end to end protocol where the two hosts which are running on the two machines they talk to each other.

**What does TCP do?**

TCP is a reliable transport layer protocol, well when you say reliable. What do you mean is that it is connection oriented reliable service. When you say connection oriented, well the user or the user application gets an illusion that there is a connection which has been established between the two end systems. So the application program running on one computer believes that a connection as been established with the application running on other end system. And I can send some message directly to the other machine. But what happens in practice is that ultimately the data will be transported by the IP layer below. When IP is an unreliable layer. So TCP has to do some additional error checking and in case of error TCP will have to recover from the error. So TCP will be given illusion to the layers above it that well everything below you is a very reliable network. So essentially the tasks which are performed by TCP is that splitting a message into packets, reassembling the packets at the destination again and it checks if some of the packets are missing.

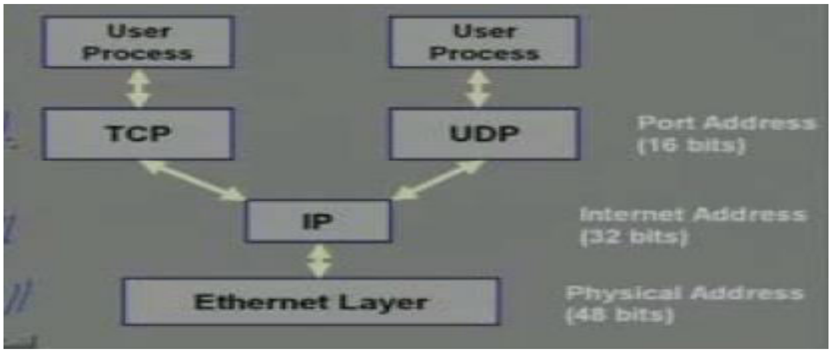
So if there are missing and explicit request is sent back to resend that packets and in this way reliability is ensured. And TCP sits on top of IP so the way it interfaces is that the packets which TCP generates will be forwarded to IP individually for delivery. But IP does not do any error checking or error control. So all error control is the responsibility of TCP. So this is a nutshell is what the TCP layer is supposed to do and the user datagram protocol called UDP. That is also a transport level protocol, but that is different. Well as compared to TCP, TCP tries to provide reliability it tries to establish provide a connection oriented service. But UDP does not try to anything UDP says that well I am trying to transport a packet as fast I can, I am not concerned about reliability. I am not concerned about in which order the packets are going. I will try to impose a minimum amount overhead. I will try to deliver the packet fast this is the philosophy behind UDP.

**What does UDP do?**

So UDP provides a connectionless service like there is no apparent connection that is set up between the source and the destination. This is an unreliable service because it does not do any explicit error checking or error control. The applications which prefer to use UDP are those for which messages are small enough which can be fit into a single packet. A typical example is the query that is sent out by the domain name system. DNS query is an example of such a message. In fact DNS uses UDP for that purpose because the protocol is much simpler. You do not have to establish a connection, do not have to maintain a history of the packet so that if a packet gets lost you cannot send back a request for retransmission.

So since you are not doing all these book keepings, this protocol is much simpler in comparison. This is simpler as well as faster as compared to TCP. So since it is simpler it will never split a data into multiple packets. It is expecting that the packet will be of a size which can fit into a single datagram and it does not care at all about error control interface with IP is very simple each UDP packet is a datagram it is simply forwards to IP for delivery. So interface with IP is also very simple here. Now there is another issue.

**Addresses in TCP/IP**



Well if you have another look at this TCP IP stack. Just showing the important protocols TCP, UDP and IP broadly there are three layers. If you recall the physical and the data link layer, the network layer and the transport layer. Talking about the addresses in each of these three layers, there are three different addresses which are used at the lowest layer. Just assuming the lowest layer is an Ethernet. Here you use something called the physical address of the interface link. In this case, this will be a so called Ethernet address and this is a 48 bit address. This 48 bit Ethernet address is embedded inside the network interface card which you plug into your computer. So it is that card which a hardware address has built into it that is the Ethernet address. So when a frame finally comes to your computer, it must come with the Ethernet address as part of the packet or frame.

But when you are looking at the network layer it is a 32 bit IP address which is used to identify the computers in the internet. Now this 32 bit IP address is allocated and assigned in a much more systematic way. This is a logical address, no computer comes with that address built in you can program a computer you can assign any IP address to it. Now the purpose of this assigning of IP address is that you are trying to make it easier for the intermediate nodes in the network to forward and route packets just by looking at the IP addresses. But once the packet enters or comes into the LAN where the final destination node is situated, so there you have to finally find out the Ethernet address of the destination. That is done through that ARP protocol I talk I talked about a little ago. So using ARP you get the Ethernet address and finally the packet will reach the final destination addressed by the 48 bit Ethernet address.

Now at the level of the transport layer, now you recall the transport layer is the end to end layer. So here two applications on two end hosts are communicating. Well here also there is an address called port address. It is a 16 bit wide address, well let us see why this is required or needed. Now at the transport layer level when there are two end hosts and two applications running there are trying to communicate, it may be very much possible that would each of these machines. There is not only one application but several applications running. That is the typical scenario in most computer systems today. There is time sharing, there is multi-tasking. So there are many programs which are running at the same time. So you must mention that when you are sending a message to exactly to which application program at the other machine you are trying to send the message.

So this port number in essence identifies an application which is running on the particular machine. This is why we have hierarchy of addresses. Physical address to ultimately identify the machine through its hardware interface card. This IP address in order to identify the network where the machine belongs and finally to send route the packet to that network and the port address to identify an application running on the end machine end host. Now any layering of software just see this networking software all of them have some sort of layering. We talked about the seven layer OSI model. Now we are talking about the 4 layer TCP IP model. Now when there is layering the packets flow in the systematic way from top

to bottom and again from bottom to top. Now as this information flows up and down some additional information gets appended or removed during these movements.

## Unit-01/Lecture-03

### Internet address and domains

The problem of basic IP addressing. Well one thing we had seen earlier that the source and destination address in the IP protocol is a 32 bit quantity that is the IP address.

Now in the internet scenario, now if you want one host to communicate with other they must all have unique IP address. Because when you are sending a packet with a destination IP address, you must be sure that there is only one computer in this world who has this particular address.

If there are more then there will be confusion and the packet may be delivered in the wrong place. So we have the concept of IP address which is supposed unique with respect to each host. This is a 32 bit quantity and representing a 32 bit quantity has a streams of 0 and 1s is inconvenient. So it is typically expressed as a so called dotted decimal notation.

Dotted decimal notation means there are 4 decimal numbers W. X. Y. Z separated by dots and W. X. Y. Z represent the decimal equivalence of each of the 4 octets or the 8 bit chunks in the address.

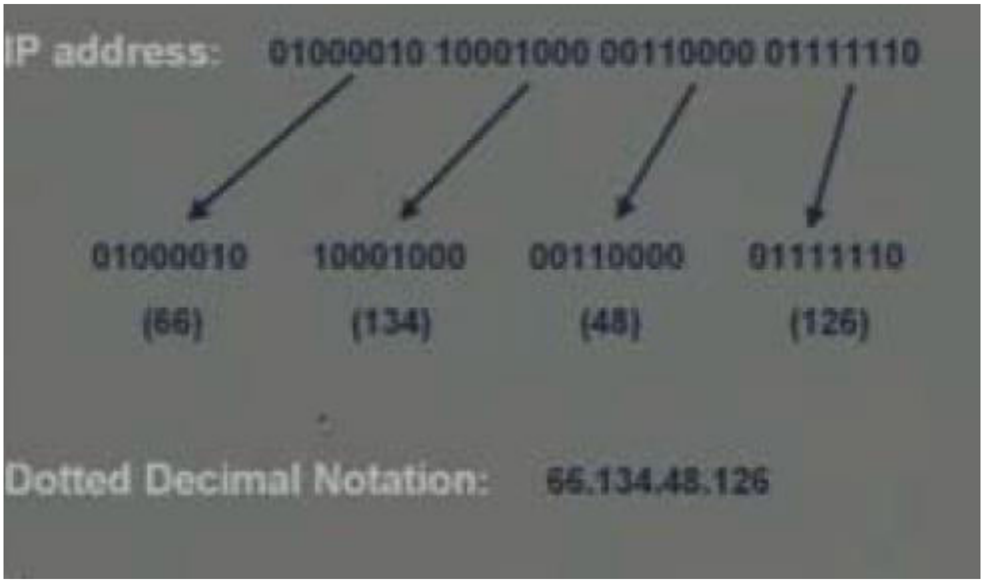
Well we have an example I will show you now this IP address logically contains two parts. Well one part identifies the network. Suppose your computer is located inside your organization network. So the first part of your compute address will identify the address of your network. So using that information the packet must reach your network.

So there is one part in the address which identifies the network where the destination host belongs. But the other part indicates the host number which means ones the packet reaches the destination host.

Then after that you use this host number to identify the computer where this packet has to be delivered or has to go. Now this division of IP address into two logical parts network number and host number.

This can be done in a very simple systematic way, we shall see very shortly. There are something called IP address classes which are defined. We can use this IP address classes for this kind of logical division

**Dotted Decimal Notation**



IP address: 01000010 10001000 00110000 01111110

01000010 (66) 10001000 (134) 00110000 (48) 01111110 (126)

Dotted Decimal Notation: 66.134.48.126

Well first let us look at the dotted decimal notation with respect with the example. Suppose I have an IP address, this is a 32 bit quantity like this. This 32 bit address I am dividing out into four 8 bit chunks. These are the 4 octets and each of the 4 chunks I am converting into decimal. This is 66, this is 134, this is 48 and this is 126. So after you convert each of these into decimal I simply write down the decimal numbers separated by dots and this is the so called dotted decimal notation. So we typically express the IP address of the machine in this form 4 numbers separated by dots this so called dotted decimal notation.

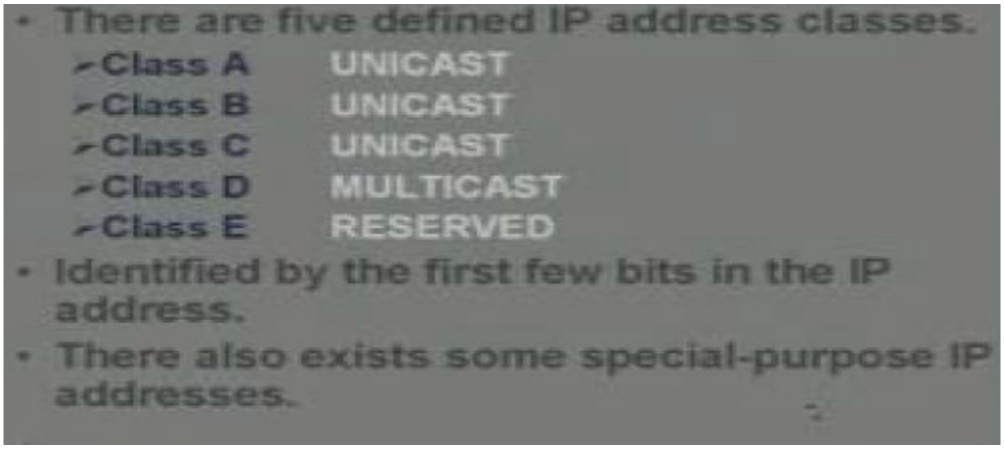
Hierarchical Addressing

So talking we just mentioned that when we want to address the computer which is connected to the internet we actually specify two things the network number and the host number this actually represents a hierarchy in terms of the addressing. Now whenever you want to mention the network number well as I had mentioned the network number is something unique to your network. So whenever a node or a router sees that the network number of the destination address is something say x. So it knows where the network x is located and it tries to throw or forward the packet direction. So somehow the network addresses must be ensured to be unique across the world. So for this purpose there has to be some kind central authority that grants and manages these network numbers.

So if your organization has a network address no other computer network in the world must have or can have the same address, this must be ensured. So the network numbers the assignment and management is done by some central authority. So just whenever when you want to set up your own network, you can apply to the central authority. They will be granting you a new network number new unused network number you can use it for your network. The next part of the address the host number of course this is a local issue. Once the packet enters your network the way you number or address a host it is up to you. So these are assigned and managed by the local network administrator. So there is one part of the address which is managed globally.

There is another part of the address which is managed locally but when you talk about routing a packet actually we talk about leading a packet to the correct destination network. So in order that the packet reaches the correct destination network we need not look at the whole of the address. We need to look at only the network portion of the address. The host portion of the address will be required only after the packet has reached or arrived at the destination network. So this is one thing which should be kept in mind this is very important that it is only the network portion of the address which is responsible or is used for the purpose of routing the packets. So this is what is mentioned out here.

**IP Address Classes**

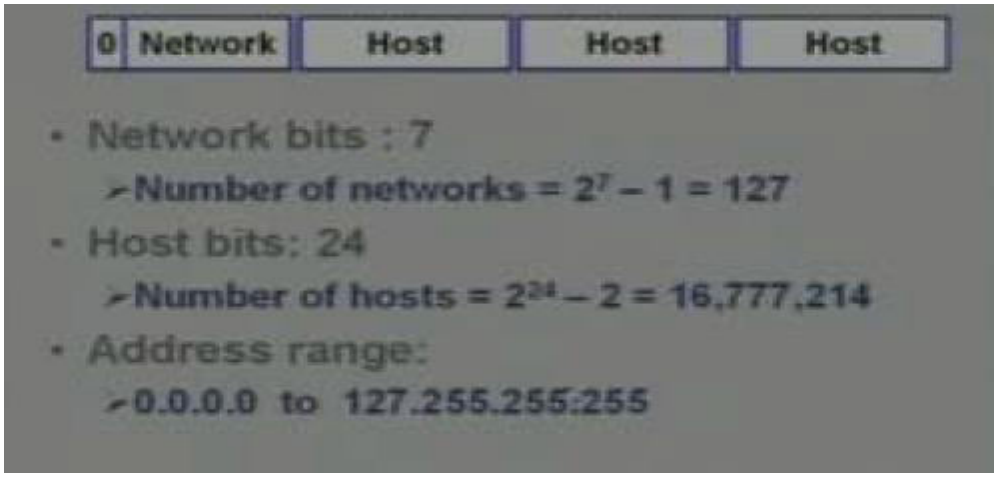


- There are five defined IP address classes.
  - Class A UNICAST
  - Class B UNICAST
  - Class C UNICAST
  - Class D MULTICAST
  - Class E RESERVED
- Identified by the first few bits in the IP address.
- There also exists some special-purpose IP addresses.

Now, in terms of the IP addresses. IP address are divided into several classes as you can here A B C D E. So out of them you can leave out E. E is really not used, this is a reserved category. Well if you want to do some experimentation on IP addresses you can use this class E the first three classes A, B and C. These are so called unicast address unicast means that using these addresses you are identifying one particular network on the internet. The addresses uniquely specify one computer. In contrast class D is a multicast address. MULTICAST means you want to send a packet to a group of computers. Say all computers which are belonging to a LAN you want to send them all at the same time. So this is something called multicasting. The address will be such that it will be broadcast or multicast to all the computers within a particular group. Now which class the IP address belongs to this is identified by the first few bits in the IP address. Now in addition to these classes we shall see that there also exists some special purpose IP addresses. Now let us see this in some detail.

Now this class based addressing this class A B C D E as I had mentioned. Here in these addressing schemes there is a fixed and well defined partition of an address with respect to the network part and host part. Network address part and the host address part, this partition is fixed and this mode of addressing a computer based on address classes is also called the so called classful model. Depending upon the network classes as we will see the network to host ratios of the network can vary. Depending on which class we use you can have different network configurations. These we will see very shortly how this is done.

**Class A address**

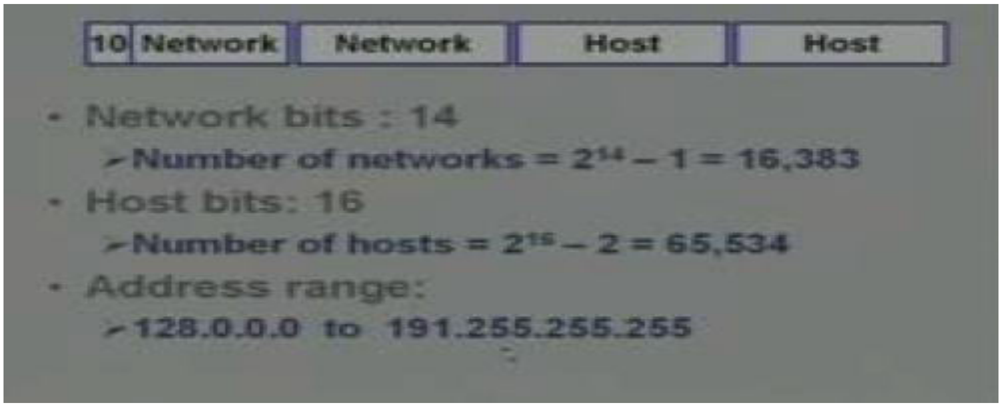


- Network bits : 7
  - Number of networks = $2^7 - 1 = 127$
- Host bits: 24
  - Number of hosts = $2^{24} - 2 = 16,777,214$
- Address range:
  - 0.0.0.0 to 127.255.255:255

First let us see the basic characteristic of the different address classes first the class A. Class A represents those networks which are very large in size particularly the very big internet service providers. They would like to have a class A kind of address. Let us see what class A address says. In a class A address first thing is that these are the 4 octets; 1, 2, 3 and 4. A class A address begins with a number 0, the 7 bits out here represents the network. So other number of network bits is 7 and host is 24. 24 bits represents the host. In 7 bits although the total number of combination is 2 to the power 7 or 128 we shall see that out of the one particular combination is left aside for some other purpose.

So actually the total number of such network addresses you can have in class A will be a one less it is 127. Similarly for the hosts each of these class and networks can have of the order of 2 to the 24 hosts. Out of them we will see again that the all 0 and the all one combinations are used for some other purpose. So actually you have two addresses less it comes to a figure like this 16 above 16 million. So number of hosts in a class and network can be large as 16 million based on this address assignment in the dotted decimal notation a class A address can range from 0, 0, 0, 0 up to 127, 255, 255, 255. That means all zeros up to all ones. This is the range, so if you see an IP address within this range you can immediately identify that is a class A address.

**Class B address**

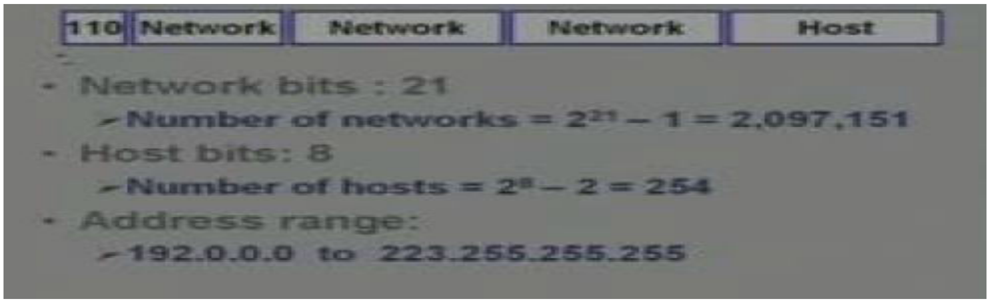


| | | | | |
|---|---|---|---|---|
| 10 | Network | Network | Host | Host |

- Network bits : 14
  - ➢Number of networks = $2^{14} - 1$ = 16,383
- Host bits: 16
  - ➢Number of hosts = $2^{16} - 2$ = 65,534
- Address range:
  - ➢128.0.0.0 to 191.255.255.255

Now class B belongs represents the network which are medium in size; smaller than class A, but larger than class C we will see later. In class B the division between network and host is like this this. Address starts with 1 0, so just by looking at the first few bits you can identify class A and class B. The next 14 bits represent the network and the last 16 bits represent the host. So total number of networks will be 2 to the power 4 again 1 less than that 16383 and total number of hosts will be last 16 bits 2 to the power 16 take out the all zero and all one combination it is 65534. And if you just put all zeros and all ones and convert it to dotted decimal notation you will see that the range of address is that comes here is 128 000 up to 191, 255, 255, 255. This is the range of addresses in class B networks. Now class C network represents the ones which are smallest in size for most purposes class C networks are sufficient for our purpose. There the number of computes is host in a network can be up to 254. Now let us see the break up.

**Class C address**

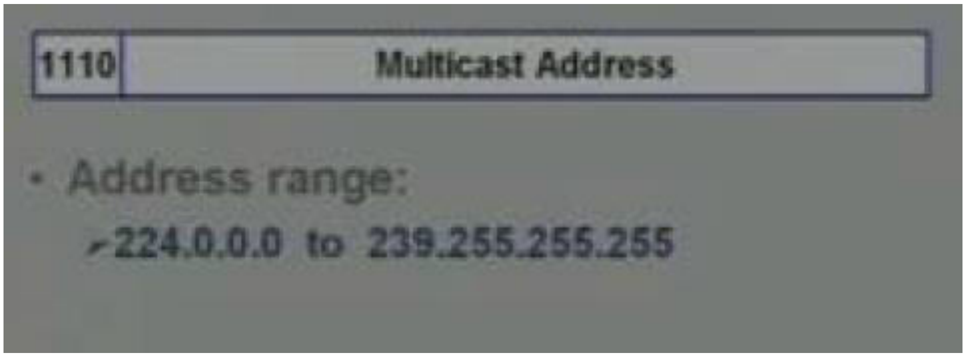


| | | | | |
|---|---|---|---|---|
| 110 | Network | Network | Network | Host |

- Network bits : 21
  - ➢Number of networks = $2^{21} - 1$ = 2,097,151
- Host bits: 8
  - ➢Number of hosts = $2^{8} - 2$ = 254
- Address range:
  - ➢192.0.0.0 to 223.255.255.255

Here the address starts with 110 in order to distinguish it from class A or class B. Next 20 bits represent the network; last 8 bit represents the host. So you can have very large number of class C networks of the order of 2 million. But the number of hosts in each network is only 254 addresses in the range 192.0.0.0 up to 223.255.255.255, they belong to the class C address category.

**Class D address**



| | |
|---|---|
| 1110 | Multicast Address |

- Address range:
  - ➢224.0.0.0 to 239.255.255.255

Now lastly is class D address. Well I am not going the detail of this now. The class address is will start with 1110 and after that whatever you specify here this will identify the address of a group. This is sometimes called a multicast address so depending upon what is the group address I am giving. The packet will be broadcast to all members of that group. This is the idea, so address range is this. Now you see just by looking at IP address you can identify which address category the address belongs to. Given an IP address, you look at only the first few bits of the address the rule is very simple. If the first bit is 0, if it is zero then you can it is class A. If it is 10 then you can say it is class B and if it is 110 then you can say it is class C. There is no ambiguity; just the first few bits uniquely identify the address classes. Depending on the bits the routers or the intermediate nodes can easily identify the address classes and accordingly perform or take the routing decisions. Now in terms of the number of addresses in the whole 2 to the power 32 bit address space of an IP address. Let see how much this classes A, B or C consume. See in class A the first bit starts with 0. So we are taking away half of the total addresses.

**Introduction World Wide Web (WWW)**

**WWW**

The terms Internet and World Wide Web are often used in every-day speech without much distinction. However, the Internet and the World Wide Web are not one and the same. The Internet is a global system of interconnected computer networks. In contrast, the Web is one of the services that runs on the Internet. It is a collection of interconnected documents and other resources, linked by hyperlinks and URLs. In short, the Web is an application running on the Internet. Viewing a web page on the World Wide Web normally begins either by typing the URL of the page into a web browser, or by following a hyperlink to that page or resource. The web browser then initiates a series of communication messages, behind the scenes, in order to fetch and display it.First, the server-name portion of the URL is resolved into an IP address using the global, distributed Internet database known as the Domain Name System (DNS). This IP address is necessary to contact the Web server. The browser then requests the resource by sending an HTTP request to the Web server at that particular address. In the case of a typical web page, the HTML text of the page is requested first and parsed immediately by the web browser, which then makes additional requests for images and any other files that complete the page image.

While receiving these files from the web server, browsers may progressively render the page onto the screen as specified by its HTML, Cascading Style Sheets (CSS), or other page composition languages. Any images and other resources are incorporated to produce the on-screen web page that the user sees. Most web pages contain hyperlinks to other related pages and perhaps to downloadable files, source documents, definitions and other web resources. Such a collection of useful, related resources, interconnected via hypertext links is dubbed a web of information. Publication on the Internet created what Tim Berners-Lee first called the **WorldWideWeb.**

Linking

Graphic representation of a minute fraction of the WWW, demonstrating hyperlinks Over time, many web resources pointed to by hyperlinks disappear, relocate, or are replaced with different content. This makes hyperlinks obsolete, a phenomenon referred to in some circles as link rot and the hyperlinks affected by it are often called dead links. The ephemeral nature of the Web has prompted many efforts to archive web sites. The Internet Archive, active since 1996, is one of the best-known efforts.

Dynamic updates of web pages

JavaScript is a scripting language that was initially developed in 1995 by Brendan Eich, then of Netscape, for use within web pages. [22] The standardized version is ECMAScript. [22] To overcome some of the limitations of the page-by-page model described above, some web applications also use Ajax (asynchronous JavaScript and XML). JavaScript is delivered with the page that can make additional HTTP requests to the server, either in response to user actions such as mouse-clicks, or based on lapsed time. The server's responses are used to modify the current page rather than creating a new page with each response. Thus the server only needs to provide limited, incremental information. Since multiple Ajax requests can be handled at the same time, users can interact with a page even while data is being retrieved. Some web applications regularly poll the server to ask if new information is available. [23]

Caching

If a user revisits a Web page after only a short interval, the page data may not need to be re-obtained from the source Web server. Almost all web browsers cache recently obtained data, usually on the local hard drive. HTTP requests sent by a browser will usually only ask for data that has changed since the last download. If the locally cached data are still current, it will be reused. Caching helps reduce the amount of Web traffic on the Internet. The decision about expiration is made independently for each downloaded file, whether image, stylesheet, JavaScript, HTML, or whatever other content the site may provide. Thus even on sites with highly dynamic content, many of the basic resources only need to be refreshed occasionally. Web site designers find it worthwhile to collate resources such as CSS data and JavaScript into a few site-wide files so that they can be cached efficiently. This helps reduce page download times and lowers demands on the Web server.

## Unit-01/Lecture-04

**Working of web browser and web server: [RGPV/Jun 2014(7)]**

LIKE much of the Internet, the World Wide Web operates on a client/server model. You run a web client on your computer—called a web browser—such as Microsoft's Internet Explorer or Firefox. That client contacts a web server and requests information or resources. The web server locates and then sends The information to the web browser, which displays the results.

When web browsers contact servers, they're asking to be sent pages built with Hypertext Markup Language (HTML). Browsers interpret those pages and display them on your computer. They also can display applications, programs, animations, and similar material created with programming languages such as Java and ActiveX, scripting languages such as JavaScript, and techniques such as AJAX.

Sometimes, home pages contain links to files the web browser can't play or display, such as sound or animation files. In that case, you need a plug-in or a helper application. You configure your web browser or operating system to use the helper application or plug-in whenever it encounters a sound, animation, or other type of file the browser can't run or play. Over the years, web browsers have become increasingly sophisticated. Browsers are now full-blown software suites that can do everything from videoconferencing to letting you create and publish HTML pages.

Browsers now also blur the line between your local computer and the Internet—in essence, they can make your computer and the Internet function as a single computer system.

Increasingly, a browser is not just a single piece of software, but an entire suite. The newest version of Internet Explorer, for example, includes security features such as an anti-phishing filter. The Firefox browser has a companion piece of email software called Thunderbird that can be downloaded as well.

When browsing the Internet, one of the most frustrating experiences is the error messages browsers display when they're having trouble contacting a website. Depending on which browser you use, and which version of the browser you're using, those messages might differ. Sometimes browsers display error messages in plain English—but more often they don’t. The final illustration in this chapter lists the most common browser error messages—and what they mean.

**How a Web Browser Works?**

1) Web browsers consist of client software that runs on your computer and displays home pages on the Web. There are clients for a wide variety of devices, including Windows, Macintosh, and Unix computers.
2) A web browser displays information on your computer by interpreting the Hypertext Markup Language (HTML) that is used to build home pages on the Web. Home pages usually display graphics, sound, and multimedia files, as well as links to other pages, files that can be downloaded, and other Internet resources.
3) The coding in the HTML files tells your browser how to display the text, graphics, links, and multimedia files on the home page. The HTML file your browser loads to display the home page doesn’t actually have the graphics, sound, multimedia files, and other resources on it. Instead, it contains HTML references to those graphics and files. Your browser uses those references to find the files on the server and then display them on the home page.
4) The web browser also interprets HTML tags as links to other websites, or to other web resources, such as graphics, multimedia files, newsgroups, or files to download. Depending onthe link, it performs different actions. For example, if the HTML code specifies the link as another home page, the browser retrieves the URL specified in the HTML file when the user clicks the underlined link on the page. If the HTML code specifies a file to be downloaded, the browser downloads the file to your computer.

**How do web servers work?**

What happens when you enter in the address field of your browser the URL http://www.aprelium.com/doc/sample.html?

First, the browser slices the URL in 3 parts:

- http://: This part indicates that the document you want to access can be retrieved from web server, which understands the HTTP protocol. The HTTP protocol is a standardized language of communication between browsers and web servers.

- www.aprelium.com: This is the host name of the computer from which the document can be downloaded.
- /doc/sample.html: This is the virtual path of the document in the www.aprelium.com's web server.

Then, the browser contacts a DNS (Domain Name Server) to know the IP address of the computer which full qualified domain name is www.aprelium.com. The domain name server is usually run by your ISP or by your company.

The browser establishes a connection channel with the web server on the computer which IP address was given by the DNS server and requests the document on the host which name is www.aprelium.com and which virtual path is doc/sample.html. The browser has to specify in the request the host name because many modern web servers (including Abyss Web Server) have the ability to serve more than a one host from a single computer with a single IP address only. This is called virtual hosting. In such a case, the IP address of this computer is associated with more than one domain name.

The server decodes the request and maps the virtual path to a real one, which should match an existing file. The server sends the file to the browser with some useful information such as its last modification time and its MIME type. The MIME type helps the browser decide how to display the received document. In our example, it is a HTML file. So the server sets its MIME type to text/html and the browser understands that it must render it as text.

Sometimes you enter a URL without an explicit filename such as http://www.aprelium.com/doc. The browser sends the request to the web server as in the previous example. The server detects that the virtual path maps to a directory and not to a file. It searches then in this directory an index file. Index files are usually named index.html or index.htm. If it finds for example index.html, it acts as if the requested URL was http://www.aprelium.com/doc/index.html. If no index file is found, the web server generates a listing of the directory contents and sends it to the browser or reports an error.

**What is the role of web server on the Internet?: [RGPV/Jun 2014(7),Jun 2012(5)]**

Web servers - the computer or the program - have a vital role on the Internet. The Server machine hosts (stores) the web site on its hard disk while the server program helps deliver the web pages and their associated files like images, flash movies etc. to clients (browsers).

The process of loading a web site/page in a web browser starts with the user either entering the URL in the address bar or clicking on a link. You should know that each web page has a unique address (or URL) on the internet; which means the same page cannot exist in two places. (If a copy does exist in another location, its address would be different from that of the original).

Once the appropriate action has been initiated by the user, the browser sends out a request for the web page. Behind the scenes, the URL of the requested web page is resolved into an I.P. address, which, in English, means, converted to an I.P. address - something that computers understand. The I.P. address points to the location of the web site host. The request is forwarded to Server computer and passed on to the server software.

The server software now gets to work and hunts for the requested web page on the hard disk. On finding the file, it sends a response to the browser followed the actual web page file which then starts displaying the page.

A typical web page not only has text but also embedded multimedia elements like images and Flash animation. These "extra" files are separate from the actual web page and are fetched by the browser from the Server one by one. Note (and an important one), ONLY the web browser determines how a web page is displayed; the web server has no control over this. The job of the web server ends once the requests from the browser are processed and the required information is sent.

Though it might seem that the request-and-response process takes a lot of time, especially when you consider that the client and server computers might be thousands of miles apart, it actually happens very fast. That's because of the HyperText Transfer Protocol (HTTP) which is a set of rules developed by the "big lads" to facilitate the transfer of data over the internet.

**Apache HTTP Server**

First released in 1995, the Apache HTTP Server is a free open-source Web server developed under the governance of the Apache Software Foundation. The Apache 2.0 license permits bundling with commercial software and does not require derivative works to be open source.

A variety of developers make code contributions to the project, including members of the Apache Software Foundation, developers who are allowed or instructed to work on Apache by their corporate employers, and even individuals contributing to Apache on their own time. Companies that use Apache range from start-ups to long-established large enterprises. Apache is used for intranets and public facing Web sites.

Apache is a key component in what’s known as the “LAMP” stack, which comprises the Linux operating system; the Apache Web server; the MySQL database; and either PHP, Perl, or Python programming language. While people often perceive Apache as a Linux Web server, it also runs on Windows.

**Internet Information Server 6.0**

With Windows Server 2003, Microsoft introduced Internet Information Server (IIS) 6.0, which has proven to be a very secure Web server, with only four vulnerabilities reported since its release in 2003. IIS security results from Microsoft investing in the Security Development Lifecycle, an end-to-end approach to security that typically reduces both the total number and the severity of vulnerabilities in software built using that methodology. This isn’t to say that Apache is not secure, as high-profile and widely available Web sites wouldn’t use it if they thought it were, but simply to point out that IIS 6.0 was designed with security in mind, and has a great security track record.

IIS 6.0 included a number of features that made it a good fit for corporations, and enabled hosting providers to offer Windows Server 2003 and IIS 6.0-based solutions. It introduced application pools to prevent one misbehaving site from taking other sites down and it also included health monitoring that allowed administrators to configure sites for automatic restart on failure. IIS 6.0 enhanced management by moving to a single, XML-based configuration file (the “metabase”) and by supporting more operations through a command-line interface. IIS 6.0 used resources more efficiently, thus increasing the performance of individual sites and allowing each server to host a greater number of sites.

**Does IIS offer the performance and scalability I need?**

IIS has proven its ability to handle the scalability and performance requirements of high-traffic sites. Both Apache and IIS 7.0 allow administrators to optimize performance and scalability with bandwidth throttling, compression, and some load balancing. Static and dynamic compressions are built in to IIS 7.0 in order to use bandwidth efficiently. IIS 7.0 also supports bandwidth throttling, while Windows Server 2008 includes full featured network load balancing.

Apache administrators are accustomed to installing Apache on a trimmed-down server installation. Microsoft provides a similar platform for IIS with the “Server Core” installation option. This option means that the operating system is using the fewest resources possible, which makes more resources available to handle the Web workload and ensures that fewer components are installed, requiring less management and maintenance. The modular nature of IIS also helps improve performance, allowing administrators to enable only the modules they need, resulting in a faster processing pipeline.

Caching often provides the biggest performance improvement for Web sites, and IIS provides built-in output caching and object caching that can automatically detect when the underlying database has changed. Apache administrators will find that these IIS 7.0 features are similar in functionality to the caching modules that they typically use with Apache.

The performance and scalability of IIS are proven by some of the most highly trafficked Web sites. For example, Match.com runs IIS to process its 30 million daily page views. In 2004, PlentyOfFish.com used one IIS 6.0 server running at 65 percent of capacity to handle 31 million daily page views from 40,000–50,000 concurrent users ; the site currently handles 1.2 billion page views per month. MySpace.com runs IIS to handle the whopping 23 billion page views it gets every month.

**Is IIS as secure as Apache?**

Microsoft developed Windows Server 2003 and Windows Server 2008 under its Security Development Lifecycle (SDL), which uses education, quality gates, threat modeling, attack surface reduction, static analysis, fuzz and penetration testing, and a final security review to ensure that products are as secure as possible. In addition, the Microsoft Security Response Center engages with external security researchers and is even involved in the security community through its participation in, for example, the Black Hat conference. These efforts have resulted in a substantial reduction in vulnerabilities across the Microsoft product suite, with particularly steep reductions in OS, Web server, and database vulnerabilities. The modular nature of IIS 7.0 further reduces the risk of exploitable flaws, as most modules are not installed by default to keep the attack surface small.

In addition to having fewer vulnerabilities, IIS includes a number of new security features. For example, IIS 7.0 isolates each Web site into its own “sandbox” to help prevent single-site exploits and failures from compromising other sites or the entire server. The IIS process, which executes requests from the web, run as a restricted user account by default, and does not require administrative privileges. To further protect the Web server, IIS 7.0 includes request filtering. Request filtering is a rules-based security module that inspects every incoming request for malicious request patterns, such as SQL injection attacks. This prevents some malicious requests from ever reaching the core Web server.

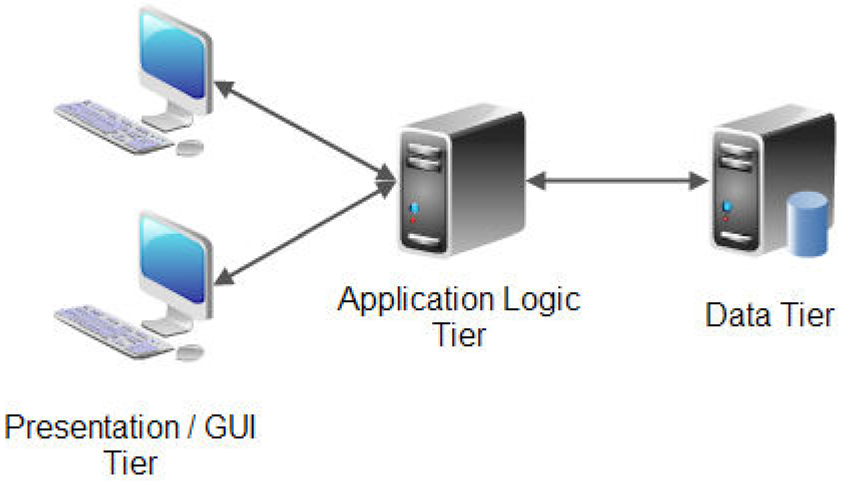
| S.NO | RGPV QUESTIONS | Year | Marks |
|---|---|---|---|
| Q.1 | Discuss the services of web servers and describe the working of web browser. | Dec.2013 | 7 |
| Q.2 | what are web servers? Explain the features and characteristics used for comparison of web servers. | June.2014 | 7 |
| Q.3 | How does the web work? Give two examples of web servers | June.2012 | 5 |

## Unit-01/Lecture-05

**N-tier architecture: [RGPV/Jun 2010(12]**

N tier architecture means splitting up the system into N tiers, where N is a number from 1 and up. A 1 tier architecture is the same as a single process architecture. A 2 tier architecture is the same as a client / server architecture etc.

A 3 tier architecture is a very common architecture. A 3 tier architecture is typically split into a presentation or GUI tier, an application logic tier, and a data tier. This diagram illustrates a 3 tier architecture:



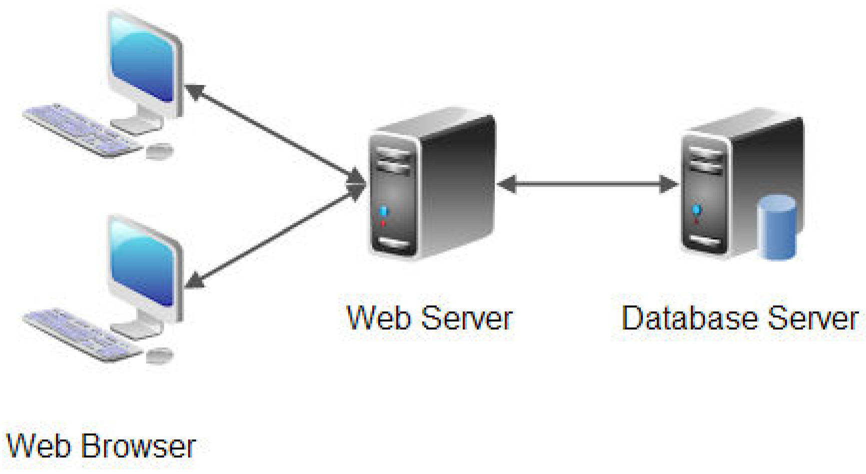
The presentation or GUI tier contains the user interface of the application. The presentation tier is "dumb", meaning it does not make any application decisions. It just forwards the user's actions to the application logic tier. If the user needs to enter information, this is done in the presentation tier too.

The application logic tier makes all the application decisions. This is where the "business logic" is located. The application logic knows what is possible, and what is allowed etc. The application logic reads and stores data in the data tier.

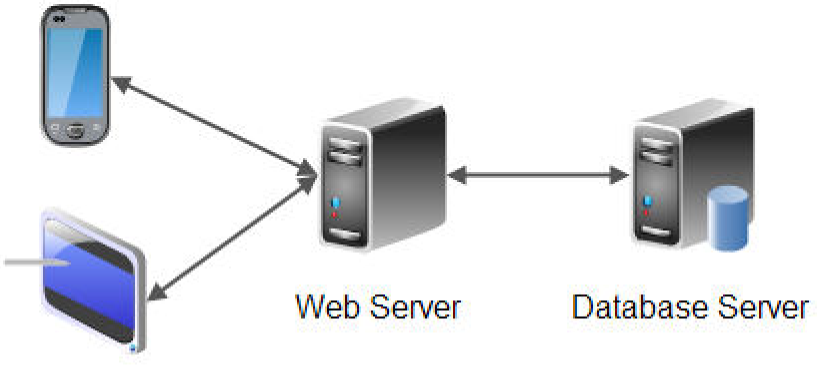
The data tier stores the data used in the application. The data tier can typically store data safely, perform transactions, search through large amounts of data quickly etc.

**Web And Mobile Applications**

Web applications are a very common example of 3 tier applications. The presentation tier consists of HTML, CSS and JavaScript, the application logic tier runs on a web server in form of Java Servlets, JSP, ASP.NET, PHP, Ruby, Python etc., and the data tier consists of a database of some kind (mysql, postgresql, a noSQL database etc.). Here is a diagram of a typical 3 tier web application:



Actually, it is the same principle with mobile applications that are not standalone applications. A mobile application that connects to a server typically connect to a web server and send and receive data. Here is a diagram of a typical 3 tier mobile application:



**Rich Internet Applications (RIA)**

In the first generations of web applications a lot of the HTML, and parts of the CSS and JavaScript was generated by scripts running on the web server. When a browser requests a certain page on the web server, a script was executed on the web server which generated the HTML, CSS and JavaScript for that page.

Today the world is moving to rich internet applications (RIA). RIA also uses a 3 tier architecture, but all the HTML, CSS and JavaScript is generated in the browser. The browser has to download the initial HTML, CSs and JavaScript files once, but after that the RIA client only exchanges data with the web server. No HTML, CSS or JavaScript is sent forth and back (unless that is part of the data, like with an article that contains HTML codes).

RIA applications are explained in more detail in the next text in the software architecture trail.

**Web Application Advantages**

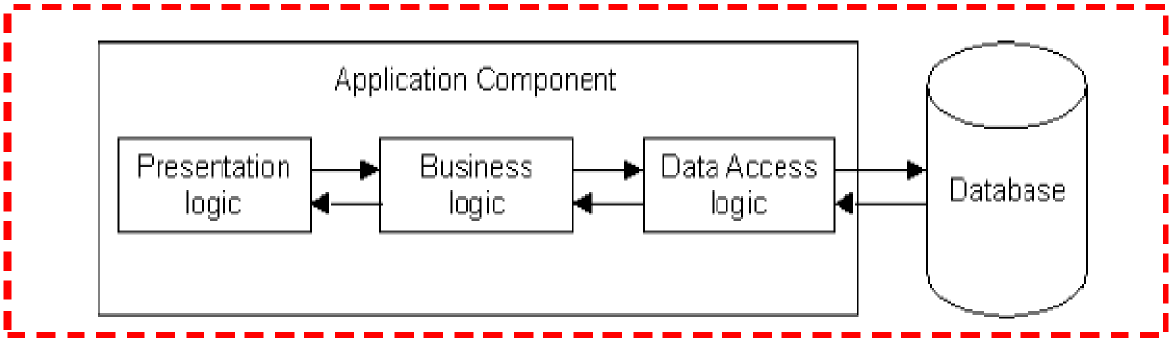
The purpose of N tier architecture is to insulate the different layers of the application from each other. The GUI client doesn't know how the server is working internally, and the server doesn't know how the database server works internally etc. They just communicate via standard interfaces.

Web applications especially have another advantage. If you make updates to the GUI client or the application logic running on the server, all clients get the latest updates the next time they access the application. The browser downloads the updated client, and the updated client accesses the updated server.

**Significance of “Tiers”**

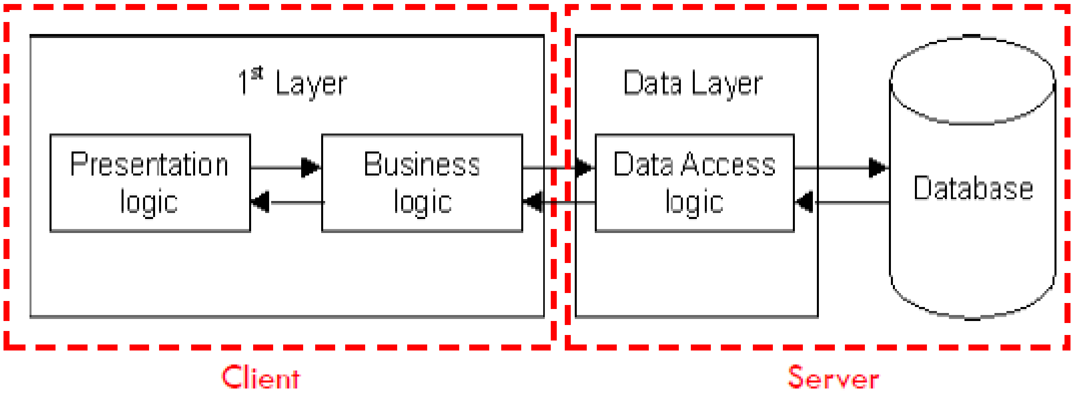
- N-tier architectures have the same components
- Presentation
- Business/Logic
- Data
- N-tier architectures try to separate the components into different tiers/layers
- Tier: physical separation
- Layer: logical separation

1-Tier Architecture



- **All 3 layers are on the same machine**
- All code and processing kept on a single machine
- Presentation, Logic, Data layers are tightly connected
- Scalability: Single processor means hard to increase volume of processing
- Portability: Moving to a new machine may mean rewriting everything
- Maintenance: Changing one layer requires changing other layers

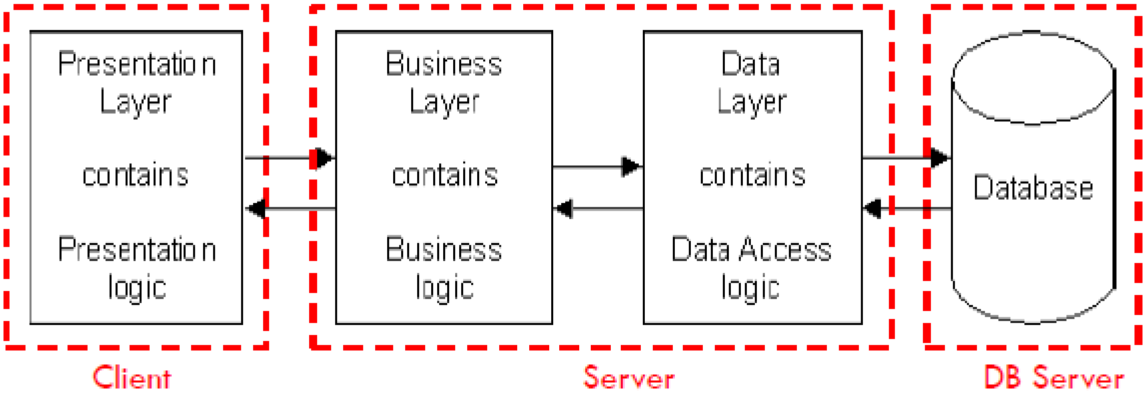
2-Tier Architecture



- Database runs on Server
- Separated from client
- Easy to switch to a different database
- Presentation and logic layers still tightly connected
- Heavy load on server

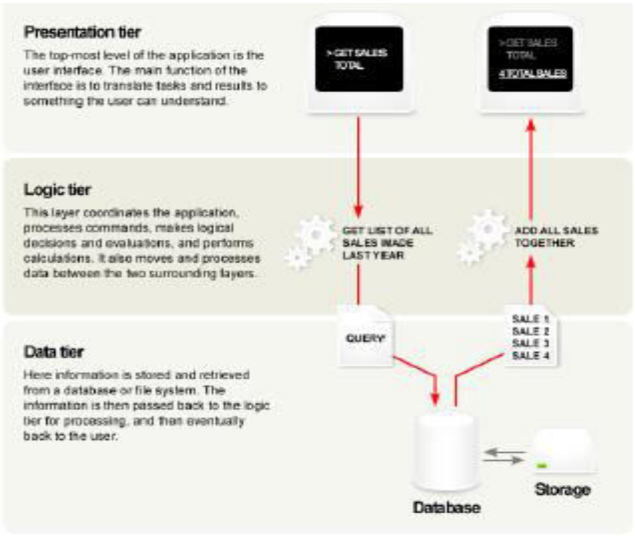
- Potential congestion on network
- Presentation still tied to business logic

3-Tier Architecture



- Each layer can potentially run on a different machine
- Presentation, logic, data layers disconnected

A Typical 3-tier Architecture


Presentation tier

The top-most level of the application is the user interface. The main function of the interface is to translate tasks and results to something the user can understand.

Logic tier

This layer coordinates the application, processes commands, makes logical decisions and evaluations, and performs calculations. It also moves and processes data between the two surrounding layers.

Data tier

Here information is stored and retrieved from a database or file system. The information is then passed back to the logic tier for processing, and then eventually back to the user.

Architecture Principles

- Client-server architecture
- Each tier (Presentation, Logic, Data) should be independent and should not expose dependencies related to the implementation
- Unconnected tiers should not communicate
- Change in platform affects only the layer running on that particular platform

Presentation Layer

- Provides user interface
- Handles the interaction with the user
- Sometimes called the GUI or client view or front-end
- Should not contain business logic or data access code

Logic Layer

- The set of rules for processing information
- Can accommodate many users
- Sometimes called・middleware/ back-end
- Should not contain presentation or data access code

Data Layer

- The physical storage layer for data persistence
- Manages access to DB or file system
- Sometimes called back-end
- Should not contain presentation or business logic code

**The 3-Tier Architecture for Web Apps**

Presentation Layer

->Static or dynamically generated content rendered by the browser (front-end)

Logic Layer

->A dynamic content processing and generation level application server, e.g., Java EE, ASP.NET, PHP, ColdFusion platform (middleware)

Data Layer

->A database, comprising both data sets and the database management system or RDBMS software that manages and provides access to the data (back-end)

**3-Tier Architecture – Advantages**

- Independence of Layers
- Easier to maintain
- Components are reusable
- Faster development (division of work)
    - Web designer does presentation
    - Software engineer does logic
    - DB admin does data model

| S.NO | RGPV QUESTIONS | Year | Marks |
|---|---|---|---|
| Q.1 | Explain three tier web architecture with suitable diagram | June.2010 | 12 |

## UNIT-01/LECTURE-06

**Services of web server**

A Web server is a program that, using the client/server model and the World Wide Web's Hypertext Transfer Protocol ( HTTP ), serves the files that form Web pages to Web users (whose computers contain HTTP clients that forward their requests). Every computer on the Internet that contains a Web site must have a Web server program. Two leading Web servers are Apache , the most widely-installed Web server, and Microsoft's Internet Information Server ( IIS ). Other Web servers include Novell's Web Server for users of its NetWare operating system and IBM's family of Lotus Domino servers, primarily for IBM's OS/390 and AS/400 customers.

Apache is the most popular UNIX web server today. Apache was originally based on code and ideas found in the most popular HTTP server of the time, NCSA httpd 1.3 (early 1995). It has since evolved into a far superior system which can rival (and probably surpass) almost any other UNIX based HTTP server in terms of functionality, efficiency and speed. Take a look at the web server feature chart to see how Apache ranks among the competition.Open Market provides software products that are used to develop the infrastructure for Internet commerce. They pride themselves on scalability, content flexibility, lower entry and maintenance costs, and enhanced security.

Netscape sells several web server software packages. The Netscape Enterprise Server offers built in advanced services such as Internet-based access controls, automatic link management, and revision control. The FastTrack Server is an easy-to-use entry-level Web server designed to let novices create and manage a Web site.

IBM's Secure Server is provided for AIX, HP-UX, and Solaris, as well as NT and OS/2. Version 4.2 servers include enhanced scalability, browser-specific response capability, enhanced CGI support, PICS support, and HTTP Version 1.1 compliance. The IB servers have consistent configuration, management, and API interfaces across all of their supported platforms.

Jigsaw is W3C's sample implementation of HTTP, a full blown HTTP server entirely written in Java. Its design goals were: will run on any machine running Java, can be extended by writing new resource objects (a replacement for CGI), minimization of file system accesses.

WebSTAR is a Mac HTTP server which performs dynamic web server file caching, has the ability to run server side Java applets, contains an administration plug-in that lets one administer essential server functions from any web browser on the Internet, honors keep-alive requests, supports a 20,000 username/passwd database, has integrated support for image maps, supports common log format, supports cgi-bin folder, does on the fly bin-hexing of Mac files, and supports an expanded command set for server-side includes.

**Common gateway interface (CGI) : [RGPV/Jun 2012,14(7)]**

An HTTP server is often used as a gateway to a legacy information system; for example, an existing body of documents or an existing database application. The Common Gateway Interface is an agreement between HTTP server implementors about how to integrate such gateway scripts and programs.

It is typically used in conjunction with HTML forms to build database applications.

The Common Gateway Interface (CGI) is a standard for interfacing external applications with information servers, such as HTTP or Web servers. A plain HTML document that the Web daemon retrieves is static, which means it exists in a constant state: a text file that doesn't change. A CGI program, on the other hand, is executed in real-time, so that it can output dynamic information.

For example, let's say that you wanted to "hook up" your Unix database to the World Wide Web, to allow people from all over the world to query it. Basically, you need to create a CGI program that the Web daemon will execute to transmit information to the database engine, and receive the results back again and display them to the client. This is an example of a gateway, and this is where CGI, currently version 1.1, got its origins.

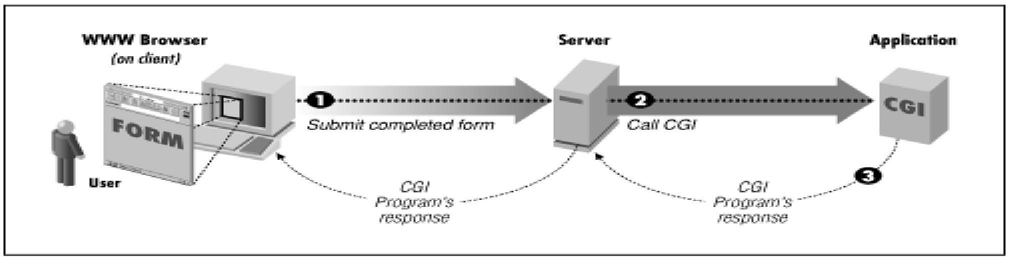
The database example is a simple idea, but most of the time rather difficult to implement. There really is no limit as to what you can hook up to the Web. The only thing you need to remember is that whatever your CGI program does, it should not take too long to process. Otherwise, the user will just be staring at their browser waiting for something to happen.

As you traverse the vast frontier of the World Wide Web, you will come across documents that make you wonder, "How did they do this?" These documents could consist of, among other things, forms that ask for feedback or registration information, imagemaps that allow you to click on various parts of the image, counters that display the number of users that accessed the document, and utilities that allow you to search databases for particular information. In most cases, you'll find that these effects were achieved using the Common Gateway Interface, commonly known as CGI.

One of the Internet's worst-kept secrets is that CGI is astoundingly simple. That is, it's trivial in design, and anyone with an iota of programming experience can write rudimentary scripts that work. It's only when your needs are more demanding that you have to master the more complex workings of the Web. In a way, CGI is easy the same way cooking is easy: anyone can toast a muffin or poach an egg. It's only when you want a Hollandaise sauce that things start to get complicated.

CGI is the part of the Web server that can communicate with other programs running on the server. With CGI, the Web server can call up a program, while passing user-specific data to the program (such as what host the user is connecting from, or input the user has supplied using HTML form syntax). The program then processes that data and the server passes the program's response back to the Web browser.

CGI isn't magic; it's just programming with some special types of input and a few strict rules on program output. Everything in between is just programming. Of course, there are special techniques that are particular to CGI, and that's what this book is mostly about. But underlying it all is the simple model shown in Figure.



**Simple diagram of CGI**

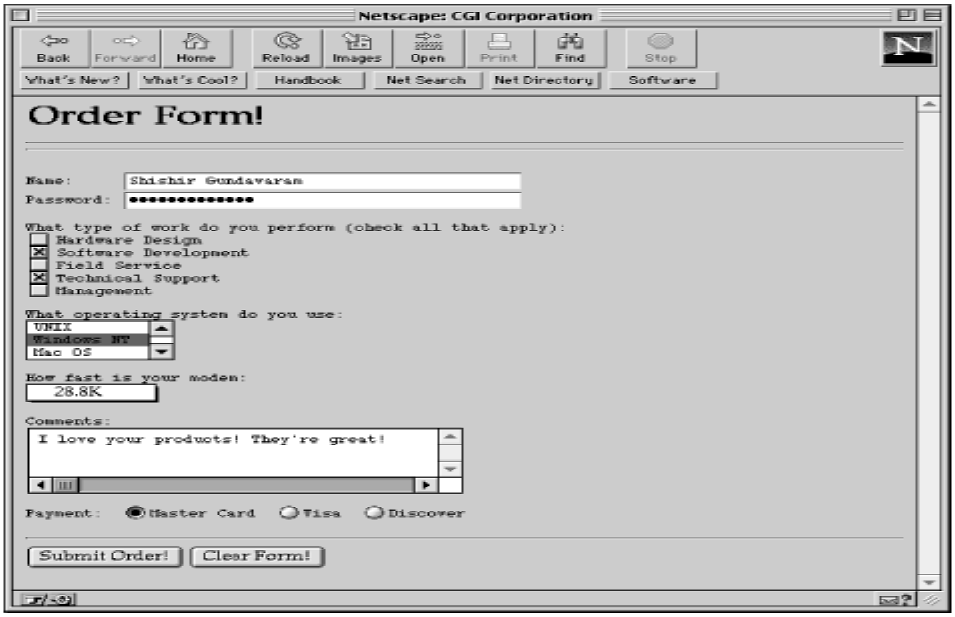
CGI Applications

CGI turns the Web from a simple collection of static hypermedia documents into a whole new interactive medium, in which users can ask questions and run applications. Let's take a look at some of the possible applications that can be designed using CGI.

Forms

One of the most prominent uses of CGI is in processing forms. Forms are a subset of HTML that allow the user to supply information. The forms interface makes Web browsing an interactive process for the user and the provider. Figure shows a simple form.

Figure: Simple form illustrating different widgets



Order Form!

Name: Shishir Gundavaram

Password: ••••••••••••••

What type of work do you perform (check all that apply):

□ Hardware Design
☒ Software Development
□ Field Service
☒ Technical Support
□ Management

What operating system do you use:

UNIX
Windows NT
Mac OS

How fast is your modem: 28.8K

Comments:

I love your products! They're great!

Payment: ● Master Card ○ Visa ○ Discover

Submit Order! Clear Form!

As can be seen from the figure, a number of graphical widgets are available for form creation, such as radio buttons, text fields, checkboxes, and selection lists. When the form is completed by the user, the Submit Order! button is used to send the information to the server, which executes the program associated with the particular form to "decode" the data.

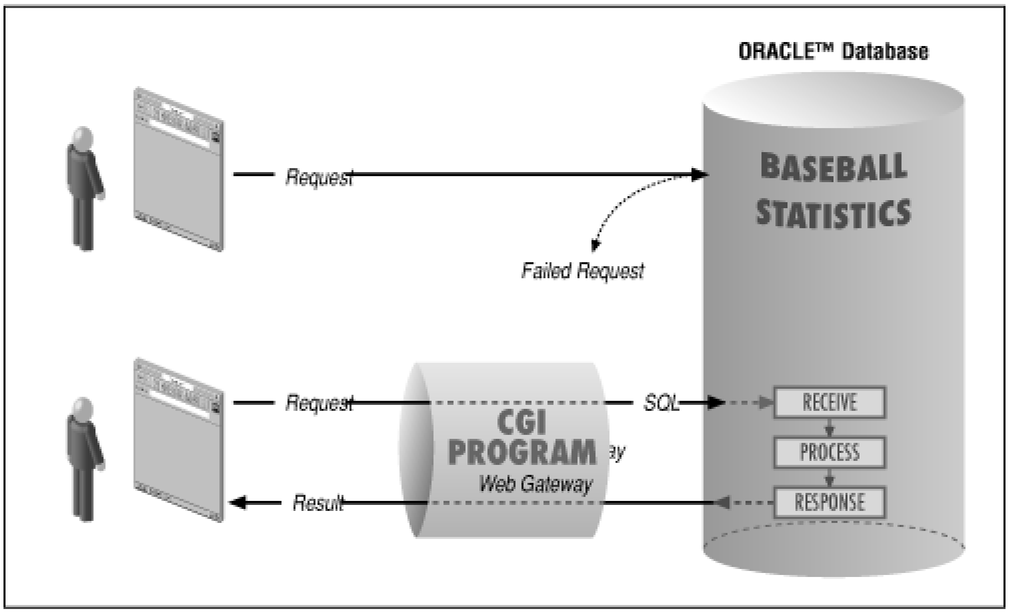
Generally, forms are used for two main purposes. At their simplest, forms can be used to collect information from the user. But they can also be used in a more complex manner to provide back-and-forth interaction. For example, the user can be presented with a form listing the various documents available on the server, as well as an option to search for particular information within these documents. A CGI program can process this information and return document(s) that match the user's selection criteria.

Gateways

Web gateways are programs or scripts used to access information that is not directly readable by the client. For example, say you have an Oracle database that contains baseball statistics for all the players on your company team and you would like to provide this information on the Web. How would you do it? You certainly cannot point your client to the database file (i.e., open the URL associated with the file) and expect to see any meaningful data.

CGI provides a solution to the problem in the form of a gateway. You can use a language such as oraperl (see Chapter 9, Gateways, Databases, and Search/Index Utilities, for more information) or a DBI extension to Perl to form SQL queries to read the information contained within the database. Once you have the information, you can format and send it to the client. In this case, the CGI program serves as a gateway to the Oracle database, as shown in Figure

Figure: A gateway to a database



Virtual Documents

Virtual, or dynamic, document creation is at the heart of CGI. Virtual documents are created on the fly in response to a user's information request. You can create virtual HTML, plain text, image, and even audio documents. A simple example of a virtual document could be something as trivial as this:

Welcome to Shishir's WWW Server!

You are visiting from diamond.com. The load average on this machine is 1.25. Happy navigating! In this example, there are two pieces of dynamic information: the alphanumeric address (IP name) of the remote user and the load average on the serving machine. This is a very simple example, indeed!

On the other hand, very complex virtual documents can be created by writing programs that use a combination of graphics libraries, gateways, and forms. As a more sophisticated example, say you are the manager of an art gallery that specializes in selling replicas of ancient Renaissance paintings and you are interested in presenting images of these masterpieces on the Web. You start out by creating a form that asks for user information for the purpose of promotional mailings, presents a search field for the user to enter the name of a painting, as well as a selection list containing popular paintings. Once the user submits the form to the server, a program can email the user information to a certain address, or store it in a file. And depending on the user's selection, either a message stating that the painting does not exist or an image of the painting can be displayed along with some historical information located elsewhere on the Internet.

Along with the picture and history, another form with several image processing options to modify the brightness, contrast, and/or size of the picture can be displayed. You can write another CGI program to modify the image properties on the fly using certain graphics libraries, such as gd, sending the resultant picture to the client.

This is an example of a more complex CGI program using many aspects of CGI programming.

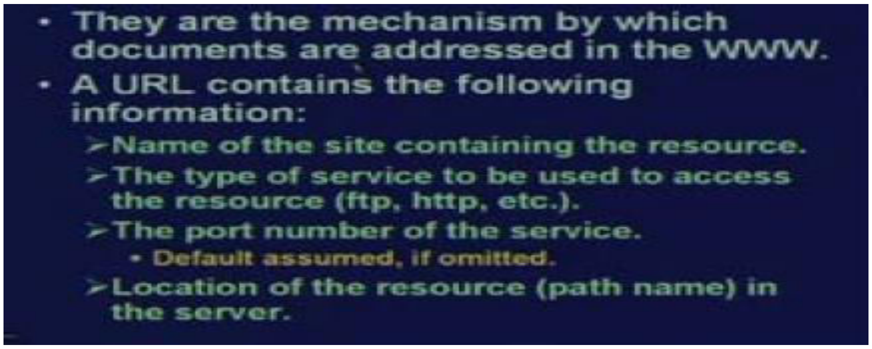
| S.NO | RGPV QUESTION | YEAR | MARKS |
|---|---|---|---|
| Q.1 | Write a brief notes on common gateway interface | June.2014 | 07 |
| Q.2 | What is CGI? What are the limitations in developing CGI | June.2012 | 05 |

| | | | |
|---|---|---|---|
| | application? | | |

## UNIT-01/LECTURE-07

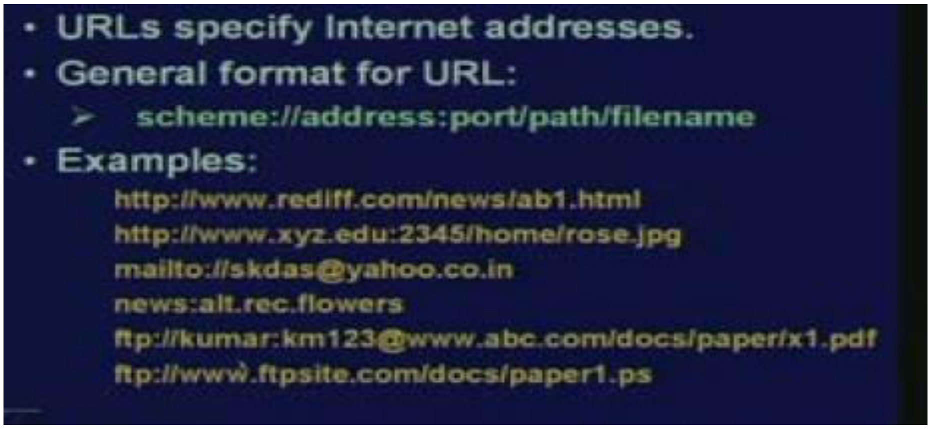
**Uniform Resource Locator (URL): : [RGPV/Jun 2012(10)]**

URL are nothing but mechanisms to locate some resource on the internet. In the simple case I want to locate that particular file in the wed server that is the example of URL. URL is the short form for Uniform Resource Locator.



- They are the mechanism by which documents are addressed in the WWW.
- A URL contains the following information:
  - ➢Name of the site containing the resource.
  - ➢The type of service to be used to access the resource (ftp, http, etc.).
  - ➢The port number of the service.
    - Default assumed, if omitted.
  - ➢Location of the resource (path name) in the server.

URL is a mechanism or it is a format whatever you call by which documents or resources can be addressed in the internet in the World Wide Web. A URL contains several information. First of course the name or the address of the site or the server where the actual resource you are trying to address is located. Secondly what is the type of service you want to use to access the resource? Do you want to use Http. So you want to use http or something else. Port number of the service well in most cases some default port number is assumed. But you can explicitly specify a port number also and once you have look at it or specified everything. You may also specify a path name on the server which specifies where the resource is located on that particular server. So all this things taken together will actually identify where the server will actual where the particular resource you are trying to address is actually located on the server.



- URLs specify Internet addresses.
- General format for URL:
  - scheme://address:port/path/filename
- Examples:

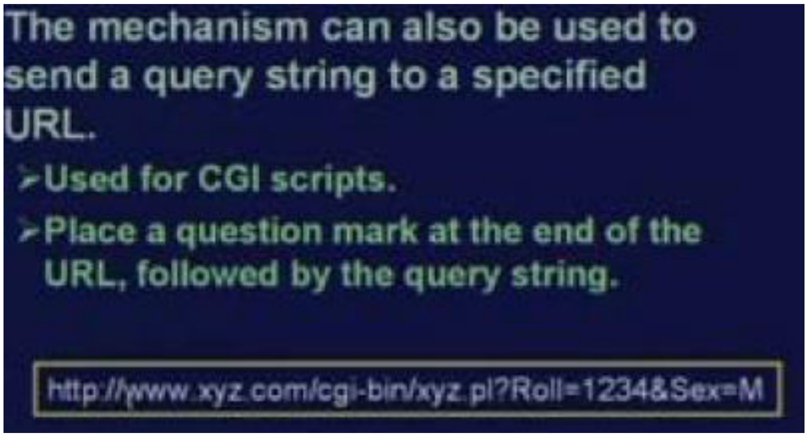
  http://www.rediff.com/news/ab1.html
  http://www.xyz.edu:2345/home/rose.jpg
  mailto://skdas@yahoo.co.in
  news:alt.rec.flowers
  ftp://kumar:km123@www.abc.com/docs/paper/x1.pdf
  ftp://www.ftpsite.com/docs/paper1.ps

Now let us see what are the different URL types. These URLs are nothing but a form of specifying internet addresses. So the whatever you have specified, it starts with a scheme this is the syntax scheme which access method you are using followed by a colon double slash where there is one example, where you do not need the double slash followed by the address of the machine. This colon port is optional if you want to specify an explicit port number give colon port followed by a path name some examples follow.

The first example is a URL which says it is an http. This is the name of the server and this is the path of the resource the resource is ab1.html. The second one is also http, but it specify an explicit port number here the http server is running on port number 2345. The resource is rose.jpg in this path. The third is an example of an email mail to at the end you have the email address. Well if you have this mail to URL as a part of a web page. If you click on it automatically a mail window will open where you can type in a mail. This is the reason why mail to URL is required.

NEWS is one example where double slash is not required. This refers to NEWS groups FTP is for file transfer in the first case after the slash, you give the user name colon; you give the password, then this at the rate sign. Then the machine where you want to do a FTP then this the path name where the path name or resource is located. The last example refers to anonymous ftp. But user name and password are not given so if the user name and password are missing then by default it is taken to the anonymous FTP and this is the address of a machine and this is the address of the document or resource under the anonymous directory.

**Sending a Query String**



The mechanism can also be used to send a query string to a specified URL.

- Used for CGI scripts.
- Place a question mark at the end of the URL, followed by the query string.

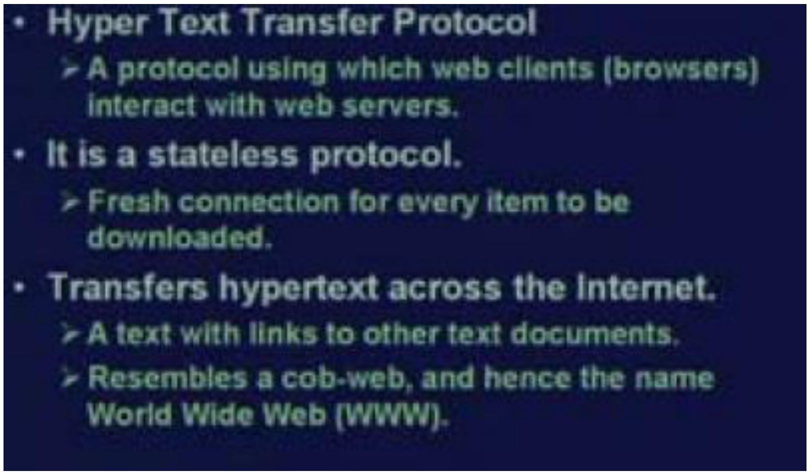
http://www.xyz.com/cgi-bin/xyz.pl?Roll=1234&Sex=M

And this URL can also be used to send a send query string like the example that we had given earlier for GET. Well GET is a way to send it as a as a http string. But even when you type a URL on the browser there also you can explicitly specify that. Like an example follows, it says http. This is the name of the machine; this is the path name of the resource. Here this path name refers to a cgi program again a question mark followed by some additional information. Now this whole thing is a URL. This you can type on a browser and press enter. What the browser will do it will generate a get http request from this and it will send it to the requested server. So actually what will go to the server will be the same GET request as I mentioned earlier. But this how you can also specify in an URL as part of http. So with this we come to the end of first part of our discussion on World Wide Web. Now let us quickly have a look at the answers to the questions that where POST in our last class.

| S.NO | RGPV QUESTION | YEAR | MARKS |
|---|---|---|---|
| Q.1 | What is URL? Explain URL Syntax with example. Also explain three parts of URL. | June.2012 | 10 |

## UNIT-01/LECTURE-08

**Hyper Text Transfer Protocol (HTTP): : [RGPV/Dec 2013(7)]**

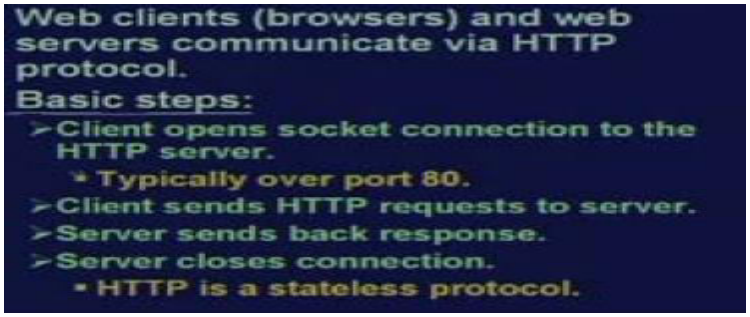


- Hyper Text Transfer Protocol
  - A protocol using which web clients (browsers) interact with web servers.
- It is a stateless protocol.
  - Fresh connection for every item to be downloaded.
- Transfers hypertext across the Internet.
  - A text with links to other text documents.
  - Resembles a cob-web, and hence the name World Wide Web (WWW).

Now this http is the driving force behind the World Wide Web. So when we talk about the World Wide Web; when you talk about web server it is the http protocol we are talking about. So the web server is running a program which is understanding the http protocol any client can send http request it will get back a http response or reply. So let us say what this http request and response is look like. So first let us try to understand http. The hypertext transport protocol transfer protocol these are protocol which the web clients which the browsers are typical example of web clients interact with web servers. This is the language using which a browser and a web server talk along with themselves. The hypertext transfer protocol the first version at least it is a stateless protocol stateless protocol means no history is maintained. Every time a request is sent a fresh connection has to be established.

If you are requesting for three documents from the same server you have to establish connection three times. Not that you are connecting only once and transferring three documents at the same time. There are of course some subsequent versions of http which allows the so called persistent connections where you can make multiple transactions over a single connection. This http hypertext across the internet sees the documents that are stored in the webpage. As I said they are some pages with links to other pages. These are called hypertext, so hypertexts are texts with links to other documents. So this http protocol allows transfer from one document to another using this hyper text by clicking a link you can go to some other place. There should be a mechanism in built in the http protocol which should support this kind of a thing.

**HTTP Protocol**



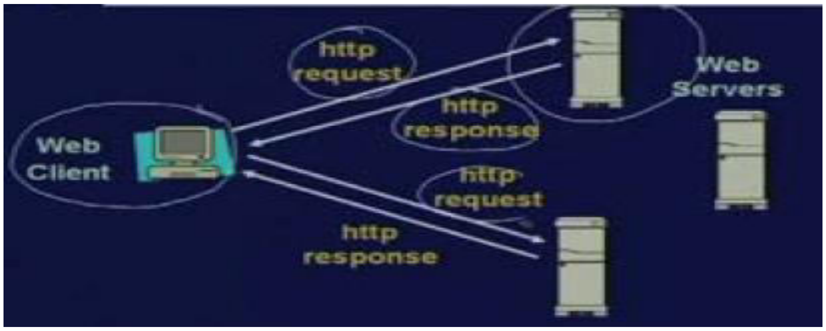
Web clients (browsers) and web servers communicate via HTTP protocol.

Basic steps:

- Client opens socket connection to the HTTP server.
  - Typically over port 80.
- Client sends HTTP requests to server.
- Server sends back response.
- Server closes connection.
  - HTTP is a stateless protocol.

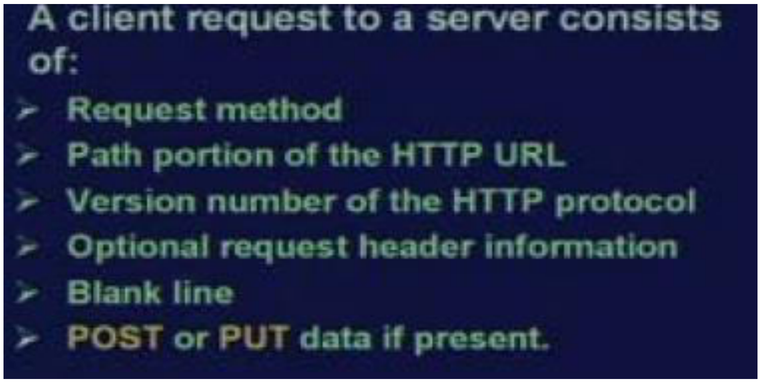
So as I mentioned browsers and web server communicate using http the basic steps look like this. The client first opens a socket connection to the http server typically the http server runs over port number 80 but some web servers also run over some other port number like 8080 or something else also. So you must know the port number on which the server is running. So that you can establish a connection to it typically by default the port number is 80. So after connection establish the client sends http request to the server. Server will send back response and the server will immediately close down the connection. This is the state less mode. For every connection you are opening and closing the connection, there is another request then the whole process will we will have to repeat it again. Again start a connection, again transact and then close.

**Illustration**



So just look at this diagram, suppose you are sitting here. This is a web client here you are using a browser here when you are typing the address of the site say www.yahoo.com, say this is the yahoo.com server. So an http request has been sent to the yahoo.com server and the server sends back a response which is actually an html page which gets displayed on your screen. Now after the html page comes back on your page and if you click on one of those so called hyper link switch which links to other documents. Then what happens? Now another http request goes to fix the other document the other document is possibly residing in some other web server and that some other web server will again be sending you back response and this will continue. Sending a request getting back a response clicking a link going to some other server, this continues. This is called web browsing and this browsing can take you to one server to another across the world. You really do not know that where you have started and where you have ultimately stops in this process.

**HTTP Request Format**

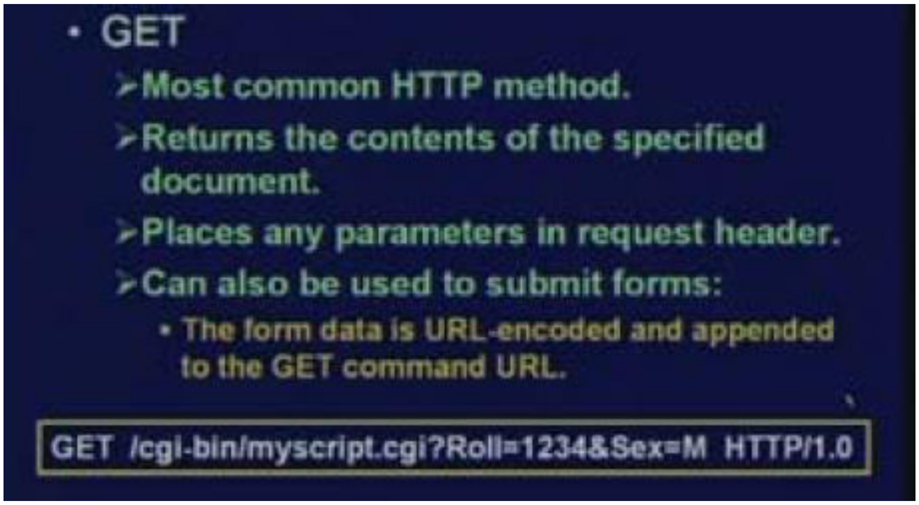


A client request to a server consists of:

- Request method
- Path portion of the HTTP URL
- Version number of the HTTP protocol
- Optional request header information
- Blank line
- POST or PUT data if present.

Now talking about the http request format a client request to the server consists of the following. It consists of a request method there are several request methods we shall be talking about get head POST port etcetera. The path position of the http URL is URL is a way of specifying the path we shall be talking about URL later in the lecture. Version number of the http protocol and for some of the request methods you may need some optional header information and in case you need to supply some additional data. The data has to be supplied after a blank line in between. This blank line acts as a delimiter between the header portion and the additional data that you are supplying. Some of the request methods like POST or PUT, they need the additional data. But for those method which do not need the data that part may not be there.

**HTTP Request Methods**



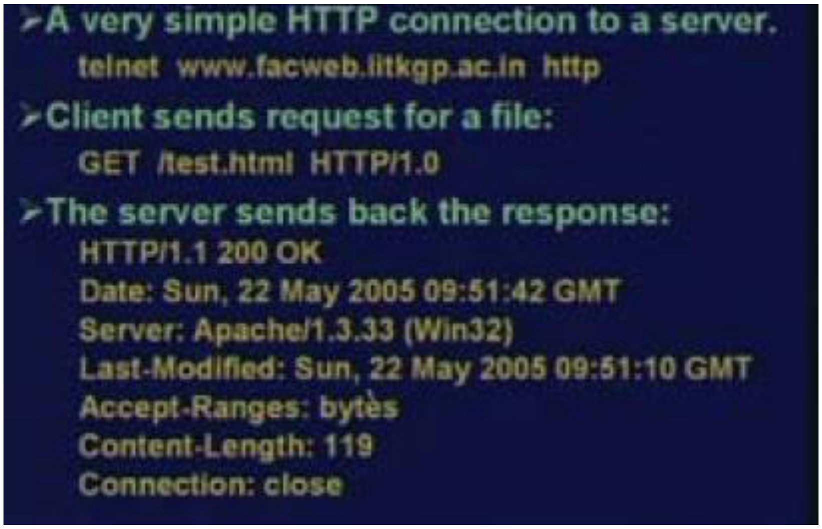
- GET
  - Most common HTTP method.
  - Returns the contents of the specified document.
  - Places any parameters in request header.
  - Can also be used to submit forms:
    - The form data is URL-encoded and appended to the GET command URL.

```
GET  /cgi-bin/myscript.cgi?Roll=1234&Sex=M  HTTP/1.0
```

So let us look at some of the important request methods one by one. GET is the most widely used method this is the method which is used to fetch a web page from a server. It returns the contents of the specified document. GET can place any parameter that you need in the request header itself. It does not need any additional lines of header information like an example is shown here. This, the get command here we are putting some additional information out here followed by the http version number. You see the first part of it cgi bin myscript.cgi specifies a path to a particular resource myscript.cgi. But after that following this question mark there is some additional data roll is equal to 1264 amber cent sex equal to m. This actually comes from so called forms. Well we shall be studying forms later when you study html in detail.

See form is something like you have seen many work pages like the search engines where some blank form like place come where you can type in something press enter or search something else comes back. So whatever you are typing in those boxes empty boxes and pressing enter you are actually filling up a form and submitting it when the data goes back to the web server it goes in this form. For example there was one space for roll no one place of sex I have typed in 1234 and m and this what which goes. This is the data you get filling you physically fill up in the form and the way this roll number is equal to amber sent this is called URL. It is URL encoding this we shall study detail when you talk about cgi scripting and html forms later in course. Of this lecture but this example shows how you can give this kind of information along with the get command.

**Illustration of GET**



- A very simple HTTP connection to a server.

```
telnet  www.facweb.iitkgp.ac.in  http
```

- Client sends request for a file:

```
GET  /test.html  HTTP/1.0
```

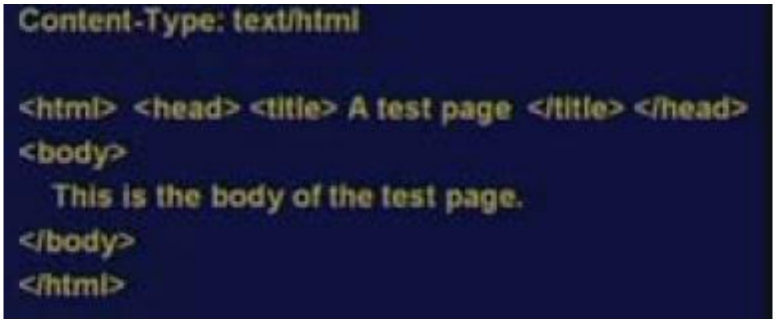
- The server sends back the response:

```
HTTP/1.1 200 OK
Date: Sun, 22 May 2005 09:51:42 GMT
Server: Apache/1.3.33 (Win32)
Last-Modified: Sun, 22 May 2005 09:51:10 GMT
Accept-Ranges: bytes
Content-Length: 119
Connection: close
```

Now a very simple illustration of GET server trying to retrieve a simple documents no sending of form data. A simple document see to start with it you can just open a command window again and you can start this transaction. You can give a command telnet you can give the name of the server to which you want to connect and http you can either give http or you can give the default port number 80 whatever. This http will mean that the default port number of the http protocol well after this connection is done. You will get back a screen with nothing displayed which means that the server is expecting you to type something. So you type a command like this get is the request method slash test dot html is the document you are requesting and this is the version of the http.

The server will be sending back a response like this. First you see there are several lines of header http. The http version name of the server it is running “1.1” version date server type last modified this document when it was last modified. Accept ranges of course this is a optional we will talk about it later content length what is the size of the document connection close. Means this is the default approach that means after the transaction is over you tell me the d http connection. This is the so called stateless approach that I was talking about. So instead of close you can also specify connection keep alive that is supported in the http version “1.1” or you can leave the connection open by default.

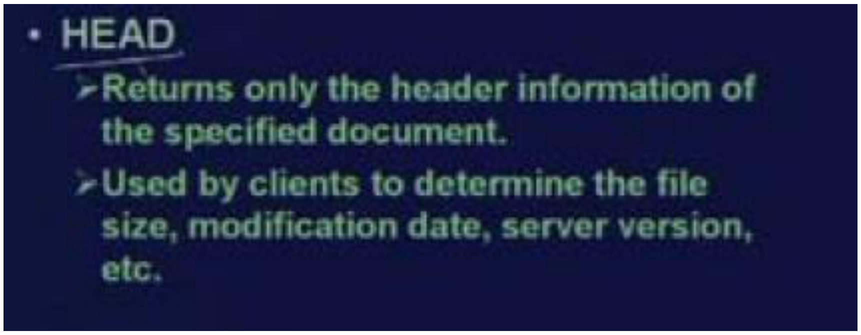
**Illustration of GET(Cont..)**



```
Content-Type: text/html

<html>  <head> <title> A test page  </title> </head>
<body>
   This is the body of the test page.
</body>
</html>
```

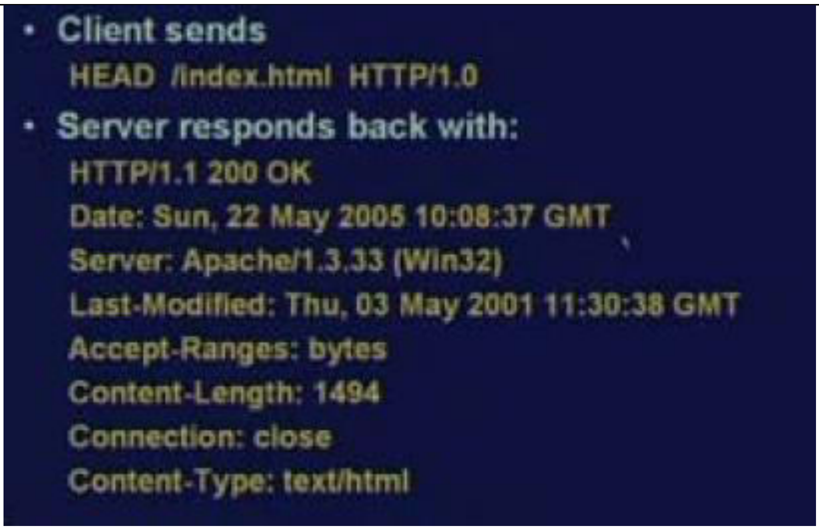
And after these lines come this content type. These are text of type html here there is a blank line you see there is a blank line in between this blank line indicates or it acts a delimiter that now the actual data follows. So the actual content of the file follows after this. This is a free simple html file which is created to give you this demonstration. So the content of the html file will come after that. So you see that this command get command can be used to fetch an html file from a server if you know of course the name of the file.



- HEAD
  - ➢Returns only the header information of the specified document.
  - ➢Used by clients to determine the file size, modification date, server version, etc.

There is another request method called head which returns only the header information. Well you may not be interested to get the whole file; but you need some information about the file like the file size when was it last modified server version etcetera.

**Illustration of HEAD**



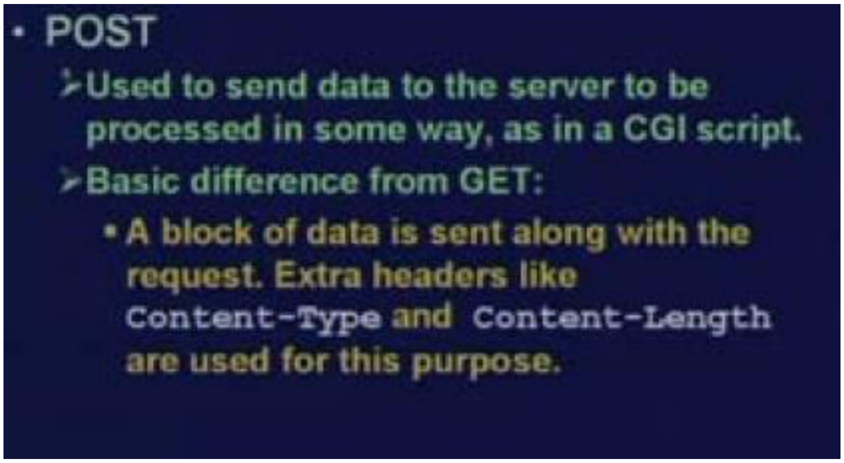
- Client sends

```
HEAD /index.html HTTP/1.0
```

- Server responds back with:

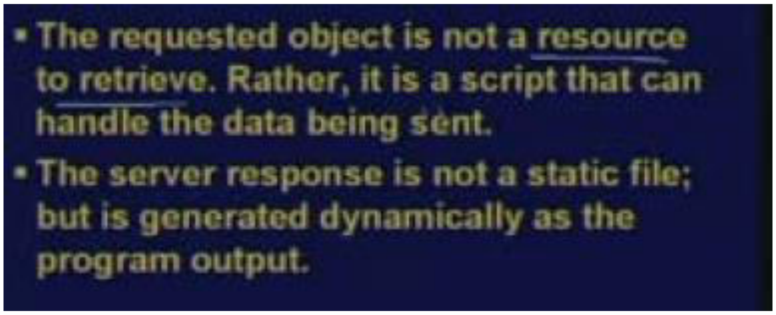
```
HTTP/1.1 200 OK
Date: Sun, 22 May 2005 10:08:37 GMT
Server: Apache/1.3.33 (Win32)
Last-Modified: Thu, 03 May 2001 11:30:38 GMT
Accept-Ranges: bytes
Content-Length: 1494
Connection: close
Content-Type: text/html
```

So a typical session with head will look like this after doing the telnet, you send a command like this. Head name of a file followed by http version you get back only the header portion. What is missing is actually the body of the contents of the file that you are not requesting here. You are requesting only the header portion. So only the header portion is coming back to you and you can have a look at the header position to see what it contains. If example this, file size is 1494 bytes.



- POST
  - ➢Used to send data to the server to be processed in some way, as in a CGI script.
  - ➢Basic difference from GET:
    - A block of data is sent along with the request. Extra headers like `Content-Type` and `Content-Length` are used for this purpose.

POST is another request method. POST is again another method which is used in conjunction with the submission of the form. So we will be talking about POST now. But you can actually put it in a proper cont to a context when we discuss suggest script later during our class. Now this POST is used to send some additional data to the server to be processed typically by a cgi script; cgi script is a program cgi stands for common gateway interface it is a program which running on the server side. Whenever you fill up a form typically the data in the form goes to the cgi script; the cgi script reads the data and does some processing and generates some output which comes back to the client and the browser. It gets displaced there; this is how cgi scripts work. Now this POST request method allows you send data to the cgi script, but the why this data sent is different from the gate approach that we have discussed. But this uses two addition header type; content type and content length.



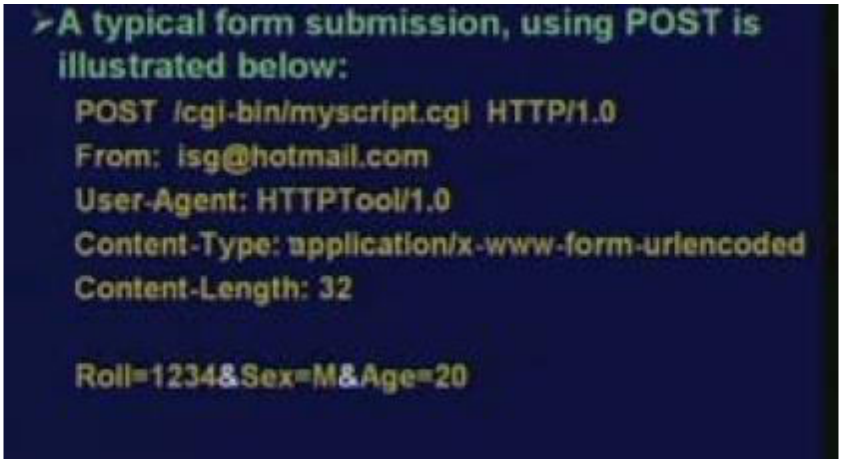
- The requested object is not a resource to retrieve. Rather, it is a script that can handle the data being sent.
- The server response is not a static file; but is generated dynamically as the program output.

The requested object is not a resource to retrieve. Rather it is a script an executable script. Similarly server response which comes back is not a static file. But rather it is the output of the cgi program which I have seen. So the server is not sending you back the contents of a file which already there. Rather it is sending you back the output of that cgi program. For instance which is receiving the data executing it and sending back the output to you?

| S.NO | RGPV QUESTION | YEAR | MARKS |
|---|---|---|---|
| Q.1 | What is HTTP? Write and explain the structure of HTTP request. | Dec.2013 | 7 |

## UNIT-01/LECTURE-09

**Illustration of POST: : [RGPV/Jun 2014(7)]**



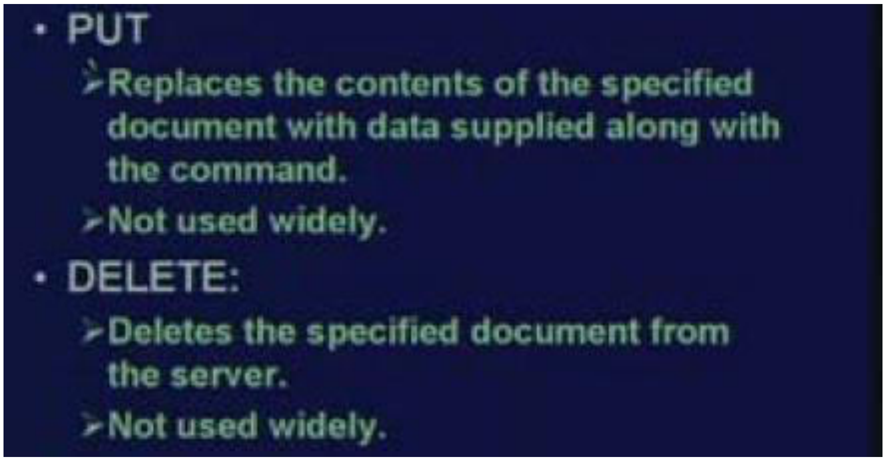
➢A typical form submission, using POST is illustrated below:

```
POST /cgi-bin/myscript.cgi HTTP/1.0
From: isg@hotmail.com
User-Agent: HTTPTool/1.0
Content-Type: application/x-www-form-urlencoded
Content-Length: 32

Roll=1234&Sex=M&Age=20
```

Now the POST command can be illustrated with the simple example. Well again after telnet, suppose you give a command like this, POST followed by a path name. This cgi slash bin myscript.cgi. This does not refer to a document or a resource which you want to download; rather this is the name of the program. The cgi script program for instance which you want to execute whenever this POST command is given. And this is the additional information you will also have supply after POST. See, in GET or head you give a one line command and a lot of response come back. But in a POST you will have to give a multiple line command. This POST is the first lines followed by from you have to identify yourself from where you are sending user agent the name or the version of the user agent that is running on your machine, content type or type of content it is. Well application something x -www form urlencoded this is an URL encoded see this is the actual content. The way this has been encoded this is called urlencoding and the name is x www form urlencoded and content length the total size is 32. There is a blank line in between followed by the actual body of this additional information. So in GET this information is put on the same line after a question mark; in POST it is put as part of the command after a blank line. Now, in POST you can put as much material as you can. So if we have a lot of data to send, POST is the preferred approach because in GET the total size of the string is limited by usually 256. The maximum size of the string that the machine can support. So other request method PUT.

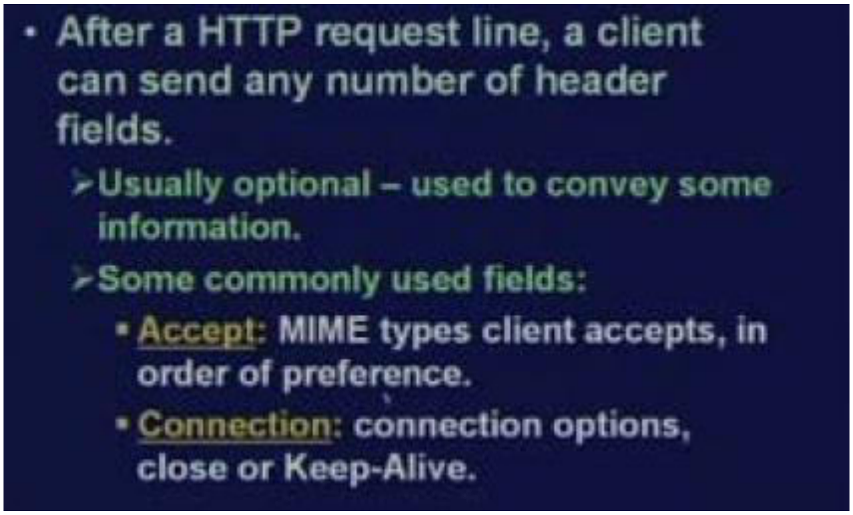
**HTTP Request Methods**



- PUT
  - Replaces the contents of the specified document with data supplied along with the command.
  - Not used widely.
- DELETE:
  - Deletes the specified document from the server.
  - Not used widely.

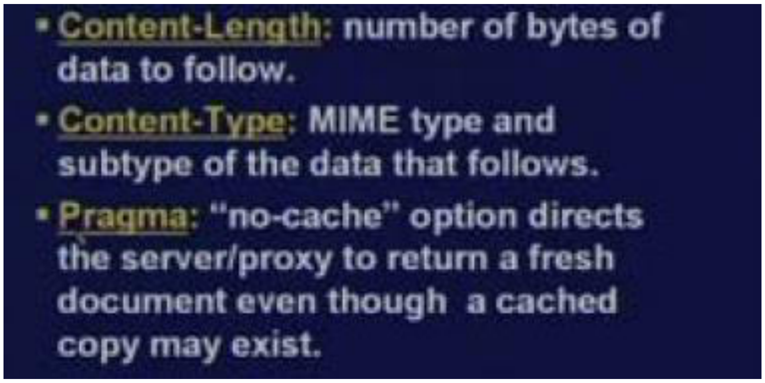
PUT is used to upload the contents of a webpage. It replaces the contents of the specified document with the data supplied along with the command. So here the data that was supplying, this you can supply exactly like the POST command. So after some initial header give a blank line, then give your entire web page which you want to upload. So the entire thing will be the data you are trying to send to the server; the server will handle it and it will be uploading it. But of course as you can understand. Not all web servers allow you uploading facility. Only a few web server will allow to do that and of course only after proper authentication. Delete is similar in a sense that this also makes updation this can delete a specified document this also is not used very widely.

HTTP Request Header:



- After a HTTP request line, a client can send any number of header fields.
  - Usually optional – used to convey some information.
  - Some commonly used fields:
    - Accept: MIME types client accepts, in order of preference.
    - Connection: connection options, close or Keep-Alive.

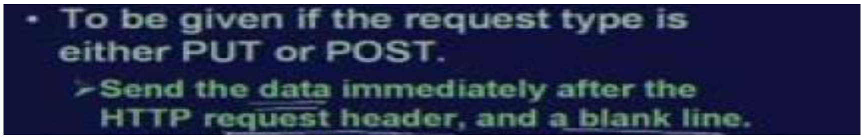
Only in some servers where you have proper authentication you can do that. And after sending the request, well. There some request header you see. That means when you are sending the request some request header information is required. Now after the initial line GET, POST, PUT, etcetera. So a client can send any number header lines. Some of these where illustrated with the example for POST. So after the POST command there were several additional lines of header followed by the data that were supplied. So this is usually optional for some request types like get you do not need it for some others you need it. Some of the common headers types are accept. This accept followed by a string indicates at which MIME types the client will accept. Connection. Connection is the connection type; if it is close you are asking it to close connection after every transaction. If it keeps alive, then you are saying that well it will not be stateless. You let the connection to be persistent I can send more than one file one transaction over that open connection. So you can specify this as part of the header.



- Content-Length: number of bytes of data to follow.
- Content-Type: MIME type and subtype of the data that follows.
- Pragma: "no-cache" option directs the server/proxy to return a fresh document even though a cached copy may exist.

Content length of course you have seen number of bytes of data. Content type what MIME type of data is there. There is a pragma header; pragma colon followed by no cache. This indicates that if you are redirecting your command to a proxy server. You are saying that you please do not send me my requested information from your cache you try to send me a fresh copy from the original server

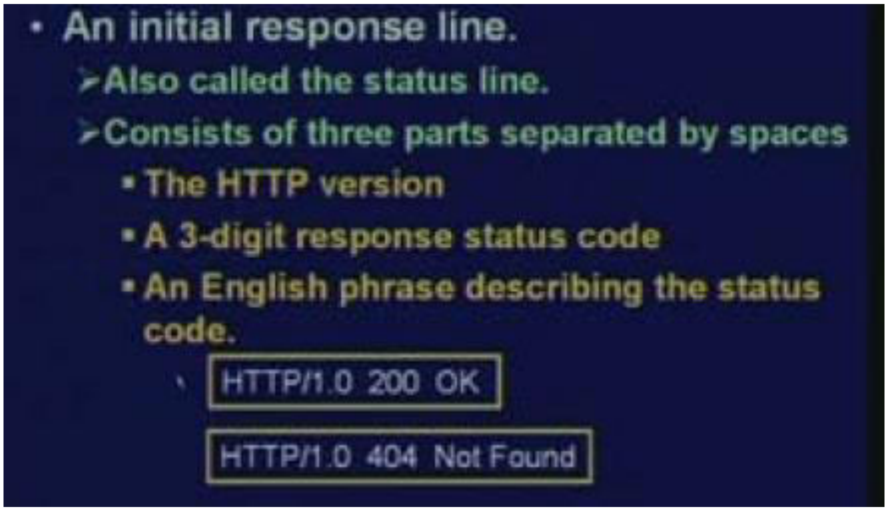
**HTTP Request Data**



- To be given if the request type is either PUT or POST.
  - ➢Send the data immediately after the HTTP request header, and a blank line.

So this pragma no cache tells you to return a fresh document well although a cached copy may exists, you may want sometime that you want the latest updated copy from the original server. And for command switch request we require request data like the POST and PUT. So you will have to put a blank line followed by the actual data. So there will be the initial header followed by a blank line then the data part. This will be the structure of your overall http request.

**HTTP Response**

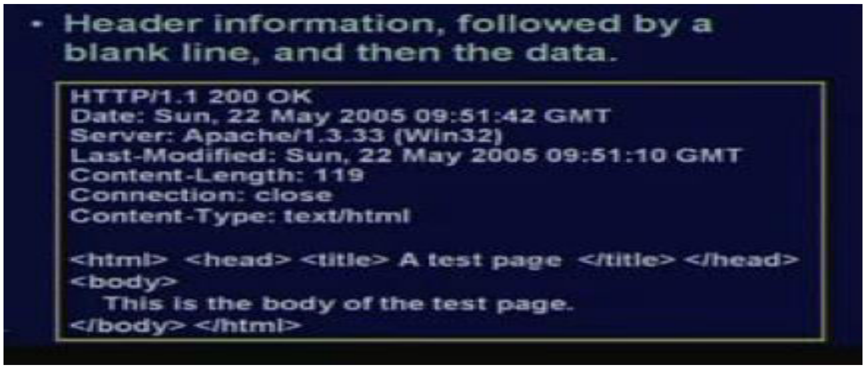


- An initial response line.
  - ➢Also called the status line.
  - ➢Consists of three parts separated by spaces
    - ▪The HTTP version
    - ▪A 3-digit response status code
    - ▪An English phrase describing the status code.

```
HTTP/1.0 200 OK
```

```
HTTP/1.0 404 Not Found
```

Now after the request is sent, now it is time to get back the responses. So the responses look similar the requests for most of them other than get. There are some standard request headers followed by optional data. The responses also look very similar. There will be an initial response line which is also called the status line; response status line. It consists of three parts the http version, a three digit response code and an English phrase indicating what type it is. Two examples I shown it will either be http version name 200 or version name 404 Not Found. This 200 404 are the error codes and Not Found are the error types.



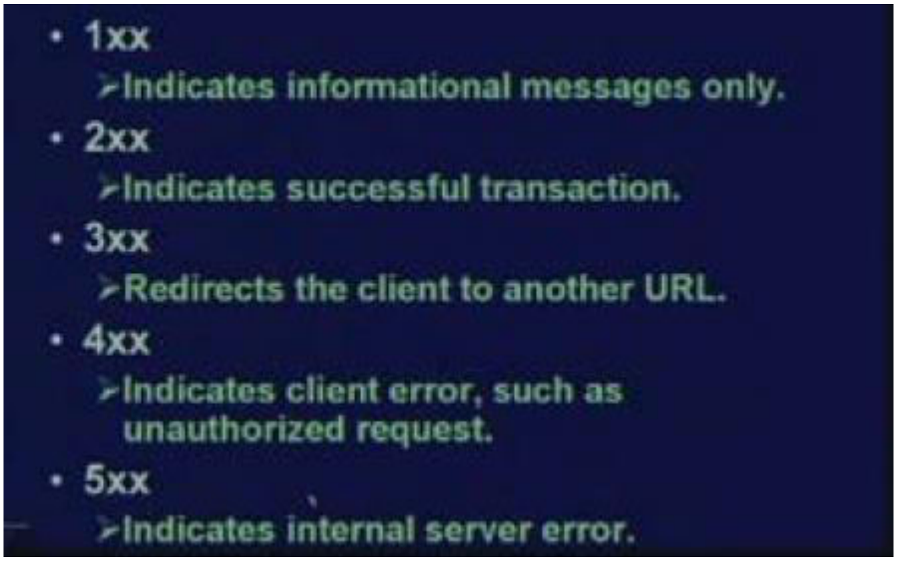
- Header information, followed by a blank line, and then the data.

```
HTTP/1.1 200 OK
Date: Sun, 22 May 2005 09:51:42 GMT
Server: Apache/1.3.33 (Win32)
Last-Modified: Sun, 22 May 2005 09:51:10 GMT
Content-Length: 119
Connection: close
Content-Type: text/html

<html>  <head> <title> A test page  </title> </head>
<body>
   This is the body of the test page.
</body> </html>
```

And after this initial status line will come, the actual content of the response. There will be some header at the beginning followed by the actual data and again a usual there will be blank line in between. So there will be a blank line separating them. So the header looks very much like the http request line. There will be some header types followed by the value. The content length, connection close, content type and at the end you will get the data that you are requesting.

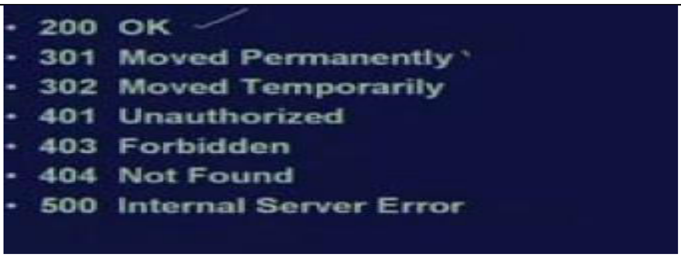
**3-digit Status Code:**



- 1xx
  - Indicates informational messages only.
- 2xx
  - Indicates successful transaction.
- 3xx
  - Redirects the client to another URL.
- 4xx
  - Indicates client error, such as unauthorized request.
- 5xx
  - Indicates internal server error.

And the 3 digit status code indicates anything starting with one, indicates that this is only information. This is not an error message. You need not do anything special with respect to this. This is just for your information some informational message. Two, anything starting with 2 indicates successful transaction. Anything starting with 3 indicates that you have been redirected to another URL; some servers supports redirection. Starting with 4 means some error condition; 5 means there is some internal server error. So these kinds of errors are reported with an error code starting with these initial digits.

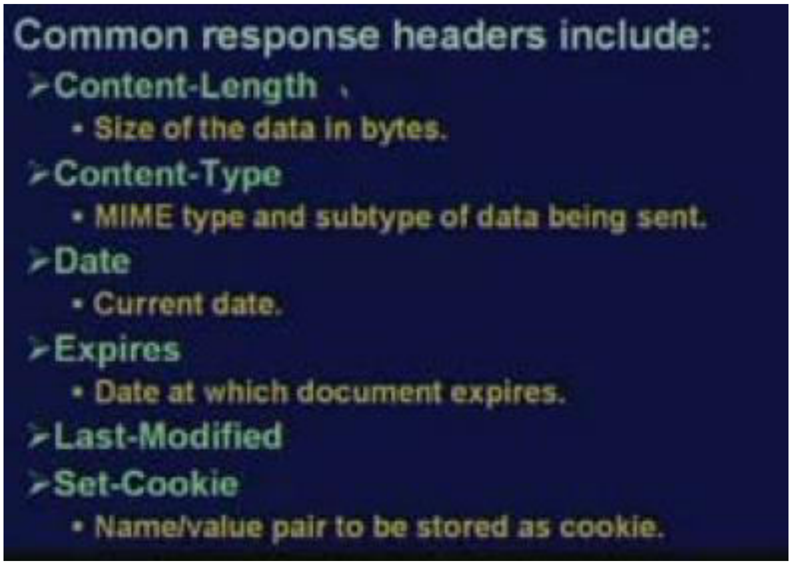
**Common Status Code**



- 200 OK
- 301 Moved Permanently
- 302 Moved Temporarily
- 401 Unauthorized
- 403 Forbidden
- 404 Not Found
- 500 Internal Server Error

Some of the common status codes are like this 200 means OK, 301 some documentary requesting has been moved permanently. 302 means temporarily; 40 means there is some authorization problem may be you have not supplied the password correctly. 403, some documents some access permissions are not there; 404 Not Found, 500 some server error.

**HTTP Response Header:**

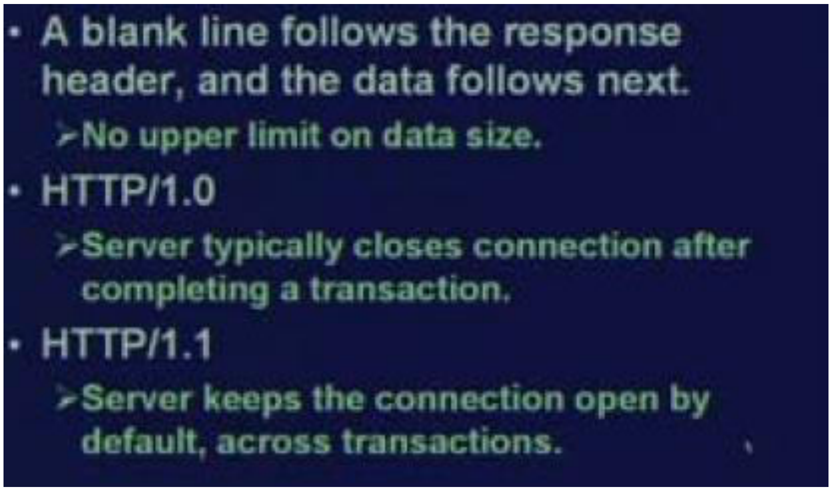


Common response headers include:

- Content-Length
  - Size of the data in bytes.
- Content-Type
  - MIME type and subtype of data being sent.
- Date
  - Current date.
- Expires
  - Date at which document expires.
- Last-Modified
- Set-Cookie
  - Name/value pair to be stored as cookie.

Now in the response header there are several things which are included like content length you already know it specify the size of the data in bytes. Content type, again MIME type and subtype what kind of data is coming. Date means the current date. Well each document that resides on the server can also have something like an expiry date associated with it. So whenever you are fetching a document it will also contain what is the expiry date? Some document may contain an expiry date; some may not contain. If it contains, then we will have a look at the expiry date and find out whether this is an absolute document or not. There is a last modified the name is obvious what it means and set-cookie. Set-cookie is another information which is sometimes sent back. See set-cookie followed by some name value pair. See whenever you want to have cookie in your browser, cookie has some values name and value pair which is sent from the other side which can be used by a browser to do some kind of a local processing. Like when you are maintaining a session you use cookies you maintain some information about the session and session id. You continuously check whether your present session variable matches with the session id stored in the cookie. If not you give an error message that your session is expired something like that. So this cookie can also be sent back.

**HTTP Response Data:**



- A blank line follows the response header, and the data follows next.
  - ➢No upper limit on data size.
- HTTP/1.0
  - ➢Server typically closes connection after completing a transaction.
- HTTP/1.1
  - ➢Server keeps the connection open by default, across transactions.

And after the initial header, again there will be a blank line which will be followed by the actual data. So the way the requests are send and the responses are sent back are very similar. They use the mind format for the heading a blank line then followed by the data. There is no upper limit to the data size. It can be very big also, but two versions of http which you see around you today. This http “1.0” is the older version. But most of the modern web servers use the later version “1.1”. Now in “1.0” the server typically closes connection after completing a transaction that is the default. Well in “1.1” the server keeps the connection, open it is persistent.

So it depends on the scenario that you are working in whether you want to keep it persistent or not. See for servers which are very heavily used keeping the connection persistent will add the load on the server. Because every machine has a maximum limit to the number of connections that it can handle simultaneously. So if it is a persistent and there are many connections which are coming and remaining because it is active for long time, it may so happen that the maximum limit may reach and after that request which are coming they will not be honored they will be denied. So for most servers which are pretty popularly used they prefer to use stateless or non-persistent connection type.

**Hyper Text Transfer Protocol Secure (HTTPS):**

Hypertext Transfer Protocol Secure (HTTPS) is a communications protocol for secure communication over a computer network, with especially wide deployment on the Internet. Technically, it is not a protocol in and of itself; rather, it is the result of simply layering the Hypertext Transfer Protocol (HTTP) on top of the SSL/TLS protocol, thus adding the security capabilities of SSL/TLS to standard HTTP communications. The main motivation for HTTPS is to prevent wiretapping and man-in-the-middle attacks.

The security of HTTPS is therefore that of the underlying TLS, which uses long-term public and secret keys to exchange a short term session key to encrypt the data flow between client and server. An important property in this context is perfect forward secrecy (PFS), so the short-term session key cannot be derived from the long-term asymmetric secret key; however, PFS is not widely adopted.

X.509 certificates are used to guarantee one is talking to the partner with whom one wants to talk. As a consequence, certificate authorities and a public key infrastructure are necessary to verify the relation between the owner of a certificate and the certificate, as well as to generate, sign, and administer the validity of certificates. While this can be more beneficial than verifying the identities via a web of trust, the 2013 mass surveillance disclosures made it more widely known that certificate authorities are a weak point from a security standpoint, allowing man-in-the-middle attacks.

In its popular deployment on the internet, HTTPS provides authentication of the website and associated web server that one is communicating with, which protects against man-in-the-middle attacks. Additionally, it provides bidirectional encryption of communications between a client and server, which protects against eavesdropping and tampering with and/or forging the contents of the communication. In practice, this provides a reasonable guarantee that one is communicating with precisely the website that one intended to communicate with (as opposed to an imposter), as well as ensuring that the contents of communications between the user and site cannot be read or forged by any third party.

Historically, HTTPS connections were primarily used for payment transactions on the World Wide Web, e-mail and for sensitive transactions in corporate information systems. In the late 2000s and early 2010s, HTTPS began to see widespread use for protecting page authenticity on all types of websites, securing accounts and keeping user communications, identity and web browsing private.

A site must be completely hosted over HTTPS, without having some of its contents loaded over HTTP, or the user will be vulnerable to some attacks and surveillance. For example, having scripts etc. loaded insecurely on an HTTPS page makes the user vulnerable to attacks. Also having only a certain page that contains sensitive information (such as a log-in page) of a website loaded over HTTPS, while having the rest of the website loaded over plain HTTP, will expose the user to attacks. On a site that has sensitive information somewhere on it, every time that site is accessed with HTTP instead of HTTPS, the user and the session will get exposed. Similarly, cookies on a site served through HTTPS have to have the secure attribute enabled.

**Overview**

HTTPS is a URI scheme which has identical syntax to the standard HTTP scheme, aside from its scheme token. However, HTTPS signals the browser to use an added encryption layer of SSL/TLS to protect the traffic. SSL is especially suited for HTTP since it can provide some protection even if only one side of the communication is authenticated. This is the case with HTTP transactions over the Internet, where typically only the server is authenticated (by the client examining the server's certificate).

HTTPS creates a secure channel over an insecure network. This ensures reasonable protection from eavesdroppers and man-in-the-middle attacks, provided that adequate cipher suites are used and that the server certificate is verified and trusted.

Because HTTPS piggybacks HTTP entirely on top of TLS, the entirety of the underlying HTTP protocol can be encrypted. This includes the request URL (which particular web page was requested), query parameters, headers, and cookies (which often contain identity information about the user). However, because host (website) addresses and port numbers are necessarily part of the underlying TCP/IP protocols, HTTPS cannot protect their disclosure. In practice this means that even on a correctly configured web server, eavesdroppers can infer the IP address and port number of the web server (sometimes even the domain name e.g. www.example.org, but not the rest of the URL) that one is communicating with as well as the amount (data transferred) and duration (length of session) of the communication, though not the content of the communication.[citation needed]

Web browsers know how to trust HTTPS websites based on certificate authorities that come pre-installed in their software. Certificate authorities (e.g. VeriSign/Microsoft/etc.) are in this way being trusted by web browser creators to provide valid certificates. Therefore, a user should trust an HTTPS connection to a website if and only if all of the following are true:

The user trusts that the browser software correctly implements HTTPS with correctly pre-installed certificate authorities.

The user trusts the certificate authority to vouch only for legitimate websites.

The website provides a valid certificate, which means it was signed by a trusted authority.

The certificate correctly identifies the website (e.g., when the browser visits "https://example.com", the received certificate is properly for "Example Inc." and not some other entity).

Either the intervening hops on the Internet are trustworthy, or the user trusts that the protocol's encryption layer (TLS/SSL) is sufficiently secure against eavesdroppers.

HTTPS is especially important over unencrypted networks (such as WiFi), as anyone on the same local network can "packet sniff" and discover sensitive information. Additionally, many free to use and even paid for WLAN networks do packet injection for serving their own ads on webpages or just for pranks, however this can be exploited maliciously, e.g., by injecting malware and spying on users.

Another example where HTTPS is important is connections over Tor (anonymity network), as malicious Tor nodes can damage or alter the contents passing through them in an insecure fashion and inject malware into the connection. This is one reason why the Electronic Frontier Foundation and the Tor project started the development of HTTPS Everywhere, which is included in the Tor Browser Bundle.

Deploying HTTPS also allows the use of SPDY, which is designed to reduce page load times and latency.

It is recommended to use HTTP Strict Transport Security with HTTPS to protect users from man-in-the-middle attacks.

HTTPS should not be confused with the little-used Secure HTTP (S-HTTP) specified in RFC 2660

**Technical**

**Difference from HTTP:**

HTTPS URLs begin with "https://" and use port 443 by default, whereas HTTP URLs begin with "http://" and use port 80 by default.

HTTP is insecure and is subject to man-in-the-middle and eavesdropping attacks, which can let attackers gain access to website accounts and sensitive information. HTTPS is designed to withstand such attacks and is considered secure against such attacks (with the exception of older deprecated versions of SSL).

**Network layers:**

HTTP operates at the highest layer of the TCP/IP model, the Application layer; as does the SSL security protocol (operating as a lower sublayer of the same layer), which encrypts an HTTP message prior to transmission and decrypts a message upon arrival. Strictly speaking, HTTPS is not a separate protocol, but refers to use of ordinary HTTP over an encrypted SSL/TLS connection.

Everything in the HTTPS message is encrypted, including the headers, and the request/response load. With the exception of the possible CCA cryptographic attack described in the limitations section below, the attacker can only know the fact that a connection is taking place between the two parties, already known to him, the domain name and IP addresses.

**Server setup**

To prepare a web server to accept HTTPS connections, the administrator must create a public key certificate for the web server. This certificate must be signed by a trusted certificate authority for the web browser to accept it without warning. The authority certifies that the certificate holder is the operator of the web server that presents it. Web browsers are generally distributed with a list of signing certificates of major certificate authorities so that they can verify certificates signed by them.

**Acquiring certificates**

Authoritatively signed certificates may be free or cost between 8 USD and 70 USD[17] per year (in 2012–2014). However, in the case of free certificate authorities such as CACert, popular browsers (e.g. Firefox, Chrome, Internet Explorer) may not include the trusted root certificates, which may cause untrusted warning messages to be displayed to end users.

Organizations may also run their own certificate authority, particularly if they are responsible for setting up browsers to access their own sites (for example, sites on a company intranet, or major universities). They can easily add copies of their own signing certificate to the trusted certificates distributed with the browser.

There also exists a peer-to-peer certificate authority, CACert.

**Use as access control**

The system can also be used for client authentication in order to limit access to a web server to authorized users. To do this, the site administrator typically creates a certificate for each user, a certificate that is loaded into his/her browser. Normally, that contains the name and e-mail address of the authorized user and is automatically checked by the server on each reconnect to verify the user's identity, potentially without even entering a password.

**In case of compromised secret (private) key**

An important property in this context is perfect forward secrecy (PFS). Possessing one of the long term asymmetric secret keys used to establish an HTTPS session should not make it easier to derive the short term session key to then decrypt the conversation, even at a later time. Diffie–Hellman key exchange (DHE) and Elliptic curve Diffie–Hellman key exchange (ECDHE) are in 2013 the only ones known to have that property. Only 30% of Firefox, Opera, and Chromium Browser sessions use it, and nearly 0% of Apple's Safari and Microsoft Internet Explorer sessions. From the larger internet providers only Google supports PFS since 2011.

A certificate may be revoked before it expires, for example because the secrecy of the private key has been compromised. Newer versions of popular browsers such as Google Chrome, Firefox, Opera, and Internet Explorer on Windows Vista implement the Online Certificate Status Protocol (OCSP) to verify that this is not the case. The browser sends the certificate's serial number to the certificate authority or its delegate via OCSP and the authority responds, telling the browser whether or not the certificate is still valid.

**Limitations**

SSL comes in two options, simple and mutual.

The mutual version is more secure, but requires the user to install a personal certificate in their browser in order to authenticate themselves.[citation needed]

Whatever strategy is used (simple or mutual), the level of protection strongly depends on the correctness of the implementation of the web browser and the server software and the actual cryptographic algorithms supported.

SSL does not prevent the entire site from being indexed using a web crawler, and in some cases the URI of the encrypted resource can be inferred by knowing only the intercepted request/response size.This allows an attacker to have access to the plaintext (the publicly available static content), and the encrypted text (the encrypted version of the static content), permitting a cryptographic attack.

Because SSL operates below HTTP and has no knowledge of higher-level protocols, SSL servers can only strictly present one certificate for a particular IP/port combination. This means that, in most cases, it is not feasible to use name-based virtual hosting with HTTPS. A solution called Server Name Indication (SNI) exists, which sends the hostname to the server before encrypting the connection, although many older browsers do not support this extension. Support for SNI is available since Firefox 2, Opera 8, Safari 2.1, Google Chrome 6, and Internet Explorer 7 on Windows Vista.

From an architectural point of view:

An SSL/TLS connection is managed by the first front machine that initiates the SSL connection. If, for any reasons (routing, traffic optimization, etc.), this front machine is not the application server and it has to decipher data, solutions have to be found to propagate user authentication information or certificate to the application server, which needs to know who is going to be connected.

For SSL with mutual authentication, the SSL/TLS session is managed by the first server that initiates the connection. In situations where encryption has to be propagated along chained servers, session timeOut management becomes extremely tricky to implement.

With mutual SSL/TLS, security is maximal, but on the client-side, there is no way to properly end the SSL connection and disconnect the user except by waiting for the SSL server session to expire or closing all related client applications.

For performance reasons, static content that is not specific to the user or transaction, and thus not private, is usually delivered through a non-crypted front server or separate server instance with no SSL. As a consequence, this content is usually not protected. Many browsers warn the user when a page has mixed encrypted and non-encrypted resources.

A sophisticated type of man-in-the-middle attack called SSL stripping was presented at the Blackhat Conference 2009. This type of attack defeats the security provided by HTTPS by changing the https: link into an http: link, taking advantage of the fact that few Internet users actually type "https" into their browser interface: they get to a secure site by clicking on a link, and thus are fooled into thinking that they are using HTTPS when in fact they are using HTTP. The attacker then communicates in clear with the client. This prompted the development of a countermeasure in HTTP called HTTP Strict Transport Security.

In May 2010, a research paper by researchers from Microsoft Research and Indiana University discovered that detailed sensitive user data can be inferred from side channels such as packet sizes. More specifically, the researchers found that an eavesdropper can infer the illnesses/medications/surgeries of the user, his/her family income and investment secrets, despite HTTPS protection in several high-profile, top-of-the-line web applications in healthcare, taxation, investment and web search.

| S.NO | RGPV QUESTION | YEAR | MARKS |
|---|---|---|---|
| Q.1 | What is HTTP? Explain the utility and the various methods used by HTTP. | June.2014 | 7 |

**REFERENCE**

| BOOK | AUTHOR | PRIORITY |
|---|---|---|
| Web Technologies-TCP/IP Architecture, and Java Programming | Achyut S. Godbole and Atul Kahate | 1 |
| Web Technologies- A computer science perspective | Jeffrey C. Jackson | 2 |